ABSTRACT

PROBABILISTIC APPROACHES TO THE DESIGN OF WIRELESS AD HOC AND SENSOR NETWORKS

by

Zhen Guo

The emerging wireless technologies has made ubiquitous wireless access a reality and enabled wireless systems to support a large variety of applications. Since the wireless self-configuring networks do not require infrastructure and promise greater flexibility and better coverage, wireless ad hoc and sensor networks have been under intensive research. It is believed that wireless ad hoc and sensor networks can become as important as the Internet. Just as the Internet allows access to digital information anywhere, ad hoc and sensor networks will provide remote interaction with the physical world.

Dynamics of the object distribution is one of the most important features of the wireless ad hoc and sensor networks. This dissertation deals with several interesting estimation and optimization problems on the dynamical features of ad hoc and sensor networks. Many demands in application, such as reliability, power efficiency and sensor deployment, of wireless ad hoc and sensor network can be improved by mobility estimation and/or prediction. In this dissertation, we study several random mobility models, present a mobility prediction methodology, which relies on the analysis of the moving patterns of the mobile objects. Through estimating the future movement of objects and analyzing the tradeoff between the estimation cost and the quality of reliability, the optimization of tracking interval for sensor networks is presented. Based on the observation on the location and movement of objects, an optimal sensor placement algorithm is proposed by adaptively learn the dynamical object distribution. Moreover,

dynamical boundary of mass objects monitored in a sensor network can be estimated based on the unsupervised learning of the distribution density of objects.

In order to provide an accurate estimation of mobile objects, we first study several popular mobility models. Based on these models, we present some mobility prediction algorithms accordingly, which are capable of predicting the moving trajectory of objects in the future. In wireless self-configuring networks, an accurate estimation algorithm allows for improving the link reliability, power efficiency, reducing the traffic delay and optimizing the sensor deployment. The effects of estimation accuracy on the reliability and the power consumption have been studied and analyzed. A new methodology is proposed to optimize the reliability and power efficiency by balancing the trade-off between the quality of performance and estimation cost. By estimating and predicting the mass objects' location and movement, the proposed sensor placement algorithm demonstrates a significant improvement on the detection of mass objects with near-maximal detection accuracy. Quantitative analysis on the effects of mobility estimation and prediction on the accuracy of detection by sensor networks can be conducted with recursive EM algorithms. The future work includes the deployment of the proposed concepts and algorithms into real-world ad hoc and sensor networks.

PROBABILISTIC APPROACHES TO THE DESIGN OF WIRELESS AD HOC AND SENSOR NETWORKS

by
Zhen Guo

A Dissertation
Submitted to the Faculty of
New Jersey Institute of Technology
in Partial Fulfillment of the Requirements for the Degree of
Doctor of Philosophy in Computer Engineering

Department of Electrical and Computer Engineering

January 2006

APPROVAL PAGE

PROBABILISTIC APPROACHES TO THE DESIGN OF WIRELESS AD HOC AND SENSOR NETWORKS

Zhen Guo

Dr. Mengchu Zhou, Dissertation Advisor | Date
Professor of Electrical and Computer Engineering, NJIT

Dr. Hongya Ge, Committee Member | Date
Associate Professor of Electrical and Computer Engineering, NJIT

Dr. Jie Hu, Committee Member | Date
Assistant Professor of Electrical and Computer Engineering, NJIT

Dr. Swades De, Committee Member | Date
Assistant Professor of Electrical and Computer Engineering, NJIT

Dr. Frank Shih, Committee Member | Date
Professor of Computer Science, NJIT

BIOGRAPHICAL SKETCH

Author: Zhen Guo

Degree: Doctor of Philosophy

Date: January 2006

Undergraduate and Graduate Education:

- Doctor of Philosophy in Computer Engineering,
  New Jersey Institute of Technology, Newark, NJ, USA, 2006.

- Master of Science in Computer Science,
  New Jersey Institute of Technology, Newark, NJ, USA, 2001.

- Bachelor of Science in Physics,
  University of Science and Technology of China, Hefei, P.R. China, 1994.

Major: Computer Engineering

Publications:

Zhen Guo and Lev Zakrevski,
"Probability-based reliable multipath selection in mobile ad hoc networks", in Proc. of the 16th IASTED International Conference on Parallel and Distributed Computing and Systems, pp. 286-291, Nov., 2004.

Zhen Guo and Mengchu Zhou,
"Prediction-based Object Tracking Algorithm with Load balance for Wireless Sensor Networks", in Proc. of 2005 IEEE International Conference on Networking, Sensing and Control, Tucson, AZ, pp. 756-760, March 2005.

Zhen Guo, Mengchu Zhou, and Lev Zakrevski,
"Optimal Tracking Interval for Predictive Tracking in Wireless Sensor Network", in IEEE Communication Letter Vol. 9 No.9 pp. 805-807, 2005.

Zhen Guo, Mengchu Zhou, and Guofei Jiang,
"Recursive Online EM Algorithm for Adaptive Sensor Deployment and Boundary Estimation in Wireless Sensor Networks", to appear in Proc. of IEEE International Conference on Networking, Sensing and Control, Ft. Lauderdale, Florida, April 2006.

Zhen Guo, Mengchu Zhou and Guofei Jiang,
"Optimal Coverage with Adaptive Sensor Placement Algorithm in Wireless Sensor Network", submitted to ACM Trans. on Sensor Networks.

Zhen Guo, Guofei Jiang, Haifeng Chen and Kenji Yoshihira,
"Tracking Probabilistic Correlation of Monitoring Data for Fault Detection and Isolation in Complex System", submitted to IEEE International Conference on Dependable Systems and Networks, Philadelphia, PA, June 2006.

"A THOUSAND-MILE JOURNEY BEGINS WITH THE FIRST STEP"

"千里之行，始于足下"

ACKNOWLEDGMENT

I would like to express my deepest appreciation to Dr. MengChu Zhou, who not only served as my research supervisor, providing valuable and countless resources, and insight, but also constantly gave me support, encouragement, and reassurance. Without his efforts this thesis could not have been written. Special thanks are given to Dr. Hongya Ge, Dr. Frank Y Shih, Dr. Jie Hu and Dr. Swades De for actively participating in my committee. I also wish to thank Dr. Lev Zakrevski, my former advisor, for consistent guidance and support throughout my academic program.

My thanks also go to Dr. Guofei Jiang, my mentor of internship at NEC Laboratories America, who offered much-appreciated advice, support and thought-provoking ideas. I have appreciated the financial support of the Department of Electrical and Computer Engineering, New Jersey Institute of Technology during the years of my Ph.D. study.

Many of my fellow graduate students in the Department of Electrical and Computer Engineering are deserving of recognition for their support. They include Zhigang Wang, Lichuan Liu, and Dekun Zou.

TABLE OF CONTENTS

TABLE OF CONTENTS
(Continued)

Chapter **Page**

TABLE OF CONTENTS
(Continued)

Chapter **Page**

LIST OF FIGURES

LIST OF FIGURES
(Continued)

Figure **Page**

LIST OF TABLES

CHAPTER 1

INTRODUCTION

1.1 Motivation

With the rapidly increasing demand for connectivity, wireless communication becomes more and more important. Most of the portable communication devices have the support of fixed base stations or access points. However, this support is not available in some extreme scenario where it may not be possible to get access to fixed access points, e.g., natural disasters, military settings, and exploration. This led to the necessity for wireless self-configuring networks, such as Mobile Ad-hoc NETworks (MANET), and Wireless Sensor Networks (WSN). Today's wireless networks have become highly flexible and can be configured and adapted to different environment much more rapidly. In particular, practical emergence of the mobile ad hoc networks and also sensor networks is widely considered revolutionary because their flexibility provides one of the missing connections between pervasive networks and physical world [1]. Since wireless ad hoc networks and sensor networks are capable of dealing with mobile objects, probabilistic estimation on mobile objects is becoming an attractive choice for wireless ad-hoc networks and also in sensor networks.

This chapter discusses the background of this dissertation, which includes the fundamental of mobile ad-hoc networks and wireless sensor networks, the mobility estimation of the objects under these networks, the performance analysis of the prediction effects on the Mobile ad-hoc networks and sensor networks, and recursive learning algorithm for estimating the distributions of moving objects in sensor networks.

Then contributions of this dissertation are briefly presented, which include a probability-based reliable ad-hoc network multi-path selection framework, a predictive object tracking methodology, the optimization of power consumption of wireless sensor networks, and adaptive optimal coverage method for sensor networks. At the end, the outline of this dissertation is given.

The advancement in wireless communications and light-weight, small-size, portable computing devices have made pervasive and mobile computing possible. One wireless network that has attracted a lot of attention recently is the *mobile ad hoc network (MANET)*. A MANET is a wireless network consisting of a set of mobile hosts which may communicate with one another and roam around at their will. Mobile hosts may communicate with each other indirectly through a sequence of wireless links without passing base stations (i.e., in a *multi-hop* manner). This requires each mobile host serve as a router. A scenario of MANET is illustrated in Fig 1.1.

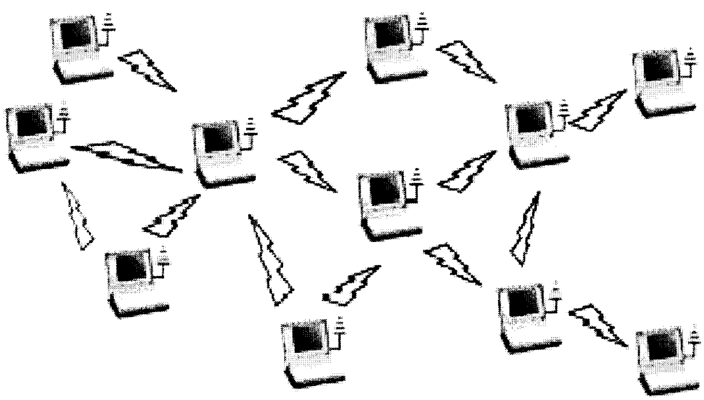


Figure 1.1 An example of mobile ad hoc network (MANET).

Applications of MANETs occur in situations like battle-fields, festival grounds, and emergency rescue actions, where networks need to be deployed immediately, but base stations or fixed infrastructures are not available [2]. For example, in an earthquake disaster, all base stations may be down since there is no electricity. In this case, a MANET driven by battery power can be quickly deployed to set up a network environment. Such technology has been recently applied to wireless sensor networks and personal-area networks too. Because of its flexibility and dynamic topology, MANET is attracting much research efforts on the route reliability issues. When choosing a routing path among several candidates, there are usually many factors to be considered, such as route length, route quality, signal strength, and route lifetime. All of those factors are dependent on the moving pattern of the mobile nodes. Motivated by the work in [2],[3], this work studies the impact of a mobility model on route reliability. It is believed that a good model of mobility prediction can evaluate the reliability of links, help the source find a most reliable route, and finally improve the performance of the entire MANET.

As the advances in wireless technology have enabled the development of tiny low-power devices capable of performing sensing and communication tasks, *wireless sensor networks* have emerged and received the attention of many researchers. Wireless sensor networks are a special type of ad hoc networks, where wireless devices get together and spontaneously form a network without the need for any infrastructure [3]. Fig. 1.2 shows the architecture of a sensor network in which sensor nodes are shown as small circles. Because of the lack of infrastructure, sensor networks inherit the multi-hop communication environment from the ad hoc networks. Although they are a special type of ad hoc networks, sensor networks have their own characteristics, such as very limited energy sources, high density of nodes deployment, and cheap, perhaps unreliable sensor nodes.

Among the technical issues to be addressed in developing sensor networks for object tracking, power efficiency is probably the most critical one since the sensor nodes are often powered by batteries which could be difficult to replace. The ideas of utilizing mobility estimation to save power are not new in mobile computing systems. For example, in cellular networks, probability based predictive techniques are proposed to reduce the paging overhead by limiting search space to a set of cells that mobile users may enter [4], [5]. Similarly in wireless sensor networks, mobility estimation and prediction based on the past reading history can reduce the number of transmission [6].

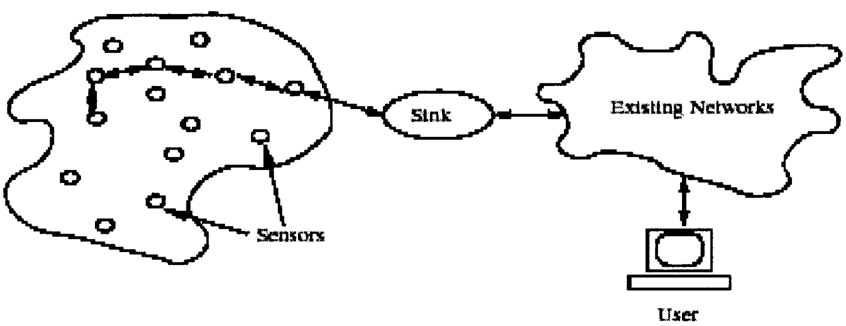


Figure 1.2 Architecture of a Sensor Network.

Sensor placement is another important issue which can be studied and/or solved with help of probabilistic estimation to learn mass characteristics of object distributions. Based on the estimations, the sensors are informed to move to the updated optimal locations accordingly. Due to dynamics of real world applications and uncertainty associated with object movement, the parameters of object distribution should be estimated with maximum likelihood or maximum a posteriori solution in a timely manner. Considering real scenarios, a large number of moving objects attracted by several points of interests are often assumed to be distributed in the Gaussian Mixture model. Given a sufficient set of observations, the parameters of Gaussian mixture can be approximately learned. With this learning process, sensors can be placed according to the estimated distributions of thousands of targets.

1.2 Contributions of Dissertation Research

This dissertation presents a formal model to predict the lifetime of a route in a MANET [7]. It is assumed that each mobile host roams around following some specific mobility models. Given a sequence of mobile hosts which form a routing path, the joint probability distribution of route lifetime is derived based on the mobility model. This differs from most existing works which directly calculate the lifetime of a wireless link based on the current locations and roaming directions of two neighbor hosts by assuming that their roaming directions do not change. The proposed model of predicting the route reliability includes: node mobility study, prediction of link availability, and evaluation of route lifetime. Based on the quantitative analysis, a reliable path selection algorithm to improve the reliability of routing in MANET is developed.

When applying the mobility estimation and prediction in wireless sensor networks, we conduct mathematical analysis of the objects' moving pattern, and then propose a prediction based object tracking approach in which the future motion of the detected objects can be predicted so as to make use of redundant local transmissions to compensate for the lack of long distance transmissions [8]. The simulation results show that this predictive object tracking approach significantly reduces the power consumptions in wireless sensor networks and outperforms the compared approaches.

Obviously the accuracy of prediction depends on the "tracking interval". Here the tracking interval is defined as the time length between two consecutive sensing points with the intuition that as the resolution becomes finer, the miss probability decreases. As the tracking interval becomes lower, in other words "more frequent", the tracking power consumption is increased. As it increases, the miss probability increases, thereby lower the tracking quality. The failure of locating a target should be recovered by a certain mechanism, which leads to extra power consumption for re-capturing the target. Hence, an analytical framework is highly desired in order to determine an optimum value of tracking intervals. By analyzing the effect of mobility estimation and prediction on power consumption in wireless sensor networks, the work constructs a quantitative model to find such an optimal tracking interval, studies the effect of the tracking interval on the miss probability, and proposes a scheme called Predictive Accuracy-based Tracking Energy Saving (PATES) to achieve the optimal power efficiency by exploiting the tradeoff between the accuracy and cost of sensing operation.

Similarly, optimal sensor placement can also be achieved by taking advantage of probabilistic estimation on the moving pattern of objects. More research attention should be drawn to the issues of monitoring and tracking a large number of moving objects. Given sufficient observations, it is desired to know where the sensors should be deployed so that optimal coverage and detection accuracy can be achieved. An unsupervised learning method for estimating the object distributions is proposed to obtain the required knowledge for sensor placement. The proposed method is aimed at optimal sensor placement for detecting a mass of objects with maximal detection accuracy. The adaptive optimal sensor placement method proposed in this dissertation provides for the first time a probabilistic solution to adaptively learn the distribution of objects and move the sensors to the optimal locations with the maximal detection accuracy. Meanwhile, it can also estimate the real-time updated boundary of the targets adaptively.

1.3 Outline of Dissertation

Chapter 2 proposes a mobility prediction model for maximizing the reliability to use for the reliable routing path selection, trying to maximize the route lifetime. Section 2.2 describes briefly the mechanism to estimate the reliability in ad hoc networks. Section 2.3 presents statistical analysis of the stability and link selection. Section 2.4 proposes a simple reliable routing scheme based on the proposed link stability prediction and presents specifications. Section 2.5 is performance analysis of the proposed reliable routing approach and its comparison with some existing routing protocols.

