Collaborative Data Processing in Wireless Sensor Networks

A DISSERTATION
SUBMITTED TO THE FACULTY OF THE GRADUATE SCHOOL
OF THE UNIVERSITY OF MINNESOTA
BY

Qingquan Zhang

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY

Professor Gerald E. Sobelman and Tian He, Advisor

December 2008

Acknowledgments

I would like to announce my acknowledgements to the people who have been supporting and collaborating me on my Ph.D. work. Without them, I will not be able to accomplish this dissertation. I am very grateful to be part of two excellent research groups where I have had a great memory, not only the knowledge I have learned but also the friendship I have built up with so many excellent people. I would really appreciate people by taking this opportunity:

First and foremost, I would like to thank my family: my parents Zugen and Binfang, my sister Xiu, and my grandma. Their endless love and unconditioned support have been my source of inspiration, power and dedication. Every time, they gave whatever they have to encourage me when I felt depressed. Without them, I could not imagine how I could ever be able to overcome all these difficulties. I still remember, my dad told me that the only wish he has in his life: to witness my growing and success in attaining the Doctor degree. Now I can say, "Dad, mission accomplished!".

I would really like to thank my advisors, Dr. Gerald E. Sobelman and Dr. Tian He, who brought me into the VLSI system design and the Wireless Sensor Network world. They have guided me through my research with their intelligence, knowledge and patience. Otherwise, I would probably never realize the beauty of research. Prof. Sobelman and He also supported me in every aspects of my life including my study, my courses and my career future. Without them, my dream of being a Ph.D. would never come true.

I am grateful to my committee members, Dr. Kiarash Bazargan and Dr.Vladimir Cherkassky. They have been giving me great help onto my dissertation. I personally appreciate their suggestions and ideas on my research projects. All my committee members were never at a loss of ideas. They have been continuously giving me advices on how to improve my work.

Next, I would like to take this opportunity to thank my colleagues and friends. I would like to thank my previous and current group memebers: Dr.Jie Gu, Dr. Barry Chad, Dr. Wenchih Kan, Dr.Fakhrul Zaman Rokhani, Woojoon Lee, Sangmin Kim, Yu Gu, Ziguo Zhong, Aaron Welly, Thomas Hatch. I also thank Dr. Chao Cheng, Dr. Jie Chen, Xue

Bai and for their generous support. They have provided all of the help I need to finish the projects. It is a great pleasure to work with those smartest people in the world.

Lastly, I would like to thank Ms. Yu Li, Ms.Yongling Zhang and Mr. Zheng Fu and anyone that ever help me and support me, including people inside and outside University of Minnesota, Twin Cities.

Abstract

Wireless Sensor Networks (WSNs) have been used in many application domains, such as target tracking or environmental monitoring. Due to limitations of power supplies, power management and power efficient target tracking techniques have become more and more critical. In this dissertation, systematic approaches are proposed to address the above problems. In particular, efficient energy-aware architectural design aspects of a sensor network are developed, with the goal to reduce the control scheduling algorithm complexity and the power consumption of various components while maintaining the data quality and performance requirements. Research results on an efficient error-bounded sensing scheduling algorithm, a novel collaborative global error implied assisted scheduling algorithm($CIES$) and fast target localization for mobile wireless sensor network are presented.

Dynamic scheduling management in wireless sensor networks is one of the most challenging problems in long-lifetime monitoring applications. In this thesis, we propose and evaluate a novel data correlation-based stochastic scheduling algorithm, called *Cscan*. Our system architecture integrates an empirical data prediction model with a stochastic scheduler to adjust a sensor node's operational mode. We demonstrate that substantial energy savings can be achieved while assuring that the data quality meets specified system requirements. We have evaluated our model using a light intensity measurement experiment on a Micaz testbed, which indicates that our approach works well in an actual wireless sensor network environment. We have also investigated the system performance using Wisconsin-Minnesota historical soil temperature data. The simulation results demonstrate that the system error meets specified error tolerance limits and up to a 70 percent savings in energy can be achieved in comparison to fixed probability sensing schemes.

Building on the results obtained from $CScan$, we further propose and evaluate a collaborative error implication assisted scheduling algorithm, called *CIES*. This computation-distributive system integrates an implied-error based prediction model together with a stochastic scheduler to adjust neighboring sensors' operational modes during the occurrence of rare or unusual sensing events. We demonstrate that substantial energy savings can be achieved while also satisfying a global error constraint. We have conducted extensive

simulations to investigate the system performance by using realistic Wisconsin-Minnesota historical soil temperature data. The simulation results demonstrate that the system error meets the specified error tolerance and produces up to a 60 percent energy savings compared several fixed probability sensing references.

In order to manage data link quality, a distributed sensor network with mobility provides an ideal system platform for surveillance as well as search and rescue applications. We consider a system design consisting of a set of autonomous robots communicating with each other and with a base station to provide image and other sensor data. A robot-mounted sensor which detects interesting information will coordinate with other mobile robots in its vicinity to stream its data back to the base station in a robust and energy-efficient fashion. The system is partitioned into twin sub-networks in such a way that any transmitting sensor will pair itself with another nearby robot to cooperatively transmit its data in a multiple-input, multiple-output (MIMO) fashion. At the same time, other robots in the system will cooperatively position themselves in such a way that the overall link quality is maximized and the total transmission energy in minimized. We efficiently simulate the system's behavior using the Transaction Level Modeling (TLM) capability of SystemC. Our results demonstrate the efficiency of our simulation approach and provide insights into operation of the network.

Finally, a fast target acquisition algorithm without the assistance of a map, call GraDrive, is introduced for search and rescue applications. We evaluate a novel gradient-driven method, which integrates per-node prediction with global collaborative prediction to estimate the position of a stationary target and to direct mobile nodes towards the target along the shortest path. We demonstrate that a high accuracy in localization can be achieved much faster than with random walk models, without any assistance from stationary sensor networks. We evaluate our model through a light-intensity matching experiment using MicaZ motes, which indicates that our model works well in a wireless sensor network environment. Through simulation, we demonstrate almost a 40% reduction in the target acquisition time, compared to a random walk model, while obtaining a small error in the estimate of the target position.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

- Chapter 1 introduces the motivation and goals pursued in this thesis.
- Chapter 2 presents a novel error-bounded scheduling algorithm for wireless sensor networks.
- In Chapter 3 we develop a collaborative mechanism for sensors to guarantee error performance in the presence of rare events.
- Chapter 4 introduces the construction of a MIMO-like data routing scheme used to improve data link quality.
- Chapter 5 presents a strategy for mobile sensors to cooperate in target acquisition.
- Chapter 6 concludes the thesis with suggestions for future work.

Integrated low-power sensing devices will permit remote object monitoring and tracking in many different contexts: in the field (vehicles, equipment, personnel) [1–5], the office building (projectors, furniture, books, people) [6], the hospital ward (syringes, bandages, IVs) and the factory floor (motors, small robotic devices). Associating these sensors together

and empowering them with the ability to coordinate amongst themselves on a larger sensing task will revolutionize information gathering and processing in many situations. Sensor nodes, the key component of a wireless sensor network, have been manufactured with a relatively low cost for many large-scale applications. Scalable, dynamically adjusting and robust sensor colonies can be deployed in inhospitable or even deadly physical environments such as inaccessible geographic regions or toxic areas which would pose great difficulties for human beings. However, for every large-scale wireless sensor network application, there are many research challenges to be addressed before they can be practically utilized.

To illustrate more clearly the goals of this dissertation, the following scenario is provided so that the challenges in designing these sensor networks can be classified into different categories: Thousands of wireless sensors are randomly deployed, e.g. spread by vehicles, over a large area. The sensors will have to collaborate together to form a communication network, execute an initialization phase of operation, adapt their sensing scheduling control to meet the specified data quality requirement and execute the sensing and monitoring tasks. Several aspects of this scenario present system design challenges different from those posed by traditional computer networks. Wireless sensor networks can encounter battery constraint issues or operation management difficulties in hostile environments, so individual sensor failure will be a common occurrence. In addition, the wireless sensors may frequently adjust their positions, configuration, power availability and even their required tasks.

All of the challenging issues described above will have to be addressed in order to satisfy the application requirements. However, these requirements will affect many other aspects of network design: routing and localization mechanisms, target tracking services, application architectures, data quality management, etc. Successful handling of these problems requires not only the careful design of the individual sensors but also collaborative data sharing and

data processing among the sensor nodes.

The goal of this dissertation is to focus on the principles of collaboration to solve the data management issues frequently faced in sensor networks. To achieve this goal, optimizations are considered at both the algorithmic and the network architectural levels. The former requires advanced investigations of algorithms and performance trade-off analysis, while the latter requires extensive architectural considerations with respect to feasibility of network communication capability and power consumption.

1.1 Summary of Motivations and Contributions

1.1.1 Error-bounded Dynamic Sensing Scheduling Algorithm

In this thesis, we propose a systematic dynamic sensing scheduling algorithm, called *Cscan*, specifically for long-lifetime applications such as military surveillance or habitat monitoring. The key idea of our framework is to activate a sensor only when there is a high probability that the model's prediction would exceed a specified error tolerance. Our approach builds on the observation that data sensed and collected by sensor networks over time may exhibit similar data patterns which may be exploited. We construct a data prediction model, i.e. an empirical model which captures the prominent features of the data collected over time. In addition, we use of an error-sensitive, stochastic, scheduling algorithm which allows sensor nodes to remain predominantly inactive while maintaining high data accuracy.

Our contributions in this area can be summarized as follows:

- We present a new energy-efficient scheduling algorithm that includes an accurate and hardware-friendly prediction model to capture recent data trends.

- We introduce the concept of error implication, which exploits data correlations among

multiple sensing cycles over a given time period.

- We provide an extensive experimental study of our framework using real data sets from different domains and compare our results against the most commonly accepted data aggregation approach. These experiments demonstrate that our algorithm can save up to 70 percent of the energy while still meeting a specified error constraint.

1.1.2 Global Implied Error Assisted Sensing Scheduling for Disruption Detection in Wireless Sensor Networks

We investigate a new Collaborative Implied Error Sensing scheduling algorithm, called *CIES*, focusing on applications such as military surveillance or habitat monitoring that require rare event detection capability. An efficient, empirically-based prediction model is utilized to schedule sensor activity when there is a shift in the environmental situation. The key idea of our $CIES$ framework is to transmit a wake-up message to its neighbors if a sensor detects an abnormal event.

Our contributions can be summarized as follows:

- We introduce the concept of collaborative error implication which exploits data correlations among multiple sensors over a sample time period. The main idea of this collaboration process is to construct and evolve over time a series of weight tables among the sensor nodes. These data structures represent sensing relationships among nodes, which can be used to improve the overall performance of the system.

- We provide both extensive simulation and an experimental study of our framework using real data sets from different domains and investigate the impact of several key parameters on the error rates.

1.1.3 MIMO-based Data Routing Architecture in Wireless Sensor Networks

In recent years, multi-input multi-output (MIMO) systems (i.e., systems having multiple transmit and receive antennas) have been widely adopted to enhance the performance of communication systems. Multiple antennas can be used to combat channel fading with space diversity and/or enable significant increases in the capacity of the wireless data link. While MIMO techniques are readily applicable to fixed base stations, the use of multiple antennas may not always be feasible for use in teams of small, mobile robots due to their size, power and complexity constraints.

We propose and study an alternative, namely to let groups of robotic sensors in geographic proximity of one another form cooperative, distributed antenna arrays. The main contributions of our approach are as follows:

- To allow single antennas on a pair of nearby information-bearing robots to cooperate with each other by pooling and transmitting their sensor data in a MIMO fashion.
- To enable a second form of cooperation in the form of relayed communications. Specifically, the non-information-bearing nodes can move to locations which relay information from the source nodes to the destination nodes in the most effective manner, i.e. by minimizing power requirements and/or by combating multipath fading effects.

1.1.4 Gradient-Driven Target Acquisition in Mobile Wireless Sensor Networks

The problem we address is to navigate a team of mobile sensor nodes toward a set of stationary targets rapidly and accurately while consuming the least amount of energy. The novelty of our approach is the seamless integration of a per-node prediction model with a

global prediction model. The per-node prediction model guarantees that a mobile node can acquire the position of a target alone, while the global prediction significantly reduces the navigation overhead and time, if collaboration among the nodes is available. Specifically, the main contributions of our prediction model are:

- Our model provides more a meaningful description of individual sensor readings in terms of their accuracy and confidence.
- Our model works with a single mobile sensor node as well as a swarm of mobile sensor nodes. In the latter case, the sensor nodes have the ability to share local information in order to draw a global picture, which helps each sensor node to acquire the target along a shorter path.
- The prediction algorithm enables faster yet accurate target position acquisition. Mobile sensor nodes are only required to reach the target when the model prediction is not accurate enough to satisfy the specified requirement with an acceptable confidence. This allows a significant reduction in navigation energy.

Chapter 2

Error Bounded Sensing Scheduling Algorithm

Dynamic scheduling management in wireless sensor networks is one of the most challenging problems in long lifetime monitoring applications. In this paper, we propose and evaluate a novel data correlation-based stochastic scheduling algorithm, called *Cscan*. Our system architecture integrates an empirical data prediction model with a stochastic scheduler to adjust a sensor node's operational mode. We demonstrate that substantial energy savings can be achieved while assuring that the data quality meets specified system requirements. We have evaluated our model using a light intensity measurement experiment on a Micaz testbed, which indicates that our approach works well in an actual wireless sensor network environment. We have also investigated the system performance using Wisconsin-Minnesota historical soil temperature data. The simulation results demonstrate that the system error meets specified error tolerance limits and up to a 70 percent savings in energy can be achieved in comparison to fixed probability sensing schemes.

2.1 Introduction

Wireless Sensor Networks (WSNs) have been used in many application domains [1–4]. Due to the limited power supply and difficulties in harvesting ambient energy, low power energy management is a critical research issue. Energy consumption for the sensing operation dominates the lifetime of a sensor network. Therefore, it is important to design protocols which minimize the amount of sensing required by the sensor nodes. In the past few years, many solutions have been proposed for energy conservation by applying different power switching strategies (e.g. [7]) in which hardware components such as CPU and memory can operate with different power modes. Other semantic-based efforts, such as TAG [8], focus on reducing the sensing and communication load. Even though those methods show some interesting results, there is a need for improvement in several directions. Moreover, most real-time power control protocols have no robust error control guarantee mechanism.

In this chapter, we propose a systematic dynamic sensing scheduling algorithm, called *Cscan*, specifically for long lifetime applications such as military surveillance or habitat monitoring. The key idea of our framework is to activate a sensor during cycles in which there is a high probability that the model's prediction would exceed a specified error tolerance. Our approach builds on the observation that data sensed and collected by sensor networks over time may exhibit similar data patterns and the data disseminated over time could be well correlated. The key techniques used in our approach are: 1) the construction of a data prediction model, i.e. an empirical model which captures the prominent features of the data collected over time, and 2) an error-sensitive stochastic scheduling algorithm. This methodology allows sensor nodes to remain predominantly inactive, while achieving a high data integrity. As we will present in this paper, our contributions can be summarized as follows:

- We present a new energy-efficient scheduling algorithm that includes a very accurate but hardware-friendly prediction model to capture recent data trends.

- We introduce the concept of error implication, which exploits data correlations among multiple sensing cycles over a given time period.

- We provide an extensive experimental study of our framework using real data sets from different domains and compare our results against the most commonly accepted data aggregation approach. We also implement our algorithm into the sensor network we built for a light intensity monitoring application. Our experiments demonstrate that our algorithm can save up to 70 percent of the energy while still meeting the error rate requirement.

2.2 Overview and Objectives

The strategies exploited in our *Cscan* framework are specifically developed for long-term environmental monitoring applications in which energy conservation and data accuracy are of most interest. The system should try to avoid any unnecessary sensing and data acquisition while assuring acceptable data quality, as defined by the application. The system performance is quantified by defining three criteria: the miss ratio, which denotes the fraction of scheduling cycles that the system fails to present acceptable prediction data, the energy consumption and the data sample error rate.

To successfully achieve our energy and error control objectives, a data management scheme is investigated and integrated into the system. The architectural framework is shown in Figure 2.1. Those functional blocks will support the following key features:

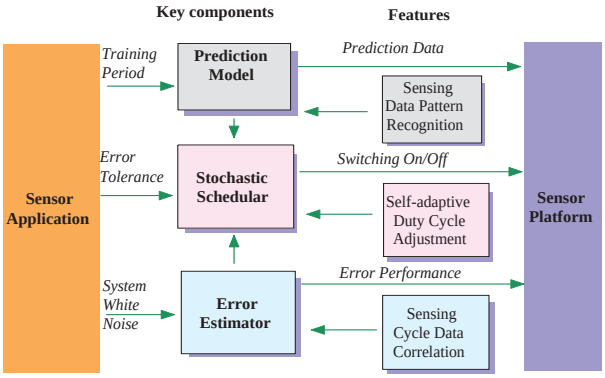

Figure 2.1: The architecture of *Cscan*.

2.2.1 Prediction model construction

We seek to identify correlated sensor data patterns in a sensing period in order to predict sensing data over time. The sensors' sampling data are fed into the model constructor during the initialization training stage and an empirical prediction model is created. After that, model constructors keep updating the model whenever new sensing data becomes available.

2.2.2 Duty-cycle optimization

This is an approach to manage the power consumption and the prediction error rate in a given cycle.

2.2.3 Error estimator

The error estimator will serve to ensure that the operation of a sensor is such that the data quality requirement is not violated. We must create a balance between energy savings and the rate of prediction errors.

2.3 Data Prediction Algorithm

A sensor can lower its operating duty cycle, meaning it can switch into a sleep state to conserve energy. This operation is based on the fact that the sensor's readings may form a recognizable pattern during certain periods, especially in the case of environmental monitoring applications. Those patterns can be well approximated and used for predicting future readings if the specific application is well understood. The system will start building the prediction model in the initialization phase. Then, we separate the sensor's operation into a data resampling phase and a prediction phase. In the data resampling phase, we use the latest sampling data to update the model built during the training cycle. In the prediction phase, the node will switch off to conserve energy and the predicted sensing results are generated by the predictor which has been updated in the resampling phase.

Empirical Model Construction

An empirical model is used to find strong correlations in the data and to arrange them in a certain way so that future data can be extracted from the empirical or historical data. Depending on the duration of the system and the data accuracy requirements of an application, the empirical models can be constructed in different ways. Here we introduce an hourly-based empirical model, as shown in Figure 2.2 During a data training cycle, initial sensing differences between two adjacent hours are calculated and updated throughout the training cycle. A weighted moving average method is used to smooth the data. For example, if the sensed temperature data at 1 AM and 2 AM are 20 degrees F and 22 degrees F respectively, then the difference between 1 AM and 2 AM is 2 degrees F. At the end of the training cycle, a model is constructed such that the sensing data difference between any two adjacent hour times can be estimated at the sensor node.

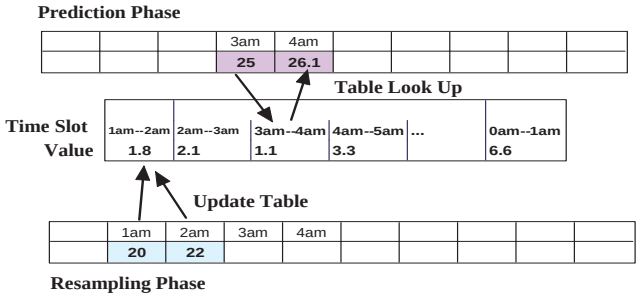

Figure 2.2: Construction of the empirical prediction model.

Prediction Model Update

Once the sensor is in the resampling phase, the system will not only get precise readings but can also refresh the empirical model parameters. The system compares the prediction values produced by the empirical model with the real sensing data. If the difference is below the specified error tolerance level, the system is regarded as good ("hit") and the prediction model can be used. This can be expressed in the following equation:

$$ABS(\frac{V_p - V_r}{V_r}) \leq e_t \tag{2.1}$$

V_p is the value output from the estimator, V_r is the true sensed data and e_t is the error tolerance level that can be accepted, as specified by the user.

A system corrective action will be taken to update the empirical model by refreshing the original model with the latest results for $V_r(k)$ and $V_r(k-1)$.

Compared to a regression model, the advantage of using an empirical model in this application domain is that it simplifies the processing requirements while providing a reliable reference for prediction. As a result, the hardware cost can be minimized. Moreover, data resampling helps to update the predictor's model parameters when the sensor nodes are in a dormant state.

2.4 Scheduling Algorithm

In this section, we present our scheduling algorithm that includes the underlying data prediction model and the data quality requirements to control the sensing/resampling of the sensors. We seek to minimize the sensor energy consumption according to different system operation modes while satisfying the data quality constraint.

2.4.1 Our problem formulation

In order to conserve their limited power supply, the sensors do not continuously sense data but rather operate only during certain cycles as long as acceptable data quality can be met. The scheduling can be adjusted based on the algorithm that we will elaborate later. We assume that the baseline sensor operation sequence consists of N data cycles, which include k cycles used for training. In each cycle i, the probability that the sensor will be active is defined as p_i. We further assume the average energy consumption for sensing (the energy cost of a node to sense, process and communicate) is E_a, the defined prediction error tolerance is e_t and the potential error at each cycle due to inactive sensor status is e_i. Therefore, the goal of our design is to minimize the energy consumption during each baseline period:

$$E = k \cdot E_a \cdot t_u + E_a \cdot t_u \cdot \sum_{i=1}^{N-k} p_i \tag{2.2}$$

under the constraint that

$$\frac{\sum_{i=1}^{N} (1 - p_i) \cdot e_i}{N} = \frac{\sum_{i=k+1}^{N} (1 - p_i) \cdot e_i}{N} \leq e_t \tag{2.3}$$

where t_u is the unit cycle length and e_t is the error tolerance set by the design requirements. The constraint will enforce that the potential statistical error caused by the prediction will

be less than the error tolerance. The range of possible values of p_i will be bounded to satisfy the constraint equation.

2.4.2 Adaptive scheduling algorithm

The minimization of energy consumption deals with several key issues, e.g. the length of the training cycle and the prediction model used. The goal of the scheduling algorithm is to find the appropriate p_i for a given error range e_i obtained from past data values. To solve for p_i at a specific e_i requires a joint distribution of a process for e_i at a specific time instance or period. This would require a heavy computational capability and storage burden on the limited resources of the sensor node. Obtaining a solution for p_i will be extremely difficult to calculate during transitions. Instead, we introduce a simpler method for computation that allows the sensor to choose the value within a range. We first determine the boundary for p_i, and the scheduling algorithm will choose one value within the boundary according to a node's operational status. It should be clear that the higher the value of p_i, the larger the expected energy consumption. The lower the value of p_i, the higher the chance that the error due to prediction will be greater than the tolerance. Therefore, analysis of the boundary of p_i will be investigated to optimize this trade-off.

2.4.3 Determining the Sensing Probability Boundary

We use a bottom-up approach to set a boundary the for sensing probability. That is, we will not violate the constraint equation during each cycle instance so that the sum of all cycle error products $(1 - p_i) \cdot e_i$ will not violate the constraint. As noted, this decision sets a stricter requirement than the constraint equation over all sampling instances. Therefore, our probability constraint problem can be simplified into choosing the p_i at each scheduling

cycle to satisfy the constraint on $(1 - p_i) \cdot e_i$, which can be solved as

$$p_i^{lb} = \begin{cases} 0 & 0 \leq e_i \leq e_t \\ \\ 1 - \frac{e_t}{e_i} & e_t \leq e_i \leq 1 \end{cases} \tag{2.4}$$

The p_i^{lb} is the lower bound of p_i which guarantees the satisfaction of the system data quality requirement at each sensing cycle instance. Only values higher than this will assure that the constraint requirement won't be violated under any circumstances. We should also be careful in the selection of p_i, as higher p_i implies more energy consumption by the sensor node.

2.4.4 The Selection of Sensing Probability

In this section, we focus on choosing the appropriate value of p_i which requires us to determine the effective error estimation e_i at cycle i. To accurately evaluate the e_i, we included two categories of errors that we believe make up the major contribution to the possible errors in prediction. The first category is the system intrinsic error, τ, that takes into account all the system white noise and environmental instability within the system. The other category is the implied error, e_{im}, which is a function of the data correlations. Our technique relies on adaptive sensing to adjust both of them. The sensors keep a record of past sensing data, comparing the authentic recorded data with the outputs from the prediction models constructed. The error rate will then be fed back to the sensor operation platform where processing of two error categories will take place. A probability estimation algorithm (Algorithm 1) is called during the initialization and update procedures of the sensing operation to select the probability value. The algorithm takes the error tolerance

e_t, initialized intrinsic error τ_i and the implied error e_{im} as inputs. The input variables will be used to choose the corresponding probability from the available rate ranges as described in Algorithm 1.

Algorithm 1 Probability Determination Algorithm

Require: e_t,τ_i,e_{im}

```
Determine the boundary of p_i from section 2.4.3
if e_i0 < e_im then
    e_j = e_im
else
    e_j = τ_i
end if
Achieve the value of p_i from the constraint equation
return p_i
```

In this algorithm, we choose the higher error estimation between the intrinsic error τ_i and implied error e_{im}. A high error rate indicates environmental instability or a poor prediction model outcome while a low value signals a potential to cut down the resampling rate for energy conservation purposes. However, since τ_i and e_{im} will change over time, a mechanism is necessary to estimate them in an adaptive manner.

Update intrinsic error τ_i

The intrinsic error τ_i represents the information about the prediction instability of data at cycle i. A high value indicates a greater chance that the prediction model will fail in estimating the real value. We update τ_i whenever the sensor node switches on at that cycle by using a moving average:

$$\tau_i = \alpha \cdot e_s + (1 - \alpha) \cdot \tau_i \tag{2.5}$$

where e_s is the error between the predicted value and the actual recorded data when the sensor switches on. In our experiments, we choose α to be 0.5. As we can see, if the prediction model outputs a lower error data value in comparison to the real value, the new

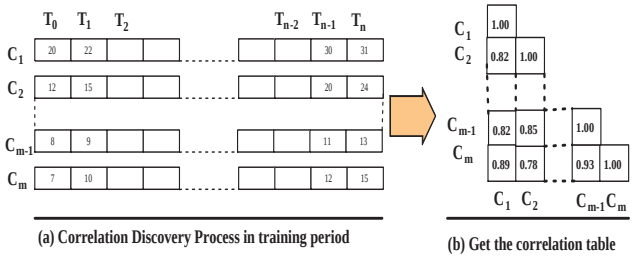

Figure 2.3: Construction of the table of correlation coefficients.

τ_i will become smaller.

Achieving the implied error e_{im}

The implied error e_{im} is obtained from the correlation coefficient between the current cycle and the latest cycle in which the sensor node switched on. It can be expected that if the two cycles have a strong correlation, the error in one cycle can be well estimated from the correlated cycle. Otherwise, two cycles having low correlation will have a high potential error mismatch. Therefore, the sensor platform must take this into account when determining whether the sensor node needs to switch on. Based on our observation, the implied error can be well estimated as:

$$e_{im} = \frac{A \cdot e_j}{C_{ij}} \tag{2.6}$$

where A is a constant, e_j is the latest measured error when the sensor node switched on at cycle j and C_{ij} is the correlation coefficient between the current cycle i and cycle j. The procedure for constructing the correlation coefficient table e_{im} is illustrated in Figure 2.3.

It should be noted that that for simplification purposes, we assume that correlation among cycles remain relatively stable throughout each operation.

Figure 2.4: Testbed and snapshot of experimental data events (inset shows the light pattern).

2.5 Evaluation

2.5.1 System Implementation

The architecture has been implemented on our newly constructed test-bed, shown in Figure 2.4, with more than 100 sensor nodes which provides a realistic controllable environment for design verification and performance evaluation. The design has been implemented on a Berkeley TinyOS/Micaz platform. Sensor nodes are placed randomly over the board, giving us a better reflection of the sensing algorithm. Both random and controlled scanning light patterns are created to emulate the light intensity change in environment and then projected onto the test-bed with three projectors switching on simultaneously. The sensed data is recorded and processed according to our sensing algorithm. The evaluation results (e.g. error rate, energy conservation vs. error tolerance) allow further analysis to optimize the overall system.

2.5.2 System Evaluation

In order to test the performance of the proposed $Cscan$ algorithm, especially for error control in terms of energy conservation, we have conducted a series of experiments to track the sensor status on our test-bed. Different light intensity patterns are projected onto the test-bed to emulate various environment conditions. The sensor nodes detect the light intensity and dynamically process those values using $Cscan$. A period of 1000 cycles (corresponding to 1000 sample points) was selected for each run. Each run was repeated multiple times with different parameter settings, such as the length of the training cycle. Figure 2.5 shows the resulting dynamic energy consumption. As seen in the figure, $Cscan$ does not conserve energy in the initialization period during which the prediction model and inter-cycle correlations are built. After that, however, the energy consumption is reduced, as desired. We can also observe in the figure that the energy conservation in certain cycles remains at a flat level, corresponding to those times in which the sensor node is in its prediction mode. The energy consumption as a function of error tolerance is shown in Figure 2.6. Three sets of results representing different experimental scenarios are presented. The first scenario is one in which a sensor randomly switches on/off with probability 50 percent. The second scenario is where the sensor has a 90 percent probability of being active in every cycle. In the third scenario, sensors operate according to the $Cscan$ scheduling algorithm. We can see that energy conservation reaches above 70 percent when the error tolerance e_t is relatively high. $Cscan$'s energy conservation is less than that in the random case when e_t is low, implying that the sensors have a higher chance of switching on if the environment is not stable. The error performance results are presented in Figure 2.7. These results also show that $Cscan$ can control the error rate according to system requirements and that the $Cscan$ algorithm can be practically and effectively implemented.

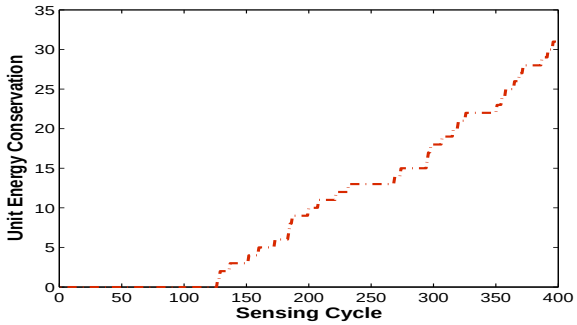

Figure 2.5: One sample of dynamic energy conservation in a sensor.

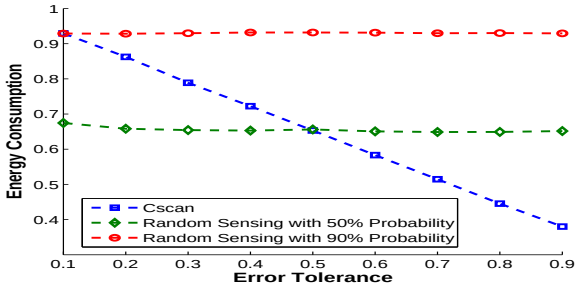

Figure 2.6: The measured energy consumption of $Cscan$ vs. other strategies.

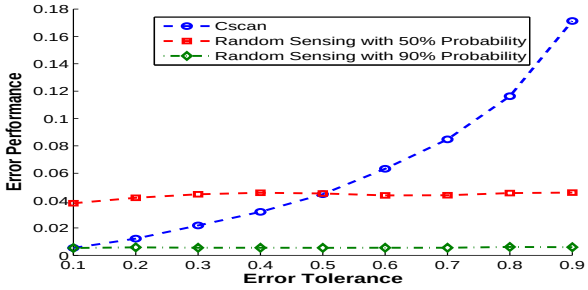

Figure 2.7: The measured sensor error performance of $Cscan$ vs. other strategies.

2.5.3 Emulation Setting

To evaluate the performance of the *Cscan* sensing strategies in a real application, a simulation program with historical soil temperature data was developed. The data was collected from the Wisconsin-Minnesota Cooperative Extension Agricultural Weather Page where soil temperature is monitored regularly. The soil temperature is sampled twice per hour, 24 hours per day. This full record of soil temperature data over the past 10 years allows us to extensively test the efficiency of our strategy. By doing so, we can reduce the randomness and investigate the impact of different configurations on the performance of energy conservation and error control.

2.5.4 Performance Analysis

In this section, we evaluate the error rates as a function of some key design parameters, including the effect of error tolerance and the length of the training period. A study of the effects of these parameters can provide insights into methods for improving system performance. We begin the evaluation by measuring the error rate of the *Cscan* system. Then, we compare the energy conservation for different parameter values. Finally, we study the miss ratio (defined as sensor prediction results which violate the error tolerance requirement) performance for our adaptive scheduling algorithm.

Impact of Error Tolerance

During this evaluation, the level of error tolerance varies from 10 percent to 90 percent while the length of the training period was kept constant at 12 percent of the total number of simulation cycles. The total number of cycles is approximately 9000, which corresponds to more than one year of data. This data set is large enough to significantly reduce unsys-

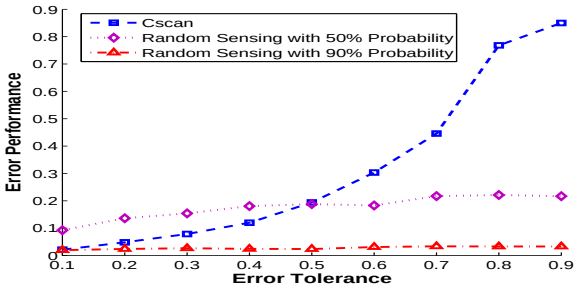

Figure 2.8: The influence of error tolerance on the average error.

tematic errors caused by limited sample size. Figure 2.8 shows the estimated error rate as a function of error tolerance under different scenarios. We measure the average prediction error of the estimator in scenarios 1 and 2, as described earlier. It turns out that the prediction error in scenario 1 is about 40 percent for most of the error tolerance levels, which means that little improvement in energy consumption is achieved in this case. Also, the error rate is low for scenario 2, as expected. As suggested from the simulation results, the prediction error for the $Cscan$ algorithm increases in proportion to the increase of error tolerance. Most importantly, the prediction error from $Cscan$ met the requirements in almost all cases. For example, $Cscan$'s data error rate is only 20 percent when the system error tolerance is 50 percent. This will likely be an acceptable error rate for many long-term monitoring applications. The results verify that the dynamic strategy in $Cscan$ can effectively meet the data quality requirements of an application. The energy consumption for different scenarios in Figure 2.9 are also provided to demonstrate the effectiveness of $Cscan$ as compared to other approaches.. It can be seen that $Cscan$ achieves a better error/energy margin when the error tolerance is between 20 and 50 percent. Intuitively, the higher the error tolerance, the more the energy consumption can be reduced. We also investigated the effectiveness of our prediction model through the measurement of the miss ratio, as shown in Figure 2.10. The prediction miss ratio for $Cscan$ increases as error tolerance increases.

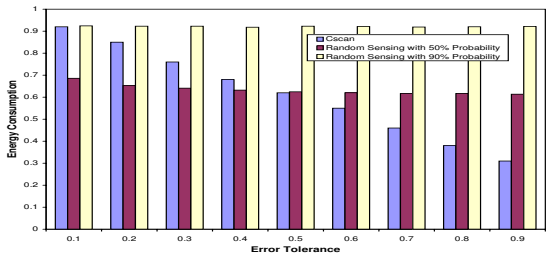

Figure 2.9: The influence of error tolerance on energy consumption.

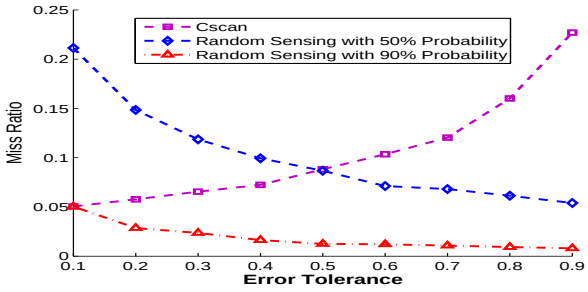

Figure 2.10: The influence of error tolerance on the prediction miss ratio.

This is not surprising because a high error tolerance implies that the sensor won't be able to anticipate an abrupt change in the environment. However, the highest miss ratio measured is only about 25 percent, which suggests that our prediction model provides a reasonably high prediction accuracy.

Impact of training period length

In this experiment, we evaluated the influence of different lengths of the training period on the error rate and energy conservation. When a sensor node begins sensing, an initialization period is required to build both the correlation table and the prediction model. The accuracy of the prediction model will depend on the sample size of the data fed to the model constructor. As we can see in Figure 2.11, the error rates decrease as the length of

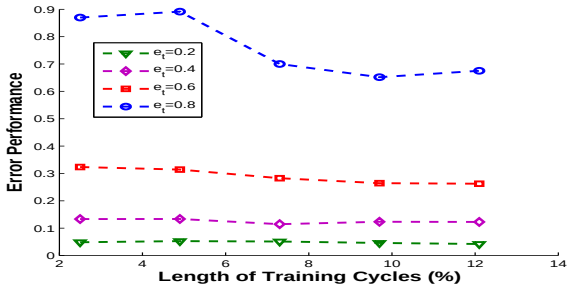

Figure 2.11: The influence of training period length on the prediction error rate.

training period becomes longer. This becomes more evident when the error tolerance e_t is larger. This can be explained by the fact that the scheduling algorithm has more flexibility to adjust the duty cycle as the error tolerance becomes larger. Notice that the energy consumption also increases with increases in the length of the training period. According to our experiments, the energy consumption increased from 27 percent to 38 percent as the length of training period was increased from 2.3 percent to 12 percent of the total number of simulation cycles. As a result, the system exhibits a trade-off between data accuracy and energy conservation.

2.6 Related Work

In recent years there has been increasing interest in studying approaches for energy-efficient operation of wireless sensor platforms. These studies include data aggregation techniques to reduce the communication overhead [7, 9, 10]. To more aggressively keep sensor nodes in a dormant state, data prediction has also been investigated. Both numerical approaches and empirical models have been implemented [11, 12]. Using a Dual Kalman Filter, Jain et al. [13] proposed a prediction model to minimize resource usage under a precision requirement. However, the prediction model that was used requires sophisticated computation that results in hardware complexity and increased power consumption at the cluster head.

In [12], empirical analysis results revealed the relationship between the configuration parameters and the quality of the search. In references [14, 15], data correlations with spatial coherence and routing efficiency were investigated. Research on dynamic sensing schedulings to balance accuracy and energy saving were also conducted [16–18]. In eSense [19], a stochastic sensing algorithm used probability bounds for the miss ratio constraint. However, their approach is not sensitive to the degree of data error. In contrast, our approach employs an empirical prediction model to predict sensing data that does not require complicated hardware. Furthermore, we also use data cycle correlations in error estimation to determine the sensing probability, which allows us to achieve significantly higher energy conservation for a given error tolerance.

2.7 Conclusions

In this chapter, we have presented a stochastic sensing algorithm to reduce energy consumption. Our approach does not require powerful computational ability at the sensor nodes to construct an accurate data prediction model. Observed correlations between different data cycles has been used to estimate the prediction error, thus allowing the scheduler to adjust its operation accordingly. The measurement and simulation results show that system prediction error remains within the specified error tolerance while saving up to 70 percent of the required energy. For our future work, we would like to evaluate the energy performance of individual sensor network components so that the algorithm can be further optimized.

Chapter 3

Collaborative Implied Error Assisted Sensing Scheduling Algorithm for Eruption Detection in Wireless Sensor Network

3.1 INTRODUCTION

Wireless Sensor Networks (WSNs) have been used in many applications [1–4,20–23]. Due to their limited power, low power energy management has been a critical research issue in long lifetime monitoring application [3,24,25]. As is well known, one of the major sources of power consumption is the energy cost of sensing activities [24]. Therefore, unnecessary sensing activities should be avoided to extend the lifetime of sensor networks. Dynamic sensing scheduling is an effective method to reduce sensing activities thereby improving network

energy-efficiency ([7, 8, 26, 27]). Rather than sensing, a prediction model is applied to provide data estimation when sensors are inactive. Methods in [19, 28–30] further introduce a self-adaptive scheduling mechanism to address the data accuracy issues. We acknowledge that these technologies do provide frameworks to manage the energy cost without comprising data accuracy. However, these design approaches are isolated in that the management of data accuracy can only be used for each individual sensor. Thus, these techniques do not fully exploit the potential of network collaboration for data accuracy management under rapidly changing conditions in the environment, and their ability to tradeoff data accuracy with energy reduction is thus inherently limited. Our work incorporates the advantages of both dynamic scheduling and sensors collaboration, seeking to optimize the tradeoff between energy and the accuracy of predictions.

In this chapter, we investigate a sensing scheduling algorithm, called *CIES*, focusing on monitoring applications such as habitat monitoring in which high sensitivity to rapid environmental change is desired. It can provide high data accuracy while minimizing the amount of energy used. Our $CIES$ framework explores the collaboration between neighboring sensors to reduce the probability of violating a data accuracy requirement in any given sampling cycle. Specifically, when an unanticipated event happens (e.g., a dramatic temperature change or environmental shift), each active sensor node will estimate its inactive neighbors' potential data prediction errors. This error estimation information will then be delivered to the neighbors. With the assistance of such error estimation information, those inactive neighbors will be able to respond to such an event. Being able to infer the neighbors' model prediction errors can significantly improve the networking performance in data accuracy.

