Map-Matching for Low-Sampling-Rate GPS Trajectories

Yin Lou
Microsoft Research Asia
v-yilou@microsoft.com

Chengyang Zhang
Microsoft Research Asia
v-chenz@microsoft.com

Yu Zheng
Microsoft Research Asia
yuzheng@microsoft.com

Xing Xie
Microsoft Research Asia
xingx@microsoft.com

Wei Wang
Fudