TRACKING VEHICLES FROM MOBILE PHONE RECEIVED SIGNAL STRENGTH SEQUENCES

A Dissertation
Submitted to the Graduate Faculty
of the
North Dakota State University
of Agriculture and Applied Science

By

Charith Devinda Chitraranjan

In Partial Fulfillment of the Requirements
for the Degree of
DOCTOR OF PHILOSOPHY

Major Department:
Computer Science

November 2015

Fargo, North Dakota

NORTH DAKOTA STATE UNIVERSITY

Graduate School

Title

TRACKING VEHICLES FROM MOBILE PHONE RECEIVED SIGNAL STRENGTH SEQUENCES

By

Charith Devinda Chitraranjan

The supervisory committee certifies that this dissertation complies with North Dakota State University's regulations and meets the accepted standards for the degree of

DOCTOR OF PHILOSOPHY

SUPERVISORY COMMITTEE:

Dr. Anne Denton
Chair

Dr. Saeed Salem

Dr. Simone Ludwig

Dr. Amiy Varma

Approved:

11/10/2015
Date

Dr. Brian Slator
Department Chair

ABSTRACT

We address the problem of tracking vehicles from received signal strength (RSS) sequences generated by mobile phones carried in them. Our main objectives are to provide travel-time estimates for selected roads and provide personal navigation assistance when GPS is unavailable or undesirable. A mobile phone periodically measures the RSS levels from the associated cell tower and several (six for GSM) strongest neighbor cell towers. Each such measurement is known as an RSS *fingerprint*.

In Chapter 3, we propose *local* alignment of mobile phone RSS measurements to track vehicles. We use local alignment instead of the traditionally used global alignment to allow for vehicles changing roads. More specifically, we use local dynamic time warping to align the RSS sequence of a phone, to a reference sequence that we had collected for the relevant road.

Due to fluctuations in RSS levels and other effects, even at the same location, the set of cell towers reported in a fingerprint and their reported RSS levels vary over time. To model these variations, in Chapter 4.1, we propose a complete observation model for RSS fingerprints that specifies for each gird-location in the area of interest, the distribution of the probability of observing any fingerprint at that location. We then use it with a Dynamic Bayesian Network to track vehicles. Unlike traditional observation models, which model only the variation of the *RSS levels*, we model the variation of the *set of cells* reported in fingerprints as well.

Accurate estimation of the parameters of either traditional or our complete observation model requires recording fingerprints by driving on the roads of interest, which is tedious and expensive. Therefore, to avoid such driving, we propose unsupervised learning in Chapter 5 to estimate model parameters using RSS sequences of phone calls made by road-users.

Experiments with RSS data collected on five roads demonstrate that our proposed algorithms produce lower errors than relevant existing methods. Furthermore, application of our algorithms to real subscriber call traces produced travel-time estimates for a given road segment that were, on average, within 13% - 14% of travel-times computed through license plate recognition.

ACKNOWLEDGEMENTS

First and foremost, I would like to express my gratitude to my advisor, Dr. Anne Denton, for her valuable advise, continuous encouragement and providing me with an opportunity to be a part of her research group. I thank her for the constructive criticism she has provided me over the years without which, this thesis would have been a lot sloppier that it is today.

I would like to thank Dr. Saeed Salem, a member of my supervisory committee, for introducing useful research tools and sharing his research ideas with me, which made my work a lot easier. I thank Dr. Simone Ludwig and Dr. Amiy Varma for their valuable feedback and for agreeing to serve as members of my supervisory committee.

I thank all my friends in Fargo for the pleasant companionship. I thank the staff of the Computer Science Department at NDSU for helping me with various administrative tasks.

I would also like to extend my gratitude to all my teachers and university professors in Sri Lanka who laid the foundation for my academic career. I thank the staff of the University of Moratuwa for giving me time off from my teaching responsibilities and covering up for me while I was away for my studies. I would like to thank members of my extended family and friends who have supported me in numerous endeavors.

Last but certainly not least, I thank my parents, my wife and my brother for the extreme breath and depth of the support they have provided me throughout my life.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

DBN ..... Dynamic Bayesian Network

EM ..... Expectation Maximization

GPS ..... Global Positioning System

GSM ..... Global System for Mobile Communications

HMM ..... Hidden Markov Model

NMR ..... Network Monitoring Report

RSS ..... Received Signal Strength

1. INTRODUCTION

In this work, we address the problem of tracking vehicles from received signal strength (RSS) traces generated by mobile phones carried in them. Our objective is to provide travel time/speed estimates for selected roads and assist in personal navigation when GPS is unavailable (e.g., urban canyons) or undesirable to use (e.g., when battery life is critical). Other applications of such vehicle tracking include location-based services such as mobile-based advertising and local weather or traffic alerts [1].

With global mobile-cellular penetration approaching 100% [2], mobile phones have become potential sensors to gather large amounts of useful data for vehicle tracking. Furthermore, the required signal strength data can be collected under normal operations of the cellular network without an additional overhead.

A mobile phone conforming to the standards of the Global System for Mobile Communications (GSM) is required to measure the strengths of the signals it receives from the associated cell tower and the six strongest neighbor cell towers, commonly referred to as a received signal strength *fingerprint*, twice every second. A fingerprint recorded at time t, denoted by R_t, is a set of ordered pairs of the form (c, y_t^c), where y_t^c is the strength of the signal received from the cell tower with the identification code (cell ID) c, at time t. Figure 1.1 shows an example of a fingerprint. Additionally, to help ensure uninterrupted communication, a phone in active mode is required to transmit the aforementioned RSS fingerprints back to the network in messages referred to as network monitoring reports (NMRs) [3].

Methods used for tracking vehicles using such RSS sequences can be classified under two approaches. They are, time series alignment and recursive Bayesian estimation. In this work, as discussed below, we address several problems with the existing methods under each approach, and propose our solutions to them.

1.1. Time series alignment approach

This approach is based on aligning the RSS sequence from a mobile phone to a reference RSS sequence that had been collected previously in a training phase. However, existing methods

cannot handle vehicles changing roads as they use global sequence alignment. In Chapter 3, we solve this problem by using local sequence alignment [4].

Furthermore, the current alignment-based methods use only the signal strength levels of the strongest cell to perform the alignment, neglecting useful neighbor cell information. We include neighbor cell signal strengths while performing the entire alignment.

1.2. Recursive Bayesian estimation approach

Examples of methods that follow this approach include, Kalman filtering [5], Dynamic Bayesian Networks (DBNs), and Hidden Markov Models [6, 7, 8, 9]. A critical component of these methods is the observation model, which models for each grid-location in the area of interest, the probability distribution of the variation of RSS levels at that location. In GSM, each fingerprint contains RSS levels from up to seven cell towers (associated + six strongest neighbors). However, due to fluctuations in signal strength levels and other effects, even at the same location, the set of cell towers included in fingerprints may *not* remain constant over time. Observation models used in existing methods only model the variation of signal strength levels but not the variation of the set of cell towers present in the fingerprint. We model both types of variations and propose a more complete and improved observation model in Chapter 4.

Furthermore, accurate estimation of the parameters of the observation models of either existing methods or the improved one that we propose, requires recording RSS readings by driving on the roads of interest [6, 7, 8, 9]. This process, commonly referred to as war-driving, is both tedious and expensive. The parameters can be estimated using radio propagation models, which is easier and cheaper but less accurate. Therefore, in Chapter 5, we propose the use of unsupervised learning with a dynamic Bayesian network (DBN) to learn model parameters without war-driving, with the aid of signal strength traces of phone calls made by road-users.

All algorithms presented in Chapters 3, 4 and 5 operate in two phases; a training phase, where several model parameters are estimated, and a tracking phase, where the tracking of vehicles takes place. Table 1.1 shows a comparison between the algorithms presented in the different Chapters with regard to the tracking and training phases.

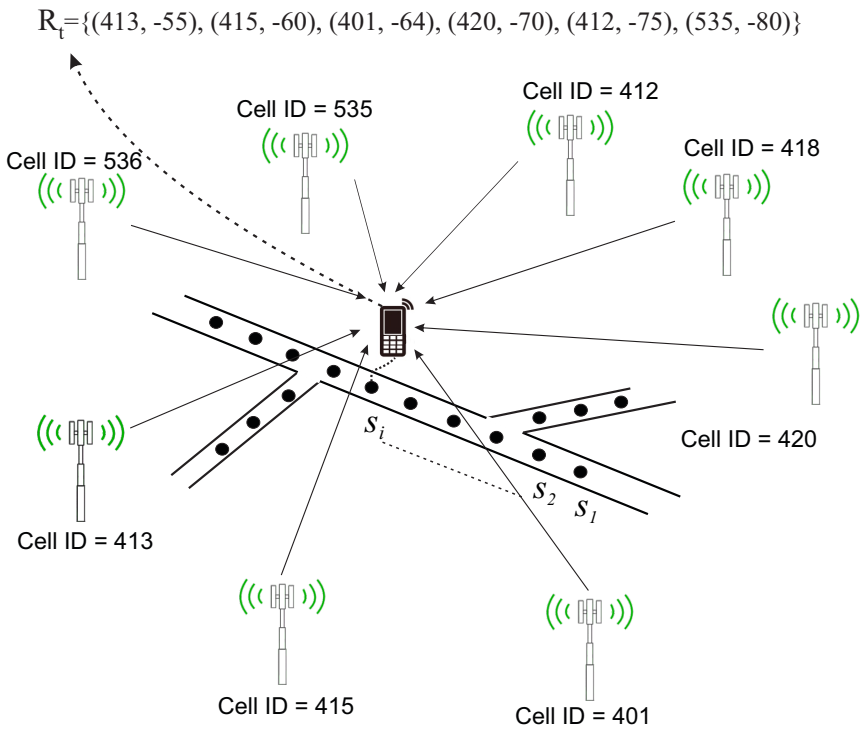


Figure 1.1. Received signal strength fingerprint: Suppose the area of interest has been divided in to grid-location (black dots) and a phone is at location s_i at time t. Then, for example, fingerprint R_t in the figure reports signal strength levels -55dbm, -60dbm, -64dbm, -70dbm, -75dbm and -50dbm as received from cells with IDs 413, 415, 401, 420, 412, and 535 respectively, at location s_i, at time t. Note that R_t does not include signal levels from cells 418 and 536. As explained in Chapter 4, such omissions can happen in practice.

In Chapter 6, we present a fast nearest-neighbor-based technique to reduce the execution time of the algorithms in the preceding Chapters.

Since radio signals are affected by various forms of noise and obstructions as they propagate through the environment, tracking vehicles from RSS measurements is more challenging than using other types of sensors such as as loop detectors, cameras and Global Positioning System(GPS)-based devices. However, loop detectors and cameras are expensive to install and maintain [10, 11]. GPS-based devices, although more precise, have less penetration among the public [10, 11, 12], perform poorly in urban areas with high-rise buildings [8, 6], incur additional costs when data is transmitted to a centralized system [11], and are also high in power consumption [12, 13, 6], which may cause users to keep them turned off even if GPS is available on their mobile phone.

Table 1.1. Comparison of algorithms

	Chapter 3	Chapter 4	Chapter 5
Tracking	Pair-wise sequence alignment: The signal strength sequence of a phone call is aligned with a reference signal sequence.	Sequence-to-model alignment: The signal strength sequence of a phone call is aligned with a model of signal strength variation and vehicular motion (DBN).	Same as Chapter 4.
Training	Requires driving on the road to collect a reference trace. Does *not* build a model of signal strength variation.	Requires driving on the road several times to collect training fingerprints. We use this data to build a probabilistic model of signal strength variation. I.e., observation model of the DBN.	Does not require driving on the roads to collect training data. We Learn the same model of signal strength variation as in Chapter 4 but do so using signal strength sequences generated by road-user phone calls instead of fingerprints collected by driving on the road.

2. LITERATURE REVIEW

Vehicle tracking from mobile phone signals fundamentally relies on estimating the position (localization) of the phone. One of the most basic ways of localizing a phone is to estimate the distance to three or more cell towers based on the strength of the signal as received from the respective towers and triangulate the location. This requires the use of some model for the propagation of electromagnetic waves in the environment. However, its localization accuracy is limited by how well this model fits the actual propagation environment and suffers from deviations caused by shadowing [14]. Other techniques such as Signal Time of Arrival (TOA), Observed Time Difference of Arrival (OTDOA), and Enhanced Observed Time Differences (E-OTD) which are based on the time a signal takes to reach the phone from a cell tower are also used to estimate the position of the phone. These techniques, however, are affected by multipath propagation of signals [14].

Several approaches have been proposed to improve the accuracy of mobile phone localization. Database correlation method (DCM) [15] is one such approach, which uses a collection of signal strength readings and/or Timing values stored in a database. In this method, field measurements are taken at each point on a predefined gird in the geographical area of interest and are stored in a database. Each measurement, referred to as a *fingerprint*, includes the signal strength levels from multiple cell towers in the neighborhood (typically, the serving cell and six neighboring cells). Later, when the location of a mobile phone needs to be estimated, its fingerprint is compared to those in the database to find the best match whose location is returned as the estimated location. Further improvements on DCM are proposed in [16, 17]. Ferris et al. [18] used Gaussian processes (GPs) to generate a likelihood model for signal strength levels to localize mobile devices. This technique also relies on a set of training data points for calibration, but can work with a more sparse grid of points compared to DCM. Experimental results in [18] show that GPs are better for rural and suburban areas compared to DCM but not as good for urban areas.

The methods discussed so far are general localization algorithms not specific to tracking moving phones/vehicles. However, algorithms can take advantages of typical dynamics of vehicular movement for better tracking accuracy. In the rest of this Chapter we discuss methods specially designed for tracking vehicles from RSS traces.

Chandrasekaran et al. [12, 19] proposed two new algorithms to track the speed of a moving vehicle by comparing the RSS trace generated by a mobile phone with a reference RSS trace that had been collected previously in a training phase, and showed that they are better than DCM for speed estimations. The first algorithm [12] computes the correlation between a test trace and multiple copies of the reference trace stretched or compressed uniformly at constant factors, and selects the copy which is most correlated with the test trace. The uniform scaling was a major limitation in this method as the predicted speeds were limited to constant multiples of reference speeds. This problem of uniform scaling was overcome in their second algorithm [19] where they used dynamic time warping to allow non-uniform scaling along the time axis so that local variations in speed could be better predicted. However, both of these methods ignored signal strengths from neighboring cell towers in their correlation/alignment computations which as we demonstrate in this work, reduced their accuracy when used to track the position of a moving vehicle. Further, they were not able compare a subsequence of the test trace with the reference trace.

Several algorithms based on recursive Bayesian estimation have been proposed to track vehicles from mobile phone RSS traces. HMM-based methods [6, 8] have the area of interest divided into discrete points or grid-cells, which are treated as hidden states. The state transition matrix of the HMM models the probabilities of transitioning from each grid-cell to each of the others. The observation model of the HMM models the distribution of RSS readings at each grid-cell. Then as in standard HMMs, a given sequence of RSS readings is mapped to the most likely sequence of grid-cells. The specific algorithms all share the same fundamental concept, however, differ in the way the transition and observation models are constructed. In [6], the transition probabilities are defined using the reciprocal of the Manhattan distance between grid-cells. The observation probabilities (emission scores) are defined using the Euclidean distance between observed and training RSS fingerprints over the cell towers common to both fingerprints that is also weighted by the number of common cell towers. [9] allows transition only among adjacent grid-cells and assumes uniform transition probabilities. Their observation model computes the probability of an observed fingerprint in a given grid-cell by using RSS histograms constructed with fingerprints collected in that grid-cell during a training phase. In [8], the transition probabilities are constructed assuming that the vehicle moves along the road with constant speed so that the transition probability between two grid-cells is proportional to the distance between them. Their observation model is

similar to that of [9], however, they fit Gaussian distributions to the RSS histograms and use them to compute the emission probabilities.

Kalman filtering is a classic tool used for tracking vehicles or other objects. Olama et al. [5] used Kalman filtering to track the position and speed of a vehicle through a mobile network. Their evaluation, however, is limited to simulated data.

Several attempts have been made to estimate traffic parameters such as density, travel-time and speed using cellular network signaling information. To describe a few of them, the STRIP(System for TRaffic Information and Positioning) [20] was carried out in France to estimate travel-time/speed by using mobile phone signal strength data. Results from the project show that Inter-city high way speeds were overestimated by 24-32% [21, 22]. Smith et al. [23] have used 160 phone calls to estimate traffic speed for 10-min intervals in the Washinton DC, USA area. They were able to estimate traffic speeds with a mean error of 6-8 mph with some intervals having errors greater than 20 mph [21, 24]. Maerivoet et al. [25] and Liu et al. [26] used call handover data to estimate travel speed in Antwerp, Belgium and Minnesota, USA respectively. Handover data provides less resolution compared to signal strength and hence, is suitable largely for free flow traffic at high speeds only [21].

Most of the previous projects on traffic parameter estimation from cellular signaling data have either used handovers or triangulation based on signal strength and/or signal timing measurements. Use of techniques such as [6, 9, 19] and those presented in our work that are expected to perform better than triangulation or DCM, has the potential to improve the estimation accuracy of such projects. We would like to direct interested readers to [21] for a thorough review of cellular network-based traffic estimation projects.

3. TRACKING VEHICLE TRAJECTORIES BY LOCAL DYNAMIC TIME WARPING OF MOBILE PHONE SIGNAL STRENGTHS AND ITS POTENTIAL IN TRAVEL-TIME ESTIMATION

3.1. Introduction

Time series alignment approach for vehicle tracking from mobile phone received signal strength levels is based on comparing the signal strength sequence from a mobile phone with a known reference signal strength sequence [12, 19]. A reference sequence is a signal strength sequence recorded for a given road segment along with the geographical location (e.g. GPS coordinates) at which each fingerprint was recorded. These methods rely on the observation that the signal strength profile for a given geographical path stays adequately stable over time. Hence, as shown in Figure 3.1, the signal strength sequence from a mobile phone (referred to as *test sequence* hereafter) can be stretched or compressed along the time axis until it best matches the reference.

However, we have identified two main problems with the state of the art alignment-based methods that reduce their effectiveness. The contributions of this paper, as listed below, are the solutions we propose to overcome these problems.

We align multivariate time series: Typical network monitoring reports (NMRs) transmitted by an active mobile phone include the received signal strength readings from the serving cell tower and 6 neighboring cell towers. However, the current alignment-based methods use neighbor cell signal strength only to determine the starting point of an alignment. Thereafter, only the signal strength readings of the strongest cell[1] are used to perform the actual alignment. Therefore, useful neighbor cell information is neglected while performing the alignment.

As a solution to this, we use neighbor cell signal strength while performing the entire alignment. This makes the signal strength sequences multivariate time series where each sample point is a vector of signal strength values instead of a scalar (Section 3.2). As an example, Figure 3.1(b) shows the alignment of bivariate signal traces (Typical traces have 7 or more dimensions).

[1]This is not specifically mentioned in their publication. We verified it by contacting the authors.

We use local sequence alignment to allow for vehicles that change roads: Current methods use semi-global[2] alignment and assume that a test sequence will align, over its entire length, with some reference sequence. However, a vehicle can change roads resulting in a test signal sequence that should not align entirely with any given reference sequence. Instead, as shown in Figure 3.2, a test sequence involving a road change should partially align with the different reference sequences for the roads traveled.

We developed local dynamic time warping (LDTW) to solve this problem by allowing partial alignment of a test sequence to a reference sequence. In particular, we show how the distance between signal vectors can be converted to a similarity score suitable for local alignment (Section 3.3).

The algorithm presented in [19] is one of the top performing time series alignment-based algorithms. The basic technique used in [19] is to align the derivative of the signal strength sequence from a mobile phone with a reference sequence and estimate the path of the mobile phone as the GPS coordinates of the series of aligned points on the reference sequence. The reference sequences can be collected in a separate training phase or as part of drive tests usually done by mobile service providers to evaluate their coverage. For example, let $X = [x_1, x_2, ..., x_{10}]$ and $Y = [y_1, y_2, ..., y_7]$ be a reference and a test sequence respectively as shown in Figure 3.1 (a). Each sample point $x_i \in X$, $y_j \in Y$ represents the strength of the signal received from a given cell tower, at a given instant in time. In this idealistic example, if Y is stretched and compressed w.r.t to X, along the time axis as shown in Figure 3.1 (a), the match in signal strength levels between X and Y is considered perfect. According to this alignment, the geographical path of Y is estimated as $[gps(x_2), gps(x_4), gps(x_6), gps(x_7), gps(x_7),\ gps(x_8), gps(x_8)]$, where $\forall x_i \in X$, $gps(x_i)$ are the GPS coordinates corresponding to x_i. Note that in [19], the derivatives of the signal traces are used to perform the alignment. The basic idea, however, is the same as explained above. Further, in [19], univariate time series representing the signal strength from a given cell is used. In our work, we use signal strengths from multiple cells forming a multivariate time series. For example, if signal strengths from two cell towers were used, the alignment would look similar to that in Figure 3.1 (b).

[2]Entire length of the test sequence is aligned to a subsequence of the reference trace

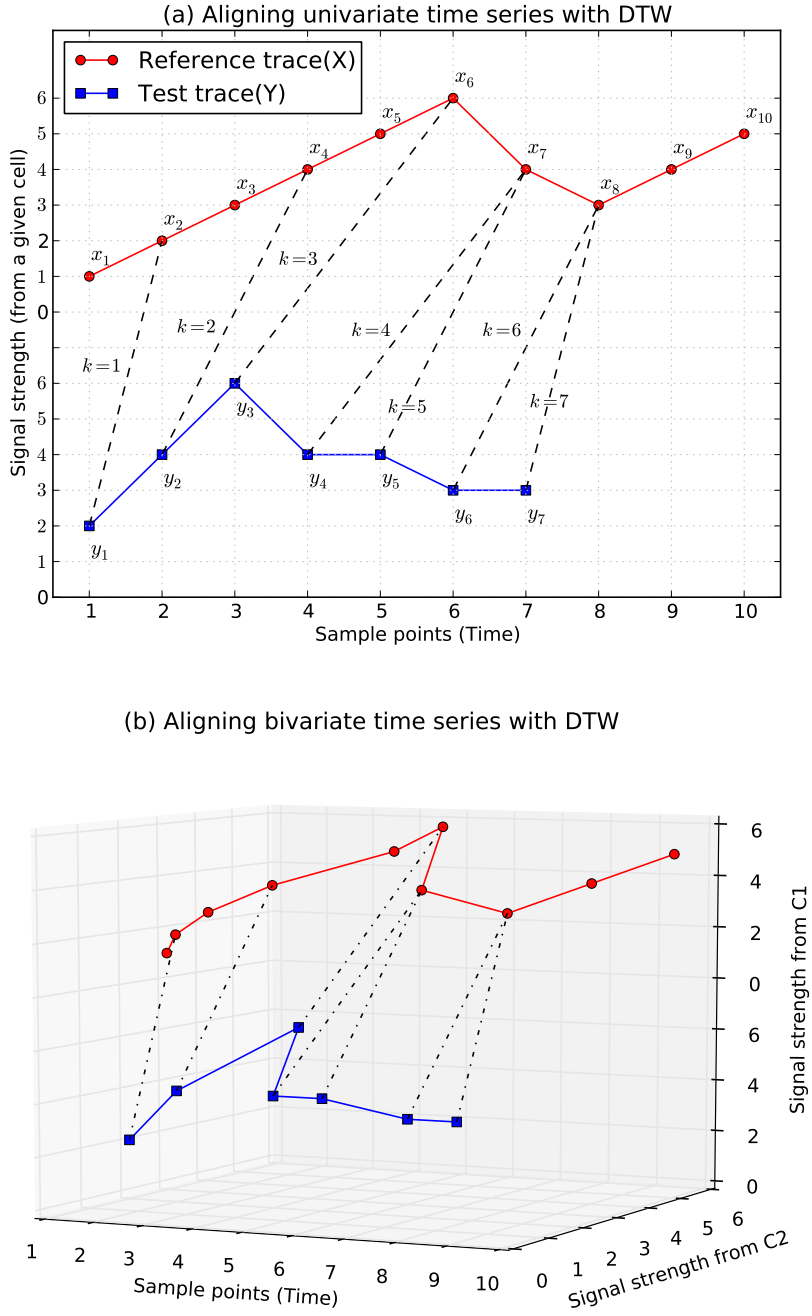


Figure 3.1. (a) Example of signal strength sequence considering only the signal strength from one cell tower (e.g. serving cell) and their alignment using DTW. 'k'represents the common index used in Equation 3.2.1.
(b) Alignment of signal traces considering signal strengths from two cells C1 and C2 (e.g. serving cell and one of the neighboring cells).