The major challenge in our design is how to determine neighboring nodes' potential risk

of violating the data accuracy requirement while minimizing the energy consumption. We solve this problem by introducing a mapping mechanism of the prediction error for neighboring sensors. We demonstrate that our design achieves better control of data accuracy than baseline approaches and still retains the energy saving properties, especially in the situation where a dramatic environmental shift occurs. As we will present in this paper, our contributions can be summarized as follows:

- We introduce the concept of node-pair inferred error that exploits the data correlations among multiple sensing sensors during a sampling time period. The key issue of this collaboration process is to construct and evolve over time a series of weight tables and graphs among sensor nodes. These weight graphs represent the virtual sensing relationship among sensor nodes, which can be used to improve the performance of the collaboration operation.

- We provide both extensive simulation and an experimental study of our framework using real data sets from different domains and investigate the impact of key parameters on data accuracy. Our experiments demonstrate that the algorithm can provide energy savings as high as 60% to benchmarks while still meeting the data accuracy requirement.

The rest of this chapter is organized as follows. Section 3.2 describes the motivation behind our work from an application perspective. We present the overview of our design in Section 3.3. Section 3.4 describes the details of the error control mechanisms. The performance evaluation is presented in Section 3.6. Section 3.7 concludes the chapter.

3.2 Related Work

Scheduling control has been an effective method in wireless sensor platforms to improve energy efficiency [31]. It allows networked nodes to reduce their transmitting and sensing power during operation while preserving sensing quality. A common technique to reduce transmitting power is applying data aggregation approaches to reduce the communication overload as in [7, 9, 10, 32–35]. An alternative approach is to reduce the duty cycle (the active duration) of sensors by turning off sensing while using a prediction model to estimate the actual data at each sensor node [36, 37]. In general, a higher duty cycle leads to a better prediction accuracy but consumes more energy. On the contrary, a reduced duty cycle is preferred due to its lower energy consumption and reduced data traffic within the network. In traditional methods, the prediction models are constructed using numerical approach. In this approach, complex and dedicated models are used frequently, aiming to determine if the model is accurate enough to ensure high precision. For example, the Dual Kalman Filter [13], typically used in target tracking applications, can be used to precisely predict the actual locations of objects. In actual situations, however, this kind of approach cannot necessarily characterize the variable nature of the environment, thus leading to an inefficiency in data prediction. This is mainly due to insensitivity of prediction model to sharp change phenomenon in nature [38]. In [12], empirical analysis results have been used to reveal the relationship between the configuration parameters and the quality of the tracking application. It provides that the empirical model [16–18] can be effectively applied without sacrificing the data prediction accuracy. One useful perspective from an empirical modeling approach is that data correlation can be utilized for other purposes [14, 15], e.g. event detection. But their scheduling controls are based on random mechanisms, in which the switching operation is built on an output of a random machine

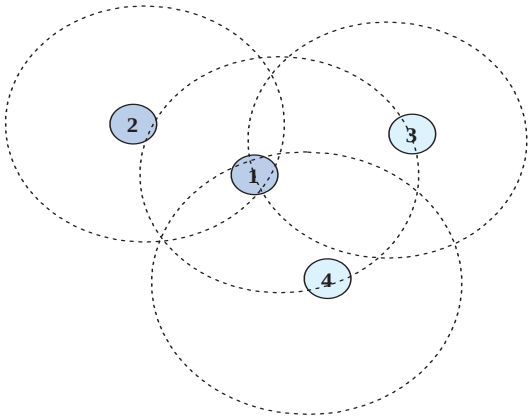

Figure 3.1: Initial operation graph

or a pre-defined routine. In eSense [19], a stochastic sensing algorithm which computes the sensor switching probability at each sampling cycle is introduced. It determines the lower and upper boundary of sensing probability to satisfy the miss ratio constraints, a metric to determine the percentages that the prediction model output will violate the data performance requirement. However in most cases, the error between the output of the prediction model and the actual observation is desired because it can be used to quantify the system performance. A combination of an empirical model together with a stochastic control approach provides the opportunity to accurately associate networking nodes for higher data accuracy and increased network capability, e.g. detection of the rapid environmental change. Motivated by this, we propose $CIES$. To the best of our knowledge, our proposed algorithm is the first to adapt strict error-bounded sensing control. Furthermore, we study the observed data correlation between nodes with error estimation to determine the sensing probability, which allows us to achieve a significant data quality improvement for a given energy cost.

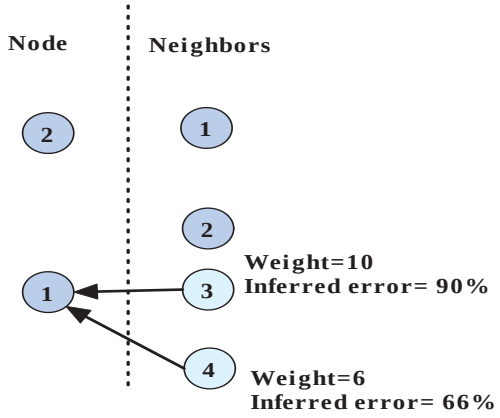

Figure 3.2: The computation of weighted inferred error

Node	Initial Status	New Status
1	Off	On
2	Off	Off
3	On	On
4	On	On

Figure 3.3: The initial and final operational modes

3.3 OVERVIEW AND OBJECTIVES

This section gives an overview of our Collaborative inferred Error System(*CIES*). We first present the assumptions of the work, and then describe the overall system design.

3.3.1 Assumptions

Our assumptions are as follows:

- Sensors are randomly and independently distributed in the field.
- The sensor network is homogeneous.
- Sensors will have limited radio range, denoted as R.
- Sensors will operate in a low-power listening mode periodically during every sampling period for message traffic tracking. There are several approaches e.g. [39, 40] to provide such kind of MAC controls.

Definition *Inferred Error.* Given a source node s and a destination node d in the neighborhood, the node-pair inferred error e_{sd} is defined as the potential error at destination node d from the point view of source node s. **Definition** *Neighbor.* For simplicity, neighbors are defined as nodes which can communicate with each other within a one-hop range. One example is shown in Figure 3.1; given sensor nodes 1 and 2 within one-hop communication range, node 1 will become node 2's neighbor and vice versa. By one-hop, we mean that the node, can communicate with its neighbor without routing the packets.

Definition *Node-pair Weight.* The weight is defined as the extent of a node-pair's data correlation and indicates how similar the sensing observation is between any two sensor nodes.

Definition *Error tolerance.* Error tolerance is defined as the maximum error rate that a prediction system can accept.

3.3.2 Motivation example

A simple example is used to illustrate the basic idea of $CIES$. Figure 3.1 shows an example with a node set $G = \{1, 2, 3, 4\}$.

1. After deployment, nodes 1, 2, 3 and 4 initialize their local error control operation processes and neighbor error-control procedures independently. Node 1 will have neighboring nodes 2, 3, and 4. And node 1 will become node 2's neighbor.

2. At one sampling time duration, nodes 3 and 4 are in full operation (in dark color as shown in Figure 3.1). Nodes 1 and 2 are scheduled to be in sleep mode, within which their radios are switching on periodically to monitor traffic.

3. Nodes 3 and 4 calculate their inferred errors on node 1 to be 90% and 66%, respectively as shown in Figure 3.2. These error estimation from node 3 and 4 will then be sent to node 1. Because both nodes 1 and 2 are in sleep mode, there is no message exchange between them.

4. After receiving the inferred error messages from nodes 3 and 4, node 1 evaluates the weighted average inferred error to be 80%. Then, it will determine whether the error tolerance is violated. If the error tolerance in this simple example is 30%, node 1 will be awakened immediately after the sampling time duration because the calculated weighted inferred error is larger than the threshold.

5. The operational modes of these nodes are indicated in Figure 3.3.

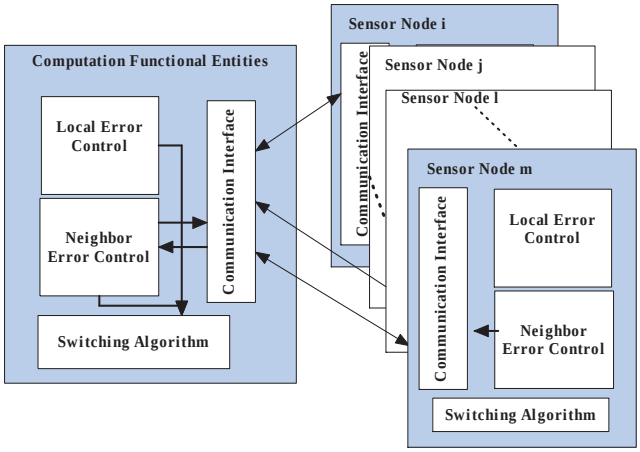

Figure 3.4: The nodes' collaboration process

From this simple example, we have the following observations. First, $CIES$ exhibits great potential for reducing error rates due to rapid environmental change. Compared to sole prediction as in eSense, $CIES$ improves the capability for sensor collaboration. Second, the prediction model and data observation correlation between nodes are critical, and distributed algorithms are required.

3.3.3 Sensor Node Error Control Architecture Design

To facilitate both the error information exchange and independent scheduling control, our node system introduces two parallel error control mechanisms: 1) local error control management; and 2) neighbor error control management. Both error control managements are implemented in each individual sensor node, as shown in Figure 3.4.

The local error control management module will enable the sensor to schedule its own error-bounded operating process without assistance from neighboring nodes, while the neighbor error control management module will handle all the error-related information exchange (e.g. inferred error) between neighboring nodes. The communication can be done through the generic communication interface. Both local error and neighbor error control

algorithms will cooperate to set the scheduling of sensors.

With the implementation of dual error control mechanisms, the sensor system can be optimized for the performance of error control and configured to detect rare events in monitoring applications as well.

3.3.4 Procedures on Sensors

The following section summarizes our collaborative error control processes as organized in Figure 3.5. In Figure 3.5 , first the sensors establish the neighborhood by using methods like flooding, as in [41, 42]. Second, the node-pair weight graphs will be constructed to represent the sensing relationship among sensors, as shown in Figure 3.5. Third, whenever a sensor detects a rare event, it will trigger the neighbor error control management procedure in order to disseminate this information to adjacent nodes. The awaking mechanism is shown in Figure 3.6. The neighbor error control procedure will estimate neighboring nodes' potential errors, sending out a wake-up message to the nodes that fall below the data accuracy requirement (e.g. the error threshold set by an application). The sensor node which receives such a message will respond accordingly, either remaining off or switching on and sensing.

3.3.5 Objective and Challenge

The main objective of this work is to develop a generic scheduling mechanism for collaborative sensors to achieve error-bounded scheduling control in monitoring applications. Users can choose either the sole local error control management or implement both mechanisms simultaneously for applications that demand rare event detection capability. The fundamental challenge in local error sensing control is how to reconcile the conflict between energy consumption and error tolerance. In collaborative $CIES$, how to accurately determine the

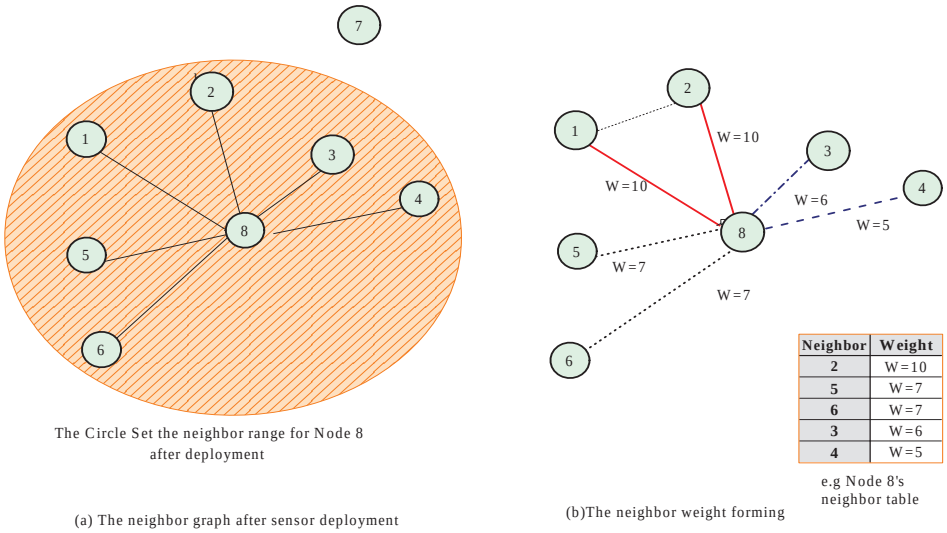

Figure 3.5: The Overview of our system design, Part I

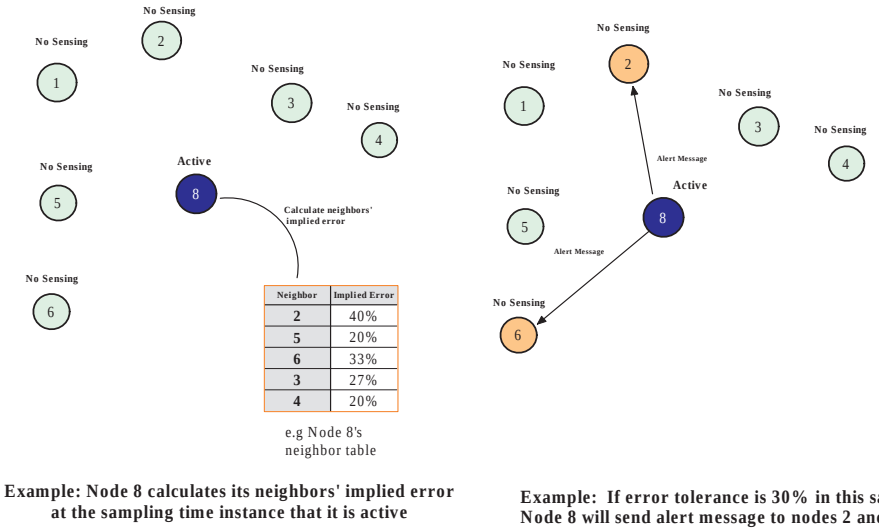

Figure 3.6: The Overview of our system design, Part II

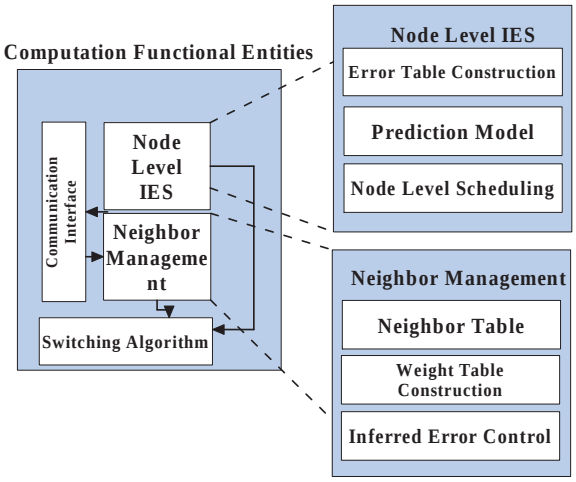

Figure 3.7: The architecture of individual node system

inferred errors for neighboring nodes will be studied. To accurately monitor the environment and minimize energy, the minimum duty need to be determined.

3.4 Details of System Design

CIES is designed as a modular architecture which can support diversified node platforms and different applications. Figure 3.7 shows the details of our distributed node level system architecture. Each computation entity incorporates three major modules: local error control (*IES*), neighbor error prediction and control, and switching control. Both error control units are built on top of a switching control module in individual nodes through deliberate interfaces. The modular-based structure also separates functionalities of individual node scheduling and neighbor management, which allows the nodes to operate independently and still maintain the necessary data accuracy. The neighbor management module can digest inferred error information from other nodes, thus affecting the underling switching activities. Overall, this generic architecture provides nodes the flexibility, reusability and efficiency in scheduling control.

As described above, two parallel error control procedures are operated independently in each node. We will discuss their implementations in the following subsections. The methods used for local error control are presented in subsection 3.4.1. Subsection 3.4.2 presents the algorithms used for predicting the inferred error. The integration of both error control procedures is also discussed.

3.4.1 Local Error Control Component

We will describe the three components built in the local error control module: prediction model, error database construction and local node level scheduling algorithm.

Prediction Model

- **Prediction Model Construction**

  In our design, the operation of the sensor system will be divided into initialization, validation and sensing phases. The prediction model is constructed in the initialization phase, in which all sensors are active to sense and record their observations. After the initialization phase, the sensor will transition to the validation phase. In the validation phase, the sensors will verify their models. After that, sensors will be operated in either a data updating phase or a prediction phase. In the data updating phase, sensors process the latest sampling observation to update the model built previously. In the prediction phase, the node will switch off the sensor to conserve energy. At this moment, the predicted sensing results are generated through its local predictor.

  The cycle-based empirical model we built [43] is shown in Figure 3.8. During an initialization phase, initial sensing differences between two adjacent cycles are calculated and updated. Then, a weighted moving average method is used to smooth those sens-

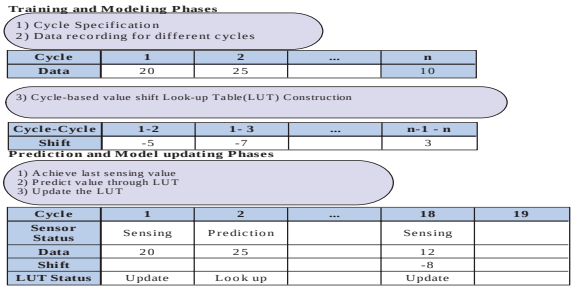

Figure 3.8: The construction of our empirical prediction model

ing differences. For example, if the sensed data at cycle 1 AM and cycle 2 AM are $20°F$ and $25°F$ respectively, the difference between those two cycle times is $-5°F$. At the end of the initialization phase, a table is generated in which the sensing difference between any two hour time periods can be estimated at the sensor node.

- **Prediction Model Update** Once the sensor is in the validating phase, the system will update the prediction look up table built in the previous phases. The system compares the prediction values produced by the empirical model with the actual sensing data. If the difference is below the system error tolerance level, a system corrective action will be taken to update the empirical model [44].

Error Database and Model Construction

In this section, we focus on determining the corresponding errors at each possible sensing cycle in order to fulfill the error boundary. For each sensor node i, we store the historical error, e_h, which is the error between observation and prediction, in the process of constructing the error Probability Density Function (PDF). Because the potential errors e_i at each cycle i are independent to each other, we can estimate the evaluation error to be a Gaussian random distribution. This is reasonable approach for long lifetime sensor applications. Therefore, a normal distribution could be used to describe the PDF of the evaluation error.

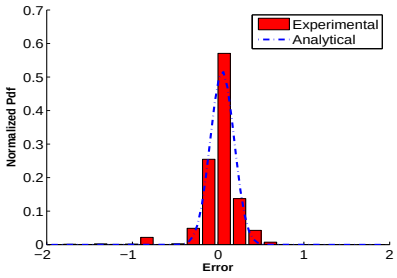

Figure 3.9: The prediction error *PDF*

We now describe the method to compute an analytical expression for the distribution of evaluation error of individual sensing cycle i. To express the normal *PDF* of the evaluation error, we note that the normal *PDF* is given as:

$$f(x) = \frac{1}{\sigma\sqrt{2\pi}} \exp(-\frac{(x-\mu)^2}{2\sigma^2}) \tag{3.1}$$

where μ and σ are the mean and the variance of the corresponding Gaussian distribution. We can compute these parameters according to mean and variance of the sampling data. The mean and variance can be expressed as: $E(x) = \mu$ and $Var[x] = \sigma^2$. The mean and variance can be solved or estimated by calculating the corresponding values during sampling periods, which will provide the parameters used to describe the *PDF* of the prediction error. We compare the outputs from our analytical analysis and experimental measurements, as shown in Figure 3.9.

We observe that the error probability distribution can be modeled quite well by a Gaussian distribution. This offers simplicity of error information dissemination within the network because nodes can reconstruct others' error information by knowledge of the σ and μ.

Local error control operation

We assume that the baseline sensor operation sequence consists of N data cycles, which include k cycles used for training. In each cycle i, the probability that the sensor will be active is defined as p_i. We further assume the average energy consumption for sensing (the energy cost of a node to sense, process and communicate) is E_a, the defined prediction error tolerance is e_t and the potential error at each cycle due to inactive sensor status is e_i. Therefore, the goal of our design is to minimize the energy consumption during each baseline period:

$$E = k \cdot E_a \cdot t_u + E_a \cdot t_u \cdot \sum_{i=1}^{N-k} p_i \tag{3.2}$$

under the constraint that

$$\frac{\sum_{i=1}^{N} (1 - p_i) \cdot e_i}{N} = \frac{\sum_{i=k+1}^{N} (1 - p_i) \cdot e_i}{N} \leq e_t \tag{3.3}$$

where t_u is the unit cycle length and e_t is the error tolerance set by the design requirements. The constraint will enforce that the potential statistical error caused by the prediction will be less than the error tolerance. The range of possible values of p_i will be bounded to satisfy the constraint equation. This problem is a challenging problem as e_i is very difficult to control and can't be solved without any prior knowledge in most cases. We use a bottom-up approach to set a boundary for the sensing probability. That is, we will not violate the constraint equation during each instance so that the sum of all cycle error products $(1 - p_i) \cdot e_i$ will automatically satisfy the constraint. As noted, this decision sets a stricter requirement than the constraint equation over all sampling instances. Therefore, our probability constraint problem can be simplified by selecting the p_i at each scheduling

cycle to satisfy the constraint on $(1 - p_i) \cdot e_i$, which can be solved as

$$p_i^{lb} = \begin{cases} 0 & 0 \leq e_i \leq e_t \\ \\ 1 - \frac{e_t}{e_i} & e_t \leq e_i \leq 1 \end{cases} \tag{3.4}$$

The p_i^{lb} is the lower bound of p_i which guarantees the satisfaction of the system data quality requirement at each sensing cycle instance. Only values higher than this will insure that the constraint requirement will not be violated under any circumstances. We should also be careful in the selection of p_i, as higher p_i implies more energy consumption by the sensor node. We use IES as the baseline algorithm of sensor operation.

3.4.2 Neighbor Error Control Procedure

In this section, we introduce the design of neighbor error control component. The neighbor error control module includes three parts: 1) neighborhood forming, 2) weight graph construction and 3) collaborative IES. These are described as follows.

Neighborhood Forming

The neighborhood forming may be based on different requirements such as vicinity, link quality or the displacement along the routing path of the sensing data. Here we will use the definition given earlier.

Weight Graph

As pointed out in our assumptions, nodes are synchronized with each other, and T_{train}, an adjustable parameter to set the model precision, is divided into equal time durations T_{build}, as in Figure 3.10. Each time duration includes m equal duration intervals, where an

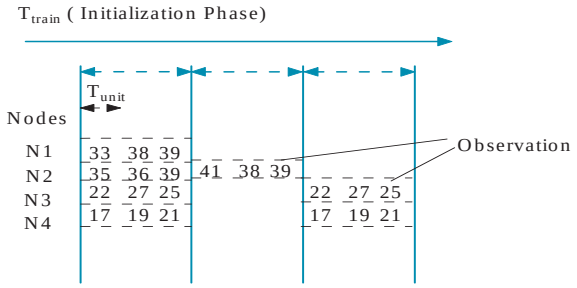

Figure 3.10: The process of Correlation Calculation

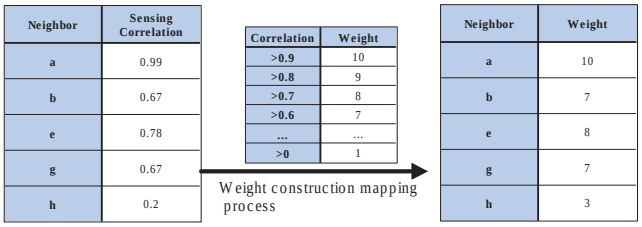

Figure 3.11: The process of weight table construction

interval is a unit sample time period set by the user. For each round, each node N_i stores its observation vector $\{o_1^i, o_2^i, ..., o_m^i\}$ obtained through discrete sampling at $T_i = \{t_1^i, t_2^i, ..., t_m^i\}$. At the end of each round, each node exchanges the observation vector, which is used to calculate the correlation between nodes. This process is repeated until the end of T_{train}, so that the average sensing correlation between nodes can be estimated. The correlation among nodes, once evaluated, remains relatively stable throughout each operation phase.

As described in Figure 3.11, the sensors will build the weight table and graph according to data correlation with neighbors through a one-to-one linear mapping.

Achieving the inferred error e_{im} from neighbors

Achieving the inferred error information from neighboring nodes is an on-going process for distributed sensor nodes. The procedure used to determine the inferred error e_{im} is illustrated in Figure 3.12. We summarize the calculation algorithm as follows:

- First, the observation error e_i at source node s is calculated when node s is active.

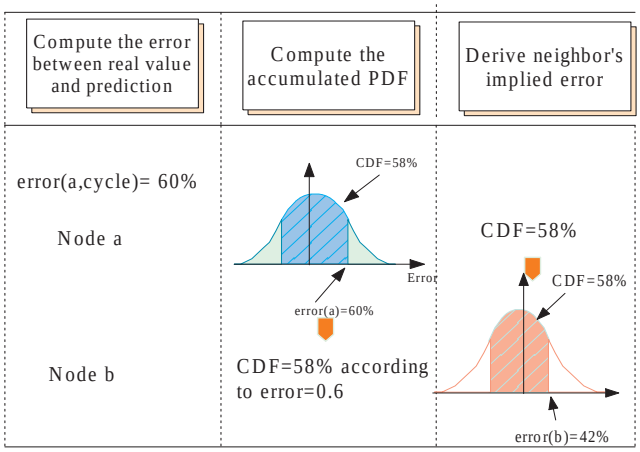

Figure 3.12: The prediction error *PDF*

- Second, node s will evaluate the accumulated *PDF* which is worse than e_i. From this result, the system can infer the confidence level that such an error e_i will occur.

- Third, node s calculates node d's inferred error using the error parameters μ_d and σ_d. Note that node s assumes node d shares similar confidence level at this time instance.

- Finally, node s will send the inferred error information e_{sd} to node d if $e_{sd} > e_t$.

Definition **weighted average inferred error**: Given a neighborhood $G(V, E)$, its weighted average inferred error e_{im} is the weighted average of all node-pair inferred errors, i.e, e_{sd}, where s and d are a neighborhood pair.

The weighted average error e_{im} is obtained according to the following rationale. Node-pairs provide different inferred errors due to correlation relationships or other influences. Pair inferred errors will have a specific weight value based on their degree of similarity. A higher weight value means a higher probability for data similarity. Therefore, the sensor platform must take this into account when determining whether the sensor node needs to

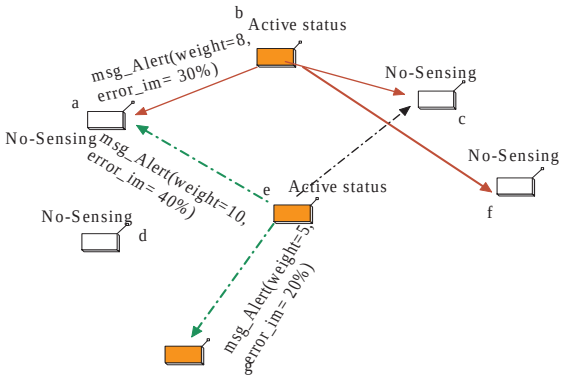

Figure 3.13: The computation of weighted implied error

change its status. Based on our observation, the inferred error can be expressed as:

$$e_{im}(s) = \sum_{k} e_{ks} \cdot w_{ks} \Big/ \sum w_{ks}, k \in N(s) \tag{3.5}$$

where k is neighborhood of the node s , w is the node-pair weight and $N(s)$ is the neighborhood list of node s. As shown in Figure 3.13, sensor node a receives two isolated error estimations from nodes b and e. The e_{im} should be viewed as a total effective contribution from all the inferred errors on a weighted basis. If one node violates the error tolerance, its neighboring nodes having a high weight value may experience a high risk of violating the data accuracy requirement.

The process that should be executed during all phases of operation is described in Algorithm 2. Local IES will be executed during the initialization phase. Algorithm 2 takes the error tolerance e_t, constructed error PDF and the inferred error e_{im} as inputs for further processing. Appropriate probability values are then be generated according to what has been described in Algorithm 2.

Algorithm 2 Sensing Error Control Algorithm

Require: e_t

1: Construct the data correlation from section 3.4.2
2: Build the error probability distribution for itself
3: Sensing and Scheduling according to its baseline algorithm
4: In sensing cycle
5: It will both update error models and compute the inferred error for neighbor $N(s)$
6: send alert message alertMsg(weight,inferred errore_{sd})
7: in prediction cycle
8: **if** $e_{im}(s) < e_t$ **then**
9: Sensor will be forced into sampling process
10: **else**
11: keep updating error models for the sensors
12: **end if**
13: Repeat 3
14: End

3.5 Insight of $CIES$ system

3.5.1 The mechanism of our error bounded approach

One of the benefits of this adaptive and collaborative approach is that it will try to reduce the average error for the entire operational period. The stricter guarantee is that it will limit the error at each sampling instance, which will enhance the system performance. The difficulty in limiting the errors is to determine when to switch on the sensors if there is a dramatic change in the environment. Our approach achieves this by relying upon both intra and inter scheduling mechanisms. Both components control the sensors' activities during drift of the sensing environment. This can be seen in Figure 3.14. As shown, the sensing activities around the rapidly changing corner of the environment's temperature becomes more extensive than during other portions of sampling period. The most effective way to guarantee the error bound is to monitor the boundary of the temperature profile because those locations are most likely to be the source of sensor prediction failures.

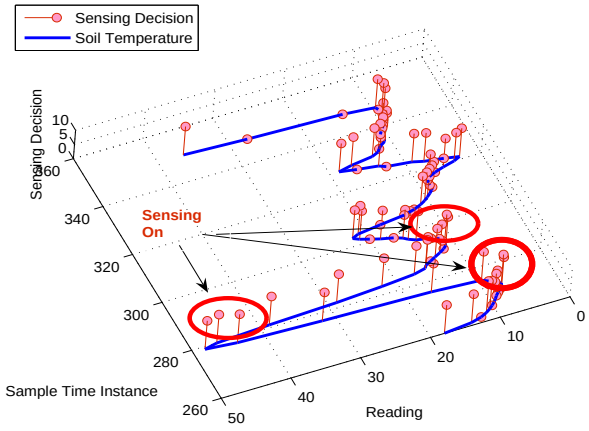

Figure 3.14: The sample of sensor activities

3.5.2 Data Quality Performance

Proposition The implementation of $CIES$ will not violate the constraint equation(3.3) that sets the error performance guarantee. That equation sets a loose requirement on the sensor scheduling over all sampling instances. Our constraint problem can be simplified into selecting the effective p_i at each scheduling cycle to satisfy the constraint on $(1-p_i)\cdot e_i$. Now there are two scenarios: i) only the sensor node i is in sensing period and all other neighbors are sleeping, ii)the sensor node i is set to be in sleep mode and there is at least one neighbor which detects the rare event. Scenario i depends on the local IES while scenario ii will rely upon the collaboration. This can be stated as followed:

$$\begin{aligned} P_i(s) = 1 - \frac{e_t}{e_{im}} + \\ P(e_{im}(s) > err_t)\cdot(1-(1-\frac{e_t}{e_{im}})), e_t \le e_{im} \le 1 \end{aligned} \tag{3.6}$$

so that

$$P_i(s) \geq P_i^{LB}(s) \tag{3.7}$$

The $P_i(s)$ is the effective probability of node s to switch on the sensors. The $P_i^{LB}(s)$ is the lower bound of sensing defined in Equation 3.4, which guarantees the satisfaction of the system data quality requirement at each sensing instance. Values that are higher than the lower bound will assure that the constraint requirement will not be violated under any circumstances.

3.5.3 Error reduction in collaborative error informing mechanism

In this section, we investigate the benefit of $CIES$ on improving the performance to rapid environmental change. As stated in section 3.3.1, all sensors are homogeneous and perform sensing operations according to their baseline algorithm. We also assume that all sensors will have the same sensing period P and will be active for the whole sensing period if they have been turned on. Since the nodes are randomly deployed, the number of sensors in a unit area follows a Poisson distribution with an expected value $\lambda = m * A$, in which A is the unit area and m is the density.

Proposition. Let P_i denote the detection probability of a rapidly changing event that occurs in the sensing area of node i, which can be viewed as the probability that the sensor i's prediction error is less than e_t. P_{i0} denotes the expected duration probability of a node, and μi and σi denote the error expectation and standard deviation for node i. Then it follows that:

$$P_i = P_{i0} + (1 - P_{i0}) \cdot (1 - \mathrm{e}^{-\lambda \cdot P_{i0} \cdot w}) \tag{3.8}$$

in which w is the average of the weight defined in our approach. **Proof**. We briefly describe the steps of the proof in the Appendix of this chapter.

As shown above, the effective probability will be much larger than P_{i0} so the miss ratio is reduced.

3.6 NUMERICAL EVALUATION

In this section, we use numerical examples of soil temperature monitoring to analyze and demonstrate the performance of $CIES$. In order to evaluate the scheduling quality of a sensor network, we define metrics as follows:

- *ErrorRate*. The error rate is defined as the error rate that the prediction system produces during the same observation window.
- *MissRatio*. The miss ratio is defined as the ratio that the sensor system fails to respond to a rare event, e.g., a rapid change in environment.
- *EnergyConsumption*. The energy consumption is defined as the total energy consumed by the network during the operation period.

The sensing schemes proposed will be assessed using the above metrics with respect to different system parameters, e.g. the error tolerance. Through these examples, comparison between different benchmarks and our proposed $CIES$ will be used to demonstrate the performance of our design.

Along with the experimental setup, a simulation program which uses historical soil temperature data was developed. The temperature data was collected from the Wisconsin-Minnesota Cooperative Extension Agricultural Weather Page [45] where soil temperature is monitored continuously, sampled twice per hour, 24 hours per day. This full record of data over the past 10 years, large enough to reduce sampling randomness, allows us to verify our

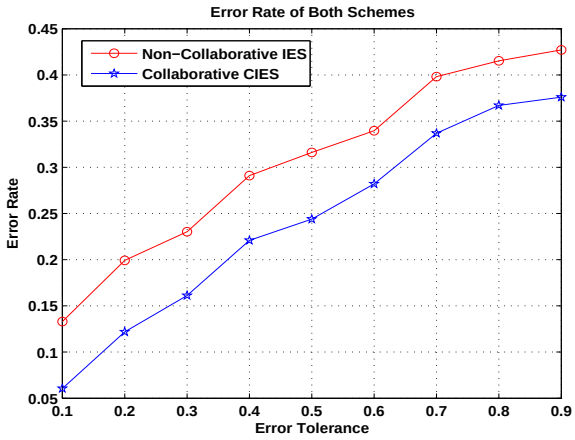

Figure 3.15: The error performance with different error tolerance

strategy and investigate the impact of different configurations on the performance of energy conservation and error control.

In our experiments, we use randomly deployed sensors in a square area of 200 m $\times$ 200 m. We define a pair of neighboring sensor nodes whose distance is less than 20 m. Moreover, we use a diffusive model to fill up the simulation environment, in which we consider the environment as a homogeneous semi-infinite medium.

In the first example, there are only two sensors that are adjacent to each other. This shows the performance of the simplest case of $CIES$.

Figure 3.15 illustrates the error performance of non-collaborative and collaborative $CIES$. Over error tolerances, we can see that both approaches can satisfy the error performance requirement. However, under the collaborative $CIES$ scheme, the error rate is at least 20% less than with stand-alone IES.

Figure 3.16 demonstrates the metric of miss ratio for stand-alone IES and $CIES$. In the extreme region, (i.e., the error tolerance is 10%), the miss ratio is about 25% less with the collaborative information. The effectiveness is well demonstrated here.

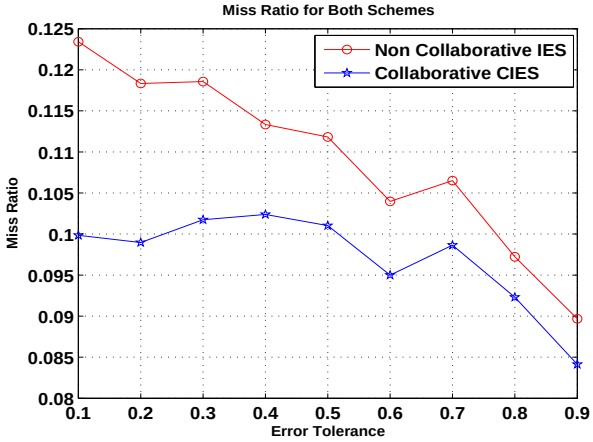

Figure 3.16: The Miss Ratio with different error tolerance

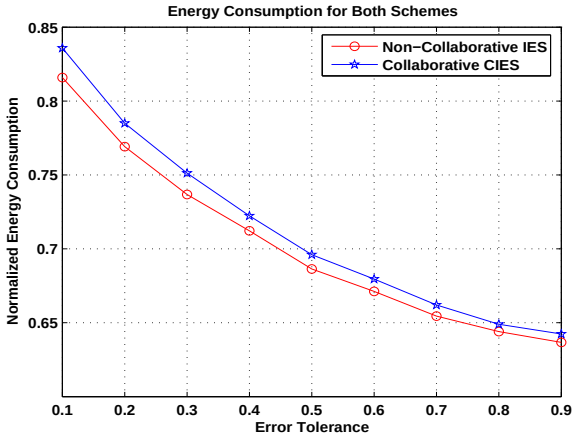

Figure 3.17: The energy consumption with different error tolerance

Figure 3.17 demonstrates the energy consumption for both schemes. Compared to the 25% error rate improvement in $CIES$, the additional energy consumption is small as the maximum difference between the two schemes is only 5%.

From the above three figures, we can conclude that the $CIES$ method is superior to the stand-alone IES scheme with slightly more energy consumption. This cost can be traded for the reduction in miss ratio, which is quite important in certain monitoring applications.

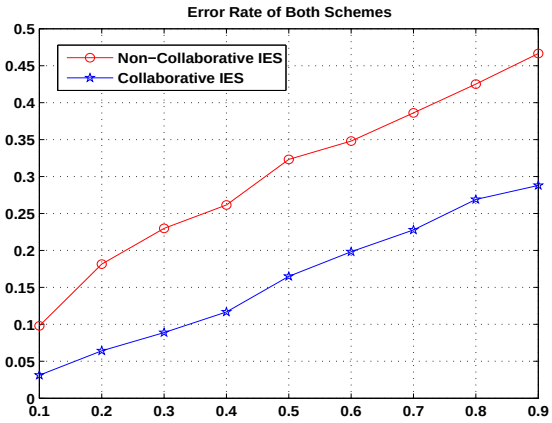

Figure 3.18: The error performance with different error tolerance

Algorithms	MR(e_t = 10%)	EC(e_t = 10%)	MR(e_t = 20%)	EC(e_t = 20%)	MR(e_t = 30%)	EC(e_t = 30%)
eSense(dynamic)	0.09	0.71	0.14	0.67	0.18	0.62
IES	0.09	0.82	0.098	0.76	0.101 s	0.72
CIES	0.075	0.85	0.077	0.80	0.081 s	0.76

Table 3.1: The Performance comparison with eSense

3.6.1 The impact of node density

In the second example, we raise the node density m to 20 and study the effect of node density on the performance metrics. Figure 3.18 shows the error rates associated with each approach. Compared to the results in Figure 3.15, the error rates are dramatically reduced while the gap between the two schemes is increased. This is expected because there is a greater chance for the sensor to be awakened by neighboring nodes.

It is even clearer from the miss ratio performance result. The miss ratio is reduced almost by 45% for different levels of error tolerance. This shows the validity of collaborative IES. However, as shown in Figure 3.20, the energy consumption is slightly higher but still acceptable considering the improvement in miss ratio. It is the application's choice to balance the trade-off between energy consumption and other performance metrics.

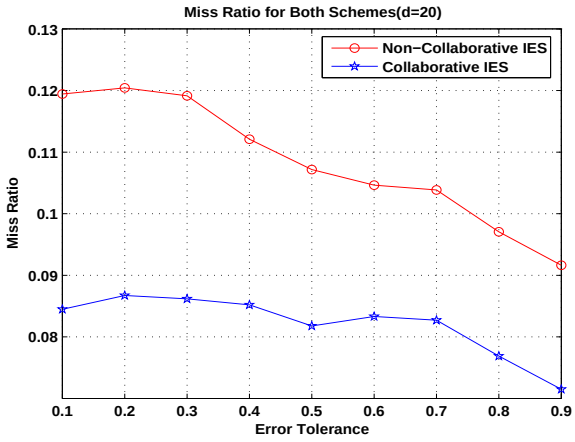

Figure 3.19: The Miss Ratio with different error tolerance

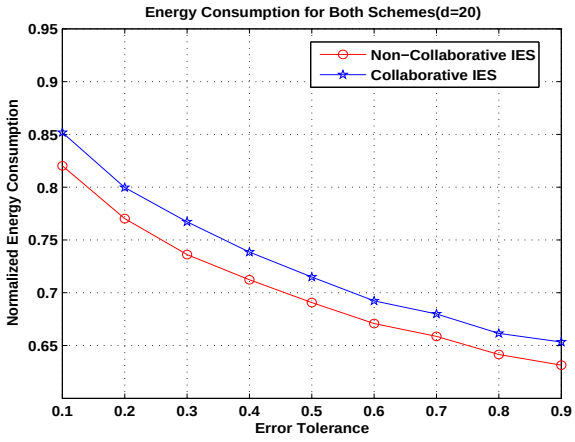

Figure 3.20: The energy consumption with different error tolerance

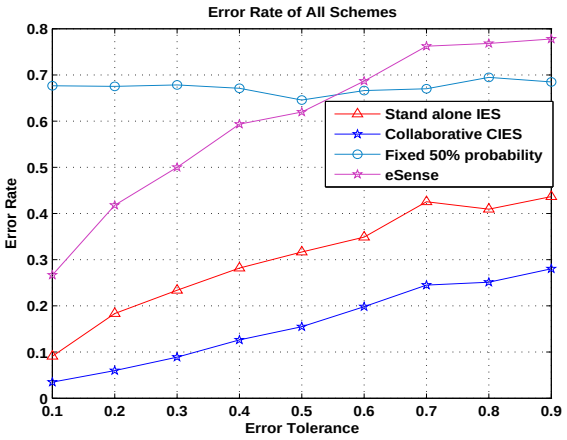

Figure 3.21: The error performance with different error tolerance

We also compare the performance of our approaches to state-of-art eSense approach and a benchmark. The benchmark is to set a random 50% probability rate for P_i for each time instance i. We implement the principle of eSense which is to control the probability for miss ratio performance into our simulation system. The performance of all approaches are demonstrated below.

