Localization Systems Using Signal Strength Fingerprinting

by

Kung-Chung Lee

BASc Engineering Physics, The University of British Columbia, 2007

A THESIS SUBMITTED IN PARTIAL FULFILLMENT
OF THE REQUIREMENTS FOR THE DEGREE OF

Master of Applied Science

in

THE FACULTY OF GRADUATE STUDIES
(Electrical and Computer Engineering)

THE UNIVERSITY OF BRITISH COLUMBIA
(Vancouver)

September 2010

© Kung-Chung Lee, 2010

Abstract

The task of estimating the location of a mobile transceiver using the Received Signal Strength Indication (RSSI) values of radio transmissions to/from other radios is an inference problem. The fingerprinting paradigm is the most promising genre of methods studied in the literature. It constructs deterministic or probabilistic models from data sampled at the site. Probabilistic formulations are popular because they can be used under the Bayesian filter framework. We also categorize fingerprinting methods into regression or classification. The vast majority of existing methods perform regression as they estimate location information in terms of position coordinates. In contrast, the classification approach only estimates a specific region (e.g., kitchen or bedroom). This thesis is a continuation of studies on the fingerprinting paradigm.

For the regression approach, we perform a comparison between the Unscentend Kalman Filter (UKF) and the Particle Filter (PF), two suboptimal solutions for the Bayesian filter. The UKF assumes near-linearity and imposes unimodal Gaussian densities while the PF does not. These assumptions are very fragile and we show that the UKF is not a robust solution in practice.

For the classification approach, we are intrigued by a simple method we name the Simple Gaussian Classifier (SGC). We ponder if this simple method comes at a cost in terms of classification errors. We compare the SGC against the K-Nearest Neighbor (KNN) and Support Vector Machine (SVM), two other popular classifiers. Experimental results present evidence that the SGC is very competitive. Furthermore, because the SGC is written in closed-form, it can be used directly under the Bayesian filter framework, which is better known as the Hidden Markov Model (HMM) filter.

The fingerprinting paradigm is powerful but it suffers from the fact that conditions may change. We propose extending the Bayesian filter framework by utilizing the filter derivative to realize an online estimation scheme, which tracks the time-varying parameters. Preliminary results show some promise but further work is needed to validate its performance.

Preface

- A version of Chapter 5 has been published. Kung-Chung Lee, Anand Oka, Emmanuel Pollakis, and Lutz Lampe, 'A Comparison between Unscented Kalman Filtering and Particle Filtering for RSSI-based Tracking,' in Proc. of 7th Workshop on Positioning, Navigation and Communication (WPNC), Dresden, Germany, March 2010.

- A version of Chapter 6 has been submitted for publication. Kung-Chung Lee and Lutz Lampe, 'Indoor Cell-Level Localization Based on RSSI Classification,' submitted to 2011 Canadian Conference on Electrical and Computer Engineering (CCECE), Niagara Falls, Ontario, Canada, May 2011.

I have transfered my copyright to the organizers of the conferences above. However, I have retained my copyright for writing this thesis.

I am the primary author for the publications above. I have performed the majority of the work. Tasks include but are not limited to literature review, software and hardware design, performing experiments, data analysis and manuscript editing. While my collaborators have provided invaluable help, their roles are secondary.

Table of Contents

List of Tables

List of Figures

Glossary

The following are abbreviations and acronyms used in this thesis listed in alphabetical order.

ANN Artifical Neutral Networks

AOA Angle of Arrival

AP Access Point

BER Bit Error Rate

EKF Extended Kalman Filter

EM Expectation-Maximization

FC Fusion Center

GPS Global Positioning System

HMM Hidden Markov Model

KF Kalman Filter

KNN K-Nearest Neighbor

LDA Linear Discriminant Analysis

LOS Line-of-Sight

ML Maximum Likelihood

MSE Mean Squared Error

NLOS Non-Line-of-Sight

PEP Pairwise Error Probability

PF Particle Filter

RBF Radial Basis Function

RF Radio Frequency

RSSI Received Signal Strength Indication

SGC Simple Gaussian Classifier

SIR Sampling Importance Resampling

SNR Signal-to-Noise Ratio

SVM Support Vector Machine

TDOA Time Difference of Arrival

TOA Time of Arrival

UKF Unscentend Kalman Filter

UWB Ultra-Wideband

WLAN Wireless Local Area Network

Acknowledgments

I have been fortunate to work with many supportive friends from Kaiser 4090 at University of British Columbia. In particular, the first half of this thesis came directly from collaborative work with Anand Oka and Emmanuel Pollakis. I want to thank the people at Wireless 2000, Vladimir Goldenberg, Harry Lam, Cuong Lai and Vincent Chan, for their incredible support even during difficult times. I want to thank Wenbo Shi for sharing his experimental data. I learned a great deal from Professor Nando de Freitas' course on Machine Learning and his helpful insights on Bayesian filtering. I would also like to thank the committee members, Professor Cyril Leung and Professor Vikram Krishnamurthy, for taking time out their busy schedules. Finally but not least, I would like to express my sincere gratitude to Professor Lutz Lampe for his support. This thesis would not have been possible without his patient and detailed help.

Dedication

in principio creavit Deus caelum et terram

terra autem erat inanis et vacua et tenebrae super faciem abyssi et spiritus Dei ferebatur super aquas

dixitque Deus fiat lux

$$\nabla \cdot \mathbf{E} = \frac{\rho}{\varepsilon_0}$$
$$\nabla \cdot \mathbf{B} = 0$$
$$\nabla \times \mathbf{E} = -\frac{\partial \mathbf{B}}{\partial t}$$
$$\nabla \times \mathbf{B} = \mu_0 \mathbf{J} + \mu_0 \varepsilon_0 \frac{\partial \mathbf{E}}{\partial t}$$

et facta est lux

Chapter 1

Introduction

Location-specific services have become increasingly popular in recent years. Applications include surveillance, access and inventory control, robotics and even location-based marketing [1]. Services enabled by the Global Positioning System (GPS) are ubiquitous. Unfortunately, the use of GPS in indoor environments is quite limited due to the fact that there is rarely a Line-of-Sight (LOS) between a device and a satellite. An alternative solution is to utilize a possibly pre-existing indoor wireless network. The network locates mobile targets carrying transceivers by exploiting metrics of Radio Frequency (RF) transmissions to/from other radios. Typically, a number of sensors are installed at fixed locations and they monitor the mobile transceivers. A sensor may simply be an Access Point (AP) of the network. It is also possible to perform localization in an ad hoc network where every radio is mobile. Traditional metrics include Time of Arrival (TOA), Time Difference of Arrival (TDOA), Angle of Arrival (AOA) and Received Signal Strength Indication (RSSI) [1, 2]. Other possible but less popular metrics include network topology and hop count [3], Bit Error Rate (BER) [4] and Signal-to-Noise Ratio (SNR) [5].

For low-cost applications, TOA and AOA are not particularly attractive because they usually require dedicated hardware components. RSSI-based schemes have the unique advantage that the information is readily available. In almost every technology, RSSI readings are given to higher levels for evaluating the qualities of communication links. Although these readings are not built for localization purposes and they are often not very precise, many studies have shown that it is possible to perform localization, albeit not at submeter accuracies. Technologies such as ZigBee, Bluetooth and Wireless Local Area Network (WLAN) are ubiquitous. Thus, in a sense, RSSI-based schemes almost come for 'free'. Compared to other possibly free metrics such as SNR, RSSI is found to be more dependent on location and thus it can be used for localization better [5]. Therefore, RSSI-based schemes are the most promising for low-cost applications.

Fundamentally, RSSI-based localization is an inference problem. Given some RSSI measurements, the goal is to estimate the location information of the mobile targets. This seemingly easy task is complicated by the fact that it is difficult to obtain propagation models for indoor environments. Due to reflections, refractions and other multipath effects, it is challenging to describe the properties of signal strength measurements. Numerous methods and algorithms have been proposed and studied. The fingerprinting paradigm is the most promising genre of methods. The paradigm works by sampling RSSI values at various locations in the area in order to construct deterministic or probabilistic models. The sampling step is known as offline training and it is the most time consuming part because it requires human intervention. We argue that, for low-cost applications, a localization algorithm needs to be robust and low in complexity. Therefore, we may sacrifice the performance by using a model that may not fit the empirical data well but is simple to train and requires less human intervention.

1.1 Scope and Contributions

The scope of this thesis is limited to RSSI-based schemes using low-cost components. We assume that there is only one mobile transceiver and all the other radios are fixed and used as sensors. We bypass the problem of data association and assume that we can uniquely determine the source and destination of a transmission. We categorize state of the art fingerprinting methods into regression or classification. Furthermore, we use Bayesian filtering to improve the performances of the methods as well as potentially solving some of the deficiencies of the fingerprinting paradigm. Specifically, the main contributions are:

- A comparison between the Unscentend Kalman Filter (UKF) and the Particle Filter (PF), two suboptimal solutions for the Bayesian filter.

- An emphasis on the use of classification due to its simplicity and a comparison between three classifiers.

- A preliminary study on using a Maximum Likelihood (ML) scheme under the Bayesian filter framework in order to handle imperfectly trained and time-varying parameters.

1.2 Organization

The rest of this thesis is organized as follows: We present a brief overview on popular methods studied in the literature in Chapter 2. Probabilistic formulations can be used under the Bayesian filter framework introduced in Chapter 3. Chapter 4 shows some empirical data in two different setups, which serve as motivation for the following chapters. The next three chapters form the novel contributions of this thesis. Chapter 5 details a comparison between the UKF and PF for the regression approach. Chapter 6 goes over the classification approach and performs a comparison between three classifiers. Chapter 7 shows our preliminary work on combating imperfectly trained and time-varying parameters. Finally, Chapter 8 concludes this thesis.

Chapter 2

Background and Related Work

A rigorous Mathematical approach for the task of RSSI-based localization is to explicitly construct a function $\mathbf{y} = h(\mathbf{x})$, where $\mathbf{y}$ are the RSSI measurements and $\mathbf{x}$ are the position coordinates. This function is rarely invertible so the standard approach is to choose an estimate $\hat{\mathbf{x}}$ that minimizes $\|\mathbf{y} - h(\hat{\mathbf{x}})\|$ [6, 7]. This optimization problem may be substituted by less rigorous methods such as the bounding box [2]. Another approach is to construct a probability density $p(\mathbf{y}|\mathbf{x})$ and the estimate $\hat{\mathbf{x}}$ should maximize the probability $p(\mathbf{x}|\mathbf{y})$. It is very common to convert the first deterministic framework to the second probabilistic framework by including additive noise terms, e.g., $\mathbf{y} = h(\mathbf{x}) + \mathbf{w}$. However, we emphasize that that there are other methods which do not rely on constructing functions or probability densities.

This chapter reviews the literature on RSSI-based localization. Filtering is not considered for now. The use of filtering will be discussed in Chapter 3. The following briefly summarize the white box approach followed by the fingerprinting approach.

2.1 The White Box Paradigm

The following are two approaches that attempt to construct models from theoretical backgrounds. They can be said to be white box approaches.

2.1.1 Ray Tracing

Ray tracing is the most fundamental method as its only assumptions are laws of Physics. Starting at the transmitter, radio waves are modeled as rays and they are traced as they hit obstacles, experiencing multipath effects. [8, 9] are examples of simple ray tracing. [10] is an example of more sophisticated methods.

2.1.2 The Path Loss Model

In free space, using laws of Physics, it can be shown that RSSI decreases proportionally to the square of the separation distance between two radios [8]. In the special case of the 2-ray model, the two radios are assumed to be high above the ground and the only other propagation path other than the direct LOS path is the ground reflection. Using the small angle approximation and assuming the radios are placed sufficiently far apart, it can be shown that the rate of decay is proportional to the fourth power of the separation distance [8]. Formally, the signal strength y in dBmW is

$$y = \Gamma - 10\rho \log_{10} d, \tag{2.1}$$

where Γ is some additive constant, ρ is the path loss exponent ($\rho = 2$ for free space propagation and $\rho = 4$ for the 2-ray model) and d is the separation distance. This is a log-linear model as ρ plays the role of the slope and Γ plays the role of the bias.

It should be emphasized that free space propagation and the 2-ray model are both special cases. In real environments, it is very difficult to derive the path loss exponent or the bias. The classical approach is to take some measurements and attempt to adjust the parameters of the log-linear model such that the model fits the data. In addition, additive Gaussian noise is often included to model shadowing. Therefore, the standard path loss model is

$$y = \Gamma - 10\rho \log_{10} d + w, \tag{2.2}$$

where Γ and ρ are parameters of the setup, d is the separation distance and w is the zero-mean Gaussian noise [8, 11]. Extensive measurements have been conducted and it has been shown that higher values of ρ correspond to more absorptive environments and Γ may be a function of antenna gains, transmit power of the transmitter and frequency of transmissions [8].

Although the path loss model has been applied successfully in outdoor environments, its use in indoor environments is more limited. [10] uses ray tracing to show that it does not hold well in environments where reflections dominate. Since the approach of adjusting the parameters to fit the data is essentially regression, [12] uses the coefficient of determination, R^2, to quantify how good the fit is.

One possible extension to the standard path loss model is the divide and conqueror strategy. Instead of using a global model for all parts of the area, the area is divided into cells and each cell possesses its own unique path loss model. To distinguish this from the global path loss model, this is denoted the piecewise path loss model. It is reasonable to assume that the path loss exponent and the Gaussian variance are global but the key idea is that the biases are local. In [11], the authors use the terms additive floor and wall attenuation factors. For instance, two different cells may have different bias parameters because they have different wall attenuation factors. This approach has been recommended by the developers of RADAR [5].

2.2 The Fingerprinting Paradigm

In contrast to the white box paradigm, fingerprinting gives no regards to laws of Physics. It simply treats the problem as a black box. The only things that matter are the inputs and outputs. Using the language of Machine Learning [13, 14], this is treated as a supervised learning problem. The goal is to train a machine that learns the model and teach it how to performance inference.

Fingerprinting works by sampling [15]. First, RSSI measurements are taken at known locations. A *fingerprint* at a specific location simply consists of a vector of RSSI measurements (*instance*) and the the location information (*label*). The information can be continuous in terms of location coordinates or discrete. In the end, a *radiomap* or *database* is constructed with a number of these instance-label pairs. This is commonly called the offline training phase. Then, a model is fitted to the empirical data. If the labels applied are continuous, this is known as regression. If the labels applied are discrete, this is known as classification. Finally, in the online working or *validation* phase, when an instance originating from a unknown location arrives, the knowledge learned is used to estimate the label.

This powerful paradigm has two important assumptions. First, sampling must be done carefully in fine spatial intervals. Second, since the model learned is chosen to fit the offline database, it is assumed that conditions of the online working phase are the same as the conditions of the offline phase.

The vast majority of the literature choose to apply continuous labels, i.e., position coordinates. This amounts of a regression problem. Many regression techniques have been proposed and studied. In fact, the path loss model discussed in Section 2.1.2 can be thought of as a regression technique as empirical data is fitted to a log-linear model. Other examples include the weighted K-Nearest Neighbor (KNN) used in RADAR [5], regression trees [16], Artifical Neutral Networks (ANN) [17, 18], Support Vector Machine (SVM) regression [19] and probabilistic methods [20–22]. The accuracies of all the proposed methods plateau around 2m using realistic setups.

To our best knowledge, the literature on the classification approach is relatively sparse compared to regression. For most applications, it is enough to know if the target is in some specific region (e.g., bedroom or kitchen). In fact, if we are only interested in contextual information, then a coordinate obtained from a localization algorithm would have to be converted using a map. This is still the same fingerprinting paradigm but the algorithm estimates a region instead of an exact position coordinate. Let a cell be a small region of interests. The entire area is divided into a finite number of cells and they are labeled numerically. Now, the labels are cell numbers instead of coordinates. Although this can be viewed as a coarse version of regression, it has two major advantages: First, the training phase is vastly simplified because cell numbers replace position coordinates, which have to be obtained tediously. This requires less human intervention. Second, this leads to a classification problem and the techniques involved are often simpler and easier to implement[1]. [4] uses the SVM and the Linear Discriminant Analysis (LDA) to perform room-level localization. [19] uses SVM for both regression and classifica-

[1]The boundary between regression and classification is often blurred. For instance, a regression technique can easily be converted to a classifier by quantizing the final output. SVM is a native classifier but it can be modified to perform regression.

tion noting that it gives good classification results. [23] measures analog outputs from energy detectors of Ultra-Wideband (UWB) radios in a cell and models them as Gaussian distributions. The authors go into great details to justify the Gaussian model. Their main assumption is that the relevant impulse responses of the energy detectors are realizations of Gaussian processes. Although the work done is for UWB energy detectors, the same principle works for RSSI-based schemes using general radios. To our best knowledge, [24] is the earliest work modeling RSSI values within a cell as Gaussian variables. We refer to it as the Simple Gaussian Classifier (SGC) due to its astonishing simplicity.

The fingerprinting paradigm is powerful precisely because detailed ray tracing is infeasible in practice. For instance, [9] uses simple ray tracing but resorts to collecting measurements in order to estimate the reflection and absorption coefficients of obstacles. The path loss model is only valid for simple cases yet regression is used in order to fit measurements to the log-linear model. Nevertheless, we would like to point out that fingerprinting-based methods cannot handle theoretical questions such as the fundamental limits on the accuracies of algorithms or the sampling interval (in space and time) required. Furthermore, how to divide the area into cells is a difficult question. The standard approach using the fingerprinting paradigm is to proceed forward with an arbitrary scheme and see if it meets the performance requirements. To answer these theoretical questions, it is required to know the exact physical model, which can only be obtained via detailed ray tracing.

Chapter 3

Bayesian Filtering

In Chapter 2, many methods reviewed construct models in the form $p(\mathbf{y}|\mathbf{x})$. This is denoted the observation likelihood. The optimal estimate, in the Bayesian sense, should maximize the probability density

$$p(\mathbf{x}|\mathbf{y}) = \frac{p(\mathbf{y}|\mathbf{x})p(\mathbf{x})}{p(\mathbf{y})}. \tag{3.1}$$

Without filtering, i.e., each task of inference is only treated as an one-shot-in-time event, $p(\mathbf{x})$ is assumed to be uniform and this becomes maximizing the observation likelihood $p(\mathbf{y}|\mathbf{x})$. To our knowledge, using Bayes' rule to perform localization first appeared in [22]. If $\mathbf{x}$ is continuous, then this optimization problem is typically difficult and requires numerical solutions. However, if $\mathbf{x}$ comes from a discrete set, then a simple search can be used.