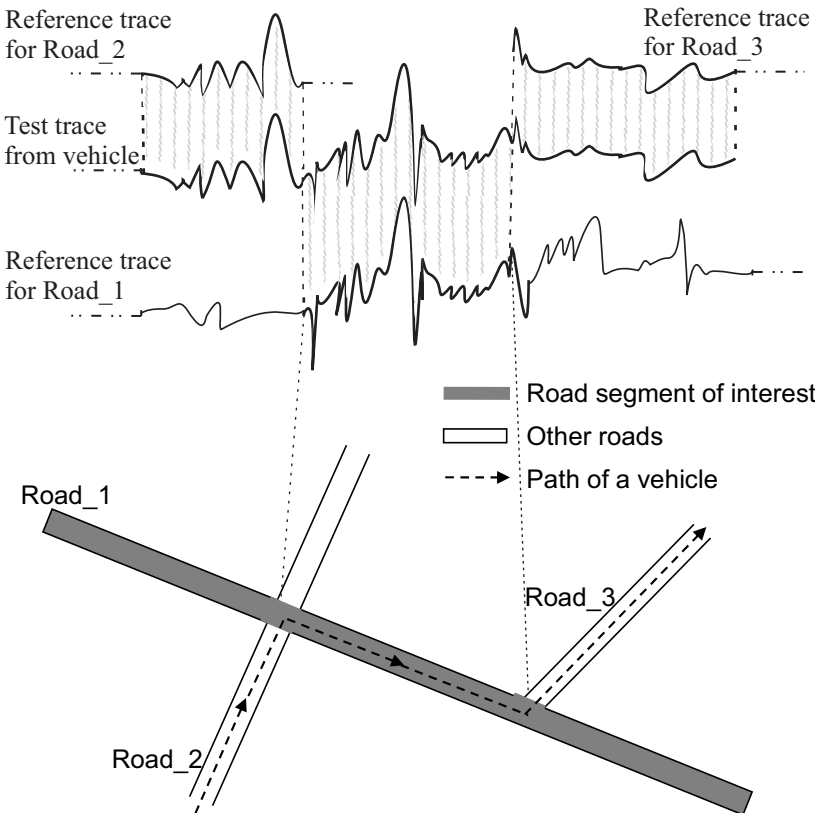


Figure 3.2. Path of a vehicle on a road network (bottom), and idealistic alignment of corresponding signal traces (top)

3.2. Dynamic Time Warping Signal Strength Traces

Dynamic Time Warping (DTW) is a popular technique for comparing series of values, especially time series. Given a pair of time series, the basic idea behind DTW is to stretch or compress them along the time axis so as to achieve minimum (maximum) distance (similarity) between their values. To elaborate, let $X = [x_1, x_2, ...x_i, ...x_M]$ and $Y = [y_1, y_2, ...y_j, ...y_N]$ be two time series. Sample points of X and Y are indexed by $i \in \{1, 2, ..., M\}$ and $j \in \{1, 2, ..., N\}$ respectively. Let ϕ_x and ϕ_y be two functions (called warping functions) that map the indices i of X and j of Y respectively, to a common axis indexed by $k \in \{1, 2, ..., T\}$, i.e.,

$$\begin{aligned} \phi_x(k) &= i \\ \phi_y(k) &= j, \forall k \in \{1, 2, ..., T\} \end{aligned} \tag{3.2.1}$$

and satisfy the monotonicity constraint

$$\phi_x(k+1) \geq \phi_x(k), \forall k \in \{1, 2, ..., T-1\}$$
$$\phi_y(k+1) \geq \phi_y(k)$$

Let ϕ aggregate ϕ_x and ϕ_y as,

$$\phi(k) = (\phi_x(k), \phi_y(k)) \tag{3.2.2}$$

Then the total distance between X and Y under the aggregate warping function ϕ, denoted by $D(X, Y, \phi)$, is defined as,

$$D(X, Y, \phi) = \sum_{k=1}^{T} d(x_{\phi_x(k)}, y_{\phi_y(k)}), \tag{3.2.3}$$

where $d(x, y)$ is some function that can measure the distance between elements x and y, e.g. Euclidean distance if x and y are vectors in an Euclidean space. The goal of DTW is to find the warping function ϕ so as to minimize $D(X, Y, \phi)$, i.e., find $\phi^* = \arg\min_\phi D(X, Y, \phi)$.

Figure 3.3 shows the minimum distance DTW alignment for the same example in Figure 3.1 described earlier. In this case the warping functions are as follows, and the distance, $D(X, Y, \phi^*) = 0$.

$$\phi_x(1) = 2, \phi_x(2) = 4, \phi_x(3) = 6,$$
$$\phi_x(4) = 7, \phi_x(5) = 7, \phi_x(6) = 8, \phi_x(7) = 8,$$
$$\phi_y(k) = k, \forall k \in \{1, 2, ..., 7\}$$

In its most conventional form, DTW is performed to align two sequences over their entire lengths (global alignment). However, in signal sequence applications, among others, the reference sequence is longer than the test sequence. Therefore, a variant called semi-global subsequence DTW is used, where the *entire* length of the test sequence is aligned to some subsequence of the reference sequence [27, 28, 29]. The example we describe in Figure 3.3 also resembles subsequence DTW. Please refer to [27, 30] for further details on DTW and its variants. In contrast, the local alignment we propose in Section 3.3 allows a *subsequence* of the test sequence to be aligned to a subsequence of the reference.

With our fingerprint sequences, each $x_i \in X$ is a set of ordered pairs of the form $\{(c_x^0, r_x^0), (c_x^1, r_x^1), ..., (c_x^6, r_x^6)\}$, where $\forall n \in \{0, 1, ..., 6\}, r_x^n$ is the strength of the signal from the cell tower with ID c_x^n, as received by the device (phone) used to collect the reference trace X. Cell IDs include the identification codes of the serving cell tower and 6 neighboring cell towers. Similarly for the test sequence, each $y_j \in Y$ takes the form $\{(c_y^0, r_y^0), (c_y^1, r_y^1), ..., (c_y^6, r_y^6)\}$.

The distance between two sample points x_i and y_j, i.e., $d(x_i, y_j)$, is defined as the normalized Euclidean distance using signal strength readings from cell towers which are present in either x_i or y_j or both as shown below. Let C_u be the the union of cell IDs present in x_i or y_i, and let $\vec{x_i}$ and $\vec{y_i}$ be signal strength vectors defined over C_u. Let β be a very low signal strength level used to represent the strength of the signal from a cell tower that was not reported by the phone (may be because it was out of range or 7 others were stronger). Then $d(x_i, y_j)$, is defined as below.

$$
\begin{aligned}
&C_u = \{c_i : (c_i, r_x^i) \in x_i \; or \; (c_i, r_y^j) \in y_i\} \\
&\vec{x_i} = (r_x^0, .., r_x^p, .., r_x^{|C_u|}), where \; r_x^p = \begin{cases} r_x^q \; , if \; (c_p, r_x^q) \in x_i \\ \beta \quad , otherwise \end{cases} \\
&\vec{y_i} = (r_y^0, .., r_y^p, .., r_y^{|C_u|}), where \; r_y^p = \begin{cases} r_y^q \; , if \; (c_p, r_y^q) \in y_j \\ \beta \quad , otherwise \end{cases} \\
&\forall p : 0 \leq p \leq |C_u| \\
&d(x_i, y_j) = \frac{Euclidean\_norm(\vec{x_i} - \vec{y_i})}{|C_u|}
\end{aligned}
\tag{3.2.4}
$$

Note that in subsequent sections of this paper, whenever the phrase *signal distance* is used, it refers to the normalized Euclidean distance defined above.

For any given pair of signal strength sequences X and Y, we define a distance matrix D_{XY}, where each element $D_{XY}[i, j] = d(x_i, y_j)$, and use it as input to perform subsequence DTW [27]. The result of subsequence DTW is the minimum distance warping path $\phi(k) = (\phi_x(k), \phi_y(k)), \forall k \in \{1, ..., T\}$. Then, the geographical path of the test sequence Y, is estimated to be $[gps(x_{\phi_x(k)}), ..., gps(x_{\phi_x(T)})]$.

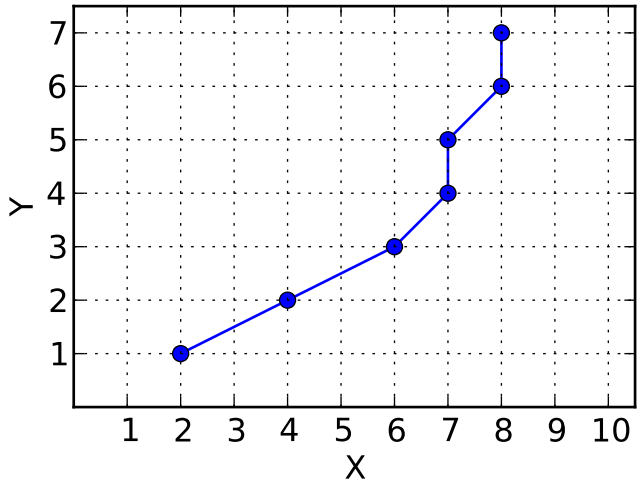


Figure 3.3. Dynamic Time Warping path for the example in Figure 3.1

3.3. Local Dynamic Time Warping

Reference sequences are typically collected for road segments of interest. However, for instance, a vehicle can enter and leave a given road segment as shown in Figure 3.2. This will result in a test sequence whose mid portion will align with the reference sequence for the road of interest, but not the ends. There can be other types of road changes as well which will produce test sequences where only certain subsequences will align with a given reference sequence. Semi-global DTW described in the previous section will be unable to align this kind of traces as it tries to align the *entire* length of the test sequence to some portion of the reference sequence. We propose *local* DTW to handle such sequences. The idea behind local alignment is to align a *subsequence(s)* of the test sequence to a *subsequence(s)* of the reference sequence.

Local sequence alignment is a well studied problem with regard to biological sequences (DNA and proteins), and usually employs the famous Smith-Waterman algorithm [31].

Given two sequences $X = [x_1, x_2, ...x_i, ...x_M]$ and $Y = [y_1, y_2, ...y_j, ...y_N]$, the goal of local alignment is to find the best alignment between a subsequence of X and a subsequence of Y. To explain more precisely, let $X_{s_x e_x} = [x_{s_x}, ..., x_{e_x}]$, and $Y_{s_y e_y} = [y_{s_y}, ..., y_{e_y}]$ be contiguous subsequences of X and Y respectively, with $1 \leq s_x \leq e_x \leq M$ and $1 \leq s_y \leq e_y \leq N$. Then, the optimal local alignment between X and Y refers to the values of s_x, e_x, s_y, e_y and the warping function ϕ that minimize, $D(X_{s_x e_x}, Y_{s_y e_y}, \phi)$. ($D$ is defined as in Equation 3.2.3).

If a suitable similarity function $s(x_i, y_j)$ is defined to measure the similarity between elements x_i and y_j, Equation 3.2.3 can be re-written as,

$$S(X, Y, \phi) = \sum_{k=1}^{T} s(x_{\phi_x(k)}, y_{\phi_y(k)}) \tag{3.3.1}$$

Now, the optimal local alignment between X and Y refers to the values of s_x, e_x, s_y, e_y and the warping function ϕ that maximize, $S(X_{s_x e_x}, Y_{s_y e_y}, \phi)$.

With a combination of dynamic time warping and the Smith-Waterman algorithm, local alignment of general time series can be attempted as follows.

Let V be a matrix (called the dynamic programming matrix) as shown in Figure 3.4, where each entry $V(i, j)$ is recursively defined as below.

Initialization: for $0 \leq i \leq M$ and $0 \leq j \leq N$

$$V(i, 0) = 0, V(0, j) = 0 \tag{3.3.2}$$

Recurrence: for $1 \leq i \leq M$ and $1 \leq j \leq N$,

$$V(i, j) = Max \begin{cases} V(i, j-1) & + & s(x_i, y_j) & (i) \\ V(i-1, j-1) & + & s(x_i, y_j) & (ii) \\ V(i-1, j) & + & s(x_i, y_j) & (iii) \\ 0 & & & (iv) \end{cases} \tag{3.3.3}$$

Matrix V can be computed iteratively, each row (column) at a time starting with the first row (column). Another matrix T, with elements defined as below, is used to record, for each cell (i, j) in V, whether the cell $(i, j-1), (i-1, j-1)$ *or* $(i-1, j)$ was used to derive $V(i, j)$, i.e., which one of the options i through iv in Equation 3.3.3 was used to assign the value for $V(i, j)$.

Trace-back: for $1 \leq i \leq M$ and $1 \leq j \leq N$,

$$T(i, j) = \underset{(u,v) \in \{(i,j-1),(i-1,j-1),(i-1,j)\}}{\arg\max} V(u, v) \tag{3.3.4}$$

In Figure 3.4, the three arrows pointing out of cell (i, j) show the different options $(i - iii)$ available for the calculation of $V(i, j)$. If none of these options gives a positive value, a zero will

be written in the cell, using option (iv). In this example, we *assume* the diagonal option (solid arrow) provided the highest value for $V(i,j)$, hence, $T(i,j) = (i-1, j-1)$ for this instance. Note that Equation 3.3.3 resembles the symmetric step pattern used in DTW. However, it can easily be modified to account for other step patterns.

The above equations make sure that $V(i,j)$ is the score of the optimal local alignment between prefixes $[x_1, ..., x_i]$ and $[y_1, ..., y_j]$. Let $(i^*, j^*) = \arg\max_{(i,j)} V(i,j)$. Then, $V(i^*, j^*)$ is the score of the optimal local alignment between X and Y. At each matrix cell (i,j), the previous cell that was used to compute $V(i,j)$ is recorded in $T(i,j)$. The optimal local alignment between X and Y is found by using the entries in T to trace the path back from the $(i^*, j^*)^{th}$ cell until a cell with value 0 is reached. In Figure 3.4, assume that cell $(i+1, j+1)$ contains the highest value $V(i^*, j^*)$, and that $V(i-3, j-2) = 0$. Then, assuming that the trace-back from (i^*, j^*) is as shown by solid arrows, the optimal local alignment between X and Y is given by,

$$\begin{array}{cccccc} y_{j-2} & y_{j-1} & y_{j-1} & y_j & y_j & y_{j+1} \\ x_{i-3} & x_{i-2} & x_{i-1} & x_i & x_{i+1} & x_{i+1} \end{array} .$$

Signal strength sequences by default, offer distance measures that have to be converted to similarities before Equation 3.3.3 can be applied. In bioinformatics, substitution matrices such as BLOSUM [32] or PAM [33] define the similarity scores for protein sequences, such that $s(a,b)$ would be positive if a and b are two amino acids similar to each other in terms of their physiochemical properties, or negative if a and b are different. The magnitude of $s(a,b)$ will be proportional to the degree of similarity/dissimilarity between a and b.

Filling a value $V(i,j)$ in the dynamic programming matrix corresponds to extending some alignment by adding the pair of characters (x_i, y_j) to it. This may increase or decrease the total score $V(i,j)$ of the growing alignment depending on the sign of $s(x_i, y_j)$. Should the score drop below 0, the alignment is not extended any further and a new alignment may be started. This corresponds to option iv in Equation 3.3.3. This works fine for biological sequences because the substitution matrices used to compute $s(a,b)$ are defined such that a score of 0 means the alignment

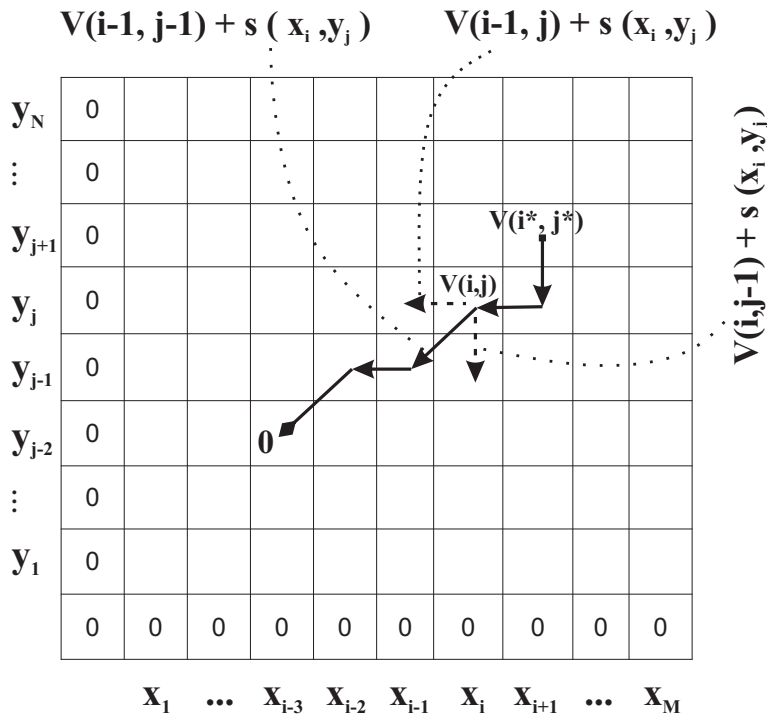


Figure 3.4. Dynamic programming matrix for local alignment. The matrix is filled using Equations 3.3.2 and 3.3.3. The path is determined using Equation 3.3.4.

is no better than a random alignment and therefore, makes good sense not to extend it any further. However, such similarity functions are not readily available for general time series.

To perform local alignment between signal strength sequences, we would ***ideally*** want to have a similarity function $s(x_i, y_j)$ with the following properties.

1. $s(x_i, y_j)$ should be a decreasing function of the actual geographical distance between x_i and y_j. The geographical distance between two sample points x_i and y_j, denoted by $d_{geo}(x_i, y_j)$, is computed as the distance between GPS coordinates $gps(x_i)$ and $gps(y_j)$.

2. For some geographical distance threshold d_{geo}^{th} (say 100m),

$$s(x_i, y_j) \begin{cases} > 0 & \text{, if } d_{geo}(x_i, y_j) < d_{geo}^{th} \\ \leq 0 & \text{, otherwise} \end{cases}$$

This will ensure that aligning geographically close points will cause the alignment score to increase and promote extension of the alignment, and that erroneously aligning geographically

distant points will cause the alignment score to decrease and the extension of the alignment to cease.

Relying on the expected positive correlation between $d_{geo}(x_i, y_j)$ and the normalized Euclidean distance $d(x_i, y_j)$ in the signal space [15],

$$\begin{aligned} s(x_i, y_j) &= C - d(x_i, y_j), \\ & \quad \textit{where } C \textit{ is some constant} \end{aligned} \tag{3.3.5}$$

usually satisfies property 1 for most pairs x_i, y_j. However, this does not readily satisfy property 2, unless a suitable value is chosen for C.

If a set of multiple reference (training) sequences, $Ref = \{X_1, X_2, ..., X_P\}$ is available for a given road segment of interest, we can estimate C using Algorithm 1 as follows. For all distinct ordered pairs X_u, X_v of reference traces, we take a random contiguous subsequence Y of X_v and align it to X_u (lines 3-6) using subsequence DTW as described in section 3.2. Then, the optimal alignment found in the previous step (line 6) is shifted to the left or right by a geographical distance d_{geo}^{th} and the DTW distance is recomputed (line 7). We treat the normalized DTW distance of an optimal alignment as an example for the signal distance between two sample points that are supposed to be aligned (positive example), and save it in the set of labeled distance tuples (LD) with class-label $+1$ (line 8). On the other hand, we treat the normalized DTW distance of a shifted alignment as an example for the signal distance between two sample points that are *not* supposed to be aligned (negative example), and save it in the set of labeled distance tuples with class-label -1 (line 9). A shifted alignment approximately simulates the situation when a vehicle moves into a different road and starts to deviate from the road segment of interest. This process repeats for a specified number of iterations (i_lim) for each distinct ordered pair $X_u, X_v \in Ref$, and all normalized DTW distances are saved in the set LD along with their respective class-labels. The reason for taking random subsequences of X_v is because otherwise, the alignments will be between pairs of full length reference sequences which does not represent the true situation when the system is in operation where relatively shorter length test sequences are aligned to a reference trace.

With the above setup, we formulate the problem of estimating C as the problem of estimating the best hyperplane (in this case a value in the 1-D space) for separating the positive and

negative classes of signal distances in LD. We do this by minimizing the loss function under the logistic model (line 12).

Algorithm 1 Estimating C with Multiple reference traces

1: $Ref = \{X_1, X_2, ..., X_P\}$ ▷ Set of reference traces
2: $LD = \{\ \}$ ▷ Set of labeled DTW distances
3: **for** $X_u, X_v \in Ref, u \neq v$ **do**
4: **for** $i = 1$ to i_lim **do**
5: $Y \leftarrow$ random subsequence of X_v
6: $norm\_dtw\_dist = \frac{D(X_u,Y,\phi^*)}{len(\phi^*)}$ ▷ See note below
7: $norm\_shifted\_dtw\_dist = \frac{D(X_u,Y,\phi_s)}{len(\phi_s)}$
8: $LD \leftarrow LD \cup (+1, norm\_dtw\_dist)$
9: $LD \leftarrow LD \cup (-1, norm\_shifted\_dtw\_dist)$
10: **end for**
11: **end for**
12:

$$C \leftarrow \arg\min_b \left\{ \sum_{(l,d)\in LD} log(1 + exp(-l(b-d))) \right\}$$

Note: $len(\phi^*)$ and $len(\phi_s)$ are the lengths of the optimal and shifted alignments respectively.

In our experiments, we estimated C using just two reference traces. However, in case collecting even two reference sequences is inconvenient, we propose the following procedure to estimate C from multiple test sequences. Let $Test\_set = Y_1, Y_2, ..., Y_Q$ be a collection of signal traces obtained from mobile phones connected to cell towers in a geographical area of interest. Note that we do *not* have GPS coordinates for these sequences which makes it difficult to know, with very high reliability, whether they belong to the road segment of interest or not. Nevertheless, they are predicted to be on the road segment of interest based on their handover sequence [34]. We then estimate C using the same procedure as in Algorithm 1 by replacing the set of reference sequences with the set of test sequences, $Test\_set$. However, for a given pair (Y_u, Y_v) of test traces, Y_v is used as it is without taking random subsequences because these sequences are representative of real test sequences.

3.4. Experimental Evaluation and Discussion

We evaluated the performance of our proposed algorithms and comparison algorithms using signal strength data we collected by driving on selected road segments as well as real call traces collected for us by a mobile network service provider.

The signal strength traces we collected were recorded on five different road segments (Figure 3.5) using two smart phones running Android OS. For each selected road segment, we drove multiple trips along the road under natural traffic conditions and recorded signal strength readings at a sampling interval of one second. GPS coordinates of the phone were also recorded at the same sampling rate (for use with reference traces and for evaluation purposes).

Let $Z = \{z_1, ..., z_N\}$ be the collection of signal strength sequences for a given road segment. In our evaluation, we took each ordered pair $(z_r, z_q) \in Z \times Z$, where $r \neq q$, and treat z_r and z_q as reference and test sequences respectively. We then break z_q into a number of random subsequences $z_q^1, \ldots, z_q^m, \ldots, z_q^M$ $(M \approx 30)$, and align each subsequence z_q^m to the reference t_r.

The reason for breaking the test sequence z_q into random subsequences is that the sequences that we have collected are longer than traces of calls made by typical phone users. Therefore, aligning the test sequence as it is does not entirely represent the true situation when actual call traces are aligned as they tend to be shorter. In fact, analysis of the call trace dataset provided to us by the mobile network provider indicated that call duration is approximately exponentially distributed with a mean of 110 seconds. Therefore, we generated random subsequences according to the said distribution. Furthermore, it allows us to do a better evaluation with the available data as this is similar to bootstrapping.

Alignments were performed using our proposed algorithms and comparison algorithms abbreviated and listed below.

- N_LDTW: Local Dynamic Time Warping as described in section 3.3.
- N_DTW: Subsequence Dynamic Time Warping as described in section 3.2.
- S_DDTW: Derivative Dynamic Time Warping using only the serving cells signal strengths.
- NN_Loc: Nearest neighbor-based localization where each point on the test trace is assigned to the nearest point on the reference trace based on the received signal strength fingerprint [15, 35].

For each sample point x_i of some test subsequence z_q^m, let $aligned(x_i)$ be the sample point of z_r to which x_i is aligned. Then the position estimation error of x_i, denoted by $pos\_error(x_i)$, is

defined as,

$$pos\_error(x_i) = d_{geo}(x_i, aligned(x_i)).$$

i.e., the geographical distance between the GPS coordinates of x_i and $aligned(x_i)$.

Figure 3.6 shows the cumulative distribution of the position estimation error for all sample points, on all random subsequences of all test traces t_q, in all ordered pairs $(z_r, z_q) \in Z \times Z$, for all three road segments. Results indicate that both N_LDTW and N_DTW are able to estimate the position with a lower error compared to S_DDTW and NN_Loc. The reason why N_LDTW and N_DTW perform equally well is because these sequences overlap with each other over their entire lengths, hence global or local alignment does *not* make a difference (Results for partially overlapping traces are presented later in this section). The figure also shows that NN_Loc performs better than S_DDTW which is unexpected because the results presented in [19] suggest otherwise. However, S_DDTW [19] was originally proposed for speed tracking, not position tracking. Therefore, we investigated the speed estimation error for the same experiments and present the results in Figure 3.7.

We see that N_LDTW and N_DTW perform well in both position and speed estimation. S_DDTW performs just as well as N_LDTW and N_DTW and better than NN_Loc in terms of speed estimation confirming the results published in [19], but performs worse than the other three algorithms in terms of position estimation. This indicates that S_DDTW produces comparatively large position errors but the error is fairly consistent (the estimated trace is mostly just translated along the road) keeping the speed calculation minimally affected by the position error. On the other hand, NN_Loc can erroneously estimate the position of some sample points such that their actual order is altered (Note that this doesn't happen with any of the alignment-based methods). This exaggerates the effect of position error on the speed calculation.