- **The error rate comparison**. The error performance is demonstrated in Figure 3.21. As can be seen, the benchmark is not affected too much by the setting of error tolerance. Although the error rate for eSense does not increase too much due to the increase of error tolerance, its error rate is much higher than both IES and $CIES$ approaches. Note that the error rate determined by eSense can not satisfy the error rate boundaries while both stand-alone IES and collaborative $CIES$ approaches meet the requirement.

- **The miss ratio comparison**. The miss ratio for eSense continues to increase with the increase of error tolerance, while our methods seem to be stable. The hike in eSense is due to the reduction in sensitivity to the change when the threshold or risk

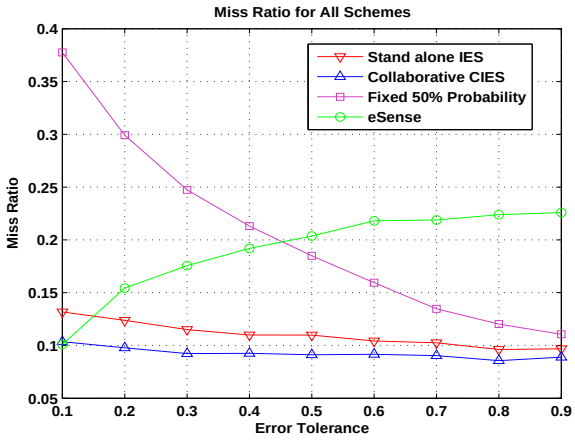

Figure 3.22: The Miss Ratio with different error tolerance

tolerance increases. The higher the risk tolerance, the less sensitive eSense is to the data change, leading to a higher miss ratio. Our system can guarantee the absolute error rate, which will keep tracking the past error and adjust accordingly. Therefore, our approaches will not experience a similar hinderance.

- **The energy consumption comparison**. The energy consumption shown in Figure 3.23 indicates that eSense consumes slightly less than 15% of that of our *CIES*. However, the stability of our system is much better than eSense as the maximum variance of energy consumption is just 12% compared to almost 100% in eSense. These achievements only cost 20% more energy consumption than eSense, which may be acceptable. In some situations, rare event detection is as important as lifetime management.

To see the comparison more clearly, we list the performance metrics of our design with those reported in *eSense* [19]. The results are shown in Table 3.1.

Based on comparisons, we conclude that *CIES* can outperform the eSense with respect to the miss ratio and total error rates. The results also confirm that the error-bounded

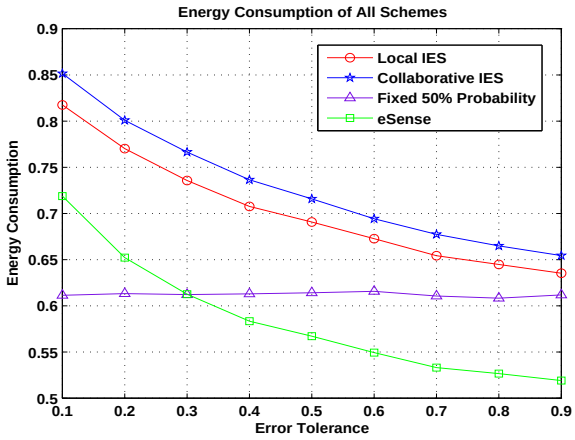

Figure 3.23: The energy consumption with different error tolerance

scheduling limitation, lacking in eSense, is achieved in our approach.

3.7 Conclusions

In this chapter, we have presented a stochastic sensing algorithm to reduce energy consumption. Our approach uses the data correlation between nodes to reduce the error rate for prediction model performance. Observed correlations between nodes are used to estimate the neighboring nodes' errors, and adjust their operation accordingly. The measurement and simulation results show that the system prediction error remains within the specified error tolerance while saving up to 60 percent of the required energy. For our future work, we will evaluate the energy performance of individual sensor network components so that the algorithm can be further optimized.

3.8 Appendix

3.8.1 Estimation of Sensing Probability

Here we prove our proposition in Section 3.5.2. To derive the detection probability, we should consider two scenarios: i) only the sensor node i is in sensing period and all other neighbors are sleeping, ii) the sensor node i is set to be in sleeping mode and there is at least one neighbor which detects the rare event. Given the probability P_{i0}, we can obtain the total probability for the two scenarios described earlier. For the first case, the probability will be P_{i0}. For the second case, in order to calculate the probability, we first determine the probability that multiple sensors can not detect the rare event P_m, so $1 - P_m$ will be our intended result. The probability that there are k neighbor sensors (including the sensor i itself) in the active area is

$$P(n = k) = \frac{e^{-\lambda} \cdot \lambda^k}{k!}$$

where $k = 0, 1, ..., \infty$, while the probability that a sensor cannot detect an rapidly changing event is $1 - P_{i0}$. The probability that there exists k neighbor nodes and at least one of them can detect the rapid changing event so as to wake up the node i with average weight w is

$$P_{\text{det}}(n = k) = \frac{e^{-\lambda} \cdot \lambda^k}{k!}(1 - (1 - P_{i0} \cdot w)^k) \tag{3.9}$$

When $k = 0$, we have

$$\begin{aligned} P_{\text{det}}(n = 0) &= \tfrac{e^{-\lambda} \cdot \lambda^0}{0!} \\ &= e^{-\lambda} \end{aligned} \tag{3.10}$$

and because $P_{n=k}$ is the Poisson distribution, we have

$$\sum P(n = k) = \sum \frac{e^{-\lambda} \cdot \lambda^k}{k!} = 1 \tag{3.11}$$

this gives:

$$\sum_{k=1}^{\infty} \frac{e^{-\lambda} \cdot \lambda^k}{k!} = 1 - e^{-\lambda} \tag{3.12}$$

From the equations provided above, we can derive the total effective probability for sensor node i to be

$$\begin{aligned} P_i &= P_{i0} + (\sum_{k=1}^{\infty} P_{\text{det}}(n=k)) \cdot (1 - P_{i0}) \\ &= P_{i0} + (1 - P_{i0}) \cdot \sum_{k=1}^{\infty} \tfrac{\mathrm{e}^{-\lambda} \cdot \lambda^k}{k!} (1 - (1 - P_{i0} \cdot w)^k) \\ &= P_{i0} + (1 - P_{i0}) \cdot \sum_{k=1}^{\infty} \tfrac{\mathrm{e}^{-\lambda} \cdot \lambda^k}{k!} - \sum_{k=1}^{\infty} \tfrac{\mathrm{e}^{-\lambda} \cdot \lambda^k}{k!} (1 - P_{i0} \cdot w)^k \\ &= P_{i0} + (1 - P_{i0}) \cdot (1 - \mathrm{e}^{-\lambda} - (\mathrm{e}^{-\lambda \cdot P_{i0} \cdot w} - \mathrm{e}^{-\lambda})) \\ &= P_{i0} + (1 - P_{i0}) \cdot (1 - \mathrm{e}^{-\lambda \cdot P_{i0} \cdot w}) \end{aligned} \tag{3.13}$$

3.8.2 Sensor lifetime estimation based on IES

One of the advantages in calculating the effective sensor switching probability is that we can use it to estimate the lifetime of a sensor node. Previously, we have established historical error distribution (PDF), which could be used to estimate the probability of an individual sensor's operational duty cycle. As shown in equation 3.14,

$$(1 - P_i) \cdot \varepsilon_i \leq \varepsilon_t \tag{3.14}$$

and we have to satisfy $P_i \geq 1 - \frac{\varepsilon_t}{\varepsilon_i}$. This relationship tells us that

$$E(P_i) \geq E(1 - \frac{\varepsilon_t}{\varepsilon_i}) = 1 - \varepsilon_t \cdot E\left(\frac{1}{\varepsilon_i}\right) \tag{3.15}$$

It is quite interesting that we can take advantage this property to estimate the maximum lifetime by assuming the equal energy consumption to be

$$E_c \cdot nT \cdot E(P_i) = E_r \tag{3.16}$$

in which E_c is the unit operation energy consumption and E_r is the total energy remaining for the sensor through the measurement. From the equations above, the expression for n can be derived as

$$n \leq \frac{E_r}{E_c} \cdot (1 - \varepsilon_t \cdot E\left(\frac{1}{\varepsilon_i}\right)) \tag{3.17}$$

This relationship can then provide us with a fundamental basis to optimize trade-offs between lifetime and error.

Note that the expectation of error can not be directly solved by a classical approach. But in a typical application, the minimum lifetime will be of the most interest rather than the maximum lifetime of sensor networks. Therefore, our approach can be rewritten as:

$$p_i = \begin{cases} 0 & 0 \leq e_i \leq e_t \\ \\ 1 - \frac{e_t}{e_i} & e_t \leq e_i \leq 1 \end{cases} \tag{3.18}$$

The expectation of P_i can be obtained as:

$$\begin{aligned} E(P_i) = & \int_{-\infty}^{-\varepsilon_t} \frac{1}{\sigma\sqrt{2\pi}}(1 + \frac{\varepsilon_t}{x}) \exp(-\frac{(x-\mu)^2}{2\sigma^2})dx \\ & + \int_{\varepsilon_t}^{\infty} \frac{1}{\sigma\sqrt{2\pi}}(1 - \frac{\varepsilon_t}{x}) \exp(-\frac{(x-\mu)^2}{2\sigma^2})dx \end{aligned} \tag{3.19}$$

Although there is no closed-form for this integration, a numerical solution can be used to provide the result. After obtaining the expectation of P_i, equation 3.17 can be solved to

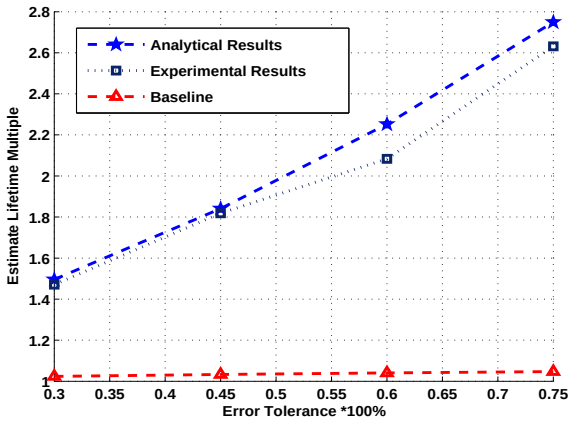

Figure 3.24: The lifetime estimation of sensor network

provide an estimation of the lifetime of the sensor node. Figure 3.24 demonstrates that our analysis matches the simulation very closely.

Chapter 4

TwinsNet: A Cooperative MIMO Mobile Sensor Network

4.1 Introduction

The 1980s and 1990s brought the rise of the industrial robot arm. Robots assembled virtually anything in high-volume, carefully controlled assembly lines. At first, they worked alone as "islands of automation." Gradually, as networking improved, the benefits of teaming were realized to create more capable systems with a trend toward larger numbers of smaller robots [46, 47].

The 2000s have brought the rise of the mobile robot. They have followed a similar trajectory by starting alone, in carefully controlled environments. Yet, a new era has suddenly emerged as mobile robots have begun to appear in uncontrolled environments. The Roomba robot can autonomously vacuum virtually any residential room in which it is placed while the DARPA Grand Challenge showed robots can navigate difficult mountainous desert terrain for long periods of time. As with robot arms, the natural trend is moving

toward robot teams (even "swarms" [48]) of larger number and smaller size.

Small size presents a significant problem for mobile robot teams. Networking is the key enabler and the only practical infrastructure for networking among large numbers of mobile robots is RF wireless. As robots get smaller, their antennae get closer to the ground. This detunes the antennae, reduces line-of-sight, and increases multi-path problems. Furthermore, small robots carry small batteries, limiting the radiated power of onboard RF systems. Finally, as robots move into extremely uncontrolled environments such as subterranean urban search and rescue [49], [50], [51–53], high performance RF networking becomes a paramount concern. Teaming to maintain the network and conserve power is just as important, and often synonymous with, teaming to accomplish the task.

For applications in urban search and rescue, high quality video is the sensor mode of choice [54, 55]. Imagery from different viewpoints must be relayed from remote mobile sensors back to the human operator. Therefore, it is desirable to maximize communication bandwidth across a team of robots while minimizing power consumption and allowing the robots to achieve their individual goals. The robots have individual goals for search and exploration, but common goals for maintaining the networking infrastructure. We assume every robot can act as both a remote sensor and a network relay. As robots fan out from the base station into the hostile RF environment, they quickly lose direct contact with the base station and must rely on multi-hop relays to maintain the network. Issues such as link quality, power minimization and robustness are major factors to be considered when designing wireless sensor networks [56–58].

In recent years, multi-input multi-output (MIMO) systems have been widely adopted to enhance the performance of modern communication systems [58, 59]. In such systems, multiple antennas are employed at the transmitter and/or receiver to combat channel fading

with space diversity while enabling significant increases in the capacity of the wireless data link. MIMO techniques are readily applicable to conventional fixed base stations because they have ample processing and power resources and can accommodate multiple antennas with beamforming or transmit diversity capabilities. Unfortunately, multiple antennas are not feasible for mobile robot teams due to their size, power and complexity constraints. An alternative is to let groups of robotic sensors in geographic proximity of one another to form corporative distributed antenna arrays and serve jointly as multiple antennas to enable distributed space diversity and low-power connectivity.

An additional form of corporation among the small-sized robotic sensors is the relayed communications [55, 60, 61]. Specifically, the non-information-bearing nodes can help relay the information from the source nodes to the destination nodes, as depicted in Figure 1(b). This can effectively extend the small coverage range by individual robotic sensors due to their limited power and the intensive multipath channel fading effects.

The remainder of this chapter is organized as follows. In Section 4.2, the cooperative MIMO dual-tree network topology and communications protocol are described. In Section 4.3, our use of SystemC-based transaction-level modeling techniques [62, 63] for this application are explained. Then, in Section 4.4, we present simulation results for the behavior of the system as a function of system sizes and robot speed.

4.2 Design of TwinsNet

In this section, we describe the structure and operation of TwinsNet, which creates dual sub-networks to allow for cooperative $2x2$ MIMO transmission of data packets between robot-mounted sensor nodes and the base station.

4.2.1 Establishment of Network Topology

Each sensor node is assigned a unique ID within the network. During network initialization, the base station issues messages which propagate throughout the system. Two disjoint sub-networks are created such that all the nodes in one tree have even ID numbers (called the Girl Tree) while all nodes in the other tree have odd ID node numbers (called the Boy Tree). Each node in both trees also has a specific father and mother, which are one hop closer to the base station. A pair of even and odd nodes are grouped together as a Sister/Brother. These two nodes correspond either to two sensors on the same robot or two sensors on separate robots which are physically close to one another. In this manner, a virtual 2×2 MIMO structure is overlaid onto the network to increase the throughput capabilities of the system. In addition, robots which do not currently have data to transmit will cooperate with an information bearing sensor by positioning themselves in such a way as to minimize overall energy consumption.

An example network topology for *TwinsNet* is shown in Figure 4.1. Nodes 7 and 8 are Brother/Sister, Node 5 is their Father and node 6 is their Mother. When information is detected by either node 7 or node 8, it will exchange information with its sibling and data will be sent back to the base station via two routing trees (Boy/Girl Trees). Node 7 and 8 communicate with nodes 5 and 5 a using 2×2 MIMO transmission technique. A series of such transmissions are used to relay the information back to the base station. After the initialization process, each node maintains a neighbor table and dynamically updates it to reflect changes in link quality due to the robot-mounted sensor movements.

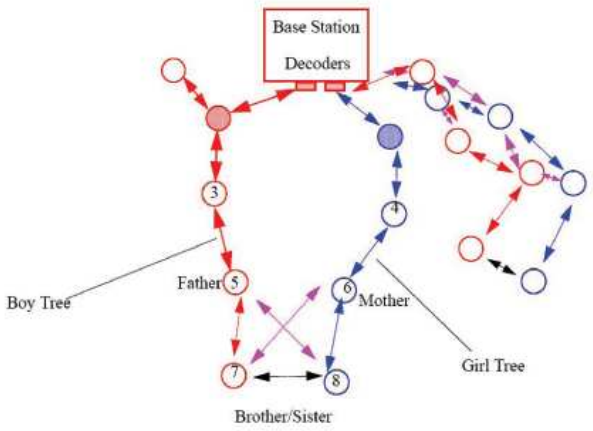

Figure 4.1: An example topology of $TwinsNet$

4.2.2 Link Quality Estimation

After the tree building process has been completed, the sensor nodes will maintain their Parent and Brother/Sister assignments although they may be moving around within some region. However, a new initialization procedure will be invoked when the link quality drops down below a certain threshold. If this condition is met, then one or more nodes would have to find another candidate link available to them and a portion of the tree building protocol will be invoked.

4.2.3 Localization

A robot's physical location can be determined by a number of methods that fall into three broad classes: proprioceptive sensing, external measurement, or landmark-based localization. Proprioceptive sensing involves measuring the internal state of the robot and estimating its motion. Examples include inertial measurement and deduced reckoning, both of which experience drift. External measurement relies on a calibrated entity in the environment to detect the presence and identity of a robot within its sensor field and report

the corresponding location. Examples of this include a stereo camera fixed in the world (e.g. surveillance cameras) but also include another robot observing the target robot's movements. Landmark-based localization relies on sensors onboard the robot to detect quasi-static, distinguishable features in the environment that establish a rigid coordinate system. This coordinate system can be either known or unknown and the landmarks can be either placed or spontaneous. GPS is an example of a landmark-based localization system with placed landmarks in a known coordinate system. On the other hand, a mobile robot with a camera might detect visual features on the walls of a collapsed structure, an example of spontaneous landmarks in an unknown coordinate system. A particular localization technique is chosen based on the payload of the robot, structure of the environment, required accuracy of localization and the need for absolute versus relative localization.

4.2.4 MIMO Transmission and Minimum Energy Positioning

The communications between the sister/brother nodes to the parent nodes take the form of cooperative antenna array. The sister/brother nodes first exchange their information to be transmitted. Depending on the required information rate and error performance, and relying on the availability of channel information, the distributed antenna arrays can deploy different corporation strategies. Examples of such strategies include transmit diversity and beamforming. These are complementary techniques: beamforming is capacity achieving when perfect channel state information (CSI) is available at at both the transmitting and the receiving sides; while space-time coding (STC) requires no CSI at the transmitting side. The former is adaptive to the acquired CSI while the latter remains robust regardless of the unknown CSI.

At the next stage, the transfer of information from the parent nodes to the data fusion center is achieved via non-information-bearing nodes serving as relays. If a low-rate

high-performance strategy is adopted in the sister/brother-to-parent link, then the boy tree and the girl tree are relaying the same information. Otherwise, they are conveying distinct data streams. A majority of existing works on relay networks focuses on coherent demodulation based on the availability of the channel state information at both the relays and the destination node (see e.g., [64–67]). Accurate estimation of the CSI, however, can induce considerable communication overhead and transceiver complexity, which increase with the number of relay nodes employed. In addition, CSI estimation may not be feasible when the channel is rapidly time-varying. To bypass channel estimation, cooperative diversity schemes obviating CSI have been recently introduced. These relay systems rely on non-coherent or differential modulations, including conventional frequency-shift keying (FSK) and differential phase-shift keying (DPSK) [68–71], as well as space-time coding (STC-)based ones [72–75].

To improve the error performance and enhance the energy efficiency of relay networks, optimum resource allocation recently emerges as an important problem attracting increasing research interests (see e.g., [76–79]). However, all of them only consider the power allocation. To this end, we take a major shift by jointly optimizing the power distribution and the source/relay locations. Interestingly, we found that location optimization may be more critical than energy optimization. In addition, it can also better exploit the mobility of the robot teams.

