NORTHWESTERN UNIVERSITY

Spatial-temporal Data Mining for Traffic Speed Clustering and Prediction

A DISSERTATION

SUBMITTED TO THE GRADUATE SCHOOL

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS

for the degree

DOCTOR OF PHILOSOPHY

Field of Computer Science

By

Bing Zhang

EVANSTON, ILLINOIS

June 2017

© Copyright by Bing Zhang 2017

All Rights Reserved

ABSTRACT

Spatial-temporal Data Mining for Traffic Speed Clustering and Prediction

Bing Zhang

Spatial-temporal data mining, with data driven model and machine learning techniques, significantly benefit the traditional transportation research. This dissertation focus on three problems related to uncertain location data, lane-level traffic speed clustering and anomalous traffic speed prediction.

We take a first step towards combining the uncertain location data i.e., fusing the uncertainty of moving objects location obtained from both GPS devices and roadside sensors. We develop a formal model for capturing the whereabouts in time in this setting and propose the Fused Bead (FB) model, extending the bead model based solely on GPS locations. We also present algorithms for answering traditional spatio-temporal range queries, as well as a special variant pertaining to objects locations with respect to lanes on road segments.

We address the problem of efficient spatio-temporal clustering of speed data in road segments with multiple lanes. We postulate that the navigation/route plans typically reported by different providers as a single-value need not be accurate in multi-lane networks. Our methodology generates lane-aware distribution of speed from GPS data and agglomerates the

basic space and time units into larger clusters. In addition, we address the problem of incorporating uncertain location data in the generation of speed profiles for vehicles on roads with multiple lanes. Moving objects location data can be obtained from different/multiple sources e.g., GPS on-board the moving objects, roadside sensors, cameras. However, each source has inherent limitations that affect the precision from pure measurement-errors, to sparsity of their distribution. Incorporating such imprecisions is paramount in any query/analytics oriented system that deals with location data. The difficulties multiply when one needs to reason about localization with lane-awareness and attempts to use the location-in-time data to enable effective navigation systems.

We improve the accuracy of short-term traffic speed prediction with an novel prediction framework that is adaptive to anomalous events. A new demand feature is proposed with an anomalous events detection algorithm to collect features when anomalies occur. An artificial neural network based prediction model is introduced to incorporate demand features and traffic speed features. Our experiment demonstrate that the proposed prediction framework offer a more accuracy short-term traffic speed prediction.

Acknowledgements

First and foremost, I am grateful to my advisor, professor Goce Trajcevski, for his educations, supports, inspirations and encouragements. I am extremely thankful and indebted to him for sharing expertise, and sincere and valuable guidance. He is my role model. I will never forget those sleepless nights working with him.

I would also thank professor Aleksandar Kuzmanovic and Jennie Rogers for being my PhD committee members. Their knowledge and insights are very valuable and enrich my work.

I especially thank my mom and dad. My parents have sacrificed their lives for me and provided unconditional love and care. I would not have made it this far without them.

Last, but not least, I want to thank my wife, Shimeng. She selflessly supports my research and study. She is my soul-mate and the best listener. With her on my side, I can always find my peace of mind.

Table of Contents

List of Tables

List of Figures

CHAPTER 1

Introduction

Spatio-temporal data mining is an emerging research area dedicated to the development and application of novel computational techniques for the analysis of large spatial-temporal databases [**5, 83, 91**]. The main motivation for a more focused research in this particular subfield of data mining stems from the fact that extremely large amount of spatial-temporal data are being generated in various applications – e.g., GIS, CAD, robotics, computer vision, traffic management; and from plethora of sensing devices – e.g., GPS, inductive loop sensors, cameras, tracking sensors, etc. [**6, 69, 116, 122**].

This research work has focused on spatio-temporal data management/mining in the context of transportation related tasks. Given that commuting is among the largest-scale people's everyday activity, efficient transportation management has an essential and tremendously important role in our society. In the past 20 years, with the development of sensing and communication technologies, as well as the improvement of computational resources, intelligent monitoring and management of transportation system is a lot more feasible than two decades ago [**104**]. This, in turn, both enables and demands the development of advanced transportation-related data mining algorithmic solution, to help extract hidden information from the vast datasets, which reflect people's mobility and behavior in the city [**24**].

However, there are many challenges and constrains in this area along the spectrum ranging from hardware limitation to state-of-art data mining. On the one hand, the GPS uncertainties and GPS errors constrain our ability for accurate sensing and related applications [**6, 101, 124, 128**]. On the other hand, people are generating data in an unprecedented way, in terms of speed, quantity and heterogeneity, and we haven't fully utilized all available features for prediction and clustering. More specifically, this dissertation tackles the spatial-temporal data mining from three aspects:

(1) We propose a data model to fuse heterogeneous location data, targeting to reduce the GPS uncertainties.
(2) We propose a framework to mining speed clusters in multi-lane setting.
(3) We propose a novel data feature and prediction framework to improve the accuracy of traffic speed prediction.

In the rest of this Chapter, we provide an overview of the main challenges and contributions regarding the three aspects of the dissertation.

1.1. Fusing Uncertain Location Data

Many applications relying on some forms of Location Based Services (LBS) [**84**] depend on efficient techniques for storing, retrieving and querying data which describes the whereabouts-in-time of moving entities. Traditionally, such topics are studied in the field of Moving Objects Databases (MOD) [**38**], and the impacts of the effectiveness of those techniques are of an extreme importance in many applications of high societal relevance such as transportation and traffic management [**22, 32, 94, 19**], disaster remediation [**50**] and location-aware social networking [**9**]. Especially so since, due to the advances in networking and miniaturization of the various GPS-enabled devices, the volume of location-in-time data exceeds the order of Peta-Bytes per year just from smartphones [**69**].

Typically, the location of a given moving object at a particular time instant is obtained either by some GPS (Global Positioning System) based devices [**106, 87**], or by some type of a road-side sensor – e.g., lane level positioning [**48, 25**]. Such sensed location data may be further combined with data from different on-board sensing devices – e.g., U.S. Xpress gathers 900 to 970 data elements of various engine/component readings [**61**].

Due to the inherent imprecision of the sensing devices – be it on-board GPS or other – typically there is a degree of *uncertainty* associated with the measurements of the location of a given moving object at a particular time instant. The problem of capturing the impact of the location uncertainty into the spatio-temporal data models [**55**] as well adding proper syntactic constructs to capture its impact on the MOD queries and the respective processing algorithms has been recognized and tackled by several earlier works [**23, 35, 38, 55, 78, 102, 101**].

At the heart of the motivation for this work is the observation that the state of the art – to the best of our knowledge – has not provided any models and algorithmic approaches that would *combine (i.e., fuse) uncertain location data from two different sources.* Specifically, we take a first step towards fusing the uncertain location data from on-board GPS devices and road-side sensors. We demonstrate that properly considering the joint impact of the uncertainties from both sources can eliminate portion of the moving objects (trajectories) from the answer-set. In other words, what may have been considered an answer under the single (e.g., GPS) source, may become a false-positive after fusing the two location uncertainties. As an example, consider the following query:

Q1: *Retrieve all the vehicles which have crossed the lane in road segment RS1 when driving less than 50km/h and carrying less than 80% of the maximum load.*

Clearly, given the imprecision of the location measurements, **Q1** needs to be re-phrased so that it incorporates uncertainty:

Q1^u: *Retrieve all the vehicles which have had* $> \Theta$ *(*$0 < \Theta \leq 1$*) probability of crossing the lane in road segment RS1 when driving less than 50km/h and carrying less than 80% of the maximum load.*

The answers to such, so called, lane-crossing queries play an important role in applications related to efficient traffic management [**43, 18, 87**] for the purpose of regulating the regime of traffic lights [**48, 65**].

The main contribution of this work can be summarized as follows:

- We propose a novel model of spatio-temporal uncertainty for moving objects, which combines the location data obtained by GPS devices on-board moving objects and the location data obtained from road-side sensors. We also report our preliminary experimental observations, demonstrating the reduction of false positives from the answers to certain spatio-temporal queries.
- We discuss the semantic implications of the model, in terms of the basic *where_at* and *when_at* location-in-time (whereabouts) queries, and we present algorithms for processing *lane-crossing* queries (exemplified by **Q1**^{u} above) and basic *range* queries.
- We present experimental observations which quantify the benefits of fusing the two uncertainties for lane-crossing and range queries in terms of the percentage of trajectories which are pruned from the answer-sets when compared to using the traditional bead-model of uncertainty for GPS-based location data.

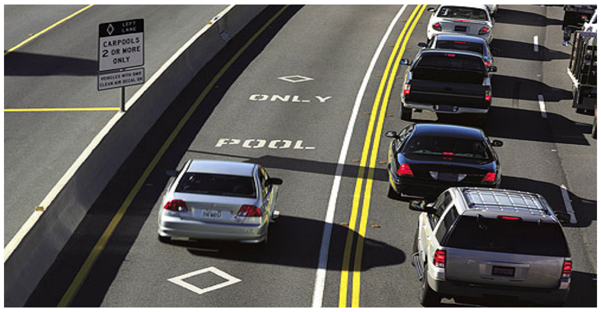
In Section 2.1 we recollect some backgrounds in terms of modeling spatio-temporal uncertainty, and introduce the basic terminology used in the rest of the work. Section 2.2 presents the details of the new uncertainty model, along with the semantics of the basic whereabouts queries along with lane-crossing and range queries. Section 2.5 describes our experimental observations. In Section 2.6 we compare our work with related literature, and we summarize and outline directions for future work in Section 2.7.

1.2. Multi-lane Speed Cluster Mining

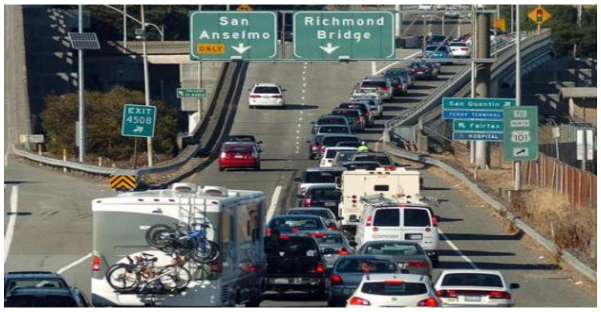
Lane level positioning and navigation have been one of the challenging tasks that have spurred a significant amount of recent research since accurate navigation is at the very core of the autonomous driving [**17, 86**]. Models for lane-level high-definition maps have been proposed in different applications' settings [**1**], but lane-aware traffic inference and route planning are still investigated, mostly from two perspectives: (a) Assuming very accurate positioning data gathered through Differential GPS (DGPS) or laser scanners [**82**]. The high cost of sensors prohibits this method from being widely deployed for production cars; (b) Fusing heterogeneous data sources, i.e., combining GPS data with camera and using computer vision for lane recognition [**17, 51**]. The bottleneck of this approach is the speed of image processing, which constrains the use case in a real-time manner. Routing and navigation in modern traffic systems have been investigated since the 1980s [**57**], with techniques coming from both databases [**32, 113**] and transportation communities [**80**]. Typically, the algorithmic solutions rely on certain estimated values of the traffic flow – e.g., average speed – along the segments of the underlying road-networks, which vary dynamically [**113**] within

(a) Uniform traffic speed with low congestion

(b) Uniform traffic speed with high congestion



(c) Nonuniform traffic speed for carpool lane



(d) Nonuniform traffic speed for highway exit

Figure 1.1. Traffic speeds vary among different lanes in different scenarios

a certain period (e.g., a day), depending on factors such as: time of day, capacity (lanes), road surface, etc.

From traditional vehicle routing problem [**57**] up to recent Eco-routing works [**63**], the methodologies (data properties, algorithms, etc.) proposed in various contexts share the assumption that on any road segment, at a certain time-period, vehicles have only one kind of a speed/motion. However, due to the multiple lanes, vehicles on the same road segment and at the same time instant/interval, may have different speeds. This, in turn, implies that using the average speed as a descriptor may not be good enough for many routing-based applications. Figure 1.1 illustrates four different real life scenarios on highways. When there are few cars on the road or the highway is fully congested, traffic speeds are relatively uniform among the lanes (cf. Figure 1.1a and 1.1b). However, Figure 1.1c, shows how a high-occupancy lane (also known as carpool lane; restricted traffic lane reserved during rush hour for the exclusive use of vehicles with one or more passengers) usually has higher speeds than the other lanes. Similarly, Figure 1.1d, shows a highway exit 450B on U.S. route 101 in California, near Richmond-San Rafael Bridge. The cars back up at the rightmost lane towards the bridge, while the left lane on northbound U.S. route 101 has very low densities. Thus, averaging the observations from particular (groups of) vehicles, could yield an inaccurate picture about the traffic distribution – and, yet, most of the popular traffic speed estimation methods are based on averaging the samples from vehicles over a period of time or area — e.g., *Time Mean Speed* and *Space Mean Speed* [**31**].

At the heart of the motivation for this work is the observation that – to the best of our knowledge – the state of the art approaches have not provided solutions that would couple the multi-lane information with location uncertainties, when designing traffic speed profiles (we note that this is also the case for the existing works on map-matching GPS points from moving objects [**13, 114**]). Consider the following query:

Q1: *What is the distribution of the traffic speed on the route 101 between San Francisco and Richmond-San Rafael Bridge, between 8:00AM and 10:00AM?*

Traditional methods [**75, 88, 113**] would answer **Q1** with a single average value, possibly varying it throughout different time intervals between 8:00AM to 10:00AM (e.g., the average speed is updated every 30min.); and along different distances from Marine city on the route 101. However, this will yield incorrect values because the averaged speed will be applied within certain distances before/after exit 450B (cf. Figure 1.1d), yielding incorrect time-estimates for trip planning. The main contribution of this work can be summarized as follows:

- We achieve a compact description of speed variations in multi-line road networks that can be used for more accurate trips planning.
- We propose an agglomerative speed clustering algorithm to represent the distribution in a more compact manner.
- We propose a novel probabilistic model to represent location uncertainties and apply it to spatial temporal data mining.
- We propose a novel distance function and an improved speed cluster mining algorithm for multi-lane road networks.
- We present experimental observations conducted on the Rome Ring Road to demonstrate the benefits of the proposed approaches.

The rest of this chapter is structured as follows. In Section 3.1 we review some preliminary background and introduce the basic formalisms used in the subsequent parts. Section 3.2 will present an agglomeration based speed cluster mining method. In Section 3.3 and 3.4, we propose a probabilistic speed profiling approach in conjunction a density based clustering algorithm. The quantitative results of experiments and interpretation are presented in Section 3.5. Section 3.6 positions the work in the context of related literature and Section 3.7 summarizes and outlines directions for future work.

1.3. Anomalous Traffic Prediction

Traffic prediction is one of the most important applications in the Intelligent Transportation System (ITS). It has been proved as the foundation for other services like routing, navigation, traffic control and signal optimization. The purpose of traffic prediction is to predict the traffic speed of a road segment. This topic has been studied in the area of transportation research for more than three decades [**110**]. Many statistical and machine learning models have been proposed to make an accurate prediction [**127, 110**], including: spectral analysis [**72**], regression methods [**107**], time series model [**70**], kalman filtering methods [**37**], support vector machine [**117**], neural networks [**68**], fuzzy logic system [**96**] and other Hybrid models [**67**].

In the traditional study of traffic prediction, models rely on the historical traffic data obtained from *traffic sensing* system as the single data source. From the point of view of the classical traffic prediction research, traffic flow is periodic, over a 24-h period for weekdays, or over a week for both weekdays and weekends [**127**]. Traffic flow is stochastic [**56**], due to exogenous factors such as traffic incident, weather and road conditions. Traffic flow is spatial-temporal correlated [**70**], because of the nature of traffic flow and the connectivity of road segments. Most existing prediction models try to capture certain patterns of the traffic flow in spatial-temporal dimensions.

In the recent few years, with the development of portable devices, more and more different types of mobility data become available. This enables researches of spatial-temporal data mining that tackle issues related to people's urban activities [**130**] and their applications in transportation researches. Some novel models are proposed to indirectly infer and predict traffic condition from heterogeneous data sources such as CO_2 concentration in large buildings [**131**] or taxi trajectories [**11**]. However, Many constraints still exist with these approaches. For example, the use of CO_2 concentration data are limited to building dense area with the deployment of CO_2 sensors.

Traffic flow is the result of people's mobility. Although most people's everyday life is periodic with predicted patterns, expected or unexpected events always happen.

Example: Since 1904, Times Square has been the place for people in the New York City to conduct celebration to greet the new year. On the New Year's Eve 2016, an estimated one millions people ushered in the new year in Times Square [**2**] as shown in Figure 1.2. The celebration event started from 3:00 p.m. and ended on 12:15 a.m. Figure 1.3 is a visualization for traffic speed of 11th Ave. on Jan 1st, 2016. There is a sudden drop of traffic speed after midnight, which reflects the traffic congestion incurred by one million people's leaving from Times Square.

Figure 1.2. One million people celebrate in Times Square

Urban anomalies are defined as spatial-temporal entries with underlining abnormal mobility distributions [**129**]. They often happen with special events that people gather and

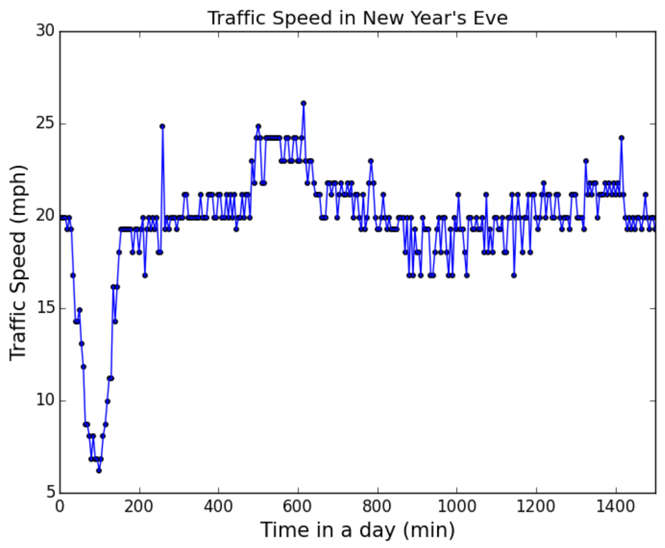


Figure 1.3. Traffic speed of 11th Ave. on Jan 1st, 2016

disperse in a short period of time. When we study cities and people's mobility with Spatial-Temporal Graph (STG) model, urban anomalies result in mobility black holes and volcanoes [**44**], which act as a sink or generation for traffic flow. Because of the infrequency and the randomness of anomalies, it is very hard for traditional models to capture the anomalous change of traffic solely based on historical traffic data. These anomalous cases, which would be classified as outliers in traditional data analytics, are very challenging for real-time traffic prediction model as well. The additional generated traffic flows would affect the prediction accuracy, or prediction power (in terms of prediction time length).

People move everyday. Their mobility essentials are fulfilled by various mobility modes (e.g. walking, bike, taxi and Uber) at different time. The traffics in road networks, as a complex system, are the aggregated result of fulfillment of mobility demands. Therefore, being able to access and analyze the data gathered from transportation supplier enable us to predict traffic from the origin.

Traffic prediction under anomalies directly benefit applications like routing and navigation. It enables ITS to manage traffic when anomalies occur. In this dissertation, we propose a prediction framework that incorporate historical traffic data with people's mobility demand to improve the prediction accuracy when urban anomalies take place.

CHAPTER 2

Fusing Uncertain Location Data From Heterogeneous Sources

In this chapter, we discuss in details our contributions related to fusing uncertain location data for moving objects trajectories from heterogeneous sources.

2.1. Preliminaries

We now present an overview of some of the techniques for obtaining location data, which we assume and rely upon in this work. Specifically, we discuss the main features of road-side sensors and GPS devices. Subsequently, we proceed with introducing the basic terminology and notation used in the rest of the chapter.

2.1.1. Road-side Sensors

Starting in the 1920s, when the traffic signals were still manually controlled, several generations of sensor types have been developed and deployed along road segments in various states – all for the purpose of more efficient traffic management. The types of such sensors vary from the older pressure-sensitive ones introduced in 1931, to more modern laser-based sensors sensors [**105**] and quite a few different types have been commercialized and used in day-to-day practical settings. For example, the AMR sensor [**43**] developed by Honeywell is a type of magnetic sensor with low cost. The WiEye [**27**] is a passive infrared sensor that can be installed on top of motes to sense road condition. The variation of sensing technologies may affect the manner of how a motion is modeled, in order to capitalize on the capabilities of a particular type of sensor. In this chapter, the data model for roadside sensor that we adopt is based on TruSense T-Series, manufactured by Laser Technology Inc. [**58**] – a kind of active infrared sensor with a very accuracy as well as a high sampling rate.

Table 2.1. Comparison among different types of sensors

Sensor technology	Count	Presence	Speed	Output Data	Classification	Multiple Lane detection
`Inductive loop`	✓	✓	✓	✓	✓	✗
`Magnetometer (two axis fluxgate)`	✓	✓	✓	✓	✗	✗
`Magnetic induction coil`	✓	✓	✓	✓	✗	✗
`Microwave radar`	✓	✓	✓	✓	✓	✓
`Active infrared`	✓	✓	✓	✓	✓	✓
`Passive infrared`	✓	✓	✓	✓	✗	✗
`Ultrasonic`	✓	✓	✗	✓	✗	✗
`Acoustic array`	✓	✓	✓	✓	✗	✓
`Video image processor`	✓	✓	✓	✓	✓	✓

Table 2.1 provides a summary of features of several different types of roadside sensors [**105**]. As can be seen, all of the popular and commercially available types can detect the presence and speed of vehicles, as well as provide a count value for the number of vehicles that have been detected in their sensing range. However, very few types provide more detailed sensing capabilities, such as classification and multiple lanes detection. We note that, unlike the GPS-based data, the location-in-time information obtained from the roadside sensors has not been exploited extensively in MOD context.

2.1.2. GPS-based Spatio-Temporal Uncertainty

As commonly done in the literature [**38**], in this paper the trajectory is defined as:

Definition 1. A *trajectory* Tr_i of a moving object with a unique identifier (oID) is a sequence of triplets $Tr_{oID} = [(L_1, t_1), (L_2, t_2), v_{max1}] \ldots, [(L_{n-1}, t_{n-1}), (L_n, t_n), v_{max\_(n-1)}]$ where each $L_i = (x_i, y_i)$ is a point in 2D space in a corresponding reference coordinate system, and t_i denotes the time instant at which the object was at location L_i. When it comes to the time-values, $i < j$ implies $t_i < t_j$, and v_{max_i} denotes the maximum speed of the object between samples at t_i and t_{i+1}

Given the possibility of errors in the discrete location samples (e.g., due to the imprecision of the GPS devices), plus the fact that one attempts to model a continuous phenomenon (motion, in this case) with a discrete set – uncertainty becomes an inevitable component of the model. The problem of incorporating the location uncertainty into the syntax and the respective algorithms for calculating the queries answers has been treated from a couple perspective in the MOD literature [**38, 102**].

One approach for modeling spatio-temporal uncertainty of moving objects is the, so called, sheared cylinder model [**102**]. The main assumption is that at any time instant t_i, the object's location is inside a given disk with a fixed radius, centered at the expected location at t_i. For time values different from sampling ones, the expected location is obtained via linear interpolation [**102**]. This model assumes a fully-known trajectory is geared towards processing continuous queries over past/historic trajectories.

The implications of the fact that the object's motion was bound by some v_{max} in-between two consecutive location updates was analyzed in [**77**]. Based on the definition as a geometric set of 2D points, it was demonstrated that the possible whereabouts are bound by an ellipse, with foci at the respective point-locations of the consecutive samples. Subsequently, [**45**] presented a spatio-temporal version of the model, naming the volume in-between two update points a *bead*, and the entire uncertain trajectory, a *necklace*. This model was actually introduced as a *space-time prism* in the geography literature [**39**]. However, the first work to present a formal analysis of the properties of the bead are [**55**]. An illustration is provided in

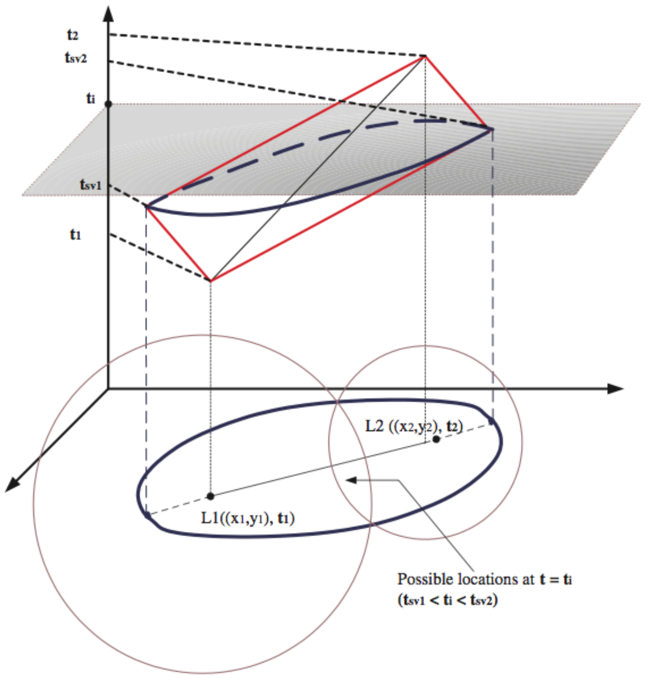


Figure 2.1. Bead and ellipse model

Figure 2.1. Letting $d = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$ denote the distance between the starting location (at t_1) and ending location (at t_2), the equation of the projected ellipse (cf. [**77**]) is:

$$\frac{(2x - x_1 - x_2)^2}{v_{max}^2(t_2 - t_1)^2} \frac{(2y - y_1 - y_2)^2}{v_{max}^2(t_2 - t_1)^2 - (x_2 - x_1)^2 - (y_2 - y_1)^2} = 1 \tag{2.1}$$

The corresponding bead (equivalently, space-time prism) is specified with the following constraints:

$$(2.2)\qquad \begin{cases} t_i \leq t \leq t_{i+1} \\ (x - x_i)^2 + (y - y_i)^2 \leq [(t - t_i)v^i_{max}]^2 \\ (x - x_{i+1})^2 + (y - y_{i+1})^2 \leq [(t_{i+1} - t)v^i_{max}]^2 \end{cases}$$

where v_{max} is the maximal speed that the object can take between t_i and t_{i+1}. We note that, what is commonly called *expected* speed in the case of crisp trajectories, now becomes *minimal* expected speed in-between the updates/samples. As shown in Figure 2.1, at any time instant t between two consecutive samples, the possible locations of the objects are bound by the lens – i.e., intersection of two circles centered at the respective foci and with respective radii $v_{max}(t - t_1)$ and $v_{max}(t_2 - t)$.

In similar spirit to [**100, 102**] we can define a *possible trajectory* to be any trajectory which has its starting point and its ending point coinciding with the foci, and is fully contained inside the given bead.

2.1.3. Trajectories and Road Networks

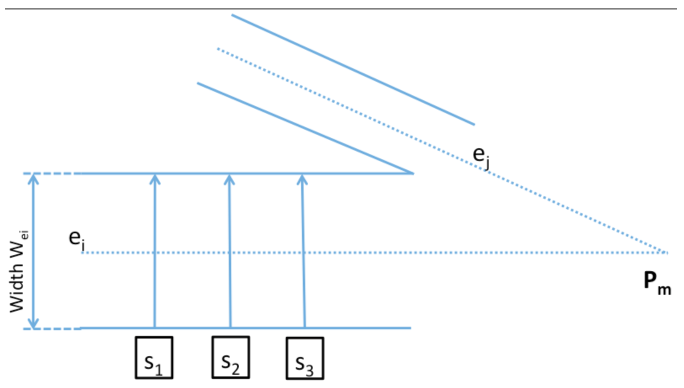


Figure 2.2. Road segments and sensors

If the objects are constrained to move along a road network, then on would expect that the corresponding space-time prisms would somehow be restricted as volumes. Specifically, if the segments of the road network are assumed to be edges in a graph, then the prisms become restricted to 2D planar figures (c.f. [**28**]).

In this work, we define a road network as an *augmented graph* $G = (P, E_{RS})$ where $P = \{p_1, p_2, \ldots, p_n\}$ denotes a set of points (commonly corresponding to intersections) and $E_{RS} = \{r_{S1}, ..., r_{Sk}\}$ is a collection of triplets of the form $r_{Si} = (e_i, w_{ei}, v_{ei})$ where:

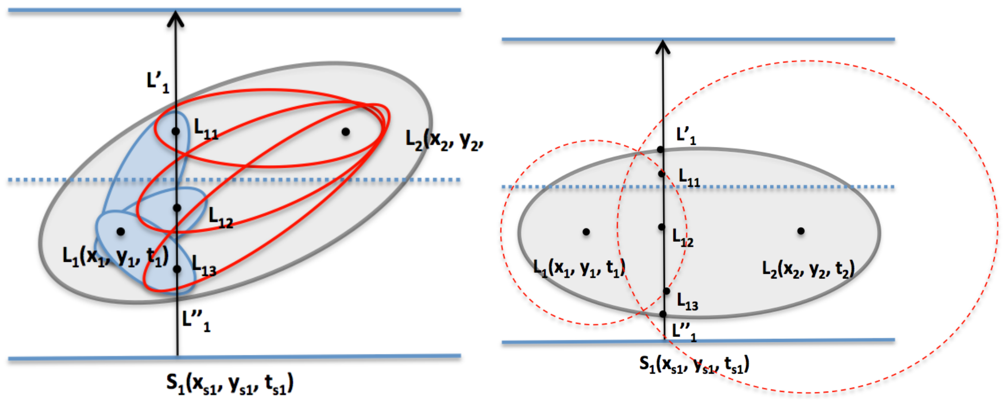
- $e_i = (p_{i1}, p_{i2})$ ($\in P$ X P) is a "regular edge" (i.e., a link between two connected vertices)
- w_{ei} denotes the width of the road segment associated with the edge e_i.
- v_{ei} denotes the maximum speed associated with r_{Si}.

We assume that the maximum speed in-between two consecutive location samples along a particular road segment corresponds to the speed-limit of that segment. Geometrically speaking, the collection of all the r_{Si}'s is the boundary of the Minkowski sum of each "regular edge" e_i and a disk with diameter w_{ei}.

We also assume the existence of a collection of sensors $S = \{s_1, s_2, ..., s_m\}$, where each sensor s_j is located at a point along the outer boundary of some road segment r_{Si}. Each s_j detects when (i.e., the time instant at which) a moving object crosses the line segment going through its location and perpendicular to e_i. The concepts are illustrated in Figure 2.2.

2.2. Modeling the Uncertainties Fusion

We now discuss the details of the new uncertainty model resulting from combining the GPS-based location data and the location data generated by road-side sensors.



(a) GPS + Roadside sensors (b) Determining boundaries

Figure 2.3. Fusing GPS and roadside sensors data

The main observation is that the road-side sensors provide additional constraints on the possible whereabouts in-between two consecutive GPS-based samples (and vice-versa). More specifically, recall that the "traditional" bead (i.e., space-time prism) was defined by the system of inequalities (2.2) (cf. Section 2). In addition to those inequalities, we now have the constraint that at a particular time instant t_{si}, the possible locations of a particular moving object detected by the roadside sensor are also known to be along a given line-segment determined by:

(1) the location of the corresponding road-side sensor, and

(2) the direction which is perpendicular to the (boundaries of the) road segment.

This can be formalized as:

$$
(2.3) \qquad \begin{cases} t_i \leqslant t \leqslant t_{i+1}, \\ (x - x_i)^2 + (y - y_i)^2 \leqslant (t - t_i)^2 v_{max}^2, \\ (x - x_{i+1})^2 + (y - y_{i+1})^2 \leqslant (t_{i+1} - t)^2 v_{max}^2, \\ y = m_i x + b_i, \text{when } t = t_{si} \\ t_i \leqslant t_{si} \leqslant t_{i+1}. \end{cases}
$$

An illustration of the system of constraints (3) is given in Figure 3.5: Specifically, as shown in Figure 2.3a, the original GPS-based locations L_1 and L_2 would yield a 2D projection which is an ellipse having them as foci (light-grey shaded shape in Figure 2.3a) – denote it El_1. Due to the road-side sensor, the possible locations of the moving object at t_{s1} can only be along the *portion* of line segment originating in (x_{s1}, y_{s1}), perpendicular to the boundaries of the road segment, and intersecting the corresponding lens ofEl_1 – i.e., along the portion of the line segment $\overline{L_1' L_1''}$. Clearly, that intersection has an uncountably many points, and we show 3 such points in Figure 2.3a: L_{11}, L_{12} and L_{13}. Each such point, in turn, can be used as a "generator" for two more space-time prisms: one originating in L_1, and the other terminating at L_2. The corresponding 2D projections (ellipses) are shown in Figure 2.3a for L_{11}, L_{12} and L_{13}. The most important implication is that when combining the original ellipse El_1 with the uncountably infinite collection of the ellipses with one of the foci along the line segment due to the road-side sensors, the additional constraint induces a significant amount of a "dead-space" in El_1. A more detailed illustration of the valid range for selecting the points that will generate the infinite collection of (pairs of) new beads is given in Figure 2.3b.

Recall that at any t_{s1} between the sampling times t_1 and t_2, the object can be located inside of the lens obtained as the intersection of the circles with radii $v_{max}(t_{s1}-t_1)$ and $v_{max}(t_2-t_{s1})$. Hence, although the ray emanating from the roadside sensor s_1 would intersect the "global boundary" (i.e., the ellipse which is the projection of the bead) at L_1' and L_1'', the only valid points to be considered as possible whereabouts are the ones along (and inside) the lens. As shown in Figure 2.3b, those are the points along the line segment bounded by L_{11} and L_{13}.

We note that there is a complementary context of having a single uncertainty source – i.e., in contrast to having GPS-based points only. Namely, if there were only the roadside sensors available, then in between two detections by consecutive sensors (say, s_1 and s_2 from Figure 2.2), the whereabouts of a given object is bounded by the infinite union $\cup(El_{si,sj})$ of uncountably many ellipses for which:

(1) The first focus is some point L_{s1} located on the line-segment originating at the location of s_1.

(2) The second focus is some point L_{s2} located on the line-segment originating at the location of s_2;

(3) The distance between L_{s1} and L_{s2} is smaller than $v_{max}(t_{s2} - t_{s1})$ (i.e., the object could travel the distance within the time-interval $[t_{s1}, t_{s2}]$ for the given speed limit).

Incorporating the GPS-based bead in this context would either amount to the case where it intersects one (or more) of the line segments originating at the respective sensors locations, or it has no intersection with any of them. In the latter case, we have a scenario in which GPS sampling frequency is higher than the sampling frequency obtained by the roadside sensors. For such settings, the possible whereabouts will be reduced to the intersection of the $\cup(El_{si,sj})$ and the bead obtained from the GPS-based samples. In the former case, the model is a generalization of the one corresponding to the scenario illustrated in Figure 3.5 – in the sense that it may be possible to have intersections of the GPS-based bead with > 1 sensor lines, as illustrated in Figure 2.4. In the rest of this chapter, we focus on detailed discussion of the scenarios in which a bead is intersected by a line segment emanating from a single roadside sensor.

We call the spatio-temporal structure induced by combining the two uncertainty sources – GPS and roadside sensors – a *Fused Bead* (FB), and it is a sixtuple FB ((x_i, y_i, t_i), $(x_{i+1}, y_{i+1}, t_{i+1})$, $v_{max}, t_s, m, b)$ consisting of:

- The 2 GPS-based location-in-time samples (x_i, y_i, t_i), and $(x_{i+1}, y_{i+1}, t_{i+1})$ along with the v_{max} speed bound.
- The time instant of detection of the road-side sensor.

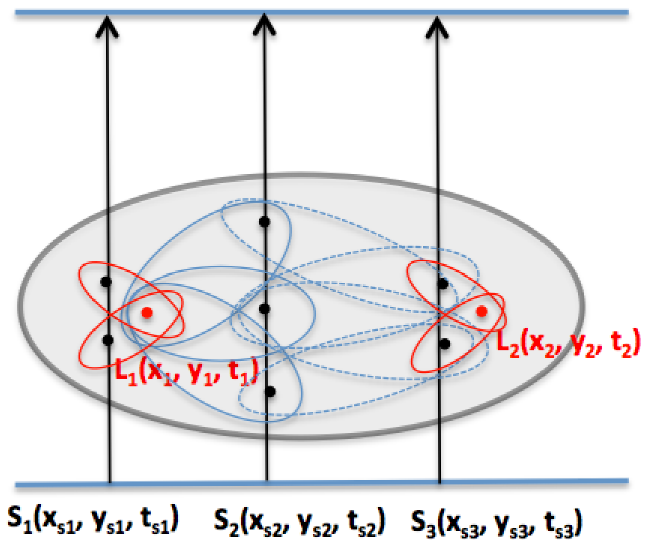

$L_1(x_1, y_1, t_1)$
$L_2(x_2, y_2, t_2)$
$S_1(x_{s1}, y_{s1}, t_{s1})$
$S_2(x_{s2}, y_{s2}, t_{s2})$
$S_3(x_{s3}, y_{s3}, t_{s3})$

Figure 2.4. Multiple roadside sensors intersecting a bead

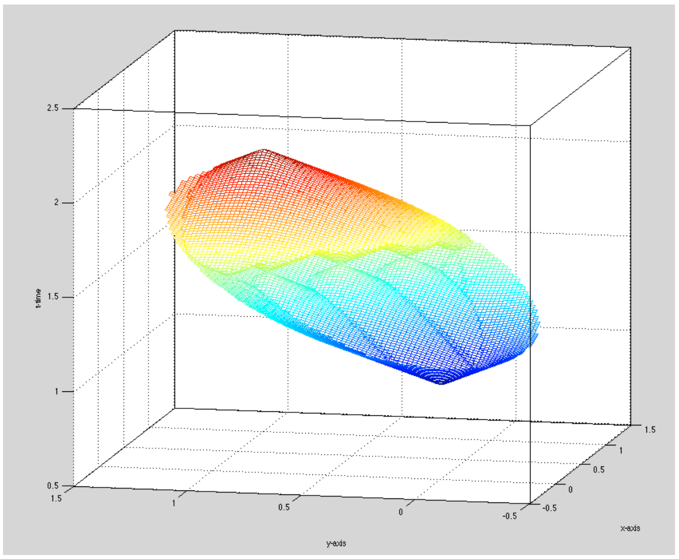


Figure 2.5. Outer boundary of the fused uncertain locations

- The parameters of the equation $y = mx + b$ (in a given referent coordinate system) of the line specifying the corresponding line-segment emanating from the roadside sensor and specifying the locations of the possible new foci.

When it comes to bounding the possible whereabouts, an intuition may cause one expect that some of the points along the intersection of the line segment with the ellipse El_1 may yield possible focal points that would generate ellipses which are not fully contained inside El_1. However, the set of constraints in (3) will eliminate every portion which is outside the intersection of the original El_1.

We now proceed with a formal analysis of an important property of the FB model, towards which we first recall some of the properties of the bead model presented in [**55**]. Let $B(x_i, y_i, t_i, x_{i+1}, y_{i+1}, t_{i+1}, v_{max})$ denote[1] the bead between two location-samples (x_i, y_i) and (x_{i+1}, y_{i+1}) at respective times t_i and t_{i+1}, during which the speed is bounded by v_{max}

[1]The original notation in [**55**] was $B(t_i, x_i, y_i, t_{i+1}, x_{i+1}, y_{i+1}, v_{max})$ and we slightly modified it for consistency with the rest of the notation in this article.

Property 1. Given (x_i, y_i, t_i), and $(x_{i+1}, y_{i+1}, t_{i+1})$ with $t_i < t_{i+1}$ and $v_{max} > 0$, any trajectory from (x_i, y_i, t_i) to $(x_{i+1}, y_{i+1}, t_{i+1})$ for which the speed at any moment $t_i \leq t \leq t_{i+1}$ is less than v_{max} is located within the bead $B(x_i, y_i, t_i, x_{i+1}, y_{i+1}, t_{i+1}, v_{max})$ and the projection of such a trajectory on the (x, y)-plane is located within $\pi_{x,y}(B(x_i, y_i, t_e, x_{i+1}, y_{i+1}, t_{i+1}, v_{max}))$. Furthermore, for any point (x, y, t) in $B(x_i, y_i, t_i, x_{i+1}, y_{i+1}, t_{i+1}, v_{max})$, there exists a trajectory from (x_i, y_i, t_i) to $(x_{i+1}, y_{i+1}, t_{i+1})$ which passes through (x, y, t).

Property 1 explains the bounding relationship between trajectory and bead. Taking the constraint (3) into consideration, one can deduce the following corollary:

Corollary 1. For any possible trajectory from x_i, y_i, t_i to $(x_{i+1}, y_{i+1}, t_{i+1})$ inside the bead, there is always a longer which passes through a point that lies on the boundary of the ellipse

$$\frac{(2x - x_1 - x_2)^2}{v_{max}^2(t_2 - t_1)^2} + \frac{(2y - y_1 - y_2)^2}{v_{max}^2(t_2 - t_1)^2 - (x_2 - x_1)^2 - (y_2 - y_1)^2} = 1 \tag{2.4}$$

is the longest possible trajectory.

Corollary 1 can be perceived as a consequence of the (extension to the) fact that the sum of the lengths between a point on the boundary of the ellipse to its foci is constant – i.e., the definition of the ellipse as a geometric set of points. In a similar spirit, and based on these properties of the bead model, we now have the following property regarding the FB model:

Lemma 1. Any bead generated by: (1) a focal point located in the GPS-based sample, and (2) a point from the line segment P_1P_2 representing possible locations obtained via a roadside sensor, is contained within the original bead.

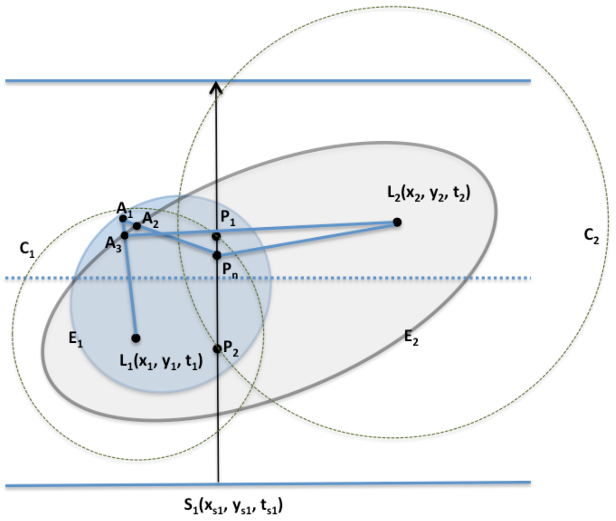


Figure 2.6. Illustrating the proof of fused bead containment

Proof. We prove Lemma 1 by contradiction. Assume that P_n is a point on the line segment P_1P_2 and consider the ellipse El_2 with foci P_n and L_1. Let A_1 denote a point which lies within El_2 but outside the original bead El_1, defined by the original bead (i.e., foci L_1 and L_2, and v_{max} bounding speed). Using Figure 2.6 as an illustration, we proceed with connecting the two line segments L_1A_1 and A_1P_n. They intersect El_1 at some points, denote them A_2 and A_3. According to Corollary 1, the polyline with two segments $L_1A_3L_2$ is the longest trajectory that the object could possibly move along from L_1 to L_2. However, by assumption, A_1 is bounded to be within El_1 which, in turn, implies that $L_1A_1P_n$ is a route of a valid trajectory from L_1 to P_n and, moreover, $L_1A_1P_nL_2$ is a route of a valid trajectory from L_1 to L_2. However, since, based on the triangular inequality, $\overline{A_3A_1} + \overline{A_1P_n} > \overline{A_3P_n}$ and

$\overline{A_3P_n} + \overline{P_nL_2} > \overline{A_3L_2}$, we have $\overline{A_3A_1} + \overline{A1P_n} + \overline{P_nL_2} > \overline{A_3L_2}$. Based on the last inequality, we can conclude that $\overline{L_1A_3} + \overline{A_3A_1} + \overline{A_1P_n} + \overline{P_nL_2} > \overline{L_1A_3} + \overline{A_3L_2}$, which implies that the trajectory $\overline{L_1A_1P_nL_2}$ is longer than trajectory $\overline{L_1A_3L_2}$. This, however, is a contradiction to the Corollary 1 which states that no other valid trajectory is longer than $L_1A_3L_2$, and we could conclude that assumption on the existence of point A_1 is not valid. $\square$

Lemma 1 demonstrates that whenever there is a location sampling from a roadside sensor in-between two GPS-based location samples, the set of (i.e., the union of) all the possible locations by the FB model are contained within the set of the possible locations bounded by the original GPS-based bead. An illustration of a FB-based segment is shown in Figure 2.5, and a visual comparison with the illustration of the full GPS-based bead (cf. Figure 2.1) reveals one of the consequences of Lemma 1. Another important consequence of Lemma 1 is in the conclusion that the FB will not introduce any false positives – in comparison with the traditional bead – when determining an intersection of the possible whereabouts with other (spatial, or spatio-temporal) entities.

2.3. Possible Locations at Time Instants

We now proceed with elaborating some basic calculations regarding the boundary of the possible locations of a given object at a specific time instant under the FB model, as well as the time-interval during which an object can be at a particular location. Subsequently, we also discuss the methodology for detecting whether the possible locations of a moving object are part of a given (spatial) range.

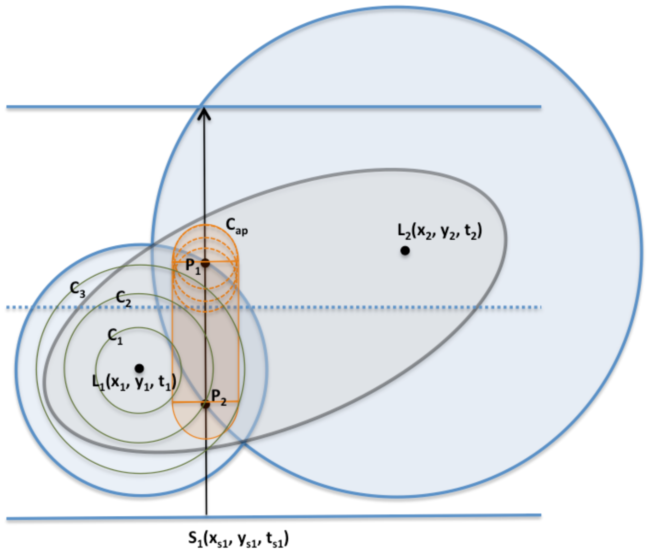


Figure 2.7. Cross section of fused bead

Recall that the FB model is based on the original bead obtained via GPS-based locations L_1 and L_2 (foci of a 2D ellipse El_1) and a road-side sensor providing possible locations along a line-segment perpendicular to a given road at a time instant t_s (cf. Figure 2.3a).

When it comes to location whereabouts at certain time instant t_{s1}, the regular bead model has a boundary defined by a lens $L_e(t_{s1})$ which obtained as the intersection of the circles with radii $v_{max}(t_{s1} - t_1)$ and $v_{max}(t_2 - t_{s1})$, centered at L_1 and L_2 respectively (light blue shaded area in Figure 2.7). If it happens that at that same time instant the object has been detected by a roadside sensor – then the object must be somewhere along the ray emanating from that sensors location and perpendicular to the road segment. However, because of the uncertainty boundary from the GPS-based location data, only the points along that ray which are inside the lens L_e are valid possible-locations – illustrated by the segment P_1P_2 in Figure 2.7.

Let $\varepsilon \in [0,1]$ denote a real variable. Any point $P(t_{s1}, \varepsilon) \in \overline{P_1P_2}$ which is a possible location of the object at t_{s1} has coordinates $x_{P(\varepsilon)} = \varepsilon x_{P1} + (1-\varepsilon)x_{P2}$ and $y_{P(\varepsilon)} = \varepsilon y_{P1} + (1-\varepsilon)y_{P2}$

With this in mind, given a time instant $t_i \in [t_1, t_{s1}]$, the possible locations of the moving object at t_i are bounded by the uncountable union of intersections between:

(1) The disk centered at L_1 and with radius $v_{max}(t_i - t_1)$.

(2) An infinite collection of disks, each centered at a point $P(t_{s1}, \varepsilon)$ along P_1P_2 and each with radius $v_{max}(t_s - t_i)$

In Figure 2.7, the circles C_1, C_2 and C_3 are examples of the boundaries of the objects whereabouts at different time-values (t_i) due to the GPS-sample at location L_1. For a fixed value of t_i Figure 2.7 also shows the boundary defined by the "envelope" of the union of the uncountably many disks centered along P_1P_2 – essentially, the sum of the line segment P_1P_2 and a disk with radius $v_{max}(t_s - t_i)$.

Depending on the time value and ε, there are five basic kinds of time-intervals during which shapes of the unions determining the object's whereabouts have distinct properties. We use the phrase *significant times* to denote the boundaries of those time-intervals.

(1) $t \in [t_1, t_i^{l1})$ (*Occurrence of the first lens*): During this interval, the possible locations are inside a disk centered at L_1 – this is the case when t_i is very close to t_1 – meaning: regardless of the value of ε, each disk with radius $v_{max}(t_s - t_i)$ centered at any point along P_1, P_2, fully covers the disk centered at L_1 with radius $v_{max}(t_i - t_1)$. Let P_c and P_f denote points along P_1P_2 which is geometrically closest and farthest to L_1 respectively. Clearly, point P_f will be the one with the earliest change of this kind

of containment with the disk – at some time instant t_i^{l1}, the intersection[2] will switch from a full-disk centered at L_1 into a lens defined by the intersection of the two disks: one centered at L_1 and one centered at P_f.

(2) $t \in [t_i^{l1}, t_i^{l_A})$ (*from a single lens, until "lenses_All"*): During this time interval, depending on the values of ε, some of the disks centered along P_1P_2 (each with radius $v_{max}(t_{s1} - t_i)$) are still fully covering the disk centered at L_1 with radius $v_{max}(t_i - t_1)$. These are the ones whose centers are closer to P_1 (i.e., $P(t_{s1}, \varepsilon)$ with ε closer to 0).

(3) $t \in [t_i^{l_A}, t_i^{d1})$ (*from lenses_All, until the first (full) disk appears*): This is the time-period during which each possible foci along P_1P_2 is a center of a disk with which yields a lens-shaped intersection with the disc centered at L_1. At the expiration of this time interval, the disk centered at P_c and with radius $v_{max}(t_s - t_i)$ is about to be fully covered by the disk centered at L_1 and with radius: $v_{max}(t_i - t_1)$

(4) $t \in [t_i^{d1}, t_i^{d_A})$ (*from a single full disk appearance, until disks_All*): similarly to the 2nd case above, during this time interval some of the disks centered along P_1P_2 have a lens-shaped intersection with the disk centered at L_1, while some are fully contained inside of it.

(5) $t \in [t_i^{d_A}, t_{s1})$ (*disks_All*): The last distinct time-interval for the part of the FB between the first GPS-based foci and the roadside sensor is similar to case "1" above, in the sense that every disk with radius $v_{max}(t_{s1} - t_i)$, regardless of where its center is located along P_1P_2, is fully contained inside the disk centered at L_1 and with radius $v_{max}(t_i - t_1)$.

[2]For clarity, we present the details of calculating t_i^{l1} and other significant times in the Appendix.

We note that for time-values $t \in [t_{s1}, t_2]$, the cases are analogous (and in reverse order) from the ones specified above, in the sense that there are four significant time instants defining five distinct intervals.

Let $D_1(t)$ denote the disk centered at L_1 and with radius $v_{max}(t - t_1)$. Also, let $D_P(t, \varepsilon)$ denote the disk centered at the point $P(t_{s1}, \varepsilon)$ with radius $v_{max}(t_{s1} - t)$. For a given 2D shape S, let $A(S)$ denote its area. Assuming a uniform distribution in each time-interval between two consecutive significant times[3], we obtain that the corresponding *pdf*s (probability density functions) are:

(1) $t \in [t_1, t_i^{l1})$:

$$f(x,y,t) = \begin{cases} \frac{1}{\pi(v_{max}(t-t_1))^2} & \text{if}(x,y) \in D_1(t) \\ 0 & \text{otherwise} \end{cases}$$

(2) $t \in [t_i^{l1}, t_i^{l_A})$

$$f(x,y,t) = \frac{1}{\pi(v_{max}(t-t_1))^2 + A(\cup_{\varepsilon > \delta_1(t)}(D_1(t) \cap D_P(t,\varepsilon)))}$$

where $\delta_1(t)$ is the smallest value of ε at a given t for which $D_P(t,\varepsilon) \not\subseteq D_1(t)$.

(3) $t \in [t_i^{l_A}, t_i^{d1})$

$$f(x,y,t) = \frac{1}{A(\cup_{\varepsilon}(D_1(t) \cap D_P(t,\varepsilon)))}$$

(4) $t \in [t_i^{d1}, t_i^{d_A})$

[3]Throughout this work, we assume independence between location-values in successive location samples (cf. [**15, 28**]).

$$f(x,y,t) = \frac{1}{\pi(v_{max}(t_{s1}-t))^2 + A(\cup_{\varepsilon>\delta_2(t)}(D_1(t) \cap D_P(t,\varepsilon)))}$$

where $\delta_2(t)$ is the smallest value of ε at the given t for which $D_1(t) \nsubseteq D_P(t,\varepsilon)$.

(5) $t \in [t_i^{d_A}, t_{s1})$

$$f(x,y,t) = \begin{cases} \frac{1}{\pi(v_{max}(t_{s1}-t))^2 + \overline{P_1P_2}\cdot(v_{max}(t_{s1}-t))} & \text{if}(\forall\varepsilon)D_P(t,\varepsilon) \subseteq D_1(t) \\ 0 & \text{otherwise} \end{cases}$$

When calculating the probability that a given moving object whose motion is modelled as an FB is inside a given spatial range at a given time instant, we need the area of the intersection. However, given the complexity of the boundary of the objects whereabouts, the calculation of overlapping area may necessitate relying on numerical integration methods.

2.3.1. Numerical Method for Complex Area Calculation

Selecting an approximate evaluation method, i.e., numerical method, depends on the task at hand. If we aim at calculating the intersection of two curves, the Newton-Raphson Method is the most widely used one, whereas calculating the area bounded by a given curve may rely upon Trapezoid Rule, Gaussian Quadrature Method or Monte Carlo Integration [**33**].

As an example, in a GPS-based bead, the location whereabouts given time instant are relatively straightforward to compute since they are either a circular disk or a lens formed by intersection of two circles. Moreover, finding the points where the boundary of the object's whereabouts intersect a given polygon is still achievable analytically, since the possibilities

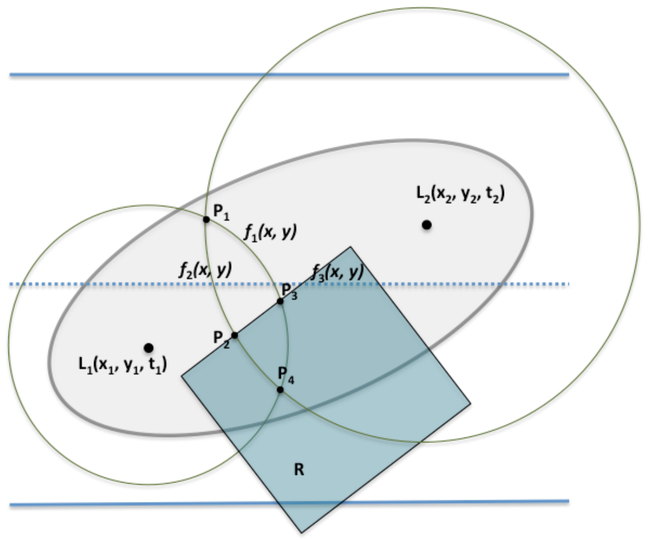


Figure 2.8. Area calculation in GPS-based bead

amount to calculate an intersection between a circle and a line(segment) is limited, as shown with P_1 and P_4 in Figure 2.8. However, even in such cases, one may need to use numerical methods for calculating the area of the intersection.

Given the complexity of the FB structure at a particular time instant, in this work we resort to approximate computations based on a spatial grid, as shown in Figure 2.9. Clearly, the size of the grid cell will affect the running time of the (execution of the) corresponding algorithms. However, there is another aspect to consider – the (im)precision. By the very definition of the FB, it is a union of uncountably many (subsets of) disks. Hence, we need to discretize the number of such disks, for which a basic unit Δd is introduced, specifying the locations of the centers of the disks that will be accounted for when calculating a particular area. These impacts are analyzed in Section 3.5.

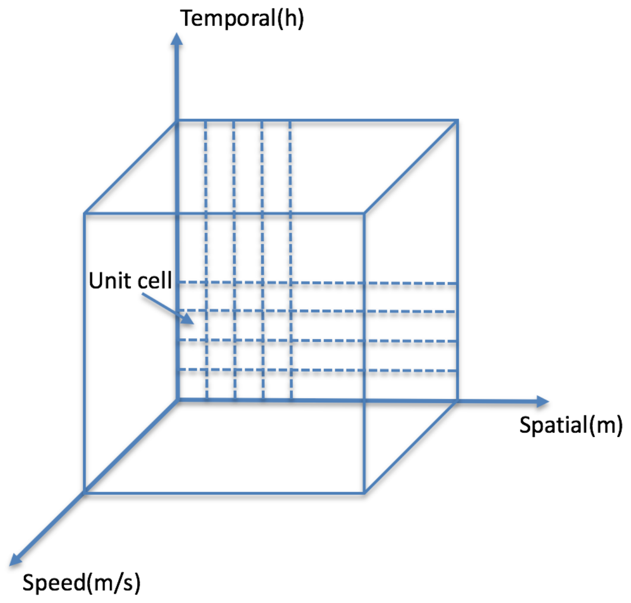


Figure 2.9. Grid based numerical approximation

2.4. Query Processing

We now turn our attention to processing spatio-temporal queries under the FB model. We start with the basic *where_at* and *when_at* location-in-time queries, followed by a range query and lane-crossing query. Lastly, we discuss the possibility of speeding up the query processing via pruning.

Without loss of generality, the presentation will use the setting of a single fused bead. However, when necessary, the issues that may arise due to considering the entire necklace will be explicitly addressed.