To evaluate the performance of the different algorithms when the test trace is not supposed to align entirely with the reference trace, we performed experiments similar to those explained above but allowed test traces involving other roads near the intersections with the road of interest. However, if a randomly selected test trace happened to be entirely within the road segment of interest, we select a random point y_i on the test trace, locate the corresponding point x_j on the reference trace, and remove the portion of the reference trace to the left (right) of x_j if y_i is

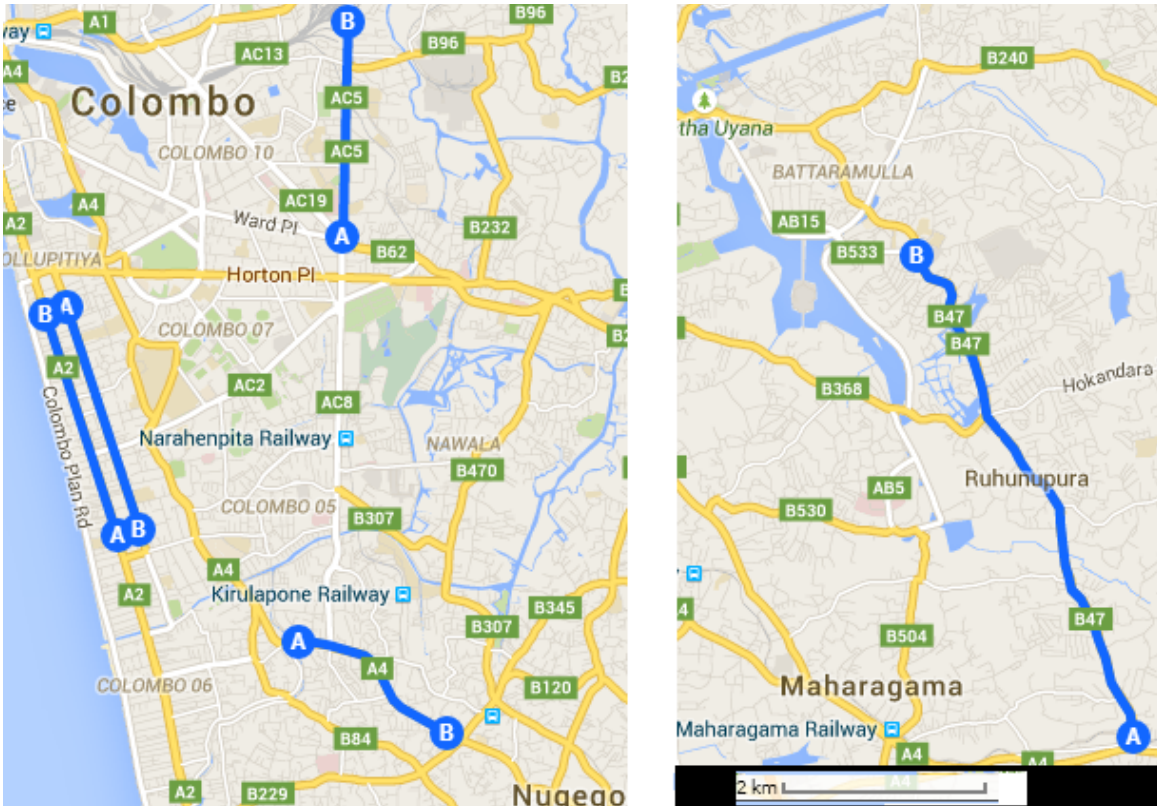


Figure 3.5. Road segments in Sri Lanka, used for the experiments conducted in this sutdy.

closer to the left (right) end of the test trace. Note that this trimming of the reference trace is only for that particular alignment and the full reference trace will be considered accordingly for subsequent alignments. Removal of a portion of the reference trace means a part of the test trace either at the left or the right end would ideally have no reference to align to. Figure 3.8 shows the cumulative distribution of position estimation error for these experiments. The figure shows that our local dynamic time warping algorithm N_LDTW outperforms all comparison algorithms. This is expected because N_LDTW was developed specifically to handle such partially overlapping traces. Not surprisingly, the global alignment methods perform poorly because they try to align the entire length of the test trace while it is not possible to do so. It is also evident that NN_Loc performs better than the global alignment methods, because it estimates the position (localization) of each sample point independently of others so that unlike in the global alignment methods, the localization of sample points that are supposed to align to the reference trace are not affected by the localization of sample points that are not supposed to align.

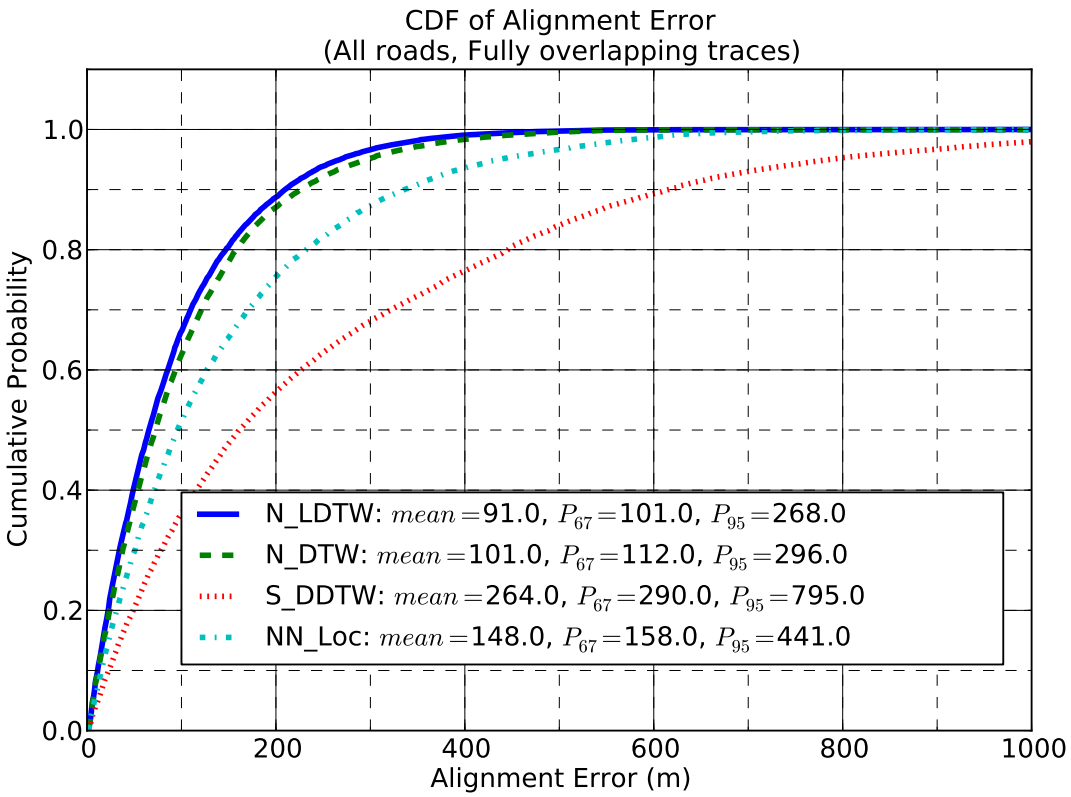


Figure 3.6. Cumulative distribution of position estimation error under different algorithms for fully overlapping traces.

To evaluate the performance of the algorithms on how they can be used to estimate average travel-time on a selected road segment, we collaborated with a mobile network service provider to collect a dataset of call traces made by their mobile subscribers (subscriber identification information was removed). The signal strength traces of calls made by users connected to a selected set of towers in the vicinity of the road segment were saved by the service provider. This included over 1300 calls which we selected as relevant for our road segment based on handover sequence [34]. During the same period of time, we recorded vehicle registration numbers and time stamps at the two ends of the road segment, and computed vehicle travel-times to be used as ground truth.

We had previously collected a reference trace for the road segment and aligned the call traces to it. Because only a few calls had lasted the full length of the road segment from start to finish, we divided the road segment into approximately 100m subsegments and computed average travel-time for each of them. Travel-time for a given vehicle (as estimated from a call trace) for a given subsegment is the difference in time when it first enters the subsegment and then leaves it. We

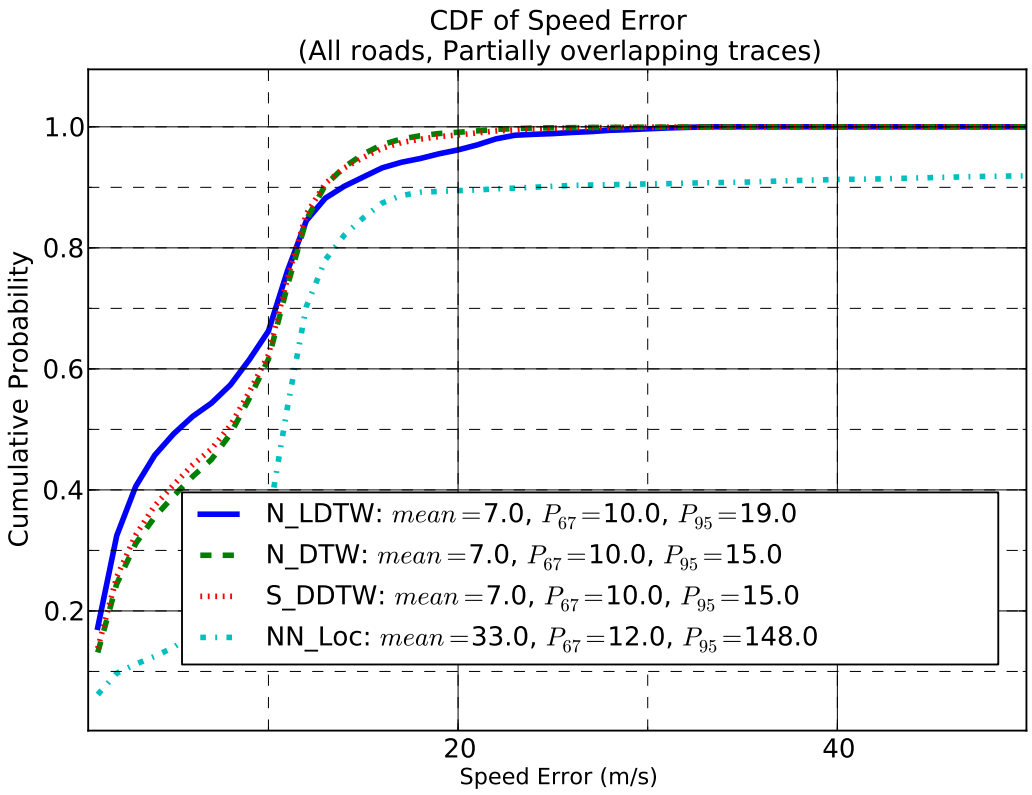


Figure 3.7. Cumulative distribution of speed estimation error under different algorithms for fully overlapping traces.

computed estimates for 10 minute periods. The average travel-time for a given 100m subsegment, for a given 10 minute interval, is the travel-time averaged over all vehicles that traveled through that subsegment within the given time interval. The average travel-time for the entire road segment for a given 10 minute interval is the sum of average travel-times for all subsegments within the given time interval.

Figure 3.9 shows how the estimated average travel-times compare with the actual average travel-times computed through manual number plate recognition. To summarize the error, we define average percentage error (avg_ pct_error) for travel-time estimation as shown below,

$$avg\_pct\_error = \frac{\sum_{all\ time\ intervals\ i} \frac{|est\_avg\_tt_i - act\_avg\_tt_i|}{act\_avg\_tt_i}}{No.\ of\ time\ intervals} \tag{3.4.1}$$

where $est\_avg\_tt_i$ and $act\_avg\_tt_i$ are the estimated and actual average travel-times for the road during the i^{th} time interval.

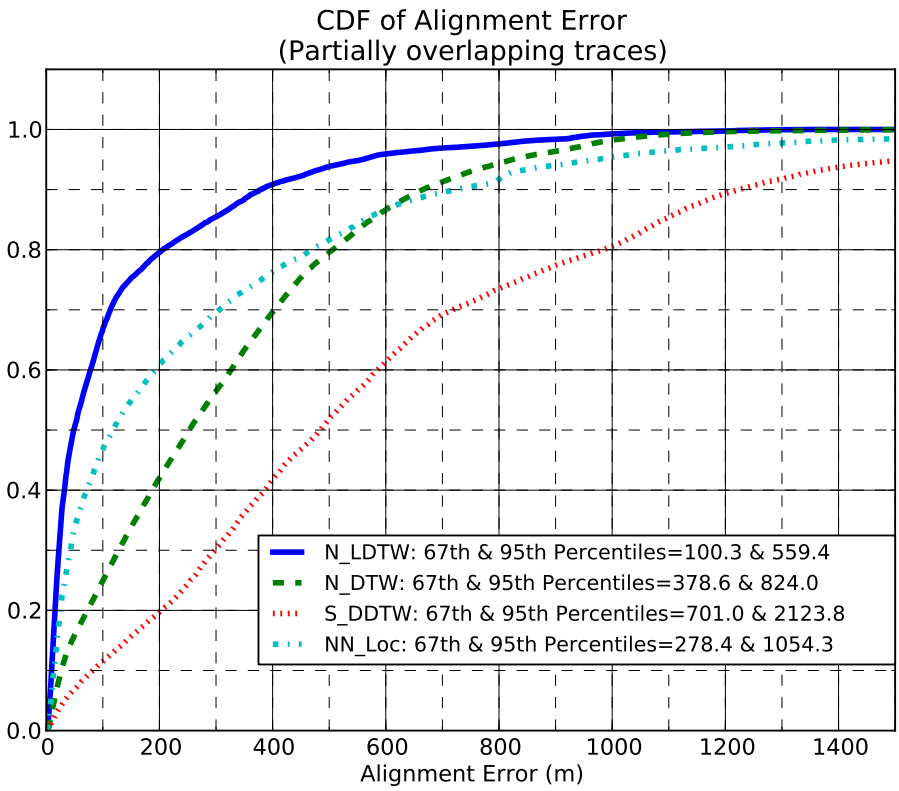


Figure 3.8. Cumulative distribution of position estimation error under different algorithms for partially overlapping traces.

We also computed the Pearson correlation coefficient w.r.t actual travel-times and its two-tailed statistical significance. As shown in Figure 3.9, N_LDTW has the lowest average percentage error of 14%, highest correlation coefficient of 0.76 and shows a significant correlation (at the 5% level) with actual travel-times with a p-value of 0.02. N_DTW and S_DDTW achieve average percentage errors of 19% and 23% respectively, but do not show significant correlation with actual travel-times.

On the other hand, NN_Loc (not shown in Figure 3.9) has performed poorly with an average percentage error of about 400%. We are unable to pinpoint the exact cause of this as we do not have GPS readings for these subscriber call traces. However, a possible reason, as mentioned before, is that NN_Loc can alter the proper geographical ordering of sample points exagerating the effect of localiation error on the travel-time computation. In fact, our investigation revealed that in 35% of cases, a vehicles estimated entry time to a 100m road subsegment was later than its exit time

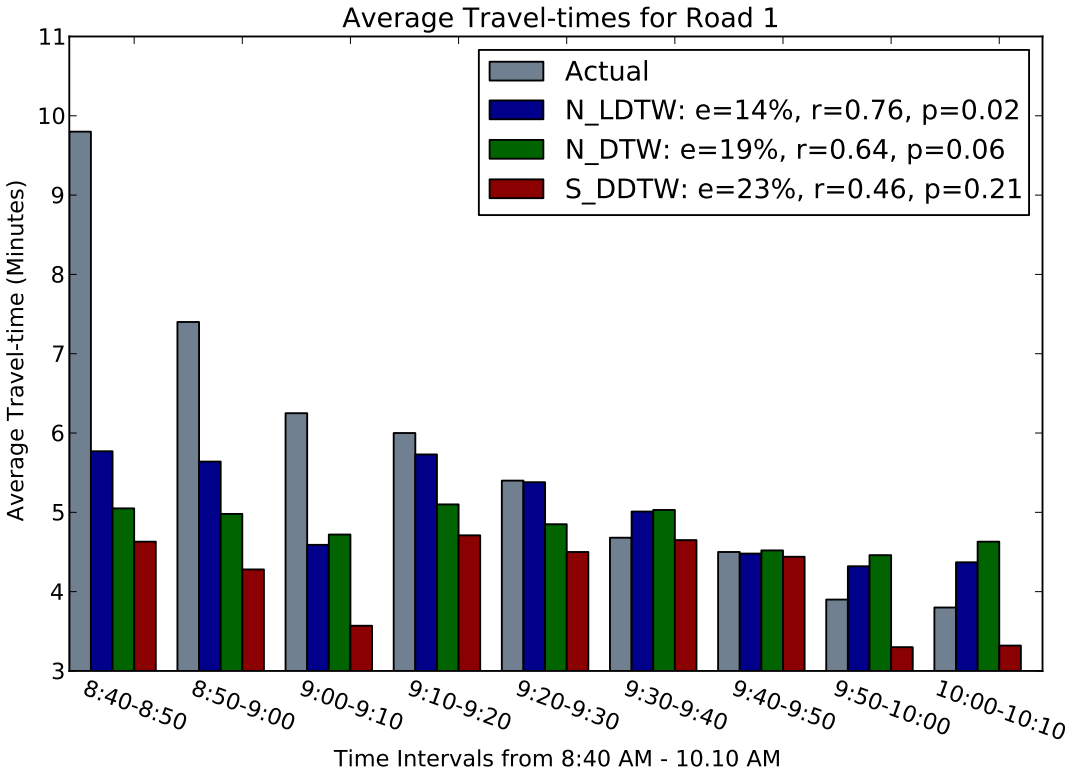


Figure 3.9. Travel-time estimation performance of the algorithms. The values e, r and p next to legend items are, average percentage error (Equation 3.4.1) and Pearson correlation coefficient w.r.t actual travel-times, and the two-tailed p-value of the correlation respectively, for each algorithm.

which indicates an incorrect ordering of sample points. Although we omited these negative time differences, they cause time overestimations for the adjacent subsegments.

3.5. Conclusions and Future Work

In this work, we developed a local dynamic time warping algorithm for tracking vehicles from mobile phone signal strength sequences, including those vehicles that make road changes. Our algorithm makes good use of all the available measurement data as we include neighbor cell signal strengths in the computation of the alignments. Our experiments on collected signal strength sequences as well as real-world subscriber call traces demonstrated better performance of our proposed algorithms (N_LDTW and N_DTW) compared to existing algorithms (S_DDTW and NN_Loc). In particular, the local DTW algorithm N_LDTW performed better than its global/subsequence DTW counterparts.

Our experiments on subscriber call trace data show that mobile phone signal traces can be used to successfully estimate average travel-time for a given road segment. However, our evaluation

was limited by the rather short duration for which call trace data was made available to us. As future work, we plan to further collaborate with a mobile service provider to acquire a larger dataset covering several days.

4. COMPLETE OBSERVATION MODEL FOR VEHICLE TRACKING FROM MOBILE PHONE RECEIVED SIGNAL STRENGTH TRACES

4.1. Introduction

Various disturbances in the radio propagation environment such as those caused by the movement of vehicles/pedestrians and changes in atmospheric conditions cause fluctuations in received signal strength levels. As mentioned in Chapter 1, a GSM fingerprint measured by a phone includes the strengths of the signals it receives from the associated cell tower and the six strongest neighbor cell towers. However, as signal strength levels fluctuate over time, the phone may associate with a different cell tower and/or the towers that act as the six strongest neighbors may change. Furthermore, even if a cell has a sufficiently high signal strength, if the phone cannot correctly decode the signal from that cell, for instance, due to low signal to noise ratio, it will not be included in the fingerprint [36, 37, 3]. Therefore, as shown in Figure 4.1 using real measurements, even for the same location, the set of cells included in a fingerprint may change when measured at different times.

A key component of the Bayesian estimation method for vehicle tracking from mobile phone RSS levels is the *observation model*, which models for each grid-location (Figure 1.1) in the area of interest, the probability distribution of fingerprints observed at that location. Let $R_t = \{(c_{t,1}, y_t^{c_{t,1}}), (c_{t,2}, y_t^{c_{t,2}}), \ldots, (c_{t,L_t}, y_t^{c_{t,L_t}})\}$ be a GSM fingerprint, where for each r with $1 \leq r \leq L_t \leq 7$, $y_t^{c_{t,r}}$ is the strength of the signal received from the cell tower with identification code $c_{t,r}$, at time t. To account for all the variability mentioned in the previous paragraph, the probability of observing a fingerprint R_t at a given location s_i, denoted by $P(R_t|s_i)$, should account for two components of variation, i.e., the variation of the set of cells included in the fingerprint and the fluctuations in signal strength levels. However, as evident in Section 4.2, existing methods do not model the former component.

The contribution of this work is that we propose an observation model that models the variation of the set of cells included in the fingerprint in addition to the fluctuations in their

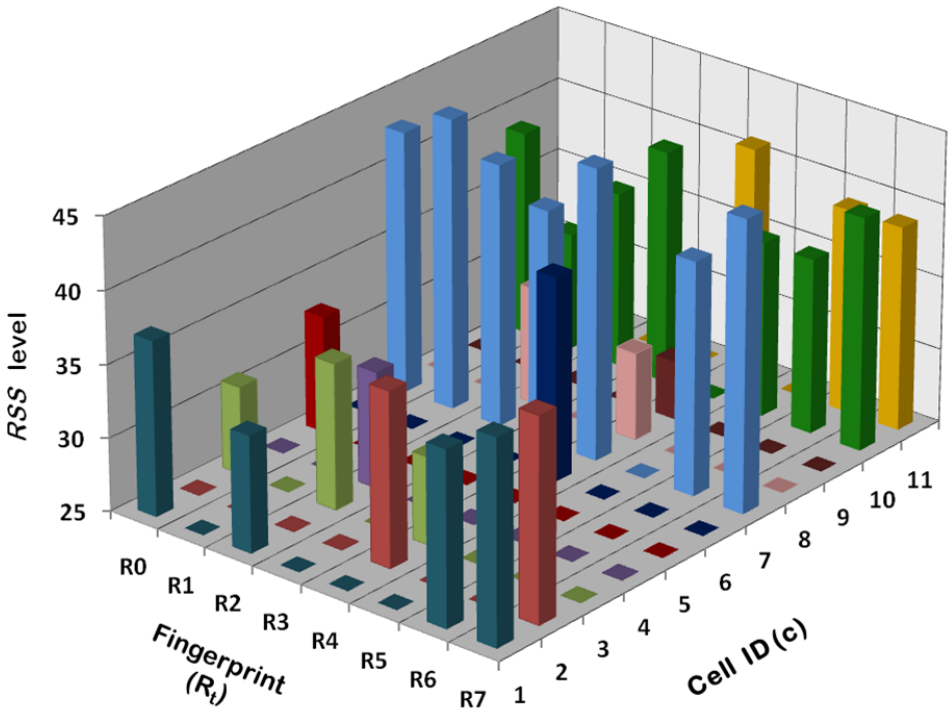


Figure 4.1. Variation of RSS levels and cell towers included in fingerprints collected at different times (several hours apart), at some *given* location. For any fingerprint R_t (horizontal axis) and cell c (depth axis) the height of the column represents the RSS level of cell c as reported by R_t. A missing bar indicates that the cell is not reported by that fingerprint. Please note that for convenience, cell IDs have been renumbered consecutively starting from 1.

signal strength levels. To achieve this, we define $P(R_t|s_i)$ as the probability of *the set of cells* $\{c_{t,1}, \ldots, c_{t,L_t}\}$ *being present in a fingerprint observed at location* s_i **and** *their received signal strength levels being* $y_t^{c_{t,1}}, \ldots, y_t^{c_{t,L_t}}$ *respectively.* We show how maximum likelihood learning can be used to estimate the parameters of the probability distribution of $P(R_t|s_i)$ for each predefined grid location s_i. Subsequently, we use the proposed observation model with a dynamic Bayesian network for tracking vehicles. Furthermore, we apply the proposed algorithm to estimate travel-times for a road segment.

The organization of this Chapter is as follows. In Section 4.2 we discuss the observation models used in existing work. In Section 4.3 we provide the details of the complete observation model that we propose. The procedure for learning the parameters of the proposed model is described in Section 4.4. We describe the experimental evaluation of our proposed method comparing

it to some of the existing ones, and discuss the results in Section 5.3. The concluding remarks of the Chapter are given in Section 4.7.

4.2. Existing Observation Models

Probabilistic methods model received signal strength levels as random variables. Let $f^c_{s_i}(y)$ be the probability mass function for the received signal strength levels from cell c at some location s_i. Then, assuming independence between cell towers, it is typical to define the probability $P(R_t|s_i)$ of observing the fingerprint R_t at location s_i as [7, 8, 16],

$$P(R_t|s_i) = \prod_{c \in C_t} f^c_{s_i}(y^c_t), \tag{4.2.1}$$

where $C_t = \{c_{t,1}, \ldots, c_{t,L_t}\}$ is the set of cells present in fingerprint R_t.

Equation 4.2.1 forms the observation model used in CellSense [7], and the probability mass functions $f^c_{s_i}$ are estimated using histograms constructed with signal strength measurements taken during a training phase that involves war-driving. For mathematical convenience and as a good fit to empirical data, many authors model signal strength levels as Gaussian random variables [8, 16, 18]. Therefore,

$$f^c_{s_i}(y) = \ N(\mu^c_{s_i}, \sigma^c_{s_i}) = \frac{1}{\sigma^c_{s_i}\sqrt{2\pi}} exp\left(-(\frac{y-\mu^c_{s_i}}{\sqrt{2}\sigma^c_{s_i}})^2\right), \tag{4.2.2}$$

and

$$P(R_t|s_i) = \prod_{c \in C_t} \frac{1}{\sigma^c_{s_i}\sqrt{2\pi}} exp\left(-(\frac{y^c_t-\mu^c_{s_i}}{\sqrt{2}\sigma^c_{s_i}})^2\right), \tag{4.2.3}$$

where $\mu^c_{s_i}$ is the mean signal strength at location s_i from cell c, and $\sigma^c_{s_i}$ is the standard deviation of the corresponding signal strength.