Chapter 3 proposes a prediction-based tracking algorithm in which the future motion of the detected objects can be predicted so as to make use of redundant local transmissions to compensate for the lack of long-distance transmissions. Section 3.2 describes a general system structure, preliminary assumptions on 1-D mobility model of the tracked object and topology for illustration and further analysis, and finally proposes an object tracking framework. Section 3.3 proposes an energy calculation model, and compares the energy consumption in the proposed method with those of other algorithms. Section 3.4 makes a performance analysis, in which the proposed method is compared with PREMON algorithm and naïve system, in order to quantify the difference of performance via a case study. Section 3.5 ends the chapter 3 with some conclusions.

Chapter 4 proposes a quantitative analytical model to find an optimal tracking interval, studies the effect of the tracking interval on the miss probability and proposes a scheme called Predictive Accuracy-based Tracking Energy Saving (PATES) by exploiting the tradeoff between the accuracy and cost of sensing operation. Section 4.2 describes the predictive tracking sensor network architecture, and presents the predictive accuracy-based tracking energy saving scheme. Section 4.3 develops a quantitative method to optimize the power efficiency by choosing an optimal tracking interval, and presents an example with an optimal result.

Chapter 5 presents the adaptive optimal sensor placement method. This chapter utilizes a recursive learning process to estimate the dynamical distribution of objects, and then proposes an adaptive optimal sensor placement strategy for detecting the objects with maximal detection accuracy. This chapter also presents the real-time estimation of boundary of object distribution. Section 5.2 briefly reviews the related works. Section 5.3 formalizes the problem of optimal sensor placement, and analyzes the distribution of mass objects. The detection coverage is modeled as a Gaussian Mixture Model. The EM and recursive EM solutions are presented to solve the coverage problem based on the Gaussian Mixture model in Section 5.4. Section 5.5 proposes a possible choice for the distributed implementation of EM algorithms in sensor networks.

The details of simulation and detection performance are discussed in Section 5.6. Section 5.7 concludes this chapter.

The dissertation is concluded in Chapter 6, which briefly reviews the algorithms, methods and their contributions. The promising applications and techniques of probabilistic estimation for wires ad-hoc and sensor networks are also discussed.

CHAPTER 2

RELIABLE PATH ESTIMATION FOR MOBILE AD HOC NETWORKS

2.1 Introduction

Mobile ad hoc networks are difficult to support QoS-driven services because of their unpredictable and frequent topology changes. One of the important concerns is the reliability of routes. Although finding the optimal path for MANET is an NP-complete problem, specific routing protocols may sacrifice efficiency for the reliability. In particular, typical reliable routing protocols use redundant paths to achieve reliability. This chapter proposes a probabilistic model for maximizing the reliability in order to select the most reliable route, and increase the lifetime of the route.

The information flow over paths is constantly disrupted by the link breakage due to topology changes, which affects the QoS for MANET. Using stable links is crucial for establishing stable paths between connection peers. Rerouting results in unexpected overhead and delays to the connections.

A promising technique for coping with link breakage is to find the strongest link, i.e., the link with lowest failure probability, between nodes, based on the mobility information of nodes and the current ages of links.

The residual *lifetime* of a link is the time span during which the link stays connected. This chapter assumes that two nodes are connected if and only if they are within a given range. It is straightforward that a path with higher lifetime can be deemed as the most stable path between the connection peers. A path is deemed to be connected if all links included in the path are connected. Much recent work [9], [10], [11], [12], [13] addresses the problem of predicting the link stability in a quantitative way. However, these models alone are not sufficient because none of them presents the estimation model for route reliability by combining the availability of links.

Some existing reliable routing protocols use instant GPS information to obtain the estimated link life, and then determine which path is optimal. By using mobile node's random mobility pattern, one can predict the future state of a network topology and thus provide the most stable route [13]. Using the current age in addition to the mobility information of the connection pairs to predict the lifetime remains unexplored. This raises the problems of how to find a simple method for link selection based upon the statistical analysis of link duration.

The rest of this chapter will be organized as follows. The following section briefly describes the mechanism to estimate reliability in ad-hoc networks. Section 2.3 presents statistical analysis of the stability and link selection. Section 2.4 proposes a simple reliable routing scheme based on link stability prediction and presents specifications. Section 2.5 discusses performance analysis results of the proposed reliable routing approach in comparison with some existing routing protocols.

2.2 Estimation Models for Reliability

This section presents a Brownian motion-based mobility model for analyzing and characterizing the distribution of node movement. On the basis of analysis of node movement, the link and path availability can be predicted by analyzing the two-body mobility problem. In [9] a discrete approximate model for Brownian motion is presented to provide the basis for analytical derivation of random-independent link availability.

This chapter assumes that the 2-D free space transmission model is used. Hence the link status is determined by the distance between nodes. The basic assumptions of the proposed estimation algorithm are similar to those used in [9] and [10]. Hereby, some definitions are presented here, they were also used by McDonald and Znati[9].

Definition 2.1 *Mobility epoch* is a random length interval during which a node moves in a constant direction at a constant speed.

Definition 2.2 *Mobility profile* is defined for a given node moving according to a random ad-hoc mobility model based on three parameters: mobility epoch, the mean speed and the speed variance. Before a mobility model is described, some assumptions are made in developing this model:

- The epoch lengths are Identically, Independently Distributed (IID) with mean $1/\lambda_n$.
- The speed during each epoch is an IID distributed random variable with mean μ_n and variance σ_n, and remains constant only for the duration of the epoch.
- The direction of a mobile node during each epoch is IID uniformly over $(0,2\pi)$ and remains constant only for the duration of the epoch.
- Speed, direction and epoch are uncorrelated.

2.2.1 Node Mobility Models

Currently four mobility models widely used are Random Waypoint model, Reference Point Group Mobility model, Freeway Mobility model, and Manhattan Mobility model.

1. Random Waypoint Model

Each node chooses a random destination and moves toward it with a random velocity chosen from [0, Vmax]. After reaching the destination, the node stops for a duration defined by the "pause time" parameter. After this duration, it again chooses a random destination and repeats the whole process again until the simulation ends.

2. Reference Point Group Mobility Model

Each group has a logical center (group leader) that determines the group's motion behavior. Each node within a group has a speed and direction that are derived by randomly deviating from that of the group leader.

3. Freeway Mobility Model

Each mobile node is restricted to its lane on the freeway. The velocity of a mobile node is temporally dependent on its previous velocity. If two mobile nodes on the same freeway lane are within the Safety Distance (SD), the velocity of the following node cannot exceed the velocity of its preceding node.

4. Manhattan Mobility Model

The movement of mobile nodes can be viewed as random vectors. These vectors characterize the direction and distance moved by a mobile node during a single epoch.

This work assumes the nodes operate in a truly ad-hoc manner, hence the mobility should be random, and link breakages caused by motion of nodes are independent events. Suppose that a node can obtain the instant mobility information from a GPS system periodically. The movement of each node is assumed to be a 2-dimensional space motion model. Two more assumptions for the movement of mobile nodes are needed:

- Mobility epoch lengths are assumed to be exponentially distributed with mean $1/\lambda$.

$$E(T) = P\{\, epoch\ length < T \} = 1 - e^{T}$$

- Node mobility is uncorrelated.

Definition 2.3 *$\vec{R}_n(t)$ is the random mobility vector for node n. Its magnitude R(t) is equal to the distance from (X(t), Y(t)) to (X(t + Δ t) , Y(t+ Δ t)), where (X(t), Y(t)) is the position of the node at time t. Its phase angle θ(t) is the angel of line joining the node's initial position to its position at time t. The random mobility vector can be expressed as a random sum of the epoch random mobility vectors:*

$$\vec{R}_n(t) = \sum_1^{N(t)} \vec{R}_n^{\,i}$$

The work in [11] derives the mobility vector as follows. Its expected value of the equivalent random mobility vector is,

$$E[\vec{R}_{m,n}(t)] = \sqrt{\frac{\pi t}{2}\left(\frac{1}{\lambda_m}(\sigma_m^2 + \mu_n^2) + \frac{1}{\lambda_n}(\sigma_m^2 + \mu_n^2)\right)}$$

The following mobility metrics are used:

<u>Relative Speed (mobility metric I)</u>

The average magnitude of the relative speed of two nodes over all neighborhood pairs and all time

$$\overline{RS} = \frac{1}{P}\sum_{t=0}^{T}\sum_{i}^{N}\sum_{\substack{j=1\\ j\neq i}}^{N}(|\vec{v}(i,t) - \vec{v}(j,t)|)$$

<u>Spatial Dependence (mobility metric II)</u>

The extent of similarity of the velocities of two nodes that are not too far apart, average over all neighbor pairs and through all the time

$$\overline{D}_{spatial} = \frac{1}{P}\sum_{t=0}^{T}\sum_{i=1}^{N}\sum_{\substack{j=1\\ j\neq i}}^{N}\frac{\min(\vec{v}(i,t),\vec{v}(j,t))}{\max(\vec{v}(i,t),\vec{v}(j,t))}\times\frac{\vec{v}(i,t)\cdot\vec{v}(j,t)}{|\vec{v}(i,t)||\vec{v}(j,t)|}$$

<u>Temporary dependence (mobility metric III)</u>

The value of extent of similarity of the velocities of a node at two time slots that not far away, average over all neighborhood pairs and all time

$$\overline{D}_{temporary} = \frac{1}{P}\sum_{i=1}^{N}\sum_{t=0}^{T}\sum_{\substack{t'=0 \\ t'\neq t}}^{T} \frac{\min(\vec{v}(i,t),\vec{v}(i,t'))}{\max(\vec{v}(i,t),\vec{v}(i,t'))} \times \frac{\vec{v}(i,t)\cdot\vec{v}(i,t')}{|\vec{v}(i,t)||\vec{v}(i,t')|}$$

2.2.2 Link Stability Prediction

In this section, the prediction for link availability in form of time t is proposed based on the distribution of the distance between two mobile nodes during each mobility epoch. The term "link availability" defined in this chapter is different from that in [9].

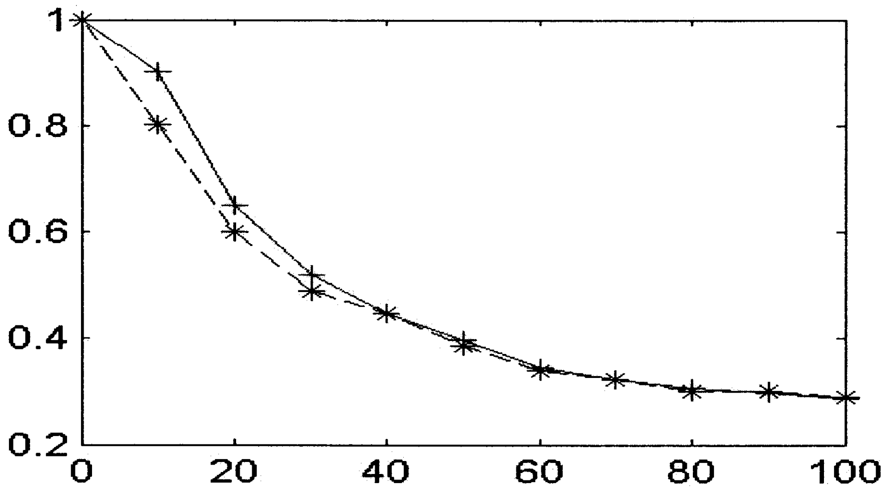
Definition 4: *Link availability is the conditional probability that an active link, which is available between two nodes at time t_0 is continuously available from t_0 to $t_0 + t_c$.*

Since it is assumed that the 2-dimensional free space transmission mechanism is used, a link between two nodes is considered as "available" at any time if and only if the distance between the two nodes is smaller than the transmission. Suppose that the joint location distribution of every two nodes is given. The link maintenance probability can be predicted by

$$P_{i,j}(t) = \iint_{|\vec{z}_i(t)-\vec{z}_j(t)|<r} f(z_i(t), z_j(t))dz_i(t)dz_j(t)$$

where $f(z_i(t), z_j(t))$ is the joint PDF for locations of nodes i and j. Since the motions of mobile nodes are assumed independent with each other as following,

$$f(z_i(t), z_j(t)) = f(z_i(t)) \cdot f(z_j(t))$$



T(sec)

Figure 2.1 Link availability.

The authors in [2] proposed an estimation model for the link reliability, which gives an expression for link availability at moment t_p.

$$L(t_p) = \frac{1-e^{-2\lambda t_p}}{2\lambda t_p} + \frac{\lambda t_p e^{-2\lambda t_p}}{2}$$

The numerical analysis in [10] gives an approximate formula,

$$T_{p,peak} = \frac{1+\sqrt{5}}{2}\lambda^{-1} \approx 1.618\lambda^{-1}$$

The link availability

$$p_{i,j}(T_p) = \frac{1-e^{-2\lambda T_p}}{2\lambda T_p} + \frac{\lambda T_p e^{-2\lambda T_p}}{2}$$

2.2.3 Path Stability

Section 2.2.2 demonstrates how the link availability is calculated. Now this metric can be used by a routing algorithm in order to construct paths that support a maximum reliable probability over an interval. Preliminarily the path availability model has to be studied as the basis for the construction of the reliable route. Lots of papers, such as [12], [14], [15], make similar assumptions on path availability. One of those assumptions, discussed in this section, is independent link failure. From this assumption, path availability can be expressed as

$$\phi^k_{m,n}(t) = \prod_{(i,j)\in k} p_{i,j}(t) \tag{2.1}$$

where $\phi^k_{m,n}(t)$ is the availability of the path between nodes m and n. $p_{i,j}$ is the link availability between nodes i and j. Based upon this assumption, Dijkstra algorithm can be used to find the most reliable path by using $\log(p_{i,j})$ as the "*link cost*".

Each node periodically broadcasts its neighbor information. It can periodically compute the link active probability with each 1-hop neighbor based on the probability method described in Section 2.2.2.

Considering the example shown in Fig 2.2, one can view the reliable path selection mechanism as the reliability maximization problem of a parallel-series graph system [3]. A MANET can be modeled as a probabilistic graph G_P = $\{V,E\}$ with probabilities of link activity assigned to the edges.

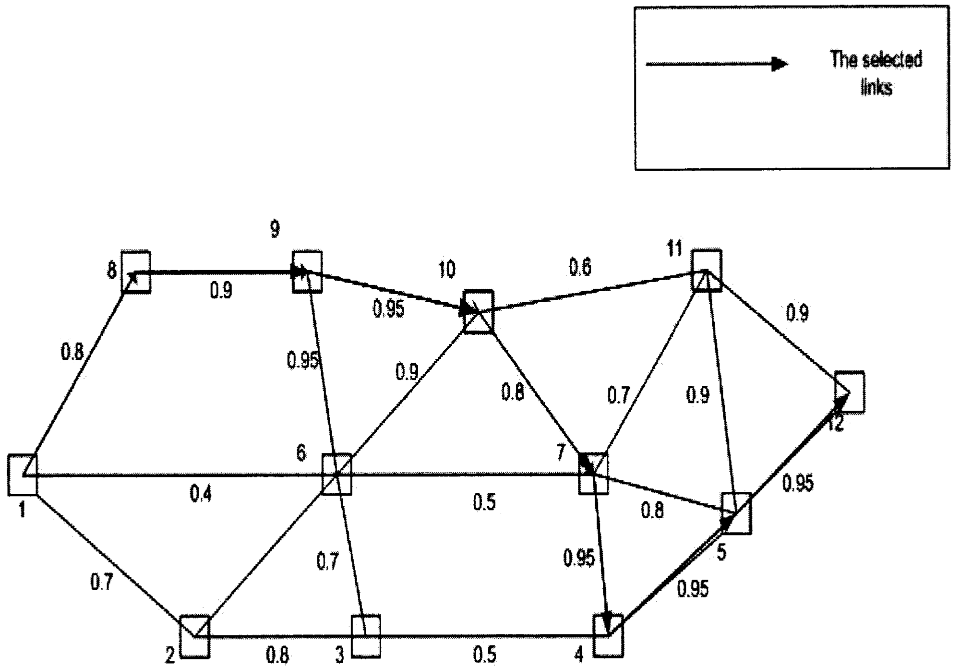


Figure 2.2 An example of topology graph.

The algorithm starts by locally constructing several paths as its detail is presented in Section 2.3.2. Each path comprises several links, and these paths may share some common links. As a result, some discovered paths may consist of unreliable links. Consequently, the goal is to choose the most reliable links to form a specific path.

