

國立臺灣大學電機資訊學院電機工程學系

碩士論文

Department of Computer Science and Information Engineering

College of Electrical Engineering and Computer Science

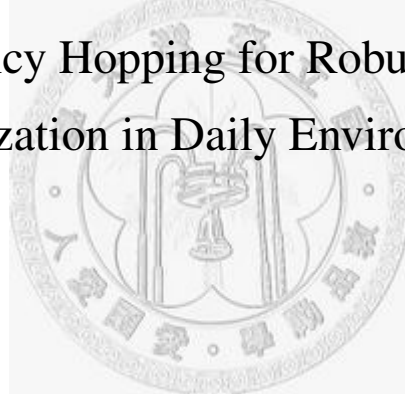
National Taiwan University

Master Thesis

適用於室內定位系統之跳頻機制

Frequency Hopping for Robust Indoor

Localization in Daily Environments



吳意曦

I-Hei Ng

指導教授：黃寶儀 博士

Advisor: Polly Bau-Yi Huang, Ph.D.

中華民國九十七年六月

June, 2008

# 誌謝

沒有以下這些人的幫忙，這篇畢業論文是絕對無法完成的。

首先要感謝我的指導教授黃寶儀老師，這兩年來給了我很多很多的指導，雖然很多時候我不太明白老師的想法，也一直犯了許多重覆的錯誤，但老師還是一直糾正我。老師分析問題的方法及如何把事情一步步從繁化簡不僅對我完成論文，我想對我往後的工作處事都會是很大的幫助。

林宗翰同學是我這篇論文最大的支援者，很多的想法和技術內容其實都是他貢獻的，他清楚的邏輯與系統的程式撰寫和除錯方法對我幫助很多，也感謝他讓出了機會使我在畢業前有幸到美國參加HotEmnets的國際會議，相信以他的實力在哈佛一定能繼續做出許多出色的研究。

感謝黃得源同學在論文的撰寫後期給予我數學模型的指導，還有很多個晚上和我在實驗室討論和做實驗到深夜，她真是個優秀但又樂於助人的好伙伴，祝福她在史丹福能成為sigcomm paper machine。

劉承榮學長一直都是sensor net group最重要的人物，沒有他豐富的硬體經驗，這些的實驗都無法完成，謝謝他在我最困難的時刻和我一起討論和除錯。

也感謝程式高手陳致名同學的幫助，無論什麼樣的程式問題他都能迅速的給予我正確的答案或解決方向。

還有這一年來和我一起奮鬥學弟妹們：陳宥融、陳冠名、黃介廷，實驗室的王江怡萱，實驗室的閃光彈製造者唐彬雲，另一位教授Jeffery，感謝你們給予我許多的鼓勵，也帶給我許多的歡樂的時間。

最後當然要感謝無怨無悔地一直聽我抱怨又每天接送我往返宿舍和系館的陸小奕，一直都很關心我的爸爸、媽媽、大姐、二姐和弟弟。



## 摘要

定位系統的產值在2016年預計將達到兩億七千一百萬美元，並會廣泛地使用在一般的居家環境及商業建築物內，因此近年來定位系統成爲了熱門的研究題目。然而要在日常生活環境中使用無線射頻技術來達到穩定的定位是一個難題。透過由二十四個IEEE 802.15.4無線感測節點組成的感測網路平台來實現一比對信號強度特徵值的定位系統，我們可以有效的分析IEEE 802.11網路封包對室內定位系統準確度的影響。量測的結果顯示當IEEE 802.11的資料量變高時百分之八十的定位誤差會提高141%。定位準確度的降低是由於信標封包與WiFi封包碰撞而遺失。一個簡單的解決方法是延長信標封包的收集時間，但這樣會增加定位的延遲。爲了建構一個強健的實時定位系統，我們提出一跳頻的機制使系統能因應當時無線電波受干擾的程度而跳頻。當系統受到嚴重的干擾時，它能跳至一個新的無線頻帶而避開信標封包被撞掉的情形。實驗結果顯示使用跳頻機制百分之八十的定位誤差在無線網路繁忙時由1.82米降低至1.32米(27%)，在最繁忙的20分鐘內由2.74米降低至1.24米(55%)。

**關鍵字**— 室內定位系統 (Indoor Localization), 信號強度 (Received Signal Strength Indicator), 跳頻 (Frequency Hopping), 干擾 (Interference), 共存 (Coexistence), 隱馬爾可夫模型 (Hidden Markov model)

# Abstract

With an expected market value of \$2.71 billion in 2016, supporting daily use of real-time location systems in households and commercial buildings is an increasingly important subject of study. A growing problem in providing robust location estimations in real time is the use of wireless transmissions in RF frequencies in the daily environments. Having implemented a simple RSSI-signature-based location system on a 24-node IEEE 802.15.4-based sensor network testbed, we are able to analyze the effect of background IEEE 802.11 traffic to the localization error. The measurement results demonstrate that the 80th-percentile of the localization error may increase by 141% when the background 802.11 traffic is high. Such performance degradation is a result of RSSI reading loss as the beacon messages collide with background traffic. A common solution to this problem is to extend the time for beacon message collection. This approach, although effective, adds extra delay before robust estimations can be obtained. Aiming at achieving robust real-time localization in daily environments, we propose a frequency hopping mechanism that enables the system to adapt to the current interference level. When the interference level is high, the system hops to a new channel to avoid the foreseen high loss period. Our experimental results show that

the proposed frequency hopping mechanism can reduce the 80th-percentile localization error from 1.82 to 1.32 meters (27%) in a busy hour and from 2.74 meters to 1.24 meters (55%) in a 20-minute period where the 802.11 traffic rate is at its peak.

**Keywords**— keywords Indoor Localization, Received Signal Strength Indicator, K-Nearest-Neighbor Algorithm, Frequency Hopping, Interference, Coexistence, Hidden Markov model

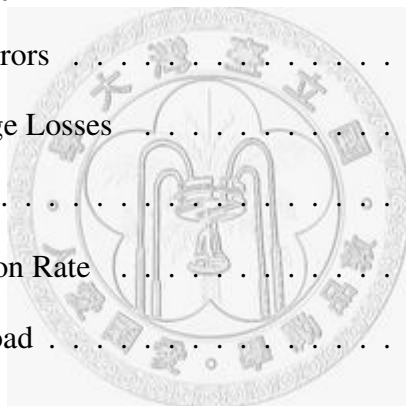


# Contents

誌謝	ii
摘要	iv
Abstract	v
List of Figures	x
List of Tables	xii
<b>Chapter 1 Introduction</b>	<b>1</b>
<b>Chapter 2 Localization System</b>	<b>6</b>
2.1 Beacon . . . . .	7
2.2 Training Phase . . . . .	7
2.3 Tracking Phase and the KNN Estimator . . . . .	7
<b>Chapter 3 Testbed</b>	<b>9</b>

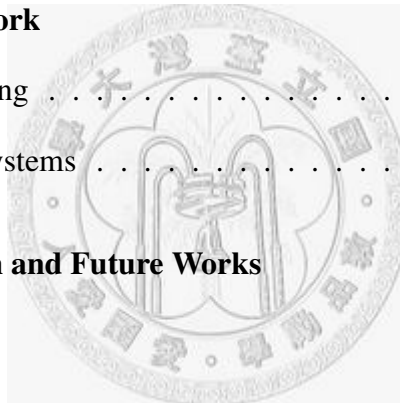


3.1	USB Connectivity . . . . .	10
3.2	Power Supply . . . . .	12
<b>Chapter 4 Measurement Methodology</b>		<b>13</b>
4.1	Location of Measurement . . . . .	14
4.2	Data Logging . . . . .	14
4.3	Loss in Beacon Message Logging . . . . .	16
4.4	Beacon Message Synchronization . . . . .	17
<b>Chapter 5 Trace Analysis</b>		<b>19</b>
5.1	Localization Errors . . . . .	19
5.2	Beacon Message Losses . . . . .	22
5.3	RSSI Values . . . . .	23
5.4	Packet Reception Rate . . . . .	24
5.5	WiFi Traffic Load . . . . .	25
<b>Chapter 6 Frequency Hopping Mechanism</b>		<b>26</b>
6.1	Beacon Node . . . . .	26
6.2	Diagnostic Test . . . . .	27
6.2.1	State Modeling . . . . .	29
6.2.2	State Estimation . . . . .	31
6.3	Signaling Protocol . . . . .	33
<b>Chapter 7 Evaluation</b>		<b>36</b>





7.1	Trace Synthesis . . . . .	36
7.2	Hopping Time . . . . .	38
7.3	Diagnostic Test . . . . .	38
7.3.1	PRR Thresholds . . . . .	39
7.3.2	Hidden Markov Model . . . . .	41
7.4	Localization Error with Frequency Hopping . . . . .	44
7.4.1	Busy vs. Quiet Hour . . . . .	45
7.4.2	Sliding Window Size . . . . .	46
<b>Chapter 8</b>	<b>Related Work</b>	<b>48</b>
8.1	Channel Hopping . . . . .	48
8.2	Localization Systems . . . . .	49
<b>Chapter 9</b>	<b>Conclusion and Future Works</b>	<b>51</b>
	<b>Bibliography</b>	<b>53</b>



# List of Figures

3.1	Testbed Layout. The survey area is denoted as blue. . . . .	10
3.2	(a) chained connection (b) cut the power connection (VCC) between USB extender and USB hub . . . . .	11
5.1	Distribution of Localization Errors . . . . .	20
5.2	Impact of Beacon Packet Loss . . . . .	21
5.3	Effect of Background Traffic to RSSI Readings . . . . .	23
5.4	PRR vs Localization Errors . . . . .	24
5.5	WiFi Traffic Load . . . . .	25
6.1	Illustration of Two-State HMM . . . . .	33
7.1	One-Hour WiFi Data Rate . . . . .	37
7.2	Hopping Time . . . . .	39
7.3	Localization Error with Different PRR Thresholds . . . . .	40
7.4	Hopping Decision of PRR-Thresholding. Red dots: Hop. Green dots: No-Hop. . . . .	41

7.5	State Diagram of HMM . . . . .	42
7.6	State Prediction of HMM. Red dots: Hop. Green dots: No-Hop. . . . .	43
7.7	Localization Accuracy in Busy/Quiet Hour . . . . .	46
7.8	Impact of Window Size on Localization Accuracy . . . . .	47



# List of Tables

3.1 Details of the Data Sets . . . . . 11



# Chapter 1

## Introduction

The market for real-time location systems for assets and personnel tracking is expected to reach \$1.26 billion by 2011 [3], and \$2.71 billion in 2016 [1]. For widespread adoption and everyday use of real-time location systems in households and commercial buildings, the systems must be able to provide accurate and stable location estimations with little delay.

Most indoor localization systems employ an RSSI-signature-based approach which exploits temporal stability in the received signal strength indication (RSSI) from a set of pre-deployed beacons at identified locations, which is referred to as the RSSI signature. When a target carrying a receiving tag enters the space, the received RSSI values are compared to the RSSI signatures. The corresponding location of the closest RSSI signature identifies the location of the target. Methods of ensuring robust mapping between the measured RSSI values and the pre-recorded RSSI signature have been studied intensively in recent years [11] [14] [32] . Such methods are needed to

minimize localization errors induced by unstable RSSI values.

An often overlooked problem is the increasing use of wireless transmission of RF frequencies in the everyday environment. Bluetooth (IEEE 802.15.3), WiFi (IEEE 802.11) and Zigbee (IEEE 802.15.4) all operate in the 2.4x GHz frequency band. The stability and availability of RSSI information for WiFi- or Zigbee-based localization systems may vary depending on interference from other WiFi, Bluetooth, and Zigbee sources.