Location information is highly correlated in time. Intuitively, if the target is known to be at a specific location, it is highly likely to be in the vicinity of the same location at some later time. In the literature, it is extremely common to use the discrete-time Bayesian filter framework[1]. This approach is commonly named target tracking instead of static localization. Instead of assuming $p(\mathbf{x})$ is uniform, time correlation is considered. First, let $\mathbf{x}_{1:t} \triangleq [\mathbf{x}_1, \ldots, \mathbf{x}_t]$, where t is the time index. $\mathbf{x}_t$ is the unknown *state* and it lives in the *state space* of the framework. The *transition* or *maneuver* model $p(\mathbf{x}_t|\mathbf{x}_{t-1})$ describes how the target evolves in

[1]The introduction in [25] is an extremely good read.

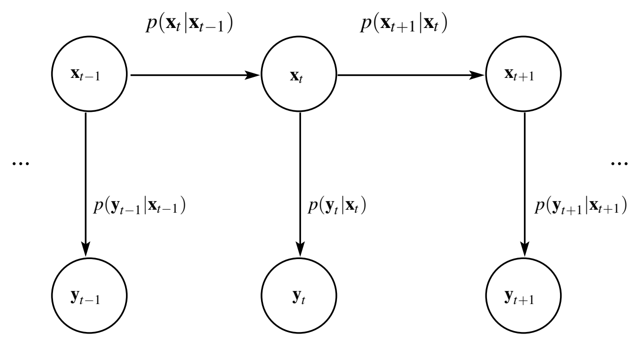

Figure 3.1: A Graphical Illustration of the Bayesian Filter

time in a Markovian fashion. Let $\mathbf{y}_{1:t} \triangleq [\mathbf{y}_1, \ldots, \mathbf{y}_t]$, where t is the time index. The *observation* model is the constructed model $p(\mathbf{y}_t|\mathbf{x}_t)$. These two densities form the basis of the Bayesian filter

$$\begin{array}{l} p(\mathbf{x}_t|\mathbf{x}_{t-1}) \\ p(\mathbf{y}_t|\mathbf{x}_t) \end{array}. \tag{3.2}$$

This can be illustrated by the directed graph in Figure 3.1. The standard choice for the initial density $p(\mathbf{x}_0)$ is the uniform distribution if the initial state of the target is unknown.

The Bayesian filter consists of two stages, prediction[2]

$$p(\mathbf{x}_t|\mathbf{y}_{1:t-1}) = \int p(\mathbf{x}_t|\mathbf{x}_{t-1}) p(\mathbf{x}_{t-1}|\mathbf{y}_{1:t-1}) d\mathbf{x}_{t-1} \tag{3.3}$$

and update

$$p(\mathbf{x}_t|\mathbf{y}_{1:t}) = \frac{p(\mathbf{y}_t|\mathbf{x}_t) p(\mathbf{x}_t|\mathbf{y}_{1:t-1})}{p(\mathbf{y}_t|\mathbf{y}_{1:t-1})}. \tag{3.4}$$

The normalization factor in the denominator can be calculated using

$$p(\mathbf{y}_t|\mathbf{y}_{1:t-1}) = \int p(\mathbf{y}_t|\mathbf{x}_t) p(\mathbf{x}_t|\mathbf{y}_{1:t-1}) d\mathbf{x}_t. \tag{3.5}$$

This steps ensures that

$$\int p(\mathbf{x}_t|\mathbf{y}_{1:t}) d\mathbf{x}_t = 1. \tag{3.6}$$

[2]This is also known as the Chapman-Kolmogorov equation.

It is common to combine the two steps into one, i.e.,

$$p(\mathbf{x}_t|\mathbf{y}_{1:t}) = cp(\mathbf{y}_t|\mathbf{x}_t)\int p(\mathbf{x}_t|\mathbf{x}_{t-1})p(\mathbf{x}_{t-1}|\mathbf{y}_{1:t-1})d\mathbf{x}_{t-1}, \tag{3.7}$$

where c is the normalization constant.

The filtered *posterior* density $p(\mathbf{x}_t|\mathbf{y}_{1:t})$ is conditioned on the history of observations $\mathbf{y}_{1:t}$. The framework is recursive. Starting at some initial distribution $p(\mathbf{x}_0)$, the framework moves forward in time and calculate $p(\mathbf{x}_1|\mathbf{y}_1)$, $p(\mathbf{x}_2|\mathbf{y}_{1:2})$, ... every time slot. As pointed out in [26], this deceptively simple framework has one major problem. These integral equations have no solutions except in two limited cases. The two exceptions are:

- If the transition and observation models written as probability densities are Gaussians, then the Kalman Filter (KF) [27] is the optimal solution. All the probability densities involved are Gaussians.

- If the state space is finite, i.e., $\mathbf{x}_t$ comes from a finite set, then the integrals in Equation 3.3 and Equation 3.4 are converted to sums because the probability densities involved are discrete probability mass functions. Normalization becomes trivial and this is called the Hidden Markov Model (HMM) filter [28]. This is the case for the classification approach since the number of cells is finite!

In all other cases, approximations must be used. This is another advantage of classification compared to regression because there is a closed-form solution.

Summarizing Chapter 2 and this chapter, probabilistic formulations are the most popular in the literature and they can be used in the Bayesian filter framework. Deterministic methods can be converted to probabilistic ones. The ones that cannot be converted typically use simple time averaging. For instance, [4] uses the SVM, which is a deterministic classifier. The authors use simple time averaging to improve the results.

Chapter 4

Two Examples of Applying the Path Loss Model

As discussed in Section 2.1.2, the path loss model is often cited in the literature. For localization purposes, its use is often said to be unreliable and unpredictable [2]. Although it can be backed up by theoretical analysis in specific cases, its use in real environments is backed up by the fingerprinting paradigm as discussed in Section 2.2. In this chapter, we take a closer look into this issue and demonstrate how well the path loss model works in two different setups. For each setup, the area is deemed geometrically homogeneous such that it makes no sense to further divide the area into smaller cells. Therefore, the global path loss model is applied. This chapter serves as motivation for later chapters.

4.1 The First Setup

We take two ZigBee radios and perform a simple experiment at room Kaiser 2020 at University of British Columbia (UBC). The modules used are Rabbit 4510W kits[1]. According to the datasheet, the frequency of transmissions is 2.4GHz. One radio is fixed and the other one is placed at various distances from the fixed radio.

Figure 4.1 and Figure 4.2 show the setup. Because there is a LOS between the two radios and there are no major obstacles, the environment is close to ideal and

[1] www.rabbit.com

Figure 4.1: Room Kaiser 2020

the path loss model fits the data very well. Figure 4.3 shows the measured values and the fitted model. The fitted model is

$$\hat{y} = -18.3912 \log_{10} d - 50.5350, \tag{4.1}$$

where $\hat{y}$ is the signal strength and d is the separation distance. The R^2 of the fit is 0.6826, which is not perfect but good enough.

Now, we take a look at the residuals of the fit, i.e., a residual is

$$r_i = \hat{y}_i - y_i, \tag{4.2}$$

where $\hat{y}_i$ is the predicted value using the model from a specific distance and y_i is a measurement from that distance. According to the standard path loss model [8, 11], the residuals are realizations of a Gaussian process. This cannot be confirmed by applying the chi-squared test at 95% confidence. Figure 4.4 shows the residuals. Although the figures do not seem perfectly Gaussian, they are close. We cannot claim that our simple experiment is conclusive and we speculate that the distribution will look more Gaussian with a better experimental setup. This serves

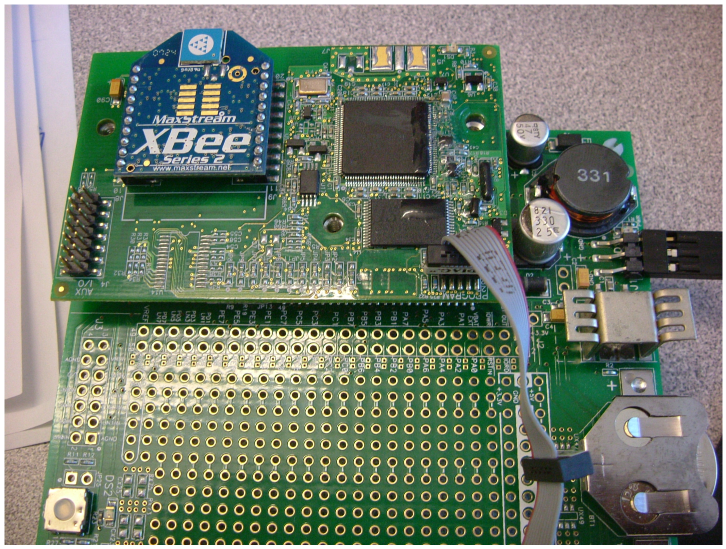

Figure 4.2: A ZigBee Radio

as motivation for the work in Chapter 5.

4.2 The Second Setup

The test area is a big conference table and we place the target at every possible location on this table. The table sits in a conference room and it is spaced at least one meter from the walls and corners. The mobile target uses a small loop antenna transmitting at 433MHz. One receiver, which uses MAX1473[2] modules, receives the transmissions using $\lambda/4$ antennas. The receiver is mounted high above the ground on one of the walls. Figure 4.5 shows the setup.

The fitted model is

$$\hat{y} = -2.2215 \log_{10} d - 74.4224, \tag{4.3}$$

where $\hat{y}$ is the signal strength and d is the separation distance. The R^2 of the fit is 0.0000, which literally says that the fit is useless. Figure 4.6 shows that the histogram of residuals of the fitted model looks the same as the histogram of the

[2] www.maxim-ic.com

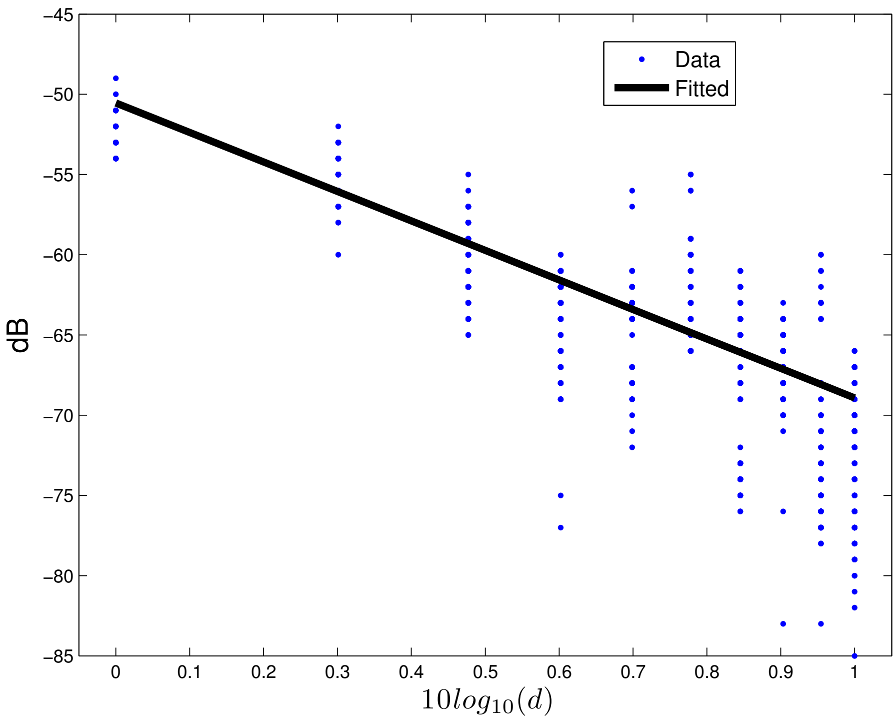

Figure 4.3: RSSI Values and the Fitted Model for the First Setup

RSSI values themselves, i.e., we can just use this constant model

$$\hat{y} = -74.4224. \tag{4.4}$$

This is easily explained by the fact that the logarithmic term ceases to pose any effect for realistic values of the path loss exponent, i.e., if variations in $\log_{10} d$ are small. Furthermore, it will be impossible to determine the exact location coordinate since any coordinate on the desk returns the same RSSI value according to the model. This is exactly the case of the piecewise path loss model. Since we divide the area into multiple small cells, the path loss exponent is irrelevant and can be set to zero if the cells are small enough. In this experiment, our test area is an isolated table in a room and it is evidently small enough. In addition, we note that the histograms in Figure 4.6 do resemble Gaussian distributions somewhat although neither passes the chi-square test at 95%. This leads to the revelation

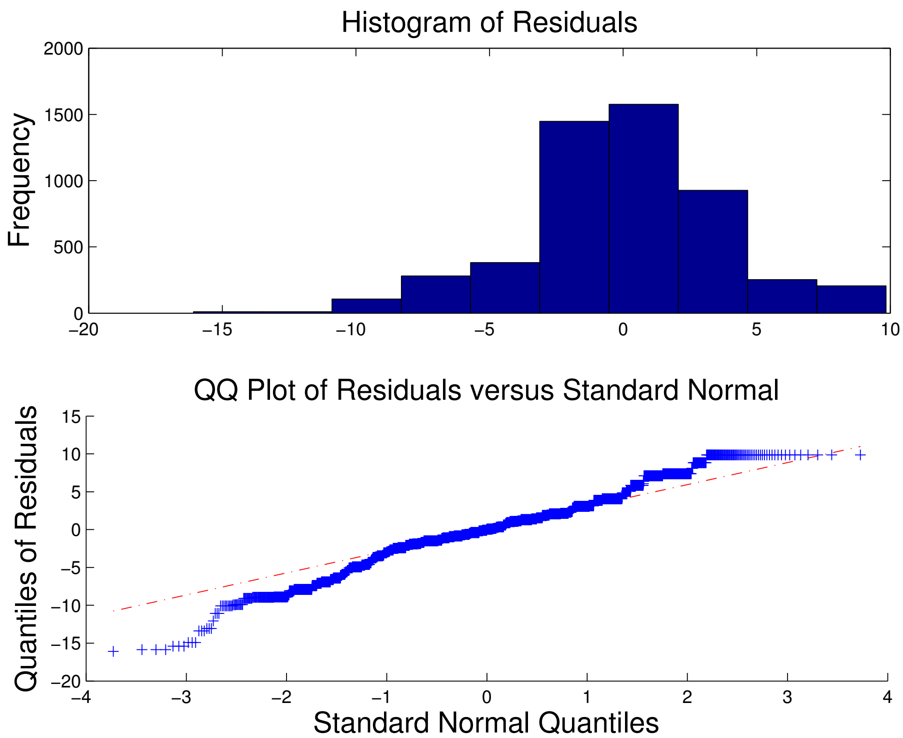

Figure 4.4: Distribution of the Residuals for the First Setup

that, within a small cell, RSSI values can be modeled as Gaussian disregarding the precise location coordinate of the target. This motivation leads to our work in Chapter 6.

Of course, this chapter has assumed that the data actually follows the path loss model. [10] shows that the path loss model fails in environments where multipath effects dominate, i.e., if there are obstacles close to the radios. However, if we relax the definition of 'good fit', the path loss model serves a purpose because it is simple and easy to train via regression. This is a sharp contrast to regression schemes such as the ANN. Furthermore, the slope of the model, the path loss exponent, has some physical meaning as higher values indicate denser obstructions. This is a sharp contrast to other regression techniques whose parameter have no meaning at all. Therefore, there is a benefit of using the path loss model at a cost of obtaining a worse fit. The rest of this thesis will use the path loss model and variations of it as

Figure 4.5: The Conference Table

the basis for the propagation model.

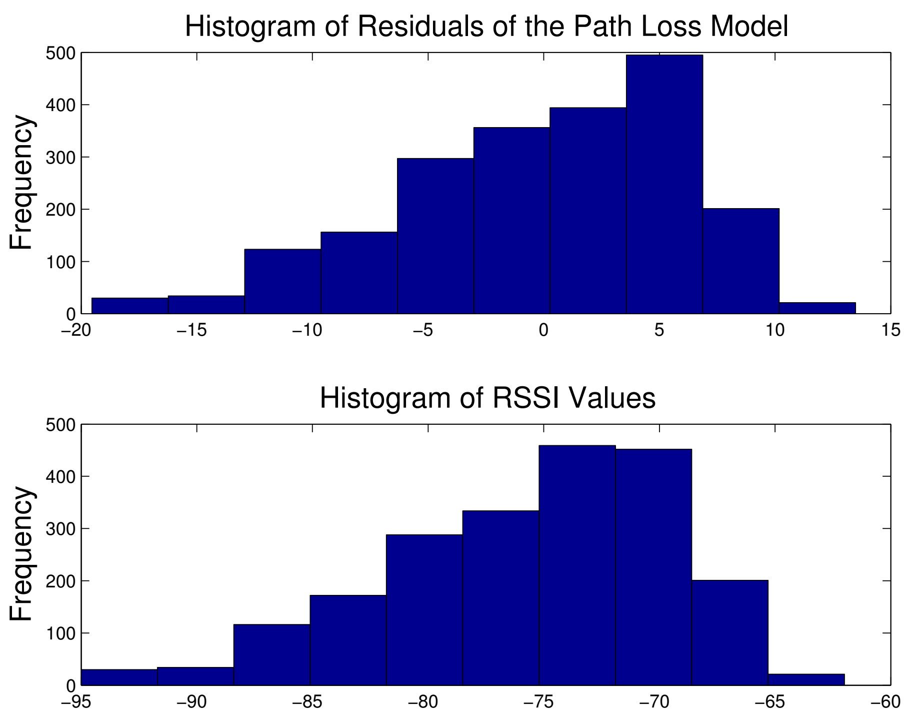

Figure 4.6: The Residuals and the RSSI Values for the Second Setup

Chapter 5

Unscented Kalman Filtering versus Particle Filtering

As discussed in Chapter 3, the HMM filter [28] is used for the classification approach. The state space is finite so it is the exact and optimal solution. The case for the regression approach is not so straightforward. Since the state space is continuous, no exact solution is known except one special case. If the transition and observation models are both linear and all the noise processes are additive Gaussian, then the famous KF [27] is the optimal solution. However, those conditions are rarely met in practice. In particular, the observation model cannot be expected to be linear.

In cases where the strict conditions are violated but not too severely, one possible solution is the classical Extended Kalman Filter (EKF) [27], which relies on linearizing the transition and observation models via Taylor series expansion. However, the UKF, a newer variation of the KF, is found to perform better because it can capture more terms of the Taylor series expansion of a nonlinear function [29]. In particular, [30] shows that the UKF performs better than the EKF for TOA tracking. The UKF uses several deterministic 'sigma points' to capture the transformation of a probability density though a nonlinearity. Another competitive solution to deal with nonlinear conditions is the PF, which solves the general Bayesian filter by simulations. A practical and robust implementation of the PF is the Sampling Importance Resampling (SIR)-Particle Filter (PF) [25, 26].

In the literature, authors have used variations of the KF (EKF [31] and UKF [17]) and the PF [32] for RSSI-based tracking using the regression approach. However, a head-to-head comparison of these techniques has not been made. This chapter addresses this issue by comparing and contrasting UKF and PF in terms of their accuracies and consistencies.

The following sections describe the assumptions, summarize the two techniques and present the conclusion based on simulations as well as experimental results.

5.1 System Model

This section presents a popular transition model [33] and a reasonable observation model based on discussions from Chapter 2 and Chapter 4. The goal is to track a transceiver moving in a bounded two-dimensional region. N sensors are placed at arbitrary but known locations in this area. The transceiver broadcasts to all sensors every T seconds. Each sensor listens, evaluates the signal strength from the target and forwards all the data to a central Fusion Center (FC).

The unknown state vector of the target at time t is defined as

$$\mathbf{x}_t = \left[p_t^{(1)}\, v_t^{(1)}\, p_t^{(2)}\, v_t^{(2)}\right]^T, \tag{5.1}$$

where the variables are the position coordinates and velocity components with respect to some fixed two-dimensional coordinate system. The transition model is

$$\mathbf{x}_t = f\left(\mathbf{x}_{t-1}, \mathbf{u}_{t-1}\right) = \mathbf{A}\mathbf{x}_{t-1} + \mathbf{B}\mathbf{u}_{t-1}, \tag{5.2}$$

where

$$\mathbf{A} = \begin{bmatrix} 1 & T & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & T \\ 0 & 0 & 0 & 1 \end{bmatrix} \\ \mathbf{B} = \begin{bmatrix} \frac{T^2}{2} & 0 \\ T & 0 \\ 0 & \frac{T^2}{2} \\ 0 & T \end{bmatrix}. \tag{5.3}$$

This transition model arises from discretizing Newton's laws of motion [33]. The first term corresponds to inertia and the second term corresponds to accelerations due to random maneuvers. The $\mathbf{Bu}_{t-1}$ term is a stochastic process of accelerations assumed to be i.i.d. $\mathcal{N}(\mathbf{0}, \mathbf{Q})$, where $\mathbf{Q} = \sigma_u^2 \mathbf{I}$[1].

As for the observation model, there is no universally accepted and applicable model as discussed in Chapter 2. A reasonable one is chosen and the implications will be discussed after it is studied. The choice is the piecewise path loss model as it is generally applicable in indoor environments. As discussed in Section 2.1.2, the path loss model is characterized by the path loss exponent, the bias and the noise. It is reasonable to assume that the path loss exponent and the noise distributions are global and one can apply the same parameters for all sensors in the area. However, it is not reasonable to apply a uniform bias for all, because that would imply a monotonic environment. Therefore, we apply different biases according to some cell division scheme.

Formally, the signal strength in dBmW of a transmission from the target heard at sensor n at time t is

$$y_{n,t} = \Gamma_n(\psi(\mathbf{x}_t)) - 10\rho \log_{10} \|\psi(\mathbf{x}_t) - \mathbf{r}_n\| + w_{n,t}. \tag{5.4}$$

Here, $\psi(\mathbf{x}_t) = \left[p_t^{(1)} p_t^{(2)}\right]^T$ is a function that simply returns the position part of the state vector. It describes the cell division scheme. $\Gamma_n(\cdot)$ and ρ are the familiar

[1]Because the system is discrete in time, the actual physical accelerations are assumed to be piecewise constant, which is a reasonable assumption provided T is small relative to the dynamics of the target.

bias and path loss exponent. $\mathbf{r}_n$ is the known location of sensor n. As named in Section 2.1.2, this is the piecewise path loss model instead of the global one. Finally, $w_{n,k}$ is the familiar i.i.d. Gaussian for all n,t, distributed according to $\mathcal{N}(0,\sigma_w^2)$. The noise models shadowing and any other factors not accounted in the model. Thus, there are N RSSI observations at each time t. Let us collect these observations in a vector $\mathbf{y}_t$. Similarly, let the observation noise terms be collected in a corresponding vector $\mathbf{w}_t$. (Let us name the covariance of it $\mathbf{R}$.) One can succinctly rewrite the observation equation as

$$\mathbf{y}_t = h(\mathbf{x}_t, \mathbf{w}_t), \tag{5.5}$$

where $h(\cdot)$ is implicitly defined according to Equation 5.4. Together, Equation 5.2 and Equation 5.5 form the basis of the Bayesian filter, which will be solved by either the UKF or the PF next.

5.2 The Unscented Kalman Filter (UKF)

The UKF is a modified version of the classical KF. In the original KF, all the probability densities involved in the calculations are Gaussians because everything is linear. Thus, they are completely characterized by means and covariances. However, transforming a Gaussian density through a nonlinear function results in a non-Gaussian product. If the nonlineararity is not too severe, the UKF assumes that the resulting density can be approximated by a Gaussian. The UKF uses deterministic 'sigma points' to deal with the nonlinear transformation. It can capture up to the third order term of the Taylor series expansion of the nonlinear transformation [29].

First, let us discuss the process to create sigma points. We start with a L-dimensional distribution with mean $\mathbf{m}$ and covariance matrix $\mathbf{S}$. We have parameters α, β, κ and

$$\lambda = \alpha^2(L+\kappa) - L. \tag{5.6}$$

The set of sigma points $\left\{\varsigma^i\right\}_{i=0}^{2L}$ are

$$\varsigma^0 = \mathbf{m} \tag{5.7}$$

$$\varsigma^i = \mathbf{m} + \left(\sqrt{(L+\lambda)\mathbf{S}}\right)_i \;, i = 1 \ldots L \tag{5.8}$$

$$\varsigma^i = \mathbf{m} - \left(\sqrt{(L+\lambda)\mathbf{S}}\right)_i \;, i = L+1 \ldots 2L. \tag{5.9}$$

$\left(\sqrt{(L+\lambda)\mathbf{S}}\right)_i$ is i^{th} column of the square root of the matrix $(L+\lambda)\mathbf{S}$ using the definition $\mathbf{B} = \mathbf{A}\mathbf{A}^T$ if $\mathbf{A}$ is the square root matrix of $\mathbf{B}$. It is common to use the Cholesky decomposition for this calculation. The weights for these sigma points are

$$W_s^0 = \frac{\lambda}{L+\lambda} \tag{5.10}$$

$$W_c^0 = \frac{\lambda}{L+\lambda} + (1 - \alpha^2 + \beta) \tag{5.11}$$

$$W_c^i = W_s^i = \frac{1}{2(L+\lambda)} \;, i \neq 0. \tag{5.12}$$

These weights are not used for any stochastic purposes and do not sum up to one necessarily. Typical values for the parameters α, β and κ are 10^{-3}, 2 and 0 respectively.

The UKF retains the same structure of the original KF [27]. The prediction and update steps are done in the same manner except that the sigma points are used to deal with the nonlinearities and calculate the Kalman gain. The transition model Equation 5.2 is linear so the prediction step can be done using the original KF. However, in the interest of generality, the complete UKF algorithm is shown in Algorithm 1. Regarding the augmentation process, some authors have omitted it but the results are not the same as shown in [34].

5.3 The Particle Filter (PF)

The PF [25, 26] is a simulation-based method for solving the general Bayesian filter problem. Since it poses no assumptions on linearities of the problem, it is able to solve any Bayesian filter. The idea is simple. One simply simulates candidates, runs them through the transition model Equation 5.2 and weights them according to the observation model Equation 5.5. The estimate of the unknown state is a weighted average of these candidates. As the number of candidates increases, the performance approaches that of the optimal filter.

Algorithm 1 For each time slot t

Start with previous time slot's estimated state $\hat{\mathbf{x}}_{t-1}$ and covariance $\mathbf{P}_{t-1}$.
Augment with the Gaussian processes in Equation 5.2 and Equation 5.5, i.e.,

$$\hat{\mathbf{x}}^a_{t-1} = \left[\hat{\mathbf{x}}^T_{t-1}\ E[\mathbf{u}^T]\ E[\mathbf{w}^T]\right]^T$$

$$\mathbf{P}^a_{t-1} = \begin{bmatrix} \mathbf{P}_{t-1} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{Q} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{R} \end{bmatrix}.$$

Create sigma points from this augmented result. Denote the sigma points χ^i_{t-1}. Let us distinguish the state part $\chi^i_{x,t-1}$, the $\mathbf{u}_t$ part $\chi^i_{u,t-1}$ and the the $\mathbf{w}_t$ part $\chi^i_{w,t-1}$.

Prediction
$\chi^i_{x,t|t-1} = f\left(\chi^i_{x,t-1}, \chi^i_{u,t-1}\right)$
$\hat{\mathbf{x}}_{t|t-1} = \sum_{i=0}^{2L} w^i_s \chi^i_{x,t|t-1}$
$\mathbf{P}_{t|t-1} = \sum_{i=0}^{2L} w^i_c \left[\chi^i_{x,t|t-1} - \hat{\mathbf{x}}_{t|t-1}\right]\left[\chi^i_{x,t|t-1} - \hat{\mathbf{x}}_{t|t-1}\right]^T$

Update
$\gamma^i_t = h\left(\chi^i_{x,t|t-1}, \chi^i_{w,t-1}\right)$
$\hat{\mathbf{y}}_t = \sum_{i=0}^{2L} w^i_s \gamma^i_t$

Kalman Gain
$\mathbf{P}_{yy} = \sum_{i=0}^{2L} w^i_c \left[\gamma^i_t - \hat{\mathbf{y}}_t\right]\left[\gamma^i_t - \hat{\mathbf{y}}_t\right]^T$
$\mathbf{P}_{xy} = \sum_{i=0}^{2L} w^i_c \left[\chi^i_{x,t|t-1} - \hat{\mathbf{x}}_{t|t-1}\right]\left[\gamma^i_t - \hat{\mathbf{y}}_t\right]^T$
$\mathbf{K}_t = \mathbf{P}_{xy}\mathbf{P}^{-1}_{yy}$

Current Estimate
$\hat{\mathbf{x}}_t = \hat{\mathbf{x}}_{t|t-1} + \mathbf{K}_t(\mathbf{y}_t - \hat{\mathbf{y}}_t)$
$\mathbf{P}_t = \mathbf{P}_{t|t-1} - \mathbf{K}_t\mathbf{P}_{yy}\mathbf{K}^T_t$

Formally, let us denote P candidates of the true state vector $\left\{\mathbf{x}^i_t\right\}^P_{i=1}$. They have the same dimensionality and components as the true state vector $\mathbf{x}_t$ at time t. Each candidate is called a particle with weight w^i_t. Together, these particles approximate the posterior density $p(\mathbf{x}_t|\mathbf{y}_{1:t})$. However, it is not easy to sample from the posterior directly. Instead, the standard is to draw particles from $p(\mathbf{x}_{1:t}|\mathbf{y}_{1:t})$. We draw samples from an arbitrary proposal density q and weight the particles according to the importance ratio

$$w_t(\mathbf{x}_{1:t}) = \frac{p(\mathbf{x}_{1:t}|\mathbf{y}_{1:t})}{q(\mathbf{x}_{1:t}|\mathbf{y}_{1:t})}. \tag{5.13}$$

The standard is to assume the proposal density can be constructed recursively in time, i.e.,

$$q(\mathbf{x}_{1:t}|\mathbf{y}_{1:t}) = q(\mathbf{x}_{1:t-1}|\mathbf{y}_{1:t-1})q(\mathbf{x}_t|\mathbf{y}_t, \mathbf{x}_{t-1}). \tag{5.14}$$

Therefore, the importance weights can also be updated recursively in time, i.e.,

$$w_t(\mathbf{x}_{1:t}) = \frac{p(\mathbf{x}_{1:t}|\mathbf{y}_{1:t})}{p(\mathbf{x}_{1:t-1}|\mathbf{y}_{1:t-1})q(\mathbf{x}_t|\mathbf{y}_t, \mathbf{x}_{t-1})} w_{t-1}(\mathbf{x}_{1:t-1}). \tag{5.15}$$

Furthermore, $p(\mathbf{x}_{1:t}|\mathbf{y}_{1:t})$ can be factored into

$$p(\mathbf{x}_{1:t}|\mathbf{y}_{1:t}) \propto p(\mathbf{x}_{1:t}, \mathbf{y}_{1:t}) = \prod_{k=1}^{t} p(\mathbf{y}_k|\mathbf{x}_k)p(\mathbf{x}_k|\mathbf{x}_{k-1}) \tag{5.16}$$

because of the conditional independence property of the Bayesian filter. This allows us to only keep the most current samples $\{\mathbf{x}_t^i\}_{i=1}^{P}$ at time t instead of keeping the entire history of samples, i.e., Equation 5.15 becomes

$$\widetilde{w}_t^i \propto \frac{p(\mathbf{y}_t|\mathbf{x}_t^i)p(\mathbf{x}_t^i|\mathbf{x}_{t-1}^i)}{q(\mathbf{x}_t^i|\mathbf{y}_t, \mathbf{x}_{t-1}^i)} \widetilde{w}_{t-1}^i. \tag{5.17}$$

Since we only have a finite number of these particles, the unnormalized weights can be normalized by

$$w_t^i = \frac{\widetilde{w}_t^i}{\sum_{j=1}^{P} \widetilde{w}_t^j}. \tag{5.18}$$

Finally, this achieves the goal of using a set of particles $\{\mathbf{x}_t^i\}_{i=1}^{P}$ and associated weights $\{w_t^i\}_{i=1}^{P}$ to approximate the posterior density. This scheme is known as *sequential importance sampling*.

For this chapter, the transition model Equation 5.2 consists of Gaussians and they are extremely easy to generate. Therefore, the transition model Equation 5.2

is used as the proposal density and the algorithm is further simplified, i.e.,

$$q(\mathbf{x}_t^i|\mathbf{y}_t, \mathbf{x}_{t-1}^i) = p(\mathbf{x}_t^i|\mathbf{x}_{t-1}^i) \tag{5.19}$$

and

$$\widetilde{w}_t^i \propto p(\mathbf{y}_t|\mathbf{x}_t^i)\widetilde{w}_{t-1}^i. \tag{5.20}$$

Owning to Equation 5.4, the observation likelihood between the target and a single sensor n is

$$p(y_{n,t}|\mathbf{x}_t^i) = \frac{1}{\sqrt{2\pi\sigma_w^2}} \exp\left(\frac{-(y_{n,t} - \Gamma_n(\psi(\mathbf{x}_t^i)) + 10\rho\log_{10}\|\psi(\mathbf{x}_t^i) - \mathbf{r}_n\|)^2}{2\sigma_w^2}\right). \tag{5.21}$$

Therefore, the total observation likelihood given the total observation vector $\mathbf{y}_t$ is

$$p(\mathbf{y}_t|\mathbf{x}_t^i) = \prod_{n=1}^{n=N} p(y_{n,t}|\mathbf{x}_t^i). \tag{5.22}$$

It is well known that the weights will be concentrated on a small number of particles using this scheme [25, 26]. This is statistically harmful as the other particles have negligible weights and the number of useful particles is reduced. The Sampling Importance Resampling (SIR) scheme combats this *degeneracy* by resampling, i.e., discarding the negligible particles. How often resampling takes place is a matter of design choice. We simply resample at every time slot. Therefore, all the particles will have the same weight after each iteration of the algorithm. Unfortunately, in scenarios where the process variances in the transition model Equation 5.2 are small, this leads to *sample impoverishment*, where all the particles collapse to near-identical states. This work opts for a simple solution that uses larger variances for drawing from the transition model Equation 5.2 instead of the true values. This will slightly alter behavior of the standard SIR-PF.

A summary of the PF is presented in Algorithm 2. The resampling step is described in Algorithm 3.

Algorithm 2 For each time slot t

Start with previous time slot's particles and weights, $\{\mathbf{x}_{t-1}^i\}_{i=1}^P$ and $\{w_{t-1}^i\}_{i=1}^P$.

Draw New Particles
Draw P samples of the Gaussian process in Equation 5.2. Denoted each sample $\mathbf{u}_{t-1}^i$.
$\mathbf{x}_t^i = f\left(\mathbf{x}_{t-1}^i, \mathbf{u}_{t-1}^i\right)$

Weight
$\widetilde{w}_t^i = w_{t-1}^i p(\mathbf{y}_t|\mathbf{x}_t^i)$, where the likelihood is given by Equation 5.21 and Equation 5.22.
Normalize the weights using $w_t^i = \widetilde{w}_t^i / \sum_{j=1}^P \widetilde{w}_t^j$.

Current Estimate
$\hat{\mathbf{x}}_t = \sum_{i=1}^P w_t^i \mathbf{x}_t^i$

Resample using Algorithm 3 if necessary.

Algorithm 3 $(\mathbf{x}_t^j, w_t^j) = \text{RESAMPLE}(\mathbf{x}_t^i, w_t^i)$

Construct the cumulative probabilities $\{c_p\}_{p=1}^{p=P}$ of w_t^i.

for $j = 1 \cdots P$ **do**
Generate a uniform random variable $u \sim U[0,1]$.

Find the smallest p such that $u \leq c_p$.

Set $\mathbf{x}_t^j = \mathbf{x}_t^p$ and $w_t^j = 1/P$, i.e., all particles have the same weight.
end for

5.4 Simulations

Simulations are carried out in MATLAB to compare the UKF against the SIR-PF for the RSSI-based tracking problem described in Section 5.1. Target trajectories are generated using the system model and both filters are used to localize the target. The performance metric is the Mean Squared Error (MSE) between the true position

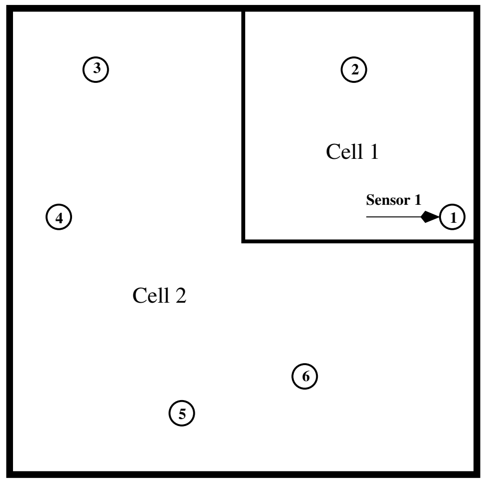

Figure 5.1: The Floor Plan for Simulations

coordinates and the estimates, i.e.,

$$\Delta^2 = E\left[\|\psi(\mathbf{x}_t) - \psi(\hat{\mathbf{x}}_t)\|^2\right]. \tag{5.23}$$

The simulation MSE is obtained by

$$\frac{1}{S}\sum_{t=1}^{S}\|\psi(\mathbf{x}_t) - \psi(\hat{\mathbf{x}}_t)\|^2, \tag{5.24}$$

where S is the length of the simulation.

The area is a 20m by 20m room divided into two cells. $N = 6$ sensors are placed at $[9\ 1]^T, [5\ 7]^T, [-7\ 7]^T, [-8\ -1]^T, [-2\ -7]^T$ and $[3\ -6]^T$, as depicted in Figure 5.1. Let $l = 1$ denote if the target is in the first cell and let $l = 2$ denote if the target is in the second cell. The global noise variance for the observation model Equation 5.4 is $\sigma_w^2 = 2$. The global path loss exponent is $\rho = 2$. The cells are used to define the $\Gamma_n(\cdot)$ biases in the observation model Equation 5.4. The mobile target, which starts at rest at the center of the room, is tracked as the it pings the sensors every $T = 1$ second. The variance for the transition model Equation 5.2 is $\sigma_u^2 = 10^{-\frac{25}{10}} \approx 0.0032$. The following results are averaged over 10^5 time slots. In order to reduce the number of particles required and to avoid sample impoverishment, the PF resamples at every time slot and it uses a variance that is 100 times larger than the true value for drawing particles. Because the number of particles required is not known and must be determined empirically, the number of particles is varied until the performance ceases to increase. The number of sigma points required for the UKF is $2L+1=25$, where L is the dimension of the augmented state space. The following two tests demonstrate how the behavior of the $\Gamma_n(\cdot)$ terms affects the performances of the filters.

5.4.1 Uniform Radio Environment

For this test, all the $\Gamma_n(\cdot)$ terms are set to zero, i.e., the radio environment is uniform and one global path loss model applies to all. The area is essentially free space.

Table 5.1 shows the MSE of the filters in terms of squared meters. The number of particles required for the PF is larger than the 25 sigma points required for the UKF. While the filters are different and the numbers are not directly comparable, this is a rough estimate indicating that the UKF is less computationally intensive. Since the transition model Equation 5.2 is linear, the only nonlinearity in the system is the path loss observation model Equation 5.4 and the UKF is able to handle it. The UKF is the better solution in this case due to lower computation costs. In terms of the finer statistics, the UKF is less consistent compared to the PF counterparts given sufficient number of particles. The 5-th and 95-th percentiles of the UKF are tighter but the variance is larger. This implies that the near-linearity assumption imposed by the UKF is correct the vast majority of the time but it fails and introduces large errors beyond the 95-th percentile occasionally. Finally, the PF is not able to reach the performance of the UKF despite a large number of particles because of the resampling scheme and the proposal density does not use the true value of the variance.

Table 5.1: Target Tracking Results in a Uniform Radio Environment

Mean Squared Error (m^2) over Time					
Filter	Number of Particles	Mean	Variance	5-th percentile	95-th percentile
PF	25	2.2313	210.1638	0.0357	6.0137
PF	50	1.2452	46.4644	0.0311	3.5388
PF	75	0.9318	13.1694	0.0292	3.0309
PF	100	0.9902	38.1152	0.0291	2.8899
PF	125	0.8034	1.8729	0.0289	2.7679
PF	150	0.7945	1.6093	0.0285	2.7339
PF	175	0.7735	1.1036	0.0287	2.6958
PF	200	0.7663	1.0823	0.0290	2.6695
PF	225	0.7617	1.0594	0.0285	2.6534
PF	250	0.7577	1.0515	0.0284	2.6278
UKF		0.6075	10.7986	0.0116	1.7033

5.4.2 Diverse Radio Environment

For this test, the $\Gamma_n(\cdot)$ biases are varied, simulating a nonuniform environment, i.e.,

$$\begin{aligned} \Gamma_{n=1\ldots2}(l=1) &= 0 \\ \Gamma_{n=3}(l=1) &= -15 \\ \Gamma_{n=4}(l=1) &= -30 \\ \Gamma_{n=5}(l=1) &= -30 \\ \Gamma_{n=6}(l=1) &= -15 \\ \Gamma_{n=1}(l=2) &= -20 \\ \Gamma_{n=2}(l=2) &= -20 \\ \Gamma_{n=3\ldots6}(l=2) &= 0 \end{aligned}. \tag{5.25}$$

This is a 'switching' behavior and it leads to multimodalness in the system.

Table 5.2 shows the MSE results. The nonuniform terms severely degrade the performance of the UKF to the point that the UKF basically fails. This is because the UKF imposes Gaussians for the densities required for the Bayesian filter. A Gaussian density is unimodal and thus it cannot handle the new multimodal system.

Table 5.2: Target Tracking Results in a Diverse Radio Environment

		Mean Squared Error (m^2) over Time			
Filter	Number of Particles	Mean	Variance	5-th percentile	95-th percentile
PF	25	3.4454	783.1535	0.0365	6.8851
PF	50	1.0531	7.1887	0.0300	3.6132
PF	75	0.8970	3.0619	0.0291	3.0880
PF	100	0.8553	3.2835	0.0288	2.8842
PF	125	0.8096	1.8312	0.0280	2.7910
PF	150	0.7987	2.0580	0.0278	2.7404
PF	175	0.7772	1.8331	0.0278	2.6760
PF	200	0.7646	1.1146	0.0275	2.6573
PF	225	0.7567	1.0768	0.0284	2.6289
PF	250	0.7547	1.0828	0.0280	2.6259
UKF		7.4229	1313.1010	0.0130	36.0688

For instance, in Figure 5.2, we show the distribution of the first position coordinate part of the particles at some time index for the case of 300 particles. Since we do resampling at every time step, all the weights are equal and the histogram resembles the true posterior density $p(\mathbf{x}_t|\mathbf{y}_{1:t})$ if the number of particles is sufficient. As shown, this is clearly multimodal and the UKF is not equipped to deal with it. Obviously, the level of uniformness in the radio environment affects how badly the UKF performs. However, the PF has the same performance and consistency because it does not assume any linearity and the densities of the Bayesian filter are not fitted to unimodal Gaussians. Therefore, the PF is much more robust and insensitive to the fine details of the environment and scenario.

5.5 Experimental Results

This section presents experimental results conducted in an office, which is characterized by diverse radio characteristics. The setup is a wireless network of seven ZigBee radios in room Kaiser 4090 at UBC. The modules used are the same radios used in Section 4.1. One radio is designated as the target, which broadcasts to other radios every $T = 1.5$ seconds. Five radios are used as sensors. The last

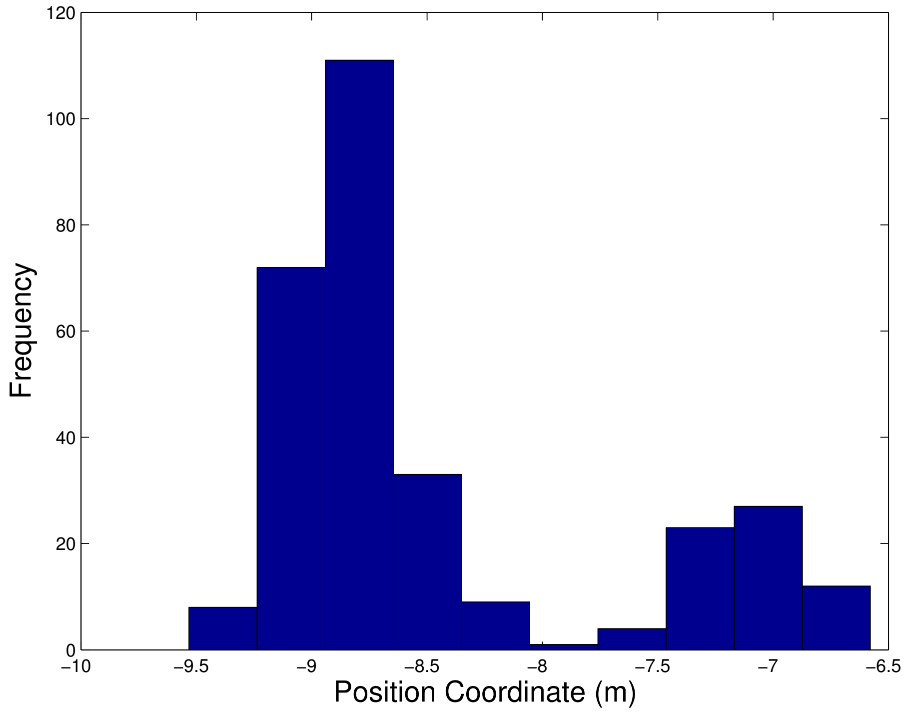

Figure 5.2: The First Position Coordinate Part of the Particles from Simulation Results

radio is attached to a personal computer acting as the FC, performing centralized data collection and tracking. The area plan is depicted in Figure 5.3. The area is 17m by 14m and the left side is shown as a dotted line because only the right half of the office is used.

Since the model chosen is the piecewise path loss model Equation 5.4, a cell division scheme and various parameters are needed. Fifteen pre-defined calibration points are chosen. They are placed evenly and cover the entire area. Each point is the center of a cell and each cell possesses its own unique bias parameters. The orientations of all the antennas are kept constant and human movements are controlled. The standard procedure is to collect different sets of data for training and validation and collect measurements at different locations within the cells. However, for simplicity, only one set of data is collected and it is used for both training and validation. Measurements are only taken at the centers of the cells. This is not

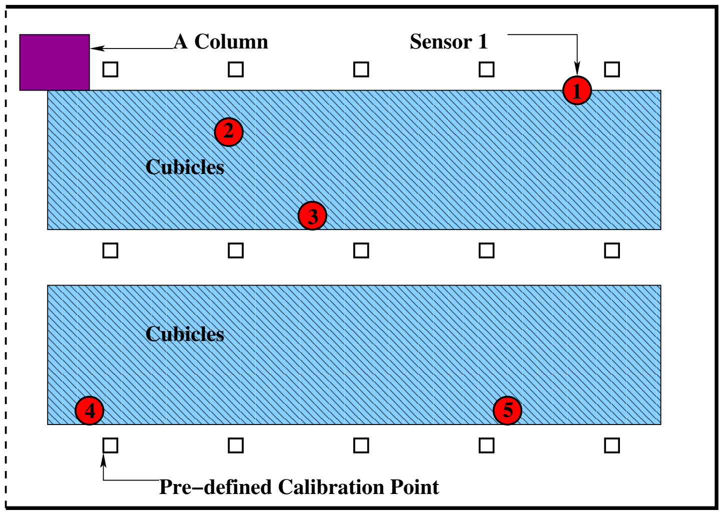

Figure 5.3: Floor Plan of Kaiser 4090

a huge problem as the focus of this chapter is on comparing two solutions for the Bayesian filter. The $\Gamma_n(\cdot)$ parameters are calculated from the sample averages from this single set of data. As for the other parameters required, they are all global and they are estimated to be $\sigma_u^2 = 0.02$, $\rho = 2.5$ and $\sigma_w^2 = 8$. Cross-validation is used to optimize these parameters, using which the final MSE after filtering is minimized.

Figure 5.4 shows the distribution of the estimated $\Gamma_n(\cdot)$ values. This range of values is roughly the same as the ones used in Section 5.4.2, exhibiting 'switching' behavior. Thus, similar results are expected.

The same procedure from Section 5.4 is applied to the experimental data. One final note of interest is that, a mechanism is needed to deal with missing and improbable RSSI values. Missing values are assigned 0dBmW and improbable values are obviously wrong values outside the valid range of RSSI measurements. Since the UKF and PF are both probabilistic, this chapter chooses to ignore these values, e.g., Equation 5.22 becomes

$$p(\mathbf{y}_t|\mathbf{x}_t^i) = \prod_{y_{n,t}\neq 0,\, y_{n,t}\leq -30,\, y_{n,t}\geq -95} p(y_{n,t}|\mathbf{x}_t^i). \tag{5.26}$$

Table Table 5.3 summarizes the results. The only difference is that the setup

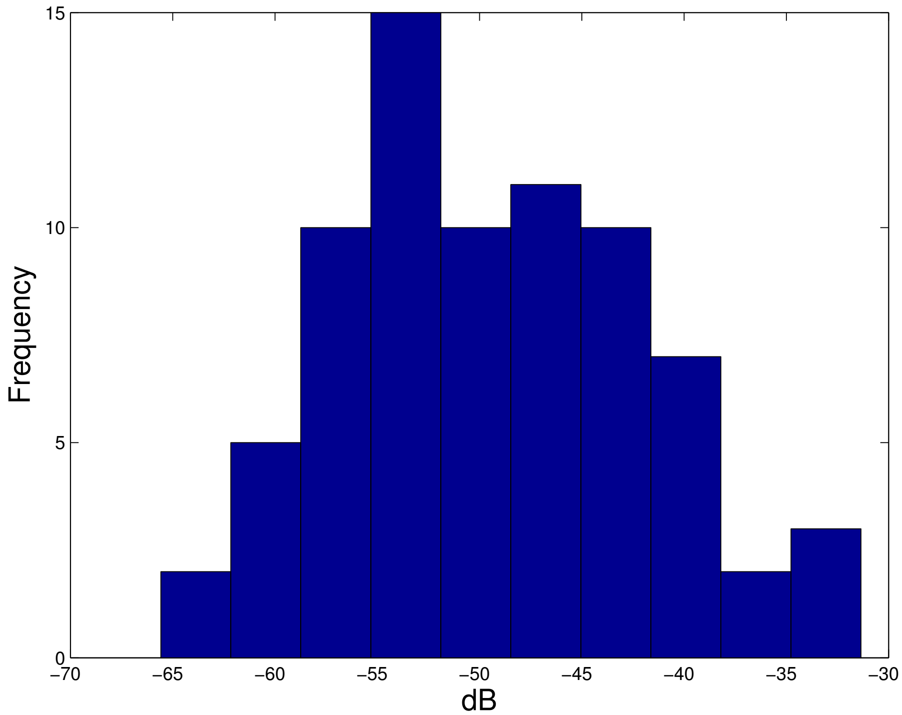

Figure 5.4: The $\Gamma_n(\cdot)$ Values from Experimental Results

is slightly different so the number of particles required for the PF is different and the number of sigma points for the UKF is $2L+1=23$. It can be seen that the UKF fails to function and loses track frequently. For instance, in Figure 5.5, the true horizontal position (the longer dimension in the floor plan) of the target is shown against the UKF and PF estimates. The UKF fails to track two centers of the cells. Compared to the simulation results in Section 5.4.2, the experimental results are worse due to a larger observation noise variance as well as realistic conditions introduced. However, in both cases, the PF, given sufficient number of particles, performs better than the UKF.

5.6 Conclusion

Simulations and real life experiments have been carried out to analyze the tracking results of the UKF and the PF. The UKF assumes near-linearity and imposes uni-

Table 5.3: Tracking Results in an Office

Mean Squared Error (m^2) over Time					
Filter	Number of Particles	Mean	Variance	5-th percentile	95-th percentile
PF	25	4.1094	98.4970	0.0386	26.2628
PF	50	4.7245	214.2936	0.0222	30.5959
PF	75	2.4589	113.7523	0.0197	8.1251
PF	100	2.0719	69.7546	0.0150	8.9874
PF	125	2.4531	103.4518	0.0168	9.8931
PF	150	1.8719	60.6488	0.0142	7.5954
PF	175	2.1849	87.9553	0.0123	8.9751
PF	200	2.7101	125.0296	0.0123	11.2968
PF	225	1.9210	65.4151	0.0152	8.2067
PF	250	2.0727	80.2627	0.0145	7.9518
PF	275	2.0467	68.2252	0.0128	9.5385
PF	300	2.0114	78.1461	0.0116	8.1292
UKF		7.1944	189.2249	0.0151	41.8589

modal Gaussian densities while the PF does not. These assumptions are very fragile and easily violated when the radio propagation characteristics of the tracking area are diverse, exhibiting 'switching' behavior. While the piecewise path loss observation model Equation 5.4 considered does not cover all regression techniques, we believe that the same principle can be generalized. Any indoor environment introduces occlusions and multipath propagation effects, resulting in highly diverse parameters. Unless free space propagation is the dominant mode of propagation in the environment, we speculate that a regressed model will exhibit 'switching' behavior. The UKF cannot handle multimodal systems. Therefore, the UKF is unsuitable under realistic conditions.