For a given path, which is analogy to a series system, the overall reliability is worse than or equal to each of its links. But for a path set from source to destination, which is a parallel system, the overall optimum path should be better than each of single paths in the path set [16]. This work summarizes this system with $R_{opt}\{G\}$ as the reliability of optimal path and R(links) as the reliability of each link in a given path as follows:

$$R_{opt}\{G\}=Max(1-\prod_{path\,\in\,path\,set}[1-(\prod_{j\,\in\,path}P_{i,j}(t)])\qquad(2.2)$$

However, the path availability from the experiment does not match the Equation 2.2 very well. The reason for this mismatch is that the probability that two neighboring links are broken cannot be considered to be independent with each other, since these two parameters depend on the movement of the same node. Fig. 2.3 shows the simulation and analytical results of the path availability. The mismatch of the simulation and calculation can be observed from the dashed line and dotted line in Fig. 2.3. The solid lines p1, p2 and p3 stand for the connected probability of the link1, 2 and 3, respectively. It is noted that as the time goes by, the mismatch of the calculation and simulation becomes larger and larger. Hence, the two links sharing a same intermediate node can not be assumed to be independent with each other is drawn.

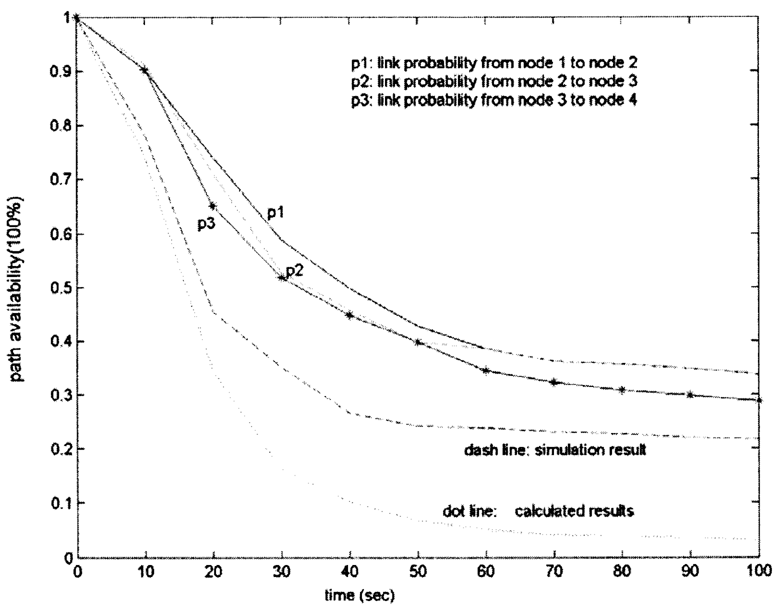


Figure 2.3 Simulation and analysis of path stability.

2.3 Path Prediction and Selection

Although a closed form expression of path availability model has yet to propose so as to match the experiment data very well, a common sense that the path availability mainly depends on the availability of the weakest link is straightforward. Based on the connected probability of each link in the whole network, a routing framework, referred as Probability Based Reliable Multi-path Selection (PBRMS) is proposed. By broadcasting the RREQ message and receiving the RREP message from the other nodes, the source node is able to obtain the global information of the link availabilities. The route constructed by PBRMS is expected to be the route of which the weakest link has greater availability than the weakest link of every other possible route.

2.3.1 PRMS Description

The shortest path routing is not suitable for wireless ad hoc networks due to the mobility of nodes. Since the topology changes frequently, the shortest route may be the most unreliable route among all possible paths because it chooses the route with the smallest hop count, where the chosen links are potentially too long and tend to be broken easily. The objective of reliable routing is to minimize the expected number of transmissions per route from source to destination, given by $E = \sum 1/(P_{i,j} \cdot P_{j,i})$, where $P_{i,j}$ and $P_{j,i}$ is the link reliability, and each link is symmetric.

The PRMS constructs a set of reliable paths iteratively. It begins with finding the path with the highest active probability. As time goes on, the probability may be decreasing, hence the previous optimal route may no longer be the most reliable one. Many new paths are kept to be appended to the existing path set. Due to the highly dynamic topology of MANET, the adaptive reliability is a huge issue in the concern of PRMS. The algorithm keeps comparing the reliability of the paths in the path set. Hence, one has to compromise the computational overhead to reach high route reliability.

Path	Hop count	Expect # of Retransmission
A→C	1	4
A→B→C	2	2.34

2.3.2. Route Discovery Phase

When source node S requires a route to destination D, S enters the route discovery phase and checks whether adequate "fresh" routes to D are available in the *Fresh_Routes_Cache* first. If some fresh routes to D are found, S sends packets simply according to these existing routes. If not, S runs *New_Route_Discovery_Process* to find a new route to the destination node.

Source node S broadcasts RREQ (Route discovery REQuest) to all neighbor nodes, and then each neighbor node forwards them to their neighbors by flooding. RREQ includes a sequence number field to distinguish every route discovery process from the others, and a route content field records nodes' IDs along the path from S to D. After the intermediate node receives RREQ from an upstream node X, it inserts its ID and the predicted lifetime of the link between itself and its upstream node into the route content field and lifetime field, respectively, of the RREQ, then sends this modified RREQ to its neighbor nodes (except the upstream node X). The RREQ_caches of the intermediate nodes also store the routing information, including the sequence number of the RREQ and the IDs of neighbor nodes.

If a node receives the RREQ with the same sequence number from another neighbor, e.g., Y, then it checks whether the route content of RREQ includes its ID; if so, the node discards this RREQ. Otherwise the node inserts its ID into the RREQ, and then checks whether Y is in the RREQ_cache. If so, the node clears Y from the RREQ_Cache, and then forwards the RREQ again to its neighbors, specified in the RREQ_Cache. If Y has no more downstream nodes in the RREQ_Cache, the RREQ is discarded. The node discards only the duplicate RREQ with its ID in the route content of RREQ. Hence, it not only avoids the infinite loops but also protects the existence of multipath.

After the destination node D receives the first RREQ, and waits for more RREQ packets in a tolerable period T, when the clock of node D reaches T, D collects the lifetime information of each link, and finds out which RREQ is correspondent to the route where the minimum lifetime value of its links is greater than those of all others. The route whose weakest link has greater lifetime than the weakest link in every other route is the most reliable route among all the routes [17].

As the process of reliable path selection is completed, Node D sends back RREP(Route discovery Response) packet to the source node S according to the most reliable route recorded in the chosen RREQ.

2.3.3. An network example

Figure 2.2 presents an example of MANET with some probability of link activity assigned to each link. The algorithm starts by finding an initial path from 1 to 12, say, {16, 7, 5, 12}, and appends it to the path set G. The first iteration is completed. At the following iterations, it appends all other possible paths to the end of the path set. By multiplying every link active probability, the proposed algorithm can choose an optimal path with maximum $\prod P_{i,j}(t)$ at moment *t*. In this example, the path (1,8,9,10,7,4,5,12) is chosen. Thus, at moment t it is the optimal path between nodes 1 and 12.

2.4 Performance Analysis

This section compares the performance of PRMS with a shortest path algorithm , e.g., Ad hoc On-demand Distance Vector (AODV). In the following simulations, each experiment is repeated 10 times, each time the topology is different from others. Network size is 20~60 nodes, which are randomly distributed in an area of 1000meters × 1000 meters. The power range of each node is assumed to be 200 meters.

The free space propagation function is used, i.e., the link between two nodes stay connected if and only if they are within 200 meters apart. The pair of nodes with the longest distance is selected as the source-destination pair. Each experiment is repeated 100 times to study the performance of the proposed algorithm, and the averages are taken.

Three metrics are used to compare the performance of PRMS with that of AODV. They are: the expected path lifetime, hop count of the selected path and number of overhead packets. Note that the route with the minimum hop count may not be the most reliable one. The expected number of retransmissions per sending request is given by $\sum 1/\ (P_{i,j} \cdot P_{j,i})$ and used as a measure of the path life. The lower this number is, the longer the path lasts.

The lifetime of Dynamic Source Routing (DSR) vs. reliable routing with lifetime prediction is shown in Fig. 2.4. Each data point is obtained from the experiments repeated 1000 times on 500m*500m square where free space transmission is assumed. From Fig. 2.4, it can be observed that the lifetimes of a reliable path are 50% to 100% longer than those of the shortest paths. The hop count of reliable and shortest paths is shown in Fig. 2.5 where each experiment is repeated 1000 times. The hop count of reliable paths is found to be greater than those of shortest paths. The overhead routing packets of reliable and shortest path routing is shown in Figure 2.6. It is found that shortest path routing has no much change in the number of overhead packets as the network size increases, while the overhead of reliable routing is much lower and decreases with the increase of the network size.

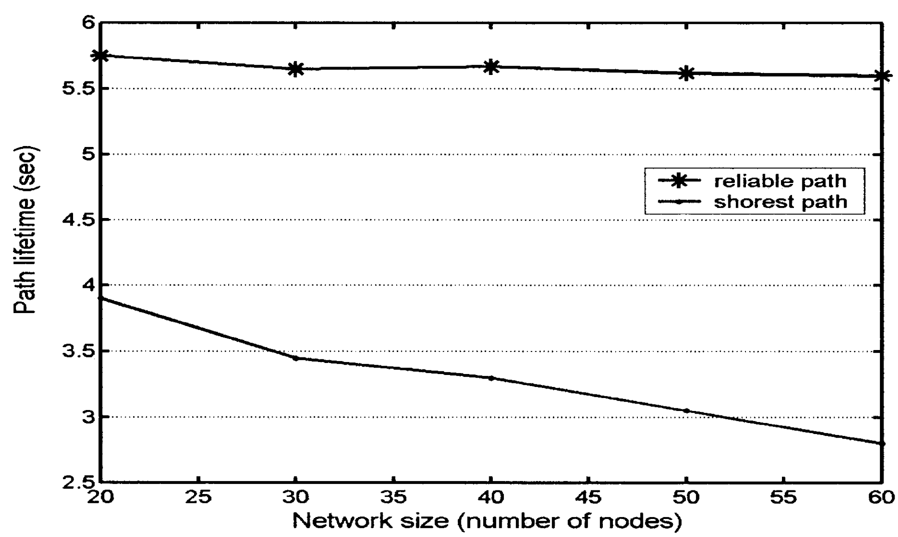


Figure 2.4 Lifetime study.

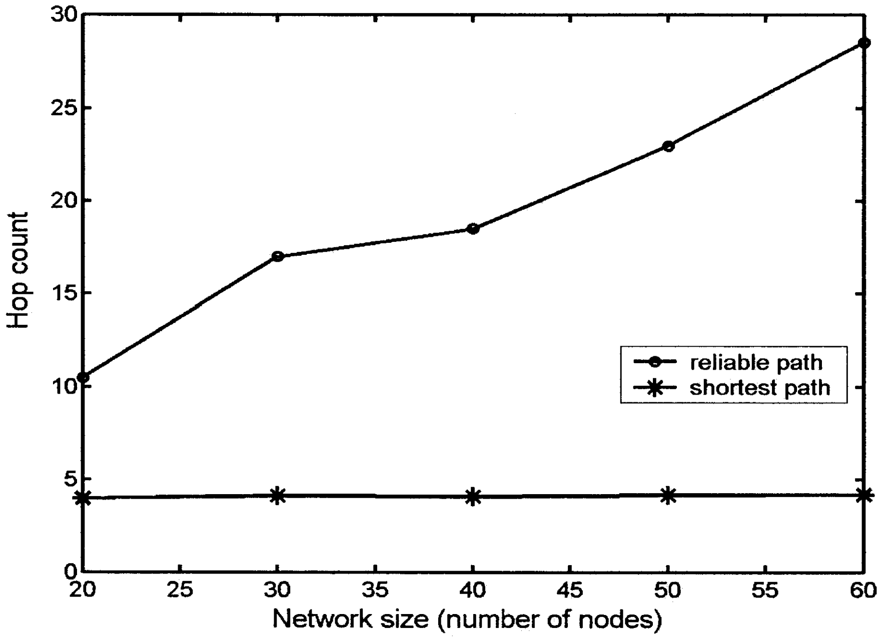


Figure 2.5 Hop count study.

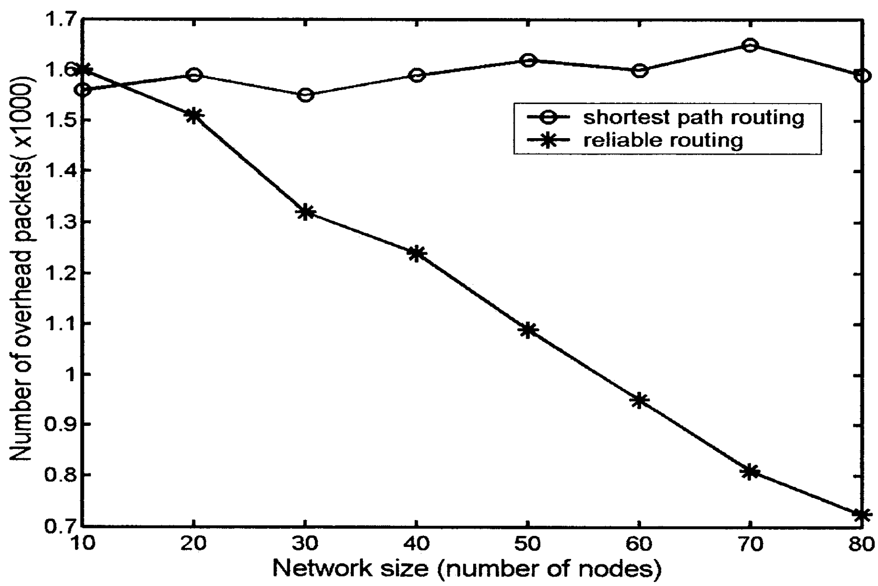


Figure 2.6 Routing overhead packet.

2.5 Summary

The QoS applications of MANET motivate this research on the probabilistic model of dynamically changing topology. In order to construct the most reliable route for each moment, Probability-based Reliable Multi-path Selection (PRMS) is proposed to find and maintain the most reliable route. With the use of PRMS, one can take advantage of redundant routing overhead to make route impervious to link failures due to topology changes.

Theoretical analysis of the lifetime of a routing path should be extended, and takes into account the correlation between lifetimes of the neighbor links in the path. Based on this analysis the routing protocol can be modified, and additional experiments can be designed and performed.

CHAPTER 3

PREDICTION-BASED OBJECT TRACKING ALGORITHM

3.1 Introduction

Power efficiency is one of the major concerns of wireless sensor network applications. Most existing power efficient routing algorithms for wireless sensor networks concentrates on finding efficient ways to forward data. Yet not much work has been done on collecting local data, tracking the detected objects, predicting the future motion, and then generating the data report. This chapter presents the effect of object tracking prediction on power saving for wireless sensor networks. An object tracking prediction method, i.e., Prediction-based Object tracking Algorithm with Load balance (POTL) has been proposed to achieve significant power savings by reducing unnecessary transmission. Performance analysis, based on numerical analysis for one dimensional case, demonstrates that the proposed scheme can dramatically reduce the power consumption and improve the lifetime of the overall network system.

More and more research efforts have been conducted on energy efficiency for wireless sensor networks deployed for the remote interaction with the real world. A good energy efficient protocol is expected to conserve as much power as possible while ensuring the given QoS requirement. The lifetime of a wireless sensor network strongly depends on the energy efficiency of the employed protocol. The proposed protocol in this chapter aims at reducing the redundant sampling and transmission, which is one of major methods of improving energy efficiency for wireless sensor networks.

Object tracking consists of detecting and monitoring locations and motions of real-world objects. Numerous applications of tracking are currently in place, such as air traffic control, railway monitoring and habitat monitoring. Networked sensors have been demanded for use for this tracking purpose [18][19][20].

Most of the existing sensor boards provide four different modes for radio transmission: *Transmit, Receive, Idle* and *Sleep* [21], in which transmit and receive mode are so called *active modes*. Power consumption studies on wireless sensor networks conclude that:

1) Long distance transmission consumes much more power than short distance one does at wireless integrated network sensors (WINS) nodes;
2) Idle mode consumes nearly as much power as receiving mode does;
3) Sleeping mode consumes only around one-sixth of the power in active modes.

These findings provide the clues of power savings for wireless sensor networks, i.e., higher energy efficiency can be achieved by reducing the amount of long distance transmission with the cost of utilizing more local communication. This chapter proposes a prediction-based object tracking approach in which the future motion of the detected objects can be predicted so as to make use of redundant local transmissions to compensate for the lack of long distance transmissions. Another design goal is fault-tolerance, which means the mechanism of load balance should be utilized to improve the Max-Min value of power residue of sensor nodes in the whole system. Hence, the lifetime of the whole sensor network can be increased. It is believed that this work contributes to improve the reliability performance and lifetime of the wireless sensor networks by using object-tracking prediction.

3.2 Object Tracking Mechanism

This section describes a general system structure, preliminary assumptions on 1-D mobility model of the tracked object and topology for illustration and further analysis, and finally proposes an object tracking framework.

3.2.1 System Structure

Generally, each sensor node consists of two different functional units: sensing and communication. This chapter focuses on communication unit. The term "sensor node" refers to its communication unit, unless otherwise stated.

Zone-based hierarchy is used in the proposed approach to improve the power efficiency and reliability. Zone head token is initiated and assigned to one sensor node within the zone randomly when the system starts. As the time goes by, the zone head will consume more power than any other sensors. Hence in order to balance the load, the token is passed to other nodes periodically according to the proposed algorithm to be described later. A node broadcasts the prediction information to its neighbors and the zone head. Only zone head is responsible for sending predictions to base station every k (>1) time slots, the other nodes do not send data to the base station over long distance transmission unless they receive the head token. Inside the zone, TDMA is used as the MAC protocol, and sensor nodes contact each other with low power paging channel. Suppose that each zone covers n nodes, each slot is assigned to each sensor node evenly, the period of the round robins is T sec. Hence in each cycle, a time slot each sensor node takes T/n sec to broadcast information to its neighbors. Otherwise it is in sleep mode to save power. Furthermore, sensor nodes can be awaked by requests from other nodes.

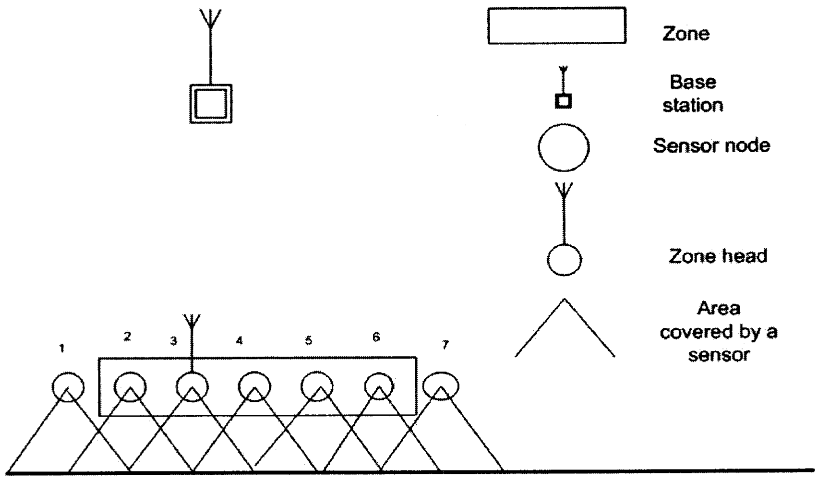


Figure 3.1 A system setup architecture.

The regions covered by sensor nodes are highly overlapped. From Fig. 3.1, it can be observed that wherever the object appears, at least two sensors should be able to detect it. Obviously, the device redundancy is doubled, while the total power consumption almost remains unchanged due to the load balance consideration in the proposed algorithm. Furthermore, since more nodes share the power consumption, the reliability and lifetime of the whole system can be significantly improved.

3.2.2 Mobility Model and Object Tracking

Before the mobility model and object tracking algorithm is described, it is necessary to make some preliminary definitions:

Definition 3.1 *Moving average* is a series of successive averages of a defined number of the past variables.

The proposed method can predict the trend of next values. As each new variable is included in calculating the average, the oldest variable of the series is deleted.

Definition *3.2 Moving window* is a predefined number of the past variables based on which the future value can be predicted.

The movement of tracked objects can be viewed as random vectors. Since only one dimensional sensor networks application are concerned in the proposed algorithm, the movement characters are all 1-D vectors. These vectors characterize the direction and distance moved by an object during a single time slot.

The sensor nodes detect and measure the motion of the object. Considering 1-D applications, this work focuses on 1-D motion only in this chapter. Initially, the sensors, that succeed in detecting the objects track the objects based on the monitoring and positioning utility of the sensors. The following assumptions are made:

1. All the sensor nodes are static and capable of obtaining the instant position and velocity of the moving objects by its stand-alone positioning utility within the coverage area.
2. The moving objects can be tracked by at least two sensors.
3. The speed during each time slot is an Independent, Identically Distributed (IID) random variable.
4. Sensors do not broadcast predictions unless the actual location or movements are different from those of predictions.

For object tracking applications, the state of a moving object, such as direction, velocity and route, can be retrieved by sensor nodes through collecting moving patterns of the tracked objects. For example, the current movement of an object is a reflection of the patterns of the moving history. Hence a sensor node may be able to predict the object's future moving patterns. Let l_t be the predicted location of the node at time t, v_t be the velocity of the node at time t, the moving average window size is w and s is the length of each time slot. Then,

$$l_t = l_{t-1} + [v_{t-1} + \Delta_{t-1}v \cdot s] \cdot s \text{, where } \Delta_t v = \sum_{i=t-w+1}^{t} \frac{v_{i+1} - v_i}{w} \tag{3.1}$$

is a moving average of acceleration of an object. The moving information of an object, such as location and velocity, can be obtained by target locating functional units of the sensor nodes. The moving history, i.e., the values of the past velocity and location, are transferred from one node to its neighbors. The prediction and transmission part of the proposed algorithm is described next.

3.2.3 Prediction-based Routing and Scheduling

Prediction models refer to the prediction functions that incorporate the topology knowledge and strategies to predict objects' movement.

From Equation (3.1), the future location of an object is predicted based on the moving history information by using the moving average method to predict the acceleration and then the location in the next time slot. The sensing units of nodes are always active in detecting and tracking the objects, but the communication units are not. Furthermore, a sensor does not transmit prediction information to base station over long distance transmission unless the head token arrives. For the convenience of description, POTL is separated into 3 phases: Initiation, information collection, and decision making

Initiation:

Suppose that an object enters the area covered by the sensor network. The first sensor node that detects and monitors the object is called the *initial node*. The initial node bewares of the object, obtains its moving information during the first several time slots until moving window size is reached, tracks it and loads its moving information into the processing unit. As long as the moving window is filled up with the available historical data, i.e., the number of past data equals to the window size, the next state of the location and velocity of the tracked object can be predicted.

Information collection:

As mentioned before, TDMA is assumed as a MAC protocol for transmission between sensor nodes and their zone head. Zone head is responsible for collecting moving information of the objects and transmitting it to each sensor node in its zone during each different time slot. An object can be monitored by two sensor nodes, but only the one with more power residue sends report to zone head to balance power. The occupancy of the head token and power residue information of each node are transmitted by broadcasting to all nodes within the zone through low power paging channel periodically. Hence in every zone, the zone head should switch to other sensor nodes periodically. A sensor node cannot send and receive data reports to/from the zone head until its assigned time slot is on duty. During the non-duty time slots, its communication unit stays asleep except when it is time to broadcast the power residue information over low power paging channel.

Decision making:

The zone head collects the moving information of the tracked objects, make predictions based on the information according to Equation (3.1). The expected location in the next interval can be predicted. The zone head sends a prediction report to the node that is expected to cover the predicted location, its zone head broadcasts alert to all the nodes within the zone and the base station, only if the corresponding sensor node does not find the objects. The global information is then quickly released from the base station and broadcasted to all the zone heads. All the nodes are activated and trying to catch the object. Again the information is collected by the corresponding zone head.

3.3 An Energy Model

The parameters used in the analysis are summarized in Table 3.1. A popular energy model [22], [23] is as follows:

Energy consumption in transmitting a P-bit message over a distance D, represented by $E_{T_x}(P,D)$, consists of two parts: the energy required to run the transmitter, $E_{T_{x-elec}}(P)$; and the energy needed for the transmitter amplifier, $E_{T_{x-amp}}(P,D)$ can be expressed by:

$$E_{T_x}(P,D) = E_{T_{x-elec}}(P) + E_{T_{x-amp}}(P,D) = E_{elec} \cdot P + \varepsilon \cdot P \cdot D^2 \tag{3.2}$$

Energy consumption in receiving this message is,

$$E_{R_x}(P) = E_{elec} \cdot P \tag{3.3}$$

where E_{elec} is the energy required to run the transmitter or receiver with a typical value of 50nJ/bit, and ε is the energy needed for the transmitter amplifier with a typical value of 0.1nJ/bit/m^2.

In POTL, the energy consumption should include the energy for correct prediction and incorrect prediction as well as the energy for overhead (including the head token switch overhead, and the broadcasting missing alert overhead). For convenience of expression, this work summarizes the calculative parameters' definition into Table 3.2 as shown. The energy consumption should be expanded in the form as follows

$$
\begin{aligned}
E_{POTL} &= N \cdot \alpha \cdot (E_{T_x}(P_{prediction}, D_{head}) + E_{R_x}(P_{prediction})) \\
&+ \frac{K}{2} \cdot (1-\alpha) \cdot (E_{T_x}(P_{naive}, D_{head}) + E_{R_x}(P_{naive})) + \frac{K}{2c} \cdot m \cdot E_{broadcast}
\end{aligned}
\tag{3.4}
$$

where m is the number of sensor nodes in each zone, and c is the number of intervals in which the overheads are broadcasted.

Table 3.1 System Parameters

Parameters	**Description**
N	Number of sensor nodes involved in object tracking
K	Number of transmissions between sensors and zone head
m	The number of sensor nodes in each zone
P	Size of the message in transmission
D	Distance of transmission
C	Number of intervals involved in overhead broadcasting
L	Length of a TDMA time slot
α	Accuracy of prediction
ε	Energy needed for the amplifier of transmitter

Table 3.2 Definitions of POTL calculative parameters

$E_{broadcast}$	The consumed energy for the zone head's broadcasting
$P_{prediction}$	The size of prediction packet
D_{head}	Average distance from nodes to its zone head
P_{naive}	The size of a packet that were transmitted without prediction

In a naïve system, i.e., a system without using prediction, the energy consumption is as follows.

$$E_{naive} = \frac{K}{2} \cdot (E_{T_x}(P_{naive}, D_{head}) + E_{R_x}(P_{naive})) \tag{3.5}$$

In PREMON [24], the moving history is transmitted when the sensor node clocks to communicate the zone head no matter whether the readings differ from the prediction or not, and the size of a prediction packet from a zone head, referred to *PREMON packet*, is much bigger than that of POTL. The total energy consumption in PREMON is:

$$\begin{aligned} E_{\mathrm{Premon}} &= \frac{K}{2} \cdot (E_{R_x}(P_{\mathrm{Premon}}) + E_{T_x}(P_{\mathrm{Premon}}, D_{head})) \\ &+ \frac{K}{2} \cdot (1-\alpha) \cdot (E_{T_x}(P_{\mathrm{Premon}}, D_{head}) + E_{R_x}(P_{naive})) + \frac{K}{2} \cdot E_{comp} \end{aligned} \tag{3.6}$$

By subtracting Equation (3.4) from (3.6), the amount of saved power consumption used in POTL compared with Premon can be:

$$\begin{aligned} E_{\mathrm{Premon}} - E_{POTL} &= \frac{K}{2}(E_{comp} - \frac{m}{c} \cdot E_{broadcast}) + \frac{K}{2} \cdot (1-\alpha) \cdot E_{T_x}(P_{\mathrm{Premon-prediction}}, D_{head}) \\ &+ \frac{K}{2} \cdot (E_{R_x}(P_{\mathrm{Premon}}) + E_{T_x}(P_{\mathrm{Premon}}, D_{head})) - N \cdot \alpha \cdot (E_{T_x}(P_{prediction}, D_{head}) + E_{R_x}(P_{prediction})) \end{aligned} \tag{3.7}$$

3.4 Performance Analysis

This section compares POTL with PREMON algorithm and naïve system, in order to quantify the difference of performances via a case study. Suppose that these algorithms are applied to a sensor network for monitoring a railway system. This system can be simplified as a 1-D application for wireless sensor network. In this case it is assumed that the average of distance between sensor nodes and zone head is 25 m, or $D_{head} = 25$. Let K and N be 500 and 50, respectively, in other words, the total number of transmissions between sensors and zone head is 500, and among them only 50 are actively sending data packets from sensors to zone head. This system is also assumed to be event-driven networked sensor system, i.e., it supports the function of retrieving only critical event information and sending them out. Finally P_{naive}, P_{premon}, and P_{POTL} are assumed to be 8, 7, and 7, respectively, although P_{POTL} is supposed to be much smaller than P_{premon} because in POTL, the prediction packet only consists of its current motion information without its neighbour information. For example, a PREMON packet with 7 bytes long is composed of current motion information of 5 bytes and neighbour information of 2 bytes, while a POTL packet has 5 bytes long only. Using the same value is merely for the demonstration purpose.

From Equations (3.4)-(3.6), the energy consumptions in *Naïve, PREMON* and *POTL* are calculated respectively with different value of prediction accuracy. Obviously prediction accuracy has much effect on the power consumptions.

Figure 3.2 illustrates the effect of prediction on the power consumption. In a naïve system, it stays the same as the prediction accuracy varies from 0 to 1, but in PREMON and POTL, as the prediction accuracy increases, the power consumption declines significantly. It is easy to understand----incorrect prediction causes re-monitoring and re-calculation overhead, which leads to the higher power consumptions. It can be also observed that POTL needs much lower accurate prediction model to consume the same amount of power as PREMON does. The performance comparison shows only when the prediction model's accuracy is approximately above 70%, the PREMON can save power than a naïve system, though POTL can save power even if the prediction model achieves only 5% accuracy. This is because, for PREMON, the reduction of transmitting readings from sensor nodes is at the cost of transmitting motion information whenever the time slot arrives, thus the predictions overhead is much bigger than that of POTL.

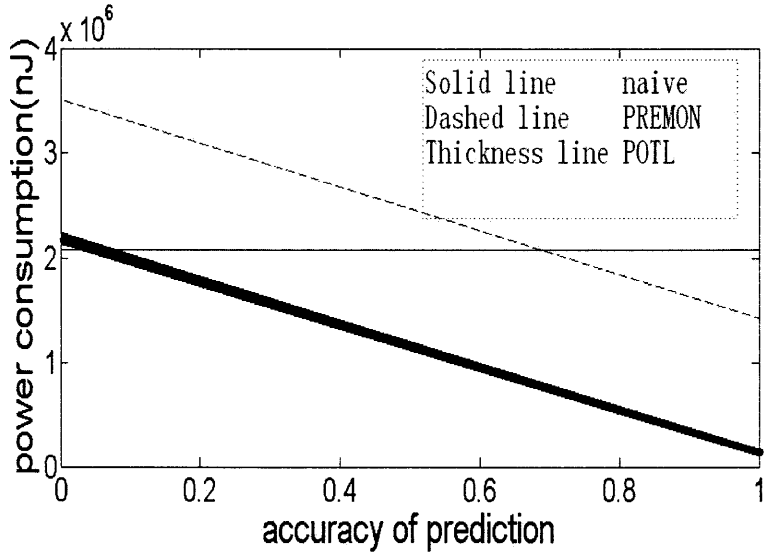


Figure 3.2 Power consumptions of three algorithms.

3.5 Summary

This chapter proposes a prediction based approach for minimizing power consumption in wireless sensor networks---POTL. This chapter makes three main contributions. First the effects of prediction accuracy on power consumption have been studied and presented. Second, this chapter describes an approach of routing and scheduling to reduce the power consumptions. Third, load balance is taken into concern to utilize redundant devices without consuming much higher total energy but significantly improve the reliability and lifetime of the whole networked sensor system.

The following chapter studies the effect of tracking interval on the power consumption given the known prediction accuracy curves.

CHAPTER 4

OPTIMAL TRACKING INTERVAL FOR SENSOR NETWORKS

4.1 Introduction

An important application of wireless sensor networks is the tracking of moving objects. Prediction-based techniques have been proposed to reduce the power consumption in wireless sensor networks by limiting the sensor active time. This chapter proposes a quantitative method to optimize the power efficiency by analyzing the effect of prediction on the energy consumption in wireless sensor networks. This work is the first attempt to calculate the optimal tracking interval for a given predictive tracking algorithm. Based on this method, the lifetime and power efficiency of the sensor networks can be effectively improved.

Object tracking is an important application in wireless sensor networks, e.g., terrorist attack detection and traffic monitoring. Since a node in wireless sensor networks is battery-powered, power efficiency demands much concern. Object tracking consists of detecting and monitoring locations and motions of real-world objects.

There has been much research on object detection and tracking with sensor networks. Most of them concentrate on tracking objects and finding efficient ways to forward the data reports to the sinks. Goel and Imielinski [24] argue that readings at a sensor node can be predicted based on the past reading history and spatio and temporal relationships of readings from surrounding sensors. Predictive tracking algorithms accommodate the sensor-hibernation mechanism to save energy and prolong the sensor network's lifetime. At any time, only those sensors that are chosen to sense the targets need to be active, while all others can still hibernate. This is made possible by predicting the target's next location. While the "putting-sensor-into-hibernation" mechanism does save energy, it also induces the overhead of information collection and computing power consumption.

