STATISTICAL INFERENCE IN WIRELESS SENSOR AND MOBILE NETWORKS

A Dissertation

presented to

the Faculty of the Graduate School

University of Missouri

In Partial Fulfillment

of the Requirements for the Degree

Doctor of Philosophy

---

by

PENG ZHUANG

Dr. Yi Shang, Dissertation Supervisor

May 2010

The undersigned, appointed by the Dean of the Graduate School, have examined the thesis entitled

STATISTICAL INFERENCE IN WIRELESS
SENSOR AND MOBILE NETWORKS

presented by Peng Zhuang

a candidate for the degree of Doctor of Philosophy

and hereby certify that in their opinion it is worth acceptance.

Dr. Yi Shang

Dr. Wenjun Zeng

Dr. Michael Jurczyk

Dr. Zhihai He

ACKNOWLEDGMENTS

First and foremost, I would like to thank Dr. Yi Shang, my advisor for his continuous support to my research. I am impressed by his broad scientific insights and deep knowledge in so many fields. I have learned both theoretical and practical skills under his guidance. His passion about new technologies has also been a constant source of motivation.

I would like to thank Dr. Wenjun Zeng, Dr. Michael Jurczyk and Dr. Zhihai He for reviewing my dissertation. As the director for Gilliom Cyber security fellowship, Dr. Wenjun Zeng has provided constant support for my research. I have benefited from working with him in various projects as well as from attending his class about multimedia security. Dr. Michael Jurczyk is also the advisor of my academic society, Upsilon Phi Epsilon (UPE). He has taught me how to network with students from different backgrounds and how to enjoy being a part of MU engineering family. Dr. Zhihai He has served in both my master and PhD committees and provide tireless help and guidance for my thesis and dissertation. I really feel fortunate that I have the opportunity to work with so many great faulty members.

I also want to thank all the people in our wireless sensor network research group for their suggestions and helps to my research. Our exchanges of ideas, techniques, and theories have been an indispensable part of my graduate study. Many of my research have been conducted in collaboration with them and I want to express my gratitude for their selfless helps.

Finally, I would like to thank my family for always supporting me, especially my wife, who always believes in me.

STATISTICAL INFERENCE IN WIRELESS SENSOR AND MOBILE NETWORKS

Peng Zhuang

Dr. Yi Shang, Dissertation Supervisor

ABSTRACT

In recent years, wireless sensor networks have emerged as a cost effective alternative to traditional wired sensor systems. Compared to wired sensors, wireless sensors are relatively small in size, operated on batteries, communicate using wireless radios, and can last for years of operations. In the meantime, mobile networks have also gained many momentums. With the popularity of modern smart phones such as the Apple iPhone, we have witnessed a gold rush of mobile applications and services. The two emerging networks share many common features. Firstly, both networks consist of network nodes equipped with sensors that monitor the physical environment. Secondly, they both have short-range wireless communication (e.g., ZigBee, Bluetooth or WiFi). Finally, both network nodes operate on batteries, which requires power efficient programs in order to extend the length of operating time.

In this dissertation, we focus on four important problems in wireless sensor and mobile networks: a) data authentication, b) faulty sensor detection, c) indoor localization and tracking, and d) prediction. We formulate them as spatial/temporal statistical inference problems and develop efficient centralized and decentralized solution methods.

In the problem of ***data authentication***, we aim at providing an energy efficient means for data authentication using spatial correlations. A centralized method is proposed and is suitable for a wide range of sensor network applications that emphasize data integrities, such as traffic monitoring and control. Compared to three competing methods, it reduces the average data error by up to 60% and reduces the security overhead by an order of one magnitude.

In the problem of ***faulty sensor detection***, we introduce a new method for detecting faulty sensor nodes without human or centralized interventions. The proposed method is based on the principles of probabilistic collective theory. The method consistently outperforms two competing methods with up to 50% higher detection accuracy. It is suitable for decentralized sensor networks operated in remote or harsh environments.

In the problem of ***indoor localization and tracking***, we propose a new method for simultaneously tracking a target and constructing an indoor logic map using smart phones. The method is designed based on temporal inference and particle filtering. Simulation results show the proposed method outperforms an existing method by approximately 9 times in tracking accuracy and constructs maps of 89% accuracy on average. It can be used for location based services like a restaurant finder and for internet map services.

In the problem of ***prediction***, we focus on the area of traffic sensor data prediction and present an analytical method to derive the spatial correlation model. We show that the analytical method acquires close estimation to the learned correlation model without the need for extensive training sensor deployment.

List of Figures

Contents

Chapter 1

Introduction

Wireless sensor networks consist of small battery-powered sensor nodes that monitor the physical environment and report sensing data via low power and short range radio. In order to achieve years of operation time without human intervention, computation overhead and data load need to be kept to minimal. Small wireless sensor devices are also prune to being faulty due to many reasons, such as sensor aging, battery drain and environmental interference. Wireless communications are naturally vulnerable to malicious attacks, like packet fabrication, packet modification and packet replay. When deployed in uncontrolled or hostile environments, sensor nodes can be physical compromised and the encryption keys are exposed to the adversary. The design of a sensor network requires a great amount of consideration on both efficiency and robustness.

As another emerging network, mobile networks share many common characteristics as wireless sensor networks. Modern smart phones such as the Apple iPhone and most Android phones carries an array of sensors, including microphones, GPS sensors, accelerometers and digital compass. These sensors can be used for ambient fingerprinting[4], where a smart phone tries to classify the user's surroundings into several categories (e.g., restaurant, pub, bookstore, etc.), or traffic monitoring [40], where smart phones' onboard GPS sensors are used to predict traffic flow in urban settings. Another similarity between the mobile network and the wireless sensor network lies on their communication approaches. Wireless sensor devices usually are equipped with short range radio (e.g., IEEE 802.15.4 or ZigBee), and modern smart phones are also equipped with local communication capabilities (e.g., WiFi, Bluetooth). As a result mobile phones can be used as a media for RF-based localization/tracking [30], which is also a common technique used in the field of wireless sensor networks. Other similarities include their power constraints (both devices are battery powered), computation constraints (when compared to desktop class devices), and correlation between sensor data. To some extent, mobile networks can be viewed as a more powerful and mobile version of general wireless sensor networks.

In this dissertation, four important issues in wireless sensor networks and mobile networks are identified and are addressed based on either spatial or temporal inference. The three fields are 1) energy-efficient data authentication, 2) faulty sensor detection, 3) indoor tracking and map construction, and 4) traffic sensor data prediction and sensor placement. Next, we give a brief introduction to each of the four problems.

1.1 Energy-efficient data authentication

Many sensor network security protocols have been proposed based on cryptography. The sensor readings are treated as independent data and are authenticated separately. This is inefficient in many sensor network applications since sensor readings are usually correlated [76, 39]. Several methods based on content analysis have been proposed to authenticate sensor readings using their correlations. The protection level is usually low comparing to the cryptography based methods.

In this dissertation, we propose a new correlation model based method *Cobra (**Co**rrelation **b**ased **r**ecovery and **a**uto-detection)* that takes advantage of both cryptography and content analysis. Cobra is resilient to the following data integrity attacks: 1) packet fabrication, 2) packet modification, and 3) compromised node. Compared with traditional cryptography based system, Cobra introduces much less computation and communication overhead. Compared with other content analysis methods, it provides higher detection rate and more accurate data recovery. Like other content analysis methods, Cobra does not aim at providing absolute accuracy. It offers an energy efficient solution in applications where a small error is allowed.

1.2 Faulty sensor detection

Common sensor faults include short fault – a single-sample spike in sensor readings, random fault – a longer duration of noisy or random readings, constant fault – anomalous constant offset readings, drift fault – sensor readings are linear functions of correct values, and loss of sensitivity – sensor readings' variances are reduced often due to the aging of the sensor. The faults may be transient or last for a longer period of time. They have different characteristics that can be used in fault detection. For example, drift faults result in the shifts of mean and/or variance of the sensor readings and losses of sensitivity lead to the reduction of sensor readings' variance. To our best knowledge, most existing works assume a particular fault type and propose detection methods accordingly.

In this dissertation, a general measurement *mutual divergence* is introduced for detecting arbitrary types of sensor faults. Mutual divergence does not rely on a particular fault type. We prove the correctness of mutual divergence and introduce three detection methods in both centralized and distributed manner. We argue that centralized methods are inferior to distributed methods due to their high demand of network synchronization, long detection delay and large communication overhead. In our experiments, we have found that the search space for mutual divergence has many local optimal and one of our proposed methods, the distributed collection detection, is more capable of finding global optimal than other methods. In addition, it reports a sensor's faulty probability. Such additional information can be utilized to achieve more functionalities in practical systems. For example, the replacement of faulty sensors can be scheduled based on the order of their faulty probabilities.

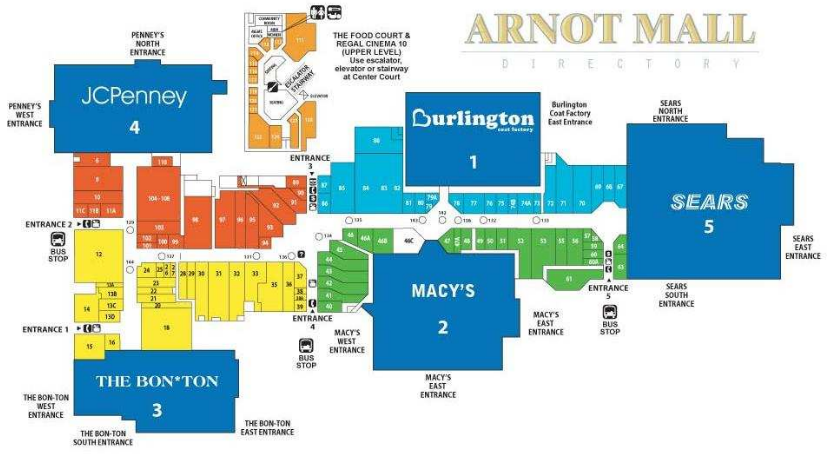
1.3 Indoor tracking and map construction

Modern smart phones such as the Apple iPhone or Android powered phones use a hybrid approach to acquire location [2]. The approach first acquires location information based on the locations of the nearby cellular towers, then uses nearby WiFi access points for finer localization, and finally utilizes the build-in GPS signal receiver (if available) for more accuracy. When a smart phone is used indoors, GPS signal is usually not available and the approach can produce a localization error as large as several hundred feet. Another approach is based on the idea of dead reckoning and has been implemented in digital cameras for geo-tagging indoor photos [34]. The approach uses last available GPS information, usually as the location before a user enters a building, and build-in speed and heading sensors to track user's location. The speed sensor and the heading sensor are usually implemented with accelerometers and magnetometers, respectively. The quality of location estimates depends on the quality of the speed/heading estimations, and the error propagates over the course of tracking.

Another line of localization research focuses on finding logical locations instead of physical locations. Take context aware mobile advertising as an example. It may be useful to know an user's environment (e.g., grocery stores, bookstores, or movie theaters) in order to deliver the most relevant ads. SurroundSense [4] is an example of such systems. A smart phone's build-in sensors are used to identify the ambient fingerprints, including the audio, acceleration and visual fingerprints. Examples of the audio fingerprint are human voices, background music, and background noises (e.g., noise from coffee machines at Starbucks). Acceleration fingerprint reveals the subject's moving pattern. One would expect the moving patterns at hallways and meeting rooms to be different. Visual fingerprint includes hue, saturation and lightness patterns. Floor tiles and carpet patterns have been found as reliable indicators of different fingerprints [4]. Using the combination of various types of fingerprints, SurroundSense has achieved a classification accuracy of 87% in a trial of 51 logic locations.

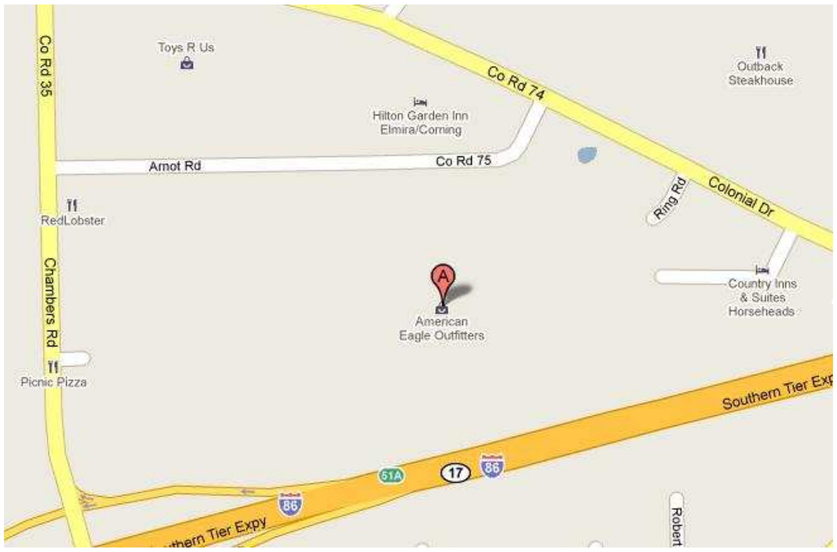
In location based applications, it is also critical to prepare a database of geo-tagged information, such as points of interests (POI) and maps. Obtaining such information is often difficult and involve tremendous amount of manual inputs. To our best knowledge, existing global/national map services do not provide detailed floorplans and their POI lists are far from sufficient. Figure 1.1(a) is the detailed floorplan for Arnot mall, Horseheads, NY. Figure 1.1(b) shows the Google map coverage at the same address. Information provided by Google map is not sufficient for navigation or POI lookup in the mall. In addition, it is difficult to promptly reflect floorplan or POI changes. For example, if a bookstore moves to a different place at a strip mall, there is no efficient way to update a national POI database unless a user reports the error.

This dissertation focus on a heterogenous mobile network consisting of mobile phones and WiFi access points, and propose SMART (***S***imultaneous ***M***ap ***A***cquisition and ***R***epeated ***T***racking). SMART is a system based on smart phones and cloud servers. It comprises three components. Firstly, a particle filtering based method is proposed to achieve location estimates of a mobile device uti-



ARNOT MALL
DIRECTORY
JCPenney 4
Burlington 1
SEARS 5
MACY'S 2
THE BON*TON 3
THE FOOD COURT & REGAL CINEMA 10 (UPPER LEVEL) Use escalator, elevator or stairway at Center Court

(a)



(b)

Figure 1.1: (a) The detailed mall map of Arnot mall, Horseheads, NY. (b) Google map at the same address.

lizing WiFi signals and motion speed/heading. The method outperforms a dead reckoning method by approximately 9 times in our experiments. Secondly, an ambient fingerprinting method is described. It is based on the method of SurroundSense [4]. Finally, the fingerprints and the location estimates are combined to generate a fingerprint map.

1.4 Traffic sensor data prediction and sensor placement

Measuring and reporting traffic counts in realtime provides a guidance for traffic light controllers and in-vehicle navigation systems. Traffic statistics are also vital in analyzing road usage and pavement wearing. In recent years, sensor networks have been proved to be a low cost, reliable, and non-intrusive alternative to traditional traffic measuring systems such as inductive loops [17]. Usually, the sensor nodes are placed every few hundred feets along the roads to acquire the complete traffic information [15]. In a real street system, such networks require tens of thousands of deployed sensors, which is neither practical nor cost effective.

To reduce the installation and maintenance cost of the network infrastructure, we explore the spatial correlation between sensory readings. Figure 1.2 shows the trace of traffic counts in every 50 cycles at two sensor locations from Green Light District simulator [71]. It is evident that the data share a strong correlation. Using a multivariate Gaussian distribution to model the correlation, and selecting the most informative sensing locations based on variances/covariances, we acquire

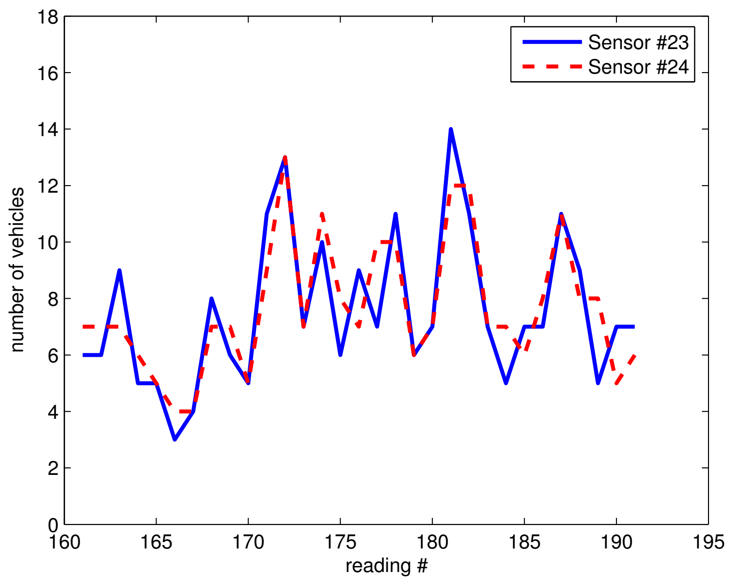


Figure 1.2: Trace of vehicle counts at two sensor locations at nearby places.

predictions to the complete information with an acceptable level of error. The selected sensors are merely a small portion of all points of interest. Figure 1.3 shows a sensor network monitoring 20 locations in a city highway system, whose measurements are used to predict the traffic at any points with an average error of 30%.

We define and analyze an optimization problem, whose goal is to minimize the average prediction error. We propose an analytical method as an alternative to machine learning based methods in acquiring the correlation model. We demonstrate its accuracy in modelling the correlation using simulation results, and discuss its applications and advantages compared to machine learning based methods.

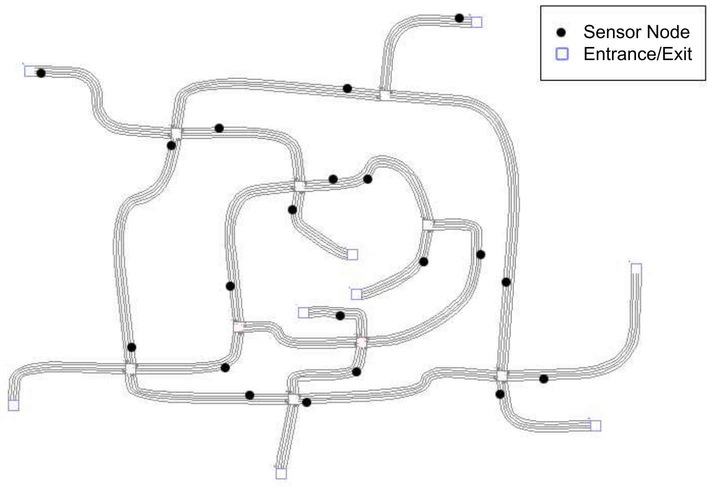


Figure 1.3: A street system with 20 sensor nodes deployed.

1.5 Our contributions

We introduce a new energy-efficient method to authenticate sensor data. Compared with existing methods, it achieves up to 60% higher detection rate for the forged data and saves energy consumption by an order of one magnitude. It also protects sensor networks from aggressive attacks by limiting the maximal damage on data reliability.

In the problem of faulty sensor detection, we introduce mutual divergence as a general measurement for sensors' faulty likelihood. Mutual divergence does not rely on assumptions of particular fault type and can be used to measure unknown fault types. We also propose a distributed method utilizing mutual divergence and it is shown to achieve detection accuracies close to 100% in most test cases, outperforming competing methods by up to 50%.

We propose a new system for tracking mobile subject and for constructing logic maps. The tracking method outperforms an existing tracking method by 9 times. The map construction method constructs logic maps with up to 89% accurate compared with the ground truth. We propose a system architecture to implement such a system based on smart phones and cloud servers.

We propose a new analytical method to acquire spatial correlation model in the area of traffic monitoring. During our simulation, the analytical models are shown to closely estimate the models learned based on training data. The analytical method eliminates the need for an extensive sensor deployment for collecting the training data. We also study three algorithms for sensor deployment and find out that a simple greedy algorithm outperforms a random-trial based method with only little space for a local search algorithm to improve.

1.6 Dissertation organization

The dissertation is organized as follows. In the next chapter, we summarize related works and describe theoretical backgrounds, including multivariate Gaussian distribution, probabilistic collective, and particle filtering. In chapter 3, we focus on the problem of energy efficient sensor data authentication and in chapter 4 we discuss solutions for distributed faulty sensor selection. The problem of simultaneous tracking and map construction is discussed in chapter 5 and the problem of road traffic sensor data prediction and sensor placement if discussed in chapter 6. Finally, we reach our conclusions in chapter 7.