To verify the advantage of the optimum energy allocation, we plot in Figure 4.2 the SER of the cooperative system with and without energy optimization, at 10dB and with various numbers of relays L=(1,2,3,4). As a benchmark, we also plot the SER of a direct transmission using all available energy. In the un-optimized system, a uniform energy allocation is employed. From Figure 4.2, we observe that: 1) the un-optimized cooperative

transmissions may even under-perform the direct transmission, regardless of the number of relays L; 2) as L increases, the SER performance can get even worse unless the energy optimization is performed; 3) the energy-optimized cooperative transmissions universally outperform the direct transmission with the same total energy; and 4) the energy-optimized cooperative transmissions universally outperform the un-optimized ones.

Interestingly, notice that the minima of the un-optimized SER curves almost coincide with those of the energy-optimized ones. This implies that the near-optimum SER can be achieved even with the uniform energy allocation, provided that the relay location is carefully selected. As shown in Figure 4.2, the optimum relay location corresponding to the uniform energy allocation shifts from the midpoint for L=1, to 0.35, 0.29 and 0.26 for L=2, 3 and 4, respectively.

In Figure 4.3, we verify the benefits of the optimum distance allocation. In the system without location optimization, the relays are placed at the midpoint of the $s - d$ link, as suggested in [RiCG05,ZhLi04]. Similar to the energy optimization case, Figure 4.3 confirms the SER improvements of the location optimization. However, the curves in Fig. 4.3 exhibit more flatness compared with the ones in Figure 4.3. In addition, the minima of the un-optimized curves in Figure 4.3 are much worse than the location-optimized ones, except for the L=1 case. In other words, placing the relay nodes at the midpoint cannot achieve the minimum SER even with careful energy allocation, for any $L > 1$. All these indicate that the system SER is more sensitive to the location distribution than the energy distribution. To the best of our knowledge, we are the first to make this discovery, which is critical for the resource management and optimization of cooperative networks.

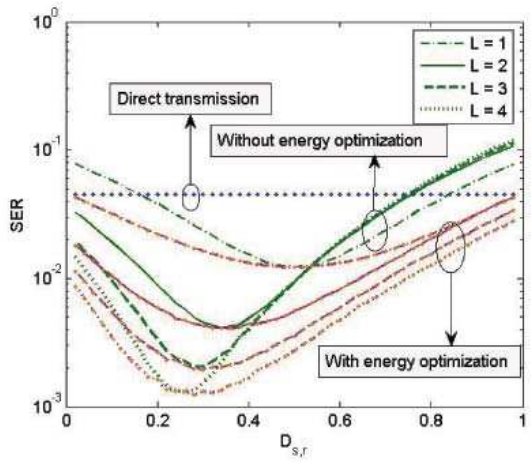

Figure 4.2: SER comparison between relay systems with and without energy optimization.

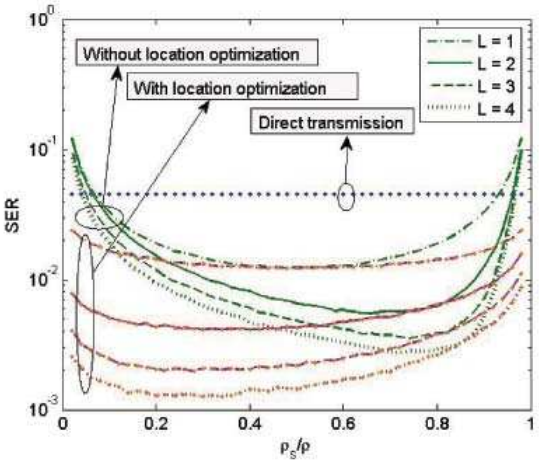

Figure 4.3: comparison between relay systems with and without location optimization

4.2.5 Prioritized Packet Propagation

The packets transmitted by a node fall into two categories, those originating at that node and those which are being passed from a downstream node. A priority scheme is used to determine which messages have a higher priority at any given time. Memory queues are used to store packets which are waiting to be sent.

The prioritization scheme is largely system dependent. It could be entirely user specified such that the user specifies what sensor readings are delivered. It could also involve a weighting of user-desired sensor packets, packets required to maintain some minimum level of situational awareness, packets required to maintain minimum accuracy of localization, and packets required to minimize energy consumption.

4.3 Transaction-Level Modeling of TwinsNet with SystemC

Recently, a system design methodology based on Transaction-Level Modeling (TLM) [80–82] has been gaining acceptance in many application areas. This methodology is enabled by using a language such as SystemC. SystemC is a C++ class library and simulation kernel that have been developed to allow designers to simulate hardware and software components together, thereby improving overall design productivity. SystemC can be used for modeling both concurrent and sequential systems given its event-driven simulation environment [5]. A working group of the Open SystemC Initiative (OSCI) is currently defining a set of standards for TLM and is developing associated libraries. TLM-based system design has several advantages compared with a traditional design methodology:

1) TLM emphasizes the functionality of the data transfers, i.e. what data are transferred to and from which locations. The pin-level details of how the transfers are accomplished are not usually included. 2) Synchronization details are abstracted into categories called block-

ing and non-blocking I/O rather than modeling the various handshake signals explicitly. 3) TLM enables higher simulation speed than pin-accurate interfaces because non-essential details are not modeled.

We have implemented an efficient TLM-based simulation model of TwinsNet in order to evaluate its performance. The basic elements in our TLM-based SystemC simulation are the channel, base station and sensor node modules, which are described below.

4.3.1 Channel

This mechanism creates a connection between components so that the original or processed information can be transmitted within the network. For simulation efficiency, we employ the sc_fifo channel, a method which implements a first-in-first-out flow. This channel defines read, non-blocking read and write interfaces that are ready for use in the data management. A channel can operate with the assistance of its interfaces to read data from the ports or to send data via the ports of the channel. The sc_fifo channel can also be used to buffer data during communication. The default depth of a channel is 16 and the data type of the elements in a channel can be characters, an arithmetic data type or other general data types. Input and output port classes are also provided which facilitate the fast implementation and easy management of the interfaces.

4.3.2 Base station

In order to implement the function of receiving and processing the data arriving from the sensors within the network, a bidirectional port array is created to enable the base station module to gain access to any channel connecting to those sensor nodes. Either of two scanning options can be chosen for the base station to communicate with the sensor nodes. One technique is to respond to its data input ports in an interrupt-driven manner. As

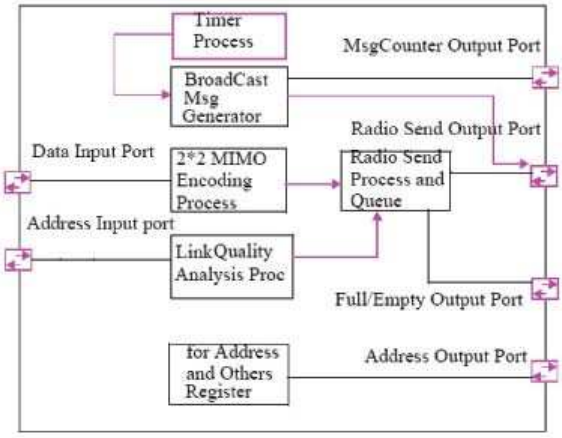

Figure 4.4: TLM model of a TwinsNet sensor node

the number of channels wishing to communicate at a specific time cannot be predicted in advance, it may be useful for the base station to automatically respond to a channel whenever there is a request from a particular sensor node. The other option is to use polling, i.e. to scan its input ports in a cyclic manner. Both of these options can be implemented by using either the SC_THREAD or SC_ METHOD process. A key difference between these two types of processes is that an SC_METHOD process can't be suspended during its execution while an SC_THREAD process may be suspended.

4.3.3 Sensor node

The processes inside each sensor node are organized to assure the successful and fast collection and delivery of data throughout the network. A sensor node includes ports which are inherited from the sc_fifo interface. It sends its data into the channel making use of the port's inherited functionality. The components within a sensor node are illustrated in Figure 4.4 and are described below.

2×2 MIMO encoding process: The level of detail describing the physical layer of the

MIMO space-time coding and transmission can be adjusted corresponding to the user's requirements.

Incoming message handling process: This module handles the processing of incoming messages sent by other sensor nodes (i.e. from the Son and Daughter) . It reads data from the input ports of sensor nodes and passes the processed data to an output radio or to an internal message queue. The sensor node's radio function can be simply simulated by employing a process for continuous transmission of the data. The type of data can be defined in advance according to different communication control mechanisms. Character and arithmetic data types are used in our abstract model.

Neighbor management process: Sensor nodes in a mobile communication domain have to continually update their neighbor table and manage the communication among themselves to ensure the connectivity of sensor network. This process broadcasts a requesting message to its neighbor. The neighbor receiving this message would add the source of this message into its neighbor table.

Link quality analysis process: This process is used to analyze the current link quality between a sensor node and its siblings and parents. A model including parameters such as motion speed, radio communication range, etc. is invoked to decide whether it needs to modify its local network topology by connecting to other nodes according to the selection protocol discussed above.

Address management process: We use a simple address generation scheme to speed up our simulation since localization information is achieved through other methods.

Timer process: A timer maybe needed to trigger the activation of a sensor node, either to mimic an event or to simulate its sleeping/active mode.

4.4 Simulation Results

We have implemented our TLM model on a Sun Solaris based SystemC platform. Specifically, we have used a Sun Blade 150 workstation having a 650 MHz clock frequency and 512 Mbytes of memory. The simulation program has been written using the SystemC 2.1 distribution available form www.systemc.org.

4.4.1 Simulation Efficiency

We first compare the influence of different scanning mechanisms on the simulation performance by using a very simple model in which sensor nodes are one-hop connected to the base station so that we can eliminate the effects of routing. One series of simulations were done for the case where the input ports connected to the sensor node's channel were modeled with sensitive variables. The base station responds to the channel only if it has data to transmit. The other series of simulations were performed when those input ports were in the cyclic scanned (i.e., polling) mode according to a predefined connection sequence of channels that was assigned by the base station during the initialization phase. The execution time cost for various network sizes when the network operation is simulated for 1000 nanoseconds are listed in Table 4.1.

Mechanism	100 sensors	200 sensors	300 sensors	400 sensors
Event response	0.11 s	0.29 s	0.34 s	0.44 s
Cyclic scanning	0.12 s	0.20 s	0.33 s	0.42 s

Table 4.1: The execution time cost

From the simulation results we can see that the execution time scales roughly linearly with the network size and that port communication mechanism does not have a significant effect on the required simulation time.

Next, the influence of different sc_fifo interfaces (i.e., non-blocking read and blocking

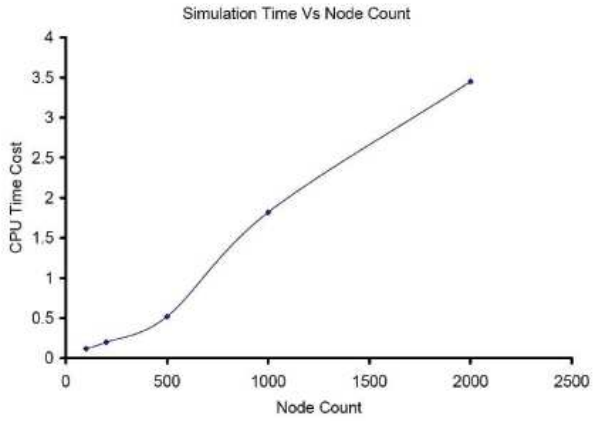

Figure 4.5: Simulation time for 1000 ns of network operation as a function of the number of nodes in the system

read) under the cyclic scanning mode described above (again for a SISO network) are presented in Table 2.

Read Operation	100 sensors	200 sensors	300 sensors	400 sensors
Non-blocking	0.12 s	0.18 s	0.27 s	0.35 s
Blocking	0.12 s	0.20 s	0.33 s	0.42 s

Table 4.2: The influence of different sc_fifo interfaces

To measure the performance of our TLM-based scheme for large scale sensor networks, we have simulated topologies consisting of sensor node counts from 100 to 2000. The run time cost for a system architecture with 100 nodes is only 0.12 seconds for a simulated time of 1000 nanoseconds. It can be seen from Figure 4.5 that the run time cost is roughly proportional to the scale of the sensor network but it remains at a relatively low value of 3.45 seconds even for a large system containing 2000 nodes.

4.4.2 Link Quality Analysis

To model if the connectivity of mobile sensor nodes is successful for any given transaction, a random process is used for the link quality. If a randomly generated number falls into the range corresponding to a viable transmission, then the message will be transmitted. For example, suppose that the current link quality for a node to connect to its father is 80%. If the C rand function ($0 < rand() < 1$) gives a result of 0.9, then the message is considered to be lost. The link quality is modeled as follows:

$$link\_quality = \frac{A}{\pi r^2} + B \tag{4.1}$$

where A and B are adjustable parameters and where r is the distance between the two sensor nodes.

In order to determine the impact of sensor movement and the density of sensors within a physical region, we considered a grid of size 500 meters by 500 meters with a grid spacing of 10 centimeters. Events are randomly generated at the leaf nodes of each of the two trees. This process is controlled by the timer process inside the model. After timing out, a simulated event is triggered. The message related to that event is then transmitted upstream by the sensor nodes. A packet loss is assumed to occur if a node cannot find any connection around it having a link quality of greater than 70%. The node parameters we considered are:

1. The movement of nodes: A robot's speed is set to be constant in each simulation but the direction is randomly chosen using a Monte Carlo algorithm.

2. The number of nodes: We consider a situation in which all of the mobile nodes are initially placed within a small circle so that all them can reliably detect other nodes.

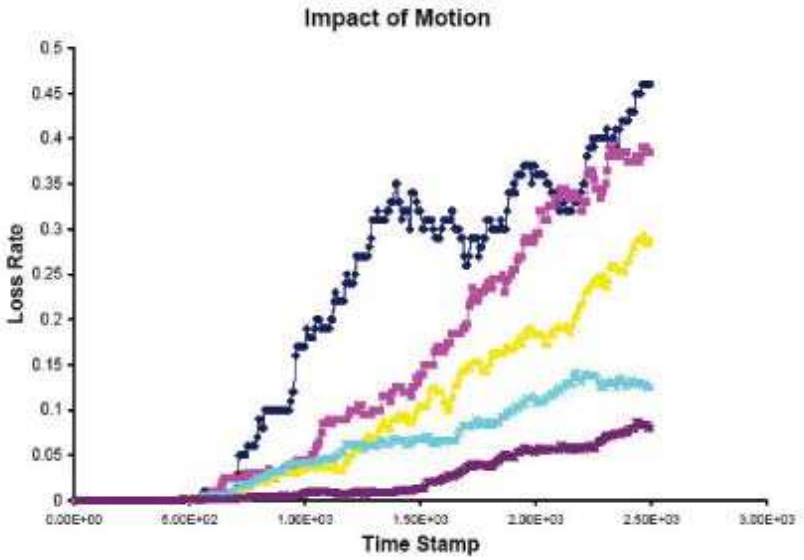

Figure 4.6: Loss rate as a function of sensor movement.

The effects due to varying the number of nodes are investigated

The impact of speed on the loss rate is shown in Figure 4.6. A range of robot diffusion rates are illustrated. The motion speed for the top trace is the highest (9 distance units per time unit) while the trace for the bottom one is the lowest (1 distance unit/per time unit). In Figure 4.7, Unit3 means moving at a rate of 3 distance units per time unit, and similarly for Unit6. Results for both single-tree (i.e., non-MIMO) and TwinsNet operational modes are shown for comparison purposes.

The impact of node count on loss rate was also studied, with our simulation results shown in Figure 4.7. As expected, the loss rate decreases for increasing node count. Moreover, the loss rate for a given rate of speed is less for the TwinsNet approach compared to a baseline single transmitter/receiver approach, particularly for the case where the robot-mounted sensors are moving at a high speed.

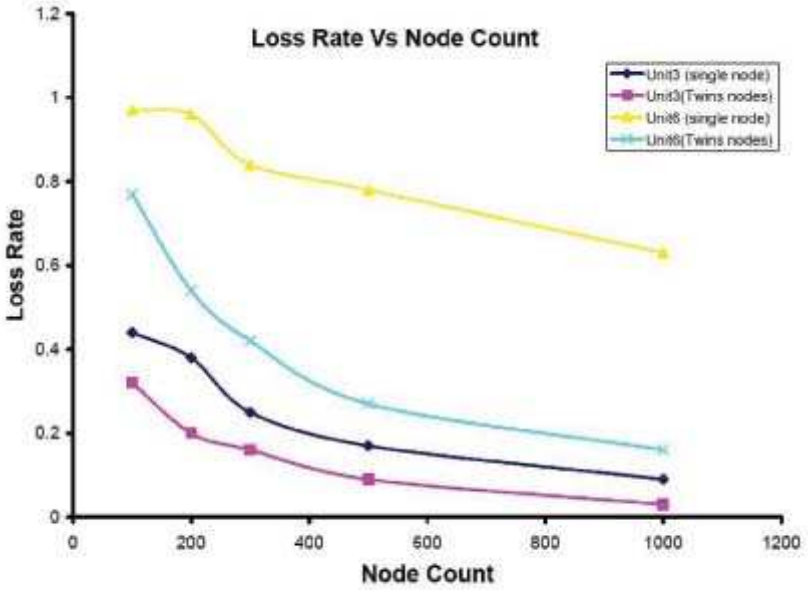

Figure 4.7: The loss rate as a function of node count.

Chapter 5

Gradient-Driven Target Acquisition in Mobile Wireless Sensor Networks

Navigation of mobile wireless sensor networks and fast target acquisition without a map are two challenging problems in search and rescue applications. In this chapter, we propose and evaluate a novel Gradient Driven method, called GraDrive. Our approach integrates per-node prediction with global collaborative prediction to estimate the position of a stationary target and to direct mobile nodes towards the target along the shortest path. We demonstrate that a high accuracy in localization can be achieved much faster than other random walk models without any assistance from stationary sensor networks. We evaluate our model through a light-intensity matching experiment in MicaZ motes, which indicates that our model works well in a wireless sensor network environment. Through simulation, we demonstrate almost a 40% reduction in the target acquisition time, compared to a

random walk model, while obtaining less than 2 unit error in target position estimation.

5.1 Introduction

Wireless sensor networks have gained extensive attention in many applications such as tracking, differentiated surveillance, and environment monitoring [83–85]. Moreover, the hybrid systems of mobile objects (e.g. Robots) and sensor networks create new frontiers for civilian and military applications, such as search and rescue missions in which the background environments are inaccessible to humans. A heterogeneous searching team consisting of robots and a wireless sensor network has greater advantage, considering its distributed computation and navigation capability achieved through the cooperation of embedded wireless sensor networks.

Although the applications of mobile sensor networks keep diversifying, several underlying capabilities remain fundamental and critical. In this work, we focus on the target acquisition – finding the locations of stationary targets using mobile sensor nodes. The challenging problem we address in this work is *to navigate a team of mobile sensor nodes toward the stationary targets fast and accurately while consuming the least amount of energy and other resources.* In this emerging research arena, most research groups employed static wireless sensor networks to navigate the mobile sensor nodes. Tan in [86] used distributed static sensor networks to collect the data and execute local calculations to generate a path for a mobile sensor network to move toward the goal. Although the in-network calculation implemented in that project was quite efficient in creating the shortest routing path, the additional requirement of a stationary distributed sensor network sets a barrier for rescue applications, because of the high cost to cover a large geographic area with a large number of sensors. Other research groups [87] proposed gradient methods in which the mobile

wireless sensor nodes move toward the gradient direction assuming that targets carried the most intensive strength of interested signals. However, in all of their implementations, the assistance of a stationary wireless sensor network was assumed to be available in generating a local signal distribution map. A probabilistic navigation algorithm is presented in [88], where a discrete probability distribution of vertex is introduced to point to the moving direction. This algorithm computes the utilities for every state and then picks the actions that yield a path toward the goal with maximum expected utility. The shortcoming of this method is that it requires the arrival of a mobile sensor node to localize the target position and significant communication overhead is introduced by the iteration process.

5.2 Contribution

In this chapter, we propose to compensate for those deficiencies by incorporating a prediction model of real-time processes into a mobile sensor network sensing and navigation architecture. We are interested in the mutually beneficial collaboration of the algorithms described above but seek to reduce the costs and provide faster target localization. *The novelty of our approach is the seamless integration of a per-node prediction model with a global prediction model.* The per-node prediction model guarantees that a mobile node can acquire the position of a target alone, while the global prediction significantly reduces the navigation overhead and time, if collaboration among the nodes is available. Specifically, the main contributions of our prediction model are:

- Our model provides more meaningful description of individual sensor readings in term of accuracy and confidence.