After conducting a systematic set of experiments on a Zigbee-based sensor network testbed, we find that the 80th-percentile error of a simple RSSI-signature-based location system may increase from 1.6 to 3.9 meters when the amount of background WiFi traffic increases from 68 to 2835 kbps. Having measured also the amount of WiFi traffic in our department building, we observe that there is a significant amount of time that the localization accuracy may suffer from the bursts of background noise.

In a detailed analysis, we discover that the degradation in localization accuracy is mainly contributed by loss of beacon messages, rather than the variance of RSSI values. This agrees with previous studies that discovered variance in RSSI values is mainly due to the multi-path effect [2] [40]. Background traffic does not add to the multi-path effect, rather causing the beacon messages to drop. A common solution to this problem is to extend the time for beacon message collection. This approach, although effective, adds extra delay before robust estimations can be obtained.

In wireless communication, frequency hopping [4] is a widely used technique to avoid sending additional data on channels that are already busy. The proposed frequency hopping mechanism for RSSI-based indoor localization exploits the same con-

cept.

Each beacon node runs a diagnostic test periodically to determine whether the operating channel is experiencing beacon message losses. When the level of losses reaches a preset limit, the beacon node issues a hopping signal to inform all nodes in the network, including the receiving tag, to hop to the next channel. The diagnostic test is essential for the efficiency of the frequency hopping location system. If the test is inadequately sensitive, the system suffers from the beacon message loss problem and provides unstable location estimations. If the test is overly-sensitive, the system may hop unnecessarily, which would incur even greater beacon message losses. Note that the receiving tag also suffers from the beacon message loss problem when beacon nodes are hopping to the different channels.

Given the time dependency observed in long-term WiFi traces and the correlation between WiFi traffic and the beacon message reception rate, whether or not the system tends to produce high error can be obtained via hidden Markov model (HMM) by observing beacon message reception rate. The resulting HMM and the Forward algorithm to infer the ‘Hop’ or ‘No-Hop’ states are reasonably tractable for a limited sensor node platform and can be used on beacon nodes for diagnostic testings.

The experimental results show that the transition time required for the network to hop to a new frequency under realistic losses is approximately 8 milliseconds for the sensor network testbed. The 80th-percentile error of the location system is reduced from 1.82 to 1.32 meters (27%) in a busy hour and from 2.74 meters to 1.24 meters (55%) in the 20-minute period where the WiFi traffic rate is at its peak. The 80th percentile and 50th percentile localization errors are kept low to approximately 1.3

meters and 0.6 meters, even when the environment is experiencing heavy interference. Although using a longer beacon message collection time may mitigate the localization error due to beacon message loss, the proposed frequency hopping mechanism pushes the envelope further, and enables accurate and stable indoor localization with minimum delay.

Although the frequency hopping mechanism is shown effective for an RSSI-signature-based indoor localization system, the mechanism can potentially be applied to any localization systems that sends wireless beacon messages for location estimations, regardless whether the system is for indoor or outdoor localization, whether the system is RSSI-signature-based or RSSI-ranging-based. This study makes the following four contributions:

- This is, to our knowledge, the first proposed use of frequency hopping for localization.
- The unique architecture of the proposed sensor network testbed enables low cost co-collection of data traces at the beacon nodes and the receiving tags.
- The systematic measurement study provides an understanding on the effect of background traffic to indoor RSSI-signature-based location systems.
- The simple frequency hopping mechanism for indoor RSSI-signature-based location systems is shown effective.

The rest of the paper is organized as follows. First, the RSSI-signature-based location



system and the implemented testbed for the study are detailed. Sections 4 and 5 describe the measurement methodology and trace analysis of the effect of background WiFi traffic on the location system. The rationale and the design of the frequency hopping mechanism are examined in Section 6. Finally, Section 7 reports the experimental results validating the efficiency of the proposed frequency hopping mechanism.



## Chapter 2

# Localization System

This study implements an RSSI-signature-based localization system. The underlying concept of this solution is to exploit the mapping between a tag's location and RSSI values of packets received from pre-deployed beacons. The RSSI set is referred to as the RSSI signature or vector. These systems typically operate in two phases, training and tracking phases. In the training phase, the area is surveyed to construct the reference RSSI signature per sampled location. The collective set of RSSI signatures obtained at various locations is referred to as the radio map.

Using the radio map, the system compares the collected RSSI vector to the reference RSSI signatures in the tracking phase to identify the closest possible location. The system employs the k-nearest-neighbor (KNN) method for location inference. The k sample locations with RSSI signatures closest to the collected RSSI vector are selected. The KNN estimator then outputs a location by averaging the coordinates of the top k locations weighted by the distances between the RSSI vector and the signature.

## 2.1 Beacon

The beacons periodically transmit short packets containing the beacon ID. The packet sending interval is set to 200ms. The radio transmission power is set to -7dBm. Thus, the tag can detect nine or ten beacons at every location. Because the beacon packets are the basis for the RSSI readings, successful delivery of the packets is critical to the performance of the localization system. To avoid packet collisions among the beacons, the DESYNC [9] protocol is implemented. The protocol ensures that neighboring beacons have different sending time to avoid collisions.

## 2.2 Training Phase

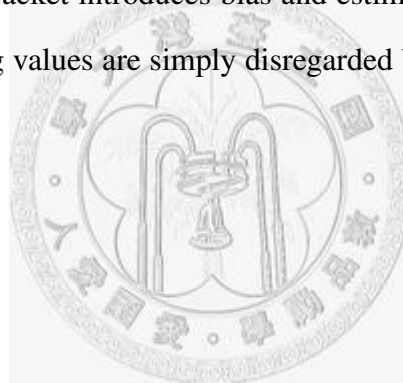
In the training phase, the RSSI signature map is constructed. The survey area is divided into grids, approximately 30cm apart, which is about the distance of one step. During the survey phase, a receiving tag is connected to a portable PC held by the user, who then walks along the corridor. The user must wait at each grid for 8 seconds until the beacon packets are received. After collecting forty RSSI vectors, the received RSSI vectors are averaged to generate a single RSSI signature vector for each sampled location.

## 2.3 Tracking Phase and the KNN Estimator

In the tracking phase, the receiving tag collects the beacon packets for 220ms and sends the RSSI vector back to the localization system. The system then compares

the received RSSI vector with the signature map to find the closest possible location. The KNN method is then used to find the k locations with the closest signature and computed the weighted average of the k location. In this study, the value of k was set to 3.

The signature distance employed is the *normalized Euclidean distance*. Restated, the Euclidean distance between two RSSI vectors is further divided by the number of beacons with significant values in the RSSI vector. This step is necessary because beacon packet loss may be due to geographic distance as well as signal instability or collisions. The semantics of packet loss are ambiguous. Simply using the lowest RSSI value for a lost beacon packet introduces bias and estimation error. To avoid the ambiguity and bias, missing values are simply disregarded by normalizing the Euclidean distance.



## Chapter 3

### Testbed

The testbed served as the platform for measurement study. Twenty-four beacon nodes were deployed on the 6th floor of a department building at this university. The beacon nodes are telos-like modules [26] equipped with TI MSP430 microcontrollers and CC2420 802.15.4 radio. The software is implemented on TinyOS, and the default MAC, a CSMA/CA-like mechanism, is on for all beacon packet transmissions.

Figure 3.1 shows the floor plan. The smaller rooms, numbered 611 to 629, are faculty offices and the remaining are graduate assistant laboratories. The twenty-four Telos-like beacon nodes are small boxes distributed evenly along the corridor. To simplify testbed debugging, every beacon node was connected via USB to one of the two testbed PCs. The PCs were installed in room 621 and room 613, and each was connected to twelve beacon nodes. The PCs served as gateways to allow easy code upgrades and data logging via USB. MoteLab [37] shares the same wired setting, but this testbed uses lower cost off-the-shelf components for long distance USB connec-

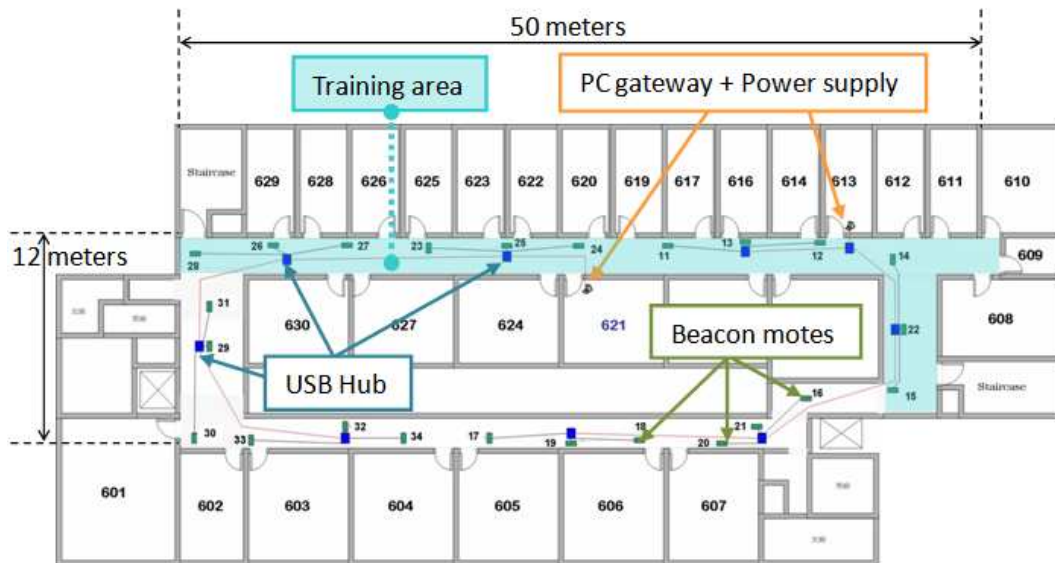


Figure 3.1: Testbed Layout. The survey area is denoted as blue.

tivity. The testbed was designed to use AC power battery replacement and to enable long-term measurements.

### 3.1 USB Connectivity

The effective transmission distance of the standard USB interface was about 5 meters. For nodes more than 5 meters apart, a USB extender [18] was used. The USB extender is an off-the-shelf product that extends the effective transmission distance by up to 45 meters. A local unit on the extender modulates the USB input to signals that transmit on any standard CAT5e network cable. At the other end of the CAT5e network cable is the remote unit which demodulates the signal back to the USB format.

To avoid deploying numerous lengthy wires throughout the building, the chaining

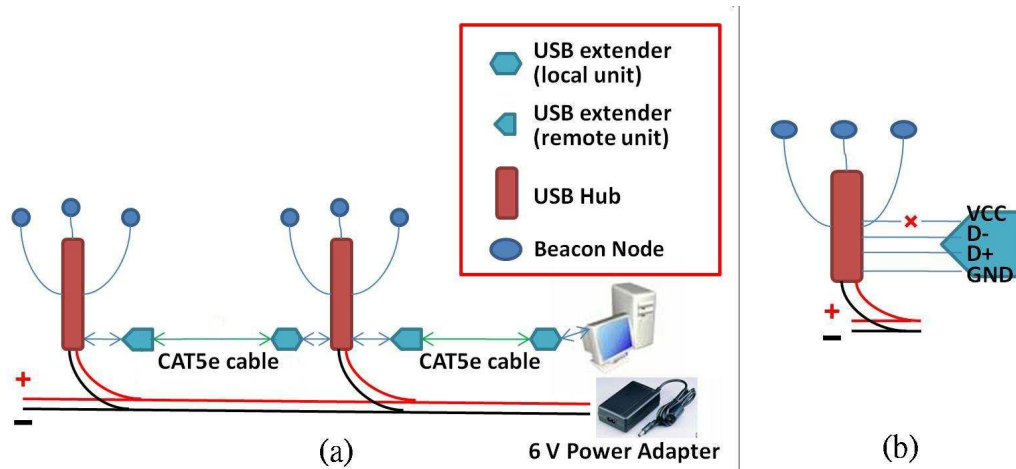


Figure 3.2: (a) chained connection (b) cut the power connection (VCC) between USB extender and USB hub

scheme in Figure 3.2(a) was used to connect nearby beacon nodes. At the beginning of the chain, a 4-port USB hub was connected to the gateway via a USB extender. Three beacon nodes were directly connected to this USB hub, and the next USB hub in the chain was connected via another USB extender. This chain was continued until it reached the maximum range of the USB extender. That is, the distance between the last USB hub and the PC did not exceed forty-five meters. This limitation is also why two PCs were required to cover the entire deployment area in the testbed.