Chapter 2

Related Works and Theoretical Background

2.1 Energy efficient wireless protocols

In battery powered wireless sensor networks and mobile networks, energy conservation is usually one of the foremost concerns for algorithm/protocol design. The problem of reducing communication cost is commonly formulated as data aggregation problems. Data aggregation occurs along the packet transmitting path where internal nodes summarize the data and thus reduce the amount of transmitted data [48, 49, 74, 19, 33, 53, 10, 19, 50, 24, 11]. Data aggregation can be classified in two categories, tree based and multi-path based. Tree based methods assume a spinning tree routing pattern and data aggregation is performed at intermediate nodes, such as [48, 49, 74]. Tree based methods achieve the highest communication saving but is not robust against communication failure. Multi-path aggregation methods have been proposed to address the problem [19, 53]. They solve the robustness issue at a cost of higher communication cost. Manjhi et. al. have proposed a hybrid method [50] that promises to take the good of both worlds. Chan et. al. address the problem of secure data aggregation [11], where methods for obtain accurate results in the presence of adversaries are discussed.

Another common formulation for reducing communication cost is sensor selection using correlation models. The solutions are based on either Bayesian theory [20, 28], or Markov random field [68]. Krause et al. also study the prediction problem using a correlation model based on multivariate Gaussian distribution [39].

2.2 Data authentication in energy-constrained environments

In wireless sensor networks, many data authentication protocols have been proposed based on cryptography. Some commonly known protocols are MiniSec [46], TinySec [35], and SNEP [57]. These protocols add $3 \sim 8$ extra bytes to the 12 bytes header in the standard TinyOS network stack. For energy consideration the protocols are designed with short-keys or even shared keys, which are secure only until 2012. The IEEE 802.15.4 based Zigbee standard provides a higher level of data security at an expense of significantly higher communication overheads [3]. The cryptography-based protocols are shown to be energy consuming.

In [12], Chang et. al. examine the energy consumption of several commonly used hashing and symmetric-key encryption schemes on Crossbow and Ember sensor motes. The tested security schemes consume 18$\sim$134 times more energy in computation and 2$\sim$3.3 times more energy in communication.

Another family of security methods verifies sensor packets by analyzing their contents. Most proposed methods use correlation models to solve the problem, including CORA [67], ART [69] and Bayesian Selection [54]. CORA suffers from very high false alarm rate. ART is not secure if an attacker modifies a large fraction ($> 10\%$) of the readings or compromises the sender node. Bayesian Selection deals with compromised nodes but it relies on a strong assumption that the bias introduced by the attacker is persistent. Otherwise the false alarm rate can be as high as 50%. Other methods are proposed using the idea of random sampling [60] or distribution-based deviation detection [56, 52]. Comparing to the correlation-based methods, they produce less accurate detections and larger errors on data recovery.

2.3 Faulty sensor detection

Most faulty sensor detection methods are based on correlation analysis between neighboring sensors' readings. The correlation models can be represented by joint distributions [76, 67, 54, 26], by correlation coefficients [61, 69], or by some arbitrary similarity measures, such as thresholding on sensor reading differences [14] and event region based clustering [47, 41]. Methods based on data fusion [38] and outlier detection [21, 56] have also been proposed. In these works, a particular type of faults is usually assumed and specialized methods are developed. It is fairly easy to break a detection method by switching fault types.

2.4 Localization and tracking in wireless sensor and mobile networks

Radio signal based localization is broadly used. RF Measurement techniques can be classified into three categories. The first category is angle of arrival [37, 62, 6], where a device make use of its antenna's receiving signal characteristics to estimate the direction a message sender is located. The second category is distance related measurements such as one-way propagation time, round-trip propagation time, and time-difference-of-arrival [51, 25, 13]. The third category is RSS (radio signal strength) profiling measurements [5, 58, 42, 65, 64], where RSS is mapped to a distance measure.

RF based localization can be classified as infrastructure-based RF and infrastructure-less RF approaches. infrastructure-based RF approach requires special hardware (beacon) in the environment and a device localizes itself based on the beacon messages. Cricket [59] and Pinpoint [75] are early examples of this approach. Recently, Hay et. al. have proposed a method to localize smart phones using low level non-authorized Bluetooth connections to location beacons [30]. infrastructure-less RF approach does not require additional hardware. Instead it utilizes signals already available in the environment, such as WiFi or GSM signals. It is more cost effective and easy to scale. Chen et. al. have proposed a method that combines the benefits of GSM and WiFi localization [16]. Active campus [27] is a localization project on UCSD campus based on pre-known WiFi hotspots. Bahl et. al. have proposed a localization method based on WiFi fingerprinting that achieves localization accuracy of up to 5 meters [5]. However it requires throughout fingerprint mapping in advance and is not scalable.

Adding mobility information to RF based methods has shown promising results in improving localization results. The combined methods usually lay their grounds on the principals of particle filtering [70] [78] [36] [66] [31]. MCL [31] utilizes maximal speed information in addition to radio proximity information. MSL and MSL* have been proposed to improve MCL [66] by utilizing neighboring nodes' location estimates. MSL and MSL* converge faster and are more robust against the decrease of location beacon density. Turgut et. al. further improve the technique by automatically restarting the process of particle filtering if the particle cloud diverges too much from the observations. Particle filtering based localization have been applied to different scenarios. In [36], a human subject carries a sensing device equipped with accelerometers (used as a pedometer), magnetometers and a radio receiver. The mobility information (steps/speed, heading) and radio proximity information are used in combination with building floorplans to perform indoor tracking. In [78], moving vehicles are equipped with receivers that overhear location beacon broadcasted by in-pavement wireless traffic sensors. The information are combined with road map and odometers for vehicle tracking. It is worth mentioning that Kalman filter or its variants can be used to solve similar problems. It provides analytical solutions in contrast to the simulation based solutions in particle filtering. However it is often difficult to derive a Kalman filter in complex problems especially when the subject's mobility model is non-linear. Kusy et. al. have proposed a least-squares optimization based improvement that achieves almost 50% higher accuracy than regular Kalman filter [43].

Context/environment sensing has also been an active research field [4] [22] [63] [23] [18]. The goal of seeing, hearing, or feeling logical locations is intriguing and has motivated many researchers to find the proper set of sensors for their applications. Cameras have been a popular choice to identifying stable scenes [22] [63] [18], or floor patterns [4] [23]. Microphone is used in [4] and [18] and accelerometer is used in [4]. SurroundSense [4] is a system that uses a combination of cameras, microphone and accelerometers. It has been prototyped on Nokia N95 phones and properly identifies the fingerprints among 51 logic locations on or around Duke University campus 87% of the time.

2.5 Traffic sensor data prediction and sensor placement

The problem of using sensory data to guide traffic light controller is well studied. Wiering et al. investigate different control policies in [71] using the traffic queue length and vehicles' waiting time. Chen et al. [15] propose a system that uses the information exchanged between roadside and in-vehicle sensors. The Green Light District (GLD) simulator [71] is a free and open source software served as a testbed for traffic control policies. We implement a sensor class in this simulator that provides traffic information for traffic light controllers. The developments of such systems are made possible by the advance in the field of sensor network. Cheung et al. develop hardware that utilizes magnetic sensors and low power radio communications [17], which is now manufactured and distributed by SenSys Network, Inc.

The problem of sensory data prediction using correlation models has also been an interesting topic for many years. The solutions are based on either Bayesian theory [20] or Markov random field [68]. To our best knowledge, Guitton et al. first introduce the concept of data prediction into the field of traffic measuring in [29], where he explores the correlation coefficient between traffic counts. We have shown in our previous study that the correlation coefficient based method is less accurate in predicting sensor data than a Gaussian based method [76].

2.6 Theoretical Background

2.6.1 Spatial inference model based on multivariate Gaussian distribution

Let V denote the set of all sensor nodes and let x_V denote the random variables corresponding to the sensor readings. A multivariate Gaussian distribution is used to model the correlation among sensors in V. It has two parameters, a mean vector $\mu = \{\mu_1, \mu_2, ..., \mu_V\}$, and a $|V| \times |V|$ covariance matrix. The parameters are pre-learned using training data collected from a temporary deployment of sensors at all points of interest. Note that a model is only good for the data that are similar to the training data. If the environment changes, new model needs to be learned to account for the difference. We assume that the model used in this dissertation is learned over an extensive period and can reflect all the conditions (though outliers may still present).

We refer to this correlation model as the *prior distribution*. If we take the mean μ_i corresponding to sensor i and the entry at the ith row and ith column from the covariance matrix, we acquire the distribution for sensor i's reading independent of other sensors. Let X_Q^t denote the readings of sensors in set Q (the observed sensors) at time period t. A *posterior distribution* for unobserved sensors in $U = V - Q$ is given by

$$P(x_U|X_Q^t) = \frac{P(x_U, x_Q = X_Q^t)}{P(x_Q = X_Q^t)}. \tag{2.1}$$

Equation 2.1 is derived using Bayes' rule. Given the probability density function of multivariate Gaussian distribution of

$$P(x_V) = \frac{1}{(2\pi)^{|V|/2}|COV|^{1/2}} e^{-\frac{1}{2}(x_V - \mu_V)^T COV^{-1}(x_V - \mu_V)}, \tag{2.2}$$

we have the mean and covariance matrix of the conditional probability as

$$\begin{aligned} \mu_{x_U|X_Q^t} &= \mu_U + COV_{YQ}COV_{QQ}^{-1}(X_Q^t - \mu_Q), & (2.3) \\ COV_{x_U|Q} &= COV_{UU} - COV_{UQ}COV_{QQ}^{-1}COV_{QU}. & (2.4) \end{aligned}$$

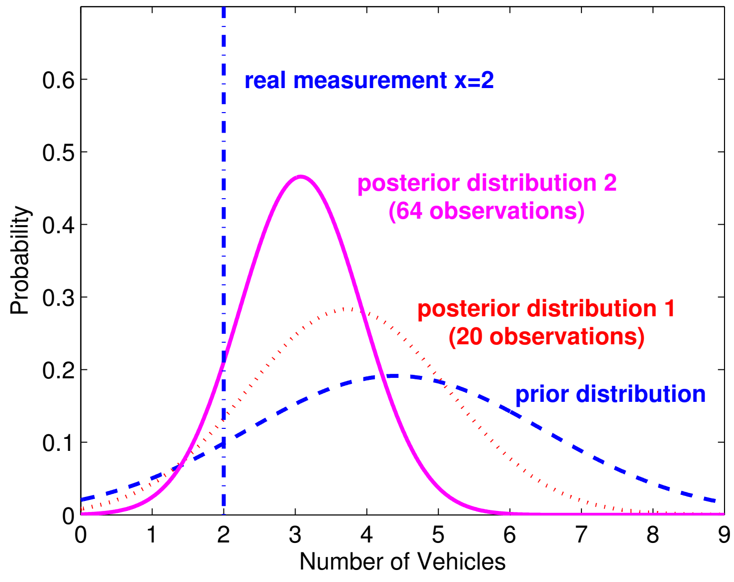


Figure 2.1: The prior distribution of the readings at a sensor location, and its posterior distribution given observations of 20 or 64 other sensors.

COV_{UQ} is acquired by taking the rows in U and the columns in Q from the covariance matrix.

Equation. 2.3 can be used to predict U's readings, i.e., the predicted value is the posterior mean. Eq. 2.4 measures the *uncertainties* of the prediction for each sensor since $COV_{x_i|Q}$ is the posterior variance for sensor i. Note that $COV_{x_U|Q}$ does not depend on the actual readings X_Q^t. It is used in sensor selection. One wants to select the set Q so that the average variance in $COV_{x_U|Q}$ is minimized.

Figure 2.1 shows a prior distribution for a sensor learned from vehicle counts collected at 84 sensor locations [76], and two posterior distributions when $|Q| = 20$ and $|Q| = 64$ ($Q_{20} \subset Q_{64}$). Usually adding more observations leads to more certain posterior distributions. However as shown in [77] the benefit of adding more sensors is diminishing as Q grows larger.

It is noted in the literature that in most correlation models nearby sensors carry stronger correlation [76, 39]. We use this property to reduce the storage space at each sensor, i.e., a sensor only stores the correlations with its neighbors.

2.6.2 Probabilistic collective theory

To perform spatial inference in a distributed manner, we adopt the theory of Probabilistic Collectives (PC). PC theory is a new framework for distributed decision making with probabilistic decisions [45, 72, 73]. In the PC theory, the optimization objective is casted as minimization of a function of the variable distributions [45][73]. Powerful distributed optimization methods using second order methods, difference utilities, annealing, data-aging, and supervised learning have been developed in the PC framework and have been shown to outperform other optimization techniques, including genetic algorithms and simulated annealing, on numerical optimization benchmark problems and classical combinatorial optimization problems, such as k-SAT, n-Queens, and bin packing [9][32]. For small-scale real applications, the PC approach has been successfully applied to real-time adaptive control of mini-flaps on trailing edges of airplane wings to minimize turbulence [7][8].

Let f_i denote the decision variable for sensor i (e.g., whether the sensor is faulty). A PC theory based method looks at finding the probability distribution for f_i instead at finding the exact value. Each sensor's decision variable f_i is considered independent from each other. We have

$$P(f_V) = \prod_i P(f_i). \tag{2.5}$$

The objective is to minimize a goal function $G(f_V)$. Given the probability distribution $P(f_V)$, the expected goal value is given by

$$E[G] = \int df_V P(f_V) G(f_V) = \int df_V \prod_{i \in V} P(f_i) G(f_V). \tag{2.6}$$

In a Nash equilibrium, every sensor adopts a mixed decision to optimize its expected goal function. However in a distributed system full rationality is absent due to the lack of information. It is often hard to for a sensor to choose the exact strategy. Shannon entropy is used to measure the uncertainty of a decision distribution. We have

$$S(P_i) = -\int df_i P(f_i) lnP(f_i). \tag{2.7}$$

A PC method is an iterative process. It usually starts with a uniform distribution. The updating rule in each iteration is given by

$$\begin{aligned} P_i(f_i = x) = \quad & P_i(f_i = x) - \alpha P_i(f_i = x) \times \\ & \{\frac{E[G|f_i = x] - E[G]}{T} + S(P_i) + lnP_i(f_i = x)\}, \end{aligned} \tag{2.8}$$

where α is the step size (usually set to 0.2), T is the temperature (usually starting with 0.1 and has a cooling rate of 1%), and $E[G|f_i = x]$ is the expected goal value given decision x. Equation 2.8 guarantee that the probabilities always sum to unity but does not prevent negative values. Negative values are set to a small constant (1×10^{-6}) and the probabilities are then renormalized.

In Eq. 2.8, the part $E[G|f_i = x] - E[G]$ measures how good a strategy is. If the expected goal value when x is adopted is smaller (better) than the average goal value, the probability of $f_i = x$ becomes larger for the next iteration. It helps to focus on the promising solutions but may lead to local optimal. The part $S(P_i) + lnP_i(f_i = x)$ compares the probability of $f_i = x$ to the average probability. If x is less likely than average this part gets a negative value and x will have a better chance being explored in the next iteration. This part helps to avoid local optimal but may prevent the solution from converging. A temperature T is used to balance the term. At first a higher temperature is used to avoid local optimal. As the temperature cools down through iterations, the solution finally converge to a few promising ones.

To compute the value for $E[G|f_i = x]$ and $E[G]$ efficiently, a Monte Carlo sampling method is used. In each iteration, each sensor draws m samples from the current distribution $P(f_V)$ and approximate the value of $E[G|f_i = x]$ and $E[G]$ as the average goal values.

A distributed implementation of a PC method requires designing a private goal function, i.e., the goal function that can be computed based on each sensor's available information. Each sensor has a ”neighborhood view” related to its neighbors and itself only. It updates its own probability distribution based on Eq. 2.8 and the

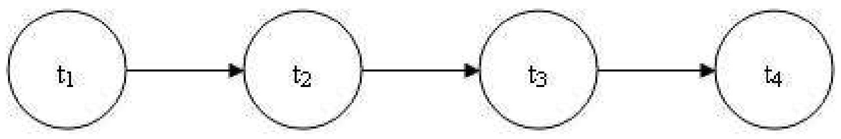


Figure 2.2: A graphic chain model for temporal inference.

private goal function, and then broadcasts its distribution to all neighbors for being used in the next iteration. Such a framework requires the message exchanges between neighboring nodes only and is both scalable and efficient.

2.6.3 Temporal inference based on particle filtering

Some sensor data such as a smart phone's location also carries strong temporal correlation. A graphic chain model (figure 2.2) is used to represent the transition over time. t_1, t_2, t_3, t_4 represent 4 time periods. Figure 2.3 gives an example of possible states in each time period and the arrows represent the transition functions.

In mobile target tracking, the states correspond to possible locations and the transition functions correspond to motion dynamics, such as motion direction and speed. The solution space for the locations is infinite and is usually represented by a distribution. The problem can be simplified by reducing the state space to a finite set of locations (*particles*), and the temporal inference is performed on the particles using the transition functions.

In this dissertation, the theory of particle filtering is used for mobile target tracking. The initial particles are randomly select from all possible locations,

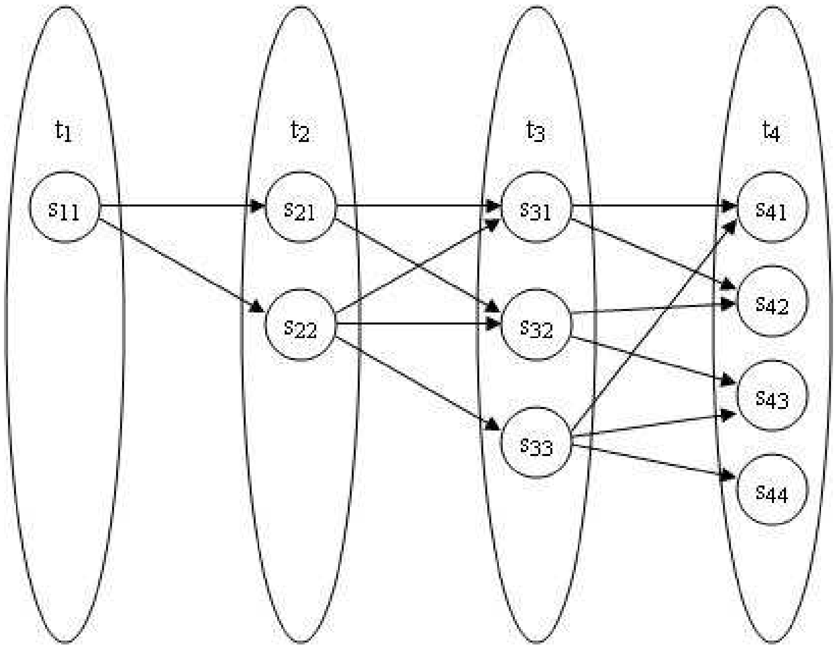


Figure 2.3: A graphic chain model representing state distributions with state transitions.

and are updated iteratively based on measured motion direction and speed (both subject to random noises). Observations such as RF proximity information are used to filter out unlikely sample in order to get more precise location distribution estimates.

Chapter 3

Cobra - A New Method for Energy Efficient Data Authentication

3.1 Problem formulation

In data collection tasks, a sensor network consists of a home server and a number of sensor nodes. Several access points may be added to connect disconnected sensor nodes. We assume the communication links between the access points and the home server are secure.

The time is divided into a number of sampling periods. In each sampling period, each sensor node transmits its current reading to a home server via multi-hop communications. A sensor node has a pre-installed unique private key. The home server has the corresponding public keys necessary to authenticate signature from each sensor node.

We assume the stealth attacker model in [60]. A stealth attacker only attacks the system by trying to convince the home server to accept the malicious readings. He does not drop a packet (denial-of-service) since it will be noticed by the home server. We focus on the four specific attacks as follows:

- *Packet Fabrication:* The adversary forges the readings of legitimate sensor nodes. In each sampling period, the home server may receive multiple readings claiming the same sender ID.

- *Packet Replay:* The adversary replays a previously received packet. This creates an effect on the home server similar to packet fabrication.

- *Packet Modification:* The adversary intercepts the packet from a legitimate sensor node, and modifies the content before forwarding it.

- *Compromised Node:* The adversary physically compromises a legitimate sensor node and reads the key. He can send forged readings with legitimate signatures.

In order to to provide reliable data at the home server, a security method must guarantee high detection rate on malicious readings that are significantly different from the real readings [60]. To measure the difference between the malicious readings and the real readings, we introduce a new metric *attack aggressiveness*. Let X_i^{tF} denote the malicious reading claiming the ID of sensor i in sampling period t and let X_i^{tT} denote the real reading. The bias introduced by the malicious reading is given by

$$\epsilon = X_i^{tF} - X_i^{tT}. \tag{3.1}$$

The attack aggressiveness is defined as $|\epsilon|$.

One performance metric is the detection accuracy given by

$$S^{acc} = \frac{\sum_{t=1}^{T}(N_t^{TN} + N_t^{TP})}{T \times n}, \tag{3.2}$$

where T is the number of sampling periods, n is the number of sensor nodes, N_t^{TN} is the number of detected malicious readings (true negative) in sampling period t, and N_t^{TP} is the number of accepted legitimate readings (true positive) in sampling period t.