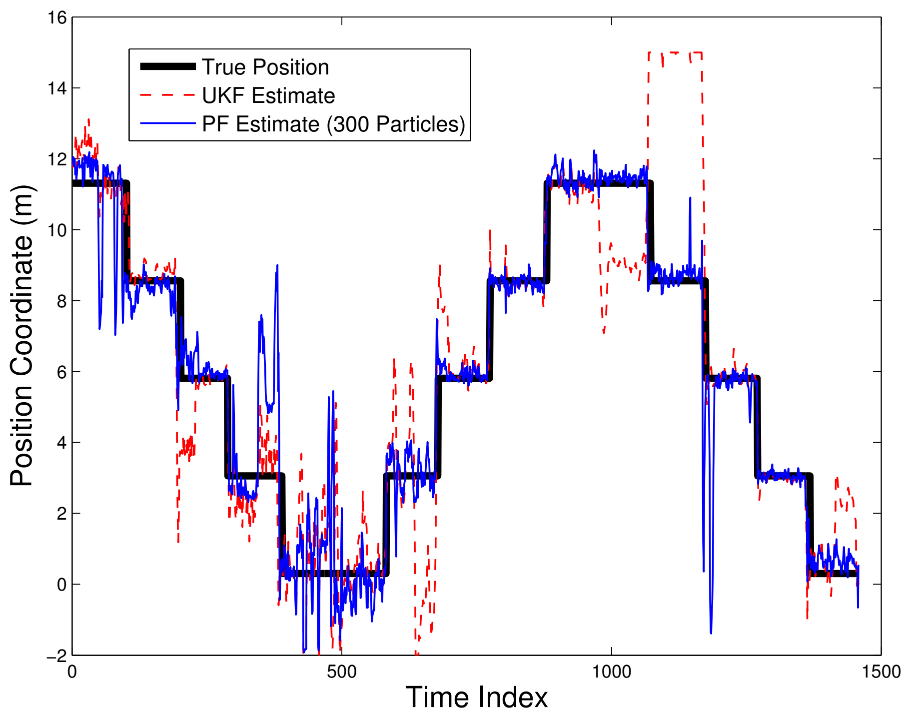

Figure 5.5: Horizontal Coordinate from Experimental Results

Chapter 6

A Comparison of Three Classifiers

As mentioned in Section 2.2, the fingerprinting paradigm is powerful and many regression techniques have been proposed and studied in the literature. However, to our best knowledge, the classification approach is relatively uncommon in the literature. This chapter addresses this issue by comparing three classifiers: KNN, SVM and SGC. Although all three achieve roughly the same performance, the SGC is the simplest in terms of computation costs and storage requirements. Furthermore, it can used under the Bayesian filter framework as discussed in Chapter 3 since it is probabilistic. On the other hand, the KNN and SVM are deterministic in nature and using the Bayesian filter framework is much more difficult.

The following sections present the system setup, introduce and summarize the three classifiers and present the conclusion based on experimental results.

6.1 System Model and Classifiers

Formally, the goal is to track a mobile transceiver moving in a bounded two-dimensional area equipped with N APs. Periodic broadcasts occur semi-frequently. Each AP acts as a sensor. It evaluates the signal strength from the target and forwards all the information to the FC. The area is divided into L cells. Let y_n be the RSSI between sensor n and the target in dBmW. Let $\mathbf{y} = [y_1, y_2, \ldots, y_n]^T$. Let x denote a cell number that takes values between 1 and L. The fingerprinting paradigm works in two phases. First, P instance-label pairs, $(x_i, \mathbf{y}_i), i = 1, \ldots, P$, are collected in known cells. Then, in the online working phase, an arriving RSSI instance from a unknown cell is compared against the radiomap and it is classified into one of the possible cells. The performance metric is the number of incorrect estimations, denoted classification error. The following are the three candidate classifiers.

6.1.1 K-Nearest Neighbor (KNN)

The KNN is the simplest classifier. Nothing is done after the offline training phase and the entire radiomap is stored. During the online working phase, an arriving instance is compared against the instances in the radiomap and K nearest instance-label pairs are selected. The K labels determine the estimated label of the arriving instance via majority voting. K is determined empirically beforehand. 'Nearest' is subject to some norm definition [5, 13].

6.1.2 Support Vector Machine (SVM)

The SVM is a binary classifier and it attempts to maximize the margin between the two classes [14]. For now, only two cells are considered and the notation is changed slightly. Let $x_i = +1$ denote if an instance comes from first cell and let $x_i = -1$ denote the opposite case. For the moment, the two classes are assumed to be linearly separable. Recalling that we have a database of training instance-label pairs $(x_i, \mathbf{y}_i), i = 1, \ldots, P$, SVM aims to find some parameters $\mathbf{w}$ and b such that

$$x_i(\mathbf{w}^T \mathbf{y}_i + b) - 1 \geq 0 \; \forall i, \tag{6.1}$$

i.e., we find two hyperplanes that separate the two classes. This is illustrated in Figure 6.1.

Given a unknown instance $\mathbf{y}$, the classifier or the hypothesis is then

$$\hat{x} = \text{sign}(\mathbf{w}^T \mathbf{y} + b). \tag{6.2}$$

Only some instance-label pairs are actually important. The pairs from the two classes that satisfy the inequalities above with equality are called *Support Vec-*

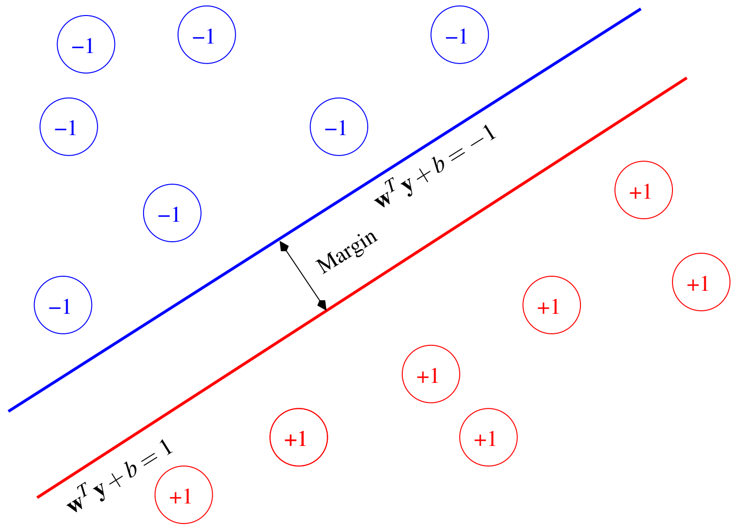

Figure 6.1: A Graphical Illustration of the SVM

tors. It is easily shown that the distance or *margin* between the two hyperplanes is $2/\|\mathbf{w}\|$. The optimal parameters can be found by solving the following quadratic optimization problem

$$\begin{array}{ll} \underset{\mathbf{w},b}{\text{minimize}} & \frac{1}{2}\mathbf{w}^T\mathbf{w} \\ \text{subject to} & x_i(\mathbf{w}^T\mathbf{y}_i + b) - 1 \geq 0 \;\forall i \end{array}. \tag{6.3}$$

Let $\lambda_i, i = 1, \ldots, P$, denote the Lagrange multipliers. The dual form can be written as

$$\begin{array}{ll} \underset{\Lambda}{\text{maximize}} & \sum_{i=1}^{P}\lambda_i - \frac{1}{2}\Lambda^T\mathbf{D}\Lambda \\ \text{subject to} & \lambda_i \geq 0 \;\forall i \\ & \sum_{i=1}^{P}\lambda_i x_i = 0 \end{array}, \tag{6.4}$$

where Λ is the vector of Lagrange multipliers and $\mathbf{D}$ is a symmetric matrix such that $\mathbf{D}_{ij} \triangleq x_i x_j \mathbf{y}_i^T\mathbf{y}_j$. Recalling that the support vectors satisfy the inequalities with equality, the Lagrange multiplier λ_i for a support vector $\mathbf{y}_i$ is strictly greater than zero. Therefore, if $\mathbf{w}^*$ is the optimal solution, then the optimal b^* satisfies $b^* =$

$x_i - \mathbf{w}^{*T}\mathbf{y}_i$ for any support vector pair $(x_i, \mathbf{y}_i)$. Finally, the classifier can be written as

$$\hat{x} = \mathrm{sign}(\sum_{i=1}^{P} x_i \lambda_i^* \mathbf{y}^T \mathbf{y}_i + b^*), \tag{6.5}$$

where the stars denote the optimal values.

If the two classes are not linearly separable, *soft margin* SVM introduces a penalty. The primal form of the optimization reads

$$\begin{aligned} \underset{\mathbf{w},b,\xi_i}{\text{minimize}} \quad & \frac{1}{2}\mathbf{w}^T\mathbf{w} + C\sum_{i=1}^{P}\xi_i \\ \text{subject to} \quad & x_i(\mathbf{w}^T\mathbf{y}_i + b) - 1 + \xi_i \geq 0 \ \forall i, \\ & \xi_i \geq 0 \ \forall i \end{aligned} \tag{6.6}$$

where C is a constant parameter of the optimization problem and ξ_i are the penalty terms. The dual form, which conveniently does not contain the penalty terms, can be written as

$$\begin{aligned} \underset{\Lambda}{\text{maximize}} \quad & \sum_{i=1}^{P}\lambda_i - \frac{1}{2}\Lambda^T\mathbf{D}\Lambda \\ \text{subject to} \quad & 0 \leq \lambda_i \leq C \ \forall i \quad . \\ & \sum_{i=1}^{P}\lambda_i x_i = 0 \end{aligned} \tag{6.7}$$

As before, Λ is the vector of Lagrange multipliers and $\mathbf{D}$ is a symmetric matrix such that $\mathbf{D}_{ij} \triangleq x_i x_j \mathbf{y}_i^T \mathbf{y}_j$.

Finally, the linear SVM can be converted to handle nonlinear boundaries by using the *kernel trick*. In the dual form, the objective contains a symmetric matrix $\mathbf{D}_{ij} \triangleq x_i x_j \mathbf{y}_i^T \mathbf{y}_j$. The trick changes the dot product $\mathbf{y}_i^T \mathbf{y}_j$ to a kernel $K(\mathbf{y}_i, \mathbf{y}_j)$. The physical meaning is that the kernel measures the similarity between two instances. The dot product is a linear kernel and sometimes using a nonlinear one yields better results. In this work, we use the standard Radial Basis Function (RBF)

$$K(\mathbf{y}_i, \mathbf{y}_j) = \exp\left(-\gamma \left\|\mathbf{y}_i - \mathbf{y}_j\right\|^2\right), \tag{6.8}$$

where $\gamma \geq 0$ is a constant parameter. The dual form stays the same as Equation 6.7

and the final classifier is

$$\hat{x} = \text{sign}(\sum_{i=1}^{P} x_i \lambda_i^* K(\mathbf{y}, \mathbf{y}_i) + b^*), \tag{6.9}$$

where the stars denote the optimal parameters.

Although the SVM is a binary classifier, it can be extended to a multiclass one. This chapter takes the one-versus-one strategy [35]. Since there are L possible cells, $L(L-2)/2$ binary classifiers are constructed. We iterate through each binary classifier and classify the unknown instance to one of the two possible labels. In the end, majority voting is conducted to determine the final label.

6.1.3 The Simple Gaussian Classifier (SGC)

Other than log-linear propagation loss, the standard path loss model [8, 11] also includes shadowing, which is modeled as Gaussian noise. Limited RSSI precision creates small bands from where the same RSSI measurement is reported regardless of the actual location of the target [10] . Combined with shadowing, it is reasonable to assume that, within a small region, the RSSI observed will follow a Gaussian profile and it is impossible distinguish where the target is precisely. Chapter 4 supports this claim. Of course, the theoretical shapes of such bands cannot be determined easily without ray tracing. Modeling RSSI values as Gaussian random variables has been proposed numerous times in the literature. The our best knowledge, [22] is the first one to perform localization using a probabilistic formulation. Nevertheless, the authors focus on estimating position coordinates instead of viewing it as a classification problem. Their choice of the probabilistic model uses a number of Gaussian kernels. We believe [24] is the first one to propose treating it as a classification problem and specifically model RSSI values as Gaussian random variables. This has been discussed afterwards in [15, 20]. We are intrigued by its astonishing simplicity and we refer to it as the SGC.

The distribution of RSSI between sensor n and any location in cell l is modeled as

$$p(y_n|x=l) = \frac{1}{\sqrt{2\pi\sigma_{l,n}^2}} \exp\left(-\frac{(y_n - \mu_{l,n})^2}{2\sigma_{l,n}^2}\right). \tag{6.10}$$

It is safe to assume that the sensors are independent. Therefore, the multivariate distribution involving all sensors is

$$\begin{aligned} & p(\mathbf{y}|x=l) \\ & = \prod_{n=1}^{n=N} p(y_n|x=l) \\ & = \frac{1}{(2\pi)^{N/2}|\Sigma_l|^{1/2}} \exp\left(\frac{-(\mathbf{y}-\mu_l)^T \Sigma_l^{-1} (\mathbf{y}-\mu_l)}{2}\right) \end{aligned}, \tag{6.11}$$

where the mean vector μ_l consists of $\mu_{l,n}$ and the covariance Σ_l is a diagonal matrix with $\sigma_{l,n}^2$ placed on the diagonal.

In the online working phase, given an arriving instance $\mathbf{y}$, the goal is to find the cell number l that maximizes the probability $p(x|\mathbf{y})$. As mentioned in Chapter 3 and [22],

$$p(x|\mathbf{y}) = \frac{p(\mathbf{y}|x)p(x)}{p(\mathbf{y})}. \tag{6.12}$$

Without filtering, i.e., treating this as an one-shot-in-time event, $p(x)$ is assumed to be uniform and the problem is equivalent to maximizing $p(\mathbf{y}|x)$. This is accomplished by a simple search over all possible cell numbers, i.e.,

$$\hat{x} = \underset{x}{\operatorname{argmax}}\ p(\mathbf{y}|x). \tag{6.13}$$

Since there are only a finite number possible cells and the SGC is expressed in closed-form, it is possible to derive the Pairwise Error Probability (PEP) between two cells a and b analytically. Let the decision metric be

$$D_1 = \log p(\mathbf{y}|x=a) - \log p(\mathbf{y}|x=b). \tag{6.14}$$

The target is estimated to be in cell a if $D_1 > 0$ and estimated to be in cell b otherwise. After some manipulation,

$$\begin{aligned} D_1 = & -\frac{1}{2}\log|\Sigma_a| + \frac{1}{2}\log|\Sigma_b| + \frac{1}{2}\mathbf{y}^T\left(\Sigma_b^{-1} - \Sigma_a^{-1}\right)\mathbf{y} \\ & + \mathbf{y}^T\left(\Sigma_a^{-1}\mu_a - \Sigma_b^{-1}\mu_b\right) + \frac{1}{2}\mu_b^T\Sigma_b^{-1}\mu_b - \frac{1}{2}\mu_a^T\Sigma_a^{-1}\mu_a \end{aligned}. \tag{6.15}$$

Therefore, if the target is known to be in cell a, the PEP is $p(\hat{x}=b|x=a)=p(D_1<0)$. $\mathbf{y}$ is a multivariate Gaussian but the term $\mathbf{y}^T(\ldots)\mathbf{y}$ is no longer Gaussian. Therefore, for simplicity, the following assumes that the covariances are equal, i.e., $\Sigma_a=\Sigma_b=\mathbf{P}$ and the suboptimal decision metric is

$$D_2=\mathbf{y}^T\mathbf{P}^{-1}(\mu_a-\mu_b)-\frac{1}{2}(\mu_a+\mu_b)^T\mathbf{P}^{-1}(\mu_a-\mu_b). \tag{6.16}$$

This new decision metric is completely Gaussian and it presents the worst case scenario since the covariances are assumed to be equal. From [15], the mean is

$$\mathbb{E}[D_2]=\frac{1}{2}(\mu_a-\mu_b)^T\mathbf{P}^{-1}(\mu_a-\mu_b) \tag{6.17}$$

and the variance is

$$\mathbb{E}\left[(D_2-\mathbb{E}[D_2])^2\right]=(\mu_a-\mu_b)^T\mathbf{P}^{-1}(\mu_a-\mu_b). \tag{6.18}$$

The term

$$H_{a,b}\triangleq(\mu_a-\mu_b)^T\mathbf{P}^{-1}(\mu_a-\mu_b) \tag{6.19}$$

is called the Mahalanobis distance between the two multivariate Gaussian densities. This explains why the KNN works so well using Mahalanobis norm instead of Euclidean norm [15, 20] for both regression and classification. Therefore, the PEP is

$$p(\hat{x}=b|x=a)=p(D_2<0)=\Phi\left(-\frac{1}{2}\sqrt{H_{a,b}}\right), \tag{6.20}$$

where Φ is the cumulative density function of the standard Gaussian with mean of 0 and variance of 1.

Similar to the idea of signal constellation from Communications, it is possible to derive a simple upper bound on the probability of returning the wrong cell number if there multiple cells. If the target is in cell a, the union bound of making the wrong estimate is

$$\sum_{l\neq a}p(\hat{x}=l|x=a). \tag{6.21}$$

Since the PEP discussed is the worst case scenario, this union bound is a sum of upper bounds. Therefore, taking the complement, the lower bound on the probability of returning the correct estimate is obtained. Unfortunately, the union bound is only tight if the noise is small. This is not the case as classification errors under realistic conditions cannot be expected to be under 1%. Therefore, this bound is extremely loose.

6.2 Results without Filtering

We now provide a direction comparison between the three classifiers introduced. The test environment is an office consisting of small rooms and we use the same custom-made radios used in Section 4.2. The environment is harsh as Non-Line-of-Sight (NLOS) conditions are the norm. The walls separating the rooms are made of wooden and paper materials. The mobile target broadcasts to APs at 433MHz using a small loop antenna every 100ms. The APs act as sensors. The APs are secured and mounted on walls, well above the ground. Each RSSI measurement is calculated based on the voltage values at the receiving antenna averaged over the packet length. A typical packet lasts approximately 10ms. Only two significant digits are available in dB scale. The APs communicate with each other and the FC via a mesh ZigBee network. The traffic in the network is kept low and APs are placed strategically to cover the entire area. Nevertheless, there is no way to discern the exact cause when an AP does not hear from the target. Either the signal is too weak or data packets are dropped in the higher level ZigBee network. When an AP receives nothing, an arbitrary RSSI value of -100dBmW is assigned. Probabilistic frameworks can deal with cases like this rather easily by ignoring these values, e.g., Equation 6.11 becomes

$$p(\mathbf{y}|x=l) = \prod_{y_n \neq -100} p(y_n|x=l). \tag{6.22}$$

However, it is not so straightforward for deterministic methods such as the SVM which relies on solving quadratic optimization problems. Therefore, to be fair to all classifiers, the -100dBmW values are simply taken at face value.