- Our model works with a single mobile sensor node as well as a swarm of mobile sensor

nodes. In the latter case, the sensor nodes have the ability to share local information in order to draw a global picture, which helps each sensor node to acquire the target along a significantly shorter path.

- The in-network prediction algorithm enables faster yet accurate target position acquisition: sensor nodes would be required to reach the target only when the model prediction is not accurate enough to satisfy the requirement with an acceptable confidence. This allows a significant reduction in navigation energy.

The remainder of this chapter is organized as follows. Section 5.3 defines the assumptions. Section 5.4 overviews the design. In Section 5.5, we present the in-network per-node prediction model. Section 5.6 describes target acquisition in the context of global prediction and the corresponding mobile sensor node navigation protocol. Section 5.7 presents empirical data obtained from the MICAZ platform as well as simulation results.

5.3 ASSUMPTIONS

Our design is based on two assumptions: network connectivity and the self-localization of mobile wireless sensors.

- **Connectivity:** First, wireless sensor nodes in the network are assumed to be able to ensure connectivity. Individual mobile sensor nodes deployed in large area is likely to lose connection to a central base station if the routing information is not updated. Therefore, it is desirable to maintain connections across a team of mobile sensor nodes while minimizing power consumption and allowing the sensor nodes to achieve their individual goals.

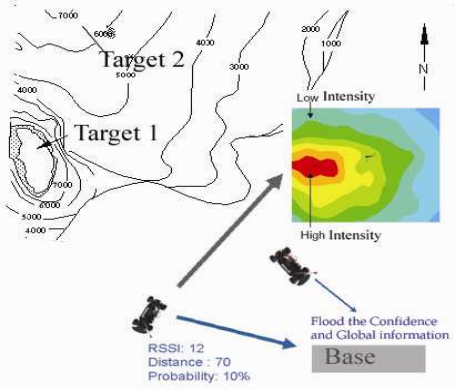

Figure 5.1: The Architecture schematic of GraDrive

- **Node Self-Localization:** The second assumption hinges on the localization availability for a mobile wireless sensor network. If a mobile sensor node enters an unknown area, it must be able to specify its own location dynamically without a map. This location can be obtained either through GPS such as used in ZebraNet [89] and VigilNet [85]. It can also use a dynamic localization scheme [90] that adjusts the estimated location of a node periodically based on the recent observed motion.

5.4 OVERVIEW OF PREDICTION MODEL

The objective of our GraDrive target acquisition scheme is to predict the location of stationary targets within allowable uncertainty (or a confidence level) dictated by a rescue plan. To illustrate the design of GraDrive, we start our description with a rescue scenario shown in Fig. 5.1. Here we note that our method is independent of this rescue application and can be applied in other scenarios as well.

- **Objective:** The control center (base) disseminates a search objective to a mobile sensor network with two parameters, *error tolerances* and *confidence level* of the target, specifying the quality of target acquisition. For example, the objective would

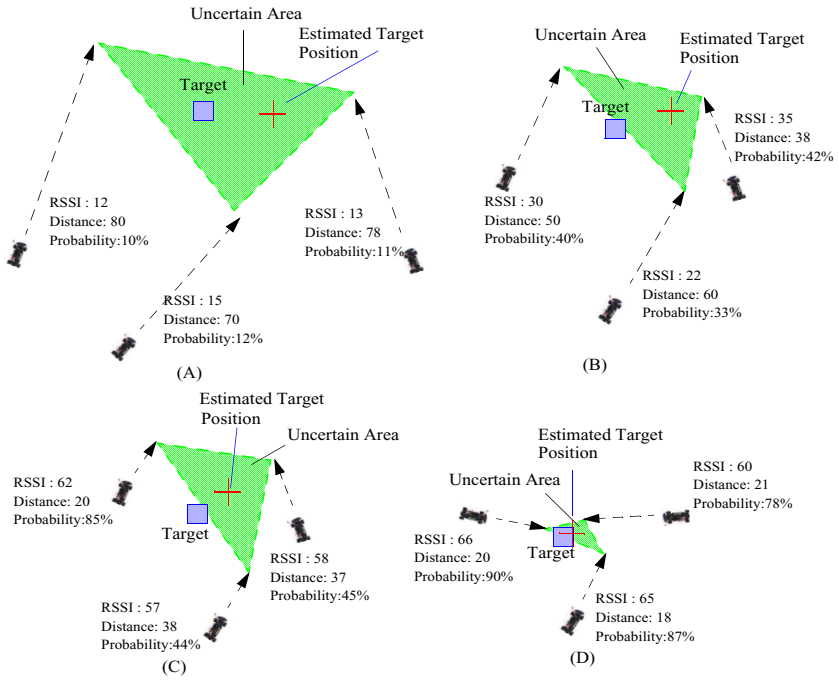

Figure 5.2: Collaborative Prediction Scheme of GraDrive

be locating a target within 2 meters with at least 95% confidence. The tolerance levels for each mobile sensor nodes can vary correspondingly in case different nodes are designed for different purposes.

- **Individual Prediction Model:** Once the search objectives are received by the mobile nodes, the nodes decide their most efficient way to locate the potential target with the requested confidence *individually*, using the per-node prediction model. A node start to move toward the direction in which it anticipates the fastest path to reach the confidence.

- **Collaborative Prediction Model:** In addition to its own plan and navigation, sensor nodes also report back to a base station, where all the individual nodes' readings and plans are collected and merged to create a global map and an uncertainty area. If computation results show probability increase by certain interval, e.g. 5% to its

previous state computation, the base station will disseminate the global prediction value over the network so that each sensor nodes in network can update their model. In other words, the prediction result based on collaborative information overrules the results from the individual prediction model.

As demonstrated in Fig. 5.2 from (A) to (D), as the individual sensor node continuously predicts the target position with increasing probability and moves toward the target, the uncertainty area where the target is located shrinks through collaboration among mobile sensor nodes. If the collaborative probability calculated reaches the dictated objective, a success of the rescue plan is achieved. The position it reports is the exact target position specified. Compared to other static sensor node navigation plans, the prediction results computed by our model still provide considerably more information than MobileRobot [88] and SafeRobot [91].

5.5 GRADRIVE MODEL DETAILS

In this section, we formally describe our per-node prediction model to estimate the position of a stationary target with a certain confidence. This per-node prediction model forms the basis for global collaborative prediction described in Section 5.6. Even though we consider an unknown area with multiple targets, the searching for separate targets is independent of each other as long as the field (RSSI) generated by one target doesn't overwhelm that generated by others. Therefore, in the remainder of this chapter, we focus on only the single target acquisition problem.

5.5.1 Prediction Problem Formulation

Conventionally, we begin with a value-prediction problem, which creates a Received Signal Strength Indicator $F(\theta)$ over a parameter set θ. For example, if $\theta = (d, t, v)$, RSSI is related to d, the distance between a mobile sensor node and the target, t, the time of sampling, and v, the speed of mobile sensor nodes. This model can be established by getting consecutive sensing readings (system states) when a mobile sensor node moves. Typically, the number of parameters in θ is much less than the number of states collected and changing one parameter changes the estimated value of many states. To approximate our model appropriately, we seek to minimize the mean squared error over some distribution, P, of the inputs.There are generally far more states than components in θ. The flexibility of the function estimator is thus a scarce resource. Better approximation at some states can be gained generally only at the expense of worse approximation at other states.

5.5.2 Distance Prediction Model

In GraDrive, we extend the familiar one-dimensional normal probability density function known as Gaussian distribution to two variants multidimensional distribution. The predicted distance from sensor nodes' current position to predicted target position can be queried or estimated from the model. The multidimensional Gaussian distribution function over two attributes, trust interval and RSSI, can be expressed as a function of two parameters: a 2-tuple vector of means, μ, and a 2×2 matrix of covariances, Σ. Further, we assume the trust interval set by a rescue team is independent of the RSSI received, which means the trust interval of the predicted distance estimation T_i to the mean of historic results μ doesn't change dynamically along the searching process. The two dimensional distribution can be separated for description purposes. Without loss of generality, it is assumed that the

predicted distance d is disproportional to RSSI, that can be expressed as $d = r_1/RSSI + r_2$, where r_1 and r_2 are two adaptive parameters that can be determined before the searching process. We note here other RSSI fading models can be used here as well without invaliding our approach. We then use historical data or experience data to construct the models, providing r_1 and r_2 at each RSSI value appropriately. Besides offering the predicted distance, a probability model, which is a function of the d, is also constructed to provide confidence of the prediction, e.g. given a predicted distance of 2 feet, the confidence for this prediction is 95%. The model must be trained before it can be used, which is a general limitation for a probabilistic model. The accuracy of the model, therefore, relies on the accuracy of the data used to train it. Once the initial model is constructed, each sensor nodes can query the predicted distance map from the stored model and come up with a confidence value. One distribution of the distance d against the confidence p over one RSSI is a Gaussian distribution. Suppose that rescue team have set a trust interval of T_i, given the distribution of distance over one RSSI, we can get the points d_i which satisfy that $P(d_i) - P(u) <= T_i$. Here, we emphasizes that if the trust interval is too small, the amount of data needed to train the model will increase exponentially.

5.5.3 Signal Strength Distribution Prediction Model

Besides obtaining the distance d information based on measured $RSSI$, we can further refine the RSSI distribution model. This distribution model can then be used to navigate the mobile sensor network toward the target along a shortest path. The central element in our approach is to construct a prediction model that represents attributes as accurately as possible in a mobile sensor network. As we discuss above, if the predicted RSSI distribution function depends on parameters including distance d and confidence or probability p, the function can be expressed as $F(d, p)$ considering the d and p's distributions are independent.

If we do a Taylor expansion on the function F, a polynomial function of attributes d and p is achieved, shown as

$$F(d,p) = f(d_0, d_1, d_2...)f(p) \tag{5.1}$$

where d_i is a function of the distance variable d. To reduce the computation energy consumption, only a second order polynomial is considered in our case, which offers a 3-tuple vector $D = [d_0, d_1, d_2]$:

$$\begin{aligned} d_0 &= \mathrm{c}_0 \\ d_1 &= 1/(d + c_1) \\ d_2 &= 1/(d^2 + c_2) \end{aligned} \tag{5.2}$$

where c_1, c_2, c_3 are constants used to avoid a singularity when $d = 0$. Now we can define our gradient distribution function in a simple form as:

$$F = D \bullet A \bullet p \;\; where \; A = [a_0, a_1, a_2] \tag{5.3}$$

Equation 5.3 is our probabilistic gradient distribution prediction function for the attributes of d and p. Suppose that each sensor node observes the value of attribute D_j to be d_j, the sensing observations are input into a vector of D_j. Thus, the vector D is extended to a matrix.

If enough sensing samplings are provided, we can apply non-linear least square fitting to estimate the parameters A. For a nonlinear least square fitting to our undetermined parameters, linear least squares fitting may be applied iteratively to a linearized form of the function until convergence is achieved. Since we can anticipate a power type of the fit and

have decided the initial parameters chosen for our model, the nonlinear fitting has good convergence properties.

In general, the computation of the matrix does cost a large amount of the wireless nodes' energy. The solution in GraDrive is to simplify the prediction distribution function as above, given that the prediction function computation can be distributed over the network with collaboration of its neighbors, or the data can be delivered back to a base station where stronger computation ability and energy are normally not limitations. If this is the case, the base station creates a gradient distribution map globally using a weighted average method as a function of probability and the predicted distribution. This kind of global information is sent back to each individual node involved in the application.

5.6 TARGET LOCALIZATION USING THE COLLABORATIVE PREDICTION MODEL

Based on the per-node prediction model, the mobile sensor nodes can infer the position of a target (x, y) and the associated confidence value p. This information is then used to perform global predictions. Specifically, we propose to use a probability-weighted average model for global collaborative prediction, due to its high efficiency and low cost characteristics. The simple rational behind our method is that the sensor nodes having a higher probability are much closer to the intended target.

Suppose if the predicted target location provided by sensor nodes $n_1, n_2, ..., n_k$, are $(x_1, y_1), (x_2, y_2), ..., (x_k, y_k)$ associated with probability value $p_1, p_2, ...p_k$. The estimated position of the target is given as:

$$X = \frac{\sum_{i=1}^{k} p_k x_k}{\sum_{i=1}^{k} p_k} \quad Y = \frac{\sum_{i=1}^{k} p_k y_k}{\sum_{i=1}^{k} p_k} \tag{5.4}$$

5.6.1 Collaborative Navigation and Prediction Protocol

With per-node and global prediction models established, we are now ready to describe how the sensor nodes navigate using these two models.

Initially, sensor nodes enter an intended region with certain moving speeds, moving directions and trust intervals. It should be noted that different mobile sensor nodes could have different moving speed or initial moving direction. After the entrance, the mobile sensor nodes continue to detect the RSSI in their sensing range. The detected RSSI readings are an important input for training the model. Thus, they use a default navigation plan, which is to keep moving forward unless they detect a smaller sensing reading. During the moving process, nodes themselves perform a per-node prediction calculation to construct the local RSSI map as described in Section 5.5.3. Meanwhile, the sensor nodes estimate their distance to the target position according to the RSSI. The predicted target location information is forwarded back to a base station. To prevent excessive energy consumption in communication, the frequency of updates can be specified in advance. As long as the global picture is not available, individual sensor nodes navigate according to the per-node prediction model. However, if the base station notifies the sensor nodes that it has constructed a global RSSI distribution with a certain confidence, each sensor node will combine the information with its current model together and change its direction toward the gradient direction. This process will be repeated until the target position has been discovered locally or at the base station within acceptable confidence.

5.6.2 Default Navigation Plan when Global Prediction Unavailable

If initially there is no global picture constructed by the base station with acceptable confidence, or if there is only one isolated node in the network for a rescue plan, or if the network is partitioned or unable to deliver the data, the mobile sensor nodes fall back to the per-node prediction model. Given its current sensing reading, it compares with previous readings stored in its memory at each motion step. Upon getting a smaller sensing reading, it rotates 90 degrees clockwise. The reason for this is that the target position is most likely located perpendicularly to its direction of motion.

5.7 EXPERIMENTS AND SIMULATION

5.7.1 Model Matching Experiment

In order to verify the feasibility of the proposed prediction model and parameter-fitting algorithms, we have prototyped a light sensing system based on Berkeley MICAZ motes. Even though it is stationary, the prediction model and parameter-fitting algorithms can still be verified at the base station site which can be transferred to individual sensor nodes. Light signal strength is used as an example of RSSI to feed the model. One laptop equipped with motherboard acts as the base station. A lamp works as a target and a series of sensor nodes are deployed as shown in Fig. 5.3. The sensor nodes detect the sensing reading and exchange the readings to their neighbors. The base station calculates the parameters for the sensor nodes by using the least square fitting method. Fig. 5.4 shows one set of data fitted by the prediction model. The distance between two adjacent sensor nodes is equal and unified for matching purposes. Since the received signal strength is not an accurate measurement, the probability approximation model comes into play. From the matching

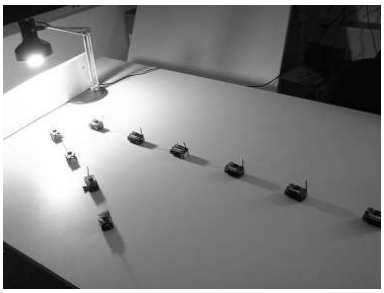

Figure 5.3: Model fitting experiment with light as source of signal and using Micaz nodes in array to sense the signal strength

results, it is shown that the least square method tries to reduce the deviation among the sensing data collected. Other sets of data can also be collected and used to train the model.

5.7.2 Simulation Setup

We have developed a program to verify the advantage of using our prediction model to locate the target. In our simulation, a 200×200 m^2 area is regarded as an unknown space with a target located at the center and a distribution along the diameter is defined. Essentially, it could be any random distribution having a gradient toward the center. Each distance unit is represented as the smallest unit that the mobile sensor nodes can travel each time during simulation. The navigation algorithm is used to simulate the mobility of objects. Initially, the mobile nodes are located at the edges of the area. The initial direction is randomly picked by each mobile sensor node. If some sensor nodes move outside the simulation region, they bounce their moving direction back into simulation area. Under simulation, each mobile sensor node moves at a constant speed in integer multiples of 1m/s. After each time unit (1 second in our case), a node determines its next moving direction according to our algorithm.

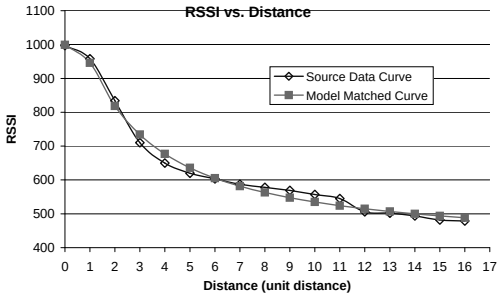

Figure 5.4: The predicted model with real sensing data

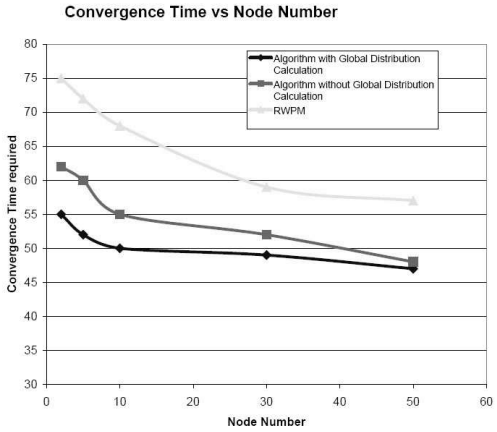

Figure 5.5: Convergence time with node number for different models

5.7.3 Delay in Target Acquisition

We first compare our algorithm (without the global distribution calculation option) against a random way point method. The simulation results (Fig. 5.5) suggest that even without a global distribution calculation mode turning on, our default algorithm (rotating 90 degrees counterclockwise) still provides 30% faster estimation than the random way method. If global calculation mode is on, then initially the sensor nodes still use default plan, but if the global signal strength distribution is available, it moves faster than the default algorithm.

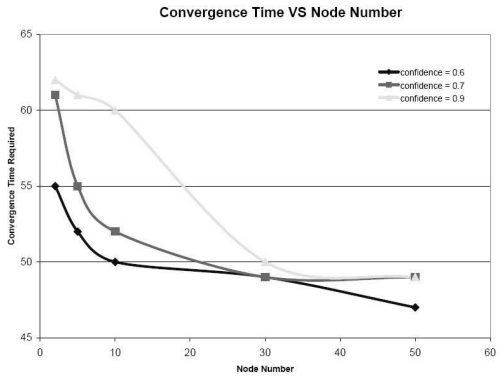

Figure 5.6: Convergence time with Node number under different required confidence level

5.7.4 Impact of the Confidence p

We also compare the impact of different required confidence levels on the convergence time, as shown in Fig. 5.6. It is clear that if the required confidence level goes beyond 90%, it will take much longer to simulate simply because it requires at least 2 nodes to get very close to target position. It is reasonable to choose a relatively high confidence level e.g. 80% in order to balance accuracy and time cost.

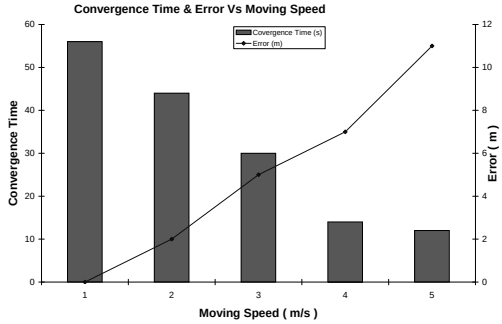

Figure 5.7: Convergence time and accuracy with different speeds of mobile sensor nodes

5.7.5 Impact of the Target Speed

In Fig. 5.7, we further investigate the relationship between the speed of sensor nodes and prediction accuracy of the target location. The convergence time correlated directly with

the speed of the movement of each sensor node since the average time for sensor nodes to get closer to the target is reduced. However, the accuracy of prediction gets worse if the speed increases because the minimum deviation for the prediction is increased as well. Therefore, the error continues to grow in the prediction as a node moves faster from its original location. In the situation of high speed, an error larger than 10 units is shown. To protect against inaccuracies in the prediction model of mobile sensor nodes, a user must set a limit for the speed of sensor nodes.

Chapter 6

Conclusion and Future Work

This thesis has presented a set of collaborative sensor scheduling algorithms and techniques for the energy-efficient implementation of wireless sensor networks in applications such as environmental monitoring and target acquisition.

First, we have presented a stochastic sensing algorithm to reduce energy consumption. Our approach does not require powerful computational ability at the sensor nodes in order to construct an accurate data prediction model. Observed correlations between data sensing cycles have been used to estimate the prediction error, thus allowing the scheduler to adjust its operation accordingly. The measurement and simulation results show that the system prediction error remains within the specified error tolerance while saving up to 70 percent of the required energy.

A collaborative error control mechanism has also been developed, which enables sensor nodes to implement a two-tier error control scheme. This approach enhances a sensor's capability for detecting rare events while still reducing energy consumption.