Obviously the accuracy of prediction depends on the "tracking interval". This work defines the tracking interval as the time length between two consecutive sensing points with the intuition that as the resolution becomes finer, the miss probability decreases. As the tracking interval becomes lower, in other words "more frequent", the tracking power consumption increases. As it increases, the miss probability increases, thereby lowers the tracking quality. The failure of locating a target must be recovered by a certain mechanism, which leads to extra power consumption for re-capturing the target. Hence, an analytical framework is highly desired in order to determine an optimum value of tracking intervals. This chapter intends to 1) propose a quantitative analytical model to find such an optimal tracking interval, 2) study the effect of the tracking interval on the miss probability, and 3) propose a scheme called Predictive Accuracy-based Tracking Energy Saving (PATES) by making good tradeoff between the accuracy and cost of sensing operation.

4.2 Predictive Tracking Sensor Network Architecture

4.2.1 Object Tracking Sensor Networks

An object tracking sensor network refers to a wireless sensor network designed to monitor and track the mobile targets in the covered area [21]. Generally, each sensor consists of three functional units: Micro-Controller Unit (MCU), sensor component and

RF radio communication component. To facilitate the energy conservation, most of today's sensor nodes allow these three basic components to be inactivated separately when they are not needed. These sensor nodes are responsible for tracking any mobile objects that intrude the covered region, and reporting the properties of the moving targets to the applications in a specified frequency. The sensor nodes sample the environment for a certain interval, referred to as *sampling duration,* to obtain the properties of the moving objects. During sampling, the MCU and sensor components are activated for data collecting and processing, while the radio components can be turned off if no communication is needed. The sampling happens with certain tracking interval adjustable based on the prediction and application requirements.

4.2.2 Predictive Accuracy-based Tracking Energy Saving

Based on the application requirements in the object tracking sensor networks described before, a scheme called Predictive Accuracy-based Tracking Energy Saving (PATES) is proposed to improve the power efficiency. In PATES, three modules must be in use.

1) *Monitoring and tracking*:

A sensor node monitors and collects the moving information of the tracked objects, and then reports to the base station. According to the prediction received from the base station, a sensor node is activated only when the object is supposed to enter its detected area with a given tracking interval that is also included in the packets from the base station. By monitoring the mobile objects, the sensor nodes are able to decide whether or not the prediction reflects the real states of the object movements. If the prediction is consistent with the monitored states, the sensor nodes continue monitoring and tracking the object with the certain resolution. If the prediction is not consistent with the real trajectory of the object, i.e., the targets are missed, then the *recovery* module is initiated

2) *Prediction and reporting:*

Initially each sensor node reports the moving history of the objects to the base station. On the server side predictions are made and sent by the base station based on the moving history. Upon receiving the prediction packets, each sensor node is informed of the tracking interval and what time it is supposed to be activated for monitoring the target. Meanwhile information about each sensor itself, such as residual power level, is communicated between the base station and each sensor through low-power paging channel periodically [6]. If during tracking interval, the monitored states of the moving objects are consistent with the prediction, no updates need to be sent to the base station, which will save considerable power by reducing the long distance communication. Although the base station receives no packets through the transmission channel, it can determine whether the power is depleted or not by periodically paging through low-power paging channel. Otherwise, the sensor networks are responsible for recovering the failure of monitoring resulting from the inaccurate predictions.

3) Recovery:

No matter what prediction algorithm is used, it is impossible to guarantee 0% missing probability. Therefore, a recovery mechanism is necessary to relocate the target when the object is lost. To be conservative to the energy resource, the recovery module is divided into two stages: a) ALL_NBR recovery. All the neighbor nodes around the current sensor are waked up to make up the deficiency of the prediction. When the current sensor realizes that the target is not in its covered area, it broadcasts and wakes up all the nodes around it. This reduces the probability of object missing, though it is still not guaranteed that the missed objects be relocated by the neighbor nodes. A more aggressive approach takes place at the second stage if the object is not found. The sensor node that captures the missed object has to not only notify the current node, thus preventing the second stage of the recovery, but also report the updated moving information to the base station. b) ALL_NODE recovery, the current sensor wakes up all the nodes in the network for object relocation, which ensures the maximal probability of re-capturing the target. The sensor node that captures the target has to report to the base station so that the updated prediction can be made and sent out by the base station.

4.3 Power Optimization and Quantitative Analysis

Depending on the sensor topology and application, several different prediction algorithms for tracking mobile objects have been reported in the literature [25]. Obviously, different prediction algorithms result in different tracking accuracy. Hence, the relationship between missing probability and tracking interval varies with the prediction algorithms. Furthermore, the missing probabilities of a specific prediction algorithm with the same tracking intervals on a mobile object with different moving pattern are different too. Based on the simulations, it can be observed that relation between missing probability and tracking interval, in the concerned span, can be fitted into a quadratic function $P(s) = as^2 + bs + c$, where s is the tracking interval, a, b, and c are the constants, Since probability, $P(s)$, is positive for any s (>0), the constant part c must be greater than zero. For the convenience of description this chapter makes a simple prediction model, Heuristic AVERAGE, as an example, in which the current location and velocity of the mobile object is derived from the average of the object movement history. The result of a simulation is shown in Fig. 4.1, in which a walking person whose average speed is 1mile/hour tracked by a sensor network with 100 sensor nodes under the Heuristic AVERAGE prediction algorithm. The values of tracking intervals above 10 sec are screened out of the curve fitting since they are beyond the accurate tolerance and should be ignored. From the curve fitting in Fig.4.1, the approximate function of missing probability can be written as $P(s) = 0.0013s^2 + 0.025s + 0.062$, based on the quadratic fitting shown with the solid line in Figure 4.1.

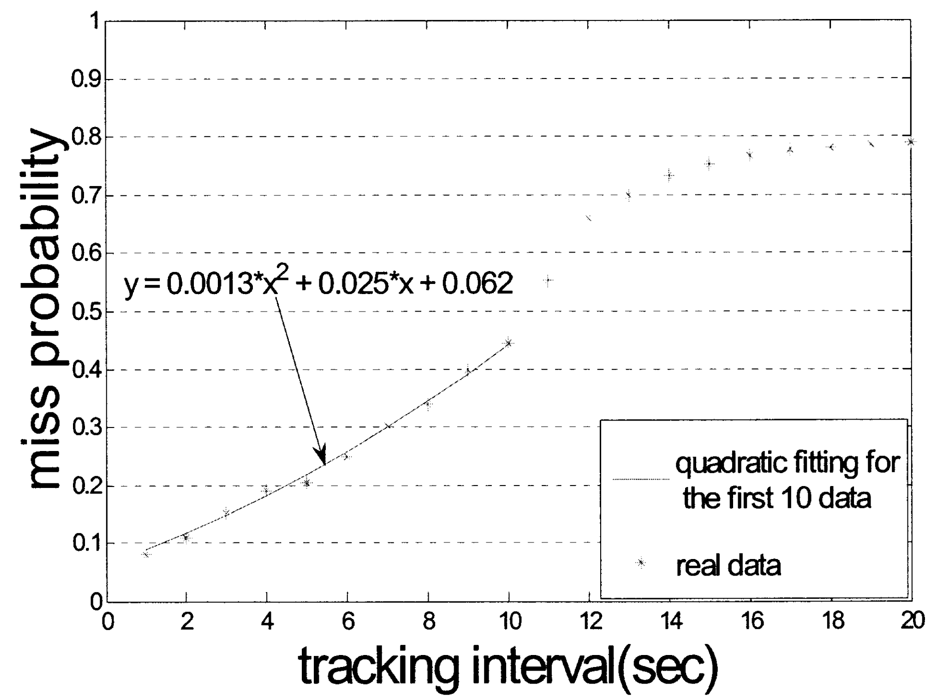


Figure 4.1 The impact of tracking interval on prediction accuracy.

A more rigorous study of the statistical characteristics of the missed targets is presented here to find the power consumed by the network. During each tracking interval, the expression of the total energy consumed by a sensor network is given by

$$E_{span} = [1 - P(s)] \cdot E_{success} + P(s) \cdot E_{recovery} \quad ,$$

where $E_{success}$ is the energy consumed by the sensor node if the prediction is accurate and $E_{recovery}$ is the energy for the failure recovery.

$$E_{success} = E_{sensor-active} + E_{MCU-active}$$

, where the energy consumed by the low power paging channel is negligible.

$$\begin{aligned} E_{recovery} = {} & p_{1st-recovery}[m \cdot (E_{sensor-active} + E_{MCU-active}) + E_{report2bs} + E_{notification}] \\ & + (1 - p_{1st-recovery})[N \cdot (E_{sensor-active} + E_{MCU-active}) + E_{report2bs}] \end{aligned} \tag{4.1}$$

, where m is the number of the neighbor sensors around the current sensor node, N is the total number of sensors in the whole network, $p_{1st-recovery}$ is the probability that the first stage recovery succeeds, and $E_{sensor-active}$ is the energy consumption of the sensor when it is active. $E_{MCU-active}$ is the energy consumption of an MCU when it is active, $E_{report2BS}$ is the energy consumption for reporting the discovery of object from the sensor to base station, and $E_{notification}$ is the energy needed for notifying the current sensor of the success in relocating the object.

Observed from the Equation (4.1), $E_{recovery}$ is always strictly greater than $E_{success}$. As described in Section 4.2, the recovery module consists of two stages. If first recovery fails, the second stage recovery mechanism is activated and all nodes are waked up. In the first recovery stage only the neighbor sensor nodes around the current node are activated, and when a neighbor node detects the target, it sends a notification to the current node to avoid the current sensor from initiating the 2nd stage recovery request to all the sensors while reporting the actual moving states to the base station. Thus during a given period T, the total energy consumed by all the sensor nodes can be written as,

$$\begin{aligned} E_{total} &= \frac{T}{s} \cdot E_{span} = \frac{T}{s} \cdot [(1 - P(s)) \cdot E_{success} + P(s) \cdot E_{recovery}] \\ &= \frac{T}{s} \cdot [E_{success} + (as^2 + bs + c)(E_{recovery} - E_{success})] \end{aligned}$$

Now the optimum value of tracking interval can be calculated by setting the derivative of E_{total} with respect to s to zero

$$\frac{d}{ds}(E_{total}) = \frac{d}{ds}\{\frac{T}{s} \cdot [E_{success} + (as^2 + bs + c) \cdot (E_{recovery} - E_{success})]\} = 0$$

The minimum value of E_{total} is proved to exist by using second derivative test:

$$\frac{d^2}{ds^2}(E_{total}) = \frac{2T}{s^3}[E_{success} + c \cdot (E_{recovery} - E_{success})] > 0$$

Since $c > 0$, $E_{recovery} > E_{success}$, hence,

$$s_{opt} = \sqrt{\frac{c}{a} + \frac{E_{success}}{a(E_{recovery} - E_{success})}}$$

exists as a positive real number under the condition of $a > 0$, which means if and only if the second degree's coefficient is greater than zero, the optimality of energy consumption can be obtained.

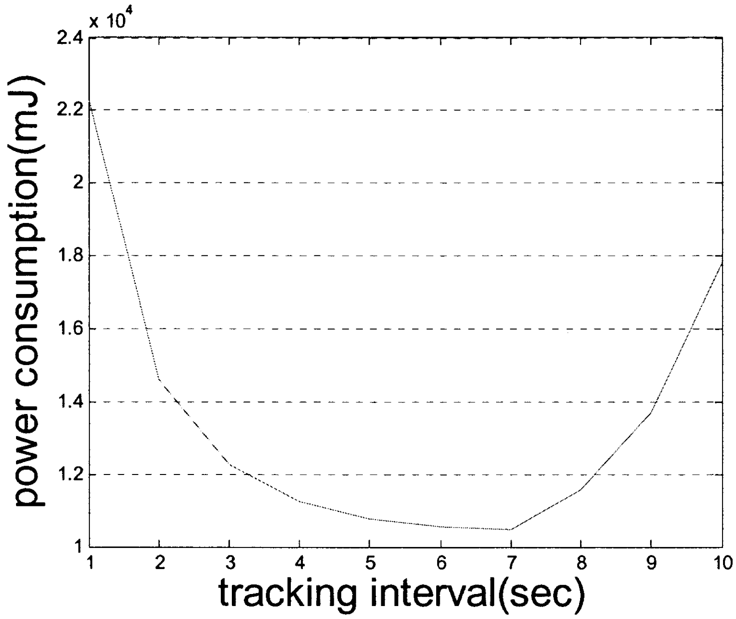


Figure 4.2 Power consumption vs. tracking interval.

For these experiments, N=100 nodes, m=7 nodes, $E_{recovery}$ =9656mJ, $E_{success}$ = 42mJ, and T=25 seconds, so the optimal tracking interval is 7.1451 seconds. Fig. 4.2 shows the relationship between the power consumption and tracking interval. The power consumption with tracking interval at about 7 sec is the minimum. From Fig. 4.2, it can be concluded that choosing an optimum value considerably saves power consumptions for tracking mobile objects in wireless sensor networks.

It should be noted that in Figure 4.2 the energy consumed by the low-power paging channel is excluded, as it is almost negligible in the calculation. While doing simulations, it is also observed that the miss probability is inversely proportional to the target's speed. This implies that a fast moving object needs a small tracking interval to counter the rise in the miss probability. Since one can identify the concerned span of tracking intervals in which the curve can be quadratic fitted, this proposed work can be well applied. In summary, the power consumption with respect to tracking intervals can be minimized with a quadratic miss probability function under a given prediction algorithm. A predictive tracking scheme to optimize the power efficiency with two stages of recovery is proposed. The method can successfully make good tradeoff between the prediction accuracy and tracking cost.

CHAPTER 5

ADAPTIVE OPTIMAL SENSOR PLACEMENT AND BOUNDARY ESTIMATION

5.1 Introduction

Wireless sensor networks have been under intensive studies. Sensor networks become a bridge between the physical world and the information systems. Since the set covering problem is NP-hard, where NP stands for non-polynomial, optimal sensor placement problem that is equivalent to the former is also NP-hard. Hence, computationally efficient sensor placement approaches with maximal detection probability are highly desired.

Object tracking in sensor networks received much research attention recently. Most of the work focuses on identifying and tracking one or more individual objects [6][8][27]. Unfortunately, sufficient research efforts have yet to conduct on the issues of monitoring and tracking a large number of objects in sensor networks. It is in demand that a large number of objects be monitored and tracked concurrently by a sensor network. There are some examples of monitoring and tracking a large number of objects, such as a surveillance system to monitor many people in a public area, vehicles in a highway, and wild animals. These objects with large population, referred as *mass objects* in this chapter, are usually distributed in a certain way because several interest points are attractive to them and thus located densely. In many real world applications, it is necessary to place sensors optimally to locate the mass objects with the maximal detection probability and resolution by deploying a limited number of sensors only. Figure 5.1 illustrates an example of a sensor network monitoring a large number of trees and vehicles in a wild area. In such cases, this work focuses on the locations of mass objects instead of individual ones. Even though the individual objects are moving frequently, the topology of the mass objects is less likely to change very fast.

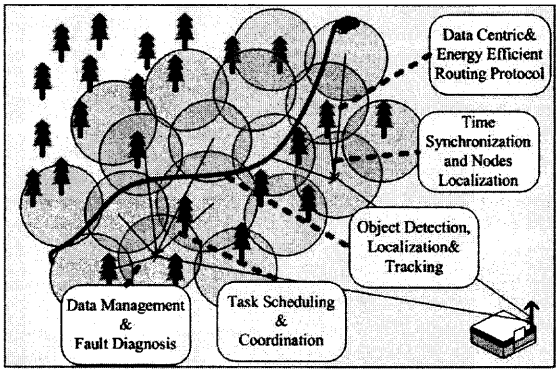


Figure 5.1 An example of sensor networks for mass objects monitoring.

With the recent advances in sensor technologies, people are allowed to make use of mobile sensors, which can move to the correct places to provide the required coverage. They are used to detect targets collaboratively and monitor environments across the area of deployment. As the mass objects' topology changes slowly, they are capable of acknowledging and moving to the desired place. The coverage may be inferior to the application requirement. Location estimation for mass objects based on signal strength received from sensors have been proposed and implemented in [28][29][30]. The positions are calculated by modeling signal propagation, which requires adequate signal detection. Hence, one of the most important issues in sensor networks for monitoring and tracking mass objects is the selection of sensors locations. Proper sensor placement is needed to provide adequate signal coverage and also maximize the probability of accurate detection and localization of the whole mass of objects. Sensor placement directly influences resource management and the type of back-end processing and exploitation that must be carried out with sensed data in distributed sensor networks. A key challenge in sensor resource management is to determine a sensor field architecture that minimizes cost [39], and provides high sensor detection and resilience to sensor failures. The detection optimization is inherently probabilistic due to the uncertainty associated with sensor coverage. This chapter proposes a methodology for maximizing the detection probability under the constraints of the limited number of sensors and limited signal strength. To the best knowledge, this work is the first effort that adaptively selects the sensor locations, maximizes the detection probability and estimates the dynamical probabilistic boundary for mass objects through an online unsupervised learning method.