Table 3.1: Details of the Data Sets

Test Case (WiFi Data Rate)	WiFi Log (pkt/MB)	Beacon (pkt/MB)	Beacon Length Error (pkt/Bytes)
68 kbps	48789/4.96	530483/11.13	21/462
264 kbps	74257/19.34	519557/10.90	16/352
1308 kbps	211008/95.80	455629/9.56	29/638
1705 kbps	247684/124.88	445589/9.35	28/616
2835 kbps	450176/207.62	368411/7.73	25/550

## 3.2 Power Supply

Powering the beacon nodes and the USB devices was a challenging problem. The peak current consumption of the beacon node, the USB hub and the USB extender was approximately 60mA, 5mA, and 20mA, respectively. Therefore, a single chain would consume more than 800mA current. Sourcing power from the PC USB port was not feasible due to the high current requirement.

Instead, the USB hubs for the system were externally powered. As Figure 3.2(a) shows, each USB hub on the chain was connected to an AC power adapter providing maximum of 3 amperes of currents. Due to the excessive length of the beacon chains deployed and the large current consumption, the voltage at the last USB hub would have dropped by as much as 1 volt. Hence, although the standard supply voltage for USB is 5 volts, a 6 volt power adapter was used for the external power source to compensate for the voltage drop. Every device in the testbed was tested to ensure it could sustain the 6 volt power supply. The final step is to cut the power connection pin (VCC) between the USB hub and the USB extender remote unit (Fig. 3.2(b)), to ensure that the USB hub was drawing power solely from the external power source.



# Chapter 4

## Measurement Methodology

The RSSI-based localization system is vulnerable to environmental noises. In a typical office or campus environment, background noise can be from WiFi, Bluetooth, a 2.4 GHz cordless telephone, a microwave-oven or other RF devices operating on a 2.4 GHz ISM band. Among these, WiFi traffic produces the most interference. To determine the effect of WiFi noise on localization accuracy and the efficiency of the proposed frequency hopping mechanism, WiFi traffic was generated at different levels. As the WiFi traffic was transmitted at different levels, the following data were collected: (1) background WiFi traffic, (2) beacon messages received at the receiving tag and (3) beacon messages received at other beacons.

## 4.1 Location of Measurement

Because the generated WiFi traffic would be traveling in the space between the source and the access point (AP), the wireless LAN for generating traffic and the location of taking the beacon measurements were carefully selected. A survey of the 6th floor of the department building revealed more than ten APs. Six APs located on the ceiling of a corridor had been installed by the university to provide general wireless Internet access for staff and students. Others were deployed by individual laboratories and had restricted access. One of the generally accessible AP on the ceiling of a corridor was selected for testing. The localization testbed was set to operate on the channel that overlaps with the WiFi channel used by the AP. The receiving tag was positioned close to the AP.

## 4.2 Data Logging

To generate WiFi traffic at different levels, a laptop PC was connected to the internet via the selected AP and a large file was downloaded from an FTP server using FlashFXP, an FTP client that allows the user to set the upload/download speed limit.

Another laptop PC near the WiFi source was used as a sniffer to log all WiFi traffic in the channel and to ensure no unexpected extra traffic occurred. Dumpcap [10], a Linux packet header capturing tool built on the pcap library, was used to log the packets. The WiFi log, referred to as 'WiFi' data, was used to measure the background noise. The experiments were conducted in midnight during weekends. Only a very small amount of traffic other than the generated one was observed by the sniffer. This

ensured that the interference patterns observed were coming from the same AP. In the five sets of experiments, the average background WiFi traffic rates were 68, 264, 1308, 1705, and 2835 kbps. The traffic rate reported here is not exactly the same as the traffic in the channel because the sniffer cannot capture corrupted packets. Also, no kernel loss was reported by the packet capturing tool.

The receiving tag was connected to another laptop PC through the USB interface. Each rate-limited file transfer session was slightly longer than 10 minutes. During that period, the receiving tag transferred all beacon messages received through the USB interface to the laptop PC. Similarly, all beacon nodes in the testbed passed the beacon messages received through the USB interface to the gateway PCs. Table 3.1 provides detailed information about the data sets.

The RSSI values in the beacon messages collected at the receiving tag were used to infer the localization errors. From the beacon messages collected within a beacon cycle (0.2 seconds), the location of the receiving tag was estimated using the mechanism described in Section 2. Each run produced 3000 location estimations. The beacon messages collected at the neighboring beacons were used to calculate the beacon packet reception rate (PRR). The beacon packet reception rate was calculated by zooming into the beacon link near the AP where the generated background traffic emerged. Taking a sliding window of 50, the percentage of beacon packets received in the past 50 cycles was calculated.

The WiFi and RSSI data sets were later used to study the effect of WiFi background traffic to the localization error. The RSSI and PRR data sets were used to observe the correlation between the beacon message reception rate at the neighboring beacons

and the localization error to determine the optimal design of the frequency hopping mechanism.

### 4.3 Loss in Beacon Message Logging

Software running on the sensor nodes as well as the PCs collecting data through the serial port was prone to errors. During the course of the experiments, several errors in the data sets were identified. Most were software bugs and were quickly corrected. The remaining errors were caused by hardware and communication problems. To ensure that traces were not contaminated by software bugs and to accurately assess the quality of the traces, three error checks were implemented to identify hardware and communication problems.

(1) Message length check. Every beacon message generated had a fixed and identical length. For unknown reasons, the CC2420 radio stack sometimes received a valid packet with an altered packet length in high contention. This problem was also noted in the TinyOS CC2420 radio module file, but no fix is currently available. The incorrect length field could be longer or shorter than normal. If the length field reported a longer value, the packet payload would still be correct but with garbage bytes appended for the extra length. However, the RSSI reading would be invalid, and would usually show 0xFF. This problem, although not critical to other uses of sensor packets, is problematic for RSSI-based localization systems. Simply recording the RSSI value without checking produces erroneous location estimations. To correct this problem, the number of bytes in the packet were verified and packets with incorrect packet length were

dropped.

(2) Serial error check. To capture bit errors during serial transmission, a 16-bit CRC checksum value was appended to each packet logged. Packets failing the checksum were discarded by the serial listener. In addition to checksum, a serial sequence number was also appended to each logged packet to check for possible serial loss. The serial listener determined the amount of packet loss from the sequence number. Throughout the experiments, no checksum failure or packet loss was reported. This also showed that wiring the sensor nodes to a central PC was effective for the measurement study.

## 4.4 Beacon Message Synchronization

The beacon sequence number in the beacon messages was used to synchronize the beacon traces. Upon receiving the beacon messages, the other beacon nodes and the receiving tag time stamped the messages using their local clocks. Assuming that the time required for the beacon messages to travel one hop to the receiving tag was the same as that required to travel to the neighboring beacon, the traces were synchronized by simplified Jigsaw approach [7].

More specifically, the local clock of the receiving tag was used as the global clock. Let  $t_m$  represent the timestamp of the reference packet received at the mobile tag and let  $t_k$  denote the timestamp of the reference packet received at the  $k_{th}$  beacon nodes. The local clock of the  $k_{th}$  beacon node would then adjusted by adding the time offset  $t_m - t_k$ . Since the testbed was a multi-hop network, no reference packet could be received by any beacons in the network. A queue of reference packets was implemented

to transitively synchronize other beacon nodes not receiving the previous reference packets. The first packet received by the mobile tag was chosen as the first reference packet. Once a beacon was synchronized, the next packet it received/sent was added to the queue. Elements in the queue were popped out sequentially until all beacons were synchronized.



# Chapter 5

## Trace Analysis

We analyze the five sets of traces collected from the localization testbed with different levels of WiFi traffic in the background.

### 5.1 Localization Errors

Figure 5.1 depicts the cumulative distribution function (CDF) of the localization errors. The localization accuracy was pretty good with 50th percentile error 53cm. We believe such good accuracy comes from the following reasons. First, DESYNC was applied on beacons. Collisions were thus reduced, and the receiving tag could receive sufficient RSSI readings to give accurate location estimation. Second, the survey and the test conditions, e.g. antenna orientation and the way the receiving tag was wore, were held the same throughout the experiments. The beacon density of the system was also high, with a beacon placed every five meters.

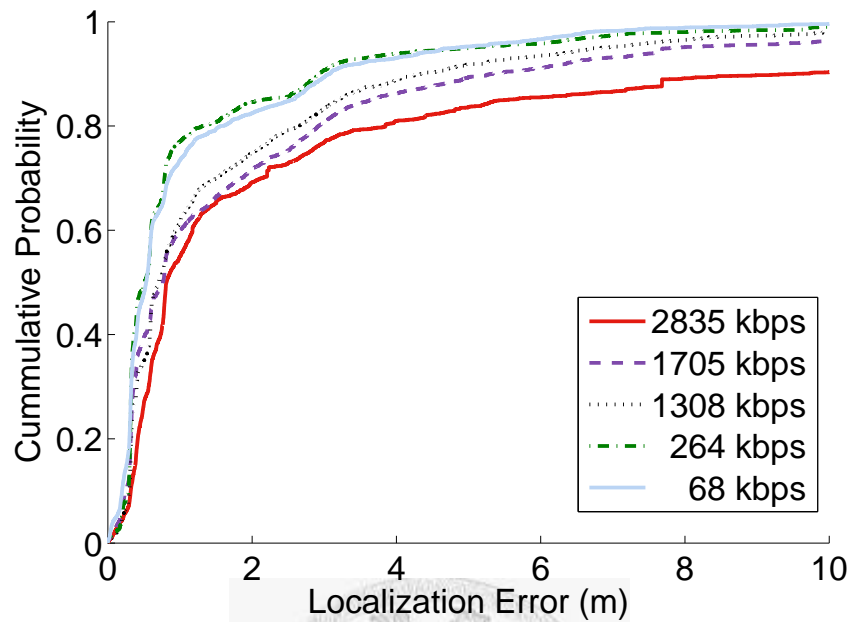
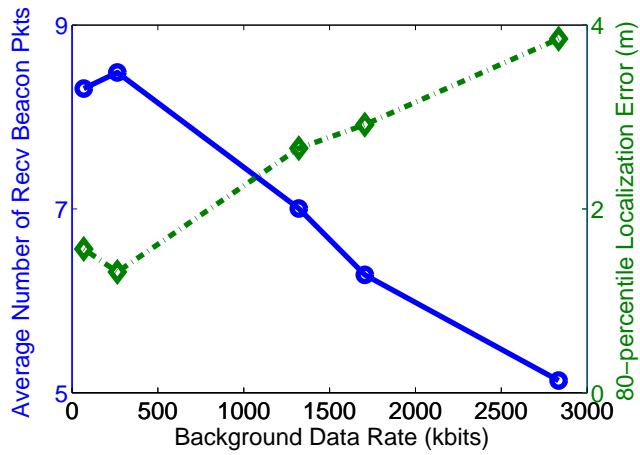


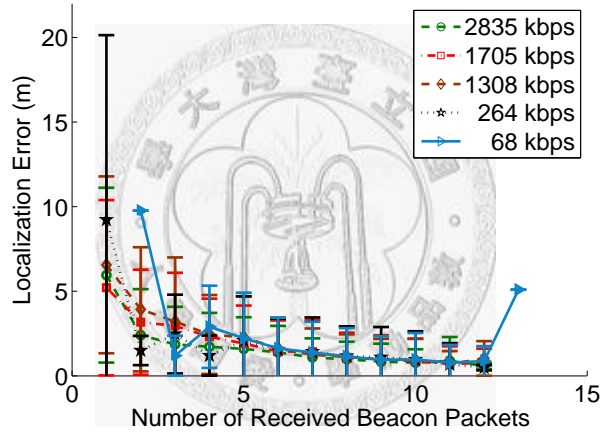
Figure 5.1: Distribution of Localization Errors

The test results showed that the localization errors were influenced by the background WiFi traffic. In the 50th percentile, the errors increased from 53cm to 81cm (53% increase) as the background WiFi traffic increased. The increase in the 80th percentile error from 160cm to 385cm (141% increase) was particularly large. This indicated that, as background traffic increases, the localization error and variance also increase. In cases of heavy background traffic, all beacon messages may be corrupted in some cycles. The localization error was set to a pre-defined maximum for analysis of these cases. In practice, the system can predict or simply report the location obtained in the previous cycle.