The observation model used in [16] is formed using Equation 4.2.3, and its authors use radio wave propagation tools to predict $\mu^c_{s_i}$. Furthermore, the standard deviation of signal strength levels $\sigma^c_{s_i}$ is assumed to be the same, say σ, at any location and for any cell, which is a reasonable assumption used in other works as well [18, 6].

In [8], Equation 4.2.3 is slightly modified as it considers only those cell towers present in the observed fingerprint for which $\mu^c_{s_i}$ has been successfully estimated during the training phase. Fur-

thermore, the logarithm of $P(R_t|s_i)$ is used as it is computationally convenient, and is normalized with respect to the number of cell towers considered in the computation.

$$\begin{aligned} ln\left(P(R_t|s_i)\right) &= \frac{1}{|C_t \cap C_{s_i}|} \sum_{c \in C_t \cap C_{s_i}} ln\left(\frac{1}{\sigma^c_{s_i}\sqrt{2\pi}} exp\left(-(\frac{y^c_t - \mu^c_{s_i}}{\sqrt{2}\sigma^c_{s_i}})^2\right)\right) \\ &= -\frac{1}{|C_t \cap C_{s_i}|} \sum_{c \in C_t \cap C_{s_i}} \left[(\frac{y^c_t - \mu^c_{s_i}}{\sqrt{2}\sigma^c_{s_i}})^2 + ln(\sigma^c_{s_i}\sqrt{2\pi})\right], \end{aligned} \tag{4.2.4}$$

where C_{s_i} is the set of cell towers for which mean signal strength levels have been successfully estimated at location s_i.

CTrack [6] is another HMM-based vehicle tracking algorithm. In [6], unlike the other methods discussed so far, the fingerprints collected during the training phase are not used to construct a histogram nor fitted to a Gaussian distribution. Instead, a score called the emission score that is proportional to the likelihood of an observed fingerprint R_t being generated from location s_i, denoted by $E(R_t, s_i)$, is computed by comparing R_t to each of the training fingerprints for location s_i as follows. As defined below, let $E_p(R_t, R_{i,j})$ be the pairwise emission score between an observed fingerprint R_t and the j^{th} training fingerprint for the location s_i.

$$E_p(R_t, R_{i,j}) = |C_t \cap C_{i,j}|\,\lambda_m + d_{max} - \frac{1}{|C_t \cap C_{i,j}|}\sqrt{\sum_{c \in C_t \cap C_{i,j}} (y^c_t - y^c_{i,j})^2}, \tag{4.2.5}$$

where C_t and $C_{i,j}$ are the set of cell towers present in R_t and $R_{i,j}$ respectively, λ_m is a weighting parameter empirically set to 3, and $d_{max} = 32$. Then, the emission score

$$E(R_t, s_i) = \arg\max_j \{E_p(R_t, R_{i,j})\}. \tag{4.2.6}$$

This emission score resembles Equation 4.2.4 because, if $\sigma^c_{s_i}$ are assumed to be the same at all locations for all the cells, then, $ln\left(P(R_t|s_i)\right)$ is proportional to the last term (Euclidean distance) in Equation 4.2.5. However, it must be noted that $E_p(R_t, R_{i,j})$ has an additional term that is proportional to the number of cell towers shared between the observed and the training fingerprints.

All of the observation models mentioned above, address the variation of received signal strength levels from the cells towers present in the fingerprint, but ignore the variation of the set of cell towers itself.

4.3. Complete Observation Model

In this Section, we propose an observation model for GSM received signal strength fingerprints that addresses the variation of the set of reported cell towers in addition to the variation of signal strength levels.

Let $C = \{1, 2, \ldots, c, \ldots, L\}$ be the set of cell IDs of the cell towers in the area of interest (actual IDs renumbered from 1 to L for convenience). Let R_t be the content of the fingerprint reported at time t, as shown below.

$$R_t = \{(c_{t,1}, y_t^{c_{t,1}}), \ldots, (c_{t,r}, y_t^{c_{t,r}}), \ldots, (c_{t,L_t}, y_t^{c_{t,L_t}})\} \tag{4.3.1}$$

where each $c_{t,r} \in C$ is a cell ID, and $y_t^{c_{t,r}}$ is the received signal strength from cell $c_{t,r}$ at time t. Then, $C_t = \{c_{t,1}, \ldots, c_{t,r}, \ldots, c_{t,L_t}\}$ is the set of Cell IDs present in R_t, and $Y_t^{C_t} = [y_t^{c_{t,1}}, \ldots, y_t^{c_{t,r}}, \ldots, y_t^{c_{t,L_t}}]$ is the vector of signal strength levels reported by R_t.

Let $S = \{s_1, s_2, \ldots, s_i, \ldots, s_N\}$ be a set of discrete locations (grid-locations) in the area of interest as in Figure 1.1. Let s_t be the location of the measuring device (phone) at time t, and let $P(R_t|s_t = s_i)$ be the probability of observing the content of a fingerprint R_t given the phone reporting it was at location s_i at time t. Then, $P(R_t|s_t = s_i)$ is the probability of the cell IDs in C_t being present in a fingerprint ***and*** their RSS values being $Y_t^{C_t}$, given that the fingerprint was generated from location s_i. We define this formally as follows.

For any $C_u \in C$, let $Y_i^{C_u} = [y_i^{c_{u,1}}, \ldots, y_i^{c_{u,h}}, \ldots, y_i^{c_{u,L_u}}]$ be a random vector, where $\forall c_{u,h} \in C_u, y_i^{c_{u,h}}$ is the random variable representing the instantaneous signal strength from cell $c_{u,h}$ as received at location s_i. Let $C_i \subset C$ be the random variable (random set) that represents the set of Cell IDs present in a fingerprint generated by a phone at location s_i. Then $Y_i^{C_i} = [y_i^{c_{i,1}}, \ldots, y_i^{c_{i,h}}, \ldots, y_i^{c_{i,L_i}}]$ is the random vector, where $\forall c_{i,h} \in C_i$, random variable $y_i^{c_{i,h}}$ represents the instantaneous signal strength from cell $c_{i,h}$ as received at location s_i. Then,

$$P(R_t|s_t = s_i) = P(C_t, Y_t^{C_t}|s_t = s_i) = P(C_i = C_t, Y_i^{C_i} = Y_t^{C_t}). \tag{4.3.2}$$

We re-write $P(R_t|s_t = s_i)$ as,

$$P(R_t|s_t = s_i) = P(C_i = C_t|Y_i^{C_i} = Y_t^{C_t}).P(Y_i^{C_i} = Y_t^{C_t}). \tag{4.3.3}$$

For a GSM network, a fingerprint includes the received signal strength levels of up to 7 strongest cell towers (associated cell + up to 6 strongest neighbors) with received signal strength levels above some minimum detectable threshold τ. However, due to low signal to noise ratio and/or other reasons, a phone may fail to decode the signal from a cell even if it is within range and among the strongest 7, in which case it will not be included in the fingerprint. Therefore, the cells reported will be the associated cell and the 6 strongest neighbors that the phone has been able to decode successfully. Let W_i^c, be the event that the phone has been able to successfully decode cell c at location s_i.

With this setup, we compute $P(C_i = C_t|Y_i^{C_i} = Y_t^{C_t})$ of Equation 4.3.3 as follows.

Case 1: $|C_t| = 7$

If signal strength readings from exactly 7 cell towers are present in the fingerprint, it means that each of the cells present in the fingerprint has a signal strength greater than the minimum detectable threshold τ and the phone has successfully decoded the signal from it, and each of the other cells either has a signal strength lower than the weakest reported cell and/or the phone has failed to decode the signal from it. Therefore, given $Y_i^{C_i} = Y_t^{C_t}$, the following statement is true.

$$C_i = C_t \Leftrightarrow [\forall c \in C_t, y_i^c > \tau \cap W_i^c] \cap [\forall d \in C - C_t, y_i^d < min(Y_t^{C_t}) \cup \neg W_i^d],$$

where $min(Y_t^{C_t}) = \min_{\forall c \in C_t}\{y_t^c\}$. Furthermore, since $Y_i^{C_i} = Y_t^{C_t}$, $y_i^c = y_t^c$ for each $c \in C_t$. Therefore,

$$\begin{aligned} &P(C_i = C_t|Y_i^{C_i} = Y_t^{C_t}) \\ &\quad = P([\forall c \in C_t, y_t^c > \tau \cap W_i^c] \cap [\forall d \in C - C_t, y_i^d < min(Y_t^{C_t}) \cup \neg W_i^d]). \end{aligned} \tag{4.3.4}$$

As in most existing work [8, 16, 18, 6, 7], we assume that the signal strengths from individual cells are independent of each other. Furthermore, we assume that the events W_i^c are independent of each other as well. Then,

$$\begin{aligned}
P(C_i =& C_t|Y_i^{C_i} = Y_t^{C_t}) \\
&= \prod_{\forall c \in C_t} P(y_t^c > \tau \cap W_i^c) \prod_{\forall d \in C - C_t} P(y_i^d < min(Y_t^{C_t}) \cup \neg W_i^d) \\
&= \prod_{\forall c \in C_t} P(W_i^c|y_t^c > \tau) P(y_t^c > \tau) \prod_{\forall d \in C - C_t} [1 - P(y_i^d > min(Y_t^{C_t}) \cap W_i^d)] \\
&= \prod_{\forall c \in C_t} P(W_i^c|y_t^c > \tau) P(y_t^c > \tau) \\
&\quad \prod_{\forall d \in C - C_t} [1 - P(W_i^d|y_i^d > min(Y_t^{C_t})) P(y_i^d > min(Y_t^{C_t}))]
\end{aligned} \tag{4.3.5}$$

Since $Y_t^{C_t}$ contains actual readings from a fingerprint, $y_t^c > \tau, \forall c \in C_t$ and $P(y_t^c > \tau) = 1, \forall c \in C_t$. Therefore,

$$\begin{aligned}
&P(C_i = C_t|Y_i^{C_i} = Y_t^{C_t}) \\
&\quad = \prod_{\forall c \in C_t} P(W_i^c|y_t^c > \tau) \prod_{\forall d \in C - C_t} [1 - P(W_i^d|y_i^d > min(Y_t^{C_t})) P(y_i^d > min(Y_t^{C_t}))]
\end{aligned} \tag{4.3.6}$$

We make the simplifying assumption that a phone's ability to decode the signal from a cell is independent of RSS given that RSS$> \tau$. Therefore, we define $w_i^d = P(W_i^d|y_i^d > \tau) = P(W_i^d|y_i^d > min(y_t^c))$. Then,

$$P(C_i = C_t|Y_i^{C_i} = Y_t^{C_t}) = \prod_{\forall c \in C_t} w_i^c \prod_{\forall d \in C - C_t} [1 - w_i^d P(y_i^d > min(Y_t^{C_t}))]. \tag{4.3.7}$$

As mentioned before in Section 4.2, we model the received signal strength level from cell $c \in C$, at location s_i, denoted by y_i^c, as a Gaussian random variable with mean μ_i^c and standard deviation σ. Therefore,

$$\begin{aligned}
P(y_i^c < y) &= \frac{1}{\sigma\sqrt{2\pi}} \int_{-\infty}^{y} exp(-(\frac{y - \mu_i^c}{\sqrt{2}\sigma})^2) dy \\
P(y_i^c = y) &= \frac{1}{\sigma\sqrt{2\pi}} exp(-(\frac{y - \mu_i^c}{\sqrt{2}\sigma})^2)
\end{aligned} \tag{4.3.8}$$

Therefore,

$$
\begin{aligned}
&P(R_t|s_t = s_i) \\
&\quad = P(C_i = C_t|Y_i^{C_i} = Y_t^{C_t}).P(Y_i^{C_i} = Y_t^{C_t}) \\
&\quad = \prod_{\forall c \in C_t} w_i^c \frac{exp(-(\frac{y_t^c - \mu_i^c}{\sqrt{2}\sigma})^2)}{\sigma\sqrt{2\pi}} \prod_{\forall d \in C - C_t} \left[1 - w_i^d \int_{min(Y_t^{C_t})}^{\infty} \frac{exp(-(\frac{y - \mu_i^d}{\sqrt{2}\sigma})^2)}{\sigma\sqrt{2\pi}} dy \right].
\end{aligned}
\tag{4.3.9}
$$

Case 2: $|C_t| < 7$

If signal strength readings from less that 7 cell towers are present in the fingerprint, it means that each of the cells present in the fingerprint has a signal strength greater than the minimum detectable threshold τ and the phone has successfully decoded the signal from it, and each of the other cells either has a signal strength lower than τ and/or the phone has failed to decode the signal from it. Therefore, given $Y_i^{C_i} = Y_t^{C_t}$, the following statement is true.

$$
C_i = C_t \Leftrightarrow [\forall c \in C_t, y_i^c > \tau \cap W_i^c] \cap [\forall d \in C - C_t, y_i^d < \tau \cup \neg W_i^d].
$$

Therefore,

$$
\begin{aligned}
&P(R_t|s_t = s_i) \\
&\quad = P(C_i = C_t|Y_i^{C_i} = Y_t^{C_t}).P(Y_i^{C_i} = Y_t^{C_t}) \\
&\quad = \prod_{\forall c \in C_t} w_i^c \frac{exp(-(\frac{y_t^c - \mu_i^c}{\sqrt{2}\sigma})^2)}{\sigma\sqrt{2\pi}} \prod_{\forall d \in C - C_t} \left[1 - w_i^d \int_{\tau}^{\infty} \frac{exp(-(\frac{y - \mu_i^d}{\sqrt{2}\sigma})^2)}{\sigma\sqrt{2\pi}} dy \right].
\end{aligned}
\tag{4.3.10}
$$

Combining Equations 4.3.9 and 4.3.10 corresponding to Cases 1 and 2 respectively, we obtain,

$$
\begin{aligned}
&P(R_t|s_t = s_i) \\
&\quad = P(C_i = C_t|Y_i^{C_i} = Y_t^{C_t}).P(Y_i^{C_i} = Y_t^{C_t}) \\
&\quad = \prod_{\forall c \in C_t} w_i^c \frac{exp(-(\frac{y_t^c - \mu_i^c}{\sqrt{2}\sigma})^2)}{\sigma\sqrt{2\pi}} \prod_{\forall d \in C - C_t} \left[1 - w_i^d \int_{y_t^{min}}^{\infty} \frac{exp(-(\frac{y - \mu_i^d}{\sqrt{2}\sigma})^2)}{\sigma\sqrt{2\pi}} dy \right],
\end{aligned}
\tag{4.3.11}
$$

where,

$$y_t^{min} = \begin{cases} \min_{c \in C}\{y_t^c\} & , if\ |C_t| = 7 \\ \tau & , otherwise \end{cases}.$$

4.4. Maximum Likelihood Learning of Model Parameters

The mean signal strength levels μ_i^c and probabilities of successful decoding w_i^c, for each cell $c \in C$, at each location s_i in the area of interest, are parameters of the observation model that need to be learned. We use maximum likelihood learning to estimate these parameters from training fingerprints gathered through war-driving.

Let $\mathbf{\Theta}$ be a set of parameters to be estimated from observed training data $\mathbf{Z}$. Then, maximum likelihood learning attempts to find the set of parameters $\mathbf{\Theta}^*$ that maximizes the posterior probability $P(\mathbf{\Theta}|\mathbf{Z})$. I.e,

$$\begin{aligned} \mathbf{\Theta}^* &= \arg\max_{\Theta} \{P(\Theta|\mathbf{Z})\} \\ &= \arg\max_{\Theta} \{logP(\Theta|\mathbf{Z})\} \\ &= \arg\max_{\Theta} \left\{ log\left(\frac{P(|\mathbf{\Theta}).\mathbf{P}(\mathbf{\Theta})}{P(\mathbf{Z})} \right) \right\} \\ &= \arg\max_{\Theta} \{log\left(P(\mathbf{Z}|\mathbf{\Theta}).\mathbf{P}(\mathbf{\Theta})\right)\} \\ &= \arg\max_{\Theta} \{logP(\mathbf{Z}|\mathbf{\Theta}) + \mathbf{logP}(\mathbf{\Theta})\}. \end{aligned} \tag{4.4.1}$$

In the absence of additional information about the prior probability of the parameters $P(\Theta)$, we assume it is uniformly distributed and ignore it in the optimization procedure. Therefore,

$$\mathbf{\Theta}^* = \arg\max_{\mathbf{\Theta}} \{\mathbf{logP}(\mathbf{Z}|\mathbf{\Theta})\}. \tag{4.4.2}$$

In our case, the set of parameters $\mathbf{\Theta}$ over which we attempt to optimize $P(\mathbf{\Theta}|\mathbf{Z})$, are the means μ_i^c and the probabilities successful decoding w_i^c. Therefore,

$$\mathbf{\Theta} = \{\mu_{\mathbf{i}}^{\mathbf{c}}, \mathbf{w_i^c} : \mathbf{1} \leq \mathbf{i} \leq \mathbf{N}, \forall \mathbf{c} \in \mathbf{C}\}.$$

Suppose for each location $s_i \in S$, a collection of fingerprints $R_i = \{R_{i,1}, \ldots, R_{i,t}, \ldots, R_{i,T_i}\}$, where $R_{i,t}$ is the t^{th} fingerprint at s_i, has been recorded by driving along the road. Then $\mathbf{Z} = \{\mathbf{R_1}, \ldots, \mathbf{R_i}, \ldots, \mathbf{R_N}\}$ and,

$$logP(\mathbf{Z}|\mathbf{\Theta}) = \mathbf{logP}(\mathbf{R_1}, \ldots, \mathbf{R_N}|\mu_\mathbf{i}^\mathbf{c}, \mathbf{w_i^c}; \mathbf{1} \leq \mathbf{i} \leq \mathbf{N}, \forall \mathbf{c} \in \mathbf{C}). \tag{4.4.3}$$

All fingerprints $R_{i,t}$ are independent of each other, and given μ_i^c, w_i^c, each $R_{i,t}$ is conditionally independent of μ_j^c, w_j^c for any $j \neq i$. Therefore,

$$\begin{aligned} logP(\mathbf{Z}|\mathbf{\Theta}) &= \sum_{s_i \in S} \sum_{t=1}^{T_i} logP(R_{i,t}|\mu_i^c, w_i^c; \forall c \in C) \\ &= \sum_{s_i \in S} \sum_{t=1}^{T_i} logP(R_{i,t}|s_t = s_i). \end{aligned} \tag{4.4.4}$$

Using Equation 4.3.11,

$$\begin{aligned} logP(\mathbf{Z}|\mathbf{\Theta}) &= \sum_{s_i \in S} \sum_{t=1}^{T_i} logP(R_{i,t}|s_t = s_i) \\ &= \sum_{s_i \in S} \sum_{t=1}^{T_i} \sum_{c \in C_{i,t}} \left(\frac{-(y_{i,t}^c - \mu_i^c)^2}{2\sigma^2} \right) + \sum_{s_i \in S} \sum_{t=1}^{T_i} \sum_{c \in C_{i,t}} ln(w_i^c) \\ &\quad + \sum_{s_i \in S} \sum_{t=1}^{T_i} \sum_{d \in C - C_{i,t}} ln \left(1 - w_i^d \int_{y_{i,t}^{min}}^{\infty} \frac{exp(-(\frac{y-\mu_i^d}{\sqrt{2}\sigma})^2)}{\sigma\sqrt{2\pi}} dy \right), \end{aligned} \tag{4.4.5}$$

where $C_{i,t}$ is the set of cell IDs present in $R_{i,t}$, $y_{i,t}^c$ is the received signal strength level from cell c at location s_i as reported in $R_{i,t}$, and

$$y_{i,t}^{min} = \begin{cases} \min_{c \in C}\{y_{i,t}^c\} & , if\ |C_{i,t}| = 7 \\ \tau & , otherwise \end{cases}.$$

It is not possible to maximize $logP(\mathbf{Z}|\mathbf{\Theta})$ in Equation 4.4.5, analytically. Therefore, we use the truncated Newton method to maximize it numerically (actually, minimize -$logP(\mathbf{Z}|\mathbf{\Theta})$). This numerical optimization requires the partial derivatives of the objective function $logP(\mathbf{Z}|\mathbf{\Theta})$, and initial guesses for μ_i^c and w_i^c.

The partial derivatives of $logP(\mathbf{Z}|\mathbf{\Theta})$ are as follows.

$$\begin{aligned}
&\forall c \in C, s_i \in S, \\
&\frac{\partial logP(\mathbf{Z}|\mathbf{\Theta})}{\partial \mu_i^c} = \\
&\sum_{t=1}^{T_i} \frac{(y_{i,t}^c - \mu_i^c)}{\sigma} g_{i,t}(c) - \sum_{t=1}^{T_i} \frac{w_i^c \frac{\partial}{\partial \mu_i^c} \left(\int_{y_{i,t}^{min}}^{\infty} \frac{exp(-(\frac{y-\mu_i^c}{\sqrt{2}\sigma})^2)}{\sigma\sqrt{2\pi}} dy \right)}{1 - w_i^c \int_{y_{i,t}^{min}}^{\infty} \frac{exp(-(\frac{y-\mu_i^c}{\sqrt{2}\sigma})^2)}{\sigma\sqrt{2\pi}} dy} [1 - g_{i,t}(c)],
\end{aligned} \tag{4.4.6}$$

where,

$$g_{i,t}(c) = \begin{cases} 1 & , if\ c \in C_{i,t} \\ 0 & , otherwise \end{cases}.$$

Using Leibniz's rule for differentiation under the integral sign,

$$\begin{aligned}
\frac{\partial}{\partial \mu_i^c} \left(\int_{y_{i,t}^{min}}^{\infty} \frac{exp(-(\frac{y-\mu_i^c}{\sqrt{2}\sigma})^2)}{\sigma\sqrt{2\pi}} dy \right) &= \int_{y_{i,t}^{min}}^{\infty} \frac{\partial}{\partial \mu_i^c} \left(\frac{exp(-(\frac{y-\mu_i^c}{\sqrt{2}\sigma})^2)}{\sigma\sqrt{2\pi}} \right) dy \\
&= \int_{y_{i,t}^{min}}^{\infty} \frac{2(y-\mu_i^c)}{2\sigma^2} \left(\frac{exp(-(\frac{y-\mu_i^c}{\sigma})^2)}{\sigma\sqrt{2\pi}} \right) dy \\
&= \frac{1}{\sigma\sqrt{2\pi}} \left[-exp\left(-\left(\frac{y-\mu_i^c}{\sqrt{2}\sigma} \right)^2 \right) \right]_{y_{i,t}^{min}}^{\infty} \\
&= \frac{1}{\sigma\sqrt{2\pi}} exp\left(-\left(\frac{y_{i,t}^{min} - \mu_i^c}{\sqrt{2}\sigma} \right)^2 \right).
\end{aligned} \tag{4.4.7}$$

Substituting this result in Equation 4.4.8,

$$\begin{aligned}
&\forall c \in C, s_i \in S, \\
&\frac{\partial logP(\mathbf{Z}|\mathbf{\Theta})}{\partial \mu_i^c} = \sum_{t=1}^{T_i} \frac{(y_{i,t}^c - \mu_i^c)}{\sigma} g_{i,t}(c) - \sum_{t=1}^{T_i} \frac{w_i^c \frac{1}{\sigma\sqrt{2\pi}} exp\left(-\left(\frac{y_{i,t}^{min} - \mu_i^c}{\sqrt{2}\sigma} \right)^2 \right)}{1 - w_i^c \int_{y_{i,t}^{min}}^{\infty} \frac{exp(-(\frac{y-\mu_i^c}{\sqrt{2}\sigma})^2)}{\sigma\sqrt{2\pi}} dy} [1 - g_{i,t}(c)].
\end{aligned} \tag{4.4.8}$$

The partial derivatives of $logP(\mathbf{Z}|\mathbf{\Theta})$ with respect to w_i^c are as follows.

$\forall c \in C, s_i \in S,$

$$\frac{\partial logP(\mathbf{Z}|\mathbf{\Theta})}{\partial w_i^c} = \frac{1}{w_i^c}\sum_{t=1}^{T_i} g_{i,t}(c) - \sum_{t=1}^{T_i} \frac{\int_{y_{i,t}^{min}}^{\infty} \frac{exp(-(\frac{y-\mu_i^c}{\sqrt{2}\sigma})^2)}{\sigma\sqrt{2\pi}} dy}{1 - w_i^c \int_{y_{i,t}^{min}}^{\infty} \frac{exp(-(\frac{y-\mu_i^c}{\sqrt{2}\sigma})^2)}{\sigma\sqrt{2\pi}} dy}[1 - g_{i,t}(c)]. \tag{4.4.9}$$

The Newton method used for optimization needs an initial estimate for the parameters to start from. We compute the initial estimate for the mean μ_i^c, denoted by $init\_\mu_i^c$, as simply the mean of all received signal strength levels from cell c at location s_i measured during the training phase. I.e.,

$$init\_\mu_i^c = average\{y_{i,t}^c : t = 1, \ldots, T_i\} = \frac{1}{T_i}\sum_{t=1}^{T_i} y_{i,t}^c. \tag{4.4.10}$$

Initial estimate for w_i^c, denoted by $init\_w_i^c$, is computed as the value of w_i^c that makes $\frac{\partial logP(\mathbf{Z}|\mathbf{\Theta})}{\partial w_i^c} = 0$, at $\mu_i^c = init\_\mu_i^c$. This too cannot be computed analytically. Therefore, we compute it numerically using the bisection method in the interval $w_i^c \in (0, 1)$.