The rest of this chapter is organized as follows. Section 5.2 briefly reviews the related works. Section 5.3 formalizes the problem of optimal sensor placement, and analyzes the distribution of mass objects. The detection coverage is modeled as Gaussian Mixture Model. The Expectation-Maximization and recursive Expectation-Maximization solutions are presented to optimize the coverage problem based on Gaussian Mixture model in Section 5.4. Section 5.5 proposes a possible choice for the distributed implementation of EM algorithms in sensor networks. The details of simulation and detection performance are discussed in Section 5.6. Section 5.7 concludes this chapter.

5.2 Related Work

There has been much research on sensor deployment. Wang *et al.* [36] propose a movement-assisted deployment approach based on Voronoi diagrams to find coverage holes and move sensors from densely deployed area to sparse area in order to improve coverage of sensor networks. With uncertainty-aware topology of sensor placement, Zou *et al.* [37] develop a probabilistic model that is targeted at average coverage as well as at maximizing the coverage of vulnerable areas. Lin *et al.* employ a simulated annealing method to place the sensors in a min-max optimization model in [38]. All these methods aim at covering monitored areas with optimal or near-optimal efficiency and accuracy. In real scenario, the essential goal is to locate the targets that are usually moving. Considering the problem of optimally detecting thousands of targets, if the mass characteristics of the targets can be learned dynamically, an optimal sensor placement strategy can be achieved accordingly. Hence the "target-oriented" approaches are more powerful than "area-oriented" ones.

The challenge here is to estimate the mass characteristics of objects. Nowak [35] proposes a distributed version of EM algorithm to estimate the density of targets for sensor networks. Although the density can be well estimated, this approach is not applicable to dynamic scenarios because the estimates are static and cannot be updated according to the dynamics of mass characteristics. The recursive EM algorithm [33] is proposed to update the estimates of observations dynamically. It is believed that a well-designed distribution implementation of a recursive EM algorithm can provide an applicable solution to optimal sensor placement for dynamically locating the mass objects.

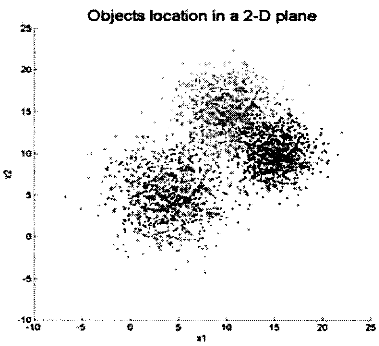
5.3 Problem Formulation

Accurate and computationally feasible sensor detection models are required for optimal sensor deployment. This work starts with the assumption used in [31] that the probability of detection of a target by a sensor varies exponentially with the distance from it to the target. In other words, a target is correctly detected by a sensor distance d away with probability of $e^{-\rho d}$, where ρ is a parameter used to model the quality and the rate at which its detection probability diminishes with distance. Obviously, the detection probability is equal to 1 if the target locates exactly where the sensor does. Recognizing that the covered regions are usually overlapped, an object may therefore be detected by several sensors. The probability of an individual object being precisely detected should be the mixture probability that sums up the conditional detection probability of a certain sensor multiplied with its mixture weight.

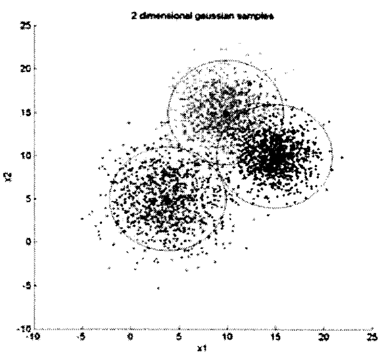
In many practical instances, objects are symmetrically distributed around the point of interest. Such cases provide us with assumptions that the probabilistic model of locations where objects appears should be Gaussian. To improve the individual object's detection and monitoring capability, the sensor has to be placed closer to the target. To monitor mass objects, this work proposes a method to reduce the sum of distances between sensors and all the objects in the covered area. Based on the above assumptions and analysis, the sensors should be located at the position with the local maximal object density in order to maximize the detection and monitoring performance. The observations of objects' previous locations collected by sensors therefore can be used to estimate and learn the centers of object clusters and their boundaries. As an example, consider three groups of mass objects distributed in a 2-D covered area illustrated in

Figure 5.2. Based on the observations, there must be at least three points (regions) of interests, and each of them must be covered by at least one sensor.

Scatter points shown in Figure 5.2 are observations of object locations collected by sensors. Note that the observations are the previous positions where individual objects are located. During an interval of information collection, each individual object may appear at several different locations. With the objective to monitor mass objects, the proposed method focuses rather on the distributions of clusters than any individual object's movement. In this sense, one individual object may correspond to several observations because of the uncertainty in their movement.



(a)



(b)

Figure 5.2 (a) Mass objects locations, and (b) Sensor placement.

The observations, recording all locations where objects have been, are a good clue of building a probabilistic model that describes how likely the objects appear at a specific location. Here an analogy of "electron cloud" can be taken, each point in the Gaussian mixture does not necessarily mean that some object appears at that location currently. Rather, it describes how likely objects appear at the position. The observations are simply a combination of previous locations where objects have happened to be. For example, there are 1000 observations, shown as scatter points in Figure 5.2(a), but in fact there may be only 500 objects present in the area. Sensors do not have to identify each individual object and to track how and when it moves. Instead, the objective of this work is to analyze the historical observations of object locations, learn the maximum likelihood parameters for Gaussian mixtures and estimate centers and boundaries of all clusters. In summary, this work considers the problem of optimizing sensor placement for locating mass objects as a group in a statistical way rather than identifying and deterministically analyzing individual objects' movement. Figure 5.2(b) illustrates an example of optimal sensor deployment to maximize the coverage performance given three sensors. The central points are the positions of sensors, and their covered areas are enclosed by three circles.

In this section, this problem is formulated in statistics. Suppose that a set of N observations $\{z_i\} = \{(x_{i,}y_i)\} \in S, i \le N$ is given, which are previous locations of objects. In real scenarios, they are likely to be distributed in Fig. 5.2(a). The task of this work is to learn the statistical parameters of each cluster and then move the sensors to the cluster's center which has the largest probability density in the cluster. The sensor detection probability for mass objects can be maximized if the sensor that is responsible for covering each cluster is placed at the center of the cluster where mass objects are most likely to appear, i.e., the position with maximal probability density in each cluster. In Gaussian mixture model, the proposed method uses the following expression to approximate the real observation distribution:

$$p(z_i) = \sum_{j=1}^{M} \alpha_j p_j(z_i \mid \theta_j) \tag{5.1}$$

where $p(z_i)$ is the mixture probability density of *i-th* observation, i.e., the probability density of the observation i in the mixture of Gaussian distributions. The parameter $\theta = (\alpha \ \mu \ \sigma)$, is the combination of mixture weight, mean value and variance of a cluster which describe the distribution of a single component of Gaussian mixture (i.e. cluster). Θ, to be used later, is the whole set of $\theta's$, α_j is the mixture weight, which means the probability that observations appear in the j-th cluster and $p(z_i \mid \theta_j)$ is the conditional probability density of *i-th* observation with respect to cluster j, accurately detected by the j-th sensor given that the location of the sensor j is known and object i is covered by sensor j. M is the number of mixture components, in other words it is the number of clusters, and N is the number of observations.

In order to model the location distribution of these objects, obviously the tracking algorithms can take advantage of the limited observations previously collected by the sensors. However, they are referred as *incomplete data* due to the lack of the cluster label information. Based on the limited knowledge, the maximum likelihood estimate of the parameters of an underlying distribution is desired from a given data set when the data is incomplete. The incomplete-data log-likelihood expression is as follows:

$$Q(\Theta) = \log(\mathrm{L}(\Theta \mid \mathrm{Z})) = \log \prod_{i=1}^{N} p(z_i \mid \Theta) = \sum_{i=1}^{N} \log\left(\sum_{j=1}^{M} \alpha_j p_j (z_i \mid \theta_j) \right) \tag{5.2}$$

where the likelihood function can be defined as

$$\mathrm{L}(\Theta \mid \mathrm{Z}) = p(\mathrm{Z} \mid \Theta) = \prod_{i=1}^{N} p(z_i \mid \Theta) \tag{5.3}$$

This function is also called the likelihood of the parameters given the observations. The likelihood function of parameter estimates is in fact a measure of coverage likelihood for mass objects by limited sensors. In Equation 5.3, it is estimated from the product of mixture probability density function and the assumption of Independent, Identical Distribution of individual observations.

The optimization problem for coverage and detection now becomes the maximization of the likelihood function.

$$\Theta = \arg\max_{\Theta} Q(\Theta) \tag{5.4}$$

By solving the above equation, the proposed method can find the positions with local maximal density of each region of interests. They are the optimal locations to place sensors. To update dynamically the optimal sensor placement, the *maximum a posteriori* solution (MAP) can be used to estimate the dynamic distribution of objects. Similarly the model selection techniques are based on maximizing the following type of criteria:

$$J(M, \theta(M)) = \log(\mathrm{L}(\Theta \mid \mathrm{Z})) - P(M) \tag{5.5}$$

where $\log(\mathrm{L}(\Theta \mid \mathrm{Z}))$ is the log-likelihood of the available data. This part can be maximized using maximum likelihood (ML) solution as mentioned above. However, introducing more sensors, hence increasing the mixture components always increase the log-likelihood but also introduce unnecessary redundant sensors. A penalty function

$P(M)$ is thus introduced to achieve the balance. $P(M)$ is a function of M and increases as the number of clusters, M, increases. In this work, it is $\prod_{m=1}^{M} \alpha_m^{c_m}$ which monotonically increases with M.

5.4 Standard EM and Recursive EM Algorithm for Optimal Sensor Detection

5.4.1 Standard EM Algorithm for fixed infrastructure

The Expectation-Maximization (EM) algorithm is an iterative procedure that searches for a local maximum of the log-likelihood function. In order to apply the EM algorithm for detection optimization in sensor networks, the EM algorithm starts with initial observations and parameter estimate θ_0. The estimate θ_k from the *k-th* iteration of the algorithm is obtained using the previous estimate θ_{k-1}:

$$Q(\theta_k \mid \theta_{k-1}) = E(\log p(Z_{i \le k-1}, z_k \mid \theta) \mid Z_{i \le k-1}, \theta_{k-1}) \tag{5.6}$$

This step, referred to as "expectation step (E-step)", finds the expected value of the "complete-data" log-likelihood with respect to the unknown parameters given the observed positions and current parameter estimates. The above E-step equation can be expanded as follows:

$$\begin{aligned} Q(\Theta \mid \Theta^g) &= \sum_{z \in S} \log(\mathrm{L}(\Theta \mid Z^g, z)) p(z \mid Z^g, \Theta^g) = \sum_{l=1}^{M} \sum_{i=1}^{N} \log(\alpha_l p_l(z_i \mid \theta_l)) p(l \mid z_i, \Theta^g) \\ &= \sum_{l=1}^{M} \sum_{i=1}^{N} \log(\alpha_l) p(l \mid z_i, \Theta^g) + \sum_{l=1}^{M} \sum_{i=1}^{N} \log p_l(z_i \mid \theta_l) p(l \mid z_i, \Theta^g) \end{aligned} \tag{5.7}$$

where it is assumed that l is a random variable that label which region an individual object belongs to, and the superscript g means that the referred parameter is available from the previous iteration [32]. Given Θ^g the algorithm can easily compute $p_j(z_i \mid \theta_j^g)$ for each observation i and cluster j. In addition, the mixture parameters, α_j can be thought of as prior probabilities of each mixture component, that is $\alpha_j = p(component\ j)$, which are uncorrelated with the observations of z_i. Therefore, by applying Bayes's rule, recursive EM algorithm can compute the "ownership function" [32]:

$$p(l \mid z_{i,}\Theta^g) = \frac{\alpha_l^g p_l(z_i \mid \theta_l^g)}{p(z_i \mid \theta_l^g)} = \frac{\alpha_l^g p_l(z_i \mid \theta_l^g)}{\sum_{k=1}^{M} \alpha_k^g p_k(z_i \mid \theta_l^g)} \tag{5.8}$$

To maximize the expression, $\Theta = \arg\max_{\Theta} Q(\Theta \mid \Theta^g)$, the algorithm can maximize the term containing α_l and the term containing θ_l independently since they are not related. This step is referred to as Maximization step (M-step). The proposed method introduces the Lagrange multiplier λ with the constraint that $\sum \alpha_l = 1$, and solve the equation:

$$\frac{\partial}{\partial \alpha_l}\left[\sum_{l=1}^{M}\sum_{i=1}^{N} \log(\alpha_l) p(l \mid z_i, \Theta^g) + \lambda(\sum_{l} \alpha_l - 1)\right] = 0 \tag{5.9}$$

The following solution is obtained,

$$\lambda = -N\,,\ \alpha_l^{new} = \frac{1}{N}\sum_{i=1}^{N} p(l \mid z_i, \Theta^g) \tag{5.10}$$

Taking the derivative of the second term of Equation (5.7) with respect to μ_l and setting it to zero, then the other two parameters can be obtained:

$$\mu_l^{new} = \frac{\sum_{i=1}^{N} z_i p(l \mid z_i, \Theta^g)}{\sum_{i=1}^{N} p(l \mid z_i, \Theta^g)} \tag{5.11}$$

$$\sigma_l^{new} = \frac{\sum_{i=1}^{N} p(l \mid z_i, \Theta^g)(z_i - \mu_l^{new})(z_i - \mu_l^{new})^T}{\sum_{i=1}^{N} p(l \mid z_i, \Theta^g)} \tag{5.12}$$

Note that the parameters $(\alpha^{new}, \mu^{new}, \sigma^{new}) = \theta^{new}$ calculated in M-step will be substituted as Θ^g into E-step to compute $p(l \mid z_i, \Theta^g)$ and then $p(l \mid z_i, \Theta^g)$ substituted into M-step to get the new parameter θ^{new}. The EM algorithm usually converges to a local maximum of the log-likelihood function. Hence it is a good choice for mixture estimation and especially distributed (and unsupervised) applications like the mixture of objects distributions in sensor networks. Because of the distributed property of sensor networks, practical and feasible sensor networks prefer distributed computation over a centralized process. A distributed implementation of the EM algorithm applied into sensor networks is further described in Section 5.5.

5.4.2 Recursive EM Algorithm for Dynamic Topology

The recursive EM algorithm is an online discounting version of EM algorithm. A stochastic discounting approximation procedure is conducted to estimate the parameters recursively and adaptively. In real-time monitoring and tracking of dynamic objects' topology in sensor networks, recursive EM algorithm is better in the sense that the new observation updates the parameters estimate of the mixture with a forgetting factor degrading the influence of the out-of-date samples on the new estimates. Hence sensor networks implemented with recursive EM monitoring techniques can track better the topological dynamics of mass objects. Unlike the standard EM using maximum likelihood estimate, recursive EM searches for MAP solution as mentioned in Section 3. A brief description of recursive EM algorithm is given below, and its detailed one can be found in [33].

Recalling from Section 5.3, the following type of criteria should be maximized.

$$J(M,\theta(M)) = \log(\mathrm{L}(\Theta \mid \mathrm{Z})) - P(M) = \log p(Z \mid \theta(M)) + \log p(\theta(M)) \qquad (5.13)$$

where this work introduces a prior $\log p(\theta(M))$ for the mixture parameters that penalize redundant sensor and complex solution. For the MAP solution, it can be expressed as,

$$\frac{\partial}{\partial \alpha_m}(\log p(Z \mid \theta) + \log p(\theta) + \lambda(\sum_{m=1}^{M} \alpha_m - 1)) = 0$$

where $p(\theta(M)) \propto \exp \sum_{m=1}^{M} c_m \log \alpha_m = \prod_{m=1}^{M} \alpha_m^{c_m}$ is a Dirichlet prior [33]. For t data samples,

$$\alpha_m^{(t)} = \frac{1}{K}\left(\sum_{i=1}^{t} p(l \mid z_{i,}\Theta^{g}) - c\right) \qquad (5.14)$$

where $K = t - Mc$, and the parameters of the prior are

$c_m = -c = -N/2$. If it is assumed that the parameter estimates do not change much when a new observation is added and the new ownership function $p(l \mid z_{i,}\Theta^{t+1})$ can be approximated by $p(l \mid z_{i,}\Theta^{t})$, the following recursive update equation can be obtained:

$$o_m^t(z^{t+1}) = p(l_m \mid z_{i,}\Theta^{t+1}) = \frac{\alpha_m^t p_m(z^{t+1} \mid \theta_m{}^t)}{p(z^{t+1} \mid \Theta^t)} \qquad (5.15)$$

$$\alpha_m^{t+1} = \chi\left(\frac{o_m^t(z^{t+1}) - c_T}{1 - Mc_T}\right) + (1-\chi)\alpha_m^t \qquad (5.16)$$

where $\chi = 1/T$ is a fixed forgetting factor that is used to forget the out-of-date statistics more rapidly. It is equivalent to introducing an exponentially decaying envelope: $\chi(1-\chi)^{t-i}$ being applied to the influence of the old observation z^{t-i}. After the new estimated mixture weight of each sensor is calculated, the online algorithm should check if there are irrelevant components to make sure that no unnecessary redundant sensor is used: If the mixture weight α_m^{t+1} <0, discard the component *m*, set *M=M-1* and renormalize the remaining mixture weights. As mentioned before, this mechanism of discarding irrelevant component is achieved by introducing a penalty function. It is straightforward that the penalty function always decreases with fewer clusters, eventually the clusters could merge to fewer necessary ones. The rest of the parameters are then updated as follows

$$\begin{aligned} &\mu_m^{t+1} = \mu_m^{\ t} + w\delta,\ \sigma_m^{t+1} = \sigma_m^t + w(\delta\delta^T - \sigma_m^t) \\ &\text{where } w = \chi\frac{o_m^t(z^{t+1})}{\alpha_m^t},\ \delta = (z^{t+1} - \mu_m^{\ t}) \end{aligned} \tag{5.17}$$

The new parameter estimates are generated as output. From the most recent estimates, sensors can slightly tune their locations to the updated local maxima and wait for new observations to make next updated estimates iteratively.