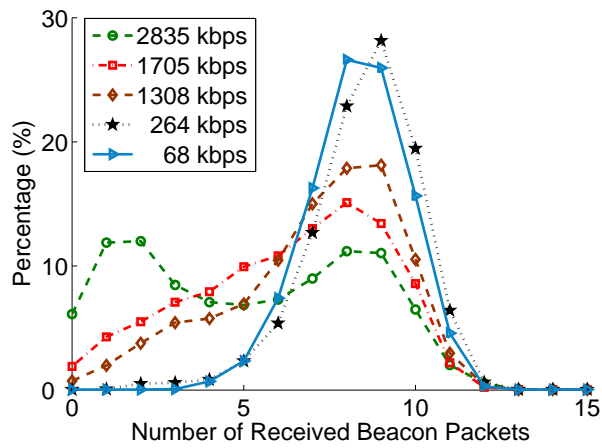




(a) Correlation of Background Traffic Rates and Localization Errors



(b) Effect of Beacon Message Loss



(c) Distribution of the Number of Beacon Messages

Figure 5.2: Impact of Beacon Packet Loss

## 5.2 Beacon Message Losses

To understand how the background traffic impacts localization accuracy, we first look into the loss of beacon messages. In Figure 5.2(a), the left x-axis shows the average number of beacon message received during each 220-ms interval, and the right x-axis plots the 80-percentile localization error. It can be seen that the average number of beacon message received goes down in high background traffic rate and shows a strong correlation with the increasing localization error.

To further clarify the impact of beacon message loss, receiving cycles were classified by the number of beacon messages received for each trace in Fig. 5.2(b). The corresponding average localization error and the variance shown in the figure indicate that fewer received beacon messages increase localization error and variation. A location estimation based on only one or two beacon RSSI readings would be very imprecise. Average errors would be as high as 9 meters since several sample signatures share similar RSSI readings for a single beacon. The insufficient information received due to the fewer beacon messages would cause larger localization errors.

Figure 5.2(c) shows the probability of the tag receiving a specific number of beacon messages for different background traffic rates. The distribution of the number of received beacon messages indicates that high background traffic increase the number of cycles in which the receiving tag observes only a small number of beacon messages. In fact, the background traffic interferes with the delivery of beacon messages and corrupts them. Thus, the overall number as well as variance in localization errors worsens when background traffic is high.

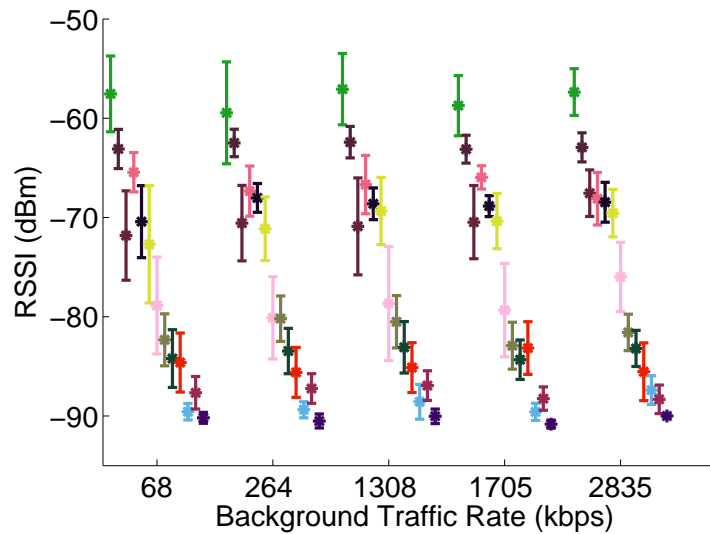


Figure 5.3: Effect of Background Traffic to RSSI Readings

### 5.3 RSSI Values

Note that in Fig. 5.2(b), if the receiving tag manages to receive sufficient beacon messages, the localization errors are all similar for different background traffic rates. This suggests that produces less distortion of RSSI readings. Figure 5.3 shows the average RSSI readings and the standard deviation for each trace. The RSSI values from different beacons are slid slightly for clarity and ease of comparison. The deviation of RSSI readings again reveals no a clear trend as background traffic increases. Generally, RSSI variance causes some localization error. However, background WiFi traffic apparently has no significant effect on RSSI variation. As Fig. 5.1 shows, the overall increase in the magnitude of errors is mainly due to the beacon message losses caused by background traffic.

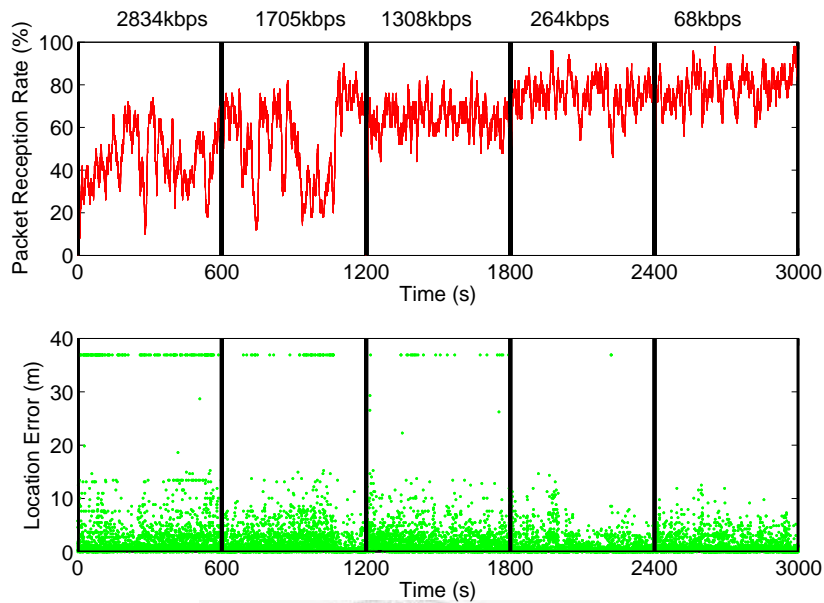


Figure 5.4: PRR vs Localization Errors

## 5.4 Packet Reception Rate

Figure 5.4 shows the beacon packet reception rate of a specific link located near the source AP and the corresponding localization errors. The figure is plotted by concatenating the five 10-minute traces at different background traffic rates. When the level of background traffic increases, the localization error increases and the packet reception rate decreases. The effect of beacon message loss at the receiving tag is apparently observable from the neighboring beacons. The implication is that each beacon node may track the packet reception rates from neighboring beacons. The packet reception rate is likely high when the channel is quiet. The packet reception rate is likely low when there is background noise.

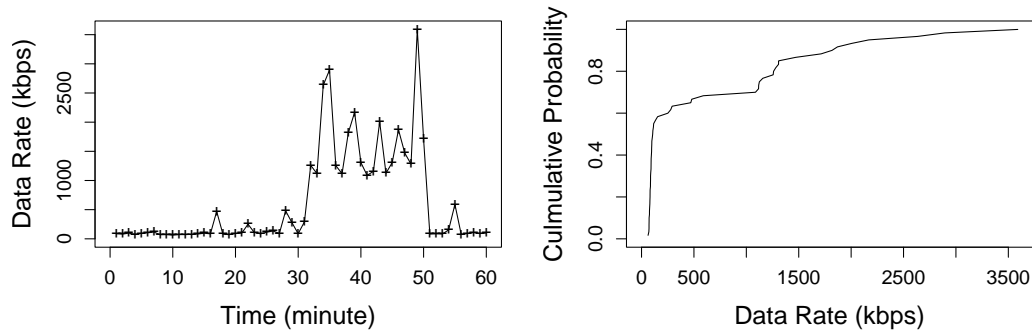


Figure 5.5: WiFi Traffic Load

## 5.5 WiFi Traffic Load

Figure 5.5 shows WiFi data rate during a busy working hour at this department building. The data rate exceeds 1 Mbps for substantial time periods. Such bursty traffic has also been reported in similar works elsewhere [29] [15]. The 80th-percentile error during these periods may be as high as 2 or even 4 meters. The localization system may temporarily blackout when WiFi traffic is high. For a stable RSSI-based localization system in an everyday working environment, a frequency hopping mechanism is proposed and will be detailed in the next section.

## Chapter 6

# Frequency Hopping Mechanism

For a RSSI-based localization system robust to background noise, this study proposes a frequency hopping mechanism consisting of (1) a diagnostic test to detect whether the sensor network is substantially influenced by background noise and (2) a protocol for signaling all nodes semi-synchronously hop to the next channel. The following subsections describe in detail the rationale for running the diagnostic test at the beacon nodes rather than at the receiving tag. The use of hidden Markov model for diagnostic testing, and the simple signaling protocol are also described.

### 6.1 Beacon Node

As observed in Section 5, the accuracy of an RSSI-based localization system is sensitive to wireless signals traveling on the same frequency band. The resulting beacon message losses cause localization errors. To hop away from busy channels, the system

must be able to accurately detect such channels.

One obvious solution is to track the reception rate of the beacon messages at the receiving tag. However, the receiving tag may appear anywhere in the surveyed region. Regardless of the beacon network density, some regions, particularly those at the corners, should receive a small number of beacon messages even when the channel is quiet. A diagnostic test based on the number of beacon messages received at the receiving tag would not be robust. On the other hand, the effect of beacon message loss at the receiving tag is observable from the neighboring beacons (see section 5.4). The diagnostic test can thus be performed on individual beacon nodes by monitoring reception of the beacon packets from well-connected neighboring beacons.

At the beacon network setup phase, the beacon nodes send messages to each other, possibly during non-working hours for example, to determine the list of well-connected neighboring beacons [38] [40]. Based on the beacon packet reception rate from these well-connected neighbors, a beacon can measure the influence of heavy background traffic.

## 6.2 Diagnostic Test

The accuracy of the diagnostic test to determine whether or not the beacon network should hop is critical to a noise-resilient RSSI-based localization system. Given the trend that packet reception rate decreases as the amount of background traffic increases, observed in Figure 5.4. A naive solution is to set a threshold on the smoothed packet reception rate. However, if the hopping threshold is set high, the diagnostic test may

suffer from occasional beacon message loss due to instantaneous noise, as opposed to sustained background WiFi traffic. The beacon network might therefore hop unnecessarily. In this case, beacon nodes as well as the receiving tag in the network require time to switch to the new frequency channel. During the transition, the receiving tag may be unable to receive beacon messages effectively and thus produce increased localization error. Conversely, if the threshold is set too low, the beacon may not be sufficiently sensitive to detect the presence of background traffic. Furthermore, the best threshold might differ between different pairs of well-connected beacon nodes. Manually tracking the best threshold for each good link would also be tedious, *i.e.*, not scalable.