2.4.1. Basic Queries

Similarly to the GPS-based bead, in order to determine the whereabouts at a given time instant t for a fused bead, we need to obtain the intersection of FB with the horizontal plane

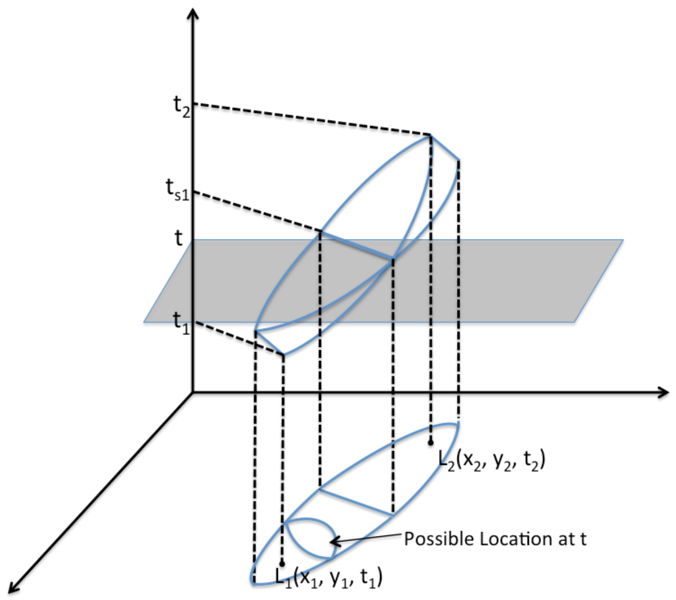


Figure 2.10. Whereabouts at Time Instant

Time = t. The corresponding illustration of the volume in 2D space + Time, along with the 2D projection, is shown in Figure 2.10. The boundary of the 2D projection is obtained as the "envelope" of the union of two collections of uncountably many intersections of disks centered along the line-segment originating at the roadside sensor, with the disk centered at L_1. The details were elaborated in Section 4.

The GPS-based bead (e.g., L_1) and the other centered at a point along the intersection chord (cf. $\overline{L_{11}L_{13}}$ in Figure 3.5) resulting from secant due to the roadside sensor and the arc from the lens of the GPS-based bead. Thus, one of the boundaries is always a circular arc originating at the focal point of the "original" GPS-based bead, centered at focus of the GPS-based bead (say, L_1) and with radius $v_{max}(t - t_1)$. The boundary is actually the boundary of the union of uncountably many disks with radii $v_{max}(t_{s1} - t)$, with centers along the intersection-chord.

The complementary query, *when_at(oID, L)* returns the times during which it is possible for the object oID to be at the location $L(x_L, y_L)$, i.e., a time-interval $[t_{L1}, t_{L2}]$. The time-interval can be defined as the two intersections between the boundary of the fused bead FB and the vertical line (i.e., ray) emanating from L. To calculate the values, we have the following observations:

(1) t_{L2} is the latest time that a circle located at the GPS-based focus from the sample at t_1 will "reach" L – hence, it can be obtained as a solution to the equation: $\overline{L_1L} = v_{max}(t_{L2} - t_1)$

(2) t_{L1}, on the other hand, is the earliest time that any circle with the center on the intersection chord(P_1P_2 in Figure 2.7) and radius $v_{max}(t_s - t_{L1})$ would pass through L.

2.4.2. Range Query Processing

A typical spatial range query aims at retrieving the spatial (static) objects which have a particular topological relationship (e.g., inside, intersect, etc...) with a given range, which is, an entity with spatial extent [**90**]. A distinct feature of spatio-temporal range queries is that they are continuous – i.e., the answer may change with time: for example, an object that was inside a given query region may subsequently exit it, and vice versa. In our settings, the key observation is that we need to take into account the uncertainty of the object's location at a given time instant when formulating the syntactic variants of the range query [**93, 102**].

In this work, we assume that the spatial region of interest for the range query is bounded by a simple polygon R (Figure 2.11) and we also assume $[t_{bq}, t_{eq}]$ values indicating the bounds of interest in the temporal dimension. We denote the set $\{\forall(x, y, t) | (x, y \in R$ and

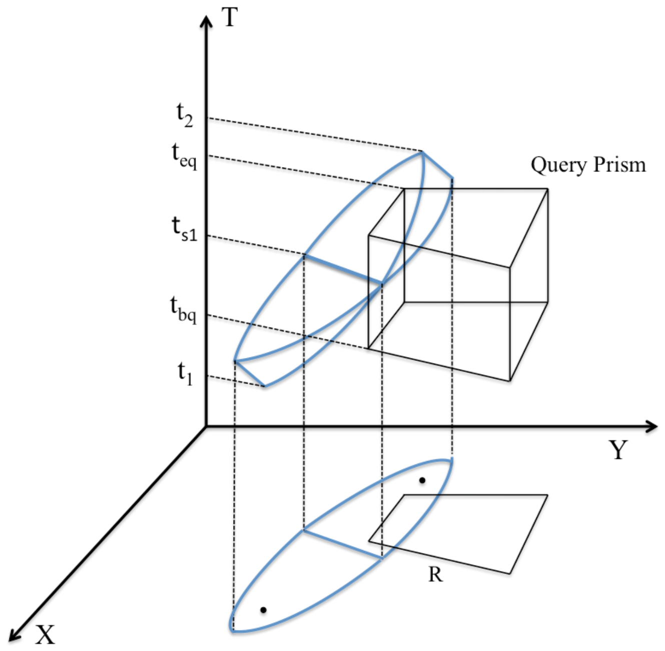


Figure 2.11. Range query for a given prism

$t \in [t_{bq}, t_{eq}])\}$ for QP_R (query prism). Earlier works [**102, 100**] have provided qualitative variants regarding the domains of space and time in the sense of uncertain object being inside R: (1) *sometimes* or *always* throughout the time-interval of interest; and (2) *possibly* or *definitely* so.

For a given uncertain trajectory represented as a sequence of FBs, $Tr = [FB_1, FB_2, \ldots, FB_n]$, where each $FB_i = ((x_i, y_i, t_i), (x_{i+1}, y_{i+1}, t_{i+1}), v_{max}, t_s, m, b)$, we are interested in answering the following type of a range query:

$\mathbf{Q}_R^u$: *Does the moving object have a probability* $\geq \Theta$ *of being inside* R *at least* ϕ *of the time-interval* $[t_{bq}, t_{eq}]$.

We use the generic notation *Inside* $(FB, R, t_{bq}, t_{eq}, \theta, \phi)$ to denote the (parameterized version of the) queries like $\mathbf{Q}_R^u$, with the intended meaning $\exists \phi$ – a sum of time-intervals (not necessarily contiguous) during which the ratio of the intersection of the FB and QP_R is greater than Θ. We note that [**100**] proposed analytical solutions for answering existential/universal variants by verifying intersecting conditions between ellipses and circles in the traditional bead model. Thus, for example, one could verify whether *Sometime Inside* (FB, R, t_{bq}, t_{eq}) based on an existence of a time instant at which the intersection between R and FB is not empty – which corresponds to any $\Theta > 0$ in the current context. Similarly, the predicate *Always Inside* (FB, R, t_{bq}, t_{eq}) would amount to $\Theta = 1$ throughout the entire time-interval of interest for the query.

Following our discussions in Section 4, the probability *Prob(X, Y, T)* (i.e., the probability that the object is inside a region bounded by implicit curves "X", "Y" throughout a time-interval "T") is defined as triple integral on 2D+time:

$$Prob(X, Y, T) = \int_T \int_Y \int_X f(x, y, t)$$

The grid based numerical method provides an estimation regarding areas of location whereabouts at a certain time instant. Assuming a uniform *pdf* at any time instant, we have :

$$\begin{aligned} Prob(X, Y, T) &= \frac{\int_T \text{Overlapping Area between FB and } R(t)}{\int_T \text{Possible FB Whereabouts}(t)} \\ &= \frac{\text{Overlapping Volume between FB and } QP_R}{\text{Overall volume of FB}} \end{aligned}$$

Let $A_{FB}(t)$ denote the area of the possible whereabouts of the object at time t (i.e., the area of the region corresponding to the answer of *where_at(t)* query) and let $A(R)$ denote the area of the query region R. We have the following algorithm:

Algorithm 1 Inside $(Tr_{FB}, R, t_{bq}, t_{eq}, \theta, \phi)$

```
1: float $T = 0$, $t_{total} = 0$;
2: int $k = 0$;
3: while $(t_q + k \cdot \triangle) < t_{eq}$ do
4:     $T = T + \triangle$;
5:     if $((A_F B(T) \cap A(R))/\ A_{FB})(T) \geq \theta$ then
6:         $t_{total} = t_{total} + \triangle$
7:         if $t_{total} \geq \phi$ then
8:             The trajectory satisfies the predicate;
9:             Exit;
10:        end if
11:    end if
12:    $k++$;
13: end while
14: Trajectory satisfies the predicate only $t_{total}$ of the $[t_{bq}, t_{eq}]$;
```

Algorithm 1, without checking the value of the "time-accumulator" and with a minor addition to sum up the values $((A_F B(T) \cap A(R))/(A_{FB})(T)) \times \triangle$, can be used to calculate the ratio of the volume (i.e., the corresponding probability) of the object being inside R. We note that, if one simply wants to calculate the probability of an object being inside QP_R, without any concerns about Θ or ϕ (i.e., overloading the argument-signature), then the "If()" test in Algorithm 1 can be eliminated, and the corresponding approximations summed up.

As our experimental results in Section 3.5 will illustrate, applying Algorithm 1 to process range queries over uncertain trajectories modeled with FB consistently yields fewer false positives, in comparison to the case of applying it to a collection of uncertain trajectories represented via regular beads.

2.4.3. Lane-crossing Query Processing

Lane-crossing query can be perceived as a special case of a range query where the query prism is degenerated from a polygon into a half plane. Figure 2.12, illustrates the lane-crossing query for the regular bead and FB models. As mentioned in Section 1., the lane-crossing query is important in applications related to fleet management and efficient traffic management. We reiterate the statement explaining such queries:

$\mathbf{Q}^u_{LC}$: *Given a fused bead* $FB(((x_i, y_i, t_i), (x_{i+1}, y_{i+1}, t_{i+1}), v_{max}, t_s, m, b))$*, does the moving object have* $> \Theta$ $(0 < \Theta \leq 1)$ *probability of crossing the lane and entering half-plane* R.

We use the generic notation *Lane-Cross* $(FB, L, t_{bq}, t_{eq}, \theta)$ to specify the corresponding predicate expressing the fact that an uncertain moving object represented via FB has crossed the lane L on a given road segment (cf. Section 2.) with a probability $\geq \Theta$, sometime between $[t_{bq}, t_{eq}]$. If we wish to calculate the total probability of an object crossing the lane L throughout the entire time-interval of interest of the query, then we can obtain

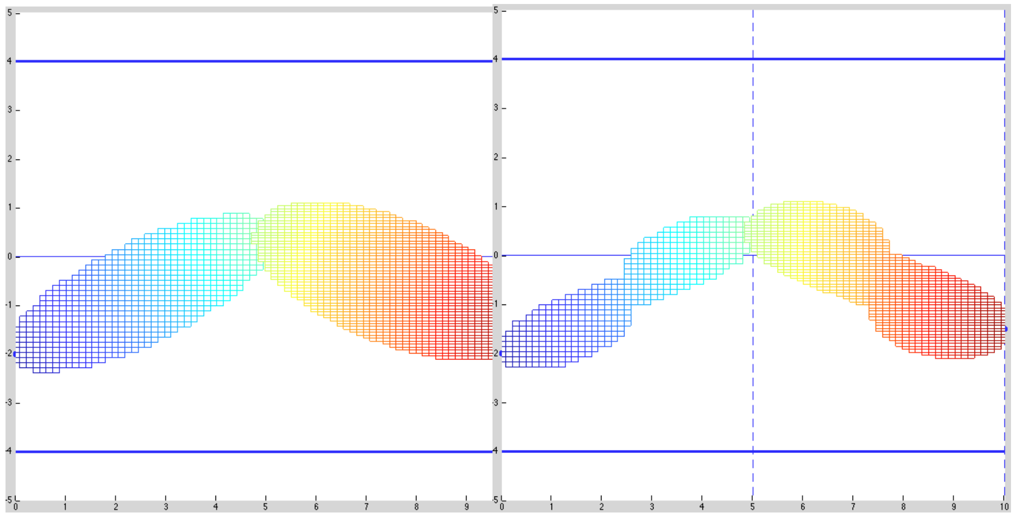

(a) Bead model (b) Fused bead model

Figure 2.12. Beads and lane-crossing query

an approximate value by applying similar ideas as in Algorithm 1 – i.e., summing up the products of the intersection area with $\triangle$.

2.4.4. Pruning Techniques

Typically, spatio-temporal query processing proceeds in three "stages" [**38**]:

(1) *Filtering*, where an index is used to eliminate those data items that are guaranteed not to satisfy the query [**97**]; followed by:
(2) *Pruning*, where some properties might be used to further reduce the set of the possible candidates for the answer, portion without introducing any false negatives;
(3) *Refinement*, where algorithmic checks and calculations are used to eliminate false positives that were not eliminated during the previous stage(s).

While the problem of efficient and effective indexing structures for processing spatio-temporal queries over the FB model is outside the scope of this work, we note that for the specific queries discussed here, there may be pruning approaches that can speed up the overall execution of the spatio-temporal queries on FB model. In the sequel we discuss few such strategies:

A. Definitely Outside – Individual Fused Bead Bounds (IBb)

Proposed in [**100**], this pruning strategy is designed for GPS-based bead. It approximates each GPS-based bead with its minimum bounding vertical cylinder. According to the Lemma 1, FB is bounded by GPS-based bead, which justifies its application to FB as well. In effect, the ellipse – which is the projection of a bead, formed by two GPS points belongs to

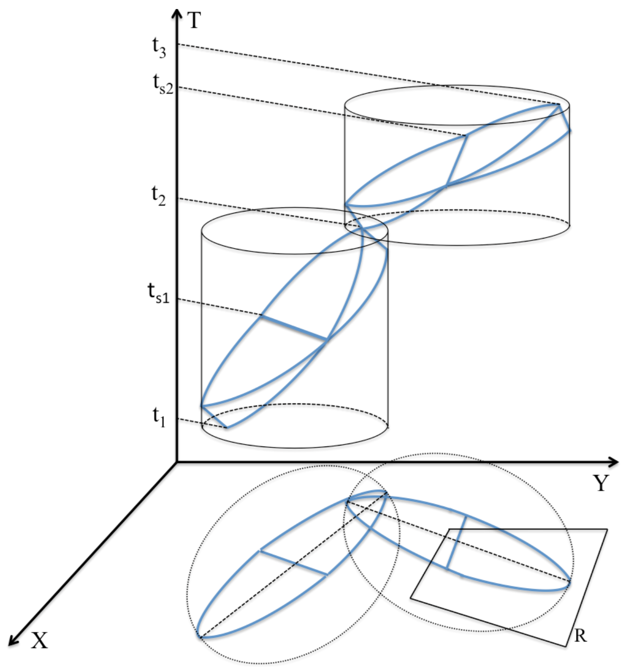


Figure 2.13. Cylinder-based pruning approximation

FB, on (X, Y) plane, becomes a circle centered in the center of the respective ellipse, as shown in figure 2.13. The radius of the approximation-disk Ad_i is: $r(Ad_i) = 1/2(v^i_{max})(t_{i+1} - t_i)$.

B. Definitely Inside – GPS points pre-screening

This pruning technique is specially designed for lane-crossing query, where the predicate determines if it is possible for a lane-cross to occur. The technique is based on the following observation: if two consecutive GPS points are located on two different sides of the central line, there must be at least one time instant at which the moving object crosses the road, in which case we are able to prune the FB and direct return true.

C. Sometime Inside – fine grained dead-space removal

We are interested in finding the time instant(s) when uncertain trajectories enter/exit the query region R – call them *critical points*. By doing so, we eliminate some redundant time-intervals with respect to the time-bounds of a particular query. The general case for time $t \in [t_i, t_{i+1}]$ being a critical point occurs when the intersection of the uncertain region at t with a query rectangle is a single point. In the time interval $[t_i, t_s]$, the single-point-intersection between disk centered at the first GPS point and query region stands for the entering moment. Similarly, in the time interval $[t_s, t_{i+1}]$, the single-point-intersection represents exiting moment. Since the query region is represented as polygon in the (X, Y) plane, each edge of the polygon is defined as a segment of 2D line $y = ax + b$. The calculation of the critical times is presented in the Appendix.

2.5. Experimental Observations

We now present the experimental observations regarding the traded-offs between the benefits of the FB model in terms of reducing the number of false positives in the queries' answers vs. the computational costs. More specifically, we implemented the proposed approach and tested it for lane-crossing query and range query, comparing the beads obtained using only GPS data against the FB model, and ran comprehensive experimental comparisons based on correctness, robustness and efficiency. In addition, we present two types of pruning techniques which we applied as part of the query processing and discuss their impact.

2.5.1. Dataset Description

In our experiments we used both synthetic and real-life datasets.

Synthetic Data: The synthetic data was generated by a modified version of Brinkhoff network-based generator, representing vehicles' movements on road network. GPS points are

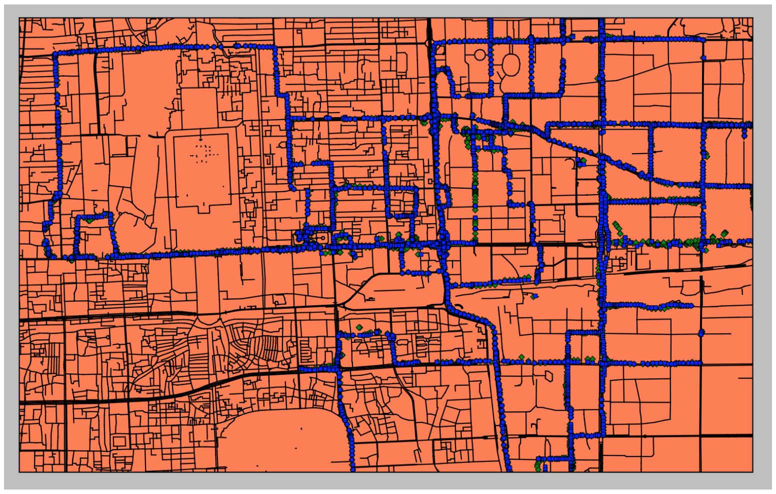


Figure 2.14. Beijing road network and taxi dataset

generated on the map of Oldenburg, which are available at the Brinkhoff generator official website (http://iapg.jade-hs.de/personen/brinkhoff/generator/).

Real-life Data: The real world dataset we used in our experiments is based on Beijing taxi data from the T-Drive project [**119, 120**]. Essentially, the Beijing road network is built based on OpenStreetMap data, containing 140207 vertices and 155997 road segments. GPS points are map-matched to road network using point-to-curve matching approach [**81**]. Figure 2.14 illustrates the map matching process, where green dots correspond to the raw GPS points, and the blue dots are the points obtained after map-matching process. To minimize the impact of the measurement errors, we filtered out the low speed GPS points (i.e., ones with speed less than 1m/s).

Following is the description of the setups that were applied in order to run the experiments for each of the queries:

(1) Vehicles are allowed to move along the road network with a speed $\leq$ 50km/h.

(2) We add a *width* parameter to the road network, the value of which is set to be $\leq$ 4m [**85**].

(3) At each time instant of the object's motion along the road, its width location is generated by a python-based random generator within a given random interval based on the width parameter used in that location. The values are selected such that 0 represents the center of the road; negative values represent left lane; and positive values represent the right lane – with respect to the direction of the object's motion.

(4) We apply additional post-processing to the trajectories by adding roadside sensor data. As mentioned, for a given location on the right (i.e., in the direction of object's motion) side of the road, we generate a ray perpendicular to the road's boundary.

(5) To cater to the variations of the speed, we vary the actual time at which the moving object crosses the ray corresponding to a particular roadside sensor. Given a bead $B(x_i, y_i, t_i, x_{i+1}, y_{i+1}, v_{max})$ and the ray $y = mx + b$ from a given roadside sensor, we calculate the time interval during which a moving object can cross the (ray generated at the) location of the sensor as:

$$\begin{aligned}[T_{smin}, T_{smax}] = [t_i + \frac{\text{distance from } (x_i, y_i) \text{ to sensor}}{v_{max}}, \\ t_{i+1} - \frac{\text{distance from } (x_{i+1}, y_{i+1}) \text{ to sensor}}{v_{max}}]\end{aligned}$$

Then we calculate the sensor time t_s following normal distribution between T_{smin} and T_{smax} with average $\mu = (T_{smin} + T_{smax})/2$ and standard deviation $\sigma = ((T_{smax} + T_{smin})/2 - T_{smin})/2.5$.

Experiments are conducted based on the synthesized dataset described above, and the executional environment was a 64 bit jdk running on a Linux system with 4-core i7-3770 CPU with 3.40GHz, and 8GB of memory.

2.5.2. Granularity of the Numerical Solution

As mentioned in Section 2.3, a grid based numerical method is used to measure the cross section area of FB given a certain time instant. Since the cell size significantly influences the area estimation accuracy, we measured the number of grids between two GPS points as a metric to determine the level of granularity. In the experiment, a traditional bead is formed by two GPS sample points with a maximum speed. We pick an arbitrary query time and calculate the location, which acts as a true value A_t. The estimated area is denoted as A_e, and the relative error is defined as: $\delta A = \frac{|A_e - A_t|}{A_t}$ as the ratio between residual and true value.

When determining the grid size, a 1% tolerance was chosen as a threshold and multiple runs of the experiment were performed for beads with different distance between two GPS sample points.

As expected, we observe in Figure 2.15 that the estimation errors decrease as the grid size increases, and the numerical estimations for different size of beads – with distance ranging from 1m to 1000m – have identical errors given the same grid size. Hence, to reach 1% error tolerance, we choose grid size to be such that there are 100 units between two successive GPS points.

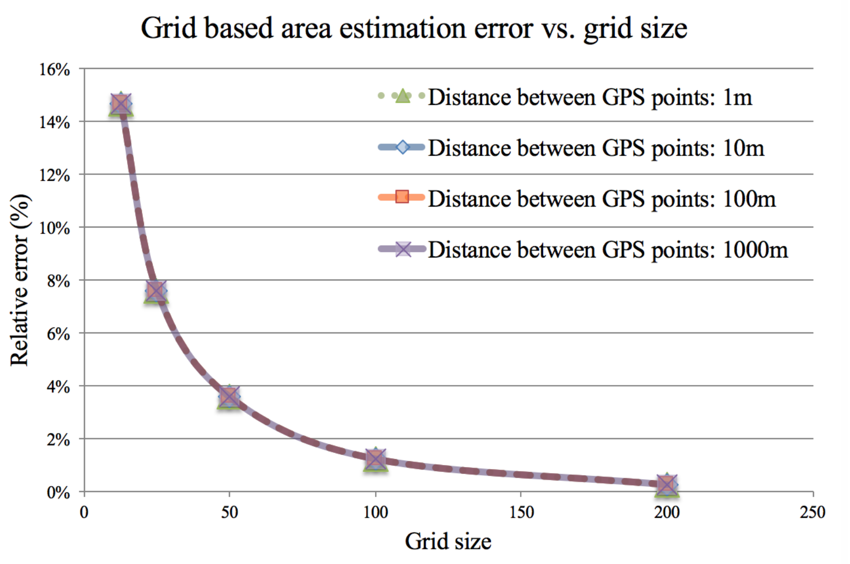


Figure 2.15. Impact of grid size on the estimation accuracy

2.5.3. Lane-Crossing Query Experiment

We assume road networks are composed of two-lane roads. A sequence of trajectories with different lengths are generated and the data sets we used in experiments are not correlated - that is, we generate each dataset separately.

Correctness improvement:

Figure 2.16 illustrates the number of false positives when lane-crossing query are applied to trajectories, under the bead and FB model. As we can see, the FB eliminates around 40% of the false positives from GPS-based bead model, due to its reduction of "dead-space".

As a follow-up experiment, we reduced the percentages of FBs contained in trajectories to five levels (0%, 25%, 50%, 75% and 100%), in order to mimic real situations when roadside sensors are not fully and densely deployed on a given road network. Figure 2.17 reveals the

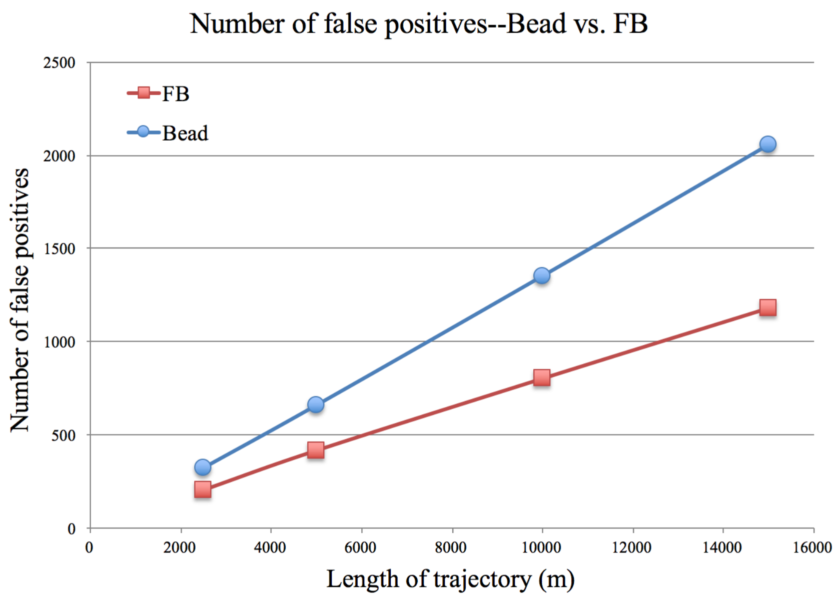


Figure 2.16. Lane-crossing query — FB reduces number of false positives

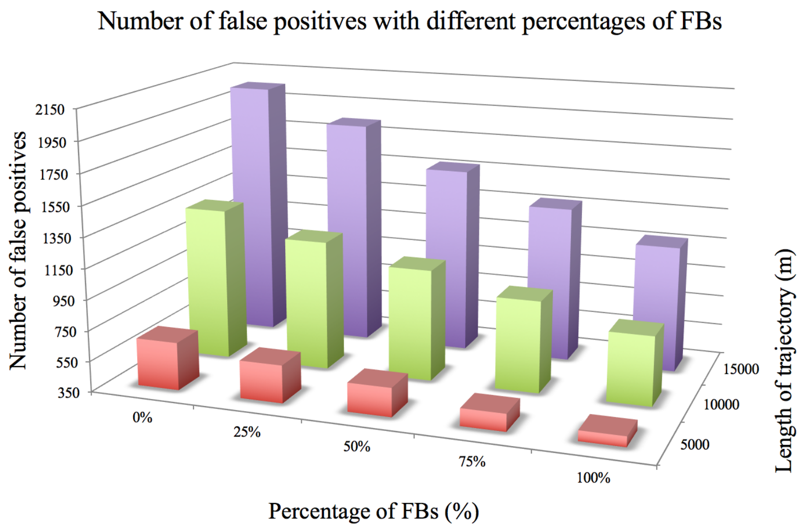


Figure 2.17. Percentage of roadside sensor deployed influence the number of false positives

relationship between percentages of FBs and number of false positives. Clearly, the more FBs contained as components of a trajectory, the smaller the overall number of false positives.

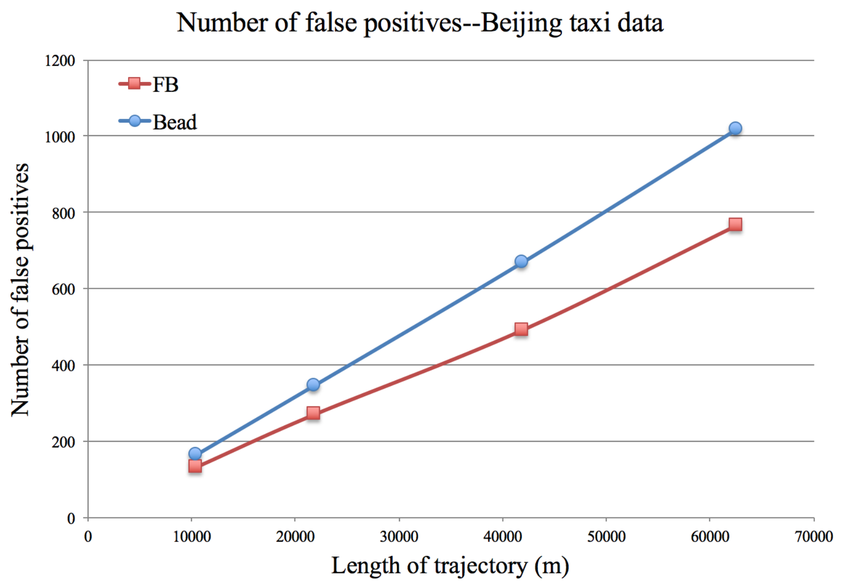


Figure 2.18. Lane-crossing query on Beijing taxi data set

When we apply the same experiment to Beijing Taxi data, its outcome indicates the same effect, where 26.6% false positives are reduced, as shown in Figure 2.18. The real life scenarios contained in Beijing taxi data is highly complicated, with continuously changing speed, compared with Brinkhoff trajectory generator where vehicles drive under a constant speed within each road segment. Despite the varieties and complications in real life data, our algorithm is adaptive and effective in reducing location uncertainties.

Sensor deployment in real world applications is largely constrained by factors such as budget, terrain, infrastructure, etc. To add to the realistic aspects of the experiments, we examined the influence of sensor deployment density. The effects of executing the lane-crossing query for the same dataset but for different sensor densities are illustrated in Figure 2.19. As shown, the higher the sensor deployment density is, the more false positive we are able to reduce with the FB model.