Figure 6.2 shows the floor plan of the office and the placement of the APs. The dimensions are 14.77m by 24.38m. The blue dots are the 8 APs and the underlined numbers enumerate the cells. Since the area is nicely divided into rooms of roughly

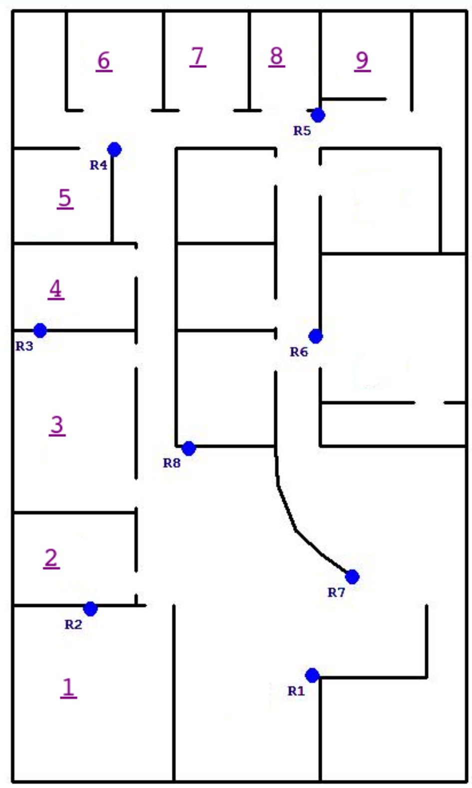

Figure 6.2: Floor Plan of the Environment and the Cell Numbers

equal sizes, each room is a cell.

Two tests performed at different times are used to compare the classification capabilities of the three classifiers. For each test, two loops around the area are recorded. The first loop is used as the training data and the second loop is used as the validation data.

There are various parameters needed for the classifiers and they are determined by cross-validation. Typically, cross-validation involves tuning using the training data only, as the validation data must remain unseen. However, the work presented here is less strict. We simply use the training data to train the classifiers with arbitrary parameters, use the validation data, evaluate the classification errors and repeat with different parameters until the best performance is achieved.

For the KNN, the ℓ_2-norm (i.e., Euclidean distance) is used to measure distance.

Table 6.1: KNN Results (K = 1210) for the First Test

Cell	Number of Time Slots	Correct Estimations	Off by One Cell	Off by More than One cell
1	124	79	34	11
2	213	187	26	0
3	366	330	34	2
4	202	1	200	1
5	166	152	1	13
6	175	12	159	4
7	133	97	27	9
8	139	47	91	1
9	147	96	10	41
Overall	1665	1001(60%)	582(35%)	82(5%)

For the SVM, LIBSVM [36] is used to handle the optimization problem. As recommended by its authors, RSSI values are scaled linearly between 0 and 1 for numerical purposes before using the library.

For the SGC, sample means and variances from the training instance-label pairs are used as the means and variances of the Gaussian variables. This is the frequencist approach. Furthermore, we take the sample variances of all the measurements in all cells and use them for calculating the Mahalanobis distance between two cells. This is only for illustration purposes and it is not used for estimation.

6.2.1 The First Test

In this test, the target is kept on a stool at constant height and it never goes near obstacles or corners. For both the training and validation set, the antenna orientation of the target is varied. In each cell, the target is placed at several locations for some time. The measurements at each time slot are recorded as an instance-label pair. The size of the training set is 3881 and the size of the validation set is 1665. 3471 out of $3881 \times 8 = 31048$ (11%) possible measurements are missing for the training data and 1220 out of $1665 \times 8 = 13320$ (9%) possible measurements are missing for the validation data.

Table 6.2: SVM Results ($C = 10^{-1.1}$ and $\gamma = 10^{0.3}$) for the First Test

Cell	Number of Time Slots	Correct Estimations	Off by One Cell	Off by More than One cell
1	124	45	55	24
2	213	176	37	0
3	366	325	35	6
4	202	13	188	1
5	166	148	11	7
6	175	119	29	27
7	133	92	36	5
8	139	68	69	2
9	147	84	28	35
Overall	1665	1070(64%)	488(29%)	107(6%)

Table 6.3: SGC Results for the First Test

Cell	Number of Time Slots	Correct Estimations	Off by One Cell	Off by More than One cell
1	124	35	59	30
2	213	152	61	0
3	366	296	68	2
4	202	106	93	3
5	166	128	34	4
6	175	57	82	36
7	133	94	34	5
8	139	89	48	2
9	147	89	12	46
Overall	1665	1046(63%)	491(29%)	128(8%)

Table 6.1, Table 6.2 and Table 6.3 show the results. The raw numbers indicate that it is difficult to distinguish between neighboring cells, as indicated by the sizable number of wrong estimates that are off by one cell. However, the estimates are off by more than one cell in only a small number of cases. This suggest that better AP placement and a higher number of APs are needed in order to obtain higher accuracies in this cell division scheme. More specifically, about 10% of all the measurements are missing and they have been assigned the arbitrary RSSI value

Table 6.4: SGC Accuracy Bounds for the First Test

Cell	The Closest Cell	Mahalanobis Distance to the Closest Cell	Accuracy Bound
1	2	0.7461	0.3694
2	1	0.7461	0.3378
3	4	2.0975	0.1071
4	3	2.0975	0.0919
5	6	1.5543	0.0735
6	5	1.5543	0
7	8	0.7092	0
8	7	0.7092	0
9	8	1.5601	0.1182
Average			0.1220

of -100dBmW. The missing measurements are more likely to be caused by weak signals rather than busy traffic in the network. Therefore, better placement of the APs will improve the performance.

Table 6.4 shows the accuracy bounds using the SGC approach. As noted, the union bound is extremely loose. This is further complicated by the fact that modeling the RSSI values as Gaussian random variables is only an approximation. Therefore, directly evaluating the classification performance is probably more useful than evaluating the PEP via calculating Mahalanobis distance.

6.2.2 The Second Test

The first test is repeated but with realistic conditions. For both the training and validation loops, the target is allowed to change height, move close to walls and major obstacles and human movements are not controlled. Therefore, there are instance-label pairs in the validation set that are not seen in the training set and vice versa. The size of the training set is 2364 and the size of the validation set is 1089. 1999 out of $2364 \times 8 = 18912$ (11%) possible measurements are missing for the training data and 817 out of $1089 \times 8 = 8712$ (9%) possible measurements are missing for the validation data.

Table 6.5: KNN Results (K = 186) for the Second Test

Cell	Number of Time Slots	Correct Estimations	Off by One Cell	Off by More than One cell
1	170	150	3	17
2	70	8	61	1
3	127	106	5	16
4	85	21	42	22
5	177	106	53	18
6	86	56	21	9
7	108	32	56	20
8	130	58	69	3
9	136	97	23	16
Overall	1089	634(58%)	333(31%)	122(11%)

Table 6.6: SVM Results ($C = 10^{0.4}$ and $\gamma = 10^{0.8}$) for the Second Test

Cell	Number of Time Slots	Correct Estimations	Off by One Cell	Off by More than One cell
1	170	130	23	17
2	70	27	42	1
3	127	104	18	5
4	85	53	19	13
5	177	111	50	16
6	86	49	25	12
7	108	37	45	26
8	130	63	62	5
9	136	81	39	16
Overall	1089	655(60%)	323(30%)	111(10%)

Table 6.5, Table 6.6 and Table 6.7 show the results. Compared to the first test, the results are slightly worse due to the realistic conditions imposed. The problem of imperfect training is a big problem for the fingerprinting paradigm. For both regression and classification, the conditions for training and testing must be kept the same to the best of abilities. Once trained, the model learned is static and the performance decreases if the conditions change. Table 6.8 shows the accuracy bounds using the SGC. The same conclusion can be drawn as the first test.

Table 6.7: SGC Results for the Second Test

Cell	Number of Time Slots	Correct Estimations	Off by One Cell	Off by More than One cell
1	170	119	46	5
2	70	24	46	0
3	127	86	27	14
4	85	46	20	19
5	177	89	76	12
6	86	36	36	14
7	108	51	43	14
8	130	75	51	4
9	136	84	28	24
Overall	1089	610(56%)	373(34%)	106(10%)

Table 6.8: SGC Accuracy Bounds for the Second Test

Cell	The Closest Cell	Mahalanobis Distance to the Closest Cell	Accuracy Bound
1	2	2.9654	0.4744
2	1	2.9654	0.4609
3	4	2.0228	0.1892
4	3	2.0228	0.0406
5	6	0.8852	0
6	5	0.8852	0
7	8	1.2691	0
8	9	0.9471	0
9	8	0.9471	0.0813
Average			0.1385

Since all three classifiers are roughly equal in terms of classification errors, the SGC is clearly the preferred classifier. First, no cross-validation is required compared to the KNN and SVM. Second, the computation and storage costs are extremely low. The SGC only requires computing the sample means and variances from the training set. This can be done recursively so that only the newest instance-label pair is needed. On the other hand, the KNN requires storing and looking at every pair in a big radiomap and the SVM requires storing a big radiomap as well

Algorithm 4 For each time slot t

Start with previous time slot's weights, $\left\{w_{t-1}^i\right\}_{i=1}^L$.

For each state i,

$$\widetilde{w}_t^i = p(\mathbf{y}_t|x_t = i)\sum_j w_{t-1}^j p(x_t = i|x_t = j)$$

Normalize using

$$w_t^i = \frac{\widetilde{w}_t^i}{\sum_j \widetilde{w}_t^j}$$

The estimate is $\hat{x}_t = \underset{i}{\operatorname{argmax}}\ w_t^i$

as solving a series of quadratic optimization problems.

6.3 The Hidden Markov Model (HMM) Filter

Now, this chapter investigates the use of Bayesian filtering for the SGC. As mentioned in Chapter 3, the HMM filter is the exact and optimal solution because the state space is finite. Let $\left\{w_t^i = p(x_t = i|\mathbf{y}_{1:t})\right\}_{i=1}^L$ be the normalized weights of the discrete posterior density and let $\widetilde{w}_t^i$ be the unnormalized ones. Replacing all the integrals in Chapter 3 by summations, the HMM filter is summarized in Algorithm 4.

For the transition model $p(x_t|x_{t-1})$, it can be easily generated from the floor plan. For the observation model $p(\mathbf{y}_t|x_t)$, the SGC is already a native probabilistic formulation so Equation 6.11 is available. On the other hand, the KNN and SVM are deterministic in nature. There are several methods for estimating probabilities for the KNN and SVM. For instance, in the majority voting stage for KNN, if a out of K labels are from cell Z, a reasonable probability estimate for $p(\mathbf{y}_t|x_t = Z)$ is a/K. Schemes for the SVM exist as well [37]. However, the SGC is already established as the preferred classifier and the next section will not include the KNN and SVM under the Bayesian filter framework.

Table 6.9: Filtered SGC Results for the First Test

Cell	Number of Time Slots	Correct Estimations	Off by One Cell	Off by More than One cell
1	124	43	56	25
2	213	167	46	0
3	366	329	37	0
4	202	104	98	0
5	166	142	24	0
6	175	78	76	21
7	133	102	31	0
8	139	91	46	2
9	147	104	18	25
Overall	1665	1160(70%)	432(26%)	73(4%)

6.4 Results with Filtering

The same sets of data from Section 6.2 are used. Considering the floor plan, the following transition probabilities are applied

$$\begin{aligned} & p(x_t = i | x_{t-1} = i) = 0.95,\ i = 1, \ldots, 9 \\ & p(x_t = i \pm 1 | x_{t-1} = i) = 0.025,\ i = 2, \ldots, 8 \\ & p(x_t = 2 | x_{t-1} = 1) = 0.05 \\ & p(x_t = 8 | x_{t-1} = 9) = 0.05 \end{aligned}. \tag{6.23}$$

Table 6.9 and Table 6.10 summarize the results. As expected, the use of filtering improves the results. In particular, the overall success rate is at or above 70%. Very few estimates are wrong by more than one cell. This is similar to the results in [24] with 8 sensors. Interestingly, the imperfect conditions of the second test do not affect the performance. We draw the conclusion that the use of filtering counteracts the effects of blocking the antennas and moving the target close to walls.

6.5 Conclusion

This chapter has compared the performance of the KNN, SVM and SGC in an indoor environment under realistic conditions. All three classifiers achieve the same

Table 6.10: Filtered SGC Results for the Second Test

Cell	Number of Time Slots	Correct Estimations	Off by One Cell	Off by More than One cell
1	170	122	48	0
2	70	26	44	0
3	127	96	27	4
4	85	74	8	3
5	177	111	64	2
6	86	70	16	0
7	108	66	40	2
8	130	97	33	0
9	136	118	15	3
Overall	1089	780(72%)	295(27%)	14(1%)

performance but the SGC is the simplest in terms of implementation and storage requirements. Furthermore, the SGC is a native probabilistic formulation and it can be used under the Bayesian filter framework. Without filtering, the accuracy obtained is around 60%. With filtering, the performance of the SGC improves to 70% and virtually no estimate is off by more than one cell.

Of course, comparing against the regression approach, the classification approach is less precise. We have carefully avoided talking about position coordinates and errors in terms of MSE. However, we can derive a simple estimate of the MSE of this scheme. Let us consider two adjacent cells such as cell 1 and 2 in Figure 6.2 and assume both cells are 5m by 5m. We use a coordinate system (u, v) and the origin is at the lower left corner of cell 1. From the experimental results, we consider the case that the classification accuracy is 70% and the mistaken classifications are off by one cell. If we assume that the target is equally likely to be anywhere, i.e., uniformly distributed over the cell area, and we always return the center of cell 1 as the position coordinate, then the maximum error is

$\sqrt{2.5^2 + (2.5+5)^2} \cong 7.91$m and the MSE is

$$\begin{aligned} &\frac{0.7}{25}\int_{u=0}^{u=5}\int_{v=0}^{v=5}(u-2.5)^2+(v-2.5)^2 dudv \\ +&\frac{0.3}{25}\int_{u=0}^{u=5}\int_{v=5}^{v=10}(u-2.5)^2+(v-2.5)^2 dudv \cong 11.67. \end{aligned} \tag{6.24}$$

Thus, the root mean squared error is 3.42m. This is in line with regression techniques presented in the literature performed under similar setups.

Chapter 7

Online Parameter Estimation for the General Bayesian Filter

The biggest drawbacks of the fingerprinting paradigm discussed in Section 2.2 are the effort required to perform training and the static nature of the model trained. To our knowledge, there is not much discussion on the second issue in the literature other than resorting to performing the offline training step again. [16] suggests placing multiple static reference APs, whose locations are perfectly known, in order to update the observation model. However, its use is limited if the number of reference APs is low. This chapter proposes extending the Bayesian filter framework introduced in Chapter 3 and applied in Chapter 5 and Chapter 6 to deal with these two issues. A probabilistic observation model constructed from the fingerprinting paradigm is governed by various parameters. Due to imperfect training or changing conditions, these parameters may not be optimal. Therefore, we view this challenge as estimating parameters of the observation model under the Bayesian filter framework. In addition to the unknown state variables, the parameters of the observation model are also unknown.

For simplicity, let us assume that there is only one unknown parameter θ. Let the subscript in $p_\theta(\mathbf{y}_t|\mathbf{x}_t)$ highlight this important unknown parameter. [38] gives an excellent overview on state of the art methods for estimating θ. We take the ML approach because it is the most mature. The optimal value of the parameter in the

ML sense maximizes the total observation likelihood

$$\log p_{\theta}(\mathbf{y}_{1:t}), \tag{7.1}$$

i.e., the value that best explains the sequence of observations up to the current time index t. For the HMM, this is an old problem and the famous Expectation-Maximization (EM) algorithm accomplishes the task [28]. This approach has been explored in [24]. Unfortunately, the EM algorithm is an offline method. It requires storing the entire sequence of observations up to time index t and iterating through the sequence several times. In contrast, the online ML scheme attempts to maximize the *predictive likelihood*

$$\begin{aligned}
&p_{\theta}(\mathbf{y}_t|\mathbf{y}_{1:t-1}) \\
&= \int p_{\theta}(\mathbf{y}_t|\mathbf{x}_t)p_{\theta}(\mathbf{x}_t|\mathbf{y}_{1:t-1})d\mathbf{x}_t \\
&= \int\int p_{\theta}(\mathbf{y}_t|\mathbf{x}_t)p(\mathbf{x}_t|\mathbf{x}_{t-1})p_{\theta}(\mathbf{x}_{t-1}|\mathbf{y}_{1:t-1})d\mathbf{x}_t d\mathbf{x}_{t-1}
\end{aligned}. \tag{7.2}$$

This can be accomplished by a stochastic gradient ascent algorithm

$$\theta_t = \theta_{t-1} + \varepsilon \frac{\partial}{\partial \theta} \log p_{\theta}(\mathbf{y}_t|\mathbf{y}_{1:t-1}), \tag{7.3}$$

where ε is the stepsize. Using a constant stepsize allows us to track the parameter if it is slowly changing in time. A large value of ε provides faster convergence at a cost of lower stability. Equation 7.3 implies that we need to differentiate Equation 7.2 with respect to the parameter. This is commonly known as the *filter derivative*. We would like to point out this in an abuse of notation because evaluating the observation model $p_{\theta}(\mathbf{y}_t|\mathbf{x}_t)$ and the predictive likelihood Equation 7.2 requires knowing the exact value of θ. To bypass the chicken and egg problem, the algorithm works as follows: At each time index, the filter uses the current estimate θ_{t-1} to evaluate the observation likelihood, i.e., it is treated as the true value of θ. Then, this likelihood is used to produce the posterior density in the exact same manner as the vanilla Bayesian filter in Chapter 3 and produce the predicative likelihood Equation 7.2. Finally, the predicative likelihood allows us to calculate a new value of the parameter θ_t via Equation 7.3.

The online ML scheme is conceptually simple just like the vanilla Bayesian filter discussed in Chapter 3 but no analytical solution exists except for some special cases. For instance, the algorithm has been studied for the HMM in [39]. In the interest of generality, we continue forward using the recent developments in particle methods, which allow us to use this parameter estimating scheme for the general Bayesian filter. This is a continuation of the PF introduced in Section 5.3.

The following summarize the essential steps for achieving online ML parameter estimation using particle methods as well as our preliminary work for combating imperfectly trained and time-varying parameters.

7.1 The Marginal Particle Filter and the Filter Derivative

The following are based on recent developments in marginal particle filtering [40]. It has been successfully used in robotics [41].