Instead of trying to find a hard threshold, this study proposed a learning approach to solve the above dilemma by using hidden Markov model (HMM) [28]. Hidden Markov model is well-known for recognizing temporal patterns, which is also seen in daily WiFi traffic (see Figure 5.5). That is, if the data rate of the background WiFi traffic is currently high, it is likely to remain high; if the data rate is currently low, it is likely to remain low. The revealed time dependency in WiFi background traffic also suggests temporal patterns in packet reception rate, knowing the correlation between the WiFi traffic rate and the packet reception rate. This relationship enables application of HMM to the beacon observed packet reception rate.

Hidden Markov model extends the concept of Markov models to include the case where the observation is a probabilistic function of the state. Thus, hidden Markov model is a doubly embedded stochastic process with a hidden underlying stochastic process which can only be observed through another set of stochastic processes produc-



ing the sequence of observations [28]. Using HMM enables modeling of the hopping decision, the underlying stochastic process, according to the sequence of measured packet reception rate, the observable stochastic process.

To obtain a model, we feed packet reception rate as the observation sequence into the trainer. The HMM consists of several hidden states and their corresponding Gaussian distributions which characterize the observation of each hidden state. Applying the Expectation-Maximization (EM) algorithm enables adjustment of model parameters to maximize the probability of the training observation sequences. The hidden states can be defined as ‘Hop’ or ‘No-Hop’ based on environments and strategies. With the model, a diagnostic test can be performed on beacons by calculating the state probability of each observed PRR. Forward algorithm [6] is chosen for the state estimation computation because of its simplicity, more suitable for resource constrained sensor nodes.

### 6.2.1 State Modeling

Given the packet reception rate of a link near the wireless AP as the observation values, the following parameters are estimated by the Expectation-Maximization (EM) algorithm. A general HMM represents by  $N$  states, denoted as  $S_1, \dots, S_i, \dots, S_N$ . Assuming the observation  $O_t$  can be characterized by a Gaussian distribution  $O_t \in \{N(\mu_i, \sigma_i^2)\}$  with mean  $\mu_i$  and variance  $\sigma_i^2$ , the state modeling problem can be formulated as, given observation sequence  $O = O_1, O_2, \dots, O_T$ , find a model  $\lambda = (N, A, B, \pi)$  that maximizes  $P(O|\lambda)$ . The notation is as follows:

- $\pi$  : Initial state distribution

$\pi_i = P(q_1 = S_i)$ ,  $q_1 \in \{S_1, S_2, \dots, S_N\}$ ,  $q_t$  represents the estimated state at time  $t$ .

- $A = \{a_{ij}\}$  : state transition probability distribution given the previous state  $i$ ; probability of the next state  $j$ .

$$a_{ij} = P(q_t = S_j | q_{t-1} = S_i)$$

- $B = \{b_i(O_t)\}$  : observation probability distribution, given the state  $i$  at time  $t$ ; the probability that an observation  $O_t$  is observed at state  $i$

$$b_i(O_t) = P(O_t \in \{N(\mu_i, \sigma_i^2)\} | q_t = S_i)$$

After training, certain hidden states can be designated as ‘Hop’, while others are ‘No-Hop’. For example, in the most conservative way, the state with the lowest observation mean is considered as ‘Hop’ and the others as ‘No-Hop’. Hopping would then only occur when PRR is low enough. In this study, two-state and three-state HMM are both used and only the state with the lowest observation mean is denoted as ‘Hop’. Figure 6.1 gives an illustration of a two-state HMM with continuous observation.

Note that the observation, PRR, is calculated over a window of fifty. Thus, only fifty-one PRR observations, *i.e.*,  $0, 1, \dots, 50$ , are possible; the state machine can therefore be modeled by discrete observations. However, the window size is in fact an adjustable parameter, and may affect the accuracy of the diagnostic tests. Because the optimal window size for individual environment is unknown, characterizing the PRR observations as a continuous Gaussian distribution is more appropriate.

## 6.2.2 State Estimation

With the derived HMM model  $\lambda$  and the observation sequence  $O = O_1, O_2, \dots, O_t$ , the probability  $P(q_t = S_i, O = O_1, O_2, \dots, O_t | \lambda)$  can be found with the Forward algorithm. The detailed derivation of the Forward algorithm can be found in [28]. The estimated state at time  $t$  would be the state with the highest probability. Here, the observation sequence can be the instantaneous PRR of a good link or a look-ahead window of size  $T_s$  that records the past  $T_s$  PRR. Since the PRR observation is characterized by Gaussian distributions, in Forward algorithm, exponential computation will be needed to compute  $b_i(O_t) = \frac{1}{\sqrt{2\pi}\sigma_i} e^{-(O_t - \mu_i)/(2\sigma_i^2)}$ , where  $O_t$  are the values of PRR observation and  $\mu_i, \sigma_i$  are the parameters of the Gaussian distribution in state  $i$ . Computing the exponential function on the sensor nodes would not be feasible. However, such function can be approximated by a look-up table, where there is a limited number of PRR values. The look-up table can be computed in advance to capture the above  $b_i(O_t)$  function. Given the state machine and an implementation of the Forward algorithm, the beacon node can perform the hopping decision inference on board. The Forward algorithm, whose pseudo code is shown in Algorithm 1, is compact and reasonably tractable on Telos-like sensor node platforms. More precisely, for a HMM with  $N$  states, only  $N(N+1)$  multiplications and  $N(N-1)$  additions are needed for each observation. There will be 6 multiplications and 2 additions for the two-state HMM case; and 12 multiplications and 6 additions for the three-state HMM case.

**Algorithm 1:** Forward Algorithm

$N$ : the number of states in the HMM

$\pi$ : the set of prior prob.  $\{\pi_i\}, i \in \{1, \dots, N\}$

$A$ : the set of transition prob.  $\{a_{ij}\}, i, j \in \{1, \dots, N\}$

$B$ : the set of observation prob. function

$\{b_i\}, i \in \{1, \dots, N\}$

$t$ : time  $t$

$O_t$ : observation at time  $t$

$\alpha_{t-1}(i)$ : the set of prob. of staying at state  $i$  at time  $t-1$

$q_t$ : estimated state at time  $t$

**Input:**  $N, \pi, A, B, t, O_t, \alpha_{t-1}$

**Output:**  $\alpha_t, q_t$

**if**  $t == 1$  **then**

**foreach** state  $i$  **do**

$\alpha_1(i) = \pi_i \times b_i(O_1);$

**end**

**else**

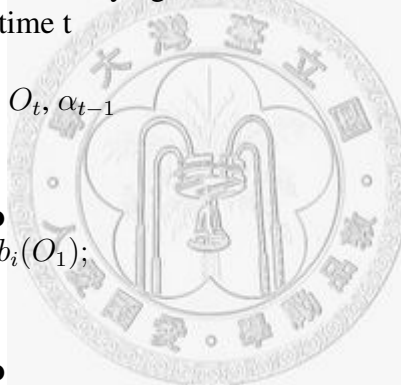
**foreach** state  $i$  **do**

$$\alpha_t(i) = \left[ \sum_{j=1}^N \alpha_{t-1}(j) \times a_{ji} \right] \times b_i(O_t);$$

**end**

**end**

$q_t = \operatorname{argmax}_i \alpha_t(i)$



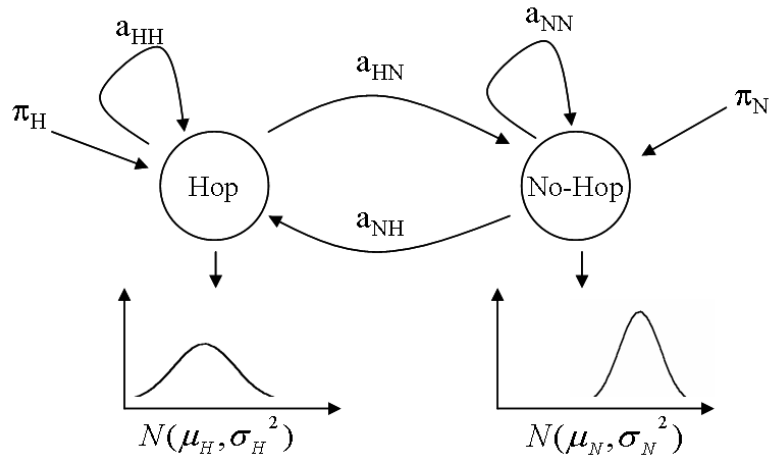


Figure 6.1: Illustration of Two-State HMM

### 6.3 Signaling Protocol

The beacon nodes as well as the receiving tag should hop to the next channel when the hopping signal is detected. If all nodes hop synchronously, the receiving tag can continuously receive beacon packets with minimum interruption. In other words, the shorter the transition time, the better the system stability. However, coordinating all nodes to hop in perfect synchronization is difficult. Instead, a mechanism, originally designed to avoid self-interference in a dense wireless ad hoc network [39], was adopted. This mechanism allows semi-synchronous hopping in beacon networks.

The signaling protocol functions as follows. Each beacon node independently runs the diagnostic test on packet reception rate. Any beacon link making a decision to hop takes the lead and generates a hopping message to its neighbors. Shortly after the message is sent, the beacon node hops to the next channel and waits for other nodes to join. Other nodes (beacon nodes and mobile nodes), upon receiving the hopping

message, propagate the message further before hopping to the next channel themselves. The process continues until there are no further hopping messages in the network. For the best case that the hopping message can reach all the nodes in the network hop by hop, the transition time  $T$  will be

$$T = \text{network hop counts} \times (\text{transmission} + \text{processing}) \text{ delay of the hopping message per hop}$$

The worst case is that the hopping message cannot reach any nodes in the network. When the first beacon node hops to a new channel, the packet reception rate (PRR) of its neighboring nodes will eventually drop to zero after a period of time. In this case, those neighboring nodes will hop due to the diagnostic test on the zero PRR. The maximum transition time  $T_{max}$  will be

$$T_{max} = \text{network hop counts} \times \text{window size of PRR}$$

For the mobile nodes, if they can receive the hopping message, they will hop very fast. Otherwise, since no diagnostic test is run on the mobile nodes, we let them hop if they cannot receive any beacon messages for a period of time (window size for calculating the PRR).

At last, to ensure that all the nodes will remain in the same channel after the hopping process, each node needs to wait for the maximum transition time before it starts the diagnostic test again in the new channel. Although the theoretical maximum transition time can be as large as dozens of seconds for middle-scale network, practically the hopping messages, propagated in such a flooding fashion, often result in multiple

copies of the messages being passed around the network. Even when the network is affected by background traffic and has a high packet loss rate, all nodes are likely to receive the hopping message and hop to the next channel within a small amount of time.



# Chapter 7

## Evaluation

In this section, trace-driven simulations, are used to evaluate the key components in the proposed frequency hopping mechanism: (1) transition delay of the hopping signaling protocol and the (2) accuracy of the diagnostic tests when using more realistic and long-term traces.