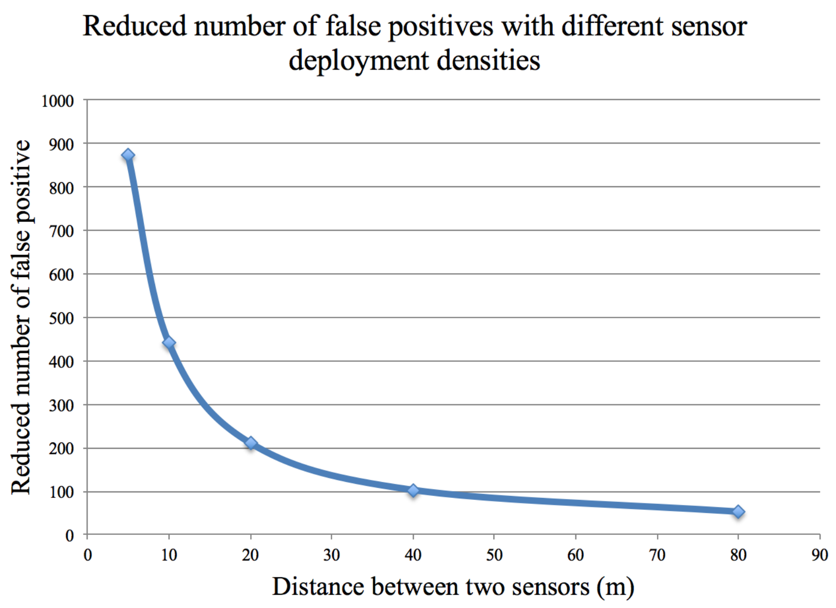


Figure 2.19. Road-side sensor deployment density influences the number of false positives

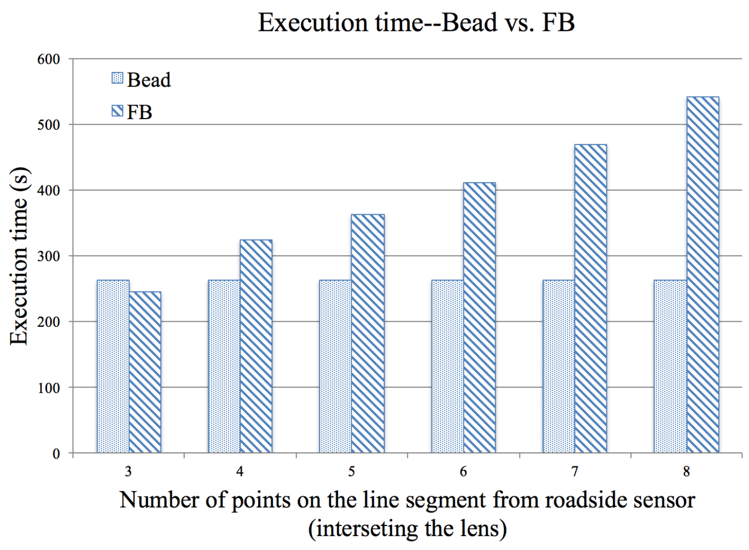


Figure 2.20. Execution time comparison

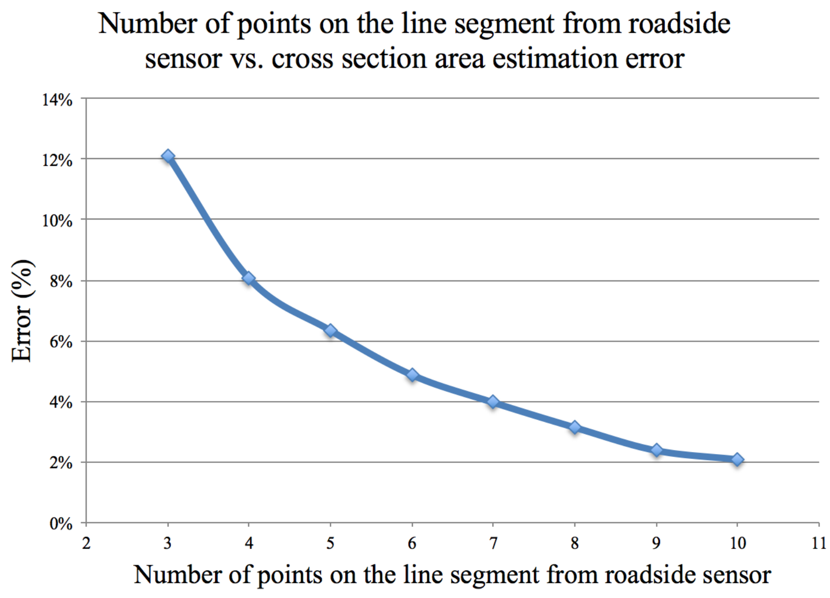


Figure 2.21. Number of points on the line segment from roadside sensor affects estimation accuracy

Efficiency:

Next, we compare the performance of GPS-based bead and FB in terms of the respective execution times. An important parameter affecting the execution time is the number of points chosen to approximate the line segment from roadside sensor (intersecting with lens), which are used to construct sub-beads. Recall from Section 3, that the FB is the union of uncountably many sub-beads, with one of the foci located along the ray emanating from the roadside sensor, within the width of the road. When approximating the uncountably many sub-beads, the trade-off is the precision (i.e., using as many points as possible to have an accurate approximation) vs. the computational overheads. Two experiments were performed to compare both execution efficiency and approximation accuracy, executing lane-crossing queries on trajectories with lenght $\leq$ 5000m, varying the number of points chosen on the roadside sensor line segment. The total execution times for each of the GPS-based

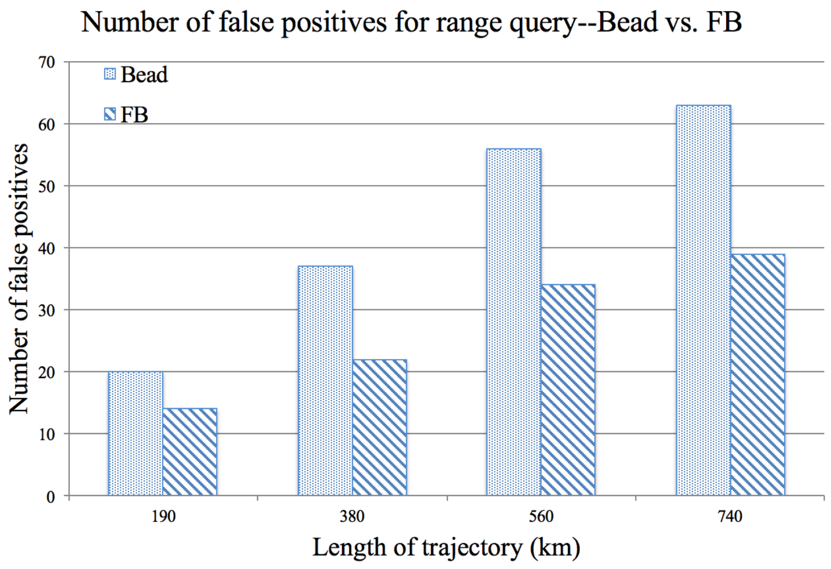


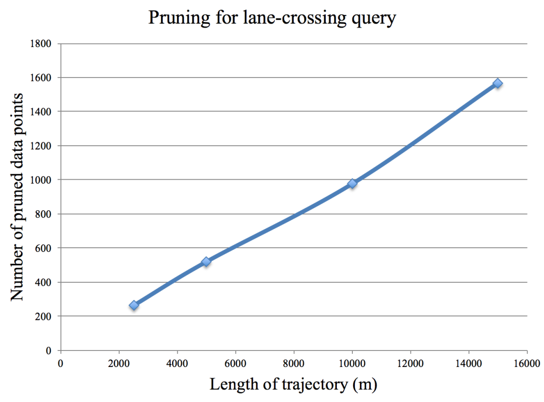
Figure 2.22. Range query — FB reduces number of false positives

bead model and FB model are displayed in Figure 2.20. On a complementary note, taking maximum number of points allowed in a given grid setting as true value, their corresponding approximation errors regarding location whereabouts areas are shown in Figure 2.21.

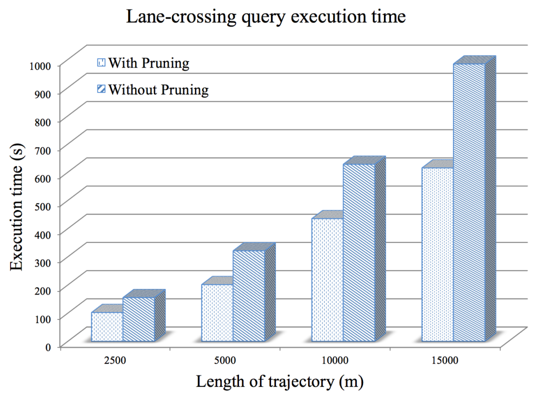
To discuss one specific setting explicitly: in the case when six points were used on the sensor line segment as a parameter for constructing the FB, in comparison with GPS-based bead model: (1) we have an overhead of a 30% latency in time to complete the query; (2) however, in return, we reduce more than 26% false positives and gain a much more narrow possible locations boundary (modulo the accuracy of the area approximation).

2.5.4. Range-query Experiment

The datasets used in range query are generated in the same fashion as the ones for the lane-crossing query, and the query regions we used were squares and disks with respective areas covering 12.5% of the total map area. As shown in Figure 2.22, similarly to the results



(a) Number of data pruned in lane-crossing query



(b) Execution time for lane-crossing query

Figure 2.23. Pruning techniques applied to lane-crossing query

for the lane-crossing query, FB outperforms GPS-based bead model by reducing 30% false positives .

2.5.5. Impact of Pruning

As discussed in section 2.4, we postulated that pruning techniques can significantly improve the efficiency of the queries processing. In addition, during the early stages of our experiments, we also observed that the fine grained dead-space removal method did not provide any significant performance boost. Thus, we introduced the Individual Fused Bead boundary (IBb) as a volume corresponding to a cylinder in 2D + time space, since in Algorithm 1 the most computationally expensive part is to aggregate the total volume in in the respective 2D + time space. Even though the redundant time removal method eliminates some time-intervals with no intersections between FB and query prism from the refinement, we note that one still needs to calculate the volume for those parts.

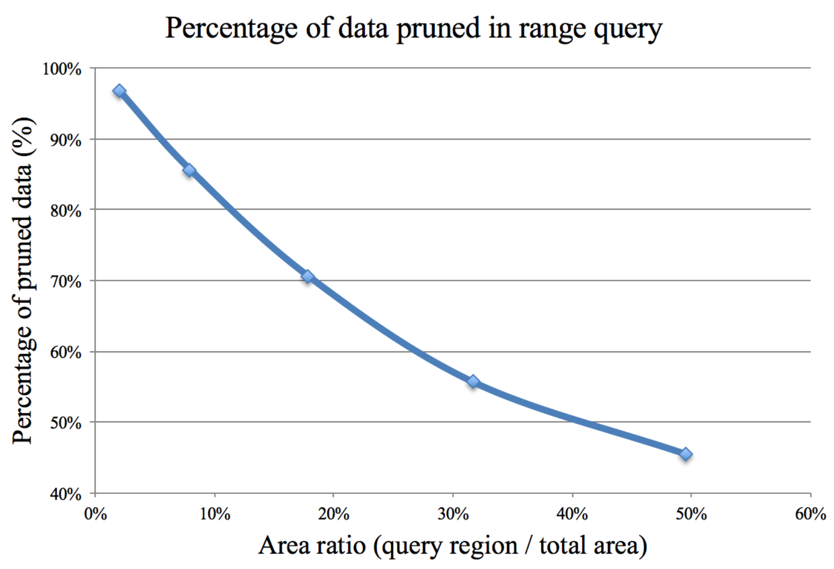


Figure 2.24. Pruning — Individual Fused Bead Bounds

Figure 2.23 shows the effects of pre-processing based pruning. As the Figure 2.23b indicates, around 30% of other total time have been saved via pruning. Figure 2.23a shows that as the trajectory length increases, the number of FB that can be pruned increase accordingly.

Our experiment for the effects of pruning in the case of range query were conducted with a data set containing trajectories with a total length of 55km. IBb rules out large amount of data points that definitely have no intersections with query prism. In Figure 2.24, x-axis represents the ratio of the areas of the query region and the entire map. It indicates the relationship between the pruning effect and the size of query region. As we expected, when the query area decrease, the IBb prunes more data points.

2.6. Related Work

There are three main bodies of research literature that are related to the work presented in this article and, in one way or another, were used as foundation for our work.

The first body of works consists of the results from the GIS, MOD and spatio-temporal databases communities, where the problem of capturing the uncertainty of motion has been studied extensively. Starting with [**39**], and more recently [**115**], the issue of uncertain whereabouts from the perspective of probabilistic time geography has been tackled by a model of emanating cones-in-time, with a vertex at the last location sample. The 2D boundary of the possible locations of moving objects with bounded speed was formalized by an ellipse in [**77**], and its 2D+time version – beads – was presented in [**45**]. Subsequently, [**54, 55**] provided a full formalization of the beads model and also provided extensions to capture the impact of road networks [**52**]. A plethora of the works dealing with uncertainty (either in free-space motion or road networks constrained) from MOD and spatio-temporal databases community have also addressed the efficient processing of popular spatio-temporal queries (range, (k)NN, reverse-NN) under various models of uncertainty [**38, 15**].

Unlike these works, we focused on fusing the uncertain location data from two sources – GPS and the roadside sensors. In addition, to illustrate some of the features of the new model and its use in query processing, unlike the traditional MOD-based works we considered the road network which has a width as a parameter, instead of graph edges.

The second body of works originates in the transportation and traffic management communities. Substantial efforts have been made to tackle the lane-crossing query and several works have focused on building novel system to overcome the shortcoming of single GPS receivers which yields unstable measurements with large uncertainty [**18, 25**]. Complementary attempts have been made to acquire location data using commercially available smartphones [**87**], but nearly 50% of the data failed to fall within the road network region. Other efforts include the use of integrated sensor like gyroscope to fill the unknown values between two GPS sample updates [**99**]. However, the works did not consider the uncertainty in-between consecutive GPS-based updates and sensor-based location detections.

Some of the works [**25, 99**], use map matching algorithms to determine which lane the vehicle belongs to and, subsequently, try to revise the measurement error using post-processing. However, the bead (or, space-time prism) model has not been exploited.

The third body of works originates from the Wireless Sensor Networks (WSN) community. Tracking of moving objects is considered to be a canonical research problem in WSN settings. Various facets of the problem have been investigated: from the trade-off between energy consumption and the accuracy of the tracking process, to routing protocols for conveying location-in-time information to a given sink (see, e.g., [**10, 14, 46, 76, 112**]). Typically, the location of a given object is determined by some form of collaborative trilateration among the tracking sensors equipped with different distance-estimation devices (e.g., vibrations, audio-strength, etc.). However, to the best of our knowledge, there have been no results on fusing the location data from heterogeneous sources.

2.7. Concluding Remarks and Future Works

We proposed a formal model – FB (fused bead) – for capturing the possible whereabouts of a moving object whose location data is obtained at discrete time-instants, either by a GPS-device, or by a roadside sensor. Each of them entails a specific kind of uncertainty for the location-in-time data, however, we demonstrated that when combining the values from the two sources it turns out that "two uncertainties are better than one". In other words, integrating/fusing the data from both sources narrows the possible whereabouts when compared to each individual location data source. We analyzed the details of the FB model, and its impact on the lane-crossing query and range query, and conducted a collection of experiments to compare the overall performance between GPS-based bead model and FB model. We demonstrated that, by accepting a small amount overhead in the processing time, our FB model reduces the number of false positives.

There are a few directions that we plan to pursue in the near future. Firstly, we would like to investigate the impact of incorporating other types of sensors and location sources – e.g., the ones obtained via cellular networks [**40**] or indoor-localization – and develop a formal model capable of multisensor fusion [**49**]. A particular challenge in these settings is that different data sources may have different horizons of spatial and temporal validity. Our second extension is to consider the processing of other popular spatio-temporal queries (e.g., Nearest Neighbor) under the new model of location uncertainty. Yet another avenue is to extend the model/formalism so that it captures the uncertainty/imprecision in the very samples [**79**] as well as the possibility of accelerated motion [**53**]. Lastly, we are also planning to investigate the impact that incorporating other kinds of semantic information in the model – both in the representation of the trajectories [**74**], e.g., type of a vehicle (for fuel consumption, size, etc.), as well as the type of roads [**62**] – can have on the processing of spatio-temporal queries.

CHAPTER 3

Multi-lane Speed Clusters Mining

We now turn the attention to the problem of speed-based clustering of trajectories [**92, 30, 36, 47, 66, 91**] moving along road networks, with a specific twist of incorporating the multi-lane awareness. The main motivation for the problem(s) addressed is that without properly considering the fact that there are more than one lane along segments of the road network, the route-planning algorithms will generate trips with inaccurate duration. As our experiments have demonstrated, we can indeed improve the accuracy of the trip-duration with our methodologies.

3.1. Preliminaries

We now present a brief overview of the related background and introduce the basic terminology.

Traditionally, in MOD [**38**] the motion of an object with a distinct ID (oID) is represented as a *trajectory* $Tr_{oID} = [p_1, p_2...p_n]$, where each point p_i is a triplet $p_i = (x_i, y_i, t_i)$; t_i being the time that the object was at location (x_i, y_i).

A *road segment* r is a octuple $r = (r_{ID}, r_{Dir}, r_s, r_e, r_{type}, r_{length}, r_{speed}, r_{lane})$, where: r_{ID} is its unique identifier; r_{Dir} is a binary value indicating whether r is one-way or two-way segment; r_s and r_e are k-tuples (k = number of lanes) representing the starting and ending points of each lane (centroids); r_{type} indicates the type of the road to which the segment belongs (e.g., urban, rural, etc...); r_{length} is its length; r_{speed} is the maximum speed; and

r_{lane} is an integer specifying the number of lanes in each direction. A *road network* is an (augmented) graph $G_{RN} = (V_{RN}, E_{RN})$ where V_{RN} is the set of nodes representing the terminal points of road segments, and E_{RN} is the collection of road segments.

Table 3.1. Lane Width for Different Types of Road

Type of Roadway:	Rural	Urban
Freeway	12ft	12ft
Ramps (1-lane)	12-30ft	12-30ft
Arterial	11-12ft	10-12ft
Local	9-12ft	9-12 ft

For types of roads and the width of the lanes, we assume the classification proposed by the FHWA (Federal Highway Administration) of the US Department of Transportation [**95**] illustrated in Table 3.1, noting that the width is often associated with the maximum prescribed speed limit. Traffic-stream studies use different measures to characterize motion along road segments [**31**], often coupled with the available technology. For example, inductive sensors are good for estimating the flow, however, they cannot characterize the speed. On-board GPS devices are good at obtaining an average speed of individual moving objects, however, they are error-prone in terms of location, and cannot capture fluctuations in-between samples. We assume that motion-relevant data is obtained from (a sequence of) GPS points.

In information theory, a classical measure of information in a stochastic setting is the *Shannon Information* [**41**]. The Shannon Information $S_P(A)$ (also called the surprisal, or self-information) of a probability distribution P for an event A is $S_P(A) = -log_2 P(A)$. The *information entropy* (also called Shannon Entropy) is the expected value of the Shannon information [**89**].

3.1.1. Path Based Map-matching

Map-matching algorithms use information generated from positioning technologies and supplement with data from a high resolution spatial road-network map to provide an enhanced positioning output. It identifies the the correct road on which vehicles travel and determines vehicles' location on that segment [**81**]. Map-matching approaches can be generally categorised into four groups: geometric, topological, probabilistic, and other advanced techniques [**81**].

In this chapter, we apply and implement a path based map-matching algorithm that uses a Hidden Markov Model (HMM) to find the most likely road route [**71**]. Compared with the traditional point-based map-matching algorithm that only utilizes the geometric information from GPS points, the path based map-matching take the connectivity relationship between consecutive GPS points into consideration. It also uses Viterbi algorithm to compute the global optimal path.

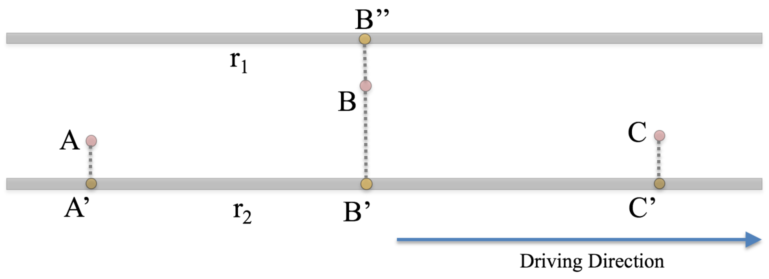


Figure 3.1. Path based map-matching

Figure 3.1 is a zoom-in view of the highway road network, which is the environment we developed for experiment. There are two parallel, directed edges r_1 and r_2 representing two directed road segments of the highway. One car is driving counterclockwise on r_2, forming a trajectory $[A, B, C]$. The correct map-matching results are $[A', B', C']$. When applying point-based map-matching algorithm, that matches GPS points to line segments with smallest matching distance [**126**], point B is map-matched to B'' due to the GPS noise and the factor that point B is closer to road segment r_1. However, trajectory $[A', B'', C']$ is invalid since a jump between r_1 and r_2 is not allowed. Path based map-matching incorporate geometric and topological information between every two consecutive GPS points, and is able to correct the map-matching result for point B from B'' to B'.

3.1.2. Partitioning

The philosophy of clustering traffic speed data is to group those GPS points that are spatially and temporally close to each other *and* with similar speed. The Unit Cell (UC) is defined as [**123**]:

Definition 2. (*Unit Cell*): A Unit Cell $UC_{kl} = (\Delta^S_{kl}, \Delta^T_{kl}, V_{kl}, D_{kl})$ in the l^{th} lane of a given road segment is the minimal partition in spatial and temporal dimension, characterized by a spatial range $\Delta^S_{kl} = d^+_{kl} - d^-_{kl}$, temporal interval $\Delta^T_{kl} = t^+_{kl} - t^-_{kl}$, and a set of trajectories $D_{kl} = [Tr_1, Tr_2, ...Tr_n]$ that belong to it. The set D_{kl} determines the speed-value V_{kl} associated with UC_{kl}

We note that the spatial range uses only "1D interval" – i.e., $d^+_{kl} - d^-_{kl}$ because the "conventional" 2-D space is constrained to 1-D along the driving direction, representing the distance(s) from starting point of the road segment (for the corresponding lane) until the beginning of the k-th unit cell (and the width is pre-determined by the road-type).

It is possible that multiple UCs share the similar speed/speed cluster(if we regard all GPS points within one UC as a single cluster). A Merging Cell (MC) is the combination of two or more UCs.

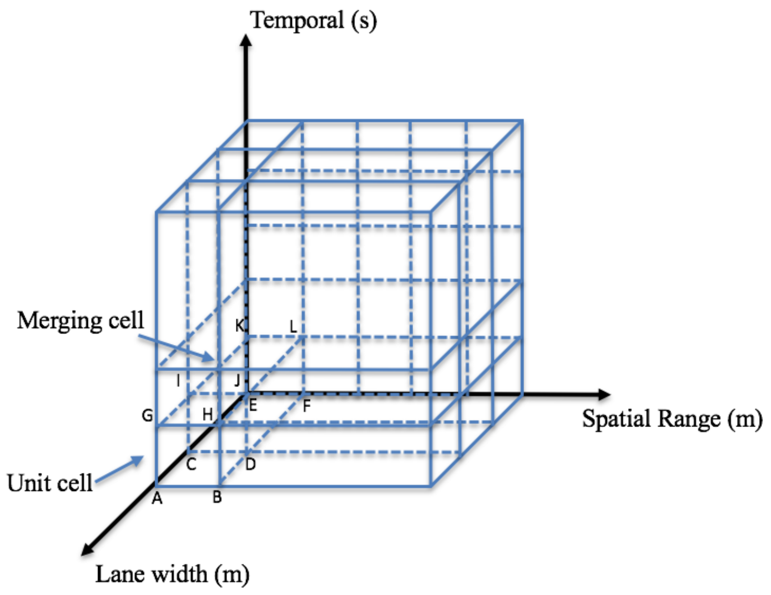


Figure 3.2. Unit cell and Merging cell

Definition 3. (*Merging Cell*): A Merging Cell (MC) is a union of multiple neighboring unit cells $MC_j = UC_1 \cup UC_2 \cup ... \cup UC_n$. Its spatial range is defined as $R_{MC} = \cup_i \Delta_i^S$ and its temporal interval $T_{MC} = \cup_i \Delta_i^T$.

The merging process follows certain criteria and procedures. We introduce two cluster mining approaches. In Section 3.2, we will discuss agglomerative merging method according to speed threshold; in section 3.3 and 3.4, a probabilistic speed profiling with a new distance measurement and merging algorithm will be proposed.

3.2. Agglomerative Speed Cluster Mining

Partition-and-merge framework for clustering trajectories has been proposed in [**60**] – however, the work did not consider the time-dimension (i.e., the speed) and was dealing with free 2D motion, not constrained to road networks. Models have been proposed based on trajectories' geographical information (including moving objects' heading and trajectory density) and semantic information. Compared with conventional trajectory clustering framework, in our problem of mining traffic speed clusters, GPS sample points are constrained by road-network. However, map-matching with uncertain location poses other challenges.

The agglomerative mining is a bottom-up approach. After partitioning the spatial-temporal space into many UCs, the criteria that we require for merging two neighboring UCs, inspired by [**21, 60**], are:

(1) $V_{(k+1)l} - V_{kl} \leq \delta''_v$ – i.e., the speed-values in the cells are close enough to each other, and

(2) $||D_{(k+1)l}| - |D_{kl}|| \leq \tau$ – i.e., the cells need to have close enough number of trajectories in their support-set.

An illustration of a merging cell MC formed by two neighboring UCs is shown in figure 3.6, in the (*space, time, lane#*) dimensionality. We constrain the union to consist of neighboring UC's in order to keep the spatial and temporal continuity. The first observation that, depending on the merging order chosen, a given collection of *UCs* need not yield a unique (collection of) *MCs*. A slight generalization of the scenario illustrated in Figure 3.6 can easily demonstrate that a particular cell can participate in as many as six different mergings, provided that there are ≥ 3 lanes. One can envision cells as being nodes in a graph and edges existing between neighboring cells. Upon merging, two nodes coalesce into one – and, to select a criterion for merging, a priority needs to be assigned among the adjacent vertices. Figure 3.6 illustrates merging of two neighboring edges from two lanes, sharing the same time-interval and distance from the start of the road-segment. However, in the current implementation, this is the 3rd criterion:

(3.1) First we check whether cells can be merged along the spatial dimension within the same lane.

(3.2) If not, we check next whether two cells with the same spatial extent can be merged along the temporal dimension.

(3.3) Lastly, we check whether two cells can be merged along neighboring lanes.

For every iteration of merging process there are three basic steps that are followed when processing each node.

I If there are no neighbors along the merging direction, the current cell is skipped and marked as visited.

II We check the cross-section coordinates between current cell and its neighbor in the order of preferences of merging directions. If cross-section coordinates are aligned and they satisfy the merging criteria, a merged cell is formed to replace them. One example is shown in figure 3.6–I. When we merge UC_3 with UC_4 along lane width dimension, their cross-section are aligned and a new MC_3 is formed. If we try to merge MC_3 with UC_5 along spatial range dimension in figure 3.6–II, their cross-section cannot match, and the cell will be skipped.

III Newly merged cell inherits all the neighboring relationship from merged cells. At the same time, all neighbors of two merged cells update their neighbor lists, by replacing original cells with the new one.

Speed Cluster: When the merging process for a particular (unit or merged) cell can no longer continue, we call that cell a *Speed Clusters* (SC)

Algorithm 2 formalizes the above description.

Assume that there are a total of n GPS points in the database, and (on the average) a road segment is composed of $O(K)$ spatial intervals in each lane and $O(M)$ intervals in the temporal dimension, defining the unit cells. Under a uniform distribution, each unit cell will consist of $O(n/(KM))$ GPS points from various trajectories. Calculating the merged cells

Algorithm 2 Cluster Mining (CellSets, DirectionSets)

```
1: ClusterID = NextID(NULL);
2: for Merging Direction MD IN DirectionList do
3:     while Cell in CellSets is unvisited do
4:         Neighbor = Cell.Neighbors(i) along MD;
5:         if Neighbor.size() == 0 then
6:             Cell.visited = True;
7:             Continue;
8:         else if Align(Cell, Neighbor) == True AND CanMerge(Cell, Neighbor) == True then
9:             NewCell = merge(Cell, Neighbor);
10:            NewCell.Neighbors = mergeNeighbor(Cell, Neighbor);
11:            updateNeighbor(Cell, Neighbor, NewCell);
12:            delete Cell, Neighbor;
13:        else
14:            Cell.visited = True;
15:        end if
16:    end while
17:    CellSets.visited = False;
18: end for
19: Return CellSets;
```

and clusters along a road segment can have an upper bound of $O((KM)^2)$ provided each one of the $K \cdot M$ cells is taken as a starting point to obtain the best possible clustering in terms of the minimal final number of clusters.

The agglomerative mining is a simple but effective clustering approach. Its benefit of reducing minimal travel time will be discussed in the experiment section. However, it is not adaptive to the errors incurred by GPS uncertainties. During the lane labeling process, large GPS errors will incur potential mislabeling, which later lead to noise points in the merging phase. One possible direction is to soft allocate the weight of GPS points to different lanes based on certain probabilities.