Formally, let

$$p_\theta(\mathbf{x}_t|\mathbf{y}_{1:t}) \triangleq \frac{\xi_\theta(\mathbf{x}_t,\mathbf{y}_{1:t})}{\int \xi_\theta(\mathbf{x}_t,\mathbf{y}_{1:t})d\mathbf{x}_t}. \tag{7.4}$$

Then, from Chapter 3,

$$\xi_\theta(\mathbf{x}_t,\mathbf{y}_{1:t}) = p_\theta(\mathbf{y}_t|\mathbf{x}_t)\int p(\mathbf{x}_t|\mathbf{x}_{t-1})p_\theta(\mathbf{x}_{t-1}|\mathbf{y}_{1:t-1})d\mathbf{x}_{t-1}. \tag{7.5}$$

Differentiating with respect to θ,

$$\frac{\partial}{\partial\theta}p_\theta(\mathbf{x}_t|\mathbf{y}_{1:t}) = \frac{\frac{\partial}{\partial\theta}\xi_\theta(\mathbf{x}_t,\mathbf{y}_{1:t})}{\int \xi_\theta(\mathbf{x}_t,\mathbf{y}_{1:t})d\mathbf{x}_t} - p_\theta(\mathbf{x}_t|\mathbf{y}_{1:t})\frac{\int \frac{\partial}{\partial\theta}\xi_\theta(\mathbf{x}_t,\mathbf{y}_{1:t})d\mathbf{x}_t}{\int \xi_\theta(\mathbf{x}_t,\mathbf{y}_{1:t})d\mathbf{x}_t} \tag{7.6}$$

and

$$\begin{aligned}\frac{\partial}{\partial\theta}\xi_\theta(\mathbf{x}_t,\mathbf{y}_{1:t}) = {} & p_\theta(\mathbf{y}_t|\mathbf{x}_t)\frac{\partial}{\partial\theta}\log p_\theta(\mathbf{y}_t|\mathbf{x}_t) \\ & \times \int p(\mathbf{x}_t|\mathbf{x}_{t-1})p_\theta(\mathbf{x}_{t-1}|\mathbf{y}_{1:t-1})d\mathbf{x}_{t-1} \\ & + p_\theta(\mathbf{y}_t|\mathbf{x}_t)\int p(\mathbf{x}_t|\mathbf{x}_{t-1})\frac{\partial}{\partial\theta}p_\theta(\mathbf{x}_{t-1}|\mathbf{y}_{1:t-1})d\mathbf{x}_{t-1}\end{aligned}. \tag{7.7}$$

Now, realizing that all these integrals above are intractable, we invoke the idea of the PF as in Section 5.3. A particle $\mathbf{x}_t^i$ is a candidate of the unknown state vector and it has a weight w_t^i. P particles and the weights approximate the posterior density, i.e.,

$$\widehat{p_\theta}(\mathbf{x}_t|\mathbf{y}_{1:t}) = \sum_{i=1}^{P} w_t^i \delta_{\mathbf{x}_t^i}(\mathbf{x}_t^i). \tag{7.8}$$

Similar to Section 5.3, a mixture of the transition model is used as the proposal distribution

$$q_\theta(\mathbf{x}_t|\mathbf{y}_{1:t}) = \sum_{j=1}^{P} w_{t-1}^j p(\mathbf{x}_t|\mathbf{x}_{t-1}^j). \tag{7.9}$$

This can be easily sampled using composition sampling. The importance weights are defined in the marginal space

$$w_t = \frac{p_\theta(\mathbf{x}_t|\mathbf{y}_{1:t})}{q_\theta(\mathbf{x}_t|\mathbf{y}_{1:t})}. \tag{7.10}$$

In contrast to Equation 5.13, we are in the marginal space instead of the joint space. This is called the marginal PF and it is only equivalent to the original SIR-PF when the transition model is used as the proposal density. The weight update formula remains the same, i.e.,

$$\widetilde{w}_t^i = p_\theta(\mathbf{y}_t|\mathbf{x}_t^i), \tag{7.11}$$

where $\widetilde{w}_t^i$ is the unnormalized weight.

Similar to the posterior density, the filter derivative is approximated by the *same set of particles* but with different weights, i.e.,

$$\begin{aligned} \widehat{\frac{\partial}{\partial\theta}\xi_\theta}(\mathbf{x}_t, \mathbf{y}_{1:t}) &= \sum_{i=1}^{P} \rho_t^i \delta_{\mathbf{x}_t^i}({\mathbf{x}^i}_t) \\ \widehat{\frac{\partial}{\partial\theta}p_\theta}(\mathbf{x}_t|\mathbf{y}_{1:t}) &= \sum_{i=1}^{P} w_t^i \beta_t^i \delta_{\mathbf{x}_t^i}({\mathbf{x}^i}_t) \end{aligned}. \tag{7.12}$$

Let $\widetilde{\rho}_t^i$ be the unnormalized weight of the filter derivative. From Equation 7.7, the update rule is

$$\widetilde{\rho}_t^i = \widetilde{w}_t^i \frac{\partial}{\partial\theta} \log p_\theta(\mathbf{y}_t|\mathbf{x}_t^i) + \frac{\widetilde{w}_t^i \sum_j w_{t-1}^j p(\mathbf{x}_t^i|\mathbf{x}_{t-1}^j)\beta_{t-1}^j}{\sum_j w_{t-1}^j p(\mathbf{x}_t|\mathbf{x}_{t-1}^j)}. \tag{7.13}$$

Finally, these particles allows us to do online ML parameter estimation because

$$\widehat{\frac{\partial}{\partial\theta}\log p_\theta}(\mathbf{y}_t|\mathbf{y}_{1:t-1}) = \frac{\widehat{\frac{\partial}{\partial\theta}p_\theta}(\mathbf{y}_t|\mathbf{y}_{1:t-1})}{\widehat{p_\theta}(\mathbf{y}_t|\mathbf{y}_{1:t-1})} = \frac{\int \widehat{\frac{\partial}{\partial\theta}\xi_\theta}(\mathbf{x}_t,\mathbf{y}_{1:t})d\mathbf{x}_t}{\int \widehat{\xi_\theta}(\mathbf{x}_t,\mathbf{y}_{1:t})d\mathbf{x}_t} = \frac{\sum_j \widetilde{\rho}_t^j}{\sum_j \widetilde{w}_t^j} \tag{7.14}$$

We summarize the complete Bayesian filter and the filter derivative approximated by particles in Algorithm 5[1]. The normalization step ensures that $\int p_\theta(\mathbf{x}_t|\mathbf{y}_{1:t})d\mathbf{x}_t = 1$ and $\int \frac{\partial}{\partial\theta}p_\theta(\mathbf{x}_t|\mathbf{y}_{1:t})d\mathbf{x}_t = 0$. For the initial value of the filter derivative for at $t = 0$, we can simply set $\beta_0^i = 0\ \forall i$. We would like to point out that the same procedure needs to be repeated if there are multiple unknown parameters to be estimated.

Algorithm 5 works for any Bayesian filter but the case for the HMM is further simplified because the state space is finite. Algorithm 6 is a summary of the case for the HMM with L states.

One final thing to note is that the ML approach converges slowly. Therefore, we must assume that the parameters change slowly overtime. This is typically not a problem unless something extreme happens globally. In principle, we can skip the training phase and use Bayesian filtering to blindly estimate both the unknown state vector and the parameters. However, the ML approach is not guaranteed to converge to the global optima. For instance, for the HMM, if we initialize all the parameters for all states to equal values, i.e., $p_\theta(\mathbf{y}_t|x_t = i) = p_\theta(\mathbf{y}_t|x_t = j)\ \forall i, j$, then the filter will fail as the target is equally likely to be in anywhere. We must initialize the parameters to reasonable values that are close to the true values. Therefore, the training phase is still necessary but it may be done imperfectly since we only need good ballpark values to jump start the Bayesian filter.

[1]There is no explicit resampling in the algorithm because the proposal is a mixture of the transition model. Using composition sampling for this already achieves resampling.

Algorithm 5 For a parameter θ at time slot t

Start with previous time index's posterior and filter derivative, $\left\{\mathbf{x}_{t-1}^j\right\}_{j=1}^P$, $\left\{w_{t-1}^j\right\}_{j=1}^P$ and $\left\{\beta_{t-1}^j\right\}_{j=1}^P$.

Draw new particles using composition sampling.

$$\mathbf{x}_t^i \sim \sum_j w_{t-1}^j p(\mathbf{x}_t|\mathbf{x}_{t-1}^j)$$

For each particle i,

- $$\widetilde{w}_t^i = p_\theta(\mathbf{y}_t|\mathbf{x}_t^i)$$
- $$\widetilde{\rho}_t^i = \widetilde{w}_t^i \frac{\partial}{\partial \theta} \log p_\theta(\mathbf{y}_t|\mathbf{x}_t^i) + \frac{\widetilde{w}_t^i \sum_j w_{t-1}^j p(\mathbf{x}_t^i|\mathbf{x}_{t-1}^j)\beta_{t-1}^j}{\sum_j w_{t-1}^j p(\mathbf{x}_t|\mathbf{x}_{t-1}^j)}.$$

Normalize

$$w_t^i = \frac{\widetilde{w}_t^i}{\sum_j \widetilde{w}_t^j}$$

$$w_t^i \beta_t^i = \frac{\widetilde{\rho}_t^i}{\sum_j \widetilde{w}_t^j} - w_t^i \frac{\sum_j \widetilde{\rho}_t^j}{\sum_j \widetilde{w}_t^j}$$

Update the parameter

$$\theta_t = \theta_{t-1} + \varepsilon \frac{\sum_j \widetilde{\rho}_t^j}{\sum_j \widetilde{w}_t^j}$$

7.2 Simulations

Now, armed with the ML scheme for online parameter estimation, we show how imperfect training and time-varying parameters can be handled.

Algorithm 6 For a parameter θ at time slot t

Start with previous time index's posterior and filter derivative, $\left\{w_{t-1}^{j}\right\}_{j=1}^{L}$ and $\left\{\beta_{t-1}^{j}\right\}_{j=1}^{L}$.

For each state i,

- $$\widetilde{w}_t^i = p_\theta(\mathbf{y}_t|x_t = i)\sum_j w_{t-1}^j p(x_t = i|x_{t-1} = j).$$

- $$\widetilde{\rho}_t^i = \widetilde{w}_t^i \frac{\partial}{\partial\theta}\log p_\theta(\mathbf{y}_t|x_t = i) + p_\theta(\mathbf{y}_t|x_t = i)\sum_j w_{t-1}^j p(x_t = i|x_{t-1} = j)\beta_{t-1}^j.$$

Normalize

$$w_t^i = \frac{\widetilde{w}_t^i}{\sum_j \widetilde{w}_t^j}$$

$$w_t^i\beta_t^i = \frac{\widetilde{\rho}_t^i}{\sum_j \widetilde{w}_t^j} - w_t^i\frac{\sum_j \widetilde{\rho}_t^j}{\sum_j \widetilde{w}_t^j}$$

Update the parameter

$$\theta_t = \theta_{t-1} + \varepsilon\frac{\sum_j \widetilde{\rho}_t^j}{\sum_j \widetilde{w}_t^j}$$

7.2.1 Tracking the Path Loss Exponent

This is a simplified case of Section 5.4. The transition model is a simple differential drive [33]

$$\mathbf{x}_t = \mathbf{x}_{t-1} + \mathcal{N}(\mathbf{0}, \sigma_u^2\mathbf{I}), \tag{7.15}$$

where $\mathbf{x}_t = \left[p_t^{(1)}\ p_t^{(2)}\right]^T$ consists of the two-dimensional coordinates at time t and $\sigma_u^2 = 0.1$. The area and the placement of the $N = 6$ sensors are the same as Section 5.4. The observation model is

$$y_{n,t} = -10\rho_t\log_{10}\|\mathbf{x}_t - \mathbf{r}_n\| + \mathcal{N}(0, \sigma_w^2), \tag{7.16}$$

where the path loss exponent ρ_t is now time-varying, $\mathbf{r}_n$ is still the location coordinate of the sensor n and the variance of the noise is $\sigma_w^2 = 2$. 10^4 time slots are simulated in MATLAB and the filter is tasked to track both the location coordinate of the target as well as the time-varying path loss exponent. The true path loss exponent varies according to $\rho_t = 2 + sin(\pi k/10^4)$, where k is the time index. Due to imperfect training, the filter is mistaken to believe $\rho = 3$.

We have already shown how the PF estimates the state in Chapter 5. Now, let us demonstrate the parameter estimation part. Once again,

$$p(y_{n,t}|\mathbf{x}_t^i) = \frac{1}{\sqrt{2\pi\sigma_w^2}} \exp\left(\frac{-(y_{n,t} + 10\rho \log_{10} \|\mathbf{x}_t^i - \mathbf{r}_n\|)^2}{2\sigma_w^2}\right) \tag{7.17}$$

and

$$p(\mathbf{y}_t|\mathbf{x}_t^i) = \prod_{n=1}^{n=N} p(y_{n,t}|\mathbf{x}_t^i). \tag{7.18}$$

As required by Algorithm 5,

$$p(\mathbf{x}_t^i|\mathbf{x}_{t-1}^j) = \frac{1}{2\pi\sigma_u^2} \exp\left(\frac{-(p_t^{(1),i} - p_{t-1}^{(1),j})^2}{2\sigma_u^2}\right) \exp\left(\frac{-(p_t^{(2),i} - p_{t-1}^{(2),j})^2}{2\sigma_u^2}\right) \tag{7.19}$$

and

$$\begin{aligned} &\frac{\partial}{\partial\rho} \log p(\mathbf{y}_t|\mathbf{x}_t^i) \\ &= \frac{-1}{\sigma_w^2} \sum_{n=1}^{N} (y_{n,t} + 10\rho \log_{10} \|\mathbf{x}_t^i - \mathbf{r}_n\|)(10 \log_{10} \|\mathbf{x}_t^i - \mathbf{r}_n\|) \end{aligned}. \tag{7.20}$$

$P = 300$ particles are used and we use a stepsize $\varepsilon = 10^{-4}$.

Figure 7.1 shows how well the ML scheme locks onto the true value of the path loss exponent and tracks it as it changes.

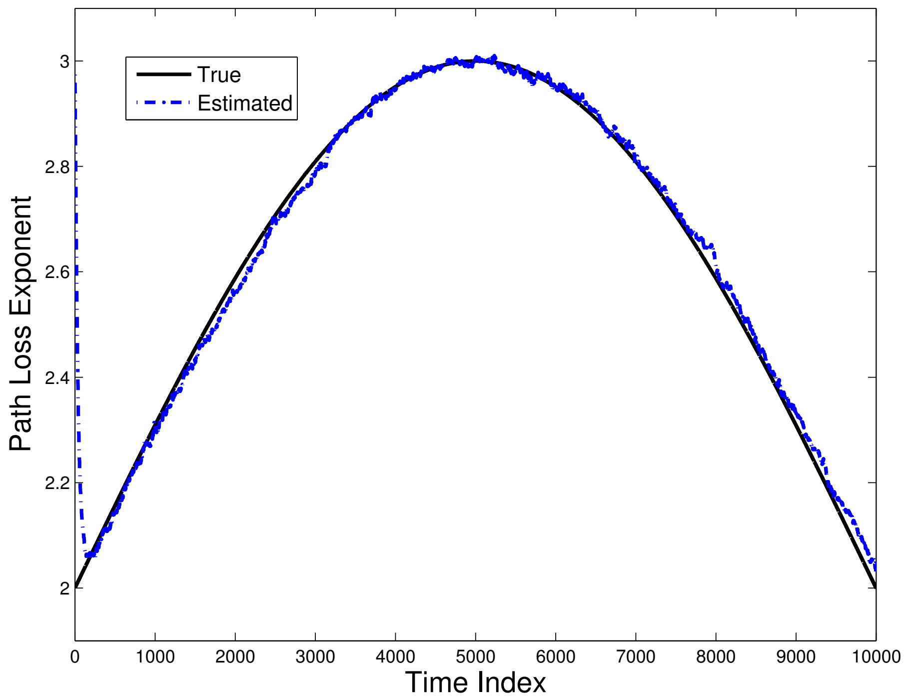

Figure 7.1: History of the Path Loss Exponent

7.2.2 Tracking Means of Gaussians

This is a simplified case of Section 6.4, i.e., the HMM filter with a finite number of cells. We have two cells $l = 1$ and $l = 2$ and the transition probabilities are

$$\begin{aligned} p(x_t = i|x_{t-1} = i) &= 0.95,\ \forall i \\ p(x_t = j|x_{t-1} = i) &= 0.05,\ \forall i \neq j \end{aligned}. \tag{7.21}$$

For simplicity, there is only $N = 1$ sensor and we have a scalar observation y_t. The Gaussian observation model is

$$p(y_n|x = l) = \frac{1}{\sqrt{2\pi\sigma_l^2}} \exp\left(\frac{-(y_t - \mu_l)^2}{2\sigma_l^2}\right). \tag{7.22}$$

We use the same noise variance for all cells, i.e., $\sigma_l^2 = 2\ \forall l$. The means of the Gaussians are time-varying

$$\begin{aligned} \mu_1 &= -70 + 5sin(\pi k/10^4) \\ \mu_2 &= -60 + 10sin(\pi k/10^4) \end{aligned}, \tag{7.23}$$

where k is the time index. Due to imperfect training, the filter is mistaken to believe $\mu_1 = -75$ and $\mu_2 = -55$. As required by Algorithm 6,

$$\frac{\partial}{\partial \mu_l} \log p(\mathbf{y}_t | x_t = i) = \frac{1}{\sigma_l^2}(y_t - \mu_l)\mathbf{I}(x_t = i), \tag{7.24}$$

where $\mathbf{I}(x_t = i) = 1$ if $x_t = i$ and $\mathbf{I}(x_t = i) = 0$ otherwise.

The stepsize is $\varepsilon = 10^{-2}$. 10^4 time slots are simulated in MATLAB and the filter is tasked with estimating both the state as well as the time-varying means of the Gaussians.

Figure 7.2 and Figure 7.3 show how well the filter locks onto the true values and tracks the changes.