5.5 Distributed Implementation of Optimal Sensor Placement

Assume that initially sufficient nodes are evenly distributed over the entire area. All nodes have the local estimates on their observations. The next EM iteration θ^{t+1} can be computed by performing two message passing cycles through the sensor nodes. Each message passing operation involves the transmission of the sufficient statistics from one node to another based on a prescribed sequence through the nodes.

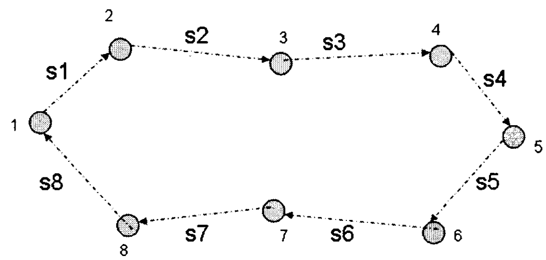


Figure 5.3 Communication cycle for message passing.

Each node computes its local updates for the sufficient statistic. Note that these local updates are computed from local observations and estimates only available locally at each sensor node. In the forward path, in the prescribed cyclic order from 1 to H to 1 as illustrated in Figure 5.3, where the number of sensor $H = 8$. It is assumed that sufficient number of sensors is provided to be more than the number of clusters, i.e. $H > M$. Each sensor increments the local estimates to the old cumulated estimates, and passes the new cumulated estimates to the downstream sensors.

$$\theta_j^t = \theta_j^t + \theta_{h,j}^t, \tag{5.18}$$

where θ_j^t is the cumulative estimate of cluster j at time t, and $\theta_{h,j}^t$ is the local estimate of cluster j by sensor node h at time t. This process is based on the fact that the E-step can be separated into H separate expectations followed by accumulation [35]. At the last sensor node H, the summary of complete sufficient statistics is available and passed over to node 1. Thus, the summary of sufficient statistics is passed in the circle incrementally. After two message-passing cycles, the incremental form of cumulative statistics of observations collected by all nodes reaches every node in the circle. Therefore, all H

sensor nodes have the sufficient statistics and can compute the M-step to obtain θ^{t+1}. This process is guaranteed to monotonically converge to a local maximum [35].

Figure 5.3 illustrates communication cycles in distributed implementation of the recursive EM algorithm for wireless sensor networks. The small circles in Fig. 5.3 indicate the sensors deployed in the target area. The forwarded messages in the algorithm proceed in a cyclic fashion with a predetermined order, i.e. messages are passed between nodes in the order of $1,2,...,H,1,2...,H,1,...$.

Similarly, the recursive EM algorithm is implemented in a distributed manner to fit the requirements of sensor networks by taking advantage of the accumulation of separate E-steps as well. Each sensor node updates its local parameter estimates in a recursive form by taking all the new local observations as inputs and then passes the local estimates with cumulative estimates to the downstream one. As long as the global statistics is reached after two message-passing cycles, each node can tune the parameter estimates in their M-steps. By recursively adding new observations, the distributed recursive EM algorithm can tune the parameter estimates to catch up with the change of the topology of mass objects adaptively. The following part describes the distributed version of recursive EM algorithm for sensor deployment.

Initially, H sensors are distributed evenly throughout the covered area, each node collects a set of observations and makes its own local estimates. The cumulative estimates are incremented with the local estimates of each node in a cyclic manner. Note that it takes several message-passing cycles to converge to a local maximum, and the EM algorithm can determine the convergence of the estimates by checking whether or not the increase of the likelihood function in the current iteration is greater than previous likelihood function multiplied with a small coefficient ε (e.g., $\varepsilon = 10^{-5}$), i.e. $L(\Theta^t \mid Z) - L(\Theta^{t-1} \mid Z) > \varepsilon \left| L(\Theta^{t-1} \mid Z) \right|$

Algorithm 5.4.1

Initiation:
While ($L(\Theta^t \mid Z) - L(\Theta^{t-1} \mid Z) > \varepsilon \left| L(\Theta^{t-1} \mid Z) \right|$) **do**
  For node $h = 1$ to H
    Compute $\theta_{h,j}^{initial}$: local estimate for each cluster j
    Increment the cumulative statistics for each cluster j with local estimate $\theta_j^{initial} = \theta_j^{initial} + \theta_{h,j}^{initial}$
    If ($h = H$)
      Transmit the cumulative estimates to node 1;
    Else
      Transmit the cumulative estimates to node $h+1$
    End if
  End for
End while
Compute locations for each sensor:
For $1 \le h \le M$, $s_h = \mu_m, m \in [1, M]$
For $M+1 \le h \le H$, s_h = locations evenly distributed in uncovered area using geometrical approaches

Dynamical Procedure:
While Loop
For node $h = 1$ to H
  Collect new observations.
  If $| Z_h^{new} - Z_h^{old} | < \varepsilon'$
    Recursively update estimates of parameters using Equation (13).
  End if
  Move to the current estimates of optimal location.
  If ($h = H$)
    Transmit the cumulative estimates to node 1;
  Else
    Transmit the cumulative estimates to node $h+1$
  End if
End for
End while

Figure 5.4 Dynamical Sensor Placement Algorithm.

An experiment on the convergence rate is provided in Section 5.6. As long as the estimates are converged, the recursive EM algorithm in a distributed manner is launched. Essentially, node 1 collects the new observations. If the difference between new observations, z_h^{new}, and old ones, z_h^{old}, is higher than a threshold (ε'), it then recursively updates cumulative estimates with one new observation in each iteration. If the difference is lower than threshold, it skips the computation of updating estimates to save power and time, then increments the cumulative statistics with its previous local estimates. Each node updates the cumulative estimates by the new observations it collects and then passes the summary of statistics to the downstream node in a cyclic manner. Hence the distributed recursive EM algorithm can be summarized in Figure 5.4.

5.6 Simulation and Performance Analysis

5.6.1 Sensor Placement Applications

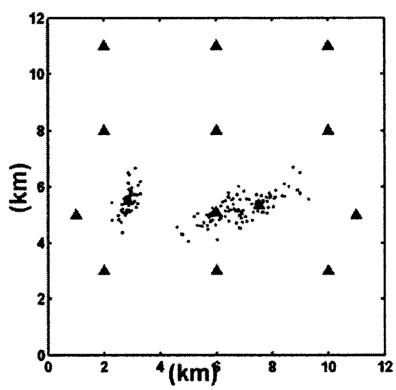


Figure 5.5 Optimal sensor placement simulation.

A simulated sensor deployment application is presented here in Fig. 5.5 to demonstrate the effectiveness of dynamical sensor placement method. The following scenario is considered. Suppose a very large number of animals are present in a wild area required to be covered by a sensor network. Understanding that there are several points of attractions but these points are moving or hard to be localized, the probability of presence around the interesting points is assumed to fall into Gaussian Mixture Model. To maximize the coverage performance, the recursive EM algorithm should be conducted to find the local maximum first, place critical sensors around the points with local maximum density, and then place other sensors evenly in sparse area to monitor rare event. Figure 5.5 illustrates a simulated application that mimics this situation with area of 12km x 12km. In this work, 300 objects are simulated, 9000 observation of those objects' locations are generated according to a Gaussian Mixture Model, and the mixture weights are selected randomly. In a timely manner the proposed method adaptively estimates the optimal sensor placement for the most recent 300 observations, and the last 300 observations are shown in Figure 5.5. 14 sensors (shown as triangle in Fig. 5.5), with covering circles whose radius are approximately 3km, are deployed and 3 groups of objects are clustered. As mentioned earlier, initially the 14 sensors are placed evenly in the area, the dynamical sensor placement method estimates the updated optimal location of sensors and moves sensors to the desired place adapted to dynamics of observations, Figure 5.5 illustrates the output of the dynamical sensor placement after the 9000 samples are taken into recursive estimation.

5.6.2 Performance of Detection

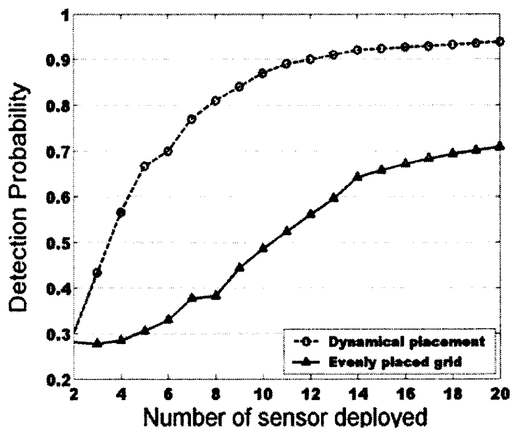


Figure 5.6 Detection accuracy probability.

As mentioned in Section 5.3, the detection probability is $e^{-\rho d}$. In this simulation the parameter is set: $\rho = 0.1\ (/km)$. Figure 5.6 demonstrates the better coverage performance of the recursive EM algorithm in comparison with evenly placed sensor approach. It is straightforward to observe that with increment of the deployed sensors, the detection probability is improved. As the number of sensors increases sufficiently, the detection improvement by adding sensors becomes more and more moderate. This presents a clue for selecting the number of deployed sensors to achieve satisfactory detection probability with the fewest necessary deployed sensors only. The simulation also demonstrates that the optimized algorithm adapt well with the topological change of the mass objects so that it can fairly accurately cover most of mass objects with limited sensors only.

5.6.3 Measurement on Model Fitness

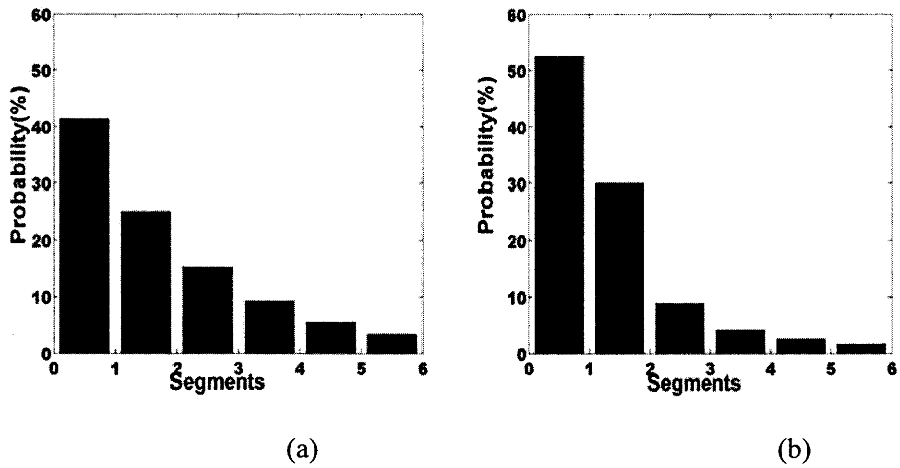


Figure 5.7 Histogram of probability (a) Estimated probability, (b) Empirical probability.

The estimates of object distribution may not exactly fit the real mass observations. This work uses the normalized difference between the estimated and empirical probability to measure how well the learned model fits the real observations. For convenience, the empirical probability is compared with estimated probability in a discrete manner. Basically, the region of a cluster is divided into several equal-sized segments. In practice, the segments of a specific cluster all have elliptic or circular border with common center. Hence a certain number of, say G, segments in a cluster should be $G-1$ "rings" and the innermost circle (or ellipse). The empirical probability of a certain segment is calculated by dividing the number of observations in that segment with the total number of observations in the cluster. The estimated probability of a segment is calculated by measuring how likely the points locate in that segment based on the estimated Gaussian distribution. Figure 5.7 illustrates the histograms of estimated and empirical probabilities of segments in the simulated application. Each cluster is divided into 6 segments in the presented simulation.

Figure 5.7 provides a visualized judgement on how well the model fits the observation. For quantitative analysis on the model fitness, "fitness score" is defined as Equation (5.19):

$$F(\theta) = [1 - \frac{\sum_{g=1}^{G} (p_{real}(g) - p_{est}(k))^2}{\sum_{g=1}^{G} (p_{real}(g) - \frac{1}{G})^2}] \cdot 100 \tag{5.19}$$

In Equation (5.19), $p_{real}(g)$ and $p_{est}(g)$, respectively, represents the empirical probability and estimated probability of data points in g-th segment, $1/G$ is the mean of probabilities in all segments, and also the probability of uniform distribution. High fitness score indicates that the mass data samples can be well reflected in the estimated Gaussian mixture model. Note that the upper bound of fitness score is 100. Sufficient data are needed to calculate the probabilities in all segments. The above example collects 5000 observations as a window of samples. The fitness score in this simulation is 89.31, which demonstrates that the estimates of Gaussian mixture model reflect the mass observations very well.

5.6.4 Adaptive Boundary Estimation

Based on the real-time estimates of mass characteristics of observations, the probabilistic boundary of mass objects can also be estimated. With same simulated observations in the sensor placement applications, this section focuses on dynamical topology of observed locations. As shown in Fig. 5.8, the real-time "likely" boundary can be adaptively estimated with certain probability coverage. Since prior knowledge about the cluster information is unknown, the proposed algorithm starts with 20 clusters.

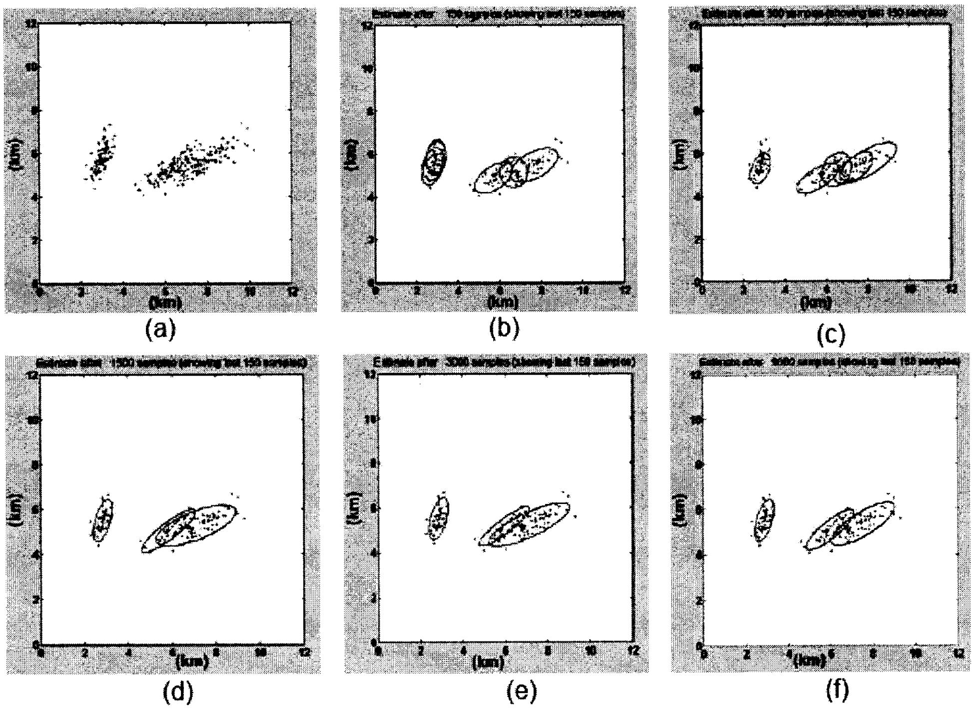


Figure 5.8 Adaptive boundary estimation at different Sampling times, a) Initial observation b) Estimates after 150 samples c) Estimates after 300 samples d) Estimates after 1500 samples e) Estimates after 3000 samples and f) Estimates after 9000 samples.