Another performance metric is the average deviation of the accepted readings from the real readings, given by

$$\Delta_{RMSE} = \sqrt{\frac{\sum_{t=1}^{T}\sum_{i=1}^{n}(\widehat{X_i^t} - X_i^t)^2}{T \times n}}, \tag{3.3}$$

where X_i^t is the real reading of sensor node i in sampling period t and $\widehat{X_i^t}$ is the accepted reading. Δ_{RMSE} is also known as the **r**oot mean **s**quare **e**rror (RMSE).

The third performance metric is the security overhead measured in terms of the communication overhead and the computation overhead. The communication overhead is measured using the number of extra transmitted bytes, given by

$$C_{comm} = \sum_{t=1}^{T}\sum_{i=1}^{n} h_i \times b_i^t, \tag{3.4}$$

where h_i is the hop distance of sensor i to the home server and b_{ti} is the extra bytes introduced by the security protocol in sensor i's packet during sampling period t.

The computation overhead is measured using the extra energy cost at the sender sensor node. For simplicity, we assume that the energy to generate a signature is fixed and equals to 1 (the assumption is valid according to the results in [12]). The overall computation overhead is

$$C_{comp} = \sum_{t=1}^{T} \sum_{i=1}^{n} c_i^t, \tag{3.5}$$

where c_i^t is either 1 (authenticated) or 0 (un-authenticated).

3.2 An overview of Cobra: correlation based data recovery and auto-detection

In Cobra, only a small fraction of sensor nodes operate in a protected mode (*white mode*) in each sampling period. Their packets are protected with digital signatures and counters (to prevent packet replay). A reading sent from a sensor node in the white mode is referred to as a *white reading*. It is only vulnerable to compromised node attacks. Other sensor nodes operate in an unprotected mode (*gray mode*) to save energy. The corresponding readings are referred to as *gray readings* and are vulnerable to all four mentioned attacks. Note that Cobra can be easily extended to offer multiple levels of protection (e.g., different key lengths).

Cobra consists of the following three components:

- **Auto-detection**: The home server first authenticates the white readings using a pre-learned correlation model (as described in section 2.6.1) and generates a set of trusted readings (*proofs*). Next it authenticates the gray readings using the proofs and the same correlation model.

- **Data recovery**: The home server recovers the true values of the detected malicious readings *black readings* using the proofs and the correlation model.

- **Distributed sensor node scheduling**: each sensor node decides whether it will operate in white or gray mode based on its remaining energy or according to a pre-installed schedule.

3.3 Cobra details and analysis

3.3.1 Against packet fabrication/replay

When the home server receives more than one readings with the same sender ID i, the posterior probability distribution of the corresponding variable x_i is derived using the proofs (Eq. 2.3 and Eq. 2.4). The reading with the highest probability in the posterior distribution is regarded as the legitimate reading. The other readings are simply discarded.

Theorem 3.3.1 *The expected detection rate of Cobra against packet fabrication/replay attacks is a monotonically increasing function of the attack aggressiveness and a monotonically decreasing function of the posterior variance.*

Proof: *Given the mean μ and variance σ^2 in the posterior distribution*

$P(x_i)$, a malicious reading is detected if

$$\begin{aligned}
& |X_i^F - \mu| > |X_i^T - \mu| \\
\Rightarrow\quad & (X_i^T + \epsilon - \mu)^2 - (X_i^T - \mu)^2 > 0 \\
\Rightarrow\quad & (X_i^T + \epsilon)^2 + \mu^2 - 2(X_i^T + \epsilon)\mu - (X_i^T)^2 - \mu^2 + 2X_i^T\mu > 0 \\
\Rightarrow\quad & (X_i^T)^2 + \epsilon^2 + 2X_i^T\epsilon - 2X_i^T\mu - 2\epsilon\mu - (X_i^T)^2 + 2X_i^T\mu > 0 \\
\Rightarrow\quad & \epsilon^2 + 2X_i^T\epsilon - 2\epsilon\mu > 0 \\
\Rightarrow\quad & (2X_i^T + \epsilon - 2\mu)\epsilon > 0 \qquad (3.6)
\end{aligned}$$

Let Υ denote the solution interval of Eq. 3.6. The detection rate (true negative) is given by

$$E(s_{TN}) = P(x_i \in \Upsilon), \qquad (3.7)$$

where P is the posterior distribution derived from the proof. We have

$$\Upsilon = \begin{cases} (\frac{2\mu-\epsilon}{2}, \infty) & , \ \epsilon > 0 \\ (-\infty, \frac{2\mu-\epsilon}{2}) & , \ \epsilon < 0 \end{cases} \qquad (3.8)$$

The expected detection rate is given by

$$\begin{aligned}
E(s_{TN}) &= \begin{cases} \frac{1}{2}[1 + erf(\frac{\epsilon}{\sigma 2\sqrt{2}})] \ , & \epsilon > 0 \\ \frac{1}{2}[1 + erf(\frac{-\epsilon}{\sigma 2\sqrt{2}})] \ , & \epsilon < 0 \end{cases} \\
&= \frac{1}{2}[1 + erf(\frac{|\epsilon|}{\sigma 2\sqrt{2}})], \ \epsilon \neq 0. \qquad (3.9)
\end{aligned}$$

erf is the error function commonly used in the integration of Gaussian probability density function. It is a monotonic function and $erf(x) \in (-1, 1)$. As a result, the expected detection rate is monotonically increasing with $|\epsilon|$. In addition, the expected detection rate increases when σ decreases. This usually happens when more proofs are added. ■

3.3.2 Against packet modification

The home server derives the posterior distribution for each unique gray reading using the proofs. A reading is declared a "black reading" and rejected if it does not fall into a $(1-\alpha)$ confidence interval (α is a constant in interval $(0, 1)$).

Theorem 3.3.2 *The expected detection rate of Cobra against packet modification attacks is a monotonically increasing function of the attack aggressiveness $|\epsilon|$ and α, and a monotonically decreasing function of the posterior variance.*

Proof: The malicious readings are detected if

$$\begin{aligned} & X_i^F < \mu - k\sigma, \text{ or } X_i^F > \mu + k\sigma \\ \Rightarrow\ & X_i^T + \epsilon < \mu - k\sigma, \text{ or } X_i^T + \epsilon > \mu + k\sigma \\ \Rightarrow\ & X_i^T < \mu - k\sigma - \epsilon, \text{ or } X_i^T > \mu + k\sigma - \epsilon. \end{aligned} \quad (3.10)$$

The expected detection rate is given by

$$\begin{aligned}
E(s_{TN}) &= P(x_i < \mu - k\sigma - \epsilon) + P(x_i > \mu + k\sigma - \epsilon) \\
&= P(x_i < \mu - k\sigma - \epsilon) + 1 - P(x_i < \mu + k\sigma - \epsilon) \\
&= 1 + \frac{1}{2}(erf(\frac{-k\sigma - \epsilon}{\sigma\sqrt{2}}) - erf(\frac{k\sigma - \epsilon}{\sigma\sqrt{2}})) \\
&= 1 - \frac{1}{2}erf(\frac{k\sigma - \epsilon}{\sigma\sqrt{2}}) - erf(\frac{k\sigma + \epsilon}{\sigma\sqrt{2}}). \qquad (3.11)
\end{aligned}$$

Take partial derivative by σ, ϵ, k *and we have*

$$\begin{aligned}
\frac{\partial}{\partial\sigma}E(s_{TN}) = \frac{1}{\sqrt{\pi}}(-\frac{\epsilon\sigma^{-2}}{\sqrt{2}}e^{-(\frac{k\sigma-\epsilon}{\sigma\sqrt{2}})^2} + \frac{\epsilon\sigma^{-2}}{\sqrt{2}}e^{-(\frac{k\sigma+\epsilon}{\sigma\sqrt{2}})^2}), \\
\frac{\partial}{\partial\epsilon}E(s_{TN}) = \frac{1}{\sqrt{\pi}}(\frac{1}{\sigma\sqrt{2}}e^{-(\frac{k\sigma-\epsilon}{\sigma\sqrt{2}})^2} - \frac{1}{\sigma\sqrt{2}}e^{-(\frac{k\sigma+\epsilon}{\sigma\sqrt{2}})^2}).
\end{aligned}$$

We have

$$\frac{\partial}{\partial\sigma}E(s_{TN}) < 0 \qquad (3.12)$$

and

$$\begin{cases} \frac{\partial}{\partial\epsilon}E(s_{TN}) > 0\ , & \epsilon > 0, \\ \frac{\partial}{\partial\epsilon}E(s_{TN}) < 0\ , & \epsilon < 0. \end{cases}$$

The detection rate increases when the degree of attack $|\epsilon|$ *increases, or when* σ *decreases. i.e. more proofs are added.* ■

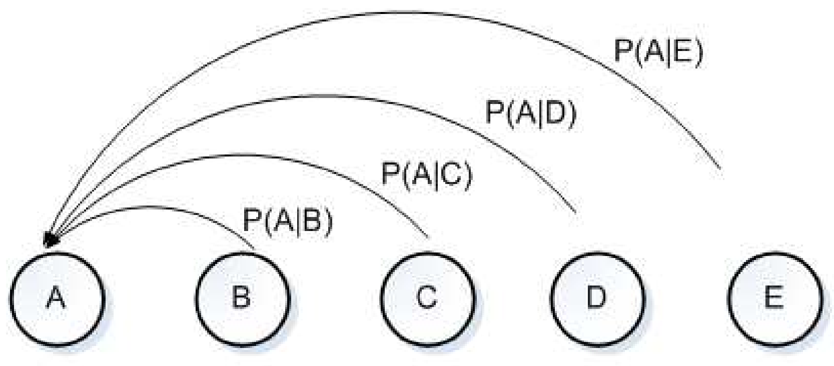


Figure 3.1: 4 posterior distributions are derived for “A”. If “A” is in the $(1 - \beta)$ confidence interval of a posterior distribution, it receives one vote.

3.3.3 Against Compromised Node

For each white reading X_i, the home server uses every other white reading to derive a number of posterior distributions of x_i. In each posterior distribution, if X_i is in a pre-defined $(1 - \beta)$ confidence interval, it receives one vote. If a white reading receives at least a fraction of γ among all votes, it is accepted as a proof. Otherwise, it is declared as a black reading. Note that to make the process independent of the order of voting, the black readings can still involve in deriving posterior distribution for other readings. Figure 3.1 illustrates the voting process.

Theorem 3.3.3 *The detection rate of malicious readings sent by a compromised node in Cobra is a monotonic function of the attack aggressiveness.*

Proof: *Let l denote the total number of legitimate nodes. Let θ denote the percentage of legitimate readings voting for a malicious reading. Let y be the random variable representing the number of legitimate nodes among nodes* *operate in the white mode.* $P(y)$ *follows a hypergeometric distribution given by*

$$P(y) = \frac{C_y^l C_{np_0-y}^{n-l}}{C_{np_0}^n}. \tag{3.13}$$

The expected detection rate is given by

$$\begin{aligned} & E(S^{TN}) \\ = \;& P((1-\theta)(y) \geq (1-\gamma)(np_0 - 1)) \\ \approx \;& 1 - P(y \leq \frac{(1-\gamma)np_0}{1-\theta}). \end{aligned} \tag{3.14}$$

From theorem 3.3.2, $(1-\theta)$ *is a monotonic function of* $|\epsilon|$*. Therefore,* $E(S^{TN})$ *is a monotonic function of* $|\epsilon|$*.* ■

The expected true positive rate is given by

$$E(S^{TP}) \approx 1 - P(y \leq \frac{\gamma np_0}{1-\beta}), \tag{3.15}$$

and the overall accuracy is given by

$$E(S^{acc}) = \frac{l}{n}E(S^{TP}) + (1 - \frac{l}{n})E(S^{TN}). \tag{3.16}$$

Take the partial derivative of Eq. 3.16 with respect to γ and set it to zero, we derive the value of γ that maximizes the detection accuracy. The solution is relevant to parameters l and θ that are not easy to estimate in real world applications. An empirical model is used where l and θ are assumed to follow certain distributions.

Assume that l and θ us independent, the value of γ in the empirical model is given by

$$\gamma = \int\int p(l)p(\theta)\gamma_{l\theta} dl d\theta, \tag{3.17}$$

where $\gamma_{l\theta}$ is the value computed using the analytical process given l, θ and $p(l), p(\theta)$ are the probability distribution functions of l and θ, respectively. In Cobra, l is assumed to follow a uniform distribution in interval $[0.1, 0.9]$ and θ is assumed to follow a uniform distribution in interval $[0.05, 0.4]$. The value of γ in the empirical model is computed as 0.482.

3.3.4 Auto-detection against all four attacks

When the network is subject to all four attacks, the home server first adopts the voting-based method to select proofs among the white readings. Next the home server checks against the packet fabrication/replay attack, leaving one unique reading for each sensor node operating in the gray mode. At last, the gray readings are checked to detect the packet modification attack. The process is illustrated in Fig. 3.2

3.3.5 Data Recovery

The real value of the black readings are estimated as the mean of their posterior distribution derived using the proofs (Eq. 2.3).

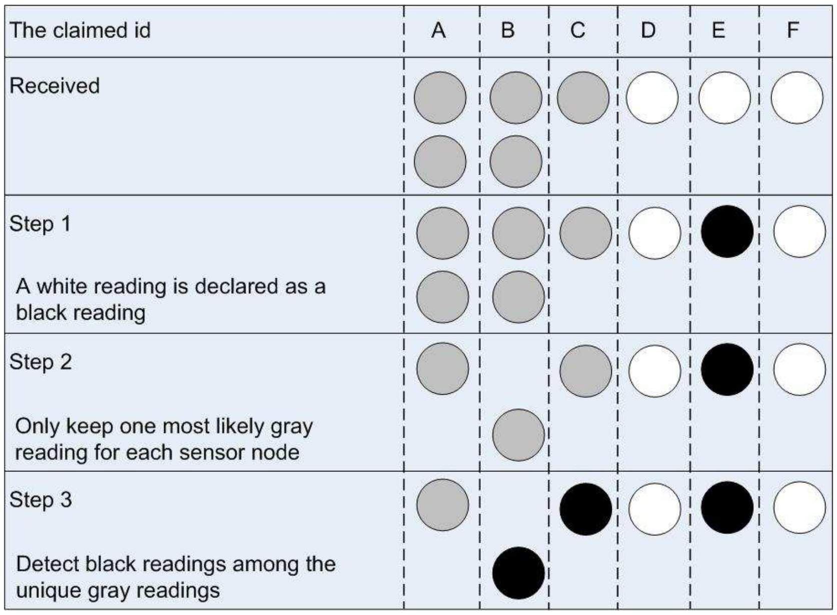

The claimed id
A
B
C
D
E
F
Received
Step 1
A white reading is declared as a black reading
Step 2
Only keep one most likely gray reading for each sensor node
Step 3
Detect black readings among the unique gray readings

Figure 3.2: The process of auto-detection against all four attacks. Step 1: detect white readings sent by the compromised nodes. Step 2: detect packet fabrication/replay attacks. Step 3: detect packet modification attacks.

3.4 Experimental result

Cobra is evaluated using the 4 weeks' traffic volumes collected at 100 locations in Chicago highway system. The data of the first week was used to train the correlation model. Two models were trained using the maximum likelihood estimator, one for weekdays and one for weekends. The length of a sampling period was 5 minutes. The following parameters of Cobra:

- p_0: the percentage of nodes operating in the white mode,
- $(1-\alpha)$: the confidence interval used in detecting packet modification attacks,
- $(1-\beta)$: the confidence interval used in detecting node compromisation attacks,

were set to various values.

In the first set of experiments, one false reading was inserted for each real reading. The fabricated readings magnified the real readings X^T by 1.2 or 1.5 times. Figure 3.3 shows the detection rate of fabricated readings against p_0. Higher aggressiveness results in higher detection rate. When $p_0 < 0.2$, adding more white readings significantly reduces the posterior variances and increases the detection rate. When $p_0 > 0.2$, adding more white readings produces only slight improvements.

In the second set of experiments, 50% of the gray readings were randomly chosen in each sampling period and magnified by 1.2 or 1.5 times. The confi-

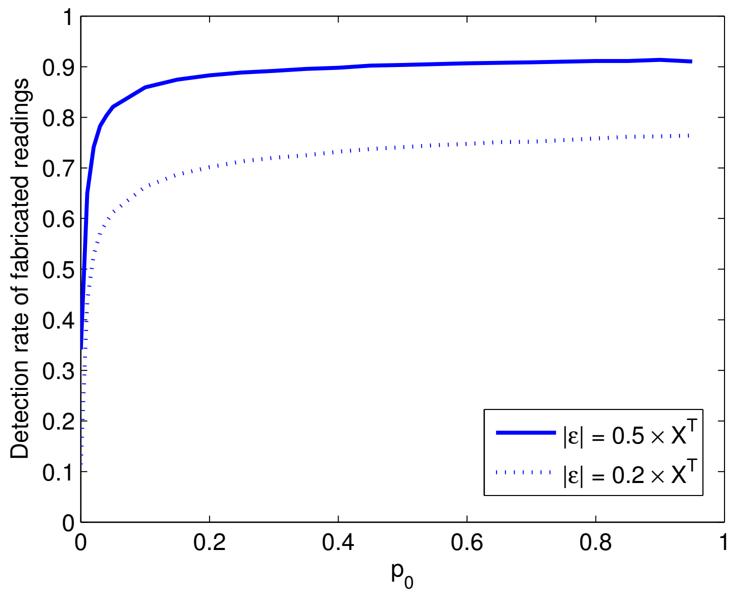


Figure 3.3: The detection rate of fabricated readings v.s. p_0.

dence interval $(1 - \alpha)$ is set to 0.95 or 0.8. Figure 3.4 shows the detection rate of modified readings against p_0. Higher detection rate is associated with higher aggressiveness, higher value of p_0, and higher value of α. Similar to the previous experiments, the detection rate starts to converge after $p_0 > 0.2$. Note that the detection rate is not always monotonically increasing in respect to p_0. This is because the data have some outliers and the outliers being included in the proofs can generate inaccurate posterior distributions. This is especially the case when p_0 is larger since it is more likely to have an outlier in the signed data packets.

In the third set of experiment, 20% of the sensor nodes were randomly chosen as compromised nodes. The readings of the compromised nodes were magnified by $1.1 \sim 2$ times. p_0 is set to 0.1 and the confidence interval $(1 - \beta)$ is set

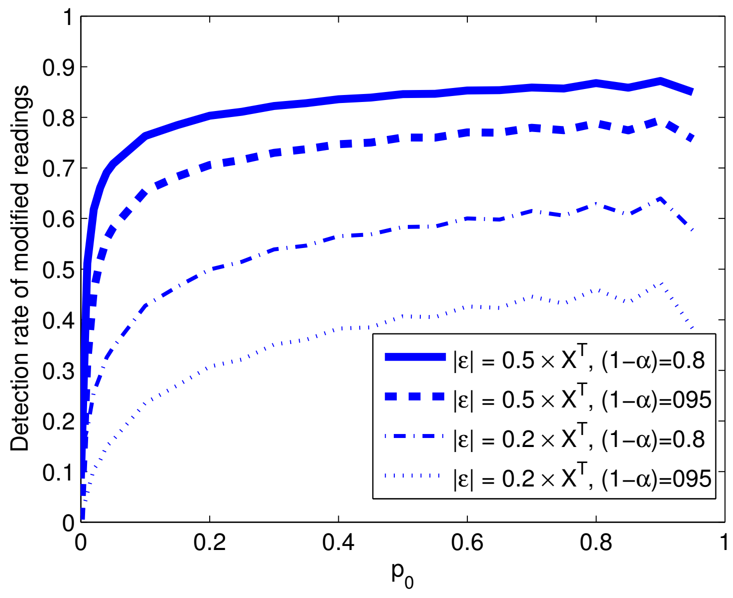


Figure 3.4: The detection rate of modified readings v.s. p_0.

to 0.8. The detection rate and the detection accuracy (Eq. 5.1) were measured among white readings only. Figure 3.5 shows that the detection rate increases dramatically as the relative attack aggressiveness increases. The true positive rate is always around 85% regardless of the attack aggressiveness. Since the number of legitimate nodes are 4 times as many as the compromised nodes, the overall detection accuracy only slightly increases as the attacks become more aggressive.