### 7.1 Trace Synthesis

Results of the evaluation depend on the interaction between a Zigbee-based indoor localization system and WiFi noise in an actual environment. To obtain long-term RSSI, PRR and WiFi co-traces, we could follow the measurement methodology detailed in Section 4 and simply let the receiving tag, beacon nodes, and the WiFi traffic logger run simultaneously for several days. However, because the system was installed in a public area, continuously monitoring the receiving tag throughout the experiment



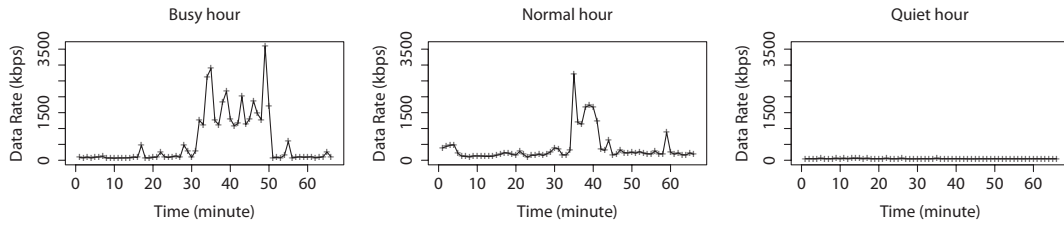


Figure 7.1: One-Hour WiFi Data Rate

would be impractical. Leaving the receiving tag in the hallway unmonitored would also raise technical and security concerns. Pedestrians walking by the receiving tag would have significantly increased RSSI variance. Factoring out the effect of nearby obstacles in the evaluation would be difficult without knowing the pedestrian traffic in the hallway. The receiving tag might also be removed by curious trespassers. Early attempts to collect long-term traces by leaving the receiving tag in the hallway also generated complaints from other residents in the building.

Therefore, only long-term WiFi traces were taken. The long-term RSSI and PRR traces were synthesized by using the short-term RSSI and PRR traces collected at different WiFi traffic rates. The average data rate of the collected long-term WiFi traffic was calculated once per minute, and a one-minute segment was randomly selected from the RSSI and PRR traces at the closest WiFi data rate. All the selected one-minute segments were merged into hours of RSSI and PRR traces. In the experiments described in detail below, three one-hour segments were selected from the synthesized traces, *i.e.*, during busy (peak hours in a day), normal (throughout weekdays), and quiet (after midnight) hours. (see Fig. 7.1).

## 7.2 Hopping Time

Assuming the pattern of hopping message losses is similar to that of beacon message losses, the time required for all beacons to hop to the next channel is simulated by using the beacon packet reception rates in the PRR traces measured during different levels of background WiFi traffic. Figure 7.2 shows the average time required to signal every beacon node and hop to another channel. Here the average transmission and processing delay of the hopping message per hop is assumed to be 5ms. When background traffic rate is low, hopping requires an average of 8ms. The flooding-like approach used by the signaling mechanism produces a reliable and consistent signal, and all beacons can be reached from the initiating beacon through the shortest path. The hopping delay is therefore minimized. If the interference level is higher, signaling packets are more susceptible to corruption by jamming signals. Hopping time increases slightly as the average hop count of the signaling packets increases, as Figure 7.2 shows. Although packet loss becomes more severe, all beacon nodes in the testbed still receive the signaling packets because of the flooding mechanism. Throughout the experiments, the worst case as described in Section 6.3 is not observed, so that the hopping time here is not bounded by the window size for calculating the packet reception rate.

## 7.3 Diagnostic Test

In this subsection, we examine qualitatively the inferred ‘Hop’ and ‘No-Hop’ behaviors using the PRR threshold and HMM approaches.

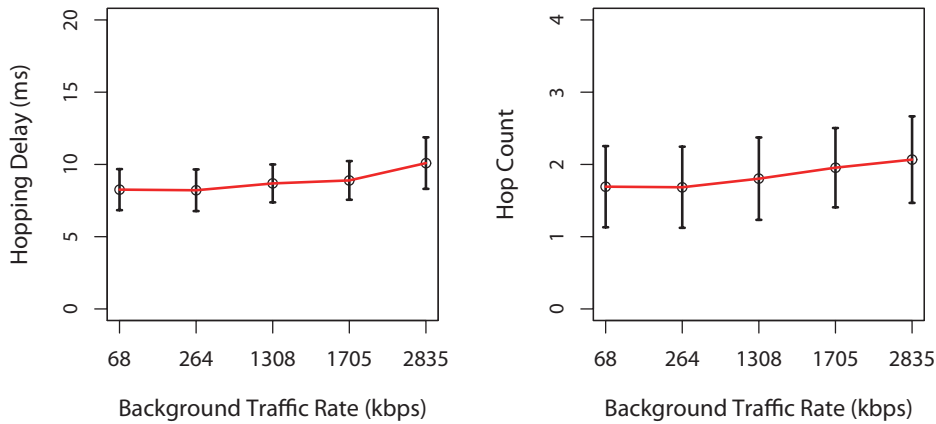


Figure 7.2: Hopping Time

### 7.3.1 PRR Thresholds

As mentioned in Section 6, a naive, though not scalable, solution for making hop decision is to set a threshold on the observed packet reception rate (PRR). The challenge is how to find a proper threshold. A brute-force approach is to conduct a trace-driven simulation on every possible threshold setting and select the one that can minimize localization error. We take the normal-hour PRR trace for the simulations. Taking one PRR value at a time, if the observed PRR is lower than the threshold, a ‘Hop’ decision would be made; otherwise, a ‘No-Hop’ decision would be made. When the ‘No-Hop’ decision is made, the simulation then takes the next PRR data in the trace and repeats the decision making process. When ‘Hop’ is indicated, the simulation moves to a data point randomly selected from the PRR trace to simulate hopping to the “new” channel. We assume no beacon messages are received in the first period after hopping, *i.e.*, the first localization error in the new channel is set to be a predetermined maximum. If the simulation reaches the end of the trace, it rounds to the beginning of the trace. An

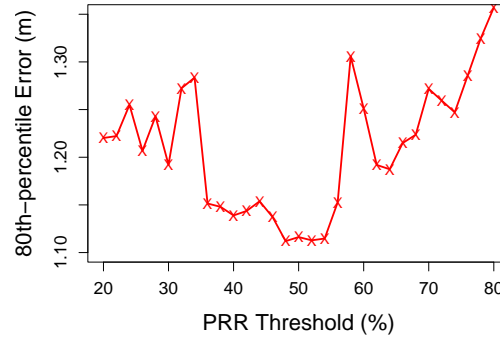


Figure 7.3: Localization Error with Different PRR Thresholds

implicit assumption in such simulations is that all channels behave similarly during the normal hours. Each simulation runs for 3 hours. Note that, simulating in this fashion, when PRRs are computed using a sliding window, the first PRR value should be calculated based on the dynamics of the beacon messages in the first and last forty-nine cycles. The first forty-nine PRRs should be computed similarly.

Figure 7.3 shows the 80th-percentile localization error achieved by setting PRR threshold from 20% to 80%. From the simulation results, setting PRR threshold at 48% can achieve the most accurate localization estimation. The system would be unable to detect the presence of background traffic with too low thresholds, resulting in higher packet losses. Therefore, the localization error is higher for PRR threshold lower than 48%. On the other hand, frequent unnecessary hopping decisions are made with too high thresholds. Therefore, the localization error increases when the threshold is higher than 48%.

Setting 48% as the PRR threshold, another set of simulation was carried out with the busy-hour trace. Figure 7.4 shows the data points predicted as ‘Hop’ in red(dark)

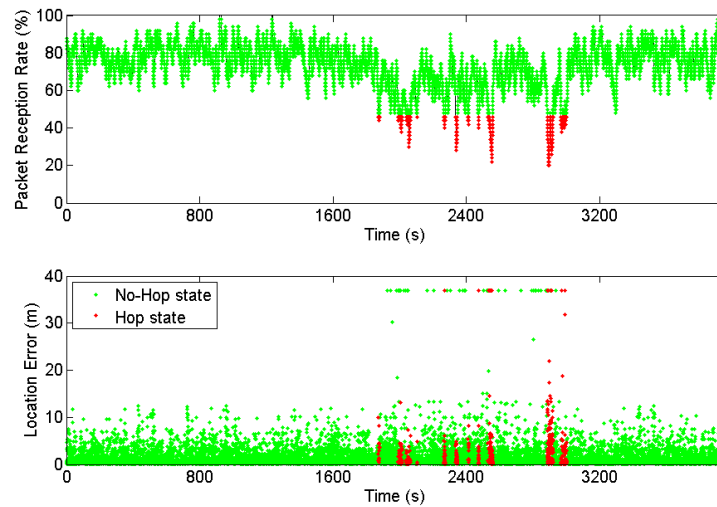


Figure 7.4: Hopping Decision of PRR-Thresholding. Red dots: Hop. Green dots: No-Hop.

and those predicted as ‘No-Hop’ in green(gray). The ‘Hop’ data points generally correspond to the 20-minute interval during which a surge of WiFi traffic occurs. A substantial number of false negative were noted while no false alarms were detected. Here, the false negative is defined as a data point that is within the WiFi traffic burst and results in high localization error, but the diagnostic test reveals a prediction of ‘No-Hop’. The false alarm is defined as a data point that is beyond the WiFi traffic burst but is predicted as a ‘Hop’ point.

### 7.3.2 Hidden Markov Model

Instead of brute-force searching for a good threshold, with HMM the system can learn to distinguish busy and quiet channel from measured packet reception rates and make hopping decisions. To train a model, the normal-hour is applied, and the initial

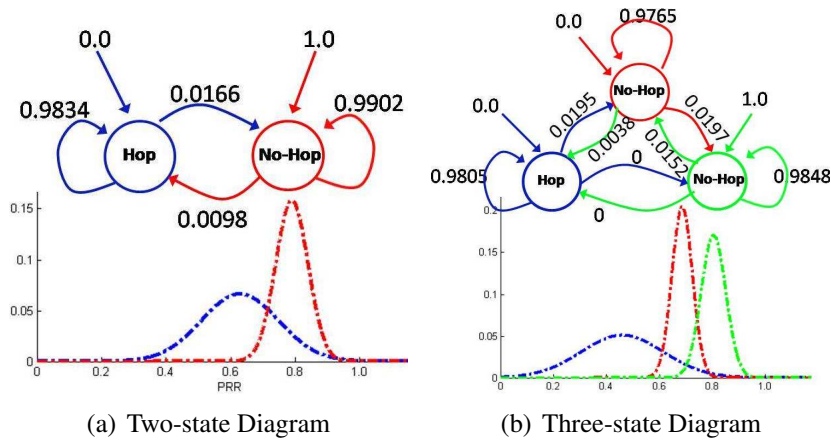
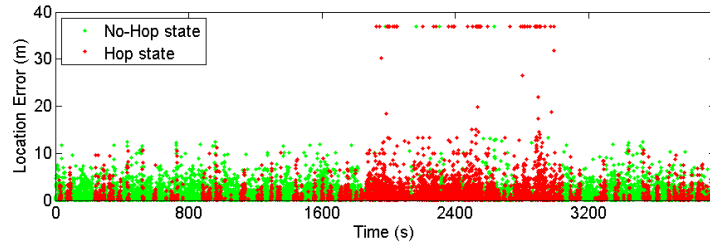
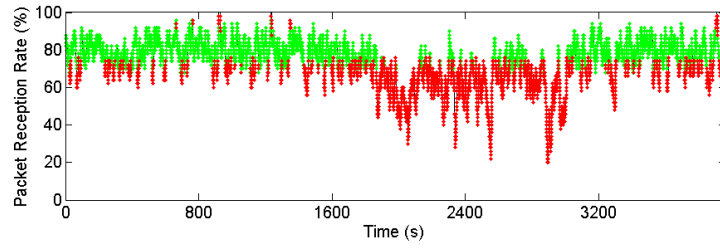


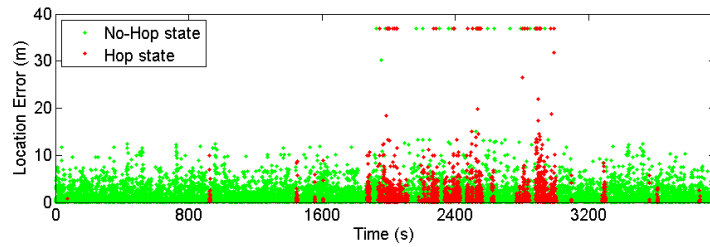
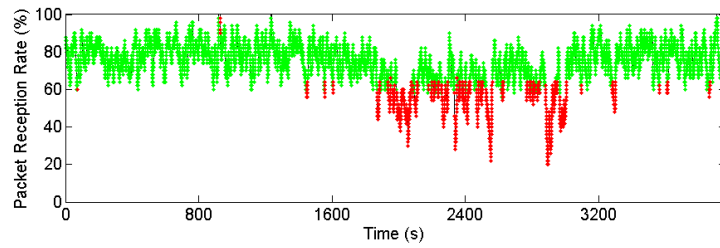
Figure 7.5: State Diagram of HMM

values for parameters  $\mu$  and  $\sigma$  in each state is derived from the K-means algorithm. The K-means algorithm identifies partitions such that the data points within the same state are as close together as possible, and the data points in different states are as far apart as possible. For the HMM-related computation, a Matlab HMM toolbox was employed [21].