As mentioned in Section 5.4.2, a penalty function is utilized to discard those unnecessary clusters. This explains why the first couples of estimates include several unnecessary clusters and later on they merge to three clusters. Note that the "likely" boundary in this work refers to the ellipse with certain probability coverage, and all the points on the border have equal probability density. In the simulation shown above, the proposed method chooses the boundaries with 95% *probability coverage*, which means the expected percentage of observations that the boundary can enclose in a statistical sense. Figure 5.8.(a)-(f) illustrate the real-time boundary estimates in a timely manner as the most recent estimates update the boundary of the observations.

5.6.5 Convergence Test

As mentioned earlier, initially the sensors are placed evenly throughout the target area. The proposed algorithm starts with original observations and takes certain iterations and communication cycles through all the nodes to achieve the initial convergence.

The initial convergence should be achieved before the dynamical optimal estimates of sensor placement could be available. To some extent the performance and feasibility of the proposed algorithm is affected by the initial convergence. An experiment on the convergence is presented here to study how many communication cycles are needed for different number of clusters, i.e., points of interests, and the number of sensors deployed in the target area to achieve the convergence. Figure 5.9 shows the result of convergence experiments. The simulation randomly generates a certain number (e.g. 2, 4, 6, 8, 10 in the experiment) of points of attraction that are randomly placed in a 12km x 12km area. The experiments also show that the number of deployed sensors imposes impacts on the convergence, as illustrated in Figure 5.9, the communication cycles needed for sensor networks consisting of 10, 15, 20 sensors are studied. It is clear that 1) the more points of attractions, the more communication cycles are needed for convergence; and 2) the more sensors are deployed, the more communication cycles are needed for convergence.

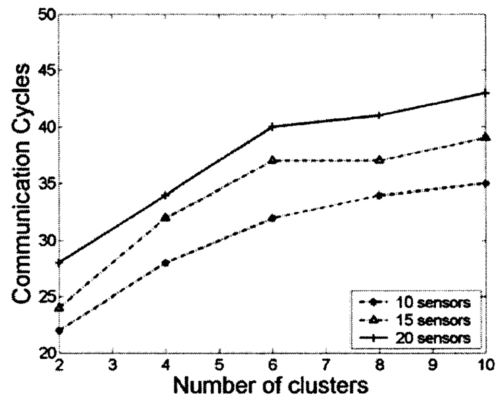


Figure 5.9 Communication cycles needed for convergence.

5.7 Summary

This chapter proposes an adaptive optimal sensor placement method based on maximum likelihood estimates of mass object locations. The proposed method studies the mass characteristics of observations by assuming a Gaussian mixture model, and searches for the optimal locations with online updating parameter estimates of the real-time observations to place the sensors for maximizing detection probability. To the best knowledge, it is for the first time that an unsupervised learning method on adaptive optimal sensor placement for monitoring mass objects has been proposed. The probability that an object can be detected by sensors is assumed to be a mixture probability that sums up the conditional probability of object detection by different sensors. The distributed implementation of the recursive EM algorithm is proposed as well to reduce the communication cost. The proposed method is able to make sensors placed at the positions with the local maximal density, which will eventually maximize the detection accuracy of sensor networks. The proposed method is also useful in mass objects' boundary estimation. Essentially, the online recursive learning on mass objects distribution can estimate the real-time elliptic boundary with certain probability coverage. The simulation results demonstrate the effectiveness of the proposed method on adaptive optimal sensor placement and real-time boundary estimation.

CHAPTER 6

CONCLUSION AND OUTLOOK

6.1 Conclusion

This dissertation has formulated several optimization and estimation problems in wireless ad hoc and sensor networks, wherein optimal reliable routing path, optimal power consumptions and optimal tracking interval and sensor placement are studied with probabilistic estimation and prediction approaches to provide effective and efficient services in wireless ad-hoc and sensor networks.

Wireless communication system could play an essential role in emergency or extreme situations such as monitoring and detection of fire, wild animals and battlefields. Unfortunately, the existing systems often provide insufficient information about the topology of mobile objects and infrastructure of communications. As a result, the stochastic problem of estimation and prediction on objects' location and movement becomes crucial in the wireless communication world, especially for wireless ad-hoc and sensor networks. Supplying the physical world with mobile ad-hoc agents or wireless sensors can generate massive amounts of data that can be processed by probabilistic estimation and prediction efficiently and usefully. This is where estimation and optimization comes into work for wireless ad-hoc and sensor networks. Many techniques can be applied in this field, such as learning theory, adaptive filter theory, statistical pattern recognition, and time series analysis. This research area is promising and still in progress. The proposed work in this dissertation is hence in demand for development and improvement. It offers beneficial perspectives for the ad-hoc and sensor network community to consider.

6.2 Outlook

Most applications for sensor networks can be formulated as estimation on the environment through examples. Stochastic estimation can be applied to many applications for the object tracking in sensor networks. Usually, people are interested in a particular feature of the environment that can be approximately represented by a probabilistic model for estimation and optimization. Applying the estimation algorithms to sensor networks involves these problems:

- How much computation should be distributed to individual nodes, and how much of it should be centralized?
- Which stochastic model should be chosen?
- Which values of parameters for sensor deployment are the best?

Centralized computation requires much long-range communication with a base station, which consumes much energy but might allow for more intensive data processing by a more powerful base station computer. This is the usual tradeoff between communication and computation in sensor networks. However, because transmitting a bit of information is still an order of magnitude more expensive than computing it [40], a more distributed computation is preferable. The following examples suggest how to formulate particular sensor network applications from a stochastic estimation viewpoint.

Intrusion and/or fault detection

When sensor networks are applied to hostile area, intrusion is very likely to happen. Usually attackers may behave like a normal sensor and spoof ID's of other sensors. These attacks are intelligent, however, through statistical learning algorithm or linear regression model the normal behavior could learned and the attacks can be separated from the normal behaviors.

Environmental monitoring

In this case, some unknown scalar environmental function defined on the covered region should be estimated, such as temperature, air pressure, or humidity. It can be assumed that the measurement of the function at each node is a normally distributed random variable. When a certain stochastic estimation algorithm is applied, each node obtains a local estimate. The upper bounds for the local estimation error can be predicted with desired confidence.

Continuous object tracking

Suppose a plume of hazardous gas moving slowly is expected to be monitored and tracked. Each node measures the concentration of the gas at its location and outputs 1 if the concentration is higher than some threshold; otherwise 0. Based on the labeled observations, supervised learning techniques can be applied to estimate the boundary and the moving speed.

Sensor localization

Sensor localization can be achieved similarly. Suppose that a certain number of nodes know their positions with certain accuracy. Often the sensor network's first task is to localize the remaining nodes. Then the node with unknown position can communicate with the neighbors who know their positions and gather the information of their locations and an attribute data that indicates if the distance between them is less than a threshold or not. Learning from the given observations, the node with unknown position obtains an estimate of the indicator function. The optimal solution to its position is the point where the maximum likelihood estimate converges.

Through the proposed research and outlook, it is demonstrated that stochastic estimation offers an effective approach to design and operation of sensor networks. This dissertation shows how well the estimation algorithms can be used in the context of reliability optimization and object tracking. The proposed estimation should take into account the power consumptions, which can also be optimized with stochastic estimation.

REFERENCES

[1] T. Arici, and Y. Altunbasak, "Adaptive sensing for environment monitoring using wireless sensor networks," in *Proc. of 2004 IEEE Wireless Communications and Networking Conference*, Vol. 4, pp. 2347-2352, Atlanta, GA, March, 2004.

[2] Y. Tseng, and Y. Li, Y. Chang, "On route lifetime in multihop mobile ad hoc networks," in *IEEE Transactions on Mobile Computing*, 2003 Vol. 2(4), pp. 366-376, 2003.

[3] Q. Jiang, and D. Manivannan, "Routing protocols for sensor networks," in *Proc. of First IEEE Consumer Communications and Networking Conference,* pp. 93-98, Las Vegas, NV, January, 2004.

[4] A. Bhattacharya, and Das, S.K., "Lezi-update: An information-theoretic framework for personal mobility tracking in PCS networks," in *ACM/Kluwer Journal on Wireless Networks*, Vol. 8(2-3), pp. 121-135, March-May 2002.

[5] G. Wan, and E. Lin, "A dynamic paging scheme for wireless communication systems," In *Proc. of the Third Annual ACM/IEEE International Conference on Mobile Computing and Networking*, pp. 195-203, September 1997.

[6] Y. Xu, and W.C. Lee, "On localized prediction for power efficient object tracking in sensor networks," in *Proc. of 23rd International Conference on Distributed Computing Systems Workshops*, pp. 434-439, Providence, RI, May, 2003.

[7] Z. Guo, and L. Zakrevski, "Probability –based reliable multi-path selection in mobile ad hoc networks," in *Proc. of the 16th IASTED International Conference on Parallel and Distributed Computing and Systems*, pp. 286-291, Boston, MA, Nov., 2004.

[8] Z. Guo, and M.C. Zhou, "Prediction-based Object Tracking Algorithm with Load balance for Wireless Sensor Networks," in *Proc. of 2005 IEEE International Conference on Networking, Sensing and Control*, pp. 756-760, Tucson, AZ, March, 2005.

[9] A. Bruce McDonald, and T. Znati, "A Path Availability Model for Wireless AdHoc Networks," in *Proc. of IEEE Wireless Communications and Networking Conference* 1999 (WCNC '99), pp. 35-40, New Orleans, LA, 1999.

[10] S.Jiang, D.He and J. Rao, "A Prediction-based Link Availability Estimation for Mobile Ad Hoc Networks," in *Proc. of IEEE INFOCOM,* pp.1745-1752, Anchorage, AK, April, 2001.

[11] A. Bruce McDonald, and T. Znati, "A mobility-based framework for adaptive clustering in wireless ad-hoc networks," in *IEEE Journal on Selected Areas in* Communications, Special Issue on Ad-Hoc networks, Vol.17(8), pp1466-1487, Aug, 1999.

[12] T. Hou, and T.J. Tsai, "An access-based clustering protocol for multihop wireless ad hoc networks," in *IEEE Journal on Selected Areas in Communications*, Vol.19(7), July 2001 pp.1201-1210, 2001.

[13] J. Tang, G. Xue, and W. Zhang, "Reliable routing in mobile ad hoc networks based on mobility prediction," in *Proc. of 2004 IEEE International Conference on Mobile Ad-hoc and Sensor Systems*, Fort Lauderdale, FL, pp. 466-474, 2004.

[14] C. E. Perkins, E. M. Belding-Royer, and S. R. Das, "Ad hoc on-demand distance vector (AODV) routing," Internet Engineering Task Force, *Internet Draft (Work in Progress),* June 2002.

[15] D. Turgut, S. Basagni and S. Das, "Mobility-adaptive protocols for managing large ad hoc networks," in *Proc. of IEEE International Conference on Communications (ICC), vol. 5,* pp. 1539-1543, Helsinki, Finland, Jun 2001.

[16] S. Abhyankar, and D.P. Agrawal, "Distributed mobility-aware route selection for wireless ad hoc networks," in *Proc. of the 21st IEEE International Conference on Performance, Computing, and Communications*, pp. 241-247, Phoenix, AZ, April, 2002.

[17] L.M. Feeney, "A Taxonomy for Routing Protocols in Mobile Ad Hoc Networks," in *SICS Technical Report* T99/07, October 1999.

[18] T.C. Schardt, and C. Yuan, "A dynamic communication model for loosely coupled hybrid tracking systems," in *Proc. of the Fifth International Conference on Information Fusion*, 2002, Vol. 2 , pp.1236-1242, Annapolis, MD, July 2002.

[19] V. Raghunathan, C. Schurgers, S. Park, and M. B. Srivastava, "Energy aware wireless microsensor networks," in *IEEE Signal Processing Magazine*, Vol.19(2), pp. 40-50, March, 2002.

[20] A. Cerpa, J. Elson, D. Estrin, L. Girod, M. Hamilton, and J. Zhao, "Habitat monitoring: application driver for wireless communications technology," in *Proc. of First ACM SIGCOMM Workshop on Data Communications in Latin America and the Caribbean,* San Jose, Costa Rica, pp. 20-41 April 2001.

[21] J. Hill, R. Szewczyk, A. Woo, S. Hollar, D. Culler, and K. Pister, "System architecture directions for network sensors," in *ACM SIGOPS Operating systems Review*, Vol. 34, pp. 93-104, 2000.

[22] A. Heinzelman, A. Chandrakasan, and H. Balakrishnan, "Energy-efficient communication protocols for wireless microsensor networks," in *Proc. of the 33rd International Conference on System Sciences (HICSS '00)*, pp. 8020-8029, Vol.8, January 2000.

[23] WINS project. Rockwell Science Center. http://www.sins.rsc.rockwell.com. June, 2005

[24] J. Goel and T. Imielinski. "Prediction-based monitoring in sensor networks: taking lessons from MPEG," in *ACM Computer Communication Review*, Vol. 31(5), pp. 70-83, October 2001.

[25] H. Yang, and B. Sikdar, "A protocol for tracking mobile targets using sensor networks," in *Proc. of the First IEEE Workshop on Sensor Network Protocols and Applications* , pp. 71-81, Anchorage, AK, May, 2003.

[26] T. Arici, and Y. Altunbasak, "Adaptive sensing for environment monitoring using wireless sensor networks," in *Proc. of 2004 IEEE Wireless Communications and Networking Conference,* Vol.4, pp.2347-2352, Atlanta, GA, March, 2004.

[27] C. Lin, and Y. Tseng, "Structures for in-network moving object tracking in wireless sensor networks," in *Proc. of First International Conference on Broadband Network 2004* (BroadNets 2004), pp.718-727, San Jose, CA, October, 2004.

[28] P. Bahl, and V. N. Padmanabhan, "RADAR: An in-building RF-Based user location and tracking system," in *Proc. IEEE INFOCOM Conference*, 2000, pp. 775-784, Tel Aviv, Israel, March, 2000.

[29] M. Vossiek, and L. Wiebking, "Wireless local positioning – concepts, solution, application," in *Microwave Magazine, IEEE,* Vol.4(4), pp. 77-86, Dec.2003.

[30] S. Ray, R. Ungrangsi, "Robust location detection in emergency sensor Networks," in *Proc. of the IEEE INFOCOM Conference*, 2003, pp. 1044-1053, San Francisco, April, 2003.

[31] S. Dhillon, "Sensor placement for effective coverage and surveillance in distributed sensor networks," in *Proc. of IEEE Wireless Communications and Networking Conference (WCNC)*, pp. 1609-1614, Atlanta, GA, March, 2003.

[32] J. A. Bilmes, "A gentle tutorial of the EM algorithm and its application to parameter estimation for Gaussian mixture and hidden Markov models," Technical report, U.C. Berkeley, April 1998.

[33] Z. Zivkovic.; F. Van der Heijden, "Recursive unsupervised learning of finite mixture models," in *IEEE Trans. on Pattern Analysis and Machine Intelligence,* Vol. 26(5), pp. 651-656, May 2004.

[34] A. Gelman, J.B. Carlin, H.S. Stern, and D.B. Rubin, "Bayesian Data Analysis", Chapter 16, Chapman and Hall, 1995.

[35] R.D. Nowak, "Distributed EM algorithms for density estimation and clustering in sensor networks," in *IEEE Trans on Signal Processing,* Vol.51[8], pp. 2245-2253, Aug. 2003.

[36] G. Wang, G. Cao, and T. La Porta, "Movement-assisted sensor deployment," in *Proc. of 23rd Annual Joint Conference of the IEEE Computer and Communications Societies: INFOCOM 2004,* Vol. 4[7-11], pp.2469-2479, Hong Kong, March 2004.

[37] Y. Zou, and K. Chakrabarty, "Uncertainty-aware sensor deployment algorithms for surveillance applications," in *Proc. of IEEE Global Telecommunications Conference,(GLOBECOM 2003).* Vol.5, pp. 2972-2976, San Francisco, CA, Dec. 2003.

[38] F.Y.S. Lin, and P.L. Chiu, "A near-optimal sensor placement algorithm to achieve complete coverage-discrimination in sensor networks" in *IEEE Communications Letters,* Vol. 9(1) pp. 43-45, Jan 2005.

[39] Z. Guo, M.C. Zhou, and L. Zakrevski, "Optimal tracking interval for predictive tracking in wireless sensor network," in *IEEE Communication Letter,* Vol. 9 pp. 805-807, Sept. 2005.

[40] S.N. Simic, "A learning-theory approach to sensor networks," in *IEEE Pervasive Computing,* Vol. 2(4), pp. 44-49, 2003.