In the last set of experiments, Cobra was compared to CORA [67], MiniSec [46], and Gaussian Estimation [76] (white readings were used to estimate the values of gray readings). In Cobra, when a node operated in the white mode, it used MiniSec protocol and 3 extra bytes were added to the packet headers. The parameters were set as follows: $p_0 = 0.1$, $\beta = 0.2$, $\alpha = 0.2$. 30 networks were

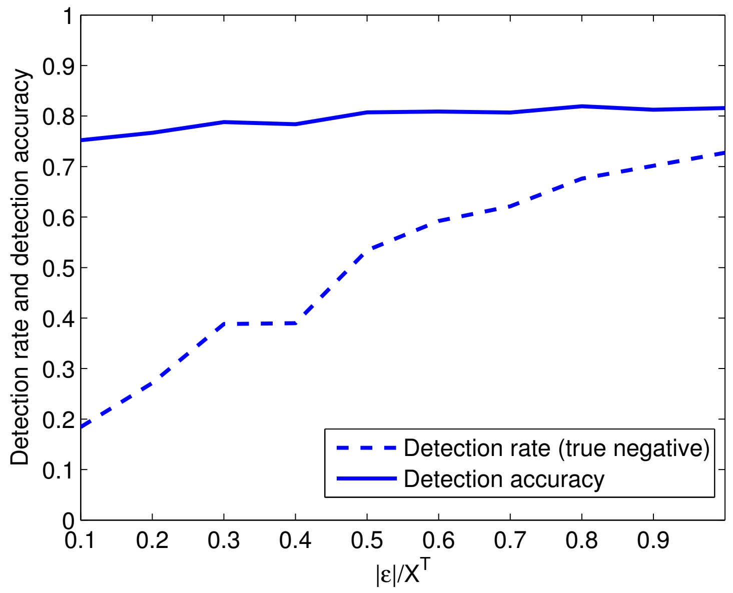


Figure 3.5: The detection rate and detection accuracy of white readings sent by compromised nodes v.s. the relative attack aggressiveness $|\epsilon|/X^T$.

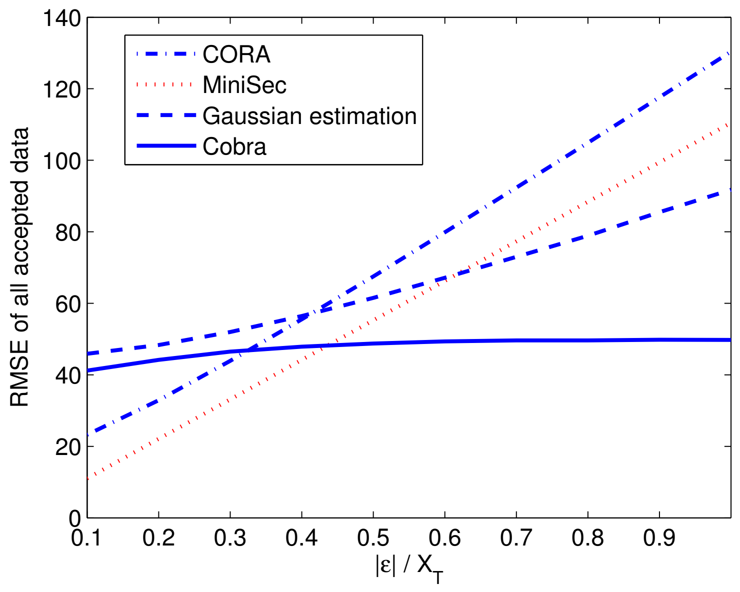


Figure 3.6: Under compromised node and packet modification attacks, Cobra's comparison to three other methods in terms of the RMSE of the accepted data.

simulated with 20% of the nodes randomly chosen as compromised nodes. The compromised nodes magnified its own readings and the readings it received before forwarding them. The packet fabrication/replay attacks were not included since CORA could not handle these attacks. The sensor nodes were placed with their real world coordinates from the Chicago data set. The communication range of a sensor node was set to 4 miles since the sensor nodes were mostly far away from each other.

Figure 3.6 shows that as the attacks become more aggressive, the RMSEs (the average of 30 simulations) of all methods except Cobra increase linearly. Cobra provides a very stable performance. Its RMSE converges when the relative aggressiveness $|\epsilon|/X^T$ is around 0.4. When the attacker is not aggressive

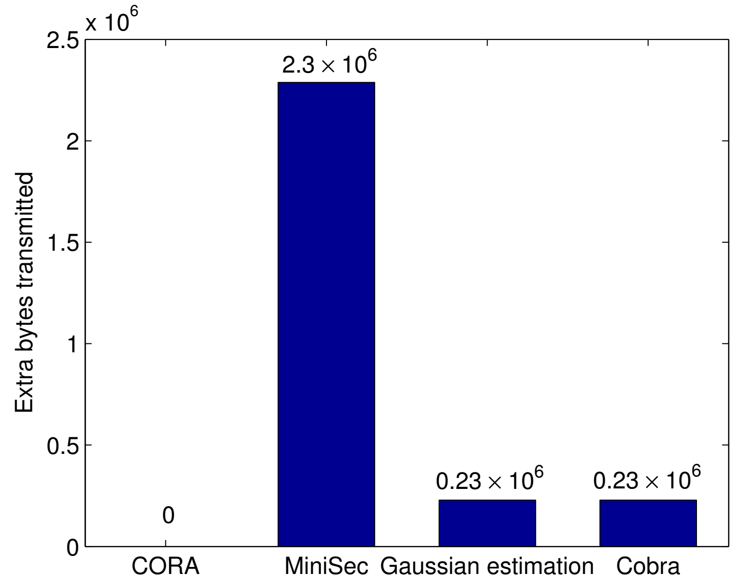


Figure 3.7: The communication overhead (number of extra transmitted bytes) of Cobra comparing to three other methods.

($|\epsilon|/X^T < 0.4$), MiniSec performs the best. This is because the biases introduced by the attacker are usually smaller than the error introduced by the data recovery.

Figure 3.7 shows the communication overhead of the four methods (CORA's overhead is 0 since no cryptography is used). Cobra only costs 10% of MiniSec. The computation overhead of Cobra is estimated at 10% of MiniSec since in each sampling period only 10% of the sensor nodes operate in the white mode.

Chapter 4

A New Distributed Method for Faulty Sensor Detection

4.1 A new metric - mutual divergence

Using the correlation model described in section 2.6.1, we propose a new metric *divergence* to measure how much the readings diverge from prior knowledge. Intuitively, if none of the sensors are faulty, all readings should fit neatly into the prior correlation model. Given two (possibly overlapping) sets of sensors L_1, L_2, their jth readings' divergence is given by

$$\delta(X^t_{L_2}, X^t_{L_1}) = \sum_{i \in L_2} \frac{(\mu_{i|X^t_{L_1 \setminus i}} - X^t_i)^2}{|L_2| \bullet COV_{i|L_1 \setminus i}}. \tag{4.1}$$

The divergence measures how much (on average) the readings of sensor nodes L_2 diverge from their posterior means derived from the readings of the reference sensor set L_1. The divergence value of one sensor's reading is normalized using its posterior standard deviation and squared in account for negative biases.

Mutual divergence is used to evaluate how good a selection of F is given the most current k readings. We have

$$\Delta(V,F) = \sum_{t=1,2,\ldots,k} \delta(X^t_{V-F}, X^t_{V-F}) - \sum_{t=1,2,\ldots,k} \delta(Y^t_F, Y^t_{V-F}). \tag{4.2}$$

The objective is to minimize $\Delta(V,F)$. In other words, the objective is to minimize the divergence among non-faulty sensor nodes $V-F$ and to maximize the divergence between non-faulty sensor nodes $V-F$ and faulty sensor nodes F. Note that for generality we do not assume that the faulty sensors' readings are uncorrelated. If such an assumption is valid in particular applications [67, 69, 61], a third term $-\sum_{j=1,2,\ldots,k} \delta(Y^t_F, Y^t_F)$ can be added.

We have the objective for the problem of faulty sensor detection as

$$\operatorname*{argmin}_F \Delta(V,F). \tag{4.3}$$

Faulty sensors' readings will be predicted based on Eq. 2.3. Since the expected root mean square error is fixed regardless of the actual readings (Eq. 2.4), the faulty sensor detection accuracy reflects the average data reliability of the final data set. In the following text we prove that mutual divergence is accurate in terms of estimating faulty sensor detection accuracy.

4.2 Theoretical analysis

Let i_0, i_1, i_2 denote three correlated sensor nodes with i_0 being the only faulty one. A correlation mode $P(i_0, i_1, i_2)$ is learned before i_0 becomes faulty. Let $\epsilon_{i_0}^t, (t = 1, 2, ..., k)$ denote the biases of i_0 in each of the k recent readings caused by its fault. From Eq. 4.1, we have

$$\begin{aligned}
& \delta(X_{i_0}^t, X_{i_1,i_2}^t, P(i_0, i_1, i_2)) \\
= \ & \frac{(\mu_{i_0|X_{i_1,i_2}^t} - X_i^t)^2}{\Sigma_{i_0|i_1,i_2}} \\
= \ & \frac{(\mu_{i_0|X_{i_1,i_2}^t} - (X_{i_0}^t + \epsilon_{i_0}^t))^2}{\Sigma_{i_0|i_1,i_2}}
\end{aligned} \tag{4.4}$$

Its expected value is given by

$$\begin{aligned}
& E(\delta(X_{i_0}^t, X_{i_1,i_2}^t, P(i_0, i_1, i_2))) \\
= \ & E(\frac{(\mu_{i_0|X_{i_1,i_2}^t} - (X_{i_0}^t + \epsilon_{i_0}^t))^2}{\Sigma_{i_0|i_1,i_2}}) \\
= \ & \sum P(X_{i_0}^t | X_{i_1,i_2}^t) \bullet \frac{(\mu_{i_0|X_{i_1,i_2}^t} - (X_{i_0}^t + \epsilon_{i_0}^t))^2}{\Sigma_{i_0|i_1,i_2}}
\end{aligned} \tag{4.5}$$

For simplicity, let μ denote the posterior mean $\mu_{i_0|X_{i_1,i_2}^t}$, X denote the true reading $X^t_{i_0}$, ϵ denote the bias $\epsilon^t_{i_0}$, and σ^2 denote $\Sigma_{i_0|i_1,i_2}$. We have

$$
\begin{aligned}
& E(\delta(X^t_{i_0}, X^t_{i_1,i_2}, P(i_0,i_1,i_2))) \\
= & \sum P(X) \bullet \frac{(\mu - (X+\epsilon))^2}{\sigma^2} \\
= & 1/\sigma^2 \bullet [\sum P(X) \bullet \mu^2 - \sum P(X) \bullet \mu(X+\epsilon) + \\
& \sum P(X)(X+\epsilon)^2] \\
= & \frac{\mu^2 - 2\mu \sum P(X)(X+\epsilon) + E(X+\epsilon)^2 + Var(X+\epsilon)}{\sigma^2} \\
= & \frac{\mu^2 - 2\mu^2 - 2\mu\epsilon + (\mu+\epsilon)^2 + \sigma^2}{\sigma^2} \\
= & \frac{\sigma^2+\epsilon^2}{\sigma^2} = 1 + \frac{\epsilon^2}{\sigma^2} \qquad (4.6)
\end{aligned}
$$

For given i_0, i_1, i_2, σ^2 is fixed and can be viewed as a constant. Now let us consider the divergence of an arbitrary faulty set L_1 and an arbitrary non-faulty set L_2. We have

$$
\begin{aligned}
& \frac{1}{|L_2|} \sum_{i \in L_2} E(\delta(X^t_i, X^t_{L_1}, P(L_2, L_1))) \\
= & \frac{1}{|L_2|} \sum_{i \in L_2} P(X^t_i | X^t_{L_1}) \bullet \frac{(\mu_{i|Y^t_{L_1 \setminus i}} - (X^t_i + \epsilon^t_i))^2}{\Sigma_{i|L_1 \setminus i}}. \\
= & \frac{1}{|L_2|} \sum_{i \in L_2} (1 + \frac{\epsilon^t_i}{\Sigma_{i|L_1}}). \qquad (4.7)
\end{aligned}
$$

Equation 4.7 leads to the following theorem:

Theorem: *The expected divergence of a sensor set L_2 in reference to another set L_1 is a monotonically increasing function of the biases in sensor set L_2 and a* *monotonically decreasing function of L_2's posterior variance.*

The theorem guarantees the correctness of the objective function defined in Eq. 4.2. Between non-faulty sensor nodes, the expected divergence is 1 ($\epsilon = 0$) according to Eq. 4.6. Between faulty and non-faulty sensor nodes, the expected divergence reflects the amount of bias. The sensitivity of the divergence function can be increased by introducing smaller posterior variance. This can be achieved by including more reference sensor nodes and/or by selecting more correlated sensor nodes as references.

4.3 A distributed detection algorithm

Using PC theory, the objective function Eq. 4.3 is rewritten as

$$\operatorname*{argmin}_{F} \sum P(F)\Delta(V, F). \tag{4.8}$$

The private goal function for a sensor i is defined as

$$\Delta(N(i), F(i)), \tag{4.9}$$

where $N(i)$ is i's neighboring node and itself, and $F(i)$ is the faulty sensor nodes among i's neighbors and itself. Usually nearby sensor nodes are more correlated. Such a locality feature helps to increase the detection rate even if only the neighboring sensor nodes are used as references.

The probability updating rule is given by

$$\begin{aligned} P_i(f_i) &= P_i(f_i) - \alpha \times P_i(f_i) \times \\ &\quad \{(E(\Delta(N(i), F(i))|f_i) - E(\Delta(N(i), F(i))))/T + \\ &\quad S(P_i) + lnP_i(f_i)\}, \end{aligned} \tag{4.10}$$

Figure 4.1 describes the details of the distributed collective detection algorithm.

At the end of the algorithm, if a sensor node's faulty probability is higher than a certain threshold, it is regarded as faulty. Decreasing this threshold increase detection rate (true negative) but also produces more false alarms. In our experiments we have found that 0.95 is a sufficiently high threshold. Note that even if a sensor node is marked as faulty, it still participates in the future decision making process. This gives the sensor node a chance to recover if the faults disappear.

4.4 Experimental result

For comparison purpose, two base-line algorithms are implemented in both centralized and distributed manners.

4.4.1 A centralized detection method

In the centralized detection method, the base-station handles the detection process. In each decision cycle, the base-station makes decision about the faulty sensors

```
Distributed collective detection
 (decision 1: non-faulty, 0: faulty)

  for each sensor $i$
    //set initial decision as a uniform distribution
    $p_i(0) \leftarrow 0.5$;
    $p_i(1) \leftarrow 0.5$;
  end
  loop
    for each sensor $i$
      $p_{N(i)\backslash i} \leftarrow neighbors'\, faulty\, probabilities$;
      $s \leftarrow n$ samples from $p_{N(i)}$;
      $E[\Delta] \leftarrow 0$;
      $E[\Delta|0] \leftarrow 0$;
      $E[\Delta|1] \leftarrow 0$;
      $h \leftarrow 0$; //number of samples where $i$ is faulty
      for each sample $r$
        $g_r \leftarrow \Delta(N(i), F_(i))$; //Eq. 4.2
        $E[\Delta] \leftarrow E[\Delta] + g_r$;
        if($f_i$ = 1)
          $E[\Delta|1] \leftarrow E[\Delta|1] + g_r$;
          $h++$;
        else
          $E[\Delta|0] \leftarrow E[\Delta|0] + g_r$;
        end
      end
      $E[\Delta] \leftarrow E[\Delta]/n$ ;
      $E[\Delta|1] \leftarrow E[\Delta|1]/h$ ;
      $E[\Delta|0] \leftarrow E[\Delta|0]/(n-h)$ ;
      update $p_i(0), p_i(1)$ based on $E[\Delta], E[\Delta|1], E[\Delta|0]$; //Eq. 4.10
      inform neighbors about the updated $p_i(0), p_i(1)$;
    end
    terminate after a certain number of iterations;
  end
```

Figure 4.1: Algorithm of distributed collective detection algorithm.

solely based on the k most recent readings received from each sensor node. In this paper, we implemented a simple random sampling method for comparison purpose. A based-station randomly draws m samples. In each sample a sensor is either faulty or non-faulty. The base-station computes the mutual divergence value (Eq. 4.2) for each sample and chooses the sample with the minimal mutual divergence.

4.4.2 Distributed simultaneous detection with pure decision

A sensor node receives broadcasts from its neighbors consisting their most recent k readings. It also receives the current decision (faulty/non-faulty) of each neighboring sensor node. A sensor node decides that it is either faulty or non-faulty based on which one yields smaller mutual divergence in the neighborhood (Eq. 4.9).

The iterative detection method starts with a random assignment of each sensor node being faulty or non-faulty. In each iteration, if a sensor node has changed its decision in the last iteration, it broadcasts the new decision to the neighbors. After a sensor node receives updated decision from one of the neighbors, it recomputes its mutual divergence and may choose to change its decision accordingly. The search process terminates if no more change is made.

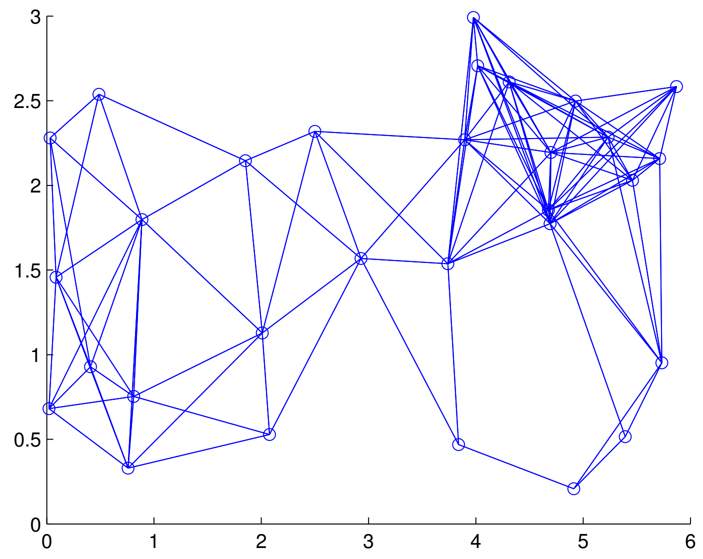


Figure 4.2: 30 sensors deployed in a 6×3 field. The sensors' communication radius are 2 and the links represent the communication links.

4.4.3 Comparison result

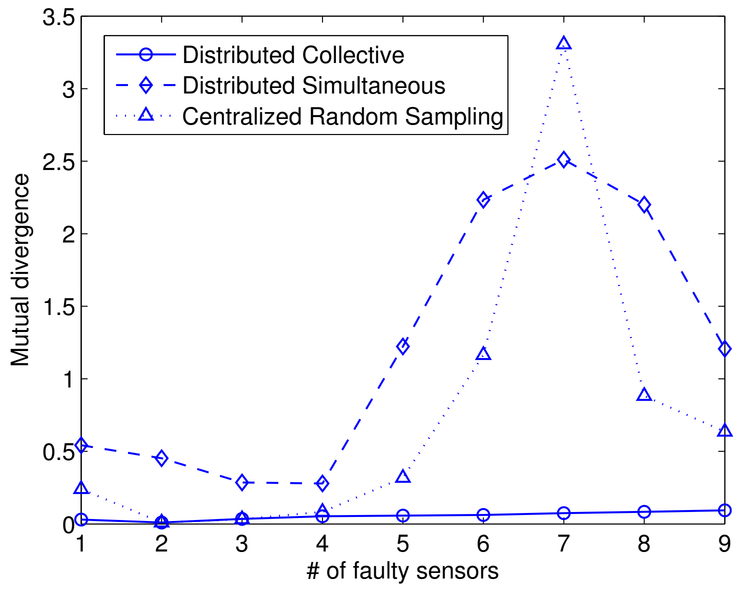
We have conducted three sets of experiments in MATLAB to evaluate the three detection methods. 30 sensors are randomly deployed in a 6×3 field. An example of such sensor network is shown in figure 4.2. The background data was simulated as temperature influenced by a 6 randomly placed heat sources. A random Gaussian noise is added to all data. The mean of the noises was set to zero and the standard deviation was set to 10% of the real readings' standard deviation. A set of data was used to train the joint distribution and another set of similar data is used for testing. Note that the noises were presented in both the training set and the test set. Therefore the trained correlation model already reflects the noises.

The communication radius of a sensor was set to 2. In the distributed detection methods, a sensor node makes decision based on readings received from neighbors within the communication radius. In the distributed collective method, the initial decision distribution of each sensor is set to a uniform distribution. In each iteration, 25 samples are drawn at each sensor to evaluate the distribution.

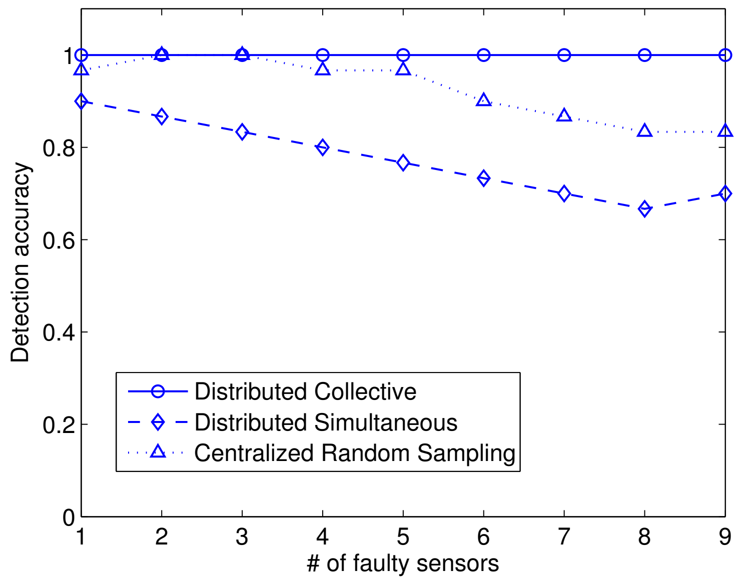
In the centralized random sampling method, 300 samples are randomly drawn from a uniform distribution and the one with the smallest mutual divergence value is used as the detection result.

In the first set of experiments, we test the three proposed detection methods with the random fault type. A fraction of the sensor nodes are randomly selected as faulty sensors. We vary the number of faulty sensors from 1 to 9. The broadcasted reading of each faulty sensor is replaced by a random number independently drawn from a uniform distribution in region $(0, 300)$. Such a fault model is selected since it yields uncorrelated data in the same magnitude as the real readings.

Figure 4.3(a) shows the mutual divergence achieved by different methods. Out of the three proposed methods, the distributed collective method achieves the smallest value, followed by the centralized random sampling method and then the distributed simultaneous method. The distributed simultaneous method performs poorly since the search space is filled with local optimal. The random sampling can usually find better results. In addition, the mutual divergence found by the distributed simultaneous is unpredictable at best. The method may fall into different local optimal given different random starting points. The algorithm of the



(a)



(b)

Figure 4.3: (a) Mutual divergence achieved by three proposed detection methods when random faults present. (b) The detection accuracy of the three methods when random faults present.

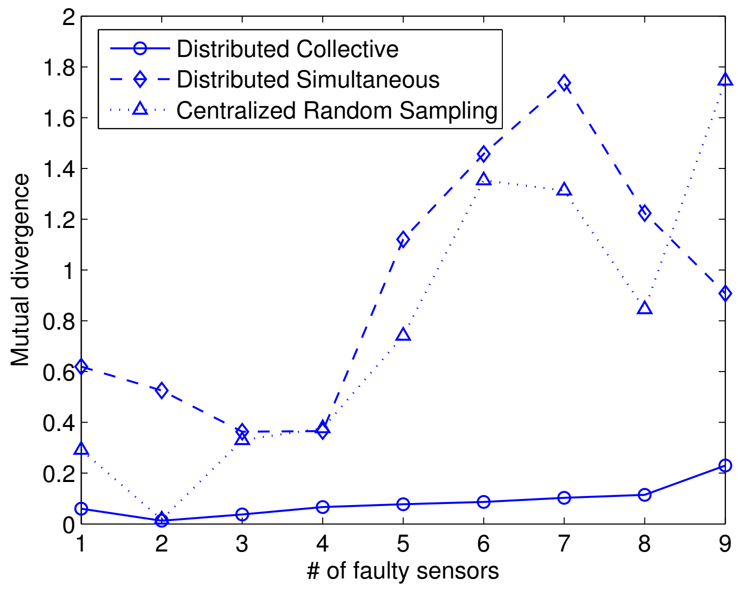
centralized random sampling suffers from the same problem.

Figure 4.3(b) show the detection accuracy computed as

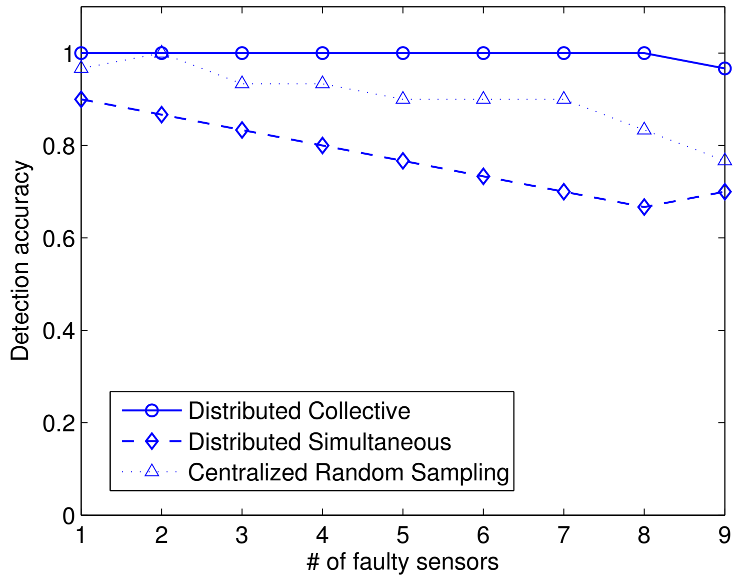
$$accuracy = \frac{true\,positive + true\,negative}{all}. \tag{4.11}$$

The distributed collective method achieves a constant detection accuracy of 100%, outperforming the other two methods in all test cases. As the number of faulty sensor nodes increases, the problem becomes harder. Yet the distributed collective method is still able to find the true faulty sensor set even when 30% of the sensors are faulty. Note that when the number of faulty sensors is 7, the centralized random sampling method achieves higher detection accuracy than the distributed simultaneous method despite have higher mutual divergence value. This is because mutual divergence does not directly measure the number of faulty sensors. It instead measures how much the faults diverge from the real value. It is not uncommon that a faulty sensor exhibits larger divergence than the average divergence of several other faulty sensors.

In the second set of experiments, we test the detection methods with drift fault type. $1 \sim 9$ sensor nodes are randomly picked as faulty sensors and their readings X are altered by a linear function $f(X) = aX + b$. We set a to 1.5 and b to -40. Such a fault model is selected to post probably the hardest challenge for the detection methods – the faulty sensors are correlated. This is likely to happen in the case of malicious attacks where the attacker somehow acquires the network correlation model and introduces correlated faults to avoid detection.



(a)



(b)

Figure 4.4: (a) Mutual divergence achieved by three proposed detection methods when drift faults present. (b) The detection accuracy of the three methods when drift faults present.

Figure 4.4(a) and figure 4.4(b) show the mutual divergence and detection accuracy achieved by the three methods. Despite facing a more challenging problem, the distributed collective method still achieves very high detection accuracy. It only misses one faulty sensor node when as many as 30% sensors are faulty. In that case, the faulty sensor node is surrounded by other faulty sensor nodes whose readings can actually validate the its faulty readings. To improve detection rate in such cases one may have to increase the neighborhood size, either by increasing communication radius or by including 2-hop neighbors. Note that this also considerably increases the communication and computation overhead. In practical system it is unlikely that 30% sensor nodes becomes faulty simultaneously. A faulty sensor node can be detected as soon as it becomes faulty and the percentage of active faulty sensor node can be limited to a small number.

Chapter 5

SMART - A New System for Simultaneous Indoor Localization and Map Construction

5.1 Problem formulation

A logical map is represented by a $W \times H$ matrix where each entry represents a square patch of unit width/height. Each patch is labelled with a logical location fingerprint (e.g., restaurant, pub, bookstore, etc.). Figure 5.1 illustrates a logical map with 5 different fingerprints. Each of the four shaded rectangles represents a distinct logical location (a shop) and the white area represents the hallway. k WiFi access points are placed at either random locations or pre-selected points. Note that in reality WiFi access points are usually placed in a systematic manner

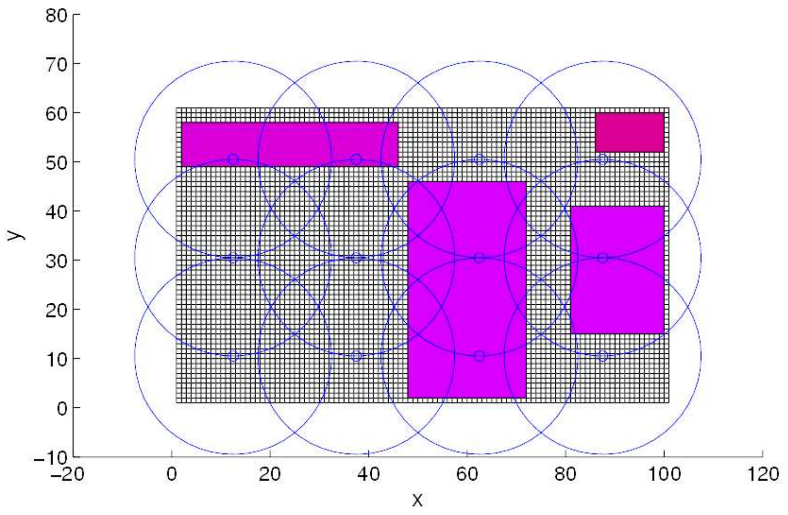


Figure 5.1: A sample field with 5 logical locations: 4 shaded areas as 4 stores and white region is the hallway. 12 access points are placed as a grid with their communication range shown by the circles.

to maximize coverage. A mobile subject moves at random speed and random directions. Let r denote the maximal communication range of an access point. For simplicity, we assume a mobile subject can hear the access point if their distance is shorter than r. In figure 5.1, access points are represented by small dots and their range are marked with the large circles.

For simplicity, time is also divided into unity length time steps.

There are m mobile subjects carrying smart phones and randomly walking within the field. The motion dynamics are given by

- Speed: the distance a subject moves in one time step is uniformly distributed in an interval (v_{min}, v_{max});
- Heading: each subject starts with a random heading and randomly chooses

a turning angle from a uniformly distribution in an interval $(-\rho, \rho)$. For simplicity we assume a subject maintain its heading in one time step.

A mobile subject has the ability to estimate its initial location, speed and heading, with random Gaussian noise. On a smart phone, these are likely to be acquired by assisted GPS [44], accelerometers [55] and magnetometers, respectively. We also assume a mobile subject can identify the fingerprint of the patch containing its current location using methods such as SurroundSense [4].

Let M' denote the estimated logical map (a $W \times H$ matrix) and let M denote the ground truth. Our goal is to minimize the difference between M and M'. The estimation accuracy is defined as the percentage of correctly estimated patches, given by

$$\frac{\sum_{x=1}^{W} \sum_{y=1}^{H} 1|_{M(x,y)=M'(x,y)}}{W \times H}, \tag{5.1}$$

where $1|_L = 1$ if L is true, and equals to 0 otherwise.

5.2 System overview

SMART consists of three components, localization, ambient fingerprinting, and map construction. Figure 5.2 shows the system architecture.

Localization: sensor inputs are feed into the localization component to generate location estimates. First, raw inputs are converted to meaningful measurements, such as from 3-axis acceleration to speed and from magnetometer readings to motion directions. Next, speed, heading and WiFi proximity information are

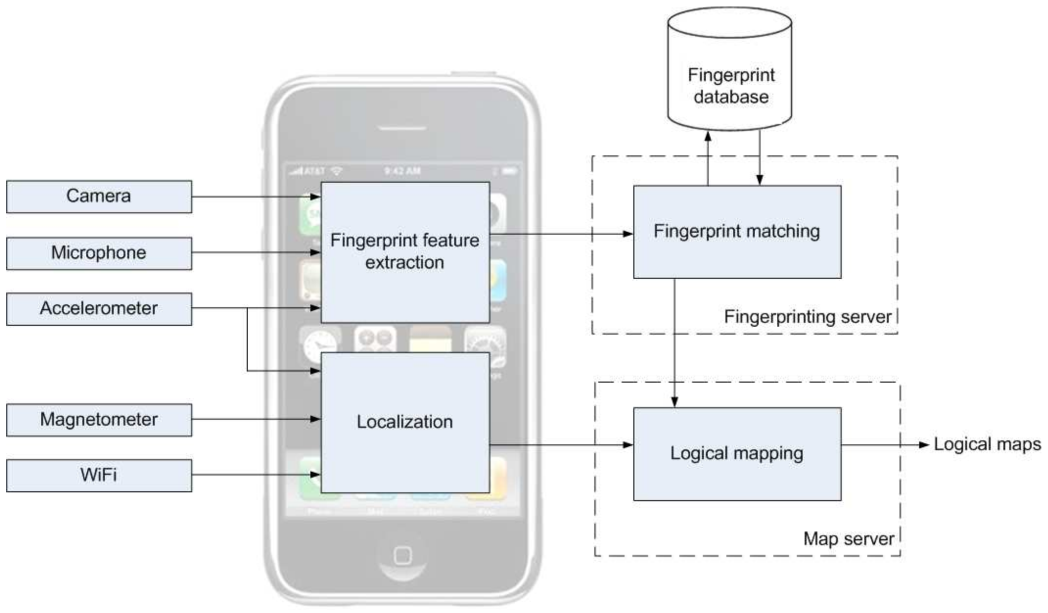


Figure 5.2: SMART architecture: fingerprint extraction and localization are performed on a mobile device. Fingerprint matching and map construction are conducted on servers.

combined in a particle filter to generate location estimates. The above procedure is fully implemented on the mobile device without the need of service side computing.

Ambient fingerprinting: it includes two sub-components. First, fingerprint features are extracted from sensor raw inputs on mobile devices. The features are transmitted to a fingerprinting server to match with entries from a fingerprint database.

Map construction: it takes inputs from both the localization component and the fingerprinting component. It also combines the inputs from multiple users to generate a new logical location map. During map construction, an incremental update approach is adopted, i.e., a new map will be integrated into the existing map rather than replaces the existing map.

5.3 Simultaneous localization and map construction

5.3.1 Particle filtering based localization

The basic idea of particle filtering is to draw N samples around an initial estimate, and update each sample according to the motion dynamics (e.g., the measured speed and heading). A sample is commonly referred to as *a particle*. Observations (e.g., WiFi proximity information) are used to filter out unlikely particles and likely particles are usually duplicated several times through a resampling process. The particles represent the location distribution and their centroid can be used as a point estimate.

Initialization: we assume that each subject holds an initial location estimate. It is acquired as the last available GPS reading before the subject enters the field. A random Gaussian error is added to the estimate to represent the error of the GPS reading. N initial samples $(x_i^0, y_i^0), i = 1, 2, ..., N$ (x, y are coordinates)are drawn from a Gaussian distribution whose mean is the initial location estimate and whose variance is the variance of the Gaussian error. A weight w_i^0 is assigned to each sample. For simplicity, we assume all sample carries equal weights, i.e., $w_i^0 = \frac{1}{N}$.

Update the samples: let v denote the measured motion speed and δ denote the measured motion direction. v and δ are subject to random zero-mean Gaussian errors whose variances are σ_v^2 and σ_δ^2, respectively. A sample (x_i^t, y_i^t) at time step t is updated as

$$x_i^{t+1} = x_i^t + (v + \epsilon_v) \times cos(\delta + \epsilon_\delta), \tag{5.2}$$

$$y_i^{t+1} = y_i^t + (v + \epsilon_v) \times sin(\delta + \epsilon_\delta), \tag{5.3}$$

where ϵ_v and ϵ_δ are artificial measurement variances drawn from Gaussian distribution $N(0, \sigma_v^2)$ and $N(0, \sigma_\delta^2)$, respectively. The purpose of adding these two variances are two-fold. First, it accounts for the measurement errors. Second, it significantly reduces the probability of updating two samples to the same location.

Update the sample weights: the sample weights are updated according to the following rules

- If a sample falls outside the boundary, its weight is set to 0;
- Compare the access point proximity of a sample to the measured access point proximity. If they do not match, set the sample weight to 0. The proximity of the sample is estimated according to the access point locations and their communication range;
- For all other conditions, maintain the sample weight at the last time step.

In other words, the sample weight update is to filter out unlikely samples and keep the likely ones.

Resample: N samples are selected from the samples from the last time step based on their updated weights. Samples with zero weights are guaranteed to be left behind and other samples may be duplicated. In cases where all samples have zero weights, N new samples are uniformly drawn from the region covered by all heard access points, or from the region with no WiFi coverage if the subject does not hear any access point.

Figure 5.3 describes the detailed algorithm.

5.3.2 Ambient fingerprinting

Azizyan et. al. have introduced a ambient fingerprinting system SurroundSense [4]. The fingerprints include acceleration, audio and visual fingerprints. Figure 5.4 shows an iPhone application we used to collect sensor data, and the accelerometer readings at a hallway, a Starbucks, and a local Bestbuy's laptop section. The difference can be easily identified. Two states *"stay"* and *"motion"* are defined

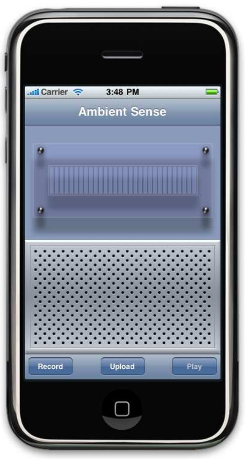
```
Localize($x^0$, $y^0$)
  //$x^0 = (x_1^0, x_2^0, ..., x_N^0), y^0 = (y_1^0, y_2^0, ..., y_N^0)$
  for $t \leftarrow 1, 2, ..., T$
    $v \leftarrow$ measured speed; $\delta \leftarrow$ measured heading;
    for $i \leftarrow 1, 2, ..., N$
      $v' \leftarrow v + \epsilon_v$; $\delta' \leftarrow \delta + \epsilon_\delta$;
      $x_i^t \leftarrow x_i^{t-1} + v' \times cos(\delta')$;
      $y_i^t \leftarrow y_i^{t-1} + v' \times sin(\delta')$;
      if $(x_i, y_i)$ contradict to observations
        $w_i \leftarrow 0$;
      else
        $w_i \leftarrow \frac{1}{N}$;
      end
    end
  $(x^t, y^t) \leftarrow resample(x^t, y^t, N)$;
  end
```

Figure 5.3: Algorithm of particle filtering based localization.

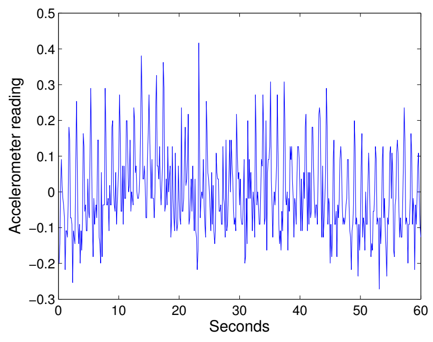
based on the variations of readings (over a window size of 8). The acceleration fingerprints are defined as the percentage of *"stay"*, the percentage of *"motion"*, and the number of state switches. Audio fingerprint is based on the amplitude frequency and visual fingerprint is based on the hue, saturation and lightness of the scene. Readers are encouraged to read [4] for additional details about ambient fingerprinting.

5.3.3 Construct probability map

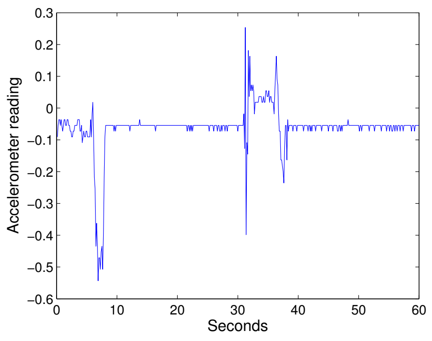
Derive a probability map based on the particles: given N location particles $(x_1, y_1), (x_2, y_2), ..., (x_N, y_N)$, a Gaussian kernel is used to derive the location dis-



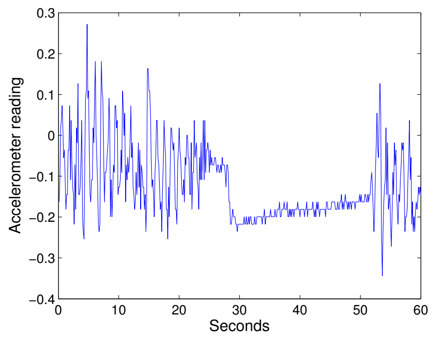
(a)



(b)



(c)



(d)

Figure 5.4: (a) An iPhone application that collects accelerometer, magnetometer, and microphone readings. (b) Accelerometer readings collected when the test subject was walking at a hallway. (c) Accelerometer readings collected when the test subject was using the cell phone at a coffee table at a Starbucks. (d) Accelerometer readings collected when the test subject was shopping at a local Bestbuy's laptop section.

tribution. The probability function $\widehat{f}(x, y)$ of a point (x, y) is given by

$$\widehat{f}(x, y) = \frac{1}{Nh} \sum_{i=1}^{N} K(x_i - x, y_i - y, h), \tag{5.4}$$

where

$$K(x_i - x, y_i - y, h) = \frac{1}{\sqrt{2\pi}} exp(-\frac{(x_i - x)^2 + (y_i - y)^2}{2h^2}), \tag{5.5}$$

and h is the bandwidth of the kernel. Figure 5.5 shows an example of location distribution when $h = 20$.

Let j denote the fingerprint sensed by a subject i's mobile phone. Its probability map f_i^j is set to the values of $\widehat{f}$.

Update the probability map: for each f_i^j, update the probability map of fingerprint j as

$$M^j = M^j + \alpha \times f_i^j, \tag{5.6}$$

where α is the step size. After all signatures have been updated, the probabilities are re-normalized. Next, each patch in the map is assigned with the most likely fingerprint. The resulting map is the fingerprint map M'. Note that if two or more fingerprints have the same probability at a patch, we set a tie breaker to always prefer the hallway's fingerprint, or randomly pick a fingerprint if the hallway is not among the candidates.

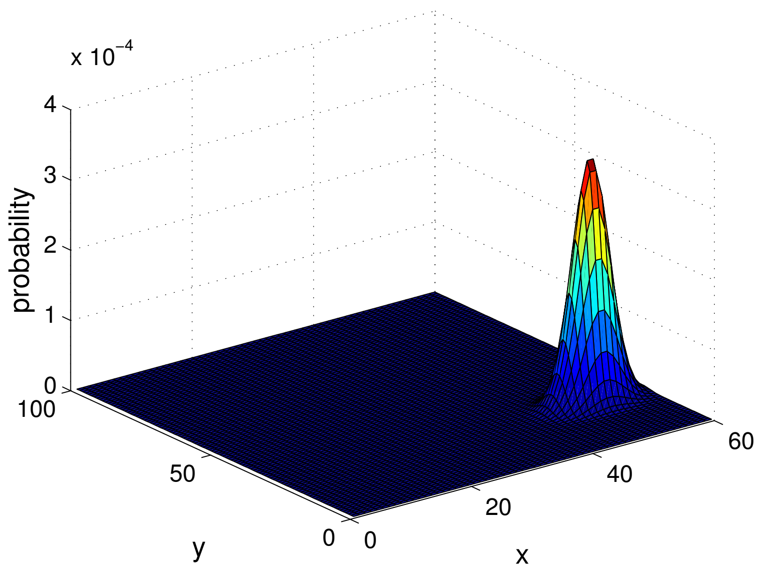


Figure 5.5: The location distribution derived from the location particles.

5.4 Experimental result

The simulations were conducted using MATLAB with test fields similar to Figure 5.1. The test parameters are listed in Figure 5.6. We ran each set of experiments for 30 random trials and presented the average results.

We first evaluated the localization accuracy over trials of 1000 time steps. For comparison purposes, we implemented a dead reckoning method where each subject used measured speed and heading, and its initial location estimate to track its physical location. The measurement errors were the same as those listed in Figure 5.6. The localization error, was defined as the distance from the estimated location to the true location. In the particle filtering based method, the centroid of all particles was used as the location estimate. Two access point placements

parameter name	**value**
width (W)	100
height (H)	60
number of access points	12
number of subjects (m)	6
number of fingerprints	5
minimal moving speed (v_{min})	0
maximal moving speed (v_{max})	3
turning angle ($(-\rho, \rho)$)	$(-0.25\pi, 0.25\pi)$
initial location est. err. std.	1
speed est. err. std.	0.05
turning angle est. err. std.	0.05
number of particles (N)	50
kernel bandwidth (h)	20
probability update step size (α)	0.5

Figure 5.6: Simulation parameters.

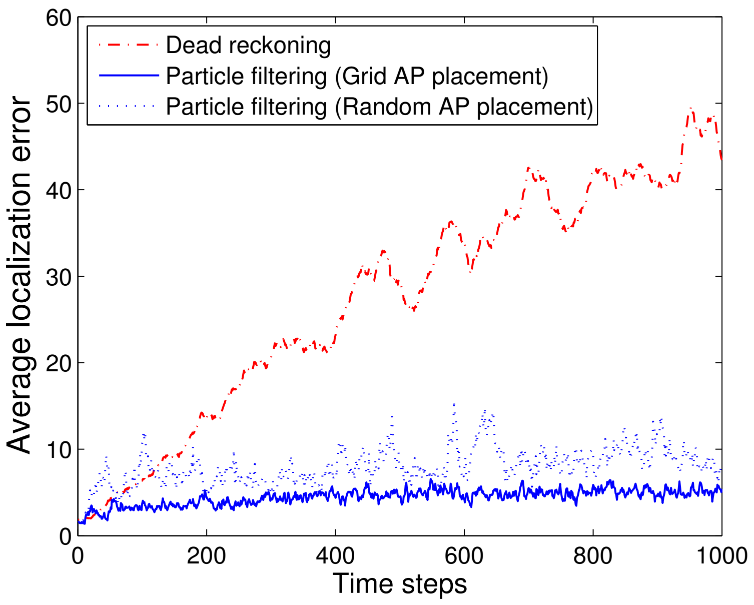
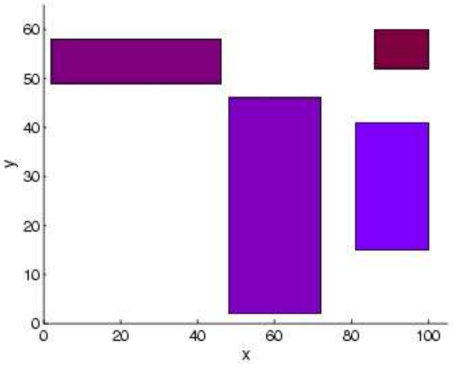


Figure 5.7: Location estimation error of dead reckoning, and particle filtering with different access point placements.

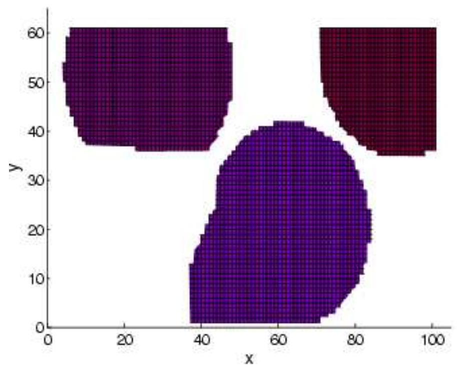
were tested, grid placement, where all access points were placed at manually selected grid points (e.g., figure 5.1), and random placement, where access points were placed randomly. In the trials of random placements, we have noticed that it was not uncommon that two access points were placed very close (distance less than 20% of the communication range) and complete WiFi coverage was difficult to achieve. In reality, the scenario of manually selected placement is usually preferred in order to maximize coverage and access point usage.

Figure 5.7 plots the location estimation error. Dead reckoning method preformed poorly and the errors propagated over the course of tracking. After 1000 time steps, the average error became stable since it reached the maximal average error bounded by the field. The particle filtering based method with random access point placement came in second, with a stable performance but larger errors in early time steps. This is because the first particle cloud was created around a flawed initial measurement. The variance of all particles was larger than the variance of the initial measurements. This can be proved by the theorem that the variance of two Gaussian random variables summed together is larger than the variance of each variable. The particle filtering based method with grid access point placement demonstrated consistently lower error than both methods. This is because the fields were fully covered by WiFi signals and the particle cloud always found observations to refine its range.

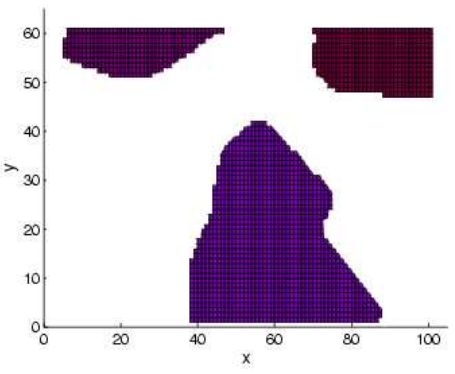
Next, we evaluated the results of logical map construction. Figure 5.8(a) shows a logical map and figure 5.8(b)$\sim$5.8(e) are the estimation results after the 1st, 10th, 50th, and 2000th time steps. After the first time step, in addition to iden-



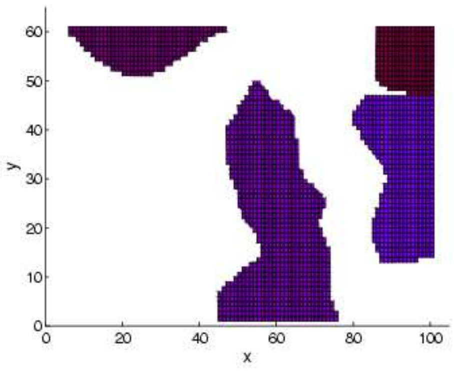
(a)



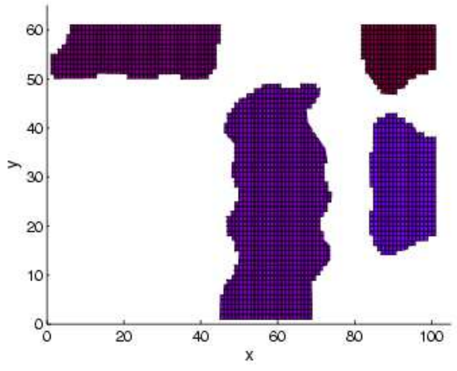
(b)



(c)



(d)



(e)

Figure 5.8: (a) The fingerprint map (ground truth). (b) The estimated fingerprint map after the first time step. (c) The estimated fingerprint map after time step 10. (d) The estimated fingerprint map after time step 50. (e) The estimated fingerprint map after time step 2000.

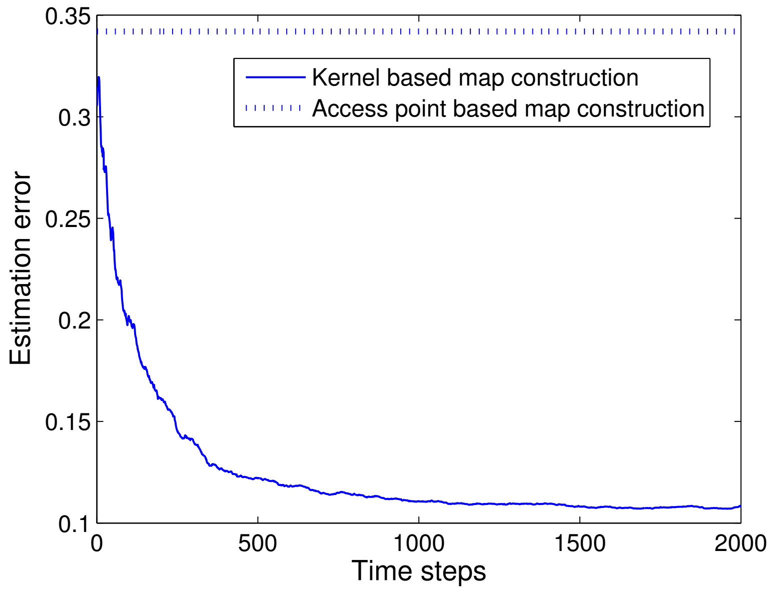


Figure 5.9: Percentage of estimation errors for kernel based method and access point based method.

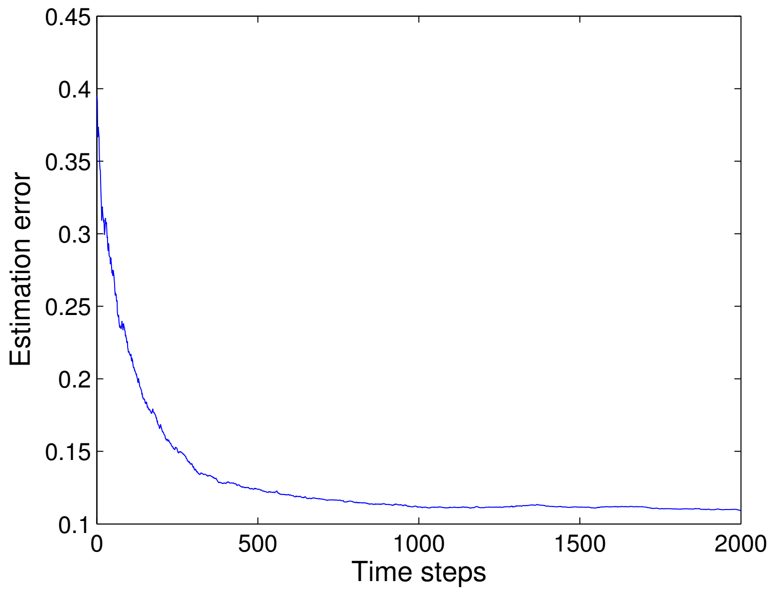


Figure 5.10: When fingerprint sensing carries 20% error, the percentage of estimation errors for kernel based method.

tifying a portion of the hallway, the kernel based method was able to find regions containing 3 out of 4 shops. The last shop was not identified because no mobile subject had visited it. It remained undiscovered until the 50th time step. Afterwards the fingerprint map continued to improve but with diminishing returns. In fact after about 1500 time steps the results tended to be stable and was less than 12% different than the ground truth.

We compared the kernel based method to a WiFi access point based method, where the fingerprints at each access point was registered and within its coverage we assumed the same fingerprint. If one patch was covered by multiple access points, a fingerprint was randomly chosen with equal probabilities. The access points were placed in a grid manner for both the kernel based method and the

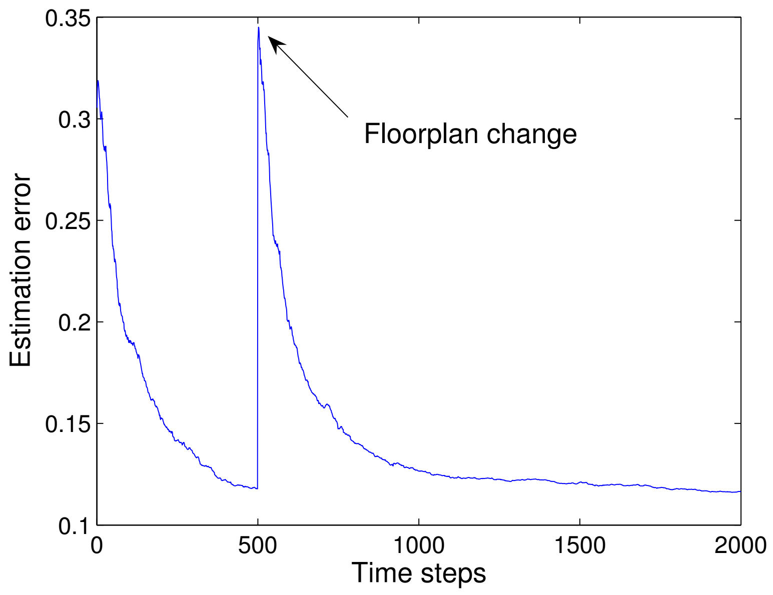


Figure 5.11: At time step 500, the floor plans are changed. This is the percentage of estimation error before and after the changes.

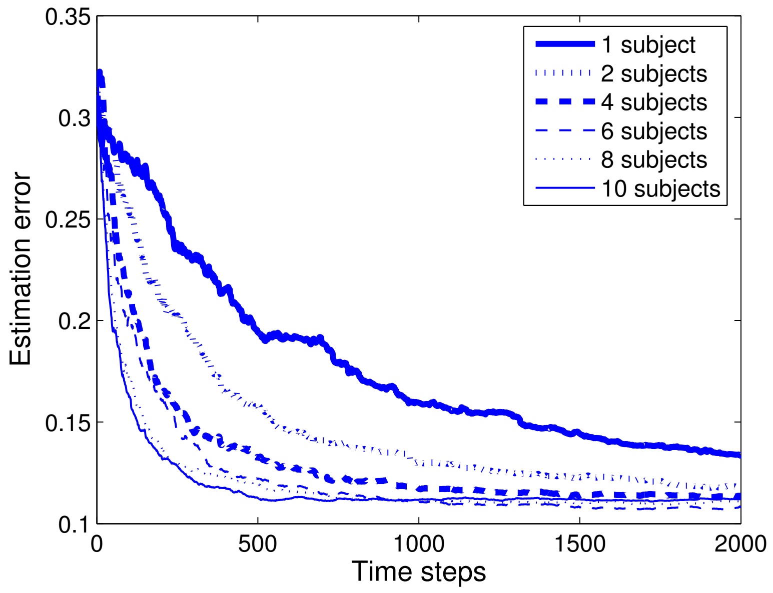


Figure 5.12: Percentage of estimation errors when there are 1, 2, 4, 6, 8 and 10 mobile subjects.

# of shops	**error of kernel method after 2000 time steps** (v_1)	**error of access point method** (v_2)	v_2/v_1
4	10.8%	34.2%	3.2
7	16.8%	50%	3.0
10	22.2%	61.2%	2.8

Figure 5.13: Fingerprint map estimation errors when there are 4, 7, and 10 shops.

access point based method. Figure 5.9 plots the comparison results. The access point based method resulted in 34.2% error on average whereas the kernel based method achieved an average error as low as 10.8%.

The kernel based method is also robust against sensing errors. We set 20% of the fingerprint readings to be faulty, and re-ran the experiments with the same setups. Figure 5.10 plots the estimation errors, which is almost identical to the experiments without sensing errors. During the 20% time when fingerprint readings were faulty, the faulty readings had equal probability to be one of the 4 other fingerprints (5% each). Therefore, the correct fingerprint (80% chance) still dominated the sensing results and its influence significantly outweighted those of the faulty readings. To prove this, we have pushed the error rate to 50% and the performance penalty remained minimal.

In reality, the floorplans may change over time. In the next set of experiments, the floorplans were changed at time step 500 without alerting the kernel based method. Figure 5.11 plots the average estimation error of 30 runs. The kernel based method automatically detected the change and makes adjustments to the fingerprint map. After 2000 time steps, the average estimation error converged around 12%.

To understand how the number of mobile subjects affects the estimation accuracy, we varied the number of subjects ($m = 1, 2, 4, 6, 8, 10$). Figure 5.12 plots the average estimation errors. As we expected, less subjects led to slower convergence. When there were only 1 mobile subject, the result did not converge until the 5000th time step. However, regardless of the subject numbers, when the curves did converge, the average estimation errors were always around 11%$\sim$12%. This leads to the conclusion that as long as the field is fully discovered by the subject(s), the system should reach a stable and accurate fingerprint map.

We also challenged the kernel based method by introducing more fingerprints. Figure 5.13 lists the estimation errors of the kernel based method and the access point based methods when 4, 7 and 10 shops are presented (5, 8 and 11 total fingerprints, respectively). As the problem became more difficult, the estimation error of the kernel method increased almost linearly. When there were 10 shops, the average estimation error converged around 22.2%, which was still 2.8 times lower than the average error of the access point based method.

5.5 Discussion

The logical map may be further fine-tuned to suit the needs of applications. For example, the center of an estimated region can be used as a point estimate for POI lookup services. Alternatively, each region can also be fit into a bounding box or a polygon for display purposes. We also envision the development of automatic map matching, where a precise floorplan is matched to a particular region of local/national fingerprint map. This is useful if the owner of a building wants to represent his/her properties in more details.

Chapter 6

A New Method for Sensor Data Prediction

6.1 Problem formulation

In chapter 3 and chapter 4, the correlation models are pre-learned using machine learning based methods. In this chapter, we focus on the application of urban traffic measuring based on the sensor network system manufactured by SenSys Network, Inc. [1]. A traffic monitoring sensor network consists of three components, sensor nodes, access points, and relay nodes. Sensor nodes are equipped with magnetometers and low-power radio modules. When mounted to the pavement, it counts the number of vehicles passing by. The traffic counts are collected at access points via multi-hop communications and then sent to the traffic control system. Relay nodes are added between access points and disconnected sensor nodes. We present an analytical method for acquiring the correlation model as an alternative to machine learning based methods described in chapter 2.6. The analytical method is based on origin-destination matrix commonly used in traffic flow analysis.

A directed graph $\overrightarrow{G} = (A, \overrightarrow{E})$ is used to represent the road map, where vertices in A stand for intersections or entrances/exits, and an edge $e = (\overrightarrow{u_1, u_2})$ in $\overrightarrow{E}$ stands for the directional road segment from intersection u_1 to u_2. For simplicity, we only consider the total traffic count of all lanes.

Among all vertices in A, let A_s denote the set of traffic sources and A_d denote the set of traffic destinations. Vehicles enter the road system $\overrightarrow{G}$ through sources and leave through destinations. An example is the entrances/exits of a metropolitan highway system like Fig. 6.1.

We divide the time into N sampling periods, and each period consists of T equal-length cycles. A typical daily traffic report has $N = 24 \times 60$, $T = 60$, and a cycle length of 1 second.

Let $I(e)$ denote all points of interest at road segment e. Let $V = \bigcup_{e \in \overrightarrow{E}} I(e)$ denote the collection of all such points.