7.3 Discussion and Future Work

We have used an online ML parameter tracking scheme to combat imperfectly trained and time-varying parameters. Nevertheless, we must point out its deficiencies. First, the observation model must be in closed-form and differentiable with respect to the parameters. This is already on top of the requirement that the Bayesian filter framework is only applicable to probabilistic formulations. Second, the stepsize is difficult to tune and it may be different for different parameters [38]. Furthermore, Algorithm 5 and Algorithm 6 are expensive as we must apply them for every parameter of interest. Nevertheless, this is the most promising way to deal with the deficiencies of the fingerprinting paradigm because it does not assume a number of known reference APs.

We have attempted to apply the scheme to experimental data. Our preliminary results have been clouded by several unforeseen difficulties. For instance, for the SGC studied in Chapter 6, we have attempted to alter the setup by covering the antennas with foils. However, doing so has not altered the sample means of the

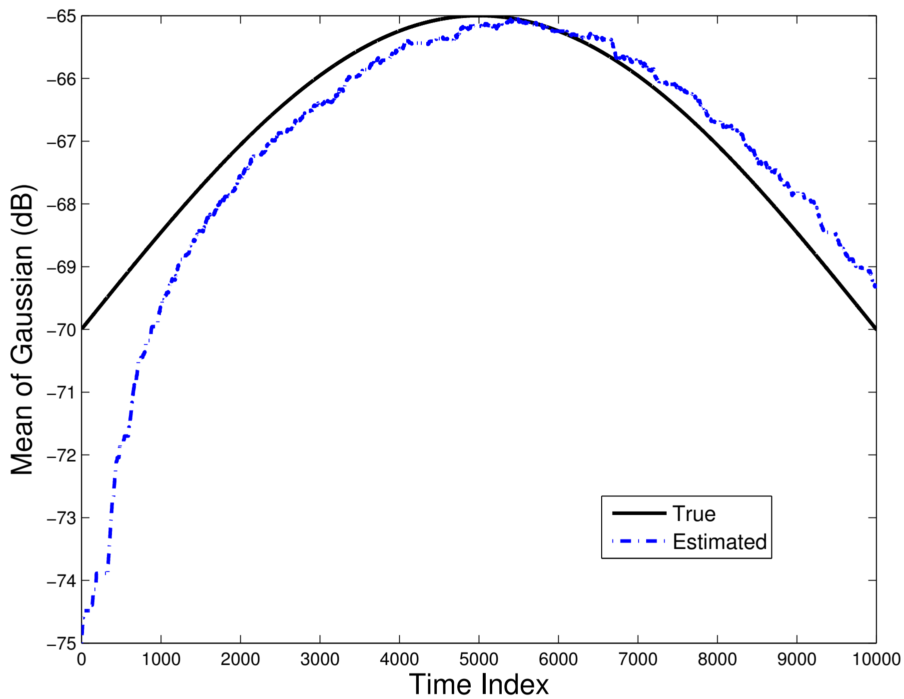

Figure 7.2: History of the Mean of the Gaussian for the First Cell

RSSI values in the cells dramatically. Intuitively, this is explained by the fact the the mean values are simple and robust. Small changes like human movements and covering the antennas do not change the conditions very much. In particular, since the model is only a rough approximation of reality, the modeling error is large. This is represented by large values for the variances for the SGC. In Equation 7.24, if a variance is large then the derivative is small and the scheme does not adapt to small changes. Furthermore, closed-form and differentiable probabilistic observation models are most likely not going to fit the empirical data well. The use of them are justified for simplicity and robustness. While these approximations have led to satisfactory performance results, we have not validated the use of online parameter estimation for them. Limited by the length and scope of this thesis, we must caution that this chapter contains preliminary results only and this online parameter tracking scheme may not be the panacea without further studies.

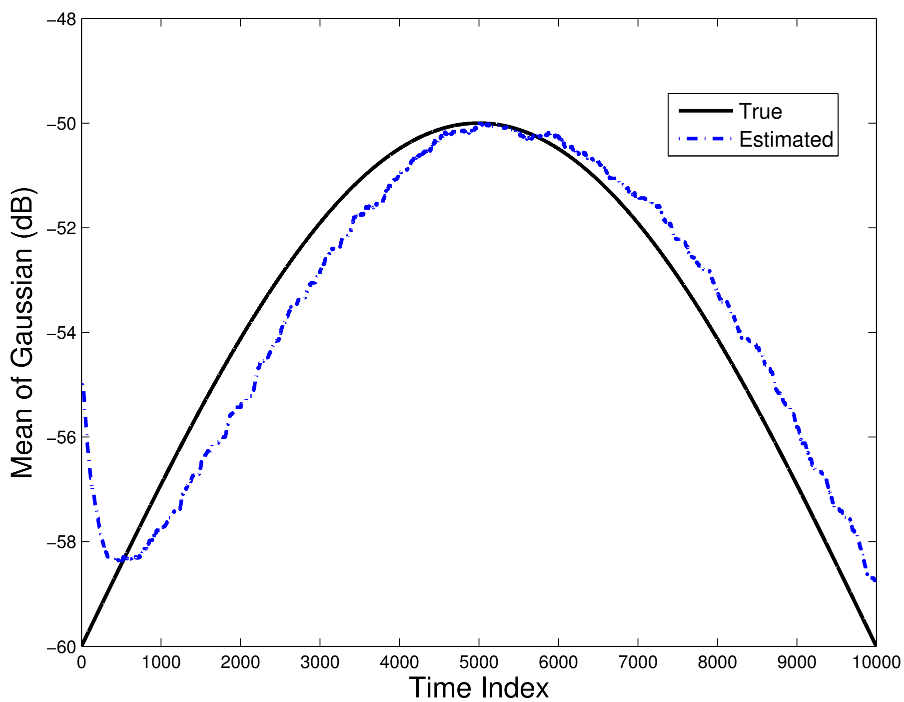

Figure 7.3: History of the Mean of the Gaussian for the Second Cell

Chapter 8

Conclusion and Future Work

Essentially, all models are wrong, but some are useful.
— George E.P. Box, Professor of Statistics

We have presented three main contributions to the fingerprinting paradigm, which aims to solve the inference problem of RSSI-based localization by constructing models from empirical data. From simulations as well as experimental data, we have shown that the PF is better than the UKF for solving the Bayesian filter problem using the regression approach. This is due to the fact that the UKF imposes unimodal Gaussian densities while realistic environments often introduce multimodal elements. Because there is not much discussion on the classification approach in the literature, we have emphasized its promising use based on experimental results. In particular, the SGC, which models the RSSI values within a specific region as Gaussian variables, is shown to be the preferred solution because it is written in closed-form and it is differentiable. It can be used under the Bayesian filter framework completely. Finally, we have demonstrated some preliminary results for using the Bayesian filter derivative to achieve online parameter tracking. While it is promising, more experimental results are needed to validate its use.

While the fingerprinting paradigm is powerful and has been studied well in the literature, there are two unsolved issues. First, the radiomap recorded in the offline training phase is static and there are no simple ways to adapt if the environment changes. We propose using a ML online parameter tracking scheme but it may not be the panacea. Second, the fingerprinting paradigm does not answer questions such as the theoretical limits on the accuracies or the spatial sampling interval required in the offline training phase or the choice of the cell division scheme required by many methods. The standard choice is to simply proceed with an arbitrary setup and evaluate its performance then repeat the procedure with a different setup until the performance is satisfactory. While this is partially alleviated by using simple and robust methods that require less human intervention and training, theoretical answers are still unknown. This thesis has used simple and robust methods such as the piecewise path loss model and the SGC. As elegantly said by Professor Box, there are literally wrong models as they do not fit the empirical data perfectly but they have been useful. We have been able to achieve satisfactory localization results but we have not been able to answer some fundamental questions. We believe that more theoretical analysis based on ray tracing and laws of Physics are required to answer these questions.

Bibliography

[1] N. Patwari, J. Ash, S. Kyperountas, A. Hero III, R. Moses, and N. Correal, "Locating the nodes: Cooperative localization in wireless sensor networks," *IEEE Signal Processing Mag.*, vol. 22, no. 4, pp. 54–69, July 2005. → pages 1

[2] A. Boukerche, H. Oliveira, E. Nakamura, and A. Loureiro, "Localization systems for wireless sensor networks," *IEEE Wireless Commun. Mag.*, vol. 14, no. 6, pp. 6–12, Dec. 2007. → pages 1, 4, 12

[3] M. Li and Y. Liu, "Rendered path: Range-free localization in anisotropic sensor networks with holes," *IEEE/ACM Trans. Networking*, vol. 18, no. 1, pp. 320–332, Feb. 2010. → pages 1

[4] D. Kelly, S. McLoone, T. Dishongh, M. McGrath, and J. Behan, "Single access point location tracking for in-home health monitoring," in *Proc. of 5th Workshop on Positioning, Navigation and Communication (WPNC)*, Hannover, Germany, Mar. 2008, pp. 23 –29. → pages 1, 7, 11

[5] P. Bahl and V. Padmanabhan, "RADAR: An in-building RF-based user location and tracking system," in *Proc. of 2000 IEEE INFOCOM, Nineteenth Annual Joint Conference of the IEEE Computer and Communications Societies*, vol. 2, Tel Aviv, Israel, Mar. 2000, pp. 775–784. → pages 1, 2, 6, 7, 38

[6] A. Beck, P. Stoica, and J. Li, "Exact and approximate solutions of source localization problems," *IEEE Trans. Signal Processing*, vol. 56, no. 5, pp. 1770 –1778, May 2008. → pages 4

[7] N. Sirola, "Closed-form algorithms in mobile positioning: Myths and misconceptions," in *Proc. of 7th Workshop on Positioning, Navigation and Communication (WPNC)*, Dresden, Germany, Mar. 2010. → pages 4

[8] T. S. Rappaport, *Wireless Communications*. Prentice Hall, 1996. → pages 5, 6, 13, 41

[9] G. Zàruba, M. Huber, F. Kamangar, and I. Chlamtac, "Indoor location tracking using RSSI readings from a single Wi-Fi access point," *Wireless Networks*, vol. 13, no. 2, pp. 221–235, Apr. 2007. → pages 5, 8

[10] F. Potorti, A. Corucci, P. Nepa, F. Furfari, P. Barsocchi, and A. Buffi, "Accuracy limits of in-room localisation using RSSI," in *Proc. of 2009 IEEE Antennas and Propagation Society International Symposium (APSURSI)*, Charleston, SC, USA, June 2009, pp. 1–4. → pages 5, 6, 16, 41

[11] S. Seidel and T. Rappaport, "914 MHz path loss prediction models for indoor wireless communications in multifloored buildings," *IEEE Trans. Antennas Propagat.*, vol. 40, no. 2, pp. 207–217, Feb. 1992. → pages 5, 6, 13, 41

[12] G. Chandrasekaran, M. Ergin, J. Yang, S. Liu, Y. Chen, M. Gruteser, and R. Martin, "Empirical evaluation of the limits on localization using signal strength," in *Proc. of 6th IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks (SECON)*, Rome, Italy, June 2009, pp. 1–9. → pages 6

[13] T. Mitchell, *Machine Learning*. McGraw-Hill, 1997. → pages 6, 38

[14] V. Vapnik, *The Nature of Statistical Learning Theory*, 2nd ed. Springer-Verlag, 2000. → pages 6, 38

[15] K. Kaemarungsi, "Design of indoor positioning system based on location fingerprinting technique," Ph.D. dissertation, University of Pittsbugh, Feb. 2005. → pages 6, 41, 43

[16] J. Yin, Q. Yang, and L. Ni, "Learning adaptive temporal radio maps for signal-strength-based location estimation," *IEEE Trans. Mobile Comput.*, vol. 7, no. 7, pp. 869–883, July 2008. → pages 7, 55

[17] A. Paul and E. Wan, "Wi-Fi based indoor localization and tracking using sigma-point Kalman filtering methods," in *Proc. of 2008 IEEE/ION Position, Location and Navigation Symposium (PLAN)*, Monterey, CA, USA, May 2008, pp. 646–659. → pages 7, 20

[18] C. Laoudias, P. Kemppi, and C. Panayiotou, "Localization using radial basis function networks and signal strength fingerprints in WLAN," in *Proc. of 2009 IEEE Global Telecommunications Conference (GLOBECOM)*, Honolulu, HI, USA, Nov. 2009, pp. 1 –6. → pages 7

[19] M. Brunato and R. Battiti, "Statistical learning theory for location fingerprinting in wireless LANs," *Computer Networks and ISDN Systems*, vol. 47, no. 6, pp. 825–845, Apr. 2005. → pages 7

[20] V. Honkavirta, T. Perala, S. Ali-Loytty, and R. Piche, "A comparative survey of WLAN location fingerprinting methods," in *Proc. of 6th Workshop on Positioning, Navigation and Communication (WPNC)*, Hannover, Germany, Mar. 2009, pp. 243–251. → pages 7, 41, 43

[21] C. Figuera, I. Mora-Jimenez, A. Guerrero-Curieses, J. Rojo-Alvarez, E. Everss, M. Wilby, and J. Ramos-Lopez, "Nonparametric model comparison and uncertainty evaluation for signal strength indoor location," *IEEE Trans. Mobile Comput.*, vol. 8, no. 9, pp. 1250 –1264, Sept. 2009. → pages

[22] T. Roos, P. Myllymäki, H. Tirri, P. Misikangas, and J. Sievänen, "A probabilistic approach to WLAN user location estimation," *International Journal of Wireless Information Networks*, vol. 9, no. 3, pp. 155–164, July 2002. → pages 7, 9, 41, 42

[23] C. Steiner and A. Wittneben, "Low complexity location fingerprinting with generalized UWB energy detection receivers," *IEEE Trans. Signal Processing*, vol. 58, no. 3, pp. 1756 –1767, Mar. 2010. → pages 8

[24] A. Haeberlen, E. Flannery, A. M. Ladd, A. Rudys, D. S. Wallach, and L. E. Kavraki, "Practical robust localization over large-scale 802.11 wireless networks," in *Proc. of 2004 ACM International Conference on Mobile Computing and Networking (MobiCom)*, Philadelphia, PA, USA, Oct. 2004, pp. 70–84. → pages 8, 41, 52, 56

[25] M. Arulampalam, S. Maskell, N. Gordon, and T. Clapp, "A tutorial on particle filters for online nonlinear/non-Gaussian Bayesian tracking," *IEEE Trans. Signal Processing*, vol. 50, no. 2, pp. 174–188, Feb. 2002. → pages 9, 19, 23, 26

[26] A. Doucet, N. de Freitas, and N. Gordon, *Sequential Monte Carlo Methods in Practice*. Springer-Verlag, 2001. → pages 11, 19, 23, 26

[27] C. Chui and G. Chen, *Kalman Filtering: with Real-Time Applications*, 4th ed. Springer-Verlag, 2008. → pages 11, 19, 23

[28] L. Rabiner, "A tutorial on hidden Markov models and selected applications in speech recognition," *Proc. IEEE*, vol. 77, no. 2, pp. 257–286, Feb. 1989. → pages 11, 19, 56

[29] S. Julier and J. Uhlmann, “Unscented filtering and nonlinear estimation,” *Proc. IEEE*, vol. 92, no. 3, pp. 401–422, Mar. 2004. → pages 19, 22

[30] G. Binazzi, L. Chisci, F. Chiti, R. Fantacci, and S. Menci, “Localization of a swarm of mobile agents via unscented Kalman filtering,” in *Proc. of 2009 IEEE International Conference on Communications (ICC)*, Dresden, Germany, June 2009, pp. 1–5. → pages 19

[31] E. Menegatti, A. Zanella, S. Zilli, F. Zorzi, and E. Pagello, “Range-only SLAM with a mobile robot and a wireless sensor networks,” in *Proc. of 2009 IEEE International Conference on Robotics and Automation (ICRA)*, Kobe, Japan, May 2009, pp. 8–14. → pages 20

[32] J. Rodas, C. Escudero, and D. Iglesia, “Bayesian filtering for a bluetooth positioning system,” in *Proc. of 2008 IEEE International Symposium on Wireless Communication Systems (ISWCS)*, Reykjavik, Iceland, Oct. 2008, pp. 618–622. → pages 20

[33] X. Rong Li and V. Jilkov, “Survey of maneuvering target tracking. Part I. Dynamic models,” *IEEE Trans. Aerosp. Electron. Syst.*, vol. 39, no. 4, pp. 1333–1364, Oct. 2003. → pages 20, 21, 61

[34] Y. Wu, D. Hu, M. Wu, and X. Hu, “Unscented Kalman filtering for additive noise case: augmented versus nonaugmented,” *IEEE Signal Processing Letters*, vol. 12, no. 5, pp. 357–360, May 2005. → pages 23

[35] C.-W. Hsu and C.-J. Lin, “A comparison of methods for multiclass support vector machines,” *IEEE Trans. Neural Networks*, vol. 13, no. 2, pp. 415–425, Mar. 2002. → pages 41

[36] C.-C. Chang and C.-J. Lin, *LIBSVM: a library for support vector machines*, 2010, software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm. → pages 46

[37] T.-F. Wu, C.-J. Lin, and R. Weng, “Probability estimates for multi-class classication by pairwise coupling,” *Journal of Machine Learning Research*, vol. 5, pp. 975–1005, Aug. 2004. → pages 51

[38] N. Kantas, A. Doucet, S. Singh, and J. Maciejowski, “An overview of sequential Monte Carlo methods for parameter estimation in general state-space models,” in *Proc. of 15th IFAC Symposium on System Identification (SYSID)*, Saint-Malo, France, July 2009. → pages 55, 64

[39] I. Collings and T. Ryden, "A new maximum likelihood gradient algorithm for on-line hidden Markov model identification," in *Proc. of 1998 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, vol. 4, Seattle, WA, USA, May 1998, pp. 2261–2264. → pages 57

[40] G. Poyiadjis, A. Doucet, and S. Singh, "Particle methods for optimal filter derivative: application to parameter estimation," in *Proc. of 2005 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, vol. 5, Philadelphia, PA, USA, Mar. 2005, pp. 925–928. → pages 57

[41] R. Martinez-Cantin, N. de Freitas, and J. Castellanos, "Analysis of particle methods for simultaneous robot localization and mapping and a new algorithm: Marginal-SLAM," in *Proc. of 2007 IEEE International Conference on Robotics and Automation (ICRA)*, Rome, Italy, Apr. 2007, pp. 2415–2420. → pages 57