4.5. Dynamic Bayesian Network and Viterbi Decoding for Tracking

We use a Dynamic Bayesian Network (DBN) to track vehicles from a series of GSM received signal strength fingerprints. The states of the DBN are the locations $S = \{s_1, s_2, \ldots, s_i, \ldots, s_N\}$ and a set of discrete velocities $V = \{v_1, v_2, \ldots, v_j, \ldots, v_M\}$. At any given time t, a vehicle may be at some location $s_t \in S$ and move with velocity $v_t \in V$. Figure 4.2 shows two slices of the DBN at time indices $t-1$ and t. To track a vehicle, we use Viterbi decoding to map a series of fingerprints (observations) $R_1, R_2, \ldots, R_t, \ldots, R_T$ to the most probable sequence of locations $s_1^*, s_2^*, \ldots, s_t^*, \ldots, s_T^*$,and velocities $v_1^*, v_2^*, \ldots, v_t^*, \ldots, v_T^*$ with each $s_t^* \in S$ and $v_t^* \in V$.

The state transition probabilities of the DBN are as follows.

$$P(s_t = s_i | s_{t-1} = s_p, v_{t-1} = v_q) = \frac{1}{\sigma_d\sqrt{2\pi}} exp\left(-\left(\frac{d(s_i, s_p) - (s_p + v_q\Delta t)}{\sigma_d\sqrt{2}}\right)^2\right), \tag{4.5.1}$$

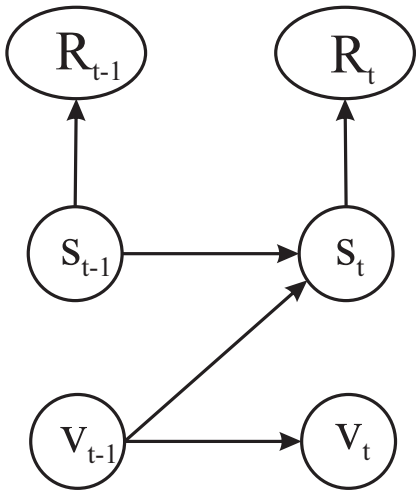


Figure 4.2. Two slices of the DBN used for tracking.

where $d(s_i, s_p)$ is the physical distance *along a road* between locations s_i and s_p, Δt is the time difference between the two slices t and $t-1$, and σ_d is the standard deviation of location. And,

$$P(v_t = v_j | v_{t-1} = v_q) = \frac{1}{\sigma_v \sqrt{2\pi}} exp\left(- \left(\frac{v_j - v_q}{\sigma_v \sqrt{2}} \right)^2 \right). \tag{4.5.2}$$

Initial distribution of grid-locations is uniform and is defined as below.

$$\pi(s_1 = s_i) = \frac{1}{N} \tag{4.5.3}$$

The initial distribution of velocities is assumed to be normally distributed around some mean velocity $\bar{v}$ with standard deviation σ_{v_1}, and is defined as below.

$$\pi(v_1 = v_j) = \frac{1}{\sigma_{v_1} \sqrt{2\pi}} exp\left(- \left(\frac{v_j - \bar{v}}{\sigma_{v_1} \sqrt{2}} \right)^2 \right). \tag{4.5.4}$$

The observation model of the DBN defines for each $s_i \in S$ and $v_j \in V$, the probability $P(R_t | s_t = s_i, v_t = v_j)$ of observing a fingerprint R_t given the phone generating it is at location s_i and moving at a velocity of v_j. We assume that a fingerprint is not significantly affected by the velocity of the vehicle. Therefore, $P(R_t | s_t = s_i, v_t = v_j) = P(R_t | s_t = s_i)$, and this as defined by Equation 4.3.11 with its parameters learned as explained in Section 4.4.

We perform Viterbi decoding, recursively, with the aid of two variables α and ψ as follows.

Initialization: $for\ t = 1$,

$$\alpha(i,j,t) = P(R_t|s_t = s_i)\pi(s_1 = s_i)\pi(v_1 = v_j). \tag{4.5.5}$$

Recursion: $for\ 2 \leq t \leq T$,

$$\begin{aligned}\alpha(i,j,t) = &\max_{p\in[1,N],q\in[1,M]}\{\alpha(p,q,t-1)P(s_t = s_i|s_{t-1} = s_p, v_{t-1} = v_q).\\ &P(v_t = v_j|v_{t-1} = v_q)P(R_t|s_t = s_i)\}\\ \psi(i,j,t) = (p^*,q^*) = &\arg\max_{p\in[1,N],q\in[1,M]}\{\alpha(p,q,t-1)P(s_t = s_i|s_{t-1} = s_p, v_{t-1} = v_q).\\ &P(v_t = v_j|v_{t-1} = v_q)P(R_t|s_t = s_i)\}\end{aligned} \tag{4.5.6}$$

The entries in ψ are used to trace back the most probable sequence of states $(s_1^*, v_1^*), (s_2^*, v_2^*), \ldots, (s_t^*, v_t^*), \ldots, (s_T^*, v_T^*)$, starting from (s_T^*, v_T^*) to (s_1^*, v_1^*) as follows.

$$\begin{aligned}&(s_T^*, v_T^*) = (s_a, v_b),\ where\ (a,b) = \arg\max_{i\in[1,N],j\in[1,M]}\{\alpha(i,j,T)\},\\ &and\ for\ T-1 \geq t \geq 1,\\ &(s_t^*, v_t^*) = \psi(p,q,t+1),\ where\ p,q \text{ are such that } (s_p, v_q) = (s_{t+1}^*, v_{t+1}^*)\end{aligned} \tag{4.5.7}$$

4.5.1. Improving the Speed of Viterbi Decoding for Tracking

Viterbi decoding described above involves computing $\alpha(i,j,t), \forall i \in [1,N], \forall j \in [1,M], t \in [1,T]$, where N is the number of grid-locations in the area of interest and M is the number of discrete velocities a vehicle may move at. This amounts to computing $N \times M \times T$ α values. Furthermore, according to Equation 4.5.6, computing each $\alpha(i,j,t)$ involves $N \times M$ computations. Therefore, its total time complexity is $O(N^2M^2T)$, which is computationally intensive. However, we can use the constraints associated with vehicular motion to reduce this time complexity as follows.

We assume that the velocity is always positive, meaning that the vehicle does not reverse its direction of motion (we construct a separate DBN for each direction for two-way roads). With

this setup, in the computation of each $\alpha(i, j, t)$, we only need to consider the grid-locations at or before s_i, along a given direction on the given road.

Furthermore, as the velocity transitions are modeled using a Gaussian distribution as shown in Equation 4.5.2, it is very unlikely that a velocity change of more than two standard deviations ($2\sigma_v$) from the current velocity will occur. Therefore, in the computation of each $\alpha(i, j, t)$, only those velocities within $2\sigma_v$ of v_j are considered. Studies conducted in [38, 39] suggest that maximum acceleration of a passenger vehicle is about $3.5ms^{-2}$. Therefore, we set $\sigma_v = 3.5ms^{-2}$ for one second time steps.

A similar argument can be applied to the movement (transitions) between grid-locations as well. Let v_{max} be the maximum velocity at which a typical passenger vehicle will move. Then according to Equation 4.5.1, for any grid-location s_i, a transition from any other grid-location s_p to s_i is very unlikely if $d(s_i, s_p) - v_{max}\Delta t > 2\sigma_d$, where Δt is the time step.

Let $N_i = \{p \in [1, i] : (d(s_i, s_p) - v_{max}\Delta t) \leq 2\sigma_d\}$ be the indices of grid-locations in the vicinity of s_i and $M_j = \{q \in [1, M] : |v_q - v_j| \leq 2\sigma_v\}$ be the indices of velocities in the vicinity of v_j. Then, we re-write Equation 4.5.6 as follows.

$$\begin{aligned}
\alpha(i, j, t) = \max_{p \in N_i, q \in M_j} \{&\alpha(p, q, t-1) P(s_t = s_i | s_{t-1} = s_p, v_{t-1} = v_q). \\
&P(v_t = v_j | v_{t-1} = v_q) P(R_t | s_t = s_i)\} \\
\psi(i, j, t) = (p^*, q^*) = \arg\max_{p \in N_i, q \in M_j} \{&\alpha(p, q, t-1) P(s_t = s_i | s_{t-1} = s_p, v_{t-1} = v_q). \\
&P(v_t = v_j | v_{t-1} = v_q) P(R_t | s_t = s_i)\}
\end{aligned} \tag{4.5.8}$$

Since $|N_i| << N$ and $|M_j| << M$, recursion with Equation 4.5.8 is faster than exploring the entire state space as in Equation 4.5.6.

4.6. Experimental Evaluation and Discussion of Results

We evaluated the performance of our proposed algorithm and comparison algorithms using signal strength data we collected by driving on selected road segments as well as real call traces collected for us by a mobile network service provider.

We collected signal strength fingerprints on five (three urban and two suburban) different roads to evaluate the performance of our DBN-based tracking algorithm with the proposed com-

plete observation model. We drove several trips along each road and collected a set of fingerprint sequences $Z = \{Z_1, \ldots, Z_a, \ldots, Z_A\}$, where each $Z_a = R_{a,1}, \ldots, R_{a,t}, \ldots, R_{a,T_a}$ is the sequence of fingerprints collected in the a^{th} trip, with each $R_{a,t}$ being a signal strength fingerprint. Each fingerprint $R_{a,t}$ has associated with it, a time stamp and GPS coordinates of the location $x_{a,t}$ at which it was generated. The GPS coordinates of $x_{a,t}$ are used to assign each training fingerprint to its nearest grid-location in S during the training phase, and used as ground truth for evaluation during the tracking phase.

For each road, we use one of the fingerprint sequences $Z_q = R_{q,1}, \ldots, R_{q,t}, \ldots, R_{q,T_q} \in Z$ as the test sequence from which the vehicle is tracked, and the remaining sequences as training data to estimate model parameters. The fingerprint sequences we have collected are longer than those generated during typical phone calls. Analysis of the mobile subscriber call trace dataset we received from the mobile service provider revealed that the duration of user calls are approximately exponentially distributed with mean 110 seconds. Therefore, to better resemble the real situation of a passive, network side tracking application, we break Z_q into random sub-sequences of the form $Z_q[e, f] = R_{q,e}, \ldots, R_{q,f}$, where e and f are random indices with $1 \leq e < f \leq T_q$ such that the sequence length $f - e$ is distributed according to the aforementioned distribution, and use these sub-sequences for tracking.

For each $Z_q[e, f]$, we use Viterbi decoding to find the most probable state sequence $(s_{q,e}, v_{q,e}), \ldots, (s_{q,t}, v_{q,t}), \ldots, (s_{q,f}, v_{q,f})$, where $s_{q,t}$ and $v_{q,t}$ are the location and velocity for $R_{q,t}$ respectively, as predicted by Viterbi decoding. Then, for each $R_{q,t}$, the localization error is defined as the distance between the true location $x_{q,t}$ and the predicted location $s_{q,t}$.

Figures 4.3 (a) and 4.3(b) show the cumulative distribution of the localization error for the three urban roads and the two suburban roads respectively. Vehicle tracking with our proposed DBN with the complete observation model (*DBN_Cmp_Obs_MRT*) achieves a mean localization error that is %37 and %42 less than tracking with a DBN with the same structure and transition model but with the traditional observation model defined in Equation 4.2.4 (*DBN_Trad_Obs_MRT*), for the urban and suburban roads respectively. *DBN_Cmp_Obs_MRT* also outperforms the dynamic time warping-based algorithm (*DTW_1RT*) and HMM-based algorithm *CTrack* [6]. However, it must be noted that *DTW_1RT* was trained using only one training sequence as it is designed to use only a single reference trace. Furthermore, the published version of CTrack uses accelerometer and

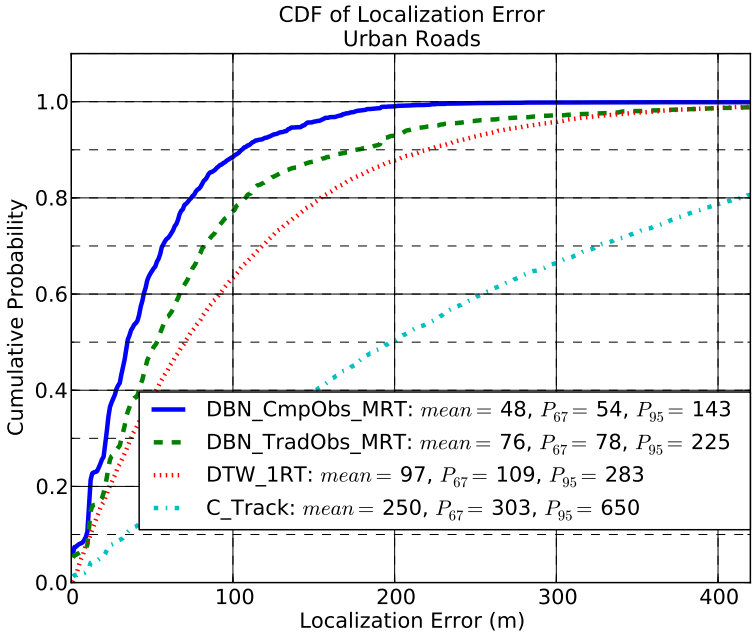
orientation sensor readings to acquire vehicle movement and turn hints, which are used to improve its tracking accuracy. Our implementation of *CTrack* did not use such sensor readings as they are not available in many tracking applications, which may have reduced its accuracy.

We used the same subscriber call trace dataset that was mentioned in Chapter 3, to evaluate the performance of the different algorithms in terms of estimating travel-times for a road segment. The experiment and the data analysis were carried out in the same way described in Section 3.4. Figure 4.4 shows the results of travel-time estimation. It is evident that *DBN_Cmp_Obs_MRT* is the best performing algorithm with a percentage error of 13% and a Pearson correlation coefficient of 0.93 with respect to actual travel-times. Further, the correlation is significant with a p-value of 0.0003. Results also indicate that *DBN_Trad_Obs_MRT* demostrates better correlation with actual travel-times than *DTW_1RT* with Pearson correaltion coefficients of 0.87 and 0.76 respectively, and p-values of 0.002 and 0.02 respectively. However, *DTW_1RT* has produced tracel-time estimates closer to the actual-values, especially for time slots with low actual travel-times, than *DBN_Trad_Obs_MRT*, which is indicated by their percentage errors of 14% and 39% respectively.

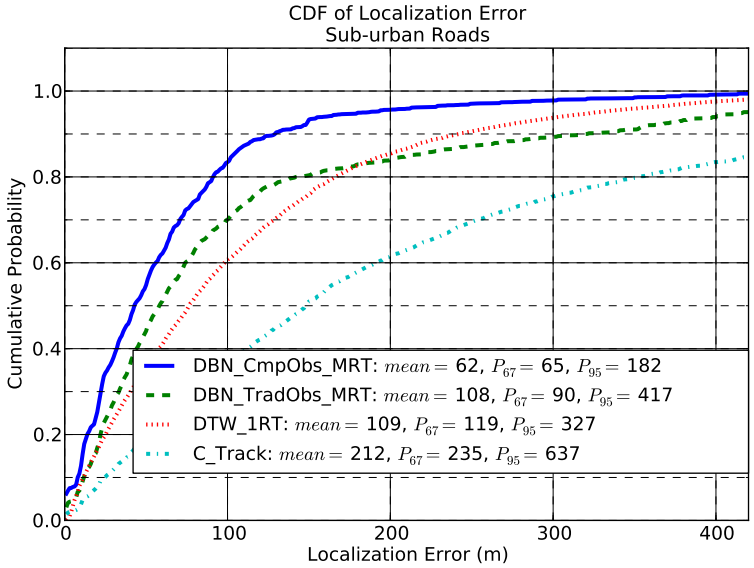
4.7. Conclusions

In this Chapter, we presented a complete observation model to represent the distribution of GSM received signal strength fingerprints and demonstrated how it is used with a dynamic Bayesian network to track vehicles from mobile phone signal traces. Our observation model accounts for the variation of cell towers present in fingerprints in addition to the variation of received signal strength levels. Our experiments on five different roads demonstrate the improvements in localization accuracy achieved by using the proposed observation model compared to existing methods.

Furthermore, our experiment using real mobile subscriber call traces indicate that the proposed DBN-based tracking algorithm with the complete observation model can be used to estimate travel-times for road segments, with reasonable accuracy, using mobile phone signal strength data.



(a) Urban roads



(b) Sub-urban roads

Key:
DBN_Cmp_Obs_MRT - Dynamic Bayesian Network (DBN)-based algorithm with the complete observation model proposed in this work.
DBN_Trad_Obs_MRT - Dynamic Bayesian Network (DBN)-based algorithm that uses the same state transition model as the DBN above but with the traditional observation model as defined in Equation 4.2.4 and used in [8].
C_Track - The HMM-based algorithm used in CTrack [6] whose observation model is given by Equation 4.2.6.
DTW_1RT - Dynamic Time Warping (DTW)-based algorithm as presented in Chapter 3.

Figure 4.3. Cumulative distribution of localization error.

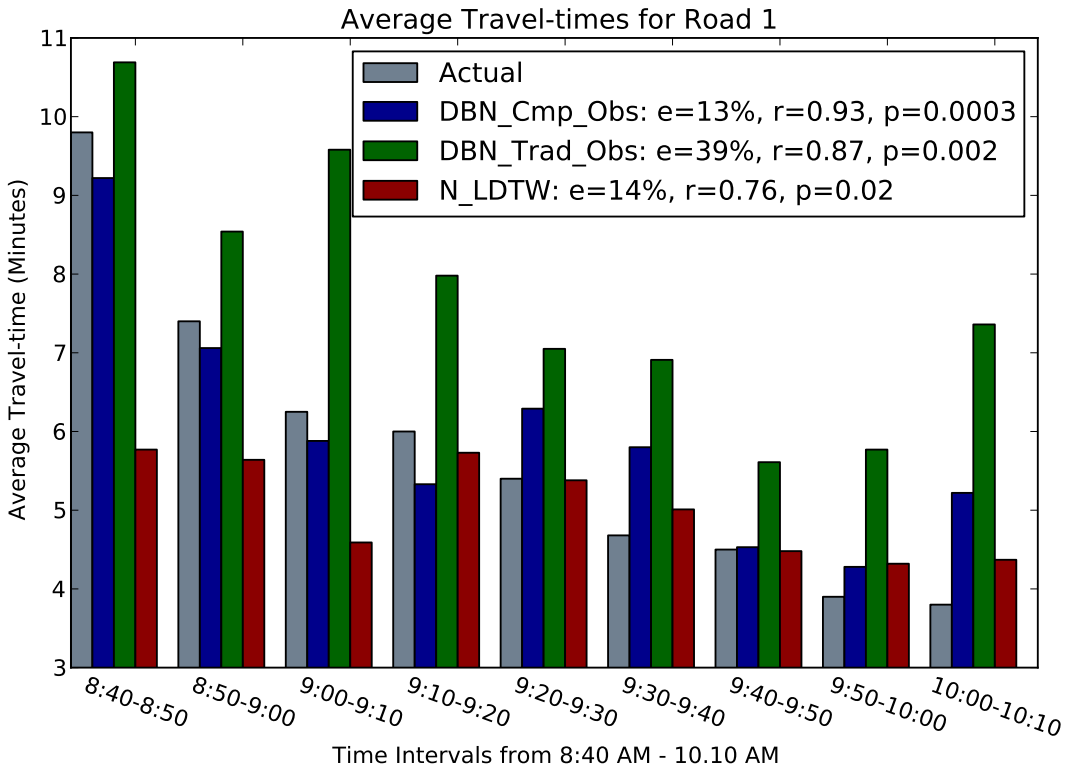


Figure 4.4. Travel-time estimation performance of the algorithms. The values e, r and p next to legend items are, average percentage error (Equation 3.4.1) and Pearson correlation coefficient w.r.t actual travel-times, and the two-tailed p-value of the correlation respectively, for each algorithm.

5. ESTIMATION OF MEAN RECEIVED SIGNAL STRENGTH LEVELS IN A MOBILE COMMUNICATION NETWORK WITHOUT WAR-DRIVING FOR TRACKING VEHICLES

5.1. Introduction

The vehicle tracking algorithms mentioned previously in this thesis and many other accurate methods presented in literature involve a supervised training phase that requires measured RSS fingerprints along with the locations at which they were measured. This training phase is needed for estimating various model parameters and/or calibration purposes. The required training data is typically acquired by measuring and recording fingerprints by driving on roads of interest (usually referred to as war-driving), which is both tedious and expensive. In this Chapter, we propose a method that can make use of fingerprint sequences generated by user phone calls to estimate the relevant model parameters in an unsupervised manner that requires no war-driving.

As mentioned in Chapter 4, the Bayesian estimation method for vehicle tracking from received signal strength (RSS) fingerprints involves an observation model that is used to compute the probability of observing an RSS fingerprint R_t at each grid-location s_i, denoted by $P(R_t|s_i)$. Computation of $P(R_t|s_i)$ using the observation model that we proposed (Equation 4.3.11) or those proposed in Literature (Equations 4.2.3 and 4.2.4 [8, 16, 18]) requires mean RSS levels at each grid-location, from each cell in the area of interest, i.e., $\mu^c_{s_i}, \forall s_i \in S, \forall c \in C$.

Traditionally, these mean RSS levels are estimated in a supervised learning phase using training fingerprints that have been recorded and assigned to grid-locations through war-driving. An alternative is to use radio propagation models to predict the aforementioned mean RSS levels [16]. However, the actual propagation of electromagnetic waves in a given environment may differ from the propagation model, which tends to diminish the accuracy of the predictions.

In this work, we propose the use of RSS fingerprints generated through phone calls made by road users to estimate the aforementioned mean RSS levels. This corresponds to unsupervised learning, where we treat the mean RSS levels as parameters of the observation model of a Dynamic

Bayesian Network (DBN) and estimate them through Viterbi learning using fingerprints generated by user calls as the training data. The same procedure also estimates the probabilities of successfully decoding the signal from each cell tower at each grid-location. The DBN mentioned here is the same as that in Section 4.5. Initial values for the mean RSS levels are predicted using a simple dual slope propagation model. Therefore, no war-driving is required. In the tracking phase, standard Viterbi decoding is used on the DBN with the estimated parameters to find the most probable sequence of grid-locations traversed by the vehicle.

5.2. Estimation of Mean Received Signal Strength Levels

5.2.1. Expectation-Maximization (EM) Algorithm

Since the mean RSS levels (μ_i^c 's) and the probabilities of successful decoding (w_i^c 's) at each grid-location are parameters of the observation model of the dynamic Bayesian network used for tracking in Chapter 4, we can use the Expectation-Maximization algorithm to estimate them. The Expectation-Maximization (EM) algorithm is widely used to estimate model parameters in the presence of latent/hidden variables as is the case with dynamic Bayesian networks. However, for improved efficiency, we actually use Viterbi learning (Maximization-Maximization), which is a faster and easier to implement approximation to the EM algorithm.

In this Section, we first briefly describe the EM algorithm. Then, we describe how we apply it specifically to estimate the mean RSS levels, along with the Viterbi approximation.

Given a set of hidden states $\mathbf{J}$, observed data $\mathbf{Z}$, and a set of parameters $\mathbf{\Theta}$, EM tries to maximize the posterior probability $P(\mathbf{\Theta}|\mathbf{Z})$ of the parameters given the observed data. The optimal set of parameters $\mathbf{\Theta}^*$ can be expressed as follows after marginalizing over the entire space of hidden states J^n.

$$\begin{aligned}
\mathbf{\Theta}^* &= \arg\max_{\mathbf{\Theta}} \sum_{\mathbf{J}\in J^n} P(\mathbf{\Theta}, \mathbf{J}|\mathbf{Z}) \\
&= \arg\max_{\mathbf{\Theta}} log\left(\sum_{\mathbf{J}\in J^n} P(\mathbf{\Theta}, \mathbf{J}, \mathbf{Z})/\mathbf{P}(\mathbf{Z})\right) \\
&= \arg\max_{\mathbf{\Theta}} log\left(\sum_{\mathbf{J}\in J^n} P(\mathbf{\Theta}, \mathbf{J}, \mathbf{Z})\right)
\end{aligned} \tag{5.2.1}$$

The EM algorithm finds a local maximum for $\mathbf{\Theta}^*$ by alternating between the following steps iteratively.

Expectation step: Compute

$$P(\mathbf{J}|\mathbf{Z}, \mathbf{\Theta^k}) \tag{5.2.2}$$

Maximization step: Compute

$$\mathbf{\Theta^{k+1}} = \arg\max_{\mathbf{\Theta}}[\mathbf{Q^k}(\mathbf{\Theta}) + \mathbf{logP}(\mathbf{\Theta})], \tag{5.2.3}$$

where $\mathbf{\Theta^k}$ are the parameter values at the start of the k^{th} iteration (initialized to some values before the first iteration), $\mathbf{\Theta^{k+1}}$ are the new values for the parameters, $P(\mathbf{\Theta})$ is the prior distribution of the parameters, and $Q^k(\mathbf{\Theta})$, as defined below, is the expectation of $logP(\mathbf{J}, \mathbf{Z}|\mathbf{\Theta})$ with respect to $P(\mathbf{J}|\mathbf{Z}, \mathbf{\Theta^k})$.

$$\begin{aligned} Q^k(\mathbf{\Theta}) &= \sum_{\mathbf{J}\in J^n} P(\mathbf{J}|\mathbf{Z}, \mathbf{\Theta^k})\mathbf{logP}(\mathbf{J}, \mathbf{Z}|\mathbf{\Theta}) \\ &= \sum_{\mathbf{J}\in J^n} P(\mathbf{J}|\mathbf{Z}, \mathbf{\Theta^k})[\mathbf{logP}(\mathbf{Z}|\mathbf{J}, \mathbf{\Theta}) + \mathbf{logP}(\mathbf{J}|\mathbf{\Theta})] \end{aligned} \tag{5.2.4}$$

We refer interested readers to [40] for further details of the Expectation-Maximization algorithm.