Let $Q \subseteq V$ denote the set of points where we deploy sensor nodes and $U = V - Q$ denote the points where we do not. We also refer to Q as measured points and U as unmeasured points. The values at points $i \in U$ are acquired through prediction. The prediction error is defined as the average root mean square (RMS)

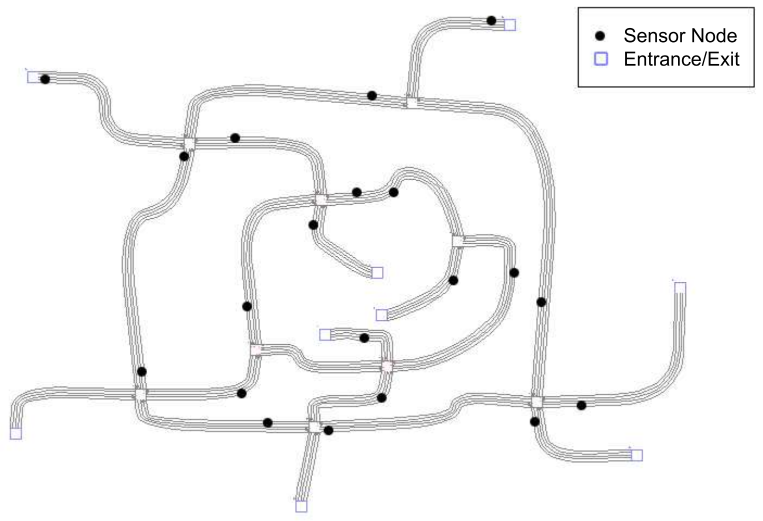


Figure 6.1: A street system with 20 sensor nodes deployed.

error of all N predictions. We have

$$\Delta(Q) = \frac{\sum_{i \in U} \sqrt{\frac{\sum_{t=1 \ldots N} (x^t_{i|X^t_Q} - X^t_i)^2}{N}}}{|U|}, \tag{6.1}$$

where X^t_i is the measured value at point i in sampling period t, and $x^t_{i|X^t_Q}$ is the predicted value at point i given the measured values at all locations in Q in sampling period t. Our objective is to minimize $\Delta(Q)$ with a cost η, formally written as

$$\underset{Q}{\operatorname{argmin}} \, \Delta(Q), \quad s.t. \; |Q| = \eta. \tag{6.2}$$

6.2 Prediction using machine learning based method

The general procedure of constructing a prediction-enabled traffic measuring system is as follows. First, a set of temporary sensors are deployed at all points in V. During the phase of initialization, a central data server collects training data and train the correlation model. Next, the measured points O are chosen and get long-term deployment. The network starts to fully function. As the data server acquires readings from the measured points, it performs prediction using the correlation model and the observations (Eq. 2.3 and Eq. 2.4).

When the environment changes, such as with a road closing or a new road opening, the correlation model needs to be updated. The updating procedure is the same as the construction procedure, which involves a costly initialization phase with a complete temporary sensor placement.

6.3 An analytical method for traffic prediction

Correlation models acquired through machine learning based methods only work well for the same environment in which they are learned. When the environment changes, the pre-learned correlation model needs to be updated. It is not feasible to initiate the learning procedure whenever a change happens. In this section, we derive a theoretical traffic model that serves as a more cost effective alternative.

6.3.1 Traffic model

Recall that we have defined the set of traffic sources A_s and traffic destinations A_d. Let $\theta_{\overrightarrow{l,k}}$ denote the percentage of vehicles originated at source l and ended at k. It is referred to as the destination frequency and is acquired through previous machine learning or other sources of knowledge. Assume the majority of vehicles travel from l to k follow the shortest path. The traffic counts at any points on edge e are the summation of all traffics travelling through e. Given the random variables of all traffic sources $Z_l,\ l \in A_s$, the random variable representing a point at edge e is formally written as

$$\forall x \in I(e),\ x = \sum_{l \in A_s} \Big(\sum_{k \in A_d(l,e)} \theta_{\overrightarrow{l,k}} \Big) \bullet Z_l, \tag{6.3}$$

where $I(e)$ is previously defined as the points of interest at edge e, and $A_d(l,e)$ is a subset of A_d representing the destinations whose shortest pathes from source l contain e. The distribution of Z_l is also acquired through prior knowledge. x follows a Gaussian distribution since the distributions of all Z_l are also Gaussian.

Assuming all sources are mutually exclusive, we have the expected value as

$$E(x) \quad = \quad \sum_{l \in A_s} \lambda(x,l) \bullet E(Z_l) \tag{6.4}$$

where

$$\forall x \in I(e),\ \lambda(x,l) = \sum_{k \in A_d(l,e)} \theta_{\overrightarrow{l,k}}. \tag{6.5}$$

To understand it better, readers may consider $\lambda(x, l)$ as the percentage of vehicles that are originated at source l and travel through point x.

We also derive the covariance as

$$Cov(x, y) = \sum_{l \in A_s} \lambda(x, l)\lambda(y, l) \bullet Var(Z_l). \tag{6.6}$$

The sample mean vector μ is estimated using $E(V)$, and the sample covariance matrix COV is filled using $Cov(V, V)$.

6.3.2 Model Refinement

Through experiments, we verified that this model is accurate in estimating the sample mean. However, its estimation on the covariance matrix carries large errors, since it omits two important factors in any traffic systems, the traffic light and the road length. Traffic lights add more uncertainty to traffic counts and thus result in higher variance than the derived values. The road lengths make the covariances smaller, because travelling from points to points usually takes a non-neglectable amount of time. In the following text, we propose two refinement methods to solve these problems.

1. Variance Correction

For any edge $e \in \overrightarrow{E}$, let $\overrightarrow{C_e}$ denote the set of edges satisfying $\forall f \in \overrightarrow{C_e}, f = \overrightarrow{(u_k, u)} \Leftrightarrow e = \overrightarrow{(u, u_j)}$. In other words, the common end point of all edges in $\overrightarrow{C_e}$ is the starting point of e, or the road segments in $\overrightarrow{C_e}$ have direct flows to the road segment e. Assume e has a probability of $\omega_{\overrightarrow{f,e}}$ to accept a vehicle from road

f at each cycle. The number of vehicles it accepts from f after a given number of cycles follows a Binomial distribution. Recall that we have defined T as the number of cycles in each sampling period. Using the Gaussian approximation for Binomial distribution, the incoming flow from an edge f in $\overrightarrow{C_e}$ results in an expected traffic count of

$$\forall y \in I(e), E_{\overrightarrow{f,y}}(y) = T \bullet \omega_{\overrightarrow{f,e}}. \tag{6.7}$$

For simplicity, assume all $\omega_{\overrightarrow{f,e}}$ carry the same value ω_e for edge e, and flows from all $f \in \overrightarrow{C_e}$ are independent. The expected total traffic count is the summation of every individual expected count at Eq. 6.7, which is

$$\forall y \in I(e), E(y) \;=\; \sum_{f \in \overrightarrow{C_e}} E_{\overrightarrow{f,y}}(y) = |\overrightarrow{C_e}| \bullet T \bullet \omega_e. \tag{6.8}$$

Since we already have a good approximation to the expected values using the previous model, we estimate ω_e as the value of $E(y)/(|\overrightarrow{C_e}| \bullet t)$ with confidence. We further approximate the variances as

$$\forall y \in I(e), Var(y) = |\overrightarrow{C_e}| \bullet T\omega_e(1 - \omega_e) = E(y) \bullet (1 - \frac{E(y)}{|\overrightarrow{C_e}| \bullet T}). \tag{6.9}$$

In other words, the variance at a road segment is estimated using the expected value, the length of a sampling period, and the number of incoming road segments.

2. Covariance Correction

We first consider a simple case of two points x, y on the same edge e, y being the down flow point of x. Their distance is defined using term $D(\overrightarrow{x,y})$. Assume all vehicles travel at a constant speed v and reach y from x in at most two sampling periods. The traffic count at y during period t is given by the summation of two parts. One is the number of vehicles leaving x in period t and arriving at y. Another is the number of vehicles leaving x in period $t-1$ but arriving in period t. The value of traffic count is formally written as

$$X^t = \frac{T - D(\overrightarrow{x,y})/v}{T} \bullet X^t + \frac{D(\overrightarrow{x,y})/v}{T} \bullet X^{t-1}. \tag{6.10}$$

Let x' be the random variable representing the previous sampling period of x. Random variables x, x' follow the same distribution but are assumed mutually exclusive. The covariance of y and x is given by

$$\begin{aligned} Cov(y,x) &= Cov(\frac{T - D(\overrightarrow{x,y})/v}{T} \bullet x + \frac{D(\overrightarrow{x,y})/v}{T} \bullet x', x) \\ &= \frac{T - D(\overrightarrow{x,y})/v}{T} \bullet Cov(x,x). \end{aligned} \tag{6.11}$$

Since $Cov(y,x) = Cov(x,y)$, we omit the arrow in $D(\overrightarrow{x,y})$ and use the absolute distance $D(x,y)$.

The generalization of Eq. 6.11 to any two points is given by

$$Cov(y,x) = \frac{T - D(x,y)/\overline{v_{x,y}}}{T} \bullet Cov'(y,x), \tag{6.12}$$

where $\overline{v_{x,y}}$ is the average speed between x and y, $D(x,y)$ is the length of the shortest path between x and y, and $Cov'(y,x)$ is the covariance computed by Eq. 6.6.

In a real street system, we recommend to use the speed limits to approximate $\overline{v_{x,y}}$.

6.4 Optimal Sensor Placement

Guitton et al. tackle this problem by formulating it as a dominating set problem [29]. The scope of this method is limited because it does not utilize 1-to-n correlations. Instead, we define a heuristic function and propose three algorithms for minimization.

It is not possible to obtain the exact value of the objective function, since computing $\Delta(Q)$ (Eq. 6.1) requires the complete sample set not available in the sensor selection phase. We use the marginal covariance matrix $COV_{U|Q}$ defined by Eq. 2.4 to approximate $\Delta(Q)$. Let function $diag(M)$ denote the diagonal elements of matrix M. We have

$$\Delta(Q) \approx h(Q) = \overline{\sqrt{diag(COV_{U|Q})}}, \tag{6.13}$$

that is, $\Delta(Q)$ is approximately the average of the square roots of the diagonal elements in $COV_{U|Q}$. Function $h(Q)$ is the heuristic function. We use three methods, greedy, m-random trial, and local search to minimize $h(Q)$. As an example, we discuss the algorithm using Eq. 6.2.

1. Greedy Algorithm

Start with an empty set Q. In every iteration, choose a point $b \in V - Q$ such that $h(Q + \{b\})$ is minimal. Terminate the process when reaching the desired number of sensor nodes η.

2. m-Random Trial Algorithm

For each of m trials, randomly generate a placement Q with η sensor nodes. Select the placement with the lowest $h(Q)$ value.

3. Local Search Algorithm

Start with a valid placement achieved by the previous methods. In each iteration, select a neighborhood of new placements that exchange any pair of points in Q and Y. Choose the neighbor resulting in the maximal reduction in $h(Q)$ and use it as the inputting solution for the next iteration. Terminate the process when no more improvement is possible or reaching a given number of iterations.

6.5 Experimental Result

We conduct the experiments in the GLD simulator [71]. It uses block as the unit for distance and assumes all vehicles travel at a constant speed of 2 blocks per cycle. Vehicles are generated at traffic sources. Each traffic source l is associated with a spawning frequency of p_l, which is the probability of generating a vehicle in each cycle. The distribution of traffic counts at the sources in each sampling period (T cycles) is approximately Gaussian. We have

$$\forall l \in A_s,\ Z_l \sim N(Tp_l, Tp_l(1-p_l)). \tag{6.14}$$

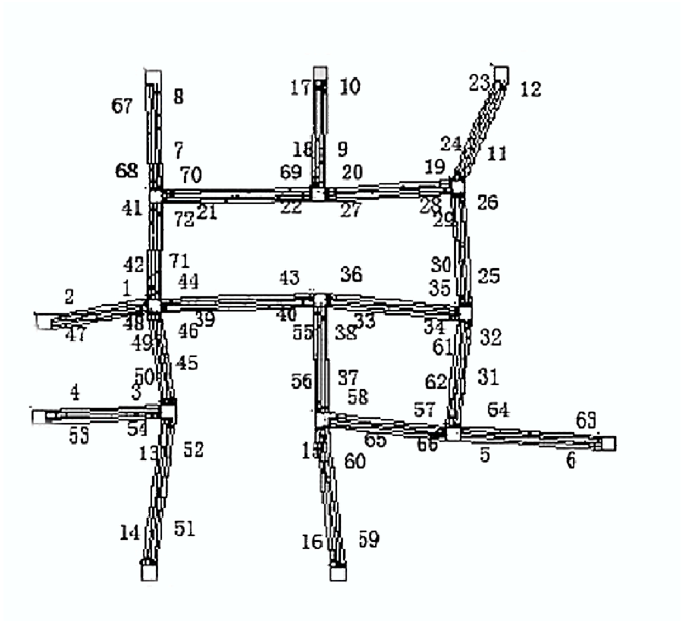


Figure 6.2: A test map with 72 points of interest distributed at each end of a road segment. The numbers in the map show the point id.

During our simulation, we set $T = 50$ and vary the spawning frequencies in $[0.1,\ 0.4]$ at different sources.

Each source is also associated with a set of destination frequencies, which is defined as the percentage of vehicles travelling to each destination. We use them as the value for $q_{\overrightarrow{l,k}}$ in Eq. 6.3 and Eq. 6.5 for deriving the traffic model. For each road segment e, we select its two ends as the points of interest $I(e)$. A sample setup is shown in Fig. 6.2.

We conducted simulations in each map and collect 3000 samples. We randomly choose 70% of the samples as the training set and use the remaining 30% as the test data set.

We first compare the prediction accuracy of the Gaussian model based method

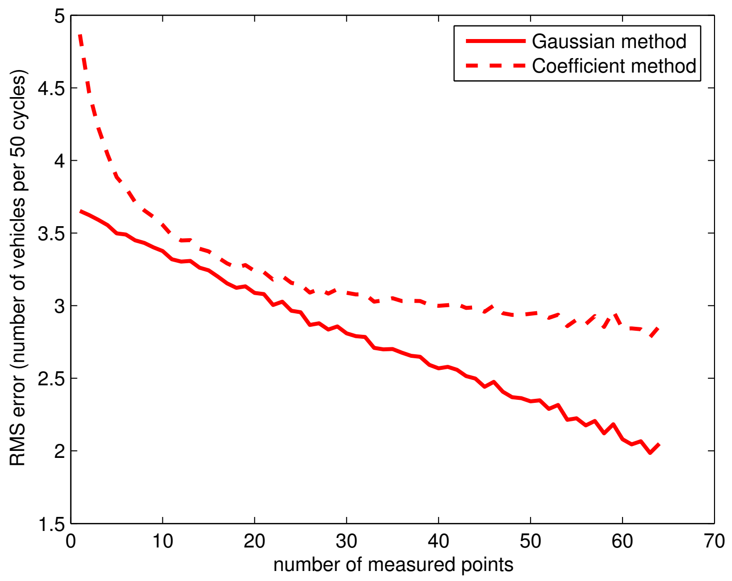


Figure 6.3: The average RMS prediction error of the Gaussian based and coefficient based methods. The average number of vehicles is 15.1 per 50 cycles.

to the coefficient based method introduced in [29]. The models were acquired based on machine learning methods. The experiments were conducted in a road map with 72 points of interest (Fig. 6.2). For each method, we systematically test different number of deployed sensors, i.e., $|Q| = 1...64$. We randomly pick 30 measured points for each number and show the average prediction error on the unmeasured points in Fig. 6.3. In general, the prediction error reduces as we add more measured points. The error is around 25% when we deploy one sensor, and reduces to 13% when we deploy 64 sensors. We have two further observations. 1) When the size of Q is small, some unmeasured points do not have any strongly correlated observations. Adding measured points usually results in a dramatic improvement in the accuracy of coefficient based method. When the size of Q is large, such improvement is getting small since most unmeasured points already carry strong correlations to some points in Q. 2) The Gaussian method outperforms the coefficient based method in all test cases, with a linearly decreasing curve as the size of Q increases. This shows that the average variance of the posterior distribution reduces at an almost constant step as we add more points of observation to Q.

Next, we compare the model acquired through the analytical method to the model trained by the machine learning based method, and study the prediction results of the analytical model. In the same maps, we compare the analytical model to the model acquired through maximal likelihood learning. We first examine them in a simple map with 16 points of interest. The analytical model is a close approximation to the learned model, and achieves predictions with similar level

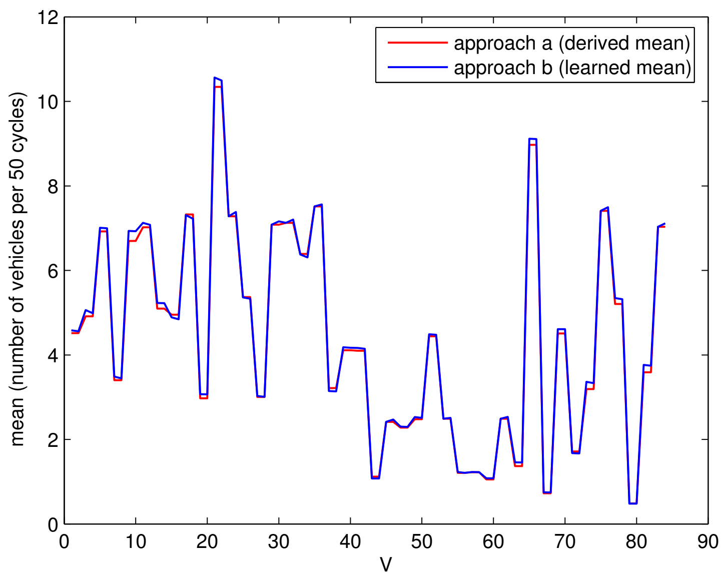


Figure 6.4: The expected values of random variables acquired through the analytical method (*approach a*) and the machine learning based method (*approach b*).

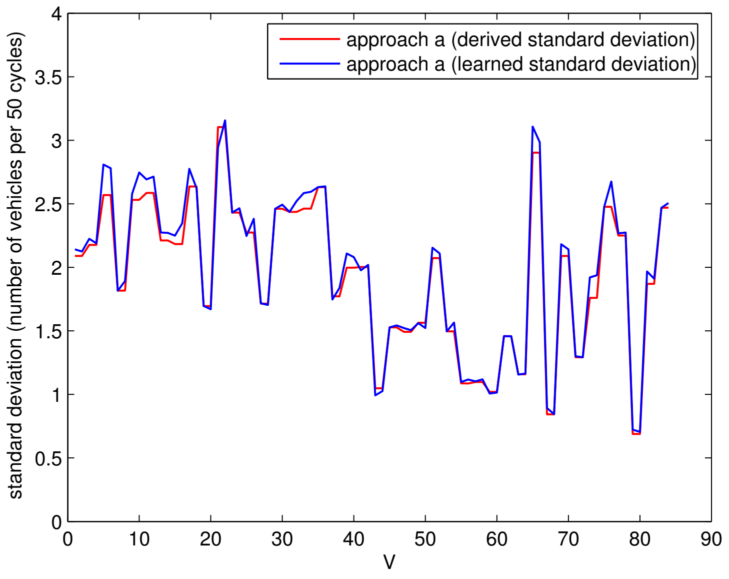


Figure 6.5: The standard deviations acquired through the analytical method (*approach a*) and the machine learning based method (*approach b*).

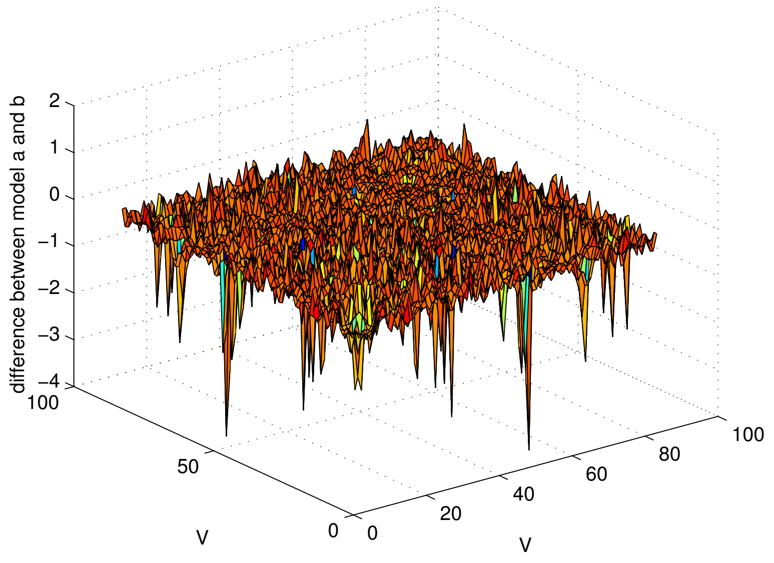


Figure 6.6: Difference of the covariance matrix between the analytical method and the machine learning based method.

of accuracy. We further conduct the comparison in a large map with 84 points of interest (Fig. 1.3). Figure 6.4$\sim$6.6 compare the resulting means, standard deviations, and covariance matrix of the analytical model and the learned model. In general, the estimations are close. Most problems occur in the approximations of covariances with underestimations. They are however not significant threats to the prediction accuracy, because 1) the number of occurrences is small, 2) the largest errors are only around half of the variances, and 3) the underestimated points automatically get lower weights in the prediction formula (Eq. 2.3), reducing the effects of underestimations. Figure 6.7 shows that using the analytical model and the Gaussian based method, we achieve prediction accuracy close to the Gaussian based method with learned models, and higher than the coefficient based method with learned models when the size of Q is small or large.