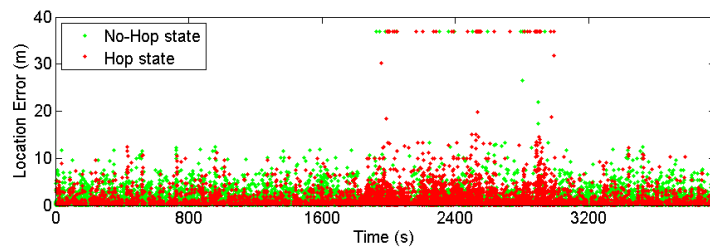
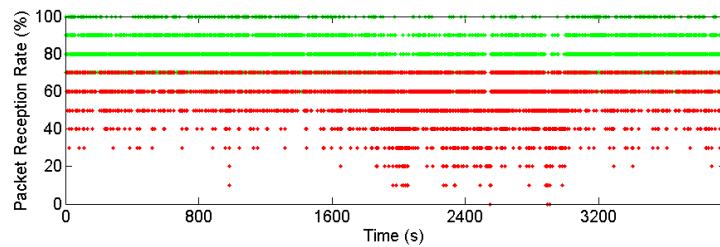
A two-state HMM is first employed, since there is a straight-forward mapping between the two states and the two decisions, ‘Hop’ or ‘No-Hop’. Figure 7.5(a) shows the state machine obtained after training with the normal trace. The state with lower PRR mean is denoted as ‘Hop’, while the other is ‘No-Hop’. A trace-driven simulation based on the derived model then runs with the busy-hour trace, and the result is shown on Figure 7.6(a). In the figure, the red(dark) points are the data points predicted as ‘Hop’, while the green(gray) points are those predicted as ‘No-Hop’. As in the previous section, the ‘Hop’ data points generally correspond to the 20-minute interval during which a surge of WiFi traffic occurs. However, a substantial number of false alarms were also noted. Frequent false alarms may cause the system to hop un-



(a) Two-state Model



(b) Three-state Model



(c) Three-state Model (Window Size=10)

Figure 7.6: State Prediction of HMM. Red dots: Hop. Green dots: No-Hop.

necessarily. In this case, the receiving tag would be unable to reliably receive beacon messages during the hopping transition and the accuracy of the location system would deteriorate.

Compared to the results of PRR thresholding in Figure 7.4, we have observed that those false alarms are composed by the data points who only suffered from mild packet losses and still can achieve high localization accuracy. This phenomenon can also be found in Figure 4(b), the localization error remains low while the number of receiving beacon packets decrease to 8 or 7 packets. An optimization of this two-state HMM is to create another hidden state to capture the PRR data points who are only affected by mild background traffic.

We then evaluate a three-state HMM, and classify the three states in the most conservative way. The resulted state machine is depicted in Figure 7.5(b). Only the states with the lowest PRR mean is designated ‘Hop’, while others are ‘No-Hop’ states. Figure 7.6(b) shows that three-state HMM can effectively lower the false alarm rate by accommodating the above mentioned data points with one more ‘No-Hop’ state.

## 7.4 Localization Error with Frequency Hopping

Figure 7.7(a) shows the CDF of localization errors during a busy hour using two-state HMM, three-state HMM, the 48% PRR threshold, as well as the localization errors without frequency hopping. Both two-state and three-state HMM improve the 80th-percentile localization error. For three-state HMM, the improvement is as high as 28% (from 1.82m to 1.32m) and achieved similar accuracy as the brute-force 48%



PRR threshold setting. The above results show that the frequency hopping mechanism can avoid unnecessary beacon packet losses and still maintain localization accuracy. Conversely, the original localization system without frequency hopping must undergo a period of high localization error.

The 50th-percentile localization errors, however, are similar because the interference level is not high during most of the one hour period. The frequency hopping mechanism is not intended to improve the average error of the localization system. Instead, it aims at minimizing the variance in localization error by avoiding occasional surges of background traffic. This task is particularly important if the system is to be deployed for everyday use.

To further highlight the advantage of frequency hopping, we zoom into the busiest 20 minutes. The 80th-percentile localization error, as shown in Figure 7.7(b), can be reduced from 2.74 meters to 1.24 meters. Localization error of the PRR thresholding approach is slightly worse than three-state HMM. The PRR thresholding approach, though introduces no false alarms, reacts slower to the degraded wireless environment. The delay in responding to busy background traffic leads to the less accurate location estimates during the busy period, but gives a better overall performance by avoiding unnecessary hopping during the quiet period.

### 7.4.1 Busy vs. Quiet Hour

Figure 7.7(c) shows the CDF of localization errors during a quiet hour. Both HMM models show slightly worse 80-percentile localization error than the no-hop case. This is because any hopping in this trace is unnecessary. In addition, even if the system

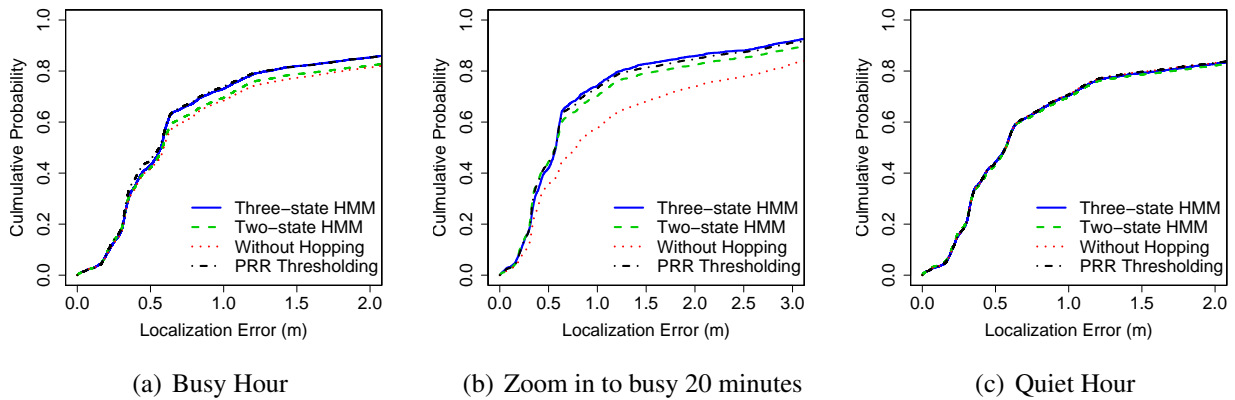


Figure 7.7: Localization Accuracy in Busy/Quiet Hour

successfully hops to a very quiet channel, no particular benefit can be achieved. In fact, there is a slight penalty of not being able to receive beacon messages during the transition of hopping.

## 7.4.2 Sliding Window Size

The sliding window size for computing the packet reception rate is a tunable parameter in our localization system. PRR represents the beacon message reception rate over a period of time, which is the sliding window size. If there is a WiFi traffic burst, it would take a certain amount of time for the PRR to reflect the change. The delay can be reduced by setting a smaller window size. However, the false alarm rate becomes higher as a smaller window size is more sensitive to short-term noise. As shown in Figure 7.6(b) and 7.6(c), a window size of 10 incurs substantial amount of unnecessary hopping while a window size of 50 makes the system more stable. To observe the effect of different window sizes, we take the busy-hour trace and 3-state HMM method and

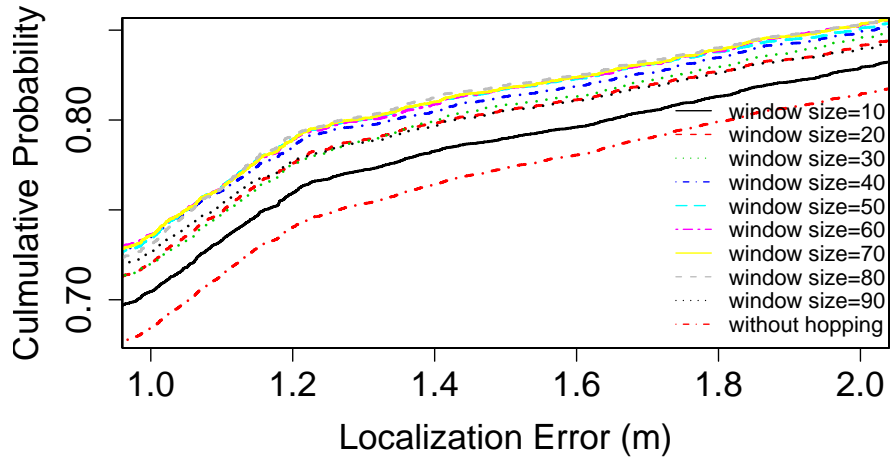


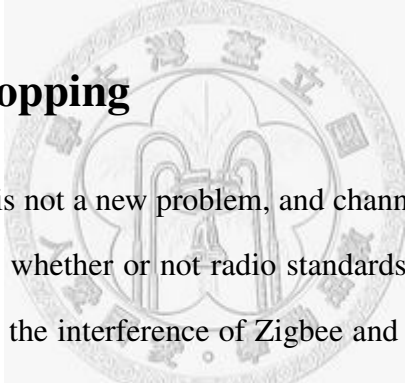
Figure 7.8: Impact of Window Size on Localization Accuracy

simulate the localization error varying the window size from 10 to 90. Figure 7.8 are the resulting localization errors. As window size increases from 10 to 40, the error decreases and reaches smallest when the window size is around 50 to 70. Then the error increases again when window size goes from 80 to 90. In this sense, a window size of 50 gives a good tradeoff between delay and false alarm rate.

# Chapter 8

## Related Work

### 8.1 Channel Hopping



Co-channel interference is not a new problem, and channel hopping is a popular solution. The problem exists whether or not radio standards are similar. Recently Gummedi et al. [12] analyzed the interference of Zigbee and cordless phone using 802.11 networks. Gummedi proposed a channel hopping mechanism to mitigate the impact. Utilizing multiple channels also helps to maximize network capacity. [4] and [20] both studied the channel hopping problem within 802.11 networks for network throughput improvement. In the domain of wireless security, [39] used channel hopping as a countermeasure to wireless jamming attack. The mechanism proposed in the current study, however, differs in that the channel hopping mechanism employs beacons for robust localization.

Several works are proposed to understand the packet delivery in low power wire-

less networks [40] [34] [16]. [33] conducted experiments to understand the interference among the same low-power radio network. [34] reported the correlation between RSSI and packet reception rate. [16] proposed to use the measured signal strength of interference to simulate packet delivery rate. Our work, on the contrary, utilizes the measured packet delivery rate to infer the level of interference.

## 8.2 Localization Systems

The various techniques developed for indoor localization commonly require signal transmissions between pre-deployed beacons and observed target. For example, sonic, ultrasonic, infrared, RF, camera, and UWB are all commonly used as signal sources. The major differences are the calibration methods and signal type.