We have also investigated the application of virtual MIMO techniques within the context of a mobile, robotic sensor network to improve the communications performance in search

and rescue applications. Mobile units which are physically adjacent to each other are paired up so that they can transmit their sensor data in a MIMO fashion. Furthermore, other intermediate relaying nodes may move to locations which minimize the required transmission energy to send the information back to the base station.

Finally, we have also presented a probabilistic prediction model for dynamic target localization. Our model does not require the use of a map to determine the positions of potential targets. Also the proposed gradient-driven algorithm leads to a 40% reduction in the time for localization compared to that of a random walk model. The relationship between sensor density and convergence time can also be used to plan the implementation of such a mobile sensor network. For future work, we would like to implement our algorithm in hardware using off-the-shelf components.

We also note that the individual solutions presented in this thesis have each been optimized for a particular problem domain. Future research efforts may be directed towards integrating some or all of these separate approaches into a general, unified framework which would be applicable to a wide range of sensing applications.

Bibliography

[1] J. M. Huang, X.-M. and L. Leblanc, "Wireless sensor network for streetlight monitoring and control," in *SIGMOD*, April 2004.

[2] T. Imielinski and S. Goel, "DataSpace - Querying and Monitoring Deeply Networked Collections in Physical Space," in *Proc. of the Intl. Workshop on Data Engineering for Wireless and Mobile Access (MobiDE'99)*, August 1999.

[3] A. Cerpa, J. Elson, D. Estrin, L. Girod, M. Hamilton, and J. Zhao, "Habitat Monitoring: Application Driver for Wireless Communications Technology," in *Proc. of the 2001 ACM SIGCOMM Workshop on Data Communications in Latin America and the Caribbean*, April 2001.

[4] V. Kottapalli, A. Kiremidjian, J. Lynch, E. Carryer, T. Kenny, K. Law, and Y. Lei, "Two-tiered Wireless Sensor Network Architecture for Structural Health Monitoring," in *Proc. of the Intl. Symp. on Smart Structures and Materials*, March 2003.

[5] T. He, C. Huang, B. M. Blum, J. A. Stankovic, and T. F. Abdelzaher, "Range-free localization and its impact on large scale sensor networks," 2005.

[6] D. Estrin, L. Girod, G. Pottie, and M. Srivastava, "Instrumenting the World with Wireless Sensor Networks," in *Proc. of Intl. Conference on Acoustics, Speech and Signal Processing (ICASSP)*, May 2001.

[7] I. Solis and K. Obraczka, "in-network aggregation trade-offs for data collection in wireless sensor networks," *International Journal of Sensor Networks*, 2006.

[8] S. Madden, M. Franklin, J. Hellerstein, and W. Hong, "TAG: A Tiny Aggregation Service for Ad-Hoc Sensor Networks," in *Operating Systems Design and Implementation*, December 2002.

[9] M. Lee and V. Wong, "An energy-aware spanning tree algorithm for data aggregation in wireless sensor networks," Victoria, BC, Canada, 2005.

[10] A. Sinha, A.; Chandrakasan, "Toward optimal data aggregation in random wireless sensor networks," in *INFOCOM 2007*.

[11] D. Kramarev, S. Bae, and K. Kim, "Empirical data fusion for decentralized detection in sensors systems with sequential structure," 2006.

[12] Q. Qiu, Q. Wu, D. Burns, and D. Holzhauer, "Lifetime aware resource management for sensor network using distributed genetic algorithm," Tegernsee, Germany, 2006//, pp. 191 – 6.

[13] A. Jain, E. Chang, and Y. Wang, "Adaptive stream resource management using kalman filters," 2004.

[14] S. Pattem, B. Krishnmachari, and R. Govindan, "The impact of spatial correlation on routing with compression in wireless sensor networks," in *IPSN*, 2004.

[15] R. Cristescu, B. Beferull-Lozano, and M. Vetterli, "On network correlated data gathering," 2004.

[16] C. L. Y.-X. H. N. Xiong, "An energy-efficient dynamic power management in wireless sensor networks," in *Parallel and Distributed Computing, 2006. ISPDC06.*

[17] C. L. A. M. R. Passos, R.M.; Coelho, "Dynamic power management in wireless sensor networks: An application-driven approach," in *Wireless On-demand Network Systems and Services, 2005. WONS 2005.*

[18] H. C. S. K. Wang, "An adaptive hybrid dynamic power management method," in *Sensor Networks, Ubiquitous, and Trustworthy Computing, 2006. IEEE International Conference on.*

[19] A. S. J. Liu, H.; Chandra, "esense: energy efficient stochastic sensing framework for wireless sensor platforms," in *IPSN 2006.*

[20] D. Puccinelli and M. Haenggi, "Wireless sensor networks: applications and challenges of ubiquitous sensing," *Circuits and Systems Magazine, IEEE*, vol. 5, no. 3, pp. 19–31, 2005.

[21] S. Krco, V. Tsiatsis, K. Matusikova, M. Johansson, I. Cubic, and R. Glitho, "Mobile network supported wireless sensor network services," *Mobile Adhoc and Sensor Systems, 2007. MASS 2007. IEEE Internatonal Conference on*, pp. 1–3, Oct. 2007.

[22] G. Crosby and N. Pissinou, "Evolution of cooperation in multi-class wireless sensor networks," *Local Computer Networks, 2007. LCN 2007. 32nd IEEE Conference on*, pp. 489–495, Oct. 2007.

[23] P. A. Morreale, "Wireless sensor network applications in urban telehealth," *Advanced Information Networking and Applications Workshops, 2007, AINAW '07. 21st International Conference on*, vol. 2, pp. 810–814, May 2007.

[24] A. Mainwaring, J. Polastre, R. Szewczyk, D. E. Culler, and J. Anderson, "Wireless Sensor Networks for Habitat Monitoring," in *Proc. of the ACM Workshop on Sensor Networks and Application (WSNA)*, September 2002.

[25] H. Liu, X. Jia, P.-J. Wan, C.-W. Yi, S. Makki, and N. Pissinou, "Maximizing lifetime of sensor surveillance systems," *Networking, IEEE/ACM Transactions on*, vol. 15, no. 2, pp. 334–345, April 2007.

[26] B. Yin, H. Shi, and Y. Shang, "A two-level strategy for topology control in wireless sensor networks," vol. Vol. 2, Fukuoka, Japan, 2005//, pp. 358 – 62.

[27] M. Cheng and L. Yin, "Transmission scheduling in sensor networks via directed edge coloring," *Communications, 2007. ICC '07. IEEE International Conference on*, pp. 3710–3715, June 2007.

[28] R. Szewczyk, A. Mainwaring, J. Anderson, and D. Culler, "An Analysis of a Large Scale Habit Monitoring Application," in *SenSys'04*, 2004.

[29] M. Dong, L. Tong, and B. Sadler, "Information retrieval and processing in sensor networks: Deterministic scheduling versus random access," *Signal Processing, IEEE Transactions on*, vol. 55, no. 12, pp. 5806–5820, Dec. 2007.

[30] S. Balasubramanian and D. Aksoy, "Adaptive online scheduling for asymmetric wireless sensor networks," *Computer Networks, 2006 International Symposium on*, pp. 73–78, June 2006.

[31] J.-J. Xiao, S. Cui, Z.-Q. Luo, and A. Goldsmith, "Power scheduling of universal decentralized estimation in sensor networks," *Signal Processing, IEEE Transactions on*, vol. 54, no. 2, pp. 413–422, Feb. 2006.

[32] V. Mansouri, M. MohammadNia-Awal, Y. Ghiassi-Farrokhfal, and B. Khalaj, "Dynamic scheduling mac protocol for large scale sensor networks," *Mobile Adhoc and Sensor Systems Conference, 2005. IEEE International Conference on*, pp. 5 pp.–, Nov. 2005.

[33] Y. Massad, M. Goyeneche, J. Astrain, and J. Villadangos, "Data aggregation in wireless sensor networks," *Information and Communication Technologies: From Theory to Applications, 2008. ICTTA 2008. 3rd International Conference on*, pp. 1–6, April 2008.

[34] S. Puthenpurayil, R. Gu, and S. Bhattacharyya, "Energy-aware data compression for wireless sensor networks," *Acoustics, Speech and Signal Processing, 2007. ICASSP 2007. IEEE International Conference on*, vol. 2, pp. II–45–II–48, April 2007.

[35] H. Liu, P. Wan, and X. Jia, "Maximal lifetime scheduling for sensor surveillance systems with k sensors to one target," *Parallel and Distributed Systems, IEEE Transactions on*, vol. 17, no. 12, pp. 1526–1536, Dec. 2006.

[36] M. Marques, L. Santos, and I. Bonatti, "An exponential energy saving strategy to wireless sensor network," *Sensor Technologies and Applications, 2007. SensorComm 2007. International Conference on*, pp. 355–360, Oct. 2007.

[37] R. Zhang, D. Yuan, and Q. Liang, "Analysis of energy consumption and lifetime of wireless sensor networks," *Industrial Electronics and Applications, 2007. ICIEA 2007. 2nd IEEE Conference on*, pp. 1159–1164, May 2007.

[38] S. Goel and T. Imielinski, "Prediction-based Monitoring in Sensor Networks: Taking Lessons from MPEG," *ACM Computer Communication Review*, vol. 31, no. 5, October 2001.

[39] S. Moon, T. Kim, and H. Cha, "Enabling low power listening on ieee 802.15.4-based sensor nodes," Kowloon, China, 2007, pp. 2307 – 2312.

[40] J. Polastre, J. Hill, and D. Culler, "Versatile low power media access for wireless sensor networks," Baltimore, MD, United States, 2004, pp. 95 – 107.

[41] W. Gu, X. Bai, S. Chellappan, D. Xuan, and W. Jia, "Network decoupling: a methodology for secure communications in wireless sensor networks," *IEEE Transactions on Parallel and Distributed Systems*, vol. 18, no. 12, pp. 1784 – 96, 2007/12/.

[42] B. Matt and M. Mundy, "Designing efficient and resilient tactical sensor network neighborhood keying algorithms," Nassau Inn, Princeton, NJ, USA, 2007//, pp. 1 – 7.

[43] Q. Zhang, Y. Gu, T. He, and G. Sobelman, "Cscan: A correlation-based scheduling algorithm for wireless sensor networks," *Networking, Sensing and Control, 2008. ICNSC 2008. IEEE International Conference on*, pp. 1025–1030, April 2008.

[44] T. H. Qingquan Zhang, Yu Gu and G. Sobelman, "Ebs, an error bounded scheduling algorithm for sensor networks," in *Technical Reports*, University of Minnesota,Twin Cities, MN, United States, 2008.

[45] "Wisconsin-minnesota cooperative extension agricultural weather page," in *http://www.soils.wisc.edu/wimnext/*, University of Minnesota,Twin Cities, MN, United States, 2008.

[46] M. R. G.T. Sibley and G. Sukhatme, "Robomote: A Tiny Mobile Robot Platform for Large-Scale Ad-hoc Sensor Networks," in *Proceedings of the IEEE Conf. on Robotics and Automation*, 2002.

[47] "Development of mobile inspection robot for rescue activities: Moira, author= K. Osuka and H. Kitajima, journal= Proceedings of the Conf. on Intelligent Robots and Systems, pages=3373-3377, year=2003,."

[48] M. Gerla and K. Xu, "Multimedia streaming in large-scale sensor networks with mobile swarms," *SIGMOD Rec.*, vol. 32, no. 4, pp. 72–76, 2003.

[49] A. Jacoff and E. Messina, "Urban search and rescue robot performance standards: progress update," vol. 6561, Orlando, FL, USA, 2007/04/27, pp. 65 611 – 1. [Online]. Available: http://dx.doi.org/10.1117/12.719692

[50] J. Scholtz, B. Antonishek, and J. Young, "A field study of two techniques for situation awareness for robot navigation in urban search and rescue," Nashville, TN, USA, 2005//, pp. 131 – 6.

[51] A. Wolf, H. Choset, J. Brown, H.B., and R. Casciola, "Design and control of a mobile hyper-redundant urban search and rescue robot," *Advanced Robotics*, vol. 19, no. 3, pp. 221 – 48, 2005//.

[52] J. Borenstein, H. Everett, and L. Feng, *Navigating Mobile Robots: Systems and Techniques*. A. K. Peters, Ltd., Wellesley, 1996.

[53] W. Green and P. Oh, "An Aerial Robot Prototype for Situational Awareness in Closed Quarters," in *Proceedings of the IEEE Conf. on Intelligent Robots and Systems*, 2003, pp. 61–66.

[54] J. Caspa, "Human-Robot Interactions during the Robot-Assisted Urban Search and Rescue Response at the World Trade Center," in *Computer Science and Engineering Report,University of South Florida*, 2002.

[55] W. Burgard, M. Moors, D. Fox, R. Simmons, and S. Thrun, "Collaborative Multi-Robot Exploration," in *In Proceeding of IEEE International Conferenceon Robotics and Automation (ICRA)*, 2000.

[56] J. M. Kahn, R. H. Katz, and K. S. J. Pister, "Next Century Challenges: Mobile Networking for Smart Dust," in *Proc. of Intl. Conference on Mobile Computing and Networking (MOBICOM)*, August 1999.

[57] A. Woo, T. Tong, and D. Culler, "Taming the Underlying Challenges of Reliable Multihop Routing in Sensor Networks," in *First ACM Conference on Embedded Networked Sensor Systems (SenSys 2003)*, November 2003.

[58] P. Pathirana and et al., "Node localization using mobile robots in delay-tolerant sensor networks," *IEEE Transactions on Mobile Computing*, vol. 4, no. 3, 2005.

[59] S. Cui and A. Goldsmith, "Energy efficient routing based on cooperative mimo techniques," *Acoustics, Speech, and Signal Processing, 2005. Proceedings. (ICASSP '05). IEEE International Conference on*, vol. 5, pp. v/805–v/808 Vol. 5, March 2005.

[60] T. Rissa, A. Donlin, and W. Luk, "Evaluation of systemc modelling of reconfigurable embedded systems," *Design, Automation and Test in Europe, 2005. Proceedings*, pp. 253–258 Vol. 3, March 2005.

[61] H. Patel and S. Shukla, "Model-driven validation of systemc designs," *Design Automation Conference, 2007. DAC '07. 44th ACM/IEEE*, pp. 29–34, June 2007.

[62] F. F. A. Fin and D. Signoretto, "The use of systemc for design verification and integration test of ip-cores," *presented at Proceedings 14th Annual IEEE International ASIC/SOC Conference*, Sept. 2001.

[63] G. Martin, "Systemc: from language to applications, from tools to methodologies," *presented at Proceedings 16th Symposium on Integrated Circuits and Systems Design*, 2003.

[64] M. O. Hasna and M. Alouini, "A performance study of dual-hop transmissions with fixed gain relays," *IEEE Trans. on Wireless Communications, vol. 3, no. 6, page=1963-1968,year= November 2004,.*

[65] D. N. C. T. J. N. Laneman and G. W. Wornell, "Cooperative diversity in wireless networks: Efficient protocols and outage behavior," *IEEE Trans. on Information Theory, vol. 50, no. 12*, December 2004.

[66] J. N. Laneman and G. W. Wornell, "Energy-efficient antenna sharing and relaying for wireless networks," *in Proc. of Wireless Communications and Networking Conf., vol. 1, Chicago, IL*, September 23-28, 2000.

[67] E. E. A. Sendonaris and B. Aazhang, "User cooperation diversity, part i: System description," *IEEE Trans. on Communications, vol. 51, no. 11*, November 2003.

[68] D. Chen and J. N. Laneman, "Cooperative diversity for wireless fading channels without channel state information," *in Proc. of Asilomar Conf. on Signals, Systems, and Computers, Monterey, CA*, November 7-10, 2004.

[69] H. M. P. Tarasak and V. K. Bhargava, "Differential modulation for two-user cooperative diversity systems," *IEEE Journal on Selected Areas in Communications, vol. 23, no. 9*, September 2005.

[70] Q. Zhao and H. Li, "Performance of a differential modulation scheme with wireless relays in rayleigh fading channels," *in Proc. of Asilomar Conf. on Signals, Systems, and Computers, vol. 1, pp. 1198-1202,year= November 7-10, 2004,.*

[71] S. Yiu and R. Schober, "Censored noncoherent distributed space-time coding for wireless sensor networks," *Wireless Communications and Networking Conference, 2007.WCNC 2007. IEEE*, pp. 600–605, March 2007.

[72] W. Cho and L. Yang, "Differential modulation schemes for cooperative diversity," *in Proc. of IEEE International Conference on Networking, Sensing and Control, Ft. Lauderdale, FL*, April,2006.

[73] C. w. and L. Yang, "Distributed differential schemes for cooperative wireless networks," *in Proc. of Intl. Conf. on ASSP, Toulouse, France*, pp. 15–19, May 2006.

[74] Y. Yao and M. Howlader, "Multiple symbol double differential space-time coded ofdm," *Vehicular Technology Conference, 2002. VTC Spring 2002. IEEE 55th*, vol. 2, pp. 1050–1054 vol.2, 2002.

[75] Y. Jing and H. Jafarkhani, "Distributed differential space-time coding for wireless relay networks," *IEEE Transactions on Communications*, vol. 56, no. 7, pp. 1092–1100, July 2008.

[76] M. K. P. A. Anghel and Z. Q. Luo, "Optimal relayed power allocation in interfernce-free non-regenerative cooperative systems," *in Proc. of Signal Proc. Workshop on Advances in Wireless Communications, Lisbon, Portugal*.

[77] X. Deng and A. M. Haimovich, "Power allocation for cooperative relaying in wireless networks," *IEEE Communications Letters, vol. 9, no. 11*, November 2005.

[78] M. O. Hasna and M. Alouini, "Optimal power allocation for relayed transmissions over rayleigh-fading channels," *IEEE Trans. on Wireless Communications, vol.3, no. 6, pp. 1999-2004,year= November 2004,*.

[79] Y. Liang and V. V. Veeravalli, "Gaussian orthogonal relay channels: Optimal resource allocation and capacity," *IEEE Trans.on Information Theory, vol. 51, no. 9, pp. 3284-3289,year= September 2005,*.

[80] S. Swan, "Systemc transaction level models and rtl verification," *Design Automation Conference, 2006 43rd ACM/IEEE*, pp. 90–92, July 2006.

[81] A. Bernstein, M. Burton, and F. Ghenassia, "How to bridge the abstraction gap in system level modeling and design," *Computer Aided Design, 2004. ICCAD-2004. IEEE/ACM International Conference on*, pp. 910–914, Nov. 2004.

[82] L. Cai and D. Gajski, "Transaction level modeling: an overview," *Hardware/Software Codesign and System Synthesis, 2003. First IEEE/ACM/IFIP International Conference on*, pp. 19–24, Oct. 2003.

[83] A. Arora, P. Dutta, S. Bapat, V. Kulathumani, H. Zhang, V. Naik, V. Mittal, H. Cao, M. Demirbas, M. Gouda, Y. Choi, T. Herman, S. Kulkarni, U. Arumugam, M. Nesterenko, A. Vora, and M. Miyashita, " A Wireless Sensor Network for Target Detection, Classification, and Tracking," *Computer Networks (Elsevier)*, 2004.

[84] J. Liu, J. Reich, and F. Zhao, "Collaborative In-Network Processing for Target Tracking," *J. on Applied Signal Processing*, March 2003.

[85] T. He, S. Krishnamurthy, J. A. Stankovic, and T. Abdelzaher, "An Energy-Efficient Surveillance System Using Wireless Sensor Networks," in *MobiSys'04*, June 2004.

[86] J. Tan, A. Verma, and H. Sawant, "Selection and navigation of mobile sensor nodes using a sensor network," 2006.

[87] I. Chatzigiannakis, S. Nikoletseas, and P. Spirakis, "Distributed communication algorithms for ad hoc mobile networks," *J. Parallel Distrib. Comput.*, vol. 63, no. 1, pp. 58–74, 2003.

[88] M. Batalin, G. Sukhatme, and M. Hattig, "Mobile robot navigation using a sensor network," in *IEEE International Conference on Robotics and Automation*, 2004.

[89] P. Juang, H. Oki, Y. Wang, M. Martonosi, L. Peh, and D. Rubenstein, "Energy-Efficient Computing for Wildlife Tracking: Design Tradeoffs and Early Experiences with ZebraNet," in *Proc. of ASPLOS-X*, October 2002.

[90] S. Tilak and et al., "Dynamic localization control for mobile sensor networks," in *Conference Proceedings of the 2005 IEEE International Performance, Computing and Communications Conference*, 2005.

[91] H. H. Gonzalez-Banos and J. C. Latombe, "Robot navigation for automatic model construction using safe regions," in *ISER '00: Experimental Robotics VII*. London, UK: Springer-Verlag, 2001, pp. 405–415.