3.3. Probabilistic GPS Model and Speed Profile

GPS devices yield measurement error associated with each GPS-based determined location, even more so for crowd-sourced GPS data collected from potable devices. Due to the constraint of device size and cost, average horizontal errors from consumer-grade GPS receiver range from few meters to tens of meters [**122**]. Thus, deterministic lane level computations based on GPS probe data are ambiguous, so much so that the position may yield a different lane. In this section, we first introduce a probabilistic model to describe the location whereabouts for GPS points, followed by a definition of speed profile for every UC.

3.3.1. Probabilistic GPS Weight

The uncertain disk model [**102**] is the most naive one for uncertain location data, assuming uniform distribution. Let $D_p(x, y, t, r)$ denote the disk centered at point $P(x, y, t)$ with radius r, and A_i denote the area of lane i, which is a rectangle shape area. The probability of a GPS point located within a certain lane can be estimated by: $P_{lane} = \frac{D_p(x,y,t,r) \cap A}{D_p(x,y,t,r)}$. More sophisticated models describe the GPS data as a zero-mean Gaussian model [**71, 20**]. In this dissertation, we do not consider the GPS errors along vertical axis, thus, measurements from GPS receivers follow 2D Gaussian model. Given a GPS point $P_i(x_i, y_i, t_i)$, the probability density function (pdf) is:

$$f(x, y) = \frac{1}{\pi \sigma_x \sigma_y} exp(-(\frac{(x - x_i)^2}{2\sigma_x^2} + \frac{(y - y_i)^2}{2\sigma_y^2})) \tag{3.1}$$

The 2D Gaussian model is illustrated in Figure 3.3, with lane-width of 5m and spatial range Δ_{kl}^S is 5m as well. The location pdf for GPS point $(12, 12, 0)$ spans over multiple UCs

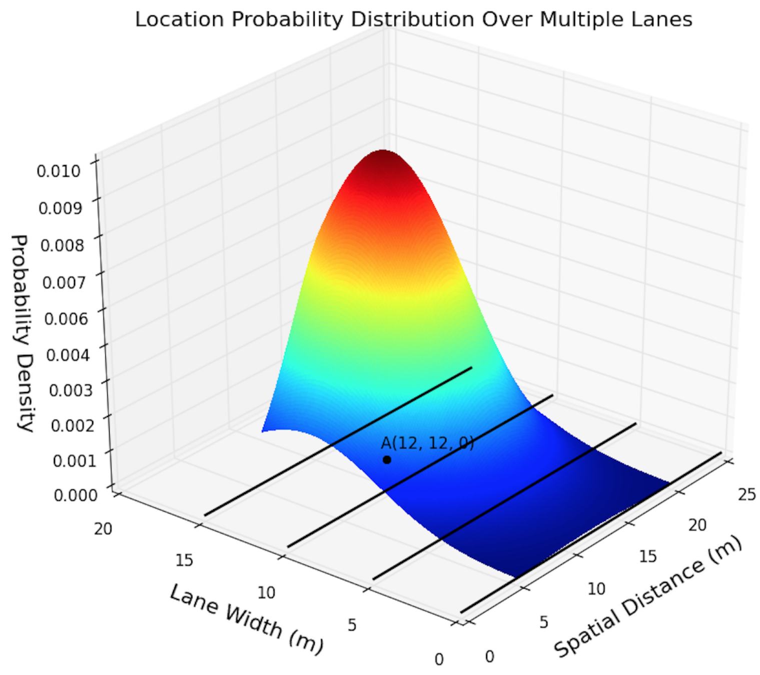


Figure 3.3. GPS point location probability distribution

and the probability of the GPS point being located into one UC is the integral of density function 3.1 over the spatial range within the lane width. Given $UC_0(\Delta^S_{k0}, \Delta^T_{k0}, V_{k0}, D_{k0})$ with spatial range $[x_{uc}, x_{uc} + \Delta^S_{k0}]$ and lane width $[y_{uc}, y_{uc} + width]$, the probability of P_i being inside UC_0 is:

(3.2)
$$P_i = \int_{x_{uc}}^{x_{uc}+\Delta^S_{k0}} \int_{y_{uc}}^{y_{uc}+width} f(x, y)dydx$$

Definition 4. GPS Contribution C_{GPS}: A GPS point G contributes to a Unit Cell UC when the probability P for G being located in UC is greater than ω – a threshold for the minimum probability value.

We augment each trajectory $Tr_{o_{ID}} = [p_1, p_2...p_n]$ within a UC by a GPS contribution, thereby making each p_i a quadruplet $p_i = (x_i, y_i, t_i, C_{GPSi})$.

3.3.2. Speed Profile

Given a UC with a set of augmented supporting trajectories, we apply a discretized histogram to estimate the traffic distribution within each UC. Given a bin size ϕ and number of bins n, instead of counting, the frequency for each bucket is the aggregation of contribution from every GPS point, reflecting a weighted sum for different points. The number of bins determines the level of granularity of the speed profile. We note that using only one bin is equivalent to a single average speed while too many bins may incur computational overhead. For a set of GPS points S with speed ranges within $(i * \phi,\ (i + 1) * \phi)$, the total GPS contribution W is:

$$W(i) = \sum_{s} C_i \tag{3.3}$$

The discrete pdf for the speed within range $(i * \phi,\ (i + 1) * \phi)$, which we call a *Speed Profile*, is:

$$pdf_v(v \in (i * \phi, (i + 1) * \phi)) = Pr(i) = \frac{W(i)}{\sum_1^n W(i)} \tag{3.4}$$

An example of a speed profile is shown in Figure 3.4, where ϕ is set to be 5km/h. There is one large peak at 80km/h, representing majority of GPS contributions; and a small peak near 20km/h, due to the contributions from slow lanes.

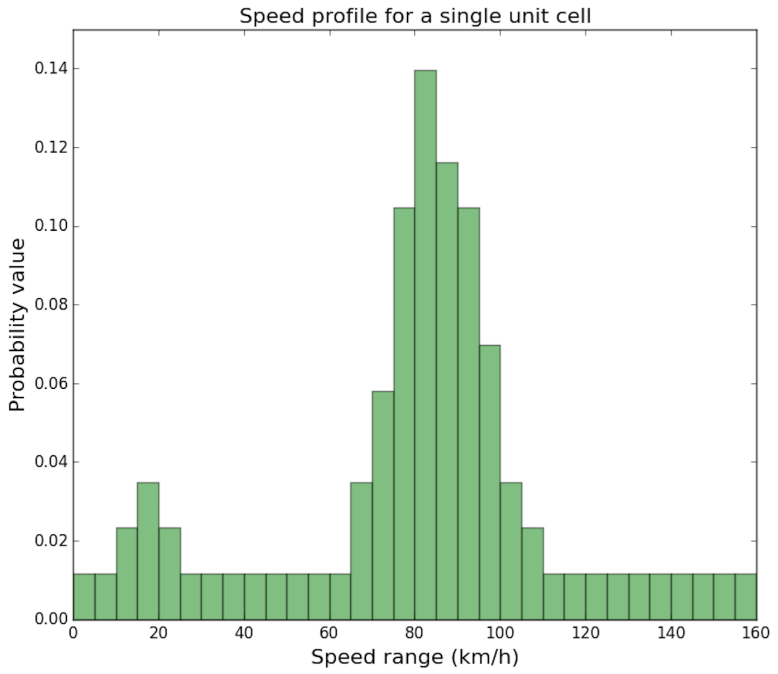


Figure 3.4. Speed profile for a unit cell

3.3.3. Merging Multiple Speed Profiles

Merging multiple UCs is the process to consolidate data from multiple sources, and we assume that the sources for GPS data within each UC are independent, because each UC is unique in spatial-temporal space and GPS measurements are independent. There are many approaches to consolidate independent data sources [**42**], like *averaging the probabilities*, and *averaging the data*, however, they have disadvantages: either do not take the differences of variances into consideration, or require averaging of dissimilar data.

Conflation is a method for consolidating a finite number of probability distributions $P1, ..., Pn$ into a single probability distribution $Q = Q(P_1, ..., P_n)$ [**41**], denoted by $\&(P_1, ..., P_n)$. Given multiple UCs $(UC_1, UC_2 ... UC_m)$ with respective speed profile $(pdf_{v1}, pdf_{v2} ... pdf_{vm})$, the merged probability distribution obtained via conflation is:

$$pdf_v^{MC}(v \in (i * \phi, (i+1) * \phi)) = Pr(i) = \frac{\prod_1^m pdf_v(i)}{\sum_{y \in n} \prod_1^m pdf_v(y)} \tag{3.5}$$

It has none of the disadvantages of the two averaging methods described above and has many advantages and important properties [**42**]:

(1) Conflation is commutative and associative:

$$\&(pdf_1, pdf_2) = \&(pdf_2, pdf_1) \text{ and}$$
$$\&(\&(pdf_1, pdf_2), pdf_3) = \&(pdf_1, \&(pdf_2, pdf_3))$$

(2) Conflation is iterative:

$$\&(pdf_1, pdf_2, pdf_3) = \&(\&(pdf_1, pdf_2), pdf_3)$$

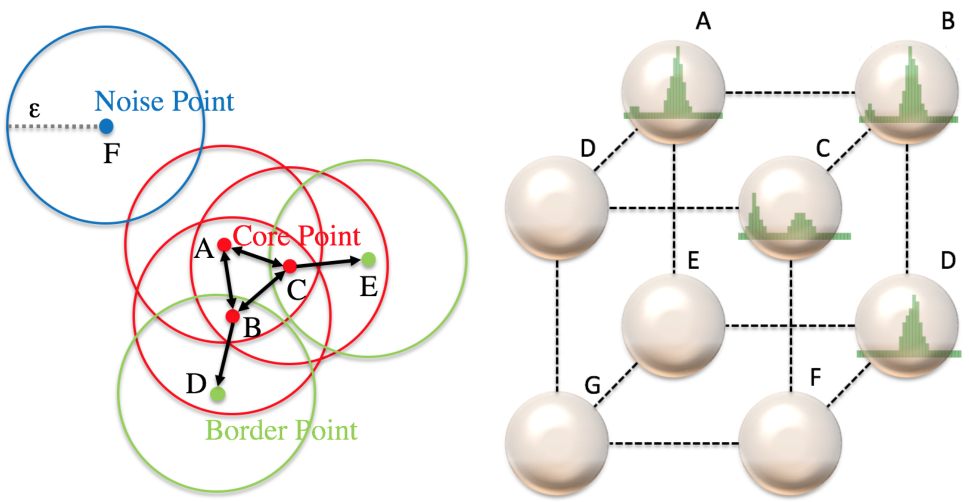
(3) Conflation minimizes the loss of Shannon information: If pdf_1 and pdf_2 are independent probability distributions, then the conflation $\&(pdf_1, pdf_2)$ of pdf_1 and pdf_2 is the unique probability distribution that minimizes, over all events, the maximum loss of Shannon information in replacing the pair pdf_1, pdf_2 by a merged distribution pdf^{MC} [**42**].

Properties (1) and (2) ensure that the conflation method can be used when merging UCs in any order and sequences, whereas (3) implies that conflation is compatible with entropy related measurement – i.e., given a threshold for the differences between two distributions in the information space, merged distribution using conflation minimizes the loss of Shannon information.

3.4. Speed Cluster Mining

After finalizing the pre-processing of the uncertain GPS data and calculating speed profile for individual UCs, we now proceed with the mining steps – i.e., detecting the speed clusters in multi-lane settings.

We note that in our previous work [**123**] we proposed a sweep line based method for merging neighboring UCs by scanning along spatial, temporal and lane dimensions. While the algorithm proposed in [**123**] is effective in terms of reducing the number of UCs via merging, we observed that different merging sequences will generate different clustering results. To alleviate this phenomenon, in the rest of this section, we present an improved merging algorithm inspired by density based clustering.



(a) Traditional DBSCAN (b) UC as "dummy point"

Figure 3.5. DBSCAN inspired merging

3.4.1. Probabilistic Distance Measurement

The most important criterion for merging two UCs is the level of similarity between them, in terms of the respective speed profiles. In the previous work [**123**], we used a simple method which compared the average speeds and the number of trajectories between two UCs with corresponding thresholds. However, in this work we use the speed profile as a more comprehensive description for drivers' behavior within certain UC and, given that we are catering to the fact of uncertainty of the locations' values, we decide whether to merge two UCs or not by calculating the distance between the respective discrete probability distributions. However, the distance between two distributions cannot be easily captured through geometric distance. Typically, in information theory, an uncertain object is treated as a random variable following a particular probability distribution. One popular measure for calculating the distance between two pdf's is the Kullback-Leibler divergence (also called information gain) – essentially, a measure of the difference between two probability distributions [**64**]. In our settings, given two speed profiles described with the respective pdf_i and pdf_j, the Kullback-Leibler divergence from pdf_i to pdf_j – denoted $D(pdf_i||pdf_j)$, describes the amount of information loss when pdf_i is used to estimate pdf_j. The equation defining the Kullback-Leibler divergence is:

$$D(pdf_i||pdf_j) = \sum_v pdf_i(v) log \frac{pdf_i(v)}{pdf_j(v)} \tag{3.6}$$

However, one specific property of the Kullback-Leibler divergence is its asymmetry, which is fine in many applications settings that rely on Bayesian inference. Contrary to this, in our settings, we would like to have the merging of UCs to be an undirected process that can start from any UC – and this makes the asymmetry an undesirable property.

Another method to measure the similarity between two probability distributions is the Jensen-Shannon divergence (JSD) (a.k.a. information radius) [**64**], which is commonly used in clustering probability distributions. It is based on the Kullback-Leibler divergence with the notable properties that it is symmetric, always a finite value and the square root is a metric. The Jensen-Shannon divergence between two speed profiles pdf_i and pdf_j is denoted as JSD_{ij}.

$$JSD_{ij} = \frac{1}{2}D(pdf_i || \frac{pdf_i + pdf_j}{2}) + \frac{1}{2}D(pdf_j || \frac{pdf_i + pdf_j}{2}) \tag{3.7}$$

In this work, we adopt the Jensen-Shannon divergence as the distance function between two UCs.

3.4.2. Mining Speed Clusters

The traditional DBSCAN [**29**] accepts a radius value ε based on a (user defined) distance measure, and a value MinPts for the number of minimal points that should occur within Eps radius. A simple illustration is shown in Figure 3.5a, where $\varepsilon = 2$ and the Euclidean distance is used as a distance so that, upon comparison with ε, one can determine whether two points are connected. The points A, B and C in Figure 3.5a are considered core points because the discs with radii ε and centered at each of them, contain at least 2 neighboring points and they belong to the same cluster. Points D and E are not core points. However, they are reachable from A through other core points – and, consequently, they belong to the

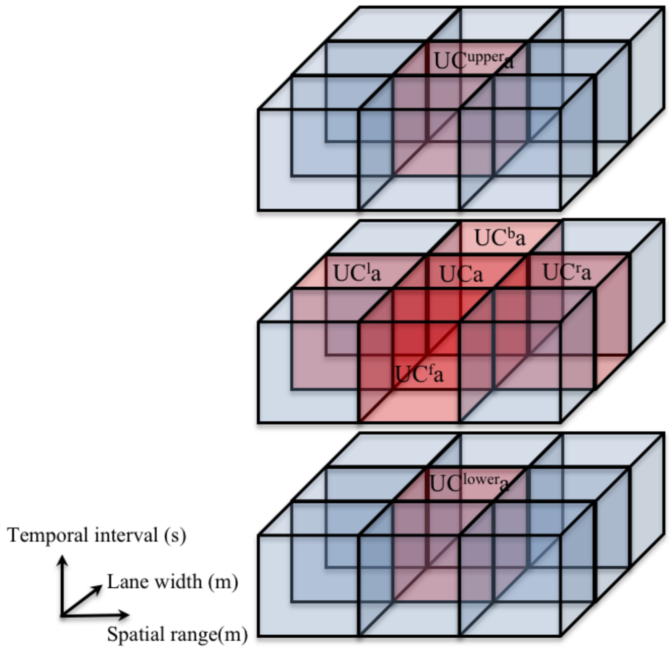


Figure 3.6. One UC has six neighboring candidates

same cluster as well. The point F is a separated/isolated noise point that is neither a core point nor density reachable [**29**].

Inspired by DBSCAN, we propose Traffic-Density-Merging (TDM), a density based clustering algorithm. When mining the speed cluster based on fine-grained UCs, each UC is treated as an “artificial point”, as shown in Figure 3.5b. They are well-identified in terms of their organization in the spatio-temporal 3D space and the Euclidean distances between two spatially-consecutive points are the same. The candidate-neighbors of a particular such “artificial point” are defined as UCs which have representative “artificial points” that are directly connected to the one representing the particular UC. Thus, each “artificial point” can have at most 6 candidate neighbors, as shown in Figure 3.6. As a specific example, in Figure 3.5b, the “point” B has 3 candidate neighbors. However, only “point” A and D have similar speed profiles compared with “point” B, whose information-distance *JSD* is smaller than the (assumed) threshold. Therefore, B has two neighbors and is a core point (assuming MinPts = 2). A, B and D can therefore be merged as a speed cluster. This process is analogous to the one occurring in the traditional DBSCAN – except, instead of the Euclidean distance, JSD is used to calculate the respective distances in the "information space" and a corresponding threshold λ is defined to determine whether two neighboring UCs belong to the same speed cluster.

Proceeding formally, and in the spirit of [**29**], the density based speed cluster is defined as follows:

Definition 5. (*Density-based speed cluster*): A cluster C is a non-empty subset of UCs satisfying the following "maximality" and "connectivity" requirements:

(1) $\forall p, q$: if $q \in C$ and p is density-reachable from q with respect to (λ) and MinPts, then $p \in C$.

(2) $\forall p, q \in C$: p is density-connected to q with respect to (λ) and MinPts.

The TDM starts with an arbitrary UC_a. If it has been visited, the iteration breaks and proceeds to the next UC. Otherwise, we call the *Neighbor-Query* to retrieve its qualified neighbors. As shown in Figure 3.6, UC_a has six candidate neighbors. Those candidate neighbors that have JSD less than λ with the currently considered UC (UC_a) become neighbors that are density reachable from UC_a. In the case that the number of such neighbors is larger than MinPts, UC_a is a core UC and a cluster is identified. If the number of neighbors is less than MinPts, it cannot form an independent cluster, and we keep this UC as a separate one.

Algorithm 3 Traffic-Density-Merging (UCSets, MinPts, λ)

```
1: ClusterID = NextID(NULL);
2: for UC IN UCSets do
3:     if UC.visited == True then
4:         continue;
5:     end if
6:     UC.visited = True;
7:     Neighbors = Neighbor-Query(UC, UCSets, $\lambda$);
8:     if Neighbors.size() < MinPts then
9:         Point.CID = Separate;
10:    else
11:        Point.CID = ClusterID;
12:        Cluster = ExpandCluster(UCSets, Neighbors, UC, ClusterID, MinPts, $\lambda$);
13:        SpeedCluster.add(Cluster);
14:        ClusterID = NextID(ClusterID);
15:    end if
16: end for
17: Return SpeedCluster;
```

In Algorithm 3, *UCSets* is either the whole set of UCs on certain road segment or a proper subset of them. MinPts and λ are parameters that are provided as input to TDM and they can be determined analytically or through experiments, based on a particular scenario.

When a new/unvisted UC is identified as a core cell, a new cluster is generated. Following that, the function *ExpandCluster* is invoked, for which the pseudo-code is presented in Algorithm 4, in order to expand the cluster based on the current UC and its neighbors. The cluster expansion process is essentially a depth-first kind of a search. A stack is used to store all the seed UCs. If the current UC is unvisited, we retrieve its neighbors using *Neighbor-Query* in the same way as described above. The qualified UCs are pushed onto the stack under the condition that the current UC is identified as a core cell. If the current UC has not been classified into any cluster or it is previously marked as *Separated*, we append it into the current cluster.

Algorithm 4 ExpandCluster (UCSets, Neighbors, UC, ClusterID, MinPts, λ)

```
1: Seeds = Neighbors;
2: Seeds.add(Point);
3: while Seeds.size() $> 0$ do
4:     CurrentUC = Seeds.first();
5:     Seeds.pop();
6:     if CurrentUC.visited != True then
7:         CurrentUC.visited = True;
8:         Cluster.add(CurrentUC);
9:         Cluster.MergeSpeedProfile(CurrentUC);
10:        NewNeighbors = Neighbor-Query(CurrentUC, UCSets, $\lambda$);
11:        if NewNeighbors.size() $>$ MinPts then
12:            Seeds.Append(NewNeighbors);
13:        end if
14:    end if
15:    if (CurrentUC.CID == NULL) OR (CurrentUC.CID == Isolated) then
16:        CurrentUC.CID = ClusterID;
17:    end if
18: end while
19: Return cluster;
```

Algorithm 5 Neighbor-Query (UC, UCSets, AverageSpeed, λ)

```
1: CandidateNeighbors = UCSets.search();
2: for Neighbor in CandidateNeighbors do
3:     if JSD(Neighbor, UC) $< \lambda$ then
4:         Neighbors.add(Neighbor);
5:     end if
6: end for
7: Return Neighbors;
```

If two clusters C_1 and C_2 are very close to each other (in JSD sense), there might be scenario that a given UC_i belongs to both clusters – which entails that such UC_i is on the boundary between C_1 and C_2. If this is the case, UC_i will be assigned to the first discovered cluster. In addition, we note that there will not be cases in which a particular cluster partially intersects or if fully contained by another cluster (otherwise the two clusters will be merged).

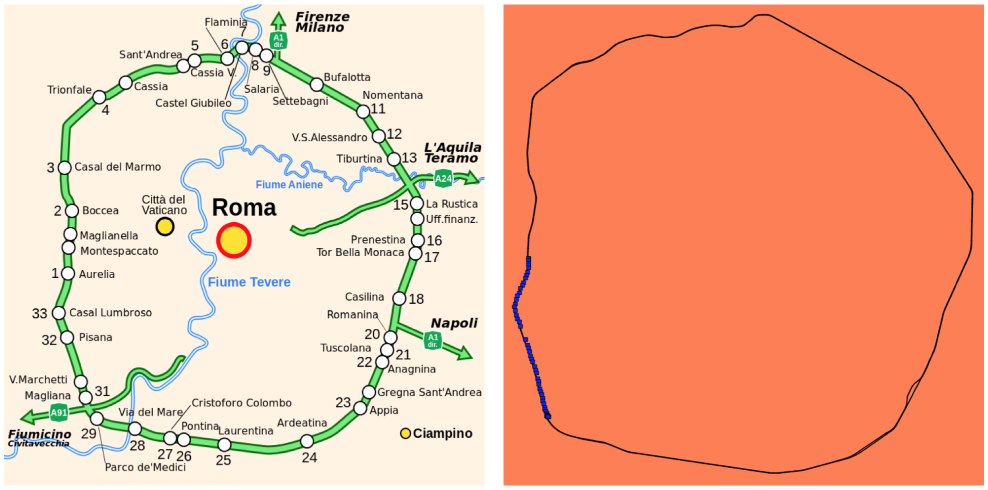
Remark: The *Neighbor Query* can be supported efficiently by spatial access method like R*-trees [**7**], which is often available in spatial database system – however, the issue of indexing is beyond the scope of this dissertation and we defer it for our future work.

Given the results for the original DBSCAN, we note that the access time for a collection of n UCs is $O(logn)$. As we discover new clusters, for each of the n points there is at most one invocation of the Neighbor-Query to be processed. Therefore the time complexity for TDM is bounded by $O(nlogn)$ – which, once again, is the time complexity of the traditional DBSCAN (cf. [**29**]) since we retain the general framework for density based searching.

3.5. Experimental Observations

We evaluate both agglomerative mining method and Traffic-Density-Merging. For our experiments, we used a data obtained from the Grande Raccordo Anulare (GRA) motorway. It is a toll-free, ring-shaped orbital motorway that encircles Rome, as illustrated in Figure 3.7a, and it is considered to be one of the most frequently used roads with heavy traffic for the most of the day. Our experiments are based on a dataset contains GPS traces of 320 taxi cabs in Rome, collected over 30 days – from February 1, until March 2 of 2014 [**8**]. The cardinality of the dataset is 8,368,858 points.

To evaluate the benefit of our proposed speed cluster mining algorithm, an augmented road network of GRA is built based on OpenStreetMap data, where road segments are augmented with three lanes. Figure 3.7b is a simple visualization of such augmented road networks. Blue dots represent GPS points that originated from two vehicles' trajectories driving along the GRA.



(a) The GRA (b) Experiment environment

Figure 3.7. The Grande Raccordo Anulare and experiment environment

3.5.1. Evaluation for Agglomerative Mining

Baseline: The baseline approach we compared with is a traditional traffic speed estimation method (cf. Section 1), by calculating the average speed among all speed samples. In order to incorporate the variation along temporal dimension, we divide speed samples into 24 subsets, which represent 24 hours in a day. Average speed within each hour is calculated accordingly.

Evaluation: The data was divided into four folds according to sample time, each containing GPS points within one week. We used three weeks data to train our model, and we randomly pick trajectories from the remaining set to estimate the travel time, by using both baseline method and traffic speed clustering model.

The experiment was repeated 15 times and the average of all the runs is shown in Figure 3.9. The x-axis represents the aggregated travel distance, and y-axis stands for average estimated travel. On average, the travel time by using speed clustering method is reduced by 20%, compared with baseline approach. For the entire road networks, there are 8,328,960 UCs before merging, and 414,982 MCs after merging, which is a 95% reduction.

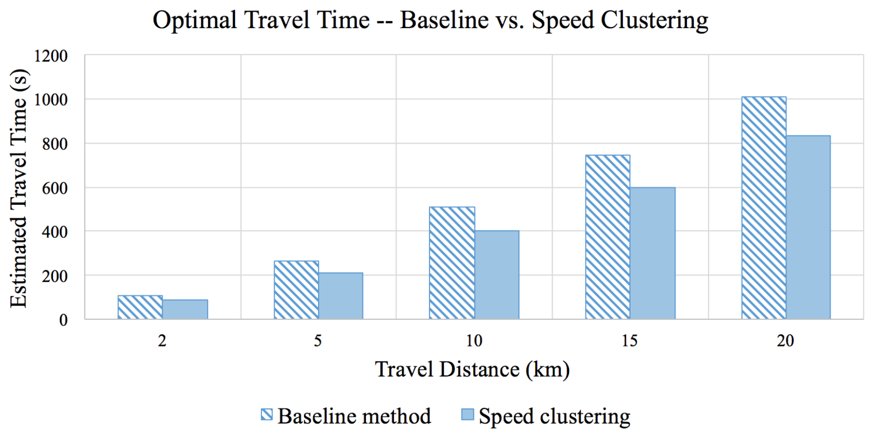


Figure 3.8. Travel time estimation using baseline approach and agglomerative method

3.5.2. Evaluation for Traffic-Density-Merging

The experimental evaluation for probabilistic speed profiling and Traffic-Density-Merging consists of two parts – training and validation; and there are two steps in the training process as well: (1) building speed profile for each UC, and (2) mining speed clusters. We first train the model from GPS traces collected from Rome taxi cabs. Subsequently, we validate our model by predicting the travel time using trained speed clusters.

The experiments were conducted on a MacOS machine with 2.7 GHz Intel CPU with 8GB 1867MHz DDR3 RAM and the implementation[1] was done in Python 2.7.

Parameter Estimation and System Implementation: Our probabilistic GPS uncertainty model calculating the GPS contribution for each UC requires two parameters σ_x and

[1]We note that the code and the datasets are publicly available at www.eecs.northwestern.edu/~bzv686.

σ_y. They are the values of the corresponding standard deviation of the Gaussion GPS noise in longitudinal and latitudinal directions. This parameter can be affected by the measurement devices and measurement environment. According to experiments conducted/reported in the related literature [**71**], we estimate the standard deviation of Gaussion GPS noise to be 5 meters.

In the Traffic-Density-Merging algorithm, there are two important parameters – MinPts, which determines whether a given UC is a "core point", and λ – which is the threshold to determine the neighboring relationship between two UCs. MinPts is an integer in the range $[1, 6]$ and λ is a real number from the interval $[0, 1]$.

There are many different ways of choosing data structure to implement our proposed traffic speed cluster mining algorithm. In our experiments, we used a simple scheme – i.e., we built a three dimensional matrix in spatial-temporal coordinates to index UCs on each road segments, which is similar to the structure shown in Figure 3.2. The respective dimensions in spatio-temporal coordinates are indexed to UCs that are stored in a key-value map. While the overall efficiency is not a topic of this work, we note that the structure used in this experimental setup allows for a fast query processing when inferring the traffic speed from trained speed clusters.

Estimated Travel Time Query: Our first set of experiments aims at illustrating how the speed clusters mining in the lane level granularity can provides a more accurate, yet compact description of the traffic distribution for road networks. Many applications could benefit from it – e.g., adaptive navigation, route planning and travel time prediction. In this experiment, we implemented the travel time prediction to demonstrate the advantage of our proposed method.