With our DBN, the set of parameters $\mathbf{\Theta}$ include the mean RSS levels and the probabilities of successful decoding. I.e., $\mathbf{\Theta} = \{\mu_\mathbf{i}^\mathbf{c}, \mathbf{w_i^c}; \forall \mathbf{c} \in \mathbf{C}, \forall \mathbf{s_i} \in \mathbf{S}\}$. The hidden sates are the set of grid-locations $S = \{s_1, s_2, \ldots s_N\}$ and the set of velocities $V = \{v_1, v_2, \ldots v_M\}$. Therefore, each $\mathbf{J} = \mathbf{j_1}, \mathbf{j_2}, \ldots, \mathbf{j_t}, \ldots, \mathbf{j_T}$ is a vector of hidden states such that each $j_t = (s_t, v_t) \in S \times V$. The observation $\mathbf{Z} = \mathbf{z_1}, \mathbf{z_2}, \ldots, \mathbf{z_t}, \ldots, \mathbf{z_T}$ is a time series, where z_t is the observation (a fingerprint) at time t. Therefore,

$$\begin{aligned} Q^k(\mathbf{\Theta}) &= \sum_{\mathbf{J}\in J^T} P(\mathbf{J}|\mathbf{Z}, \mathbf{\Theta^k})[\mathbf{logP}(\mathbf{Z}|\mathbf{J}, \mathbf{\Theta}) + \mathbf{logP}(\mathbf{J}|\mathbf{\Theta})] \\ &= \sum_{\mathbf{J}\in J^T} \sum_{t=1}^{T} P(\mathbf{J}|\mathbf{Z}, \mathbf{\Theta^k})[\mathbf{logP}(\mathbf{z_t}|\mathbf{j_t}, \mathbf{\Theta}) + \mathbf{logP}(\mathbf{j_t}|\mathbf{j_{t-1}}, \mathbf{\Theta})] \\ &= \sum_{t=1}^{T} \sum_{\mathbf{J}\in J^T} P(\mathbf{J}|\mathbf{Z}, \mathbf{\Theta^k})\mathbf{logP}(\mathbf{z_t}|\mathbf{j_t}, \mathbf{\Theta}) + \sum_{\mathbf{t=1}}^{\mathbf{T}} \sum_{\mathbf{J}\in J^T} \mathbf{P}(\mathbf{J}|\mathbf{Z}, \mathbf{\Theta^k})\mathbf{logP}(\mathbf{j_t}|\mathbf{j_{t-1}}, \mathbf{\Theta}) \end{aligned} \tag{5.2.5}$$

The second term in Equation 5.2.5 involves the state transition probabilities, which we do not attempt to optimize in this work. Therefore, we focus on the first term and define Q^k_{obs}, which represents the observation model as,

$$\begin{aligned} Q^k_{obs}(\mathbf{\Theta}) &= \sum_{t=1}^{T} \sum_{\mathbf{J}\in J^T} P(\mathbf{J}|\mathbf{Z}, \mathbf{\Theta^k}) \mathbf{logP}(\mathbf{z_t}|\mathbf{j_t}, \mathbf{\Theta}) \\ &= \sum_{t=1}^{T} \sum_{s_i\in S} \sum_{v_j\in V} \sum_{\mathbf{J}\in J^T} P(\mathbf{J}|\mathbf{Z}, \mathbf{\Theta^k}) \mathbf{logP}(\mathbf{z_t}|\mathbf{j_t}, \mathbf{\Theta}) \delta(\mathbf{s_t} = \mathbf{s_i}) \delta(\mathbf{v_t} = \mathbf{v_j}) \\ &= \sum_{t=1}^{T} \sum_{s_i\in S} \sum_{v_j\in V} P(s_t = s_i, v_t = v_j|\mathbf{Z}, \mathbf{\Theta^k}) \mathbf{logP}(\mathbf{z_t}|\mathbf{s_t} = \mathbf{s_i}, \mathbf{\Theta}). \end{aligned} \tag{5.2.6}$$

Suppose we have a collection $\mathbf{Z} = \{\mathbf{Z_1}, \mathbf{Z_2}, \ldots, \mathbf{Z_a}, \ldots, \mathbf{Z_A}\}$ of independent observation sequences, available for learning the parameters, where each $Z_a = R_{a,1}, \ldots, R_{a,t}, \ldots, R_{a,T_a}$ is a sequence of fingerprints generated by a phone with $R_{a,t}$ being the fingerprint generated at time t in the a^{th} sequence. Let $s_{a,t}$ be the location at which $R_{a,t}$ was generated and $v_{a,t}$ be the velocity of the relevant vehicle at time t. Then,

$$\begin{aligned} &Q^k_{obs}(\mathbf{\Theta}) \\ &= \sum_{a=1}^{A} \sum_{t=1}^{T_a} \sum_{s_i\in S} \sum_{v_j\in V} P(s_{a,t} = s_i, v_{a,t} = v_j|Z_a, \mathbf{\Theta^k}) \mathbf{logP}(\mathbf{R_{a,t}}|\mathbf{s_{a,t}} = \mathbf{s_i}, \mathbf{\Theta}) \\ &= \sum_{a=1}^{A} \sum_{t=1}^{T_a} \sum_{s_i\in S} \sum_{v_j\in V} f_{a,t}(s_i, v_j) logP(R_{a,t}|s_{a,t} = s_i, \mathbf{\Theta}), \\ &\text{where } f_{a,t}(s_i, v_j) = P(s_{a,t} = s_i, v_{a,t} = v_j|Z_a, \mathbf{\Theta^k}) \end{aligned} \tag{5.2.7}$$

Using Equations 4.3.9 and 4.3.10,

$$
\begin{aligned}
Q^k_{obs}(\mathbf{\Theta}) = & \\
& \sum_{a=1}^{A}\sum_{t=1}^{T_a}\sum_{s_i\in S}\sum_{v_j\in V}\sum_{c\in C_{a,t}} f_{a,t}(s_i,v_j)\left(\frac{-(y^c_{a,t}-\mu^c_i)^2}{2\sigma^2}\right) \\
& + \sum_{a=1}^{A}\sum_{t=1}^{T_a}\sum_{s_i\in S}\sum_{v_j\in V}\sum_{c\in C_{a,t}} f_{a,t}(s_i,v_j) ln(w^c_i) \\
& + \sum_{a=1}^{A}\sum_{t=1}^{T_a}\sum_{s_i\in S}\sum_{v_j\in V}\sum_{d\in C-C_{a,t}} f_{a,t}(s_i,v_j) ln\left(1 - w^d_i \int_{y^{min}_{a,t}}^{\infty} \frac{exp(-(\frac{y-\mu^d_i}{\sqrt{2}\sigma})^2)}{\sigma\sqrt{2\pi}} dy\right) \\
& - \sum_{a=1}^{A}\sum_{t=1}^{T_a}\sum_{s_i\in S}\sum_{v_j\in V}\sum_{c\in C_{a,t}} f_{a,t}(s_i,v_j) ln(\sigma\sqrt{2\pi}),
\end{aligned}
\tag{5.2.8}
$$

where $C_{a,t}$ is the set of Cell IDs present in $R_{a,t}$, $y^c_{a,t}$ is the RSS from cell $c \in C_{a,t}$ as reported in $R_{a,t}$, and

$$
y^{min}_{a,t} = \begin{cases} min_{c\in C}\{y^c_{a,t}\} & , if\ |C_{a,t}| = 7 \\ \tau & , otherwise \end{cases} .
$$

Our goal in the maximization step of the EM algorithm is to maximize $[Q^k_{obs}(\mathbf{\Theta}) + \mathbf{lnP}(\mathbf{\Theta})]$ over $\mathbf{\Theta}$, with $Q^k_{obs}(\mathbf{\Theta})$ as defined by Equation 4.3.11. In the next Section we describe the derivation of the prior distribution of parameters $P(\mathbf{\Theta})$.

5.2.2. Prior Distribution of Mean Received Signal Strength Levels

In this Section we derive the prior distribution of the parameters $P(\mathbf{\Theta})$, where $\mathbf{\Theta}$ includes the mean received signal strength level from cell c at grid-location s_i, μ^c_i and the probability of successfully decoding the signal from cell c at grid-location s_i, w^c_i, for each $c \in C$ and $s_i \in S$.

The mean received signal strength level from cell c at grid-location s_i, μ^c_i can be predicted using a suitable propagation model for electromagnetic waves. Let λ^c_i be the predicted value for μ^c_i. Then, using the dual slope propagation model [41], we can define λ^c_i as follows.

$$
\lambda^c_i = \begin{cases} Tx(c) - L_1 - 10n_1 log_{10}(d(s_i,c)) & , if\ d(s_i,c) \le r_b \\ Tx(c) - L_1 - 10n_1 log_{10}(r_b) - 10n_2 log_{10}(d(s_i,c)/r_b) & , otherwise \end{cases} , \tag{5.2.9}
$$

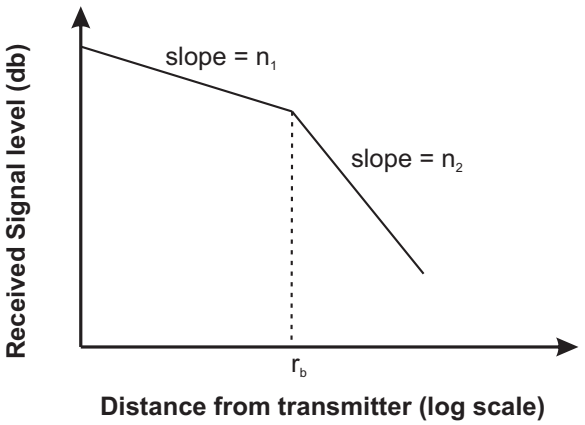


Figure 5.1. Dual slope propagation model.

where $Tx(c)$ is the transmit power level of cell c, $d(s_i, c)$ is the line of sight distance from the transmitting antenna of cell c to grid-location s_i, L_1 is the loss of power at 1m away from the transmit antenna, n_1 and n_2 are unit less constants referred to as path loss exponents, and r_b is a constant distance referred to as the breakpoint distance. Furthermore, note that all signal strength levels are in dbm. As the name *dual slope* implies, Equation 5.2.9 has two different slopes n_1 and n_2 for distances $r \leq r_b$ and $r > r_b$ respectively as shown in Figure 5.1. However, to avoid the sharp transition at $d(s_i, c) = r_b$, the following equation is commonly used.

$$\lambda_{s_i}^c = Tx(c) - L_1 - 10n_1 log_{10}(d(s_i, c)) - 10(n_2 - n_1) log_{10}(1 + d(s_i, c)/r_b) \tag{5.2.10}$$

Therefore, we also use Equation 5.2.10 to define $\lambda_{s_i}^c$ for all $s_i \in S$ and $c \in C$. We used typical values $n_1 = 2, n_2 = 4$, and $r_b = 200m$. L_1 was theoretically computed to be 32db as the free space path loss at 1m away from the antenna, for the 900MHz GSM carrier frequency [42].

However, due to shadowing caused by buildings and other obstructions, actual mean signal strength level μ_i^c may deviate from the mean predicted by the above model. This variation, called log-normal shadowing, is modeled by adding a zero mean Gaussian random variable X with standard deviation σ_s to the predicted mean as shown below.

$$\mu_i^c(dBm) = \lambda_i^c(dBm) + X. \tag{5.2.11}$$

Therefore, each μ_i^c is a normally distributed random variable with mean λ_i^c and standard deviation σ_s. Typical values for the standard deviation of shadow fading σ_s range from 6-10 dB in cities [43].

Equation 5.2.11 defines the prior distribution of each μ_i^c, individually. Then, we need to derive the joint distribution of $\mu_i^c, \forall c \in C$ and $s_i \in S$. We assume that signals from different cell towers are independent of each other. Therefore μ_i^c and μ_i^d are uncorrelated for any two cells $c, d \in C$ such that $c \neq d$. However, for any two grid-locations s_i and s_j that are sufficiently close to each other, there is correlation between μ_i^c and μ_j^c.

For any two grid-locations s_i and s_j separated by physical distance $d(s_i, s_j)$, it follows from Gudmundsons model [44] that the covariance between μ_i^c and μ_j^c is given by,

$$cov(\mu_i^c, \mu_j^c) = \sigma_s^2.exp\left(-\frac{d(s_i, s_j)}{d_{cor}} ln2\right), \tag{5.2.12}$$

where d_{cor} is a constant known as the decorrelation distance. Typically for a vehicular environment, $d_{cor} = 20m$ [44].

Following this model, we define an $N \times N$ covariance matrix K with elements $K[i, j]$ as defined below.

$$K[i, j] = \sigma_s^2.exp\left(-\frac{d(s_i, s_j)}{d_{cor}} ln2\right). \tag{5.2.13}$$

Since K is a covariance matrix, there exists a multivariate Gaussian distribution with K as its covariance matrix. Furthermore, as explained earlier, $\mu_1^c, \ldots, \mu_N^c$ are individually Gaussian random variables with means $\lambda_1^c, \ldots, \lambda_N^c$ respectively. Therefore, we model the prior joint distribution of $\vec{\mu_c} = [\mu_1^c, \ldots, \mu_N^c]$ as a multivariate Gaussian with mean vector $\vec{\lambda_c} = [\lambda_1^c, \ldots, \lambda_N^c]$ and covariance matrix K, and its probability density function is as shown below.

$$pdf(\vec{\mu_c}) = \frac{exp\left(-\frac{1}{2}(\vec{\mu_c} - \vec{\lambda_c})K^{-1}(\vec{\mu_c} - \vec{\lambda_c})^T\right)}{(2\pi)^{n/2}\sqrt{det(K)}} \tag{5.2.14}$$

Therefore,

$$lnP(\mathbf{\Theta}) = -\frac{1}{2}(\vec{\mu_c} - \vec{\lambda_c})K^{-1}(\vec{\mu_c} - \vec{\lambda_c})^T - ln\left((2\pi)^{n/2}\sqrt{det(K)}\right) \tag{5.2.15}$$

In this absence of further information, we assume that for each cell $c \in C$ and for each grid-location $s_i \in S$, the probability of successfully decoding the signal from cell c at location s_i, w_i^c is independently and uniformly distributed over the interval $[0, 1]$. Hence w_i^c are ignored in the optimization of $P(\mathbf{\Theta})$.

5.2.3. Optimization Procedure

We estimate the mean signal strength levels μ_i^c and the probabilities of successful decoding w_i^c as their values that maximize $[Q_{obs}^k(\mathbf{\Theta}) + \mathbf{lnP}(\mathbf{\Theta})]$ in the maximization step of the EM algorithm. However, in our case, there is no closed form solution for the values of μ_i^c's and w_i^c's that maximize $[Q_{obs}^k(\mathbf{\Theta}) + \mathbf{lnP}(\mathbf{\Theta})]$, hence, the optimization cannot be done analytically. Therefore, we use the Truncated Newton method to numerically minimize $-[Q_{obs}^k + lnP(\mathbf{\Theta})]$. Starting from some initial values for the variables of interest, the Truncated Newton method uses the partial derivatives of the expression to be minimized to iteratively move along directions that decrease the required expression.

The partial derivatives of $[Q_{obs}^k(\mathbf{\Theta}) + \mathbf{lnP}(\mathbf{\Theta})]$ with respect to the means μ_i^c are as follows.

$$
\begin{aligned}
&\forall c \in C, s_i \in S, \\
&\frac{\partial [Q_{obs}^k + lnP(\mathbf{\Theta})]}{\partial \mu_i^c} = \sum_{a=1}^{A} \sum_{t=1}^{T_a} \frac{(y_{a,t}^c - \mu_i^c)}{\sigma} g_{a,t}(c) \sum_{v_j \in V} f_{a,t}(s_i, v_j) \\
&\qquad - \sum_{a=1}^{A} \sum_{t=1}^{T_a} \frac{w_i^c \frac{\partial}{\partial \mu_i^c} \left(\int_{y_{a,t}^{min}}^{\infty} \frac{exp(-(\frac{y-\mu_i^c}{\sqrt{2}\sigma})^2)}{\sigma\sqrt{2\pi}} dy \right)}{1 - w_i^c \int_{y_{a,t}^{min}}^{\infty} \frac{exp(-(\frac{y-\mu_i^c}{\sqrt{2}\sigma})^2)}{\sigma\sqrt{2\pi}} dy} [1 - g_{a,t}(c)] \sum_{v_j \in V} f_{a,t}(s_i, v_j) \\
&\qquad - K^{-1}[i, :](\vec{\mu_c} - \vec{\lambda_c})^T
\end{aligned}
\tag{5.2.16}
$$

where,

$$
g_{a,t}(c) = \begin{cases} 1 & , if\ c \in C_{a,t} \\ 0 & , otherwise \end{cases},
$$

and $K^{-1}[i, :]$ is the i^{th} row of K^{-1}. Note that this follows from the symmetry of K^{-1}, which makes it possible to derive $\partial lnP(\mathbf{\Theta})/\partial \mu_{\mathbf{i}}^{\mathbf{c}} = -\mathbf{K^{-1}}[\mathbf{i}, :]$.

Using Leibniz's rule for differentiation under the integral sign,

$$\begin{aligned}
\frac{\partial}{\partial \mu_i^c}\left(\int_{y_{a,t}^{min}}^{\infty} \frac{exp(-(\frac{y-\mu_i^c}{\sqrt{2}\sigma})^2)}{\sigma\sqrt{2\pi}}dy\right) &= \int_{y_{a,t}^{min}}^{\infty} \frac{\partial}{\partial \mu_i^c}\left(\frac{exp(-(\frac{y-\mu_i^c}{\sqrt{2}\sigma})^2)}{\sigma\sqrt{2\pi}}\right)dy \\
&= \int_{y_{a,t}^{min}}^{\infty} \frac{2(y-\mu_i^c)}{2\sigma^2}\left(\frac{exp(-(\frac{y-\mu_i^c}{\sigma})^2)}{\sigma\sqrt{2\pi}}\right)dy \\
&= \frac{1}{\sigma\sqrt{2\pi}}\left[-exp\left(-\left(\frac{y-\mu_i^c}{\sqrt{2}\sigma}\right)^2\right)\right]_{y_{a,t}^{min}}^{\infty} \\
&= \frac{1}{\sigma\sqrt{2\pi}} exp\left(-\left(\frac{y_{a,t}^{min}-\mu_i^c}{\sqrt{2}\sigma}\right)^2\right).
\end{aligned} \tag{5.2.17}$$

Therefore,

$$\begin{aligned}
&\forall c \in C, s_i \in S, \\
&\frac{\partial[Q_{obs}^k + lnP(\mathbf{\Theta})]}{\partial \mu_i^c} = \sum_{a=1}^{A}\sum_{t=1}^{T_a} \frac{(y_{a,t}^c - \mu_i^c)}{\sigma} g_t(c) \sum_{v_j \in V} f_{a,t}(s_i, v_j) \\
&\quad - \sum_{a=1}^{A}\sum_{t=1}^{T_a} \frac{w_i^c \frac{1}{\sigma\sqrt{2\pi}} exp\left(-\left(\frac{y_{a,t}^{min}-\mu_i^c}{\sqrt{2}\sigma}\right)^2\right)}{1 - w_i^c \int_{y_{a,t}^{min}}^{\infty} \frac{exp(-(\frac{y-\mu_i^c}{\sqrt{2}\sigma})^2)}{\sigma\sqrt{2\pi}}dy}[1 - g_t(c)] \sum_{v_j \in V} f_{a,t}(s_i, v_j) \\
&\quad - K^{-1}[i,:](\vec{\mu_c} - \vec{\lambda_c})^T.
\end{aligned} \tag{5.2.18}$$

Partial derivatives with respect to w_i^c are as follows,

$$\begin{aligned}
&\forall c \in C, s_i \in S, \\
&\frac{\partial[Q_{obs}^k + lnP(\mathbf{\Theta})]}{\partial w_i^c} = \frac{1}{w_i^c} \sum_{a=1}^{A}\sum_{t=1}^{T_a} g_{a,t}(c) \sum_{v_j \in V} f_{a,t}(s_i, v_j) \\
&\quad - \sum_{a=1}^{A}\sum_{t=1}^{T_a} \frac{\int_{y_{a,t}^{min}}^{\infty} \frac{exp(-(\frac{y-\mu_i^c}{\sqrt{2}\sigma})^2)}{\sigma\sqrt{2\pi}}dy}{1 - w_i^c \int_{y_{a,t}^{min}}^{\infty} \frac{exp(-(\frac{y-\mu_i^c}{\sqrt{2}\sigma})^2)}{\sigma\sqrt{2\pi}}dy}[1 - g_{a,t}(c)] \sum_{v_j \in V} f_{a,t}(s_i, v_j).
\end{aligned} \tag{5.2.19}$$

When applied to a Dynamic Bayesian Network, the expectation step of the EM algorithm is carried out using the Forward-Backward algorithm, which can be difficult to implement due floating point underflow. Therefore, we use Viterbi learning, also known as Maximization-Maximization as

a more efficient and convenient approximation to the EM algorithm. With Viterbi learning, the expectation step of the EM algorithm is replaced with a maximization step, which is computed using viterbi decoding. In the k^{th} iteration of Viterbi learning, for each observation sequence $Z_a = R_{a,1}, \ldots, R_{a,t}, \ldots, R_{a,T_a}$, Viterbi decoding with current parameter values is used to find the most probable state sequence $(s_{a,1}, v_{a,1}), \ldots, (s_{a,t}, v_{a,t}), \ldots, (s_{a,T_a}, v_{a,T_a})$. Then, $f_{a,t}(s_i, v_j)$ defined in Equation 5.2.7, is redefined as,

$$f_{a,t}(s_i, v_j) = \begin{cases} 1 & , if\ s_{a,t} = s_i\ and\ v_{a,t} = v_j \\ 0 & , otherwise \end{cases} . \tag{5.2.20}$$

All equations that follow Equation 5.2.7 are used as they are along with the above redefinition of $f_{a,t}(s_i, v_j)$.

Once the estimation of the mean signal strength levels and the probabilities of successful decoding is complete, we use Viterbi decoding as in Section 4.5 to track vehicles.

5.3. Experimental Evaluation and Discussion of Results

We conducted experiments to compare the vehicle tracking performance of the proposed DBN-based algorithm under three different modes of parameter estimation. They are, model parameters estimated through supervised maximum likelihood estimation as in Chapter 4, model parameters predicted solely using the dual slope propagation model, and model parameters estimated through unsupervised Viterbi learning as proposed in this Chapter. Additionally, we compare with the DBN-based algorithm that uses the traditional observation model as well.

Ideally, we would have used the call trace dataset that was used in the previous Chapters to conduct these experiments. However, due to security reasons, the operator was unable to provide us with the locations of their cell towers involved with the data. As the tower locations are required to predict initial RSS levels, we were unable to use this dataset in our experiments. Therefore, we used the data that we collected by driving on the roads. This data corresponds to a different operator who provided us with the cell tower locations but was unable to provide real user call traces. To simulate user call traces and unsupervised learning, we ignored the location information (GPS coordinates) of the collected data.

Our data includes signal strength fingerprints on five (three urban and two suburban) different roads. We drove several trips along each road and collected a set of fingerprint sequences $Z = \{Z_1, \ldots, Z_a, \ldots, Z_A\}$, where each $Z_a = R_{a,1}, \ldots, R_{a,t}, \ldots, R_{a,T_a}$ is the sequence of fingerprints collected in the a^{th} trip, with each $R_{a,t}$ being a signal strength fingerprint. Each fingerprint $R_{a,t}$ has associated with it, a time stamp and GPS coordinates of the location $x_{a,t}$ at which it was generated. Our algorithm operates in two phases; the unsupervised learning phase, where the parameters of the DBN are estimated from user call traces, and the tracking phase, where the vehicles are tracked. The GPS coordinates of $x_{a,t}$ are used only as ground truth for evaluation during the tracking phase.

For each road, we use one of the fingerprint sequences $Z_q = R_{q,1}, \ldots, R_{q,t}, \ldots, R_{q,T_q} \in Z$ as the test trace from which the vehicle is tracked, and the remaining traces (ignoring the GPS coordinates) as training data analogous to user call traces to estimate model parameters through Viterbi learning. The fingerprint traces we have collected are longer than those generated during typical phone calls. Analysis of a mobile subscriber call trace dataset revealed that the duration of user calls are approximately exponentially distributed with mean 110 seconds. Therefore, to better resemble the real situation of a passive, network side tracking application, we break Z_q into random sub-sequences of the form $Z_q[e, f] = R_{q,e}, \ldots, R_{q,f}$, where e and f are random indices with $1 \leq e < f \leq T_q$ such that the sequence length $f - e$ is distributed according to the aforementioned distribution, and use these sub-sequences for tracking.

In the tracking phase, for each $Z_q[e, f]$, we use Viterbi decoding to find the most probable state sequence $(s_{q,e}, v_{q,e}), \ldots, (s_{q,t}, v_{q,t}), \ldots, (s_{q,f}, v_{q,f})$, where $s_{q,t}$ and $v_{q,t}$ are the location and velocity for $R_{q,t}$ respectively, as predicted by Viterbi decoding. Then, for each $R_{q,t}$, the localization error is defined as the distance between the true location $x_{q,t}$ and the predicted location $s_{q,t}$.