Assuming that signals propagate at constant velocity, TOA (time of arrival) [25] is the most common method of estimating distance by measuring signal propagation time. The AOA (angle of arrival) [23] is a network-based technique for exploiting geometric properties of arriving signals. By measuring the angles of the arriving signal at more than one receiver, AOA achieves precise localization. Another network-based technique, TDOA (time difference of arrival) [31] infers distance by measuring time difference instead of angle. Some hybrid approaches combining TOA, AOA, and TDOA have also been proposed [8]. The FDOA (frequency difference of arrival) approach [19] exploits signal interference patterns from multiple beacons for distance estimation.

Another class of techniques measures the received signal strength indication (RSSI).

These techniques exploit the decay model of electronic-magnetic fields to determine distance by monitoring RSSI distance [24] [22] [17]. Frequency bands used for transmission also vary. For example, the well-known RADAR system [5] uses radio frequency (RF) whereas LADAR and SONAR use visible light and audible sound bands, respectively. Cricket [27] is a hybrid approach using both RF and ultrasonic bands. The UWB-based systems, such as Ubisense [36], use very wideband to solve the problems of multi-path and environmental interference. However, UWB-based systems require specialized hardware to achieve GHz sampling rates and nanosecond time synchronization. Adding such specialized hardware increases costs on typically resource-constrained sensor nodes.

In addition to range-based systems, range-free systems have also been proposed. Such systems do not localize targets based on range estimation. For example, APIT [13] estimates target location by analyzing connectivity information to anchor nodes with known locations. The more anchor nodes deployed, the greater precision of the technique. Spotlight [35] and Lighthouse [30] correlate the event detection time of a sensor node with the known spatiotemporal relationship. Detection events can then be mapped to a possible position.

Both range-based and range-free localization systems are affected by co-channel interference since the frequency band they use to deliver localization messages may be shared with other users, particularly in the increasingly crowded RF frequency band. Although RSSI-signature-based localization system was adopted for evaluation, the basic concept and mechanism of frequency hopping are not limited to it.

## Chapter 9

### Conclusion and Future Works

We have demonstrated in this study that the proposed frequency hopping mechanism can achieve accurate, stable indoor localization in daily environments. The 80th percentile and 50th percentile localization errors are kept low to approximately 1.3 meters and 0.6 meters, even when the environment is experiencing heavy interference. Although using a longer beacon message collection time may also mitigate the localization error due to beacon message loss, the proposed frequency hopping mechanism pushes the envelope further. It enables accurate and stable indoor localization with minimum delay, i.e., the real-time location systems with a rising market demand.

Our intention is to implement the frequency hopping mechanism on our localization testbed and evaluate the long-term stability under realistic WiFi influences. In the design of our frequency hopping mechanism, each beacon runs a diagnostic test on beacon packets received from each well-connected neighboring beacon. If the beacon finds any of the well-connected neighbors is busy, the beacon initiates the signaling

protocol and informs everyone to hop to the next frequency. The implementation of the Forward algorithm and the signaling protocol is relatively straightforward. Obtaining the HMMs, however, requires an extended measurement study with WiFi traffic systematically generated at different locations. This is top on our future plan pursuing research in robust real-time location systems.





# Bibliography

- [1] Real time locating systems 2006-2016 (RTLS). *GIIExpress Market Research Report*, Jul 01, 2007. <http://www.giiexpress.com/products/ix37643/>.
- [2] D. Aguayo, J. Bicket, S. Biswas, G. Judd, and R. Morris. Link-level measurements from an 802.11b mesh network. In *Proceedings of SIGCOMM '04*, pages 121–132, New York, NY, USA, 2004. ACM.
- [3] B. Bacheldor. RTLS market to grow 30 percent annually. *RFID Journal*, May 10, 2006. <http://www.rfidjournal.com/article/articleview/2325/1/1/>.
- [4] P. Bahl, R. Chandra, and J. Dunagan. SSCH: slotted seeded channel hopping for capacity improvement in IEEE 802.11 ad-hoc wireless networks. In *Proceedings of MobiCom '04*, pages 216–230, New York, NY, USA, 2004. ACM.
- [5] P. Bahl and V. Padmanabhan. An in building RF-based user location and tracking system. *IEEE INFOCOM 2001*, April 2001.
- [6] L. E. Baum and J. A. Eagon. An inequality with applications to statistical estimation for probalistic functions of Markov processes and to a model for ecology. *Bull. Am. Math. Soc.*, 73:360–363, 1967.
- [7] Y.-C. Cheng, J. Bellardo, P. Benkö, A. C. Snoeren, G. M. Voelker, and S. Savage. Jigsaw: solving the puzzle of enterprise 802.11 analysis. In *Proceedings of SIGCOMM '06*, pages 39–50, New York, NY, USA, 2006. ACM.

- [8] L. Cong and W. Zhuang. Hybrid TDOA/AOA mobile user location for wideband CDMA cellular systems. *IEEE Transactions on Wireless Communications*, 1(3):439–447, July 2002.
- [9] J. Degeysys, I. Rose, A. Patel, and R. Nagpal. Desync: self-organizing desynchronization and TDMA on wireless sensor networks. In *Proceedings of IPSN '07*, pages 11–20, New York, NY, USA, 2007. ACM.
- [10] dumpcap man page. <http://www.wireshark.org/docs/man-pages/dumpcap.html>.
- [11] E. Elnahrawy, X. Li, and R. Martin. The limits of localization using signal strength: a comparative study. *Sensor and Ad Hoc Communications and Networks, 2004. IEEE SECON 2004. 2004 First Annual IEEE Communications Society Conference on*, pages 406–414, 4-7 Oct. 2004.
- [12] R. Gummadi, D. Wetherall, B. Greenstein, and S. Seshan. Understanding and mitigating the impact of rf interference on 802.11 networks. In *Proceedings of SIGCOMM '07*, pages 385–396, New York, NY, USA, 2007. ACM.
- [13] T. He, C. Huang, B. M. Blum, J. A. Stankovic, and T. Abdelzaher. Range-free localization schemes for large scale sensor networks. In *Proceedings of MobiCom '03*, pages 81–95, New York, NY, USA, 2003. ACM.
- [14] J. Hightower and G. Borriello. Particle filters for location estimation in ubiquitous computing: A case study. In *Proceedings of Ubicomp'04*, pages 88–106, 2004.
- [15] A. P. Jardosh, K. N. Ramachandran, K. C. Almeroth, and E. M. Belding-Royer. Understanding link-layer behavior in highly congested ieee 802.11b wireless networks. In *Proceedings of E-WIND '05*, pages 11–16, New York, NY, USA, 2005. ACM.
- [16] H. Lee, A. Cerpa, and P. Levis. Improving wireless simulation through noise modeling. In *Proceedings of IPSN '07*, pages 21–30, New York, NY, USA, 2007. ACM.

- [17] K. Lorincz and M. Welsh. Motetrack: a robust, decentralized approach to RF-based location tracking. *Personal Ubiquitous Comput.*, 11(6):489–503, 2007.
- [18] C. F. I. LTD. USB extender. <http://www.cfi.com.tw>.
- [19] M. Maróti, P. Völgyesi, S. Dóra, B. Kusý, A. Nádas, Ákos Lédeczi, G. Balogh, and K. Molnár. Radio interferometric geolocation. In *Proceedings of SenSys '05*, pages 1–12, New York, NY, USA, 2005. ACM.
- [20] A. Mishra, V. Shrivastava, D. Agrawal, S. Banerjee, and S. Ganguly. Distributed channel management in uncoordinated wireless environments. In *Proceedings of MobiCom '06*, pages 170–181, New York, NY, USA, 2006. ACM.
- [21] K. Murphy. Hidden markov model (HMM) toolbox for matlab. <http://www.cs.ubc.ca/murphyk/Software/HMM/hmm.html>.
- [22] D. Niculescu. Positioning in ad hoc sensor networks. *IEEE Networks*, 18(4):24–29, July-August 2004.
- [23] D. Niculescu and B. Nath. Ad hoc positioning system (aps) using aoa. *INFOCOM 2003. Twenty-Second Annual Joint Conference of the IEEE Computer and Communications Societies. IEEE*, 3:1734–1743 vol.3, 30 March-3 April 2003.
- [24] N. Patwari. Relative location estimation in wireless sensor networks. *IEEE Transactions on Signal processing*, 51(8):2137–2148, August 2003.
- [25] N. Patwari, I. Hero, A.O., M. Perkins, N. Correal, and R. O’Dea. Relative location estimation in wireless sensor networks. *Signal Processing, IEEE Transactions on [see also Acoustics, Speech, and Signal Processing, IEEE Transactions on]*, 51(8):2137–2148, Aug. 2003.
- [26] J. Polastre, R. Szewczyk, and D. Culler. Telos: enabling ultra-low power wireless research. In *Proceedings of IPSN '05*, page 48, Piscataway, NJ, USA, 2005. IEEE Press.

- [27] N. Priyantha, A. Charkraborty, and H. Balakrishnan. The cricket location support system. *ACM MobiCom 2000*, pages 1–14, August 2000.
- [28] L. R. Rabiner. A tutorial on hidden markov models and selected applications in speech recognition. *Readings in speech recognition*, pages 267–296, 1990.
- [29] M. Rodrig, C. Reis, R. Mahajan, D. Wetherall, and J. Zahorjan. Measurement-based characterization of 802.11 in a hotspot setting. In *Proceedings of E-WIND '05*, pages 5–10, New York, NY, USA, 2005. ACM.
- [30] K. Römer. The lighthouse location system for smart dust. In *Proceedings of MobiSys '03*, pages 15–30, New York, NY, USA, 2003. ACM.
- [31] A. Savvides, C.-C. Han, and M. B. Strivastava. Dynamic fine-grained localization in ad-hoc networks of sensors. In *Proceedings of MobiCom '01*, pages 166–179, New York, NY, USA, 2001. ACM.
- [32] V. Seshadri, G. V. Zaruba, and M. Huber. A bayesian sampling approach to in-door localization of wireless devices using received signal strength indication. In *Proceedings of PERCOM '05*, pages 75–84, Washington, DC, USA, 2005. IEEE Computer Society.
- [33] D. Son, B. Krishnamachari, and J. Heidemann. Experimental study of concurrent transmission in wireless sensor networks. In *Proceedings of SenSys '06*, pages 237–250, New York, NY, USA, 2006. ACM.
- [34] K. Srinivasan, P. Dutta, A. Tavakoli, and P. Levis. Understanding the causes of packet delivery success and failure in dense wireless sensor networks. In *Technical Report SING-06-00*, 2006.
- [35] R. Stoleru, T. He, J. A. Stankovic, and D. Luebke. A high-accuracy, low-cost localization system for wireless sensor networks. In *Proceedings of SenSys '05*, pages 13–26, New York, NY, USA, 2005. ACM.
- [36] Ubisense. <http://www.ubisense.net>.

- [37] G. Werner-Allen, P. Swieskowski, and M. Welsh. Motelab: a wireless sensor network testbed. In *Proceedings of IPSN '05*, page 68, Piscataway, NJ, USA, 2005. IEEE Press.
- [38] A. Woo, T. Tong, and D. Culler. Taming the underlying challenges of reliable multihop routing in sensor networks. In *Proceedings of SenSys '03*, pages 14–27, New York, NY, USA, 2003. ACM.
- [39] W. Xu, W. Trappe, and Y. Zhang. Channel surfing: defending wireless sensor networks from interference. In *Proceedings of IPSN '07*, pages 499–508, New York, NY, USA, 2007. ACM.
- [40] J. Zhao and R. Govindan. Understanding packet delivery performance in dense wireless sensor networks. In *Proceedings of SenSys '03*, pages 1–13, New York, NY, USA, 2003. ACM.