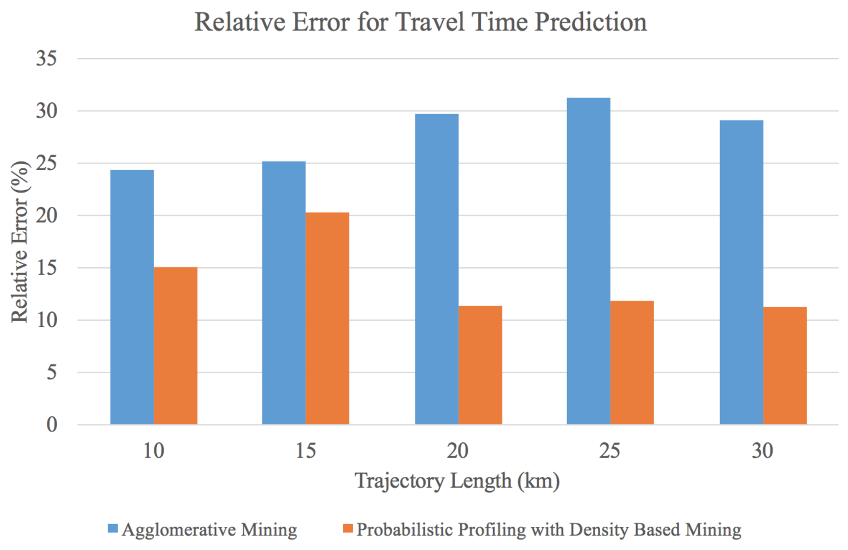


Figure 3.9. Relative error for predicted travel time

The Rome taxi dataset is divided into four folds according to sample time. Each of them contains GPS points within one week. We used three weeks data for training purpose, and the remaining one was used for validation.

GPS points in validation trajectories are assigned to corresponding lanes using the lane labeling process in [**123**]. Given a validation trajectory $Tr_{val} = [p_1, p_2 ... p_n]$, the traffic speed for certain GPS points $P_i = (x_i, y_i, t_i)$ can be inferred from the speed profile of the corresponding speed cluster. Thus, the predicted travel time $T_{predict} = \sum_1^{n-1} \frac{distance(P_i, P_{i+1})}{speed(x_i, y_i, t_i)}$. Since the ground true driving time $T_{true} = t_n - t_1$, we use relative prediction error $e = \frac{abs(T_{true} - T_{predict})}{T_{true}}$ to quantitatively measure the prediction power of the proposed model.

The baseline method we compared with is the agglomerative mining method we proposed previously [**123**]. It is a bottom-up clustering method with sweep line styled merging. The experimental results are shown in Figure 3.9. When the parameter MinPts is 3 and λ equals 0.1, the probabilistic speed profiling with density based merging method reduces prediction error by more than 20%. Compared with the simple agglomerative method, the new model describes the multi-lane traffic speed information with a probabilistic speed profile and is able to make a more accurate travel time prediction.

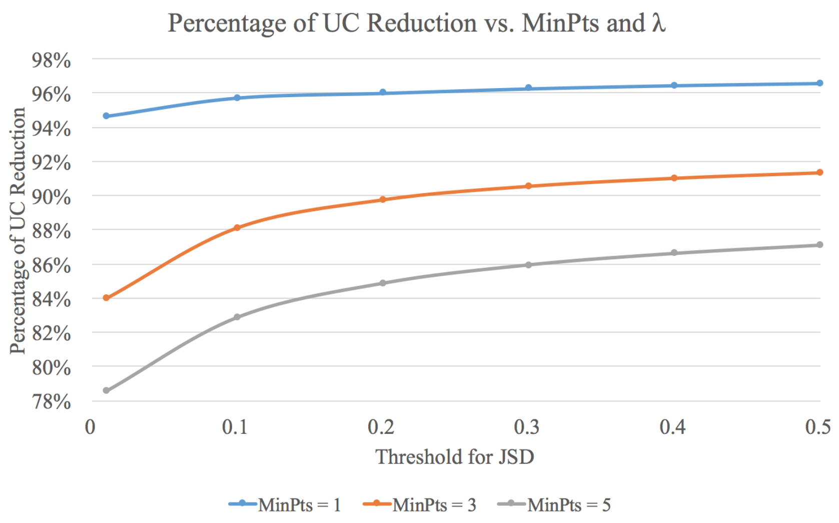


Figure 3.10. Percentage of UC reduction when changing parameter

UC Reduction: As described in Section 2.3, we partition the spatio-temporal space into fine-grained UCs. Since lots of them will share similar speed profiles, it is not necessary to store every UCs into the database. Therefore, merging is a beneficial operation in the cluster mining process. We re-iterate that the sweep line styled merging, proposed in [**123**], has three possible merging directions for each UC and each direction is processed sequentially. The disadvantage for this merging method is its instability — the merging results are affected by different merging sequences. The density-based merging algorithm proposed in this dissertation overcomes this issue. Due to the nature of depth first search within the cluster expanding process, the density-based merging is independent of merging direction and merging sequences.

Figure 3.10 shows the percentage of UC reduction with various parameter choices. The highest compression ratio reaches more then 96%, while the lowest one is still more than 70% when MinPts is 5 and λ equals 0.1. The larger λ values and smaller MinPts values (which correspond to lessening the constraints for merging neighboring UCs) will incur higher compression ratio.

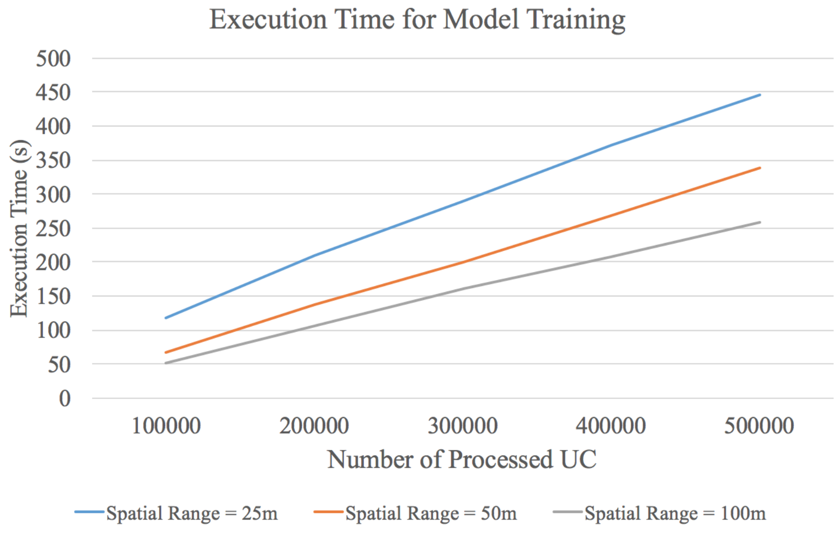


Figure 3.11. Execution time for speed cluster training

Training and Validation Time:In the last experiment that we report, we consider the respective execution times for the model training and validation. The system for this experiment is designed to run in an offline mode, where the speed clusters are mined from historical GPS data. As shown in Figure 3.11, the training process fo mining speed clusters takes relatively long time. We note that the training time is related not only to the size of the raw data, but also to the level of granularity of the partitioning. The smaller size of UC (i.e., more granular representation) yields a longer training time.

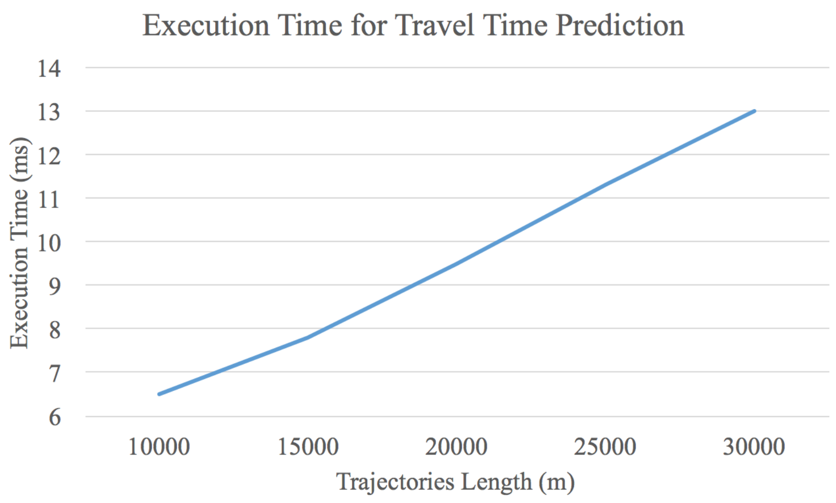


Figure 3.12. Execution time for travel time prediction

The main advantage for this offline trained model is that processing queries that depend on the traffic speed distribution is fast. When we validate the model by calculating predicted travel time, all queries are finished within a second. We note that this kind of an executional behavior is suitable for many OLAP kinds of applications.

3.6. Related Work

Lane level positioning, routing and navigation are correlated research areas with high societal impacts. There are two main categories of related works in this realm. The first one attempts to directly map-match GPS points to corresponding lanes [**25**]. This method usually requires the use of Differential Global Positioning System (DGPS) for data collection so that the GPS errors are smaller than the usual lane width. However, because of the high cost, DGPS has not been widely available in consumer grade mobile devices. Other researchers choose to pursue external calibration through computer visions [**111**], vehicle-to-vehicle (V2V) communication or vehicle-to-infrastructure (V2I) communication [**4, 26**].

These approaches require additional hardware or infrastructure and cannot be applied in large scale quickly.

Multiple models have been proposed to answer queries related to GPS uncertainties. From disk model of location uncertainty (yielding sheared cylinder model in spatio-temporal space) [**102**], through beads model [**100**], to adaptation of the bead model on road networks [**73**]. More recently, an attempt to combine heterogeneous location data sources in the context of multi-lane road networks, called fused bead model was presented in [**124**].

A complementary body of related works stems from the literature addressing problems related to trajectories clustering, for both online and offline settings. Various clustering algorithms and frameworks have been proposed, including regression [**30**], partitioning and grouping [**59**] and density based clustering [**92**]. However, most of these works are targeting the, so called, macroscopic model and focus on large scale pattern mining, which lead to application like popular region discovery, event detection or route analysis. In addition, very few of them combine the trajectories clustering techniques with the constraint of road networks and use it as a tool to analyze the traffic on the lane level granularity.

3.7. Concluding Remarks and Future Works

We proposed a methodology for mining speed clusters in multi-lane road networks, incorporating the uncertainty of the moving objects location to capture the GPS errors within the model. We proposed a basic agglomerative approach, a novel distance function and a variant of the DBSCAN algorithm for mining multi-lane speed clusters. We used the Rome taxi data to demonstrate that our proposed method yields both a more compact representation of the clusters, as well as a more accurate travel time calculation for trajectories.

There are several extensions to our work. Firstly, we plan to tackle several efficiency-related aspects – namely, data structures that will enable efficient storage and retrieval of the elementary UCs. Our next aim is to incorporate a few distinct contexts: (1) we would like to investigate the impact of changes in the type of the road (i.e., from 4 lanes expressway into a single lane local street); (2) we believe that the an attribute with a stronger impact may be the kind of a vehicle (e.g., passenger car vs. trucks); and (3) we plan investigate the impact of speed/travel-time clustering in the settings of multi-modal transportation. Our longer term vision is to develop a model that will balance the trade-offs between the precision of the clustering vs. the cost (both in terms of access as well as execution time), when multiple data sources can be combined – e.g., roadside sensors and cameras – with the GPS-based location data.

CHAPTER 4

Anomalous Traffic Speed Prediction

This Chapter presents the third major aspect of the research work conducted as part of this dissertation. Specifically, we aim at developing Artificial Neural Networks (ANN) based approach to predict the occurrence of speed anomalies in geographic area within urban settings. The main contribution is in identifying a distinct feature that, when incorporated in the ANN based model, will enable more accurate prediction on a smaller temporal scale.

4.1. Preliminaries

We now present a brief overview of the basic terminology, followed with a more formal presentation of the problem and introduction of the prediction framework.

Detailed representation of the traffic state in space and time allows a more detailed analysis of multiple aspects of traffic dynamics. We note that in most practical scenarios, data are available in the form of aggregated minute by minute data of speed and flow recorded by stationary detectors. In other words, real traffic data are only available for a small subset of locations and times, the full traffic state can only be reconstructed by spatiotemporal interpolation. An example of interpolating the traffic state in-between two discrete measurements is illustrated in Figure 4.1 (cf. [**104**]).

In this dissertation, we rely on traffic flow models [**11, 37, 107, 109**] which, in transportation literature are often categorized along different contexts – e.g., aggregation level,

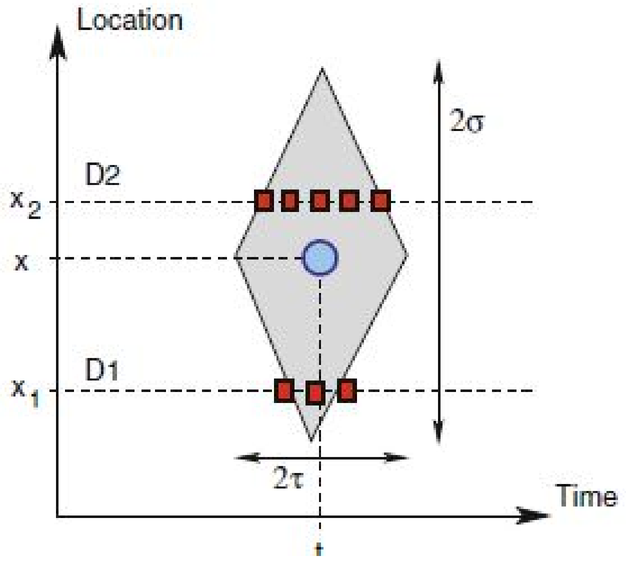
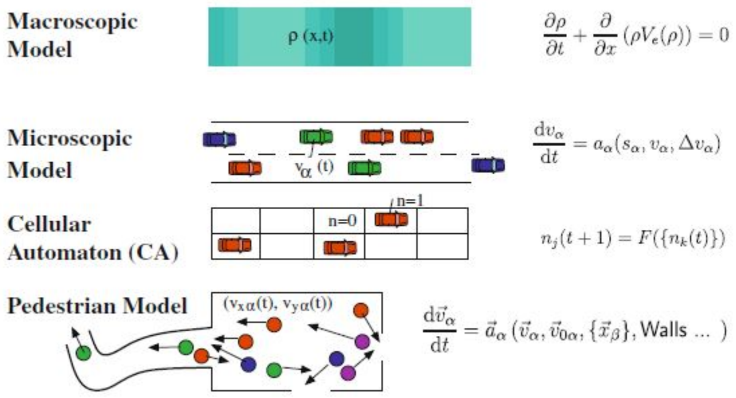


Figure 4.1. Traffic state are reconstructed by interpolation
mathematical structure, and conceptual aspects. For example, from the aggregation level perspective, the transportation literature distinguishes among (cf. [**104**]):

- Macroscopic models – where traffic flow is specified in an analogous manner to how physicists would specify the dynamics of liquids or gases in motion, using variables that capture locally aggregated quantities, e.g., the traffic density ($\rho(x,t)$), flow ($Q(x,t)$), mean speed ($V(x,t)$), etc.
- Microscopic models – (e.g., car-following models; cellular automata models) which describe individual "driver-vehicle particles" that collectively dictate the traffic flow. Microscopic models focus on impact that a reaction of an individual driver (accelerating, braking, lane-changing) can have on the surrounding traffic (and vice versa). The typical variables capturing the dynamics are individual vehicle positions ($x_\alpha(t)$), speed ($v_\alpha(t)$), and accelerations ($v^{'}(t)$).

- Mesoscopic models – which are often a "hybrid" between microscopic and macroscopic approaches. In other words, the parameters of a microscopic model may depend on macroscopic quantities such as traffic density or local speed and speed variance, and vice-versa: the dynamics of a macroscopic quantity (e.g., the number of vehicles in a traffic jam) is described in terms of microscopic stochastic rate equations for in- and out-"flowing" vehicles.



Macroscopic Model
ρ (x,t)
$\frac{\partial \rho}{\partial t} + \frac{\partial}{\partial x}(\rho V_e(\rho)) = 0$
Microscopic Model
$v_\alpha(t)$
$\frac{dv_\alpha}{dt} = a_\alpha(s_\alpha, v_\alpha, \Delta v_\alpha)$
Cellular Automaton (CA)
n=0
n=1
$n_j(t+1) = F(\{n_k(t)\})$
Pedestrian Model
$(v_{x\alpha}(t), v_{y\alpha}(t))$
$\frac{d\vec{v}_\alpha}{dt} = \vec{a}_\alpha(\vec{v}_\alpha, \vec{v}_{0\alpha}, \{\vec{x}_\beta\}, \text{Walls} \ldots)$

Figure 4.2. Scopes of transportation models

An illustration of relative scope of each of the commonly used models is shown in Figure 4.2 (cf. [**104**]). By its nature, the context of this part of our research most closely fits in-between the macroscopic and mesoscopic model – i.e., we do "expand" the spatial range of validity of our predictions from road segments to regions[1]. The data obtained from both roadside sensors and on-board GPS devices are what enables the study of different aspects of traffic characterizations (e.g., jams and their propagation). Many times[2] one can interchangeably use different quantities to describe the traffic state – e.g., flow vs. density (vs. (average) speed). However, in post-microscopic level, those quantities are typically related

[1]Defined in more detail in the sequel.
[2]The specific choice may be depending on the aggregation/granularity level used to model the reality.

– e.g., one can use $Q(x,t) = \rho(x,t) * V(x,t)$ to specify that (similarly to hydrodynamics) the traffic flow at a given (location, time) point, is a product of the traffic density and the (aggregated) speed.

We consider urban settings and we partition a given city into regions using an $I \times J$ grid map (c.f. [**125**]) based on the longitude and latitude values. We assume that each region is a square corresponding to a cell in the grid, with a given width w (resp. length $l = w$). Given the boundaries of the zone of interest (i.e., city) $(lat_{low}, lon_{low}, lat_{high}, lon_{high})$, we assume row-major ordering for enumerating the regions – i.e., $R_{i,j} = (lat_{low} + i \times w, lon_{low} + j \times w, lat_{low}(i+1) \times w, lon_{low}(j+1) \times w)$, where $0 \leq i \leq (lat_{high} - lat_{low})/\ w$, and $0 \leq j \leq (lon_{high} - lon_{low})/\ w$. From a different perspective, a region is a minimal spatial unit of demand-distribution. Similarly, a road segments is the minimal unit for (expressing the values for) traffic speed prediction.

We consider (time-stamped) *transport requests* posed by individual users. Each request, denoted T_{R_i}, is represented as a triplet $[u_i^{id}, d_i, p_i]$ – where u_i^{id} is a unique user-identifier; d_i is a triplet $(x_i,\ y_i,\ t_i)$, corresponding to the location of u_i^{id} at which the transport request was made, along with the respective time-stamp; and p_i denotes the *provider* of the request. We note that one can distinguish between the time-stamp when an order was placed, e.g., with a taxi calling center, and the time-stamp when the actual provider's vehicle arrived. Unless otherwise specified, in the rest of this dissertation the value of t_i in $[u_i^{id}, (x_i, y_i, t_i), p_i]$ is assumed to be the pick-up time, and we also assume that the location did not change between the request and the pick-up time. In addition, a collection of transportation requests can have different *modes* – e.g., driving (individual or service-provider vehicle), biking (individual or public-station bike), walking or taking a bus, train, flight, etc... – and the respective data can be gathered from corresponding (heterogeneous) sources like, for example:

- Activation of Uber App on the smart phone.
- Checking out a bike at a particular service station.
- Paying with a credit card when exiting a garage.

For a given transportation mode, we define *Modal mobility demand* at time-instant t and with a past duration window Δ as a set of all the transport requests for that mode:

$$MD^{mode}(t, \Delta) = \cup_j T_{R_j} \quad ((t - \Delta) < t_j \leq t)$$

.

Throughout this work we focus on predictions related to traffic speed within an urban area, thus, the transportation modes such as train rides, flights and cruises are not considered and are left for the future work.

Traffic Speed: For a given road segment r_j, we define its traffic speed $S_i^j = (s,\ t_i)$ be the average traffic flow speed within time interval (t_{i-1}, t_i), where s is the speed value and t_i is the time stamp.

As mentioned, traffic speeds are typically averaged over multiple data readings and aggregated up to a common time cadence. In this work, the traffic speed data are collected from traffic speed detector deployed in major arterial in the New York City [**103**], with five-minute time interval as the aggregation unit. For a given region R with spatial extent, we focus on detecting *anomalous events* – which is, time-intervals in which an occurrence of abnormality with respect to the (model-based) values of the variables that describe the state of the traffic in that region. More specifically, we combine individually-detected abnormalities (i.e., variation in speed/flow for certain number of individual participants) within a pre-defined time-interval.

In this dissertation, the spatial-temporal entries being tested for anomalous events are mobility demands $MD^{mode}(t, \Delta)$ at each time-instant.

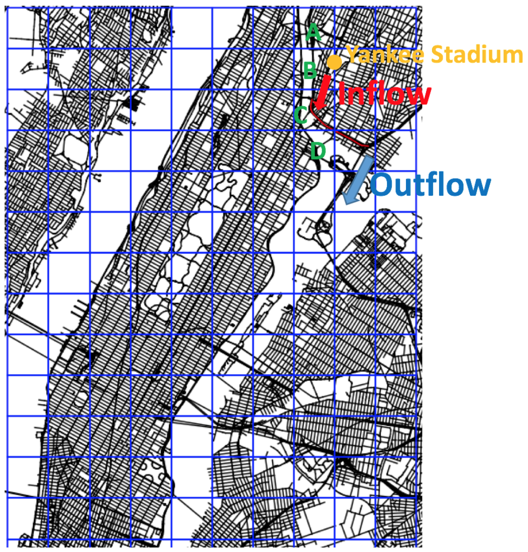


Figure 4.3. Region Segmentation

4.1.1. Problem Definition

Let TR be the data series on traffic speed for a certain road segment and D represent the mobility demand data in nearby regions, where TR_i^j denotes the traffic speed for road segment r_j at time t_i and D_i^j denotes the mobility demand in region R_j at time t_i. In addition, we extract *Demand Feature* from mobility demand data (cf. section 5) and denote it as DF. Assuming the current time t_c and prediction length l, the traditional traffic prediction problem in transportation research has the following setting:

Short-term Traffic Prediction: Given the historical traffic speed data TR within time interval $[0,\ t_c]$, develop a traffic prediction model $M(\cdot)$ that output the traffic speed at time t_{c+l}. $TR_{c+l} = M(TR)$.

When anomalous events (e.g. mobility volcano) occur, the traditional statistical model will suffer in this scenario since the additional traffic flows being injected into road networks affect the existing traffic patterns. To solve this challenge, additional features from mobility demands are proposed to be included into our prediction models.

Short-term Traffic Prediction with anomalies: Given the historical traffic speed data TR and demand features DF within time interval $[0,\ t_c]$, develop an extended traffic prediction model $EM(\cdot)$ that output the traffic speed at time t_{c+l}. $TR_{c+l} = EM(TR, DF)$.

4.1.2. Prediction Framework

In this chapter, we propose an adaptive prediction framework for short term traffic speed prediction, as illustrated in Figure 4.4. More specifically, we enhance the prediction accuracy when anomalous events occur, by utilizing mobility demand data from heterogeneous data sources.

The prediction framework can be separated into two parts, namely, batch training and online prediction. In the training stage, the road networks are first segmented and correlated regions are calculated for targeting road segment. After that, the historical mobility demands data and traffic speed data are preprocessed and stored when anomalous events are detected. Finally, we extract demand features from mobility demand data and train the anomalous traffic model with demand features and historical traffic speed. When conducting online traffic speed prediction, we first detect the anomalous events using real-time mobility demand data. If anomalies are detected, we predict the traffic speed with anomalous traffic model. Otherwise, the traffic speed is predicted with a conventional model.

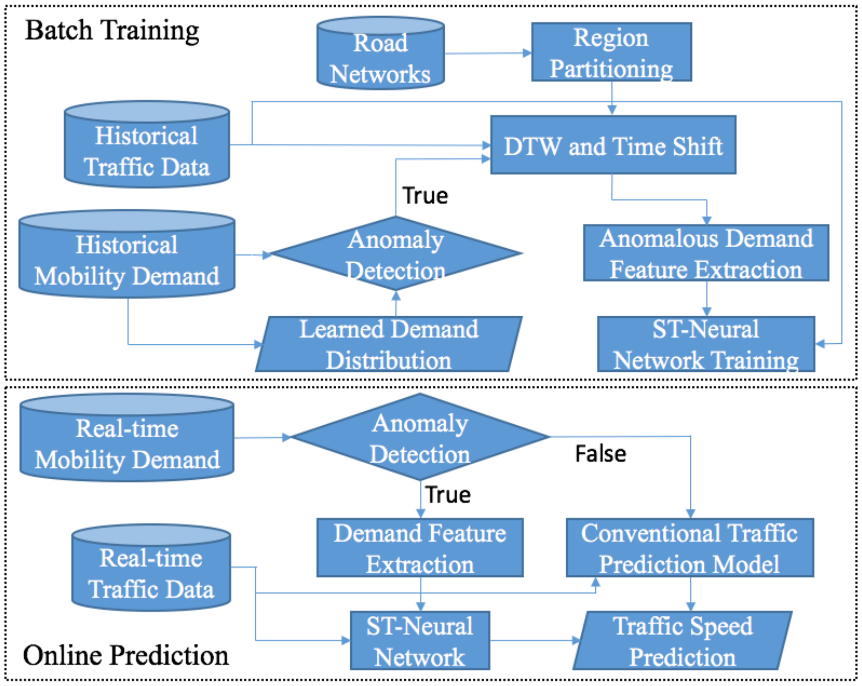


Figure 4.4. Overview of Prediction Framework

4.2. Anomalous Events Detection

Mobility demands, defined as a set of transport request, are recorded when individual makes a request with Taxi, Uber or shared bike system. If we visualize every request on the map with a dot, the painted mobility demands will cover the entire map. An appropriate aggregation, conversion and mining are essential to extract *Demand Features*, which can be applied into prediction models, from raw mobility demand data.

The anomalous events detection plays an important role in our proposed anomalous traffic speed prediction framework. On the training side, we need to classify anomalous events and collect corresponding mobility demand and traffic speed data for training purpose; on the prediction side, the anomalous events have to be detected in a real-time manner, which determines when our anomalous traffic speed prediction need to be applied to replace conventional model.

Recently, an anomaly detection method based on Likelihood Ratio Test (LRT) model has been proposed to detect anomalous event from different knowledge domains [**129, 118**]. The anomalies detection can be separated into two parts – learning and classification.

4.2.1. Region Segmentation and Demand Aggregation

Region segmentation is the foundation for demand analysis. There are various segmentation methods like hierarchy-based segmentation, morphology-based segmentation and grid-based segmentation [**121**]. Different segmentation methods have advantages and disadvantages for different applications and use cases. For example, road-network based segmentation divides the map into regions representing communities, which is semantically meaningful. However, mobility demands within one community may be different, and it is hard to control the granularity of the segmentation. In this work, we propose to adopt grid-based segmentation. The granularity of segmentation is determined by grid size $w{\times}l$. We further adopt a modified Pearson Correlation coefficient to search the correlated region with the targeting road segment.

We merge Mobility Demand data from heterogeneous data sources with different mobility mode $MD^{merge} = [MD^{driving}, MD^{biking}, ...]$. For each region, mobility demand data is aggregated within duration window Δ. Therefore, the Modal mobility demand data is converted into a *Time Series Mobility Demand* data $DT_i = (count, t_i, \Delta)$, as shown in Figure 4.5.

$$count = aggre(MD^{merge}, t_i, \Delta, R_i)$$

We choose the time interval of aggregated mobility demand to be aligned with the sampling frequency of traffic speed data that is collected with roadside sensors. Because time series mobility demand and traffic speed have significant day to day and week to week pattern, we divide time series traffics and mobility demands data into 24 hours as our minimal training/prediction unit, being denoted as TR_t^d and DT_t^d.

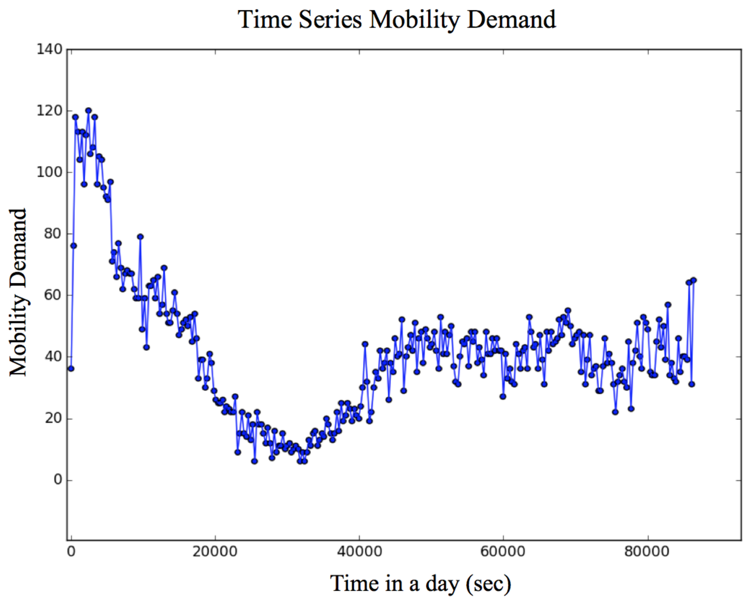


Figure 4.5. Aggregated mobility demand

4.2.2. Learning Demand Distribution

Seasonality is a natural property for mobility demands. Since mobility demands are periodic from day to day, what we are interested to learn is their long-term common pattern within a day. Based on duration window Δ, we divide a day into j time slots and adopt a statistical distribution to model the demand within each slot, as shown in Figure 4.6. The mobility demand for time slot j on day d is denoted as DT_j^d.

Poisson distributions are usually used to model time series data that happens in a completely haphazard way. In this process, the random variable counts the number of events that take place in a given time interval, and all events take place independently. For example, the number of accidents occur in an hour. However, for mobility demand data that is often over-dispersed (i.e. variance is much larger than mean), the Gaussian distribution is a more appropriate choice.