Figures 5.2(a) and 5.2(b) show the cumulative distribution of the localization error for the urban and suburban roads respectively. These figures show that DBN-based vehicle tracking with model parameters (mean signal strength levels and the probabilities of successful decoding) estimated with the aid of user call traces through Viterbi learning (*DBN_CmpObs_VitLrn*) reduces the mean localization error compared to DBN-based vehicle tracking with mean signal strength levels predicted using the dual slope propagation model (*DBN_TradObs_NoLrn*) by 46% and 40% for urban and suburban roads respectively. Note that *DBN_CmpObs_VitLrn* uses the complete obser-

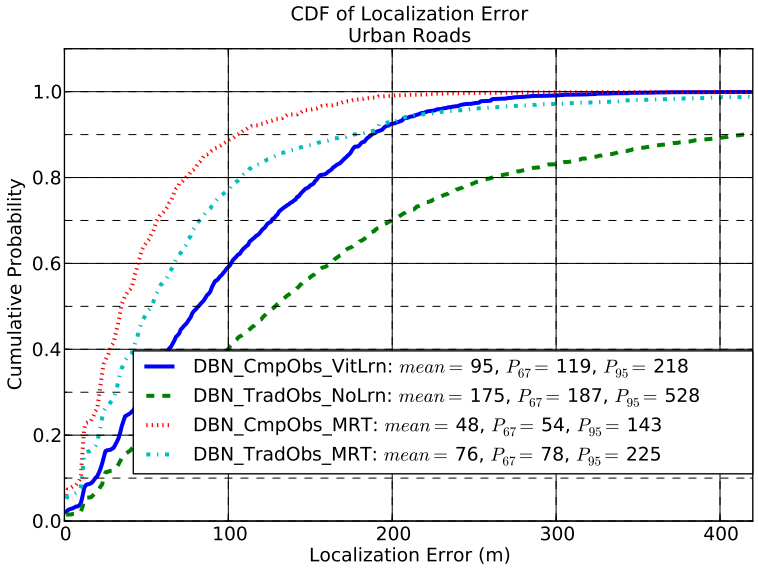
vation model proposed in Chapter 4 while *DBN_TradObs_NoLrn* uses the traditional observation model given in Equation 4.2.3.

Furthermore, for urban roads, the mean localization error of *DBN_CmpObs_VitLrn* is within 25% of that of *DBN_TradObs_MRT*, which is the DBN-based algorithm that uses the traditional observation model defined in Equation 4.2.4 that is trained using fingerprints collected by driving on the road. For suburban roads, *DBN_CmpObs_VitLrn* performs better than *DBN_TradObs_MRT* with a 36% lower mean localization error. The mean localization error of *DBN_CmpObs_VitLrn* for suburban roads is within 12% of that of *DBN_Cmp_Obs_MRT*, which s the DBN-based algorithm with the complete observation model that is trained using fingerprints collected by driving on the road. However, for urban roads, the mean localization error of *DBN_CmpObs_VitLrn* is 98% higher than that of *DBN_Cmp_Obs_MRT*.

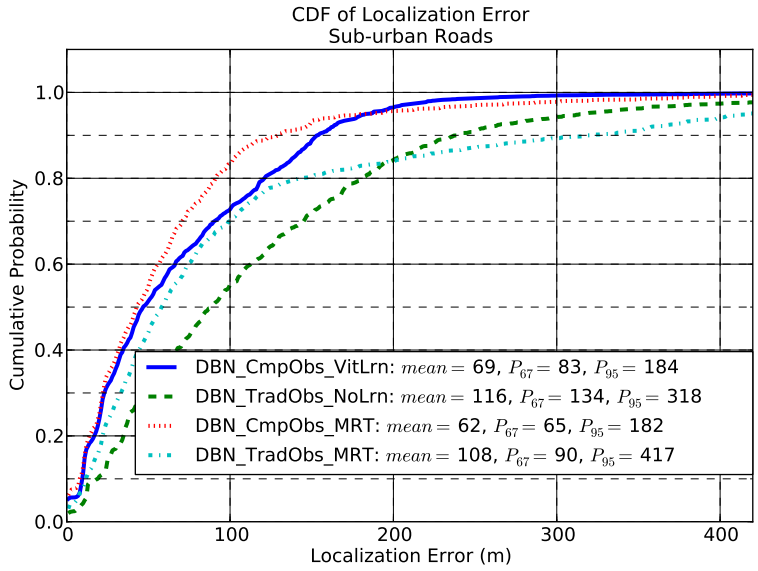
5.4. Conclusions

In this Chapter, we presented a Dynamic Bayesian Network-based tracking algorithm with its parameters estimated through unsupervised learning to track vehicles from mobile phone signal strength levels. The presented algorithm is unsupervised as it does *not* involve a training phase that requires signal strength measurements pre-assigned to grid-locations in the area of interest.

Our experimental results indicate that it is possible to use call traces made by road users to estimate mean signal strength levels in an unsupervised manner, and that it achieves better performance compared to using mean signal strength levels predicted by a simple propagation model. Furthermore, it is evident that for suburban roads, mean localization error of vehicle tracking with unsupervised-learning is within 12% of that of tracking with the best supervised-learning algorithm. However, for urban roads, mean localization error of vehicle tracking with unsupervised-learning is 98% higher than that of tracking with the best supervised learning algorithm. A possible reason for unsupervised learning performing almost as well as supervised learning for suburban roads but not for urban roads is that the suburban roads used in this study were mostly surrounded by short buildings and only a few trees, where as the urban roads were surrounded by many tall buildings, possibly obstructing the paths of signal propagation. Therefore, the received signal strength levels predicted by the propagation model can be more accurate for the suburban roads than for the urban roads. Since these predicted signal levels are used as initial values in Viterbi learning, more accurate predictions lead to better learning.



(a) Urban roads



(b) Sub-urban roads

Key:
DBN_CmpObs_VitLrn - Dynamic Bayesian Network (DBN)-based algorithm with model parameters estimated using Viterbi learning as proposed in this Chapter.
DBN_TradObs_NoLrn - Dynamic Bayesian Network (DBN)-based algorithm using mean signal strength levels as predicted by the dual slope model.
DBN_Cmp_Obs_MRT - Dynamic Bayesian Network (DBN)-based algorithm with the complete observation model proposed in Chapter 4, which uses training data labeled with GPS coordinates.
DBN_Trad_Obs_MRT - Dynamic Bayesian Network (DBN)-based algorithm that uses the same state transition model as the DBN above but with the traditional observation model as defined in Equation 4.2.4 and used in [8].

Figure 5.2. Cumulative distribution of localization error.

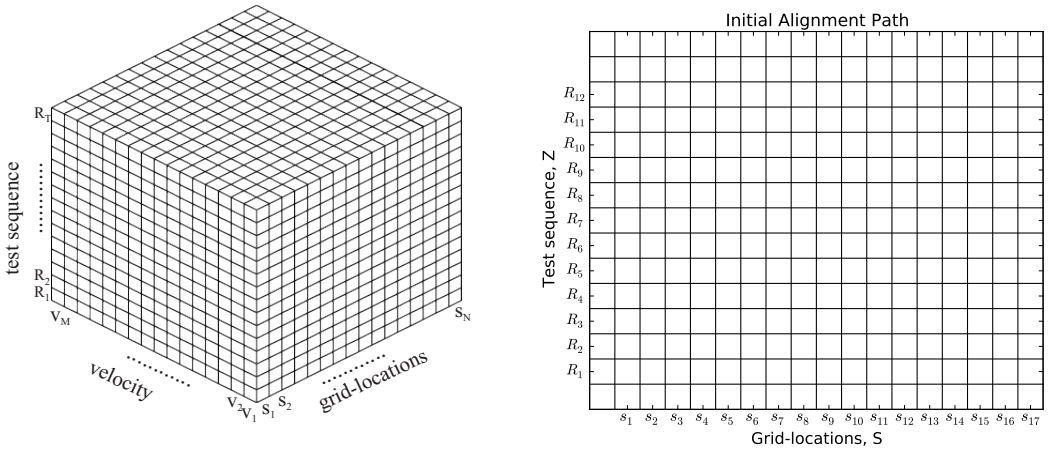
6. NEAREST-NEIGHBOR ASSISTED BANDED VITERBI DECODING FOR TRACKING VEHICLE TRAJECTORIES FROM MOBILE PHONE SIGNAL TRACES

6.1. Introduction

The time complexity of the Dynamic Bayesian Network(DBN)-based and dynamic time warping(DTW)-based tracking algorithms presented in the previous Chapters are not encouraging for applications such as traffic estimation, which require processing of a large number of signal strength sequences in real time. The time complexity of DTW is $O(TN)$, where T and N are the lengths of the two sequences (in our case, test and reference sequences) being aligned. The time complexity of Viterbi decoding as applied to our proposed DBN is $O(N^2M^2T)$, where N and M are the number of grid-locations and discrete velocities respectively. Using the constraints associated with vehicular movement, as discussed in Section 4.5.1, time complexity of viterbi decoding can be reduced to $O(KNMT)$, where K is some constant much less than NM. However, this is still worse than quadratic complexity and therefore, is not desirable.

In this Chapter, we present an algorithm that uses fast nearest-neighbor search to further restrict the number of grid-locations considered in the computation of Viterbi decoding in Section 4.5.1. As introduced in Chapter 4.1, let $S = \{s_1, s_2, \ldots, s_i, \ldots, s_N\}$ be a set of grid-locations in the area of interest, $V = \{v_1, v_2, \ldots, v_M\}$ be a set of discrete velocities a vehicle may move at, and $Z = R_1, R_2, \ldots, R_t, \ldots R_T$ be a test signal strength sequence, where each R_t is a signal strength fingerprint. Recall from Equation 4.5.8 that $\alpha(i, j, t)$ needs to be computed for each time t, for each grid-location $s_i \in S$, and for each velocity $v_j \in V$. This can be visualized as filling values of the cells in a cube as shown in Figure 6.1(a). In this work, we attempt to restrict the number of grid-locations but not the velocities. Therefore, we will consider, for each velocity v_j, the *grid location-time* plane, which amounts to computing the cells of a matrix as shown in Figure 6.1(b).

Our experiments in Chapter 3 indicate that even though nearest-neighbor based localization is not as accurate as sequence alignment-based or DBN-based methods, it is still reasonably accurate for a rather simple algorithm. Therefore, we use nearest-neighbor search to determine for each



(a) Cube of values to compute in unrestricted Viterbi Decoding.

(b) Matrix of values to compute in unrestricted Viterbi Decoding for each v_j.

Figure 6.1. Values to compute in unrestricted Viterbi Decoding.

fingerprint R_t in the test sequence, the closest grid-location in S based on Euclidean distance between signal strength levels (defined in Section 6.2), which yields an initial sequence of grid-locations that the vehicle may have traversed. Figure 6.2 shows an example of such an initial sequence of grid-locations. In an attempt to correct any errors in this initial path, we then perform Viterbi decoding restricted to a band of grid-locations around this initial sequence as explained in Section 6.3.

However, in its traditional form, nearest-neighbor search itself has quadratic time complexity $O(NT)$ when applied to find an initial sequence of grid-locations, which is not desirable. Data structures such as kd-trees [45], R-trees [46] and PAT-tress [47] improve the efficiency of nearest-neighbor search for moderate dimensions (up to 12). In our case, the dimensionality of the signal strength sequences will be the number of cells in the area of interest, which can typically be more than 20 over a short road segment of 2-3 km, in urban areas. Therefore, as described in Section 6.2, we propose a fast nearest-neighbor search algorithm for high dimensional signal sequences that uses the properties of a metric space to avoid having to search the entire space of grid-locations.

6.2. Fast Nearest-Neighbor Search

In this Section we describe a fast nearest-neighbor search algorithm to locate for any given fingerprint R_t, the *nearest* grid-location in S. The term *nearest*, as defined below, is based on

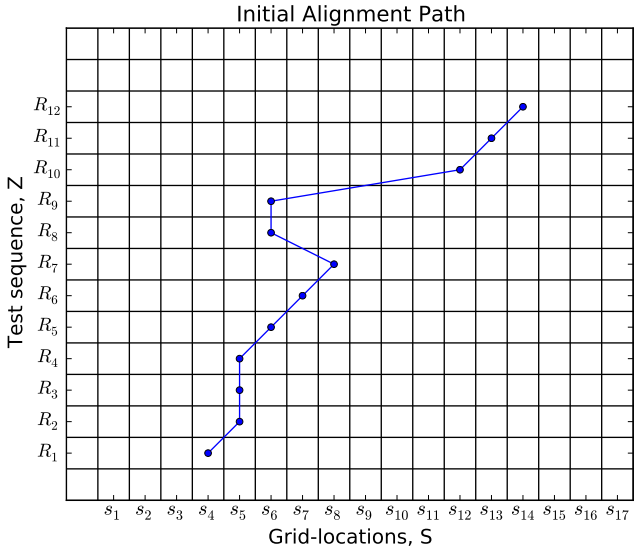


Figure 6.2. Initial sequence of grid-locations returned by the nearest-neighbor search.

Euclidean signal distance between a given test fingerprint R_t and any grid-location in $s_i \in S$, denoted by $d(R_t, s_i)$.

Let C be the set of IDs of all cell towers in the area of interest, C_t be the set of cells present in fingerprint R_t, y_t^c be the signal strength level from cell c as reported in R_t, μ_i^c be the mean signal strength level from cell c at location s_i, C_i be the set of cell towers for which μ_i^c has been estimated successfully at location s_i, and β be a small value used to represent signal levels of cells not reported in R_t or cells for which mean signal strength level is unavailable.

$$d(R_t, s_i) = \sqrt{\sum_{\forall c \in C_t \cap C_i} (y_t^c - \mu_i^c)^2 + \sum_{\forall c \in C_i \cap \neg C_t} (\beta - \mu_i^c)^2 + \sum_{\forall c \in \neg C_i \cap C_t} (y_t^c - \beta)^2}. \tag{6.2.1}$$

We search for the nearest-neighbor of a given test fingerprint R_t by starting with an initial guess s_a (How to make a good guess is explained later in this Section). We avoid having to search the entire set of grid-locations by using the neighbor lists of the grid-locations, which we create in a separate pre-processing phase. For each grid-location s_i in the area of interest, its neighbor list $list[s_i]$ contains the signal distances from s_i to all other grid-locations, and it is sorted in the ascending order. Using these neighbor lists, the inequality shown in Equation 6.2.2c, and the initial guess s_a, we restrict our search to a subset of the grid-locations in the area of interest.

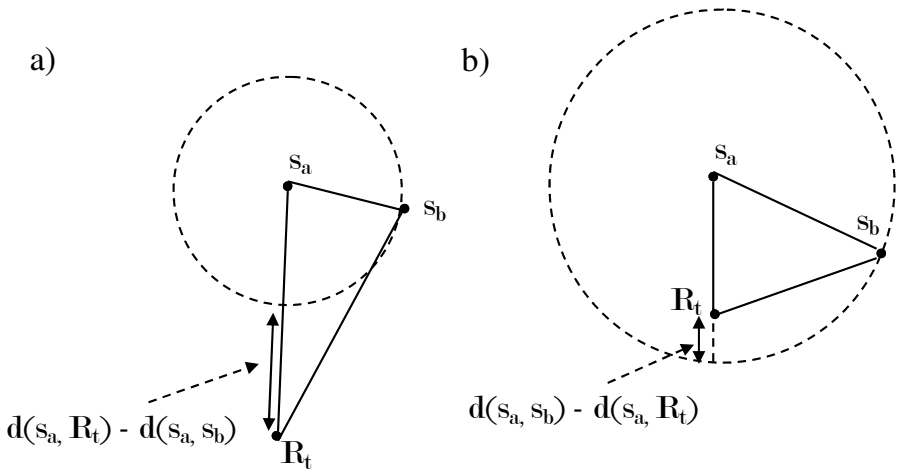


Figure 6.3. Inequalities for a metric space.

Since the Euclidean distance between signal levels $d(.)$ is a metric, the triangle inequalities shown in Equations 6.2.2a and 6.2.2b hold between any three points s_a, s_b, and R_t (Figure 6.3 for two-dimensional points). As mentioned before, let R_t be the test point for which we need to find the nearest-neighbor, s_a be the initial guess, and s_b be any other point on the reference trace. Then, the distance $d(s_b, R_t)$, between s_b and R_t, has a lower bound as shown in Equation 6.2.2b, which is a direct derivation from the triangle inequalities. This inequality is shown graphically in Figure 6.3 for a two-dimensional space.

$$\begin{aligned} d(s_b, R_t) + d(s_a, s_b) &\geq d(s_a, R_t) \\ d(s_b, R_t) &\geq d(s_a, R_t) - d(s_a, s_b) \end{aligned} \tag{6.2.2a}$$

$$\begin{aligned} d(s_b, R_t) + d(s_a, R_t) &\geq d(s_a, s_b) \\ d(s_b, R_t) &\geq d(s_a, s_b) - d(s_a, R_t) \end{aligned} \tag{6.2.2b}$$

From Equations 6.2.2a and 6.2.2b,

$$d(s_b, R_t) \geq |d(s_a, s_b) - d(s_a, R_t)| \tag{6.2.2c}$$

Algorithm 2 outlines our approach. First, we locate the index i_{pt} before which the value $d(s_a, R_t)$ can be inserted in $list[s_a]$ (line 4). Then, we search the grid-locations in $list[s_a]$ to the right of i_{pt} (lines 7-17), and update the current best nearest neighbor and its distance (lines 9-11). Note that the grid-locations in $list[s_a]$ are sorted in the ascending order of distance to s_a. Hence, $list[s_a]$ has a different index than the original set S. Therefore, at line 8, a function is used to retrieve the original grid-location index of the item at list index u. The distance value $list[s_a][u]$ at the u^{th} index of the list increases monotonically as u increases from $(i_{pt} + 1)$ to N, and so does $|d(s_a, R_t) - list[s_a][u]|$ because $d(s_a, R_t)$ is constant for this search. We stop searching in this direction as soon as $|d(s_a, R_t) - list[s_a][u]|$ becomes greater than the current best nearest-neighbors distance (lines 14, 15) because according to Equation 6.2.2b, no grid-location in the list further to the right can be any closer to y_i.

Searching $list[s_a]$ to the left of i_{pt} works in a similar manner (lines 18-28). The distance value $list[s_a][u]$ at the u^{th} index of the list decreases monotonically as u decreases from i_{pt} to 0, which again causes an increase in $|d(s_a, R_t) - list[s_a][u]|$. Just as before, we stop the search as soon as $|d(s_a, R_t) - list[s_a][u]|$ becomes greater than the current best nearest-neighbors distance. At the end of this algorithm, the true nearest-neighbor of R_t would be found.

Making a good initial guess s_a for the nearest-neighbor of R_t is important to achieve early termination of search with the above algorithm. Fortunately, it is not difficult with our signal sequences because, if s_c is the true nearest-neighbor of R_{t-1} then, the nearest-neighbor of R_t is expected to be close in geographical space to s_c. Therefore, we search a few (3 in our experiments) grid points to the left and right of s_c and choose the closest to R_t as the initial guess. Of course, to find the nearest-neighbor of the first test fingerprint R_1, the whole set of grid-locations will have to be searched.

6.3. Banded Viterbi Decoding

The nearest-neighbor search will assign each fingerprint R_t in the test sequence, some grid-location $s_i \in S$. This form an initial sequence of grid-locations that may look like the one shown in Figure 6.2. Note that in Figure 6.2, test fingerprints R_7 and R_8 are assigned to grid-locations s_8 and s_6 respectively. Since $s_1, s_2, \ldots, s_N$ are ordered as they appear along a road, s_6 comes before s_8. Therefore, this suggests a sudden reversal of the direction of motion of the vehicle, which is

Algorithm 2 Nearest-Neighbor search

Input: R_t ▷ Test fingerprint for which NN is needed
Input: s_a ▷ Initial guess for the NN
Input: $list[s_a]$ ▷ Sorted list of neighbor distances of s_a
$i_{pt} \leftarrow insertion\_index(d(s_a, R_t), list[s_a])$
$current\_best\_NN\_dist \leftarrow d(s_a, R_t)$
$current\_best\_NN\_index \leftarrow a$
for $u = (i_{pt} + 1)$ to M **do**
    $v \leftarrow index\_of(list[s_a][u])$
    if $d(s_v, R_t) < current\_best\_NN\_dist$ **then**
        $current\_best\_NN\_dist \leftarrow d(s_v, R_t)$
        $current\_best\_NN\_index \leftarrow v$
    end if
    $Lower\_Bound = |d(s_a, R_t) - list[s_a][u]|$
    if $Lower\_Bound \geq current\_best\_NN\_dist$ **then**
        break
    end if
end for
for $u = i_{pt}$ to 0 **do**
    $v \leftarrow index\_of(list[s_a][u])$
    if $d(s_v, R_t) < current\_best\_NN\_dist$ **then**
        $current\_best\_NN\_dist \leftarrow d(s_v, R_t)$
        $current\_best\_NN\_index \leftarrow v$
    end if
    $Lower\_Bound = |d(s_a, R_t) - list[s_a][u]|$
    if $Lower\_Bound \geq current\_best\_NN\_dist$ **then**
        break
    end if
end for
return $current\_best\_NN\_index$

highly unlikely. However, this kind of assignments are possible with a nearest-neighbor-based path, and should be corrected later.

We attempt to correct the assignment errors that may have been introduced in the nearest-neighbor search, by performing Viterbi decoding around the initial path. However, note that during Viterbi decoding, we use the same observation model as in Equation 4.3.11, not the Euclidean distance. First, let w be the width of a band around the initial path as shown in Figure 6.5. Performing Viterbi decoding within this band (shaded area in Figure 6.5) will correct any violations of the direction of motion and is expected to produce a better sequence of grid-locations for the vehicle. However, at a point such as (s_{12}, s_{10}), the path will become discontinuous as there is discontinuity in the band. To overcome such problems, at each test fingerprint $R_t \in Z$, we widen

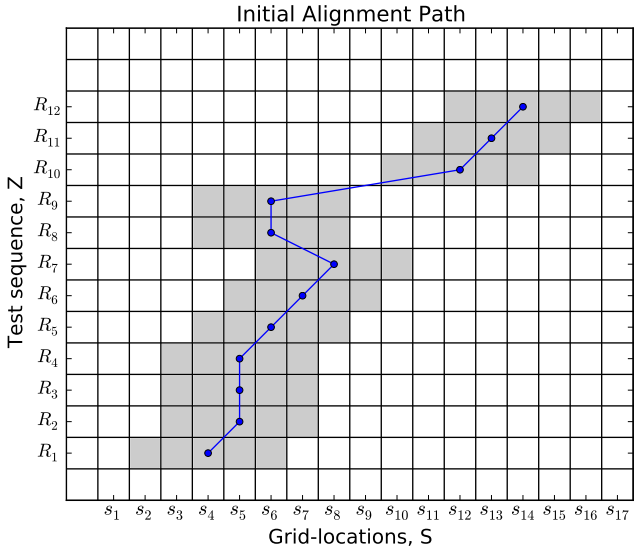


Figure 6.4. A band of constant width ($w = \pm 2$) around the initial alignment path returned by the nearest-neighbor search.

the band as needed based on the nearest-neighbors of test fingerprints adjacent to R_t. As shown below, we define the interval B_t as the band at each test fingerprint R_t.

$$
\begin{aligned}
&\textit{Let } NN(R_t) \textit{ be the index of the nearest-neighbor of } R_t \\
&\forall\ 0 \leq j \leq N \\
&B_t^+ = \max_{t-w \leq k \leq t+w} \{NN(R_t)\} \\
&B_t^- = \min_{t-w \leq k \leq t+w} \{NN(R_t)\} \\
&B_t = [min\{NN(R_t) - w, B_t^-\}, max\{NN(R_t) + w, B_t^+\}]
\end{aligned}
\tag{6.3.1}
$$

With this computation of the band, the bandwidth becomes wider in areas where the nearest neighbors of test fingerprints near R_t are more spread out over grid-locations. Therefore, it reduces discontinuities in the band.

6.4. Experimental Evaluation

We evaluated the performance of our proposed algorithm with the same signal strength sequences used in Chapter 4 and experiments performed in the same manner. Figure 6.6 shows the variation of the percentage increase in the mean localization error with respect to unrestricted Viterbi decoding for different values of the bandwidth parameter (w). As expected, results indicate

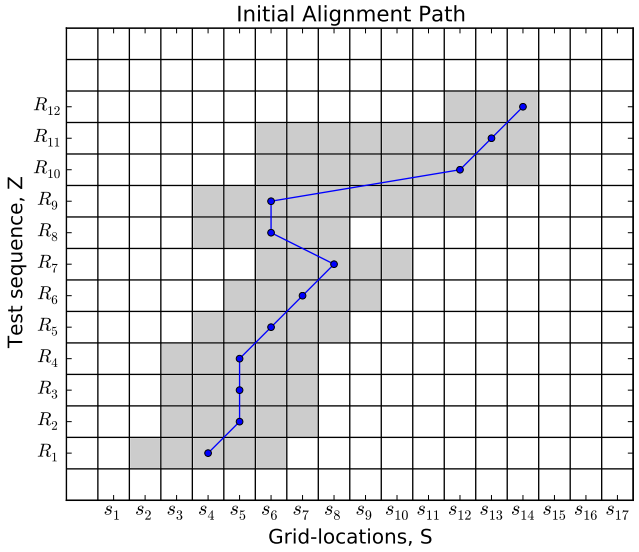


Figure 6.5. Variable width band according to Equation 6.3.1, around the initial alignment path returned by the nearest-neighbor search.

that the mean localization error decreases with w, where it becomes within 68% (37m) and 43% (23m) of that of unrestricted Viterbi decoding at $w = 10$ and $w = 15$ respectively.