In the last set of experiments, we compare the greedy, m-random trial and the local search methods for sensor placement. m is set to 100 in the m-random trial method. The results are summarized in figure 6.8. The greedy algorithm outperforms the 100-random trial method, only leaving small spaces for local search algorithm to improve. We have also noticed in the experiments that good placements are usually sparsely distributed (e.g., Fig. 6.1). This is because nearby points usually share high covariances.

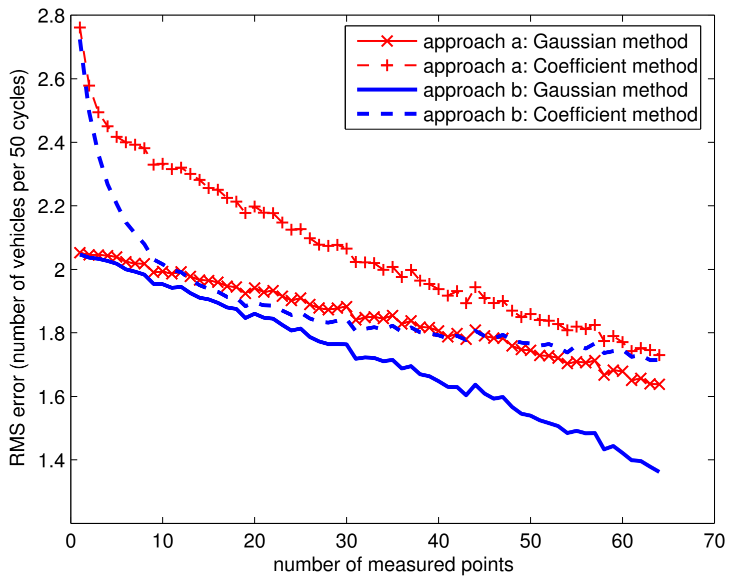


Figure 6.7: The average RMS prediction error using the analytical method (*approach a*) and the machine learning based method (*approach b*).

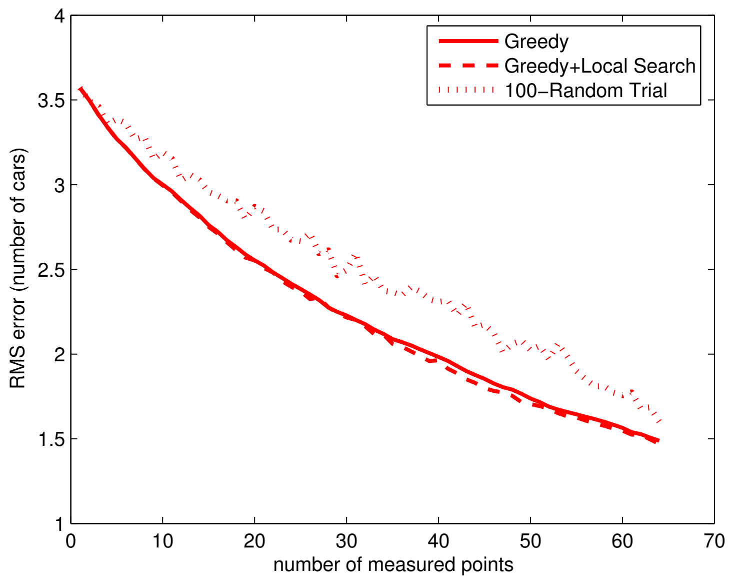


Figure 6.8: The average RMS prediction error using the measured sets selected by greedy, local search, and 100-random trial algorithms.

Chapter 7

Conclusion

Leveraging the spatial/temporal correlation between sensor data in wireless sensor and mobile networks provides new means to protect data integrity, to detect faulty sensor node, to track a subject, and to reduce sensor operating deployment cost. Statistical inference plays an important roles in the design of algorithms in wireless sensor and mobile networks.

We have introduced Cobra as a new energy-efficient means to authenticate sensor data, with energy savings of up to one magnitude lower than existing methods. It also protects sensor networks from aggressive attacks by limiting the maximal damage on data reliability. It is suitable for data collection sensor networks operated in centralized manners. Most sensor network applications require only approximated results and can be well-protected by Cobra. It is important to note that Cobra is not designed for systems that require zero error. As an inference based method Cobra inevitably introduces small error even when the attacks are not aggressive.

Based on a new proposed measurement mutual divergence, we have proposed a method for distributed faulty sensor detection. The method is shown to achieve detection accuracies close to 100% in most test cases. In addition, it does not assume particular type of fault and can be used as a general measure for a number of know fault types. It is also recommended for the systems where the type of fault is unknown. As a fully distributed method, it significantly reduces communication cost since the message exchange only happened between neighboring nodes and detected faulty sensor's readings do not need to be forwarded.

We have proposed SMART, a new system for simultaneously track mobile subjects and acquire logic maps. SMART is shown to outperforms a competing tracking method by 9 times and constructs logic maps up to 89% accurate simultaneously. SMART is also robust against sensing errors and automatically adapts to floorplan changes. We believe SMART could extend the scope of location based services from outdoor to indoor, and reduce the need for tedious manual data collection in map construction.

We also study the statistical inference problems in traffic monitoring sensor networks, where we derives analytical correlation models instead of relying on machine learning based models. We show that the models are close estimations to the learned models and also investigate a number of sensor placement methods. The model derivation is based on original-destination matrix, which is available and constantly updated in major U.S. urban areas. Out proposed method could aid in reducing sensor placement costs and the cost in collecting training data necessary for the machine learning based models.

Bibliography

[1] http://www.sensysnetworks.com/.

[2] Skyhook: a hybrid localization service. http://www.skyhookwireless.com/.

[3] ZigBee Alliance. Zigbee specification. technical report document 053474r06, version 1.0. ZigBee Alliance, Jun. 2005.

[4] M. Azizyan, I. Constandache, and R. R. Choudhury. Surroundsense: Mobile phone localization via ambience fingerprinting. In *Proc. ACM International Conference on Mobile Computing and Networking (MobiCom)*, 2009.

[5] P. Bahl and V. N. Padmanabhan. Radar: an in-building rf-based user location and tracking system. In *Proc. IEEE Conference on Computer Communications (INFOCOM)*, 2000.

[6] T.E. Biedka, J.H. Reed, and B.D. Woerner. Position location using wireless communications on highways of the future. In *Proc. 13th Asilomar Conference on Signals, Systems and Computers*, 1996.

[7] S. Bieniawski and D. H. Wolpert. Adaptive, distributed control of constrained multi-agent systems. *Proc. 3rd International Joint Conf. Autonomous Agents and Multiagent Systems*, 2004.

[8] S. R. Bieniawski. *Distributed optimization and flight control using collectives*. PhD thesis, Stanford University, 2005.

[9] S. R. Bieniawski, I. M. Kroo, and D. H. Wolpert. Discrete, continuous, and constrained optimization using collectives. In *Proc. 10th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference*, 2004.

[10] A. Boulis, S. Ganeriwal, and M. Srivastava. Aggregation in sensor networks: an energy-accuracy tradeoff. In *IEEE SNPA*, 2003.

[11] H. Chan and A. Perrig. Efficient security primitives from a secure aggregation algorithm. In *Proc. ACM Conference on Computer and Communications Security*, 2008.

[12] C.-C. Chang, S. Muftic, and D. J. Nagel. Design of wsn security systems based on energy constraints. In *5th Annual IEEE Consumer Communications and Networking Conference*, 2008.

[13] C.K. Chen and W.A. Gardner. Signal-selective time-difference of arrival estimation for passive location of man-made signal sources in highly corruptive environments. ii. algorithms and performance. *IEEE Transactions on Signal Processing*, 40 (5):1185–1197, 1992.

[14] J. Chen, S. Kher, and A. Somani. Distributed fault detection of wireless sensor networks. In *Proc. 2006 workshop on Dependability issues in wireless ad hoc networks and sensor networks*, 2006.

[15] W. Chen, L. Gao, Z. Chai, Z. Chen, and S. Tu. An intelligent guiding and controlling system for transportation network based on wireless sensor network technology. In *Proc. the Fifth International Conference on Computer and Information Technology*, 2005.

[16] Y. Chen, Y. Chawathe, A. LaMarca, , and J. Krumm. Accuracy characterization for metropolitan-scale wi-fi localization. In *Proc. ACM International Conference on Mobile Systems, Applications, and Services (MobiSys)*, 2005.

[17] S.-Y. Cheung, S. Coleri Ergen, and P. Varaiya. Traffic surveillance with wireless magnetic sensors. In *Proc. 12th Intelligent Transportation System World Congress*, Nov 2005.

[18] B. Clarkson, K. Mase, and A. Pentland. Recognizing user context via wearable sensors. In *IEEE International Symposium on Wearable Computers*, 2000.

[19] J. Considine, F. Li, G. Kollios, and J. Byers. Approximate aggregation techniques for sensor databases. In *Proc. 20th International Conference on Data Engineering*, 2004.

[20] A. Deshpande, C. Guestrin, S. Madden, J. Hellerstein, and W. Hong. Model-driven data acquisition in sensor networks. In *Proc. 30th very large database Conference*, 2004.

[21] M. Ding, D. Chen, K. Xing, and X. Cheng. Localized fault-tolerant event boundary detection in sensor networks. In *Proc. IEEE INFOCOM*, 2005.

[22] R. Elias and A. Elnahas. An accurate indoor localization technique using image matching. *Intelligent Environments*, 2007.

[23] P. Fitzpatrick and C. Kemp. Shoes as a platform for vision. In *IEEE International Symposium on Wearable Computers*, 2003.

[24] J. Gao, L. Guibas, N. Milosavljevic, and J. Hershberger. Sparse data aggregation in sensor networks. In *Proc. 6th international conference on Information processing in sensor networks*, 2007.

[25] W.A. Gardner and C.K. Chen. Signal-selective time-difference-of-arrival estimation for passive location of man-made signal sources in highly corruptive environments. i. theory and method. *IEEE Transactions on Signal Processing*, 40 (5):1168–1184, 1992.

[26] S. Ghosh, A. Freitas, and I. Marshall. Robust autonomous detection of the faulty sensors of a sensor array. In *2nd IEEE International Workshop on Computational Advances in Multi-Sensor Adaptive Processing*, 2007.

[27] W. G. Griswold, P. Shanahan, S. W. Brown, R. Boyer, and M. Ratto. Activecampus - experiments in community-oriented ubiquitous computing. *IEEE Computer*, 2003.

[28] C. Guestrin, A. Krause, and A. P. Singh. Near-optimal sensor placements in gaussian processes. In *Proc. 22nd international conference on Machine learning*, 2005.

[29] A. Guitton, A. Skordylis, and N. Trigoni. Correlation-based data dissemination in traffic monitoring sensor networks. In *Proc. IEEE Wireless Communications and Networking Conference*, 2007.

[30] S. Hay and R. Harle. Bluetooth tracking without discoverability. In *Proc. 4th International Symposium on Location and Context Awareness (LoCA)*, 2009.

[31] L. Hu and D. Evans. Localisation for mobile sensor networks. In *Proc. International Conference on Mobile Computing and Networking (MobiCom)*, 2004.

[32] C. F. Huang, S. Bieniawski, D. H. Wolpert, and C. E. Strauss. A comparative study of probability collectives based multi-agent systems and genetic algorithms. In *Proc. Conf. Genetic and Evolutionary Computation*, 2005.

[33] C. Intanagonwiwat, R. Govindan, D. Estrin, J. Heidenmann, and F. Siva. Directed diffusion for wireless sensor networking. In *ACM/IEEE Transactions on Network*, 2002.

[34] DC Watch Japan. Casio hybrid gps camera prototype for indoor localization. Available at http://dc.watch.impress.co.jp/docs/news/20100109_341568.html, 2010.

[35] C. Karlof, N. Sastry, and D. Wagner. Tinysec: A link layer security architecture for wireless sensor networks. In *the Second ACM Conference on Embedded Networked Sensor Systems (SenSys 2004)*, Nov. 2004.

[36] L. Klingbeil and T. Wark. A wireless sensor network for real-time indoor localization and motion monitoring. In *Proc. 7th International Conference on Information Processing in Sensor Networks*, 2008.

[37] D. Koks. Numerical calculations for passive geolocation scenarios. In *Technical Report DSTO-RR-0000*, 2005.

[38] F. Koushanfar, M. Potkonjak, and S.-V. Vincentelli. On-line fault detection of sensor measurements. In *IEEE Sensors*, 2003.

[39] A. Krause, C. Guestrin, A. Gupta, and J. Kleinberg. Near-optimal sensor placements: maximizing information while minimizing communication cost. In *Proc. 5th international conference on Information processing in sensor networks*, 2006.

[40] A. Krause, E. Horvitz, A. Kansal, and F. Zhao. Community sensing: Foundations and applications. In *Proc. 7th International Conference on Information Processing in Sensor Networks*, Apr. 2008.

[41] B. Krishnamachari and S. Iyengar. Distributed bayesian algorithms for fault-tolerant event region detection in wireless sensor networks. 2004.

[42] P. Krishnan, A. Krishnakumar, W.-H. Ju, C. Mallows, and S. Gamt. A system for lease: location estimation assisted by stationary emitters for indoor rf wireless networks. In *IEEE INFOCOM*, 2004.

[43] B. Kusy, A. Ledeczi, and X. Koutsoukos. Tracking mobile nodes using rf doppler shifts. In *Proc. Conference On Embedded Networked Sensor Systems (SenSys)*, 2007.

[44] Jimmy LaMance, Javier DeSalas, and Jani Jarvinen. Assisted gps: A low-infrastructure approach. GPS World, 2002.

[45] C. F. Lee and D. H. Wolpert. Product Distribution Theory for Control of Multi-Agent Systems. In *Proc. 3rd Int'l Joint Conf. on Autonomous Agents and Multiagent Systems*, 2004.

[46] M. Luk, G. Mezzour, A. Perrig, and V. Gligor. Minisec: a secure sensor network communication architecture. In *Proc. 6th international conference on Information processing in sensor networks*, 2007.

[47] X. Luo, M. Dong, and Y. Huang. On distributed fault-tolerant detection in wireless sensor networks. 2006.

[48] S. Madden, M. J. Franklin, J. M. Hellerstein, and W. Hong. Tag: a tiny aggregation service for ad-hoc sensor networks. In *SIGOPS Oper. Syst. Rev.*, 2002.

[49] S. Madden, M. J. Franklin, J. M. Hellerstein, and W. Hong. The design of an acquisational query processor for sensor networks. In *SIGMOD*, 2003.

[50] A. Manjhi, S. Nath, and P. B. Gibbons. Tributaries and deltas: efficient and robust aggregation in sensor network streams. In *Proc. ACM SIGMOD international conference on Management of data*, 2005.

[51] D. McCrady, L. Doyle, H. Forstrom, T. Dempsey, and M. Martorana. Mobile ranging using low-accuracy clocks. *IEEE Transactions on Microwave Theory and Techniques*, 48 (6):951–958, 2000.

[52] S. Mukhopadhyay, D. Panigrahi, and S. Dey. Data aware, low cost error correction for wireless sensor networks. In *Proceedings of IEEE International conference on Wireless Communications and Networking Conference (WCNC)*, Mar 2004.

[53] S. Nath, P. B. Gibbons, S. Seshan, and Z. Anderson. Synopsis diffusion for robust aggregation in sensor networks. In *Proc. Conference On Embedded Networked Sensor Systems (SenSys)*, 2004.

[54] K. Ni and G. Pottie. Bayesian selection of non-faulty sensors. In *IEEE International Symposium on Information Theory*, Jun. 2007.

[55] A. Ofstad, E. Nicholas, R. Szcodronski, and R. R. Choudhury. Aampl: accelerometer augmented mobile phone localization. In *ACM International Workshop on Mobile Entity Localization and Tracking in GPS-less Environments*, 2008.

[56] T. Palpamas, D. Papadopoulos, V. Kalogeraki, and D. Gunopulos. Distributed deviation detection in sensor networks. In *ACM SIGMOD Record*, Dec. 2003.

[57] A. Perrig, R. Szewczyk, J. D. Tygar, V. Wen, and D. E. Culler. Spins: security protocols for sensor networks. *Wireless Network*, 8(5):521–534, 2002.

[58] P. Prasithsangaree, P. Krishnamurthy, and P. Chrysanthis. On indoor position location with wireless lans. In *Proc. 13th IEEE International Symposium on Personal, Indoor and Mobile Radio Communications*, 2002.

[59] N. B. Priyantha. *The cricket indoor location system.* PhD thesis, 2005.

[60] B. Przydatek, D. Song, and A. Perrig. Sia: secure information aggregation in sensor networks. In *Proc. of the 1st international conference on Embedded networked sensor systems*, 2003.

[61] R. Rajagopal, X. Nguyen, S. Ergen, and P. Varaiya. Distributed online simultaneous fault detection for multiple sensors. In *Proc. International Conference on Information Processing in Sensor Networks*, Apr. 2008.

[62] T. Rappaport, J. Reed, and B. Woerner. Position location using wireless communications on highways of the future. *IEEE Communications Magazine*, 34 (10):33–41, 1996.

[63] N. Ravi, P. Shankar, A. Frankel, A. Elgammal, and L. Iftode. Indoor localization using camera phones. In *IEEE Workshop on Mobile Computing Systems and Applications*, 2006.

[64] S. Ray, W. Lai, and I. Paschalidis. Deployment optimization of sensornet-based stochastic location-detection systems. In *IEEE INFOCOM*, 2005.

[65] T. Roos, P. Myllymaki, and H. Tirri. A statistical modeling approach to location estimation. *IEEE Transactions on Mobile Computing*, 1 (1):59–69, 2002.

[66] M. Rudafshani and S. Datta. Localization in wireless sensor networks. In *Proc. 6th International Conference on Information Processing in Sensor Networks*, 2007.

[67] P. Schaffer and I. Vajda. Cora: correlation-based resilient aggregation in sensor networks. In *Proc. the 10th ACM Symposium on Modeling, analysis, and simulation of wireless and mobile systems*, 2007.

[68] J. Schiff, D. Antonelli, A. G. Dimakis, D. Chu, and M. J. Wainwright. Robust message-passing for statistical inference in sensor networks. In *Proc. the 6th international conference on Information processing in sensor networks*, 2007.

[69] S. Tanachaiwiwat and A. Helmy. Correlation analysis for alleviating effects of inserted data in wireless sensor networks. In *Proc. the The Second Annual International Conference on Mobile and Ubiquitous Systems: Networking and Services*, 2005.

[70] B. Turgut and R. Martin. Restarting particle filters: an approach to improve the performance of dynamic indoor localization. In *Proc. IEEE GlobeCom*, 2009.

[71] M. Wiering, J. Vreeken, J. van Veenen, and A. Koopman. Simulation and optimization of traffic in a city. In *IEEE Intelligent Vehicles Symposium*, Jun 2004.

[72] D. H. Wolpert. Information theory - the bridge connecting bounded rational game theory and statistical physics. In *Complex Engineering Systems*. 2004.

[73] D. H. Wolpert, C.M.Strauss, and D. Rajnarayan. Advances in distributed optimization using probability collectives. *Advances in Complex Systems*, 2006.

[74] Y. Yao and J. Gehrke. Query processing in sensor networks. In *Proc. Conference on Innovative Data Systems Research*, 2003.

[75] M. Youssef, A. Youssef, C. Reiger, A. Shankar, and A. Agrawala. Pinpoint: An asynchronous time-based location determination system. In *Proc. ACM International Conference on Mobile Systems, Applications, and Services (MobiSys)*, 2006.

[76] P. Zhuang, Q. Qi, Y. Shang, and H. Shi. Model-based traffic prediction using sensor networks. In *Proc. 5th IEEE Consumer Communications and Networking Conference*, 2008.

[77] P. Zhuang and Y. Shang. Cobra: Correlation-based content authentication in wireless sensor networks. In *Proc. IEEE GlobeCom*, 2008.

[78] P. Zhuang and Y. Shang. Localize vehicles using wireless traffic sensors. In *2nd International Workshop on Mobile Entity Localization and Tracking in GPS-less Environments*, 2009.

VITA

Peng Zhuang is a PhD student in the department of Computer Science of University of Missouri. He received his Bachelor's degree in Computer Science & Engineering from Beijing Institute of Technology, China in 2001, and his Master's degree in Computer Science from University of Missouri in 2006. As an active figure in the research community, he has published 3 journal papers and 10 conference papers in the fields of sensor networks, artificial intelligence and mobile computing. He received the ”Gilliom Cyber Security Fellowship” in 2007 and the ”Donald Anderson Graduate Research Assistant Award” in 2010.