For every time slot j, a Gaussian distribution $N_j(\mu_j, \sigma_j^2)$ is adopted to model the occurrence of mobility demands. When we learn the distribution from historical data, for each day, the mobility demand DT_j^d is regarded as an independent sample. Assuming the total number of days is N, we are able to estimate the mean and variance of the distribution from samples.

$$\mu_j = \frac{\sum_{d=1}^{N} DT_j^d}{N}$$
$$\sigma_j^2 = \frac{1}{N-1}\sum_{d=1}^{N}(DT_j^d - \mu_j)^2$$

4.2.3. Detection with Statistical Test

In statistics, the LRT is a hypothesis test used to compare the fit of two models. A statistical model is a parameterized family of probability density function $f(x|\theta)$, where θ is a set of

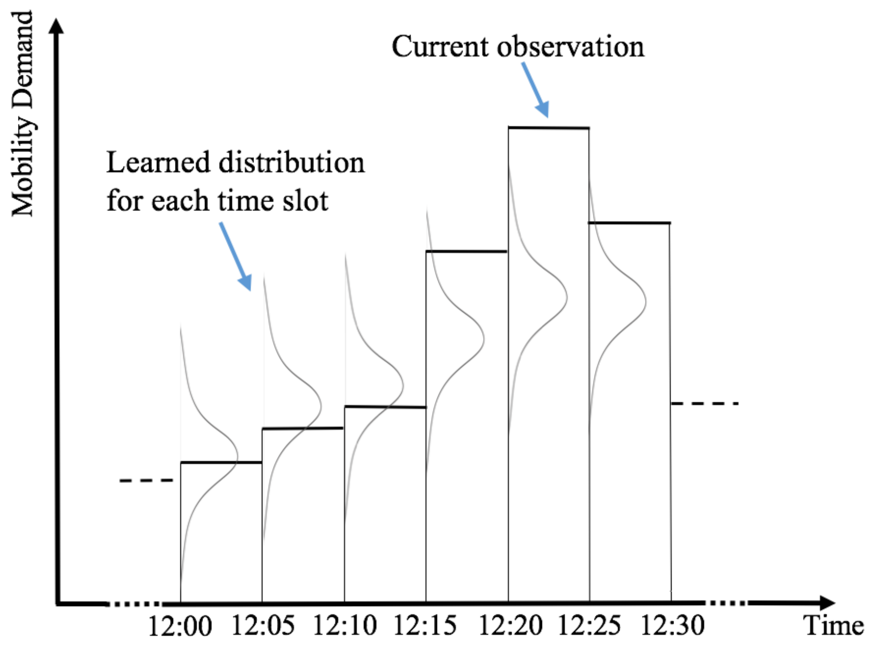


Figure 4.6. Learn null model for each time slot

parameters come from parameter space Θ. When the statistical model is Gaussian distribution, $\Theta = \{\mu, \sigma^2\}$. We assume a null hypotheses H_0 representing our knowledge based – mobility demand distribution N_j learned from historical data, and an alternative hypotheses H_1 representing our current observation.

$$H_0 = f(x|\theta_0)$$

$$H_1 = f(x|\theta_1)$$

Each of them is separated fitted to the data with the likelihood function. The likelihood ratio test is based on the likelihood ratio, which is denoted by Λ. In addition, we use the logarithm of the likelihood ratio as the test statistics so that the probability distribution of the test result can be approximated by chi-squared distribution (χ^2).

$$\Lambda(x) = -2log\frac{L(x|\theta_0)}{sup\{L(x|\theta_1)\}}$$

Where the likelihood function for N_j is defined as:

$$L(x|\theta) = (2\pi\sigma^2)^{-1/2}exp(-\frac{(x-\mu)^2}{2\sigma^2})$$

sup denotes the supremum function that finds the the θ_1 maximizing $L(x|\theta_1)$ [**129**]. The anomalous level (AL) is defined to be the *p-value* of the hypothesis test, which is estimated by:

$$AL = \chi^2(\Lambda, df)$$

The degree of freedom $df = df_1 - df_0$ represents the number of differences in terms of free parameters of the null hypothesis and alternative hypothesis. Given the significance level α, we will reject the null hypothesis when $AL \geq \alpha$, in which case the observed traffic demand is detected as an anomalous event.

4.3. Demand Feature Extraction

Once anomalous events are detected, we are able to retrieve the mobility demand and traffic speed data during the occurrence of it. However, we cannot directly fit these data into prediction model. The critical issues are the time shift and misalignment. As we discussed in section 3, there is a time delay between mobility demand and traffic speed. Furthermore, the time delay for different correlated regions are misaligned to each other for certain road segment. Therefore, we need an accurate time shift to align mobility demand with traffic speed. Given a time series mobility demand data DT_n^d within time interval $(0, n)$ and the corresponding traffic speed data TR_m^d, our task is to find an optimized point-to-point matching DT_l^d that can minimize the overall distance. We also denote a time shift $o = l - m$, where l is the time index of matched series and m is the original time index.

4.3.1. Demand Data Preprocessing

In time series analysis, Dynamic Time Warping (DTW) is a famous algorithm for measuring similarity between two temporal sequences. Inspired by DTW, we propose a Demand-DTW algorithm to find the optimized matching sub-sequence from mobility demand to traffic speed. However, there are two-step data preprocessing needed before Demand-DTW.

The first step is to transform the mobility demand data so that it is positively correlated with traffic speeds. In nature, mobility demand and traffic speeds are negatively correlated (i.e. more mobility demand will incur lower traffic speeds). However, the assumption in DTW is that, two time series should be positively correlated. Therefore, we define *Demand Feature* $DF_l^d = max(DT_l^d) - DT_l^d$ and use it as the time series being matched.

The second step is data normalization. Since mobility demand data is in the range of few hundreds while traffic speed is usually less than 100 miles per hour, in traditional DTW algorithm, many points of demand data would be matched to the point of traffic speed with the maximum value. Therefore, we first apply linear projection and normalize both mobility demand data and traffic speed data into the range $[0, 1]$.

4.3.2. Demand-DTW

Instead of calculating the global distance, we are more interested in the appropriate matching sequence and the corresponding time shift. Compared with traditional DTW problem, our data has the following properties:

a) Demand feature lead traffic speed data. Therefore, for each point of DF_l^d, we only need to consider the matching candidates with later time stamp.

b) The lengths of two time series data are not the same. Since the time shift is unknown before Demand-DTW, we usually record longer traffic speed data, so that every point of demand feature can find its optimized matching pair. When one point has multiple possible matching candidates, we choose the closest one.

c) The traditional DTW does not limit the length of connection between two points in its distance function. It is possible that a demand feature point on 9:00AM is matched to a traffic speed point on 3:00PM. One modified distance function with a penalty term is adapted [**130**].

$$Dis(DT_j^d, TR_k^d) = \begin{cases} |DT_j^d - TR_k^d|, & |j-k| < T_\Delta \\ \frac{|DT_j^d - TR_k^d|}{\beta^{|j-k|}}, & |j-k| \geq T_\Delta \end{cases}$$

β is a constant satisfying $0 < \beta < 1$ and T_Δ is the threshold for time difference penalty. When the time difference between two points are larger than threshold, a penalty is applied.

Taking the above three properties into consideration, our Demand-DTW algorithm is shown in Algorithm 6. There are two parts in the algorithm. Firstly, a distance matrix is built with n row and m col. Each cell with index $[i, j]$ represents the minimized matching distance for demand feature $DF_{0..i}$ and traffic speed $TR_{0..j}$. We fill the distance matrix using dynamic programming with transfer function $cell[i,j] = cost + min(cell[i-1,j], cell[i,j-1], cell[i-1,j-1])$. At the same time, we also keep a record of the parent for each cell. In the second part, we find the optimized matching sequence from the distance matrix and calculate the average time shift. We start from the last cell $[n,m]$ and backtrack following the trace of parent.

Algorithm 6 Demand-DTW ($DF_{0..n}$, $TR_{0..m}$)

```
1: DTW[n+1,m+1]                                              ▷ Matrix for DTW
2: for i = 0 to n do
3:     DTW[i,0].value = Inf
4: end for
5: for j = 0 to m do
6:     DTW[0,j].value = 0
7: end for
8: for i = 1 to n do
9:     for j = i to m do
10:        cost = $Dis(DF_i, TR_j)$
11:        DTW[i,j].value = cost + min(DTW[i-1,j].value, DTW[i,j-1].value, DTW[i-1,j-1].value)
12:        DTW[i,j].prev = min(DTW[i-1,j], DTW[i,j-1], DTW[i-1,j-1])
13:    end for
14: end for
15: DF = n                                                   ▷ Index for current demand
16: TR = m                                                   ▷ Index for current traffic
17: time_shift = 0
18: while DF >0 do
19:     if DTW[DF,TR].prev = DTW[DF,TR-1] then
20:         TR -= 1
21:         Continue
22:     end if
23:     time_shift += TR - DF
24:     DF = DTW[DF,TR].prev.i
25:     TR = DTW[DF,TR].prev.j
26: end while
27: Return time_shift/n
```

An average time shift o is calculated with all detected anomalous events. After that, the demand features DF_t^d are shifted to be DF_{t+o}^d, and become input to the prediction model.

4.4. Anomalous Traffic Prediction Model

Given a targeting road segment, with the knowledge of correlated region, anomalous event detection and demand features, we are able to train an anomalous traffic model and use it to predict traffic speed when anomalous events occur.

In the early stage of the development of short-term traffic prediction, most of the researches employed statistical approaches to predict traffic at a single point. For example, Autoregressive integrated moving average (ARIMA) is a classical time series model to predict future points in the time series data. However, its applicability is limited by model's linearity and stationarity of data. Recently, data driven approaches raise more and more attentions in the transportation research communities. The Artificial Neural Networks, as one of the AI-based method, outperform traditional statistical models with large flexibility in terms of input variables [**108**].

4.4.1. Artificial Neural Networks with Demand Features

We adopt the multilayer feed-forward perceptron (MLP) as our basic prediction model. For each region, the input features are the combination of historical traffic speed and demand features. In the temporal dimension, we set p be the order of lagged operation. Therefore, for traffic speed TR with current time stamp c, the input features are $(TR_c, TR_{c-1}, .., TR_{c-p})$. In addition, for each correlated region R_i, since there is a corresponding time shift o_i mined through Demand-DTW, the input demand features will be $(FD_{c+o}, FD_{c+o-1}, .., FD_{c+o-p})$. The Demand-MLP model is formulated as:

$$TR_{c+t} = EM(TR, FD) = \sum_{i=1}^{H} W_i f(\sum_{j=0}^{p} w_{ih} TR_{c-j} + \sum_{j=0}^{p} w_{ih} DF_{c+o-j} + \varepsilon)$$

where $f(\cdot)$ is the activation function; W_i and $w_i h$ are weights of the connection or synaptic (i.e.,coefficient) estimated through training; and ε is learned constant. Its topological structure is shown in Figure 4.7.

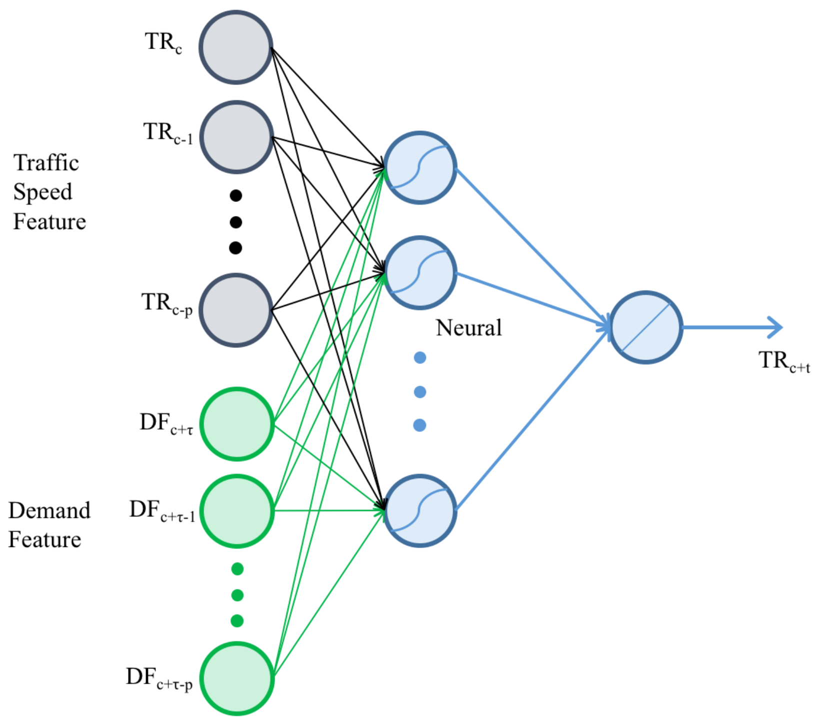


Figure 4.7. MLP with demand features

The number of neurals in the hidden layer can be systematically optimized with genetic algorithm [**109**]. To train the network, we minimize the residuals $\sum(\hat{TR}-TR)^2$ with gradient descent and backpropagation.

4.4.2. Training and Prediction

The training of Demand-MLP model is executed in batch mode. It requires the combination of anomalous events detection, Demand-DTW and Demand-MLP training. Firstly, we process the training data with anomalous events detection. Once an anomalous event is detected on time t, a continuous time series data within time interval $(t, t+\kappa)$ from demand feature and traffic speed will be store. κ represent the time window for selecting training data, $\kappa > o$. The reason we need to select additional data when anomalies are detected is that, the Demand-DTW requires points in the future to match with lagged time series. After calculating the average time shift through Demand-DTW, we shift the demand features and use them, together with traffic speeds, to train the Demand-MLP model.

The prediction is a real time process, where we have the current traffic speed data and mobility demand data. Given a targeting road segment, one MLP model is trained for every correlated region. The final prediction result is the combination of the output from multiple models. The decision making mechanism is shown in Figure 4.8.

If there is no anomalies according to mobility demand, the traditional traffic speed prediction model make a prediction based on historical traffic speeds. When an anomaly is detected in certain region, the corresponding model is triggered. The mobility demand data are converted and time-shifted into demand features, which act as input with current traffic speed into the Demand-MLP model. The predicted traffic speed will be determined by the

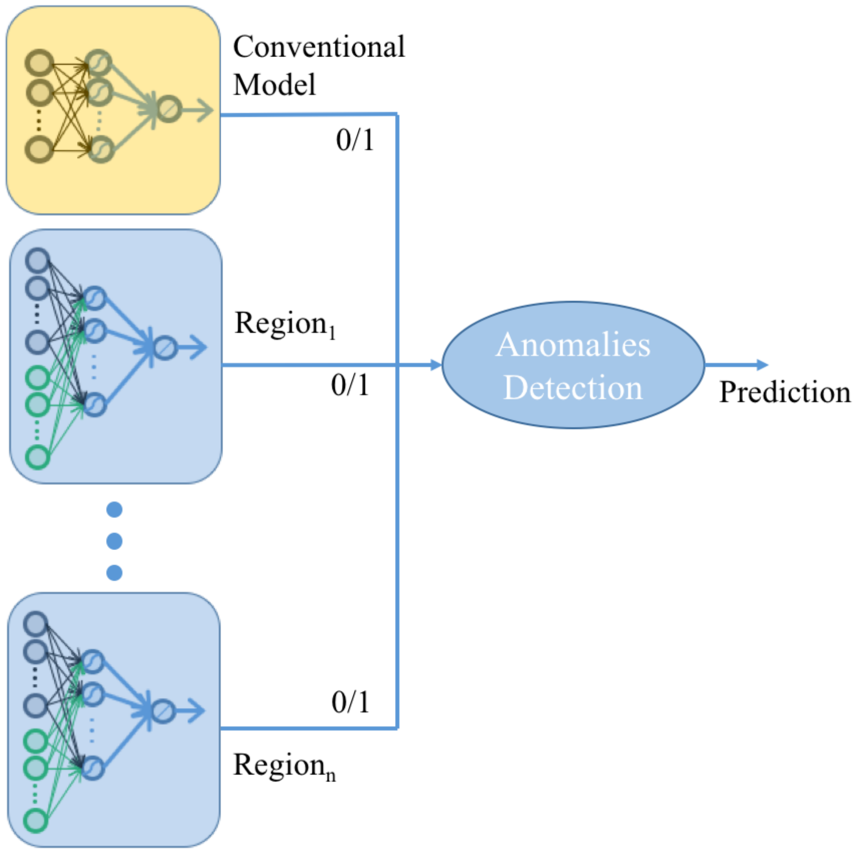


Figure 4.8. Different model is applied in different scenario

model representing the region with anomalous events. When multiple anomalies are detected simultaneously, the prediction result will be the average of output value from all triggered models.

4.5. Experiment

4.5.1. Evaluation Setup

Datasets: The datasets being used in this experiment are real-life traffic speed and taxi trip data from New York City. The traffic speed data are collected by the City of New York Department of Transportation [**103**] from April 2015 to June 2016. It records the real-time traffic speeds monitored by road-side sensors being deployed on 150 highways in the New York City. The traffic speeds are sampled every five minutes.

Taxi trip data are published by New York City Taxi and Limousine Commission [**98**], which consist of fields capturing pick-up and drop-off dates/time, trip distance, fare and other information. The trip records include the services provided by yellow and green taxi form April 2015 to June 2016. The total size of uncompressed data is more than 40 GB.

New York Yankees are an american professional baseball team based in the New York city. Their home games, which held in the Yankee Stadium (cf. Figure 4.3), attract attendance of over 50000 fans. The home games in baseball season 2015 and season 2016 are irregularly scheduled in the afternoons and nights. When a particular game ends, the departure of large groups of people results in a surge of mobility demand, which is the anomalous event that can be detected.

Our experiment focus on a road segment near Yankee Stadium. Four nearby correlated regions are identified. We use the traffic speed and taxi trip data in game season 2015, which spans from April 2015 to October 2015 as our training datasets, and use the data from April 2016 to June 2016, which belongs to game season 2016 as validation datasets.

Baseline comparison: When anomalous events occur, we compare our Demand-MLP model with traditional ANN based traffic prediction model that is trained with normal historical traffic speeds with the same data size as Demand-MLP. It has been demonstrated that, ANN based traffic prediction model outperforms statistical model like Auto-Regressive Integrated Moving Average (ARIMA) or Vector Auto-Regressive (VAR) [**108, 125**].

Evaluation parameter: The prediction length range from 5 mins to 30 mins. The order of lagged operation p is set to be 6. The significance level α of the LRT is chosen to be 90%. The New York City is partitioned using grid with width 1000m.

Evaluation criteria: We adopt the commonly used measurement metric Root Mean Square Error (RMSE) to measure our approaches.

$$RMSE = \sqrt{\frac{1}{z}\sum_{i=1}^{i=z}(TR_i - \hat{TR_i})^2}$$

where $\hat{TR}$ and TR are the predicted value and ground truth, and z is the number of all predicted values.

4.5.2. Traffic Speed Prediction when Anomalous Events Occur

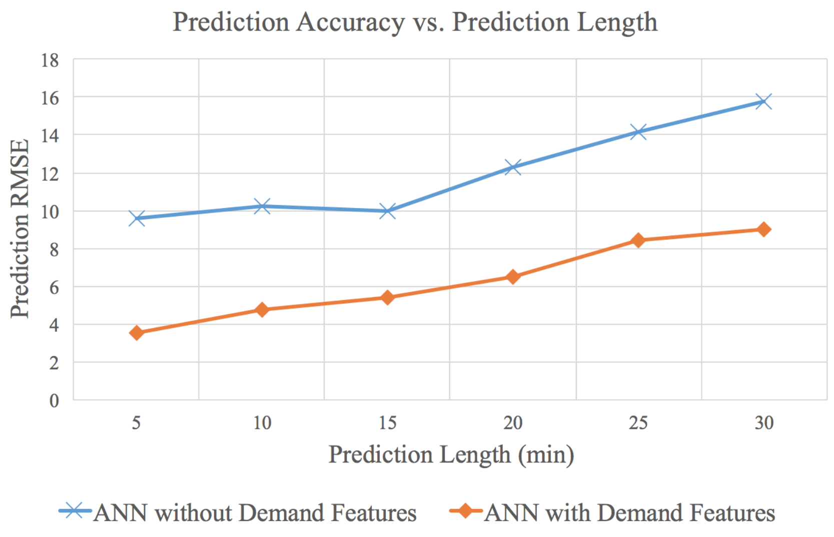


Figure 4.9. RMS with different prediction length

We train the Demand-MLP model with traffic speed and demand feature when anomalous mobility demand is detected with LRT, using data from April 2015 to October 2015. The

ANN model is trained with traffic speed data from April 2015 to October 2015, with the same data size as anomalous traffic data being used for training Demand-MLP. We predict the traffic speed when Demand-MLP and ANN model when anomalous events occur from April 2016 to June 2016. Figure 4.9 shows the RMSE for different prediction length using Demand-MLP and tradition ANN model. When predicting traffic speeds under anomalous events, the Demand-MLP outperform traditional ANN model and reduce around 30% RMSE.

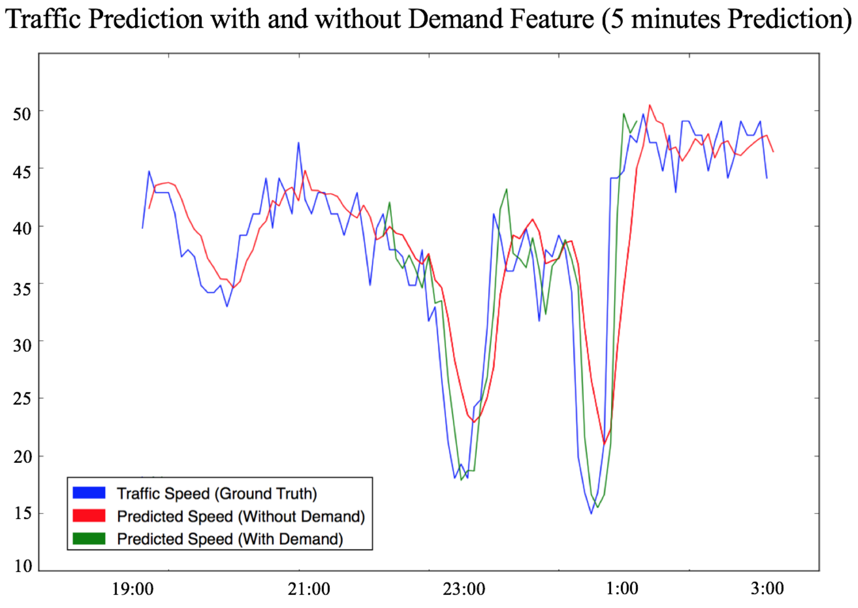


Figure 4.10. Prediction of traffic speed during the event of baseball game

Figure 4.10 is a visualization for the traffic speed for a night from 19:00 to 3:00. The baseball game started from 19:15 and ended around 22:30. There is a significant traffic speed drop near 23:00 as shown in the figure, which corresponding to the end of the baseball game. The Demand-MLP model is able to generate a prediction that better fit the ground truth.

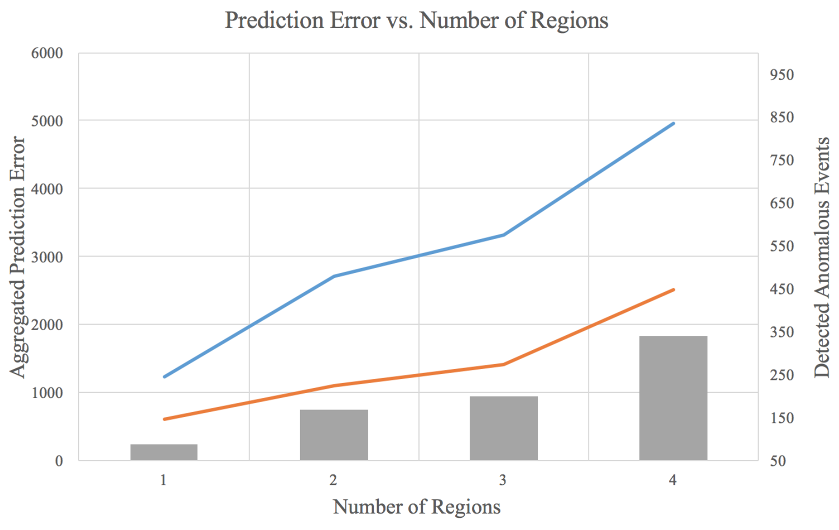


Figure 4.11. Prediction error with multiple regions

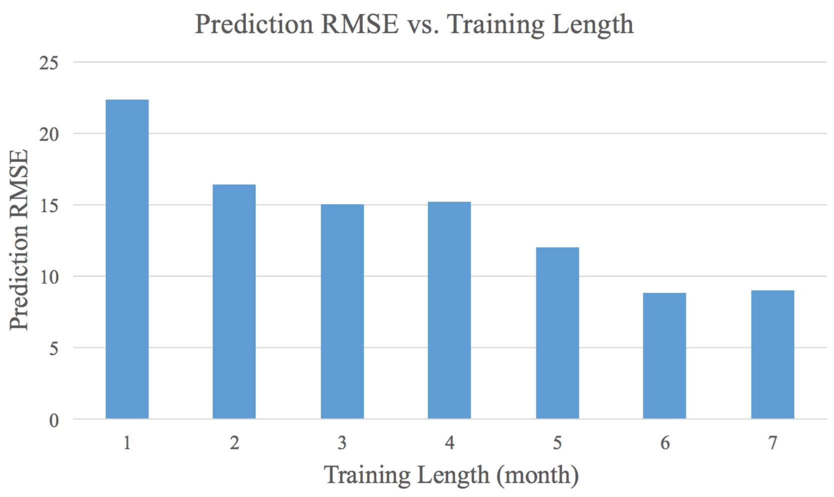


Figure 4.12. Prediction error with different size of training data

4.5.3. Prediction Accuracy with Multiple Regions

As shown in Figure 4.11, four regions $[R_A, R_B, R_C, R_D]$ are correlated with targeting road segment when anomalies occur. Different kinds of anomalous events happen in different regions at different times. When we incorporate more related regions into our prediction framework, more anomalous events are detected and our prediction framework will further reduce the amount of prediction errors, compared with cases when a single ANN model is used for all traffic scenarios. We also found in our experiment that, most anomalous events being detected in different regions are not overlapping with other anomalies.

Figure 4.12 demonstrate the effect of the size of training data. When we use larger datasets to train the Demand-MLP model, the model is able to generate a more accurate prediction.

4.6. Related Work

One way of characterizing the traffic prediction is along two broad directions: the long-term and short-term traffic prediction. Long term traffic prediction aims to model the physical process that governs the evolution of traffic [**130**]. These models can be (and are often) used for urban planning.

Short-term traffic prediction attempts to predict day-to-day and hour-to-hour status of traffic. According to the types of prediction models, there are three directions of short-term traffic predictions, namely, statistical approaches, machine learning approaches and other hybrid methods [**110**].

Furthermore, the studies of short-term traffic prediction can be categorized in to two types: one only considers the traffic data (e.g., historical traffic data collected by inductive loop sensor), and the other takes additional data into consideration (e.g., weather context). For traffic prediction using traffic sensing data, a great many learning and inference algorithms have been proposed, including linear regression, univariate and multivariate state-space methods (ARIMA), neural networks, k-nearest neighbors, Kalman filtering and many others. For traffic prediction assisted by additional information, two traditionally considered classes of information are weather information and holidays. Two recent work reports using multisource data related to traffic (including taxicabs, buses, trucks, subway, cellular data and building occupancy data) to predict traffic status [**127, 130**].

Anomaly detection refers to the problem of finding patterns in data that do not conform to expected behavior. Detecting outliers or anomalies in data has been studied in the statistics community as early as the 19th century [**12**]. Over time, a variety of anomaly detection techniques have been developed in several research communities. These techniques include classification, clustering, nearest neighbor, statistical and information theory. In the spatial-temporal data mining community, a statistical approach based on likelihood ratio test was proposed recently [**118, 129**].

4.7. Concluding Remarks and Future Works

In this chapter, we proposed a demand feature and a prediction framework that enables a more accuracy short-term prediction. Demand feature are extracted from user's transport request. An ANN based prediction model and prediction framework are proposed to incorporate demand feature with traffic speed information. A real-life traffic speed data and taxi request data for the city of New York is collected and used into the evaluation. Our experiments demonstrate that, the Demand-MLP model with demand feature yield a more accurate short-term traffic prediction, under the occurrence of anomalous events.

There are several possible extensions to our work. Firstly, we would like to investigate the possibility of application of multi-layer neural network/deep learning into our prediction framework. The reason we propose to train a separate MLP model for each region is that, the timings for anomalous events occur in different regions vary. Mixing demand feature under anomalous events with those feature in usual case would reduce the overall learning ability of ANN. A deep network with hierarchical structure maybe the solution for that. Another possible avenue is to consider the possibility of merging two or more regions and their corresponding prediction models. An large scale anomalous events may affect more that one regions. If two regions are similar enough, we may be able to merge them to simplify our model.

CHAPTER 5

Concluding Remarks and Future Works

In this dissertation, we tackled certain aspects from the broad field of spatio-temporal data management and mining. Specifically, we addressed three categories of problems:

(1) Merging heterogeneous location data to reduce GPS uncertainties.

(2) Mining speed clusters in multi-lane setting.

(3) Predicting of anomalous traffic speed with mobility demand.