Figures 6.7 and 6.8 show the run times of the algorithm for varying input sequence lengths. For a constant number of grid-locations, the run times of banded as well as unrestricted Viterbi decoding vary linearly with input sequence length. However, banded Viterbi decoding has a gradient that is less than that of the unrestricted version, and the difference of the gradients increases with the number of grid-location in the area. For areas with approximately 150 grid-locations (Figure 6.7), banded Viterbi with $w = 15$ and $w = 10$ perform about 3 and 4 times faster than unrestricted Viterbi decoding respectively. As shown in Figure 6.8, for areas with approximately 500 grid-locations, the speed up is about 5 and 7 fold, for $w = 15$ and $w = 10$ respectively.

Figures 6.9 and 6.10 show the results of travel-time estimation for a given road segment using banded Viterbi decoding. Figure 6.9 (a) indicate that for $w = 10, 15$ or 20, the percentage error in estimated travel-times with respect to actual ones remains close to that of the unrestricted version of the Viterbi algorithm. However, as shown in Figure 6.9 (b), the Pearson correlation between estimated and actual travel-times increase from 0.78 to 0.92 as w increases from 10 to unrestricted width. Figure 6.10 shows, as expected, the increase in run-times as w increases.

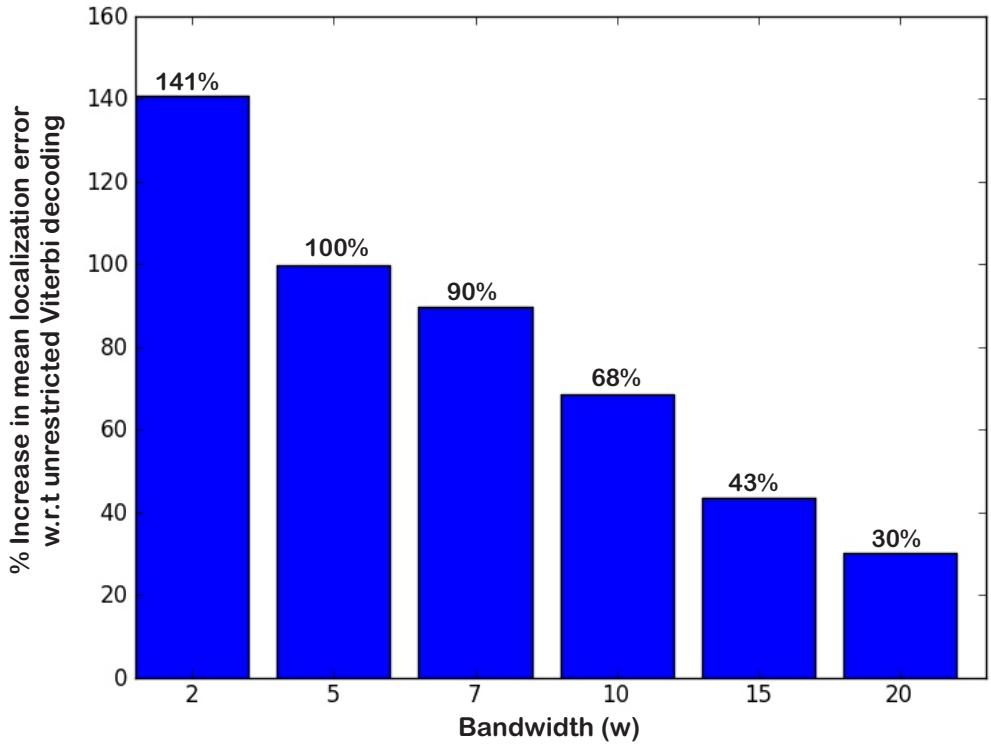


Figure 6.6. Percentage increase in mean localization error with respect to unrestricted Viterbi decoding, for different values of w .

6.5. Conclusions

In this Chapter, we presented a technique to speed up Viterbi decoding for tracking applications. Our technique is based on a fast nearest neighbor search to find an initial sequence of grid-locations for the path of the vehicle and then perform Viterbi decoding restricted to grid-locations close to this initial sequence. Our experiments revealed that moderate improvements in speed can be achieved with about a 50% increase in localization error with respect to unrestricted Viterbi decoding. Furthermore, the gain in speed is more prominent for areas with a larger number of grid-locations.

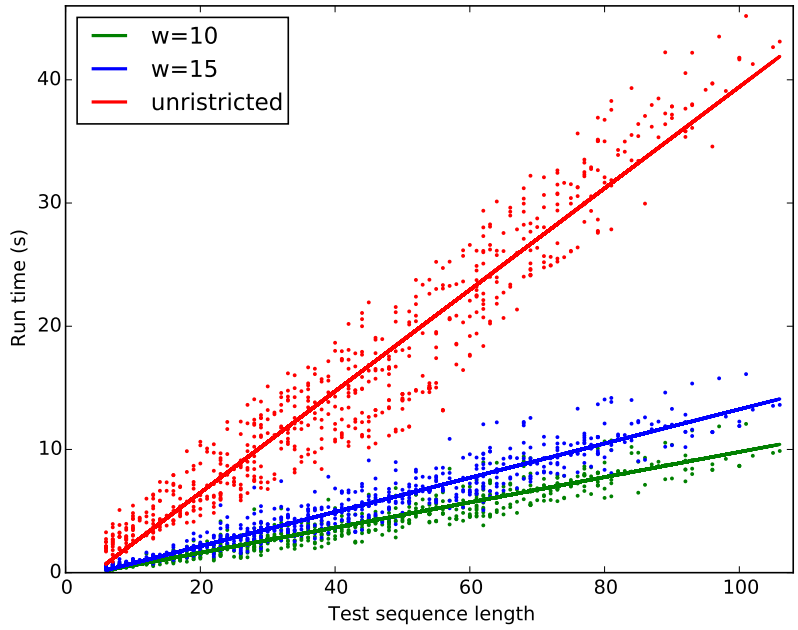


Figure 6.7. Run time variation with different test sequence lengths, for roads with approximately 150 grid-locations.

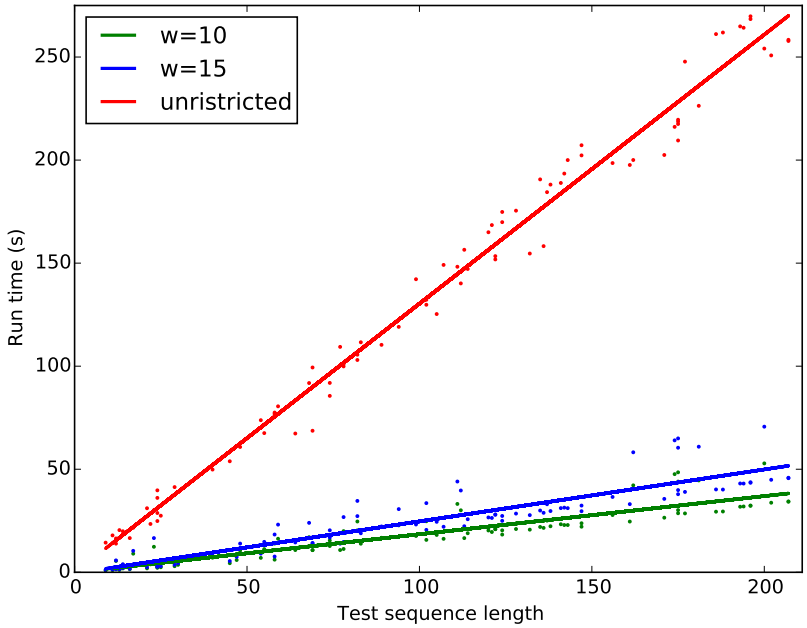
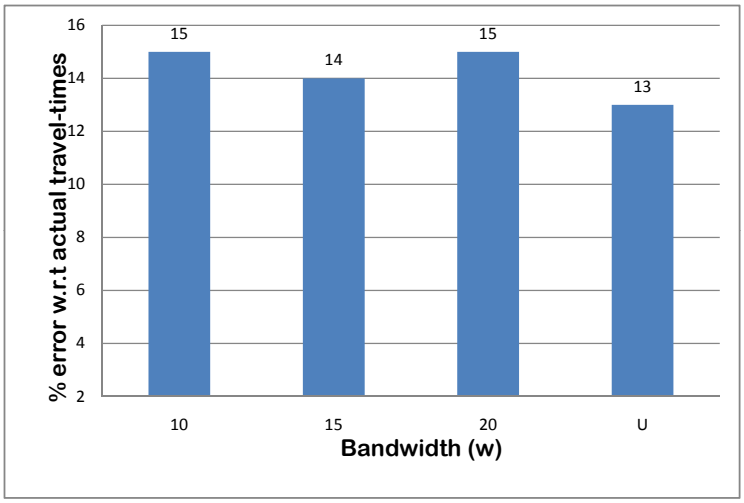
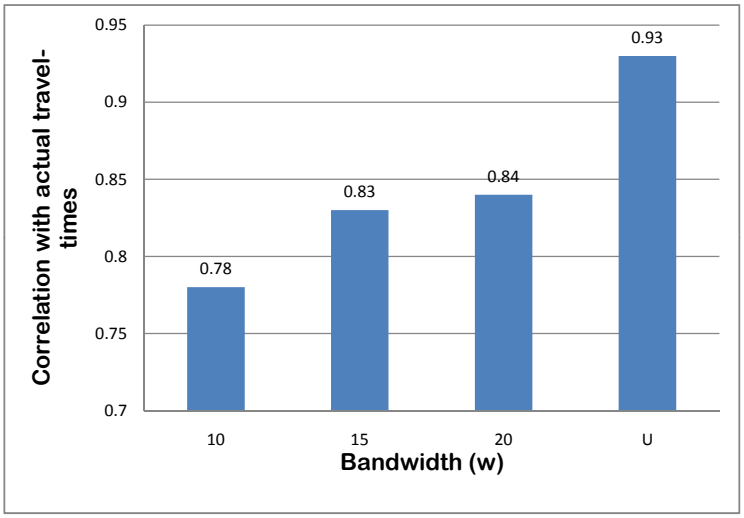


Figure 6.8. Run time variation with different test sequence lengths, for roads with approximately 500 grid-locations.



(a) Percentage error in estimated travel-times with respect to actual travel-times, for different bandwidths (w). Note: 'U' stands for unrestricted Viterbi decoding



(b) Pearson correlation coefficient between estimated and actual travel-times, for different bandwidths (w).

Figure 6.9. Travel-time estimation with banded Viterbi decoding for different bandwidths (w).

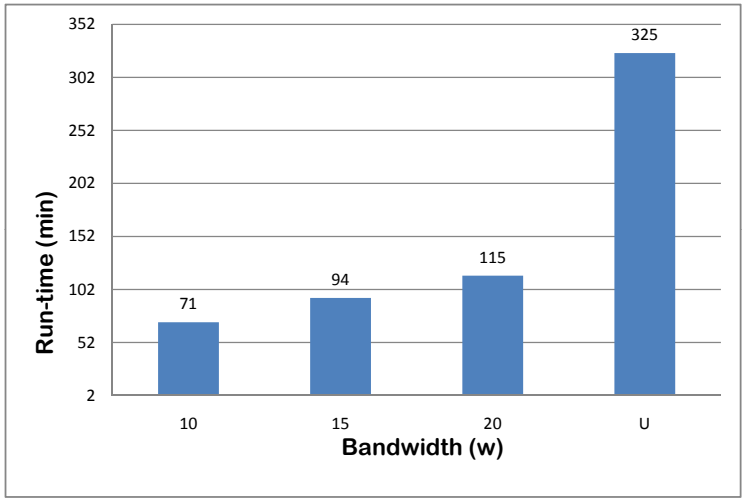


Figure 6.10. Running times of travel-time estimation with banded Viterbi decoding for different bandwidths (w).

7. CONCLUSIONS

In this thesis, we presented algorithms for tracking vehicles from signal strength measurements of mobile phones carried in them. In Chapter 3, we presented an algorithm that uses Dynamic Time Warping (DTW) to align the sequence of signal strength fingerprints from a mobile phone to a previously collected reference fingerprint sequence, and estimate the sequence of locations for the phone as those of the aligned reference fingerprints. Unlike other algorithms that are based on sequence alignment, our proposed algorithm is able to handle vehicles that change roads as it uses local sequence alignment as opposed to global alignment.

In Chapters 4 and 5, we presented algorithms based on a Dynamic Bayesian Network (DBN) that addressed two problems associated with the current state-of-the-art algorithms. First, we proposed a complete observational model for GSM received signal strength fingerprints that take into account the variation of the cell towers present in a fingerprint in addition to the variation of their signal strength levels. Then, we presented an unsupervised learning strategy to estimate the parameters of the said DBN that provides better estimates for the parameters than predicting them with a simple propagation model. Furthermore, our strategy is a reasonable alternative to the tedious and expensive parameter estimation based on supervised learning that requires signal strength measurements collected by driving on the roads.

We evaluated the performance our proposed algorithms and comparison algorithms using signal strength data that we collected by driving on several roads as well as real subscriber call trace data provided to us by a network service provider. Data collected by us was used to evaluate the localization/tracking accuracy of the different algorithms, while subscriber call trace data was used to evaluate the effectiveness of the algorithms in estimating average travel-times for a road segment.

Our results indicate that for the signal strength data that we collected, our local DTW-based algorithm outperforms the other *sequence alignment-based* algorithms and the simple nearest-neighbor-based method in terms of the localization error. For suburban roads, the DTW-based algorithm performs almost as good as the DBN-based algorithm with the traditional observation model, where as for urban roads, the said DBN-based algorithm has a lower localization error than

the DTW-based algorithm. However, it must be noted that DTW-based algorithms are trained using data from only a single drive on the relevant road where as DBN-based algorithms in Chapter 4 are trained using data from multiple drives. Our proposed DBN-based algorithm with the complete observation model has the lowest localization error for both urban as well as suburban roads.

Furthermore, we demonstrated that is possible to use Viterbi learning on our DBN-based algorithm with the complete observation model, to estimate mean received signal strength levels at predefined grid-locations with the aid of subscriber call traces. Our experimental results show that it achieves better localization accuracy than what is possible with mean received signal strength levels predicted using a simple propagation model.

The results of the experiment with the real subscriber call trace dataset indicate that using our proposed algorithms, it is possible to estimate travel-times for a road segment, with reasonable accuracy, from signal strength data of phone calls. DBN-based algorithm with the complete observation model has the lowest percentage error with respect to actual travel-times and has the highest correlation with actual travel-times. DBN-based algorithm with the traditional observation model shows better correlation with actual travel-times than the local DTW-based algorithm, however it has a much greater percentage error than the DTW-based algorithm. Further, as with signal strength data that we collected, the local DTW-based algorithm produces better travel-time estimates than the other alignment-based methods and the simple nearest-neighbor-based method.

In Chapter 6, we presented a technique to speed up Viterbi decoding for tracking applications, and showed that it achieves moderate improvements in speed while maintaining reasonable accuracy compared to complete, unrestricted Viterbi decoding.

In summary, our experiments show that our proposed DBN-based algorithm with the complete observation model performs better than all other comparison algorithms. Furthermore, the local DTW algorithm, given that it achieves the described results despite being trained using less data, can be more useful than the DBN-based algorithm with the traditional observation model, for applications where data from only a single drive is available.

Overall, we conclude that our proposed algorithms can be used as an alternative means of personal navigation if GPS is unavailable or undesirable. Our DBN-based algorithm with the complete observation model($DBN\_Cmp\_Obs\_MRT$) demonstrated the least localization error of

54m, averaged over all roads used in the experiments. Furthermore, our proposed local dynamic time warping (*N_LDTW*) algorithm and the DBN-based algorithm with the complete observation model(*DBN_Cmp_Obs_MRT*) can be used to provide travel-time estimates with reasonable accuracy. *N_LDTW* and *DBN_Cmp_Obs_MRT* achieved percentage errors of 14% and 13% respectively, with respect to actual travel-times. However, our evaluation was limited by the rather short duration for which call trace data was made available to us. We recommend further experiments with a larger dataset covering several days in order to draw stronger conclusions.

REFERENCES

[1] A. E. Stefan Steiniger, Moritz Neun, "Foundations of location based services," in *Lecture Notes on LBS*, University of Zurich.

[2] *ICT Facts and Figures*. International Telecommunication Union, Geneva, Switzerland, 2013.

[3] ETSI, *Reccommendation GSM 05.08: Radio Sub-system Link Control*. European Telecommunications Standards Institute, 1995.

[4] C. Chitraranjan, A. Denton, and A. Perera, "Tracking vehicle trajectories by local dynamic time warping of mobile phone signal strengths and its potential in travel-time estimation.," in *Proceedings of The Fourth IEEE International Workshop on the Impact of Human Mobility in Pervasive Systems and Applications (PerMoby 2015)*, pp. 445–450, IEEE, 2015.

[5] M. Olama, S. Djouadi, I. Papageorgiou, and C. Charalambous, "Position and velocity tracking in mobile networks using particle and kalman filtering with comparison," *IEEE Transactions on Vehicular Technology, Volume 57*, vol. 57, pp. 1001–1010, 2008.

[6] A. Thiagarajan, L. Ravindranath, H. Balakrishnan, S. Madden, and L. Girod, "Accurate, low-energy trajectory mapping for mobile devices," in *Proceedings of the 8th USENIX Conference on Networked Systems Design and Implementation*, NSDI'11, (Berkeley, CA, USA), pp. 267–280, USENIX Association, 2011.

[7] M. Ibrahim and M. Youssef, "Cellsense: A probabilistic RSSI-based GSM positioning system," *CoRR*, vol. abs/1004.3178, 2010.

[8] S. C. Ergen, H. S. Tetikol, M. Kontik, R. Sevlian, R. Rajagopal, and P. Varaiya, "Rssi-fingerprinting-based mobile phone localization with route constraints," *IEEE Transactions on Vehicular Technology*, pp. 423–428, 2014.

[9] Ibrahim, Mohamed, Youssef, and Moustafa, "A hidden markov model for localization using low-end gsm cell phones.," in *International Conference on Communications (ICC)*, pp. 1–5, IEEE, 2011.

[10] G. Leduc, "Road traffic data: Collection methods and applications," *Joint Research Centre,Office for Official Publications of the European Communities*, 2008.

[11] F. Calabrese, M. Colonna, P. Lovisolo, D. Parata, and C. Ratti, "Real-time urban monitoring using cell phones: A case study in rome," *IEEE Transactions on Intelligent Transportation Systems*, vol. 12, pp. 141–151, 2011.

[12] G. Chandrasekaran, T. Vu, A. Varshavsky, M. Gruteser, R. P. Martin, J. Yang, and Y. Chen, "Vehicular speed estimation using received signal strength from mobile phones," in *UbiComp*, pp. 237–240, 2010.

[13] M. B. Kjærgaard, "Location-based services on mobile phones: Minimizing power consumption," *IEEE Pervasive Computing*, vol. 11, no. 1, pp. 67–73, 2012.

[14] H. Laitinen, "Cellular location technology," *CELLO Consortium Report available at http://www.telecom.ntua.gr/cello/documents/CELLO-WP2-VTT- DO3-007-int.pdf*, 2001.

[15] H. Laitinen, J. Lahteenmaki, and T. Nordstrom, "Database correlation method for gsm location," in *Vehicular Technology Conference*, vol. 4, IEEE, 2001.

[16] K. K. Mohamed Khalaf-Allah, "Mobile location in gsm networks using database correlation with bayesian estimation," in *11th IEEE Symposium on Computers and Communications (ISCC'06)*, (Los Alamitos, CA, USA), pp. 289–293, IEEE Computer Society, 2006.

[17] T. Nypan, "Mobile terminal positioning based on database comparison and filtering," *Dissertation at the Norwegian University of Science and Technology*, 2004.

[18] B. Ferris, D. Haehnel, and D. Fox, "Gaussian processes for signal strength-based location estimation," in *Proceedings of Robotics: Science and Systems*, (Philadelphia, USA), August 2006.

[19] G. Chandrasekaran, T. Vu, A. Varshavsky, M. Gruteser, R. P. Martin, J. Yang, and Y. Chen, "Tracking vehicular speed variations by warping mobile phone signal strengths," in *PerCom*, pp. 213–221, 2011.

[20] J. L. Ygnace, "Travel time/speed estimates on the french rhone corridor network using cellular phones as probes," *Final report of the SERTI V program, INRETS, Lyon,France*, 2001.

[21] J. Steenbruggen, M. T. Borzacchiello, P. Nijkamp, and H. Scholten, "Mobile phone data from GSM networks for traffic parameter and urban spatial pattern assessment: a review of applications and opportunities," *GeoJournal*, vol. 78, pp. 223–243, 2013.

[22] Y. Yim, "The state of cellular probes," *California PATH Research Report*, 2003.

[23] B. L. Smith, M. L. Pack, D. J. Lovell, and M. W. Sermons, "Transportation management applications of anonymous mobile call sampling," *In TRB 80th annual meeting compendium of papers CD-ROM*, 2001.

[24] M. D. Fontaine, A. P. Yakkala, and B. L. Smith, "Probe sampling strategies for traffic monitoring systems based on wireless location technology," *Final contract report*, 2007.

[25] S. Maerivoet and S. Logghe, "Validation of travel times based on cellular floating vehicle data," in *Proceedings from 6th European congress and exhibition on intelligent transport systems and services*, 2007.

[26] H. X. Liu, A. Danczyk, and R. B. R. Starr, "Evaluation of cell phone traffic data in minnesota," *TRB 87th annual meeting compendium of papers CD-ROM.*, 2008.

[27] T. Giorgino, "Computing and visualizing dynamic time warping alignments in r: The dtw package," *Journal of Statistical Software*, vol. 31, 2009.

[28] Y. Sakurai, C. Faloutsos, and M. Yamamuro, "Stream monitoring under the time warping distance," in *Proceedings of the IEEE 23rd International Conference on Data Engineering*, (Los Alamitos, CA, USA), pp. 1046–1055, IEEE Computer Society, 2007.

[29] L. Rabiner, A. Rosenberg, and S. Levinson, "Considerations in dynamic time warping algorithms for discrete word recognition," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 26, pp. 575–582, 1978.

[30] L. Rabiner and B.-H. Juang, *Fundamentals of Speech Recognition.* 1 ed., 1993.

[31] T. F. Smith and M. S. Waterman, "Identification of common molecular subsequences," *Journal of molecular biology*, vol. 147, pp. 195–197, 1981.

[32] S. Henikoff and J. Henikoff, "Amino acid substitution matrices from protein blocks," *Proc Natl Acad Sci*, vol. 89, pp. 10915–10919, 1992.

[33] M. O. Dayhoff and R. M. Schwartz, "Chapter 22: A model of evolutionary change in proteins," in *Atlas of Protein Sequence and Structure*, 1978.

[34] D. Gundlegard and J. Karlsson, "Handover location accuracy for travel time estimation in gsm and umts," *ntelligent Transport Systems*, vol. 3, pp. 87–94, 2009.

[35] M. Y. Chen, T. Sohn, D. Chmelev, D. Haehnel, J. Hightower, J. Hughes, A. LaMarca, F. Potter, I. Smith, and A. Varshavsky, "Practical metropolitan-scale positioning for gsm phones," in *Ubiquitous Computing*, pp. 225–242, Springer-Verlag, 2006.

[36] Siemens, "Method and algorithm for the gsm cell reconfirmation," (Hannover, Germany), TSG-RAN Working Group 1 meeting No.7, 1999.

[37] Siemens, "Draft response to wg2 ls on gsm measurement abilities for the ue," (Dresden,, Germany), TSG-RAN Working Group 1 meeting No.9, 1999.

[38] M. C. Snare, "Dynamics model for predicting maximum and typical acceleration rates of passenger vehicles," *Thesis: Virginia Polytechnic Institute and State University*, 2002.

[39] A. Mehar, S. Chandra, and S. Velmurugan, "Speed and acceleration characteristics of different types of vehicles on multi-lane highways," *European Transport*, 2013.

[40] F. Dellaert, "The expectation maximization algorithm," *Technical Report number GIT-GVU-02-20, College of Computing, Georgia Institute of Technology*, 2002.

[41] S. R. Saunders and S. R. Simon, *Antennas and Propagation for Wireless Communication Systems*. New York, NY, USA: John Wiley & Sons, Inc., 1st ed., 1999.

[42] A. Goldsmith, *Wireless Communications*. New York, NY, USA: Cambridge University Press, 2005.

[43] H. L. Bertoni, *Radio Propagation for Modern Wireless Systems*. Pearson Education, 2000.

[44] ETSI, "TR 101.112 v3.2.0. universal mobile telecommunications system (UMTS); selection procedures for the choice of radio transmission technologies of the umts," *Technical Report*, 1998.

[45] J. H. Friedman, J. L. Bentley, and R. A. Finkel, "An algorithm for finding best matches in logarithmic expected time," *ACM Transactions on Mathematical Software*, 1977.

[46] A. Guttman, "R-trees: A dynamic index structure for spatial searching," in *International Conference on Management of Data*, pp. 47–57, ACM, 1984.

[47] J. McNames, "A fast nearest-neighbor algorithm based on a principal axis search tree," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 23, no. 9, pp. 964–976, 2001.