We conducted multiple experimental evaluations with real datasets, as well as synthetic ones. Regarding the three main aspects of this dissertation, experimental observations demonstrated that:

- Integrating/fusing the data from both sources narrows the possible whereabouts when compared to each individual location data source.
- Speed clusters mining under multi-lane setting yield a compact while accurate data representation for traffic speed, which in turn lead to a more accurate travel time estimation.
- Prediction model incorporating people's mobility demand achieve a more accurate short-term traffic speed prediction.

As a spatial-temporal data mining research in the application of transportation related tasks, our works contribute both transportation research community, as well as computer science community. We introduce data driven approaches with machine learning techniques to traditional transportation tasks, which used is dominated by statistical model. The results of our research will benefit multiple kinds of application with high societal relevance, like route planning, travel time estimation, the development of intelligent transportation system and traffic management schemes that are more dynamics-aware.

In each of the previous Chapters, we indicated directions for future works aligned with the respective topics addressed therein. However, there are some broader, more visionary application domains where we believe our results could be extended/augmented, both in terms of adapting the contexts as well as tackling the novel challenges.

One broad category is the efficient (re)routing of autonomous cars [**34**]. The particular challenge here is how to efficiently incorporate the updates of the traffic-changes in the different paradigms of computing and communication – i.e., coupling cloud and V2V communication [**3**].

Another challenging research area is maritime data. Efficiently mining and clustering vessel trajectories to help optimize ship safety and operation will lead to a huge impact on global economy and our everyday life [**16**]. Due to the volume, velocity and heterogeneity of maritime data, an integrated maritime information management system requires heterogeneous data merging, events recognition/detection and trajectories clustering. The tasks would be even more complex to combine multiple modalities of transportation including aviation data, maritime data and ground transportation data.

References

[1] Autonomous cars can only understand the real world through a map (2016). Https://goo.gl/Q7U1bw

[2] What to expect on new years (2017). Https://goo.gl/5dsJhV

[3] Abuelsamid, S.: Tesla autopilot fatality shows why lidar and v2v will be necessary for autonomous cars. Forbes (2016)

[4] Alam, N., Balaei, A.T., Dempster, A.G.: An instantaneous lane-level positioning using dsrc carrier frequency offset. Intelligent Transportation Systems, IEEE Transactions on **13**(4), 1566–1575 (2012)

[5] Andrienko, G., Malerba, D., May, M., Teisseire, M.: Mining spatio-temporal data. Journal of Intelligent Information Systems **27**(3), 187–190 (2006)

[6] Baskar, L.D., De Schutter, B., Hellendoorn, J., Papp, Z.: Traffic control and intelligent vehicle highway systems: a survey. IET Intelligent Transport Systems **5**(1), 38–52 (2011)

[7] Beckmann, N., Kriegel, H.P., Schneider, R., Seeger, B.: The R*-tree: an efficient and robust access method for points and rectangles, vol. 19. ACM (1990)

[8] Bracciale, L., Bonola, M., Loreti, P., Bianchi, G., Amici, R., Rabuffi, A.: CRAWDAD dataset roma/taxi (v. 2014-07-17). Downloaded from `http://crawdad.org/roma/taxi/20140717` (2014). DOI 10.15783/C7QC7M

[9] Brugere, I., Gunturi, V.M.V., Shekhar, S.: Modeling and analysis of spatiotemporal social networks. In: Encyclopedia of Social Network Analysis and Mining, pp. 950–960 (2014)

[10] Cao, Q., Yan, T., Stankovic, J., Abdelzaher, T.: Analysis of target detection performance for wireless sensor networks. In: DCOSS, pp. 276–292 (2005)

[11] Castro, P.S., Zhang, D., Li, S.: Urban traffic modelling and prediction using large scale taxi gps traces. In: International Conference on Pervasive Computing, pp. 57–72. Springer (2012)

[12] Chandola, V., Banerjee, A., Kumar, V.: Anomaly detection: A survey. ACM computing surveys (CSUR) **41**(3), 15 (2009)

[13] Chen, D., Driemel, A., Guibas, L.J., Nguyen, A., Wenk, C.: Approximate map matching with respect to the fréchet distance. In: ALENEX, pp. 75–83 (2011)

[14] Chen, W., Hou, J., Sha, L.: Dynamic clustering for acoustic target tracking in wireless sensor networks. In: IEEE International Conference on Network Protocols (ICNP'03) (2003)

[15] Cheng, R., Emrich, T., Kriegel, H.P., Mamoulis, N., Renz, M., Trajcevski, G., Züfle, A.: Managing uncertainty in spatial and spatio-temporal data. In: ICDE, pp. 1302–1305 (2014)

[16] Claramunt, C., Ray, C., Camossi, E., Jousselme, A., Hadzagic, M., Andrienko, G.L., Andrienko, N.V., Theodoridis, Y., Vouros, G.A., Salmon, L.: Maritime data integration and analysis: recent progress and research challenges. In: Proceedings of the 20th International Conference on Extending Database Technology, EDBT 2017, Venice, Italy, March 21-24, 2017., pp. 192–197

[17] Cui, D., Xue, J., Zheng, N.: Real-time global localization of robotic cars in lane level via lane marking detection and shape registration. IEEE Transactions on Intelligent Transportation Systems **17**(4), 1039–1050 (2016)

[18] Dao, T.S., Leung, K.Y., Clark, C.M., Huissoon, J.P.: Markov-based lane positioning using intervehicle communication. Trans. Intell. Transport. Sys. **8**(4), 641–650 (2007). DOI 10.1109/TITS.2007.908574. URL `http://dx.doi.org/10.1109/TITS.2007.908574`

[19] Department of Transportation, U.S.: Travel monitoring and traffic volume (2014). URL `http://www.fhwa.dot.gov/policyinformation/travelmonitoring.cfm`. Office of Highway Policy Information

[20] van Diggelen, F.: Update: Gnss accuracy: Lies, damn lies, and statistics. GPS World (2007)

[21] Ding, C.H.Q., He, X., Zha, H., Gu, M., Simon, H.D.: A min-max cut algorithm for graph partitioning and data clustering. In: Proceedings of the 2001 IEEE ICDM, pp. 107–114 (2001)

[22] Ding, Z., Güting, R.H.: Managing moving objects on dynamic transportation networks. In: SSDBM (2004)

[23] Ding, Z., Güting, R.H.: Uncertainty management for network constrained moving objects. In: DEXA, pp. 411–421 (2004)

[24] Dodge, S., Weibel, R., Forootan, E.: Revealing the physics of movement: Comparing the similarity of movement characteristics of different types of moving objects. Computers, Environments and Urban Systems (2009)

[25] Du, J., Barth, M.: Next-generation automated vehicle location systems: Positioning at the lane level. Intelligent Transportation Systems, IEEE Transactions on **9**(1), 48–57 (2008). DOI 10.1109/TITS.2007.908141

[26] Du, J., Masters, J., Barth, M.: Lane-level positioning for in-vehicle navigation and automated vehicle location (avl) systems. In: Intelligent Transportation Systems, 2004. Proceedings. The 7th International IEEE Conference on, pp. 35–40. IEEE (2004)

[27] EasySen LLC.: Wieye - sensor board for wireless surveillance applications. 401 North Coquillard Dr., South Bend, IN 46617 (2008)

[28] Emrich, T., Kriegel, H.P., Mamoulis, N., Renz, M., Züfle, A.: Querying uncertain spatio-temporal data. In: ICDE, pp. 354–365 (2012)

[29] Ester, M., Kriegel, H.P., Sander, J., Xu, X.: A density-based algorithm for discovering clusters in large spatial databases with noise. In: Kdd, vol. 96, pp. 226–231 (1996)

[30] Gaffney, S., Smyth, P.: Trajectory clustering with mixtures of regression models. In: Proceedings of the fifth ACM SIGKDD international conference on Knowledge discovery and data mining, pp. 63–72. ACM (1999)

[31] Gartner, N.H., Messer, C.J., Rathi, A.K.: Monograph on traffic flow theory. Federal Highway Administration (1997)

[32] George, B., Shekhar, S.: SP-TAG: a routing algorithm in non-stationary transportation networks. In: 5th Annual International Conference on Mobile and Ubiquitous Systems: Computing, Networking, and Services, MobiQuitous 2008, July 21-25, 2008, Dublin, Ireland (2008)

[33] Gilat, A.: Numerical Methods for Engineers and Scientists. Wiley (2010)

[34] Gomes, L.: Hidden obstacles for googles self-driving cars. MIT Technology Review (2014)

[35] Gowrisankar, N., Nittel, S.: Reducing uncertainty in location prediction of moving objects in road networks. In: GIScience (2002)

[36] Gudmundsson, J., van Kreveld, M.J.: Computing longest duration flocks in trajectory data. In: GIS (2006)

[37] Guo, J., Huang, W., Williams, B.M.: Adaptive kalman filter approach for stochastic short-term traffic flow rate prediction and uncertainty quantification. Transportation Research Part C: Emerging Technologies **43**, 50–64 (2014)

[38] Güting, R.H., Schneider, M.: Moving Objects Databases. Morgan Kaufmann (2005)

[39] Hägerstrand, T.: What about people in regional science? Papers of the Regional Science Association **24**, 7–21 (1970)

[40] Hellebrandt, M., Mathar, R.: Location tracking of mobiles in cellular radio networks **45**(5), 1558–1562 (1998)

[41] Hill, T.: Conflations of probability distributions. Transactions of the American Mathematical Society **363**(6), 3351–3372 (2011)

[42] Hill, T.P., Miller, J.: How to combine independent data sets for the same quantity. Chaos: An Interdisciplinary Journal of Nonlinear Science **21**(3), 033,102 (2011)

[43] Honeywell International Inc: Vehicle detection using amr sensors. Tech. rep., Defense and Space Electronics Systems, 12001 Highway 55, Plymouth, MN 55441 (2005)

[44] Hong, L., Zheng, Y., Yung, D., Shang, J., Zou, L.: Detecting urban black holes based on human mobility data. In: Proceedings of the 23rd SIGSPATIAL International Conference on Advances in Geographic Information Systems, p. 35. ACM (2015)

[45] Hornsby, K., Egenhofer, M.J.: Modeling moving objects over multiple granularities. Ann. Math. Artif. Intell. **36**(1-2), 177–194 (2002)

[46] Jeong, J., Hwang, T., He, T., Du, D.H.C.: Mcta: Target tracking algorithm based on minimal contour in wireless sensor networks. In: INFOCOM (2007)

[47] Jeung, H., Yiu, M.L., Zhou, X., Jensen, C.S., Shen, H.T.: Discovery of convoys in trajectory databases. PVLDB **1**(1) (2008)

[48] Kelly, R.: M42 active traffic management scheme. Road Traffic Technology (2007). S.M. Limited Editor. Birmingham, United Kingdom

[49] Khaleghi, B., Khamis, A.M., Karray, F., Razavi, S.N.: Multisensor data fusion: A review of the state-of-the-art. Information Fusion **14**(1), 28–44 (2013)

[50] Kim, S., Shekhar, S.: Contraflow network reconfiguration for evacuation planning: a summary of results. In: 13th ACM International Workshop on Geographic Information Systems, ACM-GIS 2005, November 4-5, 2005, Bremen, Germany, Proceedings, pp. 250–259 (2005)

[51] Kogan, V., Shimshoni, I., Levi, D.: Lane-level positioning with sparse visual cues. In: Intelligent Vehicles Symposium (IV), 2016 IEEE, pp. 889–895. IEEE (2016)

[52] Kuijpers, B., Miller, H.J., Neutens, T., Othman, W.: Anchor uncertainty and space-time prisms on road networks. International Journal of Geographical Information Science **24**(8), 1223–1248 (2010)

[53] Kuijpers, B., Miller, H.J., Othman, W.: Kinetic space-time prisms. GIS '11, pp. 162–170. ACM, Chicago, IL, USA (2011). DOI 10.1145/2093973.2093996. URL `http://doi.acm.org/10.1145/2093973.2093996`

[54] Kuijpers, B., Othman, W.: Modelling uncertainty on road networks via space-time prisms. Int.l Journal on GIS **23**(9) (2009)

[55] Kuijpers, B., Othman, W.: Trajectory databases: data models, uncertainty and complete query languages. Journal of Computer and System Sciences (2009). Doi:10.1016/j.jcss.2009.10.002

[56] Lan, L.W., Sheu, J.B., Huang, Y.S.: Investigation of temporal freeway traffic patterns in reconstructed state spaces. Transportation Research Part C: Emerging Technologies **16**(1), 116–136 (2008)

[57] Laporte, G.: The vehicle routing problem: An overview of exact and approximate algorithms. European Journal of Operational Research **59**(3), 345–358 (1992)

[58] Laser Technology Inc: Trusense t-series (2013). Online; accessed May 22, 2014

[59] gil Lee, J., Han, J., Whang, K.: Trajectory clustering: A partition-and-group framework. In: SIGMOD, pp. 593–604 (2007)

[60] Lee, J.G., Han, J., Li, X., Gonzalez, H.: Traclass: trajectory classification using hierarchical region-based and trajectory-based clustering. Proceedings of the VLDB Endowment **1**(1), 1081–1094 (2008)

[61] Leonard, T.: Delivering deeper insights with big data and real-time analytics (2012)

[62] Li, G., Li, Y., Shu, L., Fan, P.: Cknn query processing over moving objects with uncertain speeds in road networks. In: APWeb, pp. 65–76 (2011)

[63] Lin, C., Choy, K.L., Ho, G.T., Chung, S., Lam, H.: Survey of green vehicle routing problem: Past and future trends. Expert Systems with Applications **41**(4), 1118–1138 (2014)

[64] Lin, J.: Divergence measures based on the shannon entropy. IEEE Transactions on Information theory **37**(1), 145–151 (1991)

[65] Liu, W., Zheng, Y., Chawla, S., Yuan, J., Xing, X.: Discovering spatio-temporal causal interactions in traffic data streams. In: KDD, pp. 1010–1018 (2011)

[66] Liu, Y., Choudhary, A.N., Zhou, J., Khokhar, A.A.: A scalable distributed stream mining system for highway traffic data. In: PKDD (2006)

[67] Lopez-Garcia, P., Onieva, E., Osaba, E., Masegosa, A.D., Perallos, A.: A hybrid method for short-term traffic congestion forecasting using genetic algorithms and cross entropy. IEEE Transactions on Intelligent Transportation Systems **17**(2), 557–569 (2016)

[68] Ma, X., Tao, Z., Wang, Y., Yu, H., Wang, Y.: Long short-term memory neural network for traffic speed prediction using remote microwave sensor data. Transportation Research Part C: Emerging Technologies **54**, 187–197 (2015)

[69] Mckinsey Global Institute: Big data: The next frontier for innovation, competition, and productivity (2011)

[70] Min, W., Wynter, L.: Real-time road traffic prediction with spatio-temporal correlations. Transportation Research Part C: Emerging Technologies **19**(4), 606–616 (2011)

[71] Newson, P., Krumm, J.: Hidden Markov map matching through noise and sparseness. Proceedings of the 17th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems - GIS '09 pp. 336–343 (2009). DOI 10.1145/1653771.1653818. URL `http://portal.acm.org/citation.cfm?doid=1653771.1653818`

[72] Nicholson, H., Swann, C.: The prediction of traffic flow volumes based on spectral analysis. Transportation Research **8**(6), 533–538 (1974)

[73] Niedermayer, J., Züfle, A., Emrich, T., Renz, M., Mamoulis, N., Chen, L., Kriegel, H.P.: Probabilistic nearest neighbor queries on uncertain moving object trajectories. PVLDB **7**(3), 205–216 (2013)

[74] Parent, C., Spaccapietra, S., Renso, C., Andrienko, G., Andrienko, N., Bogorny, V., Damiani, M.L., Gkoulalas-Divanis, A., Macedo, J., Pelekis, N., Theodoridis, Y., Yan, Z.: Semantic trajectories modeling and analysis. ACM Comput. Surv. **45**(4), 42:1–42:32 (2013)

[75] Pascale, A., Deflorio, F., Nicoli, M., Dalla Chiara, B., Pedroli, M.: Motorway speed pattern identification from floating vehicle data for freight applications. Transportation Research Part C: Emerging Technologies **51**, 104–119 (2015)

[76] Pattem, S., Poduri, S., Krishnamachari, B.: Energy-quality tradeoffs for target tracking in wireless sensor networks. In: IPSN (2003)

[77] Pfoser, D., Jensen, C.S.: Capturing the uncertainty of moving objects representation. In: SSD (1999)

[78] Pfoser, D., Tryfona, N.: Capturing fuzziness and uncertainty of spatiotemporal objects. In: ADBIS, pp. 112–126 (2001)

[79] Pfoser, D., Tryfona, N., Jensen, C.S.: Indeterminacy and spatiotemporal data: Basic definitions and case study. GeoInformatica **9**(3) (2005)

[80] Pillac, V., Gendreau, M., Guéret, C., Medaglia, A.L.: A review of dynamic vehicle routing problems. European Journal of Operational Research **225**(1), 1–11 (2013)

[81] Quddus, M.a., Ochieng, W.Y., Noland, R.B.: Current map-matching algorithms for transport applications: State-of-the art and future research directions. Transportation Research Part C: Emerging Technologies **15**, 312–328 (2007)

[82] Rabe, J., Necker, M., Stiller, C.: Ego-lane estimation for lane-level navigation in urban scenarios. In: Intelligent Vehicles Symposium (IV), 2016 IEEE, pp. 896–901. IEEE (2016)

[83] Roddick, J.F., Hornsby, K. (eds.): Temporal, Spatial, and Spatio-Temporal Data Mining, First International Workshop TSDM 2000 Lyon, France, September 12, 2000, Revised Papers, *Lecture Notes in Computer Science*, vol. 2007. Springer (2001)

[84] Schiller, J., (editors), A.V.: Location-Based Services. Morgan Kaufmann (2004)

[85] Schramm, A.J., Rakotonirainy, A.: The effect of road lane width on cyclist safety in urban areas. In: Proceedings of the 2009 Australasian Road Safety Research, Policing and Education Conference: Smarter, Safer Directions. Roads and Traffic Authority of New South Wales, Australia (2009)

[86] Seif, H.G., Hu, X.: Autonomous driving in the icity - hd maps as a key challenge of the automotive industry. Engineering **2**(2), 159–162 (2016)

[87] Sekimoto, Y., Matsubayashi, Y., Yamada, H., Imai, R., Usui, T., Kanasugi, H.: Lightweight lane positioning of vehicles using a smartphone gps by monitoring the distance from the center line. In: ITSC 2012, pp. 1561–1565 (2012). DOI 10.1109/ITSC.2012.6338737

[88] Shang, J., Zheng, Y., Tong, W., Chang, E., Yu, Y.: Inferring gas consumption and pollution emission of vehicles throughout a city. In: Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining, pp. 1027–1036. ACM (2014)

[89] Shannon, C.E.: A mathematical theory of communication. ACM SIGMOBILE Mobile Computing and Communications Review **5**(1), 3–55 (2001)

[90] Shekhar, S., Chawla, S.: Spatial Databases: A Tour. Prentice Hall (2003)

[91] Shekhar, S., Jiang, Z., Ali, R.Y., Eftelioglu, E., Tang, X., Gunturi, V.M.V., Zhou, X.: Spatiotemporal data mining: A computational perspective. ISPRS Int. J. Geo-Information **4**(4), 2306–2338 (2015). DOI 10.3390/ijgi4042306. URL `http://dx.doi.org/10.3390/ijgi4042306`

[92] da Silva, T.L.C., Zeitouni, K., de Macêdo, J.A.: Online clustering of trajectory data stream. In: Mobile Data Management (MDM), 2016 17th IEEE International Conference on, vol. 1, pp. 112–121. IEEE (2016)

[93] Sistla, A., Wolfson, P., Chamberlain, S., Dao, S.: Querying the uncertain positions of moving objects. In: O. Etzion, S. Jajodia, S. Sripada (eds.) Temporal Databases: Research and Practice (1999)

[94] Southwest Research Institute: Advanced traffic management systems. Http://www.swri.org/4ORG/d10/its/atms/default.htm

[95] for State Highway, A.A., (AASHTO), T.O.: A Policy on Geometric Design of Highways and Streets (2011)

[96] Stathopoulos, A., Karlaftis, M., Dimitriou, L.: Fuzzy rule-based system approach to combining traffic count forecasts. Transportation Research Record: Journal of the Transportation Research Board (2183), 120–128 (2010)

[97] Tao, Y., Xiao, X., Cheng, R.: Range search on multidimensional uncertain data. ACM Trans. Database Syst. **32**(3), 15 (2007)

[98] Taxi, N.Y.C., Commission, L.: Tlc trip record data. http://www.nyc.gov/html/tlc/html/about/trip˙record˙data.shtml

[99] Toledo-Moreo, R., Betaille, D., Peyret, F.: Lane-level integrity provision for navigation and map matching with gnss, dead reckoning, and enhanced maps. Intelligent Transportation Systems, IEEE Transactions on **11**(1), 100–112 (2010). DOI 10.1109/TITS.2009.2031625

[100] Trajcevski, G., Choudhary, A., Wolfson, O., Li, Y., Li, G.: Uncertain range queries for necklaces. In: MDM (2010)

[101] Trajcevski, G., Tamassia, R., Cruz, I., Scheuermann, P., Hartglass, D., Zamierowski, C.: Ranking continuous nearest neighbors for uncertain trajectories. VLDB J. **20**(5), 767–791 (2011)

[102] Trajcevski, G., Wolfson, O., Hinrichs, K., Chamberlain, S.: Managing uncertainty in moving objects databases. ACM Trans. Database Syst. **29**(3) (2004)

[103] of New York Department of Transportation, C.: Real-time traffic speed data. `http://www.nyc.gov/html/dot/html/about/datafeeds.shtml` (2017)

[104] Treiber, M., Kesting, A.: Traffic flow dynamics. Springer (2013)

[105] Turner-Fairbank Highway Research Center: Traffic Detector Handbook. U.S. Department of transportation

[106] United States Department of Defense: Navstar gps: Global positioning system standard (2008)

[107] Van Der Voort, M., Dougherty, M., Watson, S.: Combining kohonen maps with arima time series models to forecast traffic flow. Transportation Research Part C: Emerging Technologies **4**(5), 307–318 (1996)

[108] Vlahogianni, E., Karlaftis, M.: Testing and comparing neural network and statistical approaches for predicting transportation time series. Transportation Research Record: Journal of the Transportation Research Board (2399), 9–22 (2013)

[109] Vlahogianni, E.I., Karlaftis, M.G., Golias, J.C.: Optimized and meta-optimized neural networks for short-term traffic flow prediction: A genetic approach. Transportation Research Part C: Emerging Technologies **13**(3), 211–234 (2005)

[110] Vlahogianni, E.I., Karlaftis, M.G., Golias, J.C.: Short-term traffic forecasting: Where we are and where we17 are going. Transportation Research Part C: Emerging Technologies **43**, 3–19 (2014)

[111] Vu, A., Ramanandan, A., Chen, A., Farrell, J., Barth, M.: Real-time computer vision/dgps-aided inertial navigation system for lane-level vehicle navigation. Intelligent Transportation Systems, IEEE Transactions on **13**(2), 899–913 (2012)

[112] Wang, H., Yao, K., Estrin, D.: Information-theoretic approaches for sensor selection and placement for target localization and tracking in sensor networks. JCN **7**(4), 438–449 (2005)

[113] Wang, Y., Zheng, Y., Xue, Y.: Travel time estimation of a path using sparse trajectories. Proceeding of the 20th SIGKDD (5) (2014)

[114] Wenk, C., Salas, R., Pfoser, D.: Addressing the need for map-matching speed: Localizing global curve-matching algorithms. In: SSDBM (2006)

[115] Winter, S., Yin, Z.C.: Directed movements in probabilistic time geography. International Journal of Geographical Information Science **24**(9) (2010)

[116] Wivfat, V.T.: A planning tool to assess advanced vehicle sensor technologies on traffic flow, fuel economy, and emissions (2016). MS Thesis, UC Irvine

[117] Wu, C.H., Ho, J.M., Lee, D.T.: Travel-time prediction with support vector regression. IEEE transactions on intelligent transportation systems **5**(4), 276–281 (2004)

[118] Wu, M., Song, X., Jermaine, C., Ranka, S., Gums, J.: A lrt framework for fast spatial anomaly detection. In: Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining, pp. 887–896. ACM (2009)

[119] Yuan, J., Zheng, Y., Xie, X., Sun, G.: Driving with knowledge from the physical world. In: Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '11, pp. 316–324 (2011)

[120] Yuan, J., Zheng, Y., Xie, X., Sun, G.: T-drive: enhancing driving directions with taxi drivers' intelligence. Knowledge and Data Engineering, IEEE Transactions on **25**(1), 220–232 (2013)

[121] Yuan, N.J., Zheng, Y., Xie, X.: Segmentation of urban areas using road networks. MSR-TR-2012–65, Tech. Rep. (2012)

[122] Zandbergen, P.A.: Accuracy of iphone locations: A comparison of assisted gps, wifi and cellular positioning. Transactions in GIS **13**(s1), 5–25 (2009)

[123] Zhang, B., Trajcevski, G., Liu, F.: Clustering speed in multi-lane traffic networks. In: Proceedings of the 25th ACM International on Conference on Information and Knowledge Management, pp. 2045–2048. ACM (2016)

[124] Zhang, B., Trajcevski, G., Liu, L.: Towards fusing uncertain location data from heterogeneous sources. GeoInformatica **20**(2), 179–212 (2016)

[125] Zhang, J., Zheng, Y., Qi, D.: Deep spatio-temporal residual networks for citywide crowd flows prediction. arXiv preprint arXiv:1610.00081 (2016)

[126] Zhang, T., Arrigoni, S., Garozzo, M., Yang, D.g., Cheli, F.: A lane-level road network model with global continuity. Transportation Research Part C: Emerging Technologies **71**, 32–50 (2016)

[127] Zhang, Y., Zhang, Y., Haghani, A.: A hybrid short-term traffic flow forecasting method based on spectral analysis and statistical volatility model. Transportation Research Part C: Emerging Technologies **43**, 65–78 (2014)

[128] Zheng, K., Trajcevski, G., Zhou, X., Scheuermann, P.: Probabilistic range queries for uncertain trajectories on road networks. In: EDBT, pp. 283–294 (2011)

[129] Zheng, Y., Zhang, H., Yu, Y.: Detecting collective anomalies from multiple spatio-temporal datasets across different domains. In: Proceedings of the 23rd SIGSPATIAL International Conference on Advances in Geographic Information Systems, p. 2. ACM (2015)

[130] Zheng, Z., Wang, D., Pei, J., Yuan, Y., Fan, C., Xiao, F.: Urban traffic prediction through the second use of inexpensive big data from buildings. In: Proceedings of the 25th ACM International on Conference on Information and Knowledge Management, pp. 1363–1372. ACM (2016)

[131] Zheng, Z., Wang, D., Pei, J., Yuan, Y., Fan, C., Xiao, F.: Urban traffic prediction through the second use of inexpensive big data from buildings. In: Proceedings of the

25th ACM International on Conference on Information and Knowledge Management, pp. 1363–1372. ACM (2016)

APPENDIX A

Significant Times in Instantaneous Possible Location Query

In section 4 we analyze the boundary of the possible locations at a given time instant under the FB model. The detailed significant times calculation will be presented here. Let d_{min} and d_{max} denote the shortest and longest distance from L_1 to any point $P(t_{s1}, \varepsilon) \in \overline{P_1 P_2}$.

$$t_i^{l1} = \frac{t_1 + t_s}{2} - \frac{d_{max}}{2v_{max}}$$

$$t_i^{lA} = \frac{t_1 + t_s}{2} - \frac{d_{min}}{2v_{max}}$$

$$t_i^{d1} = \frac{t_1 + t_s}{2} + \frac{d_{min}}{2v_{max}}$$

$$t_i^{dA} = \frac{t_1 + t_s}{2} + \frac{d_{max}}{2v_{max}}$$

APPENDIX B

Enter/Exit Time Calculation for Range Query

The general case for time $t \in [t_i, t_{i+1}]$ being a critical point occurs when the intersection of the uncertain region at t with a query rectangle is a single point. In the time interval $[t_i, t_s]$, the single-point-intersection between disk centered at the first GPS point and query region stands for the entering moment. Similarly, in the time interval $[t_s, t_{i+1}]$, the single-point-intersection represents exiting moment. Since the query region is represented as polygon in the (X, Y) plane, each edge of the polygon is defined as a segment of 2D line $y = ax + b$.

The entry boundary of FB is:

$$(x - x_i)^2 + (y - y_i)^2 = (t - t_i)^2 v_{max}^2$$

Substituting for y for the equation of the line, we have:

$$(x - x_i)^2 + (ax + b - y_i)^2 = (t - t_i)^2 v_{max}^2$$

This yields an equation in x and t:

$$A * x^2 + x * (B + C * t) + D * t^2 + E = 0$$

Where A, B, C, D, E are constant. Solving for x, as a function of t, we have:

$$x_{1,2} = \frac{-(B + C * t) \pm \sqrt{(B + C * t)^2 - 4 * A * (D * t^2 + E)}}{2 * A}$$

To be noted that, we need to check the solution for x against the boundaries of the respect edge of the query region. To find the time for critical point, we set the discriminant to be zero:

$$\sqrt{(B + C * t)^2 - 4 * A * (D * t^2 + E)} = 0$$

The real root t_{in} is the time instant when the uncertain trajectory start to enter the query prism.

In the time interval $[t_s, t_{i+1}]$, we can use the similar method to find the exiting time t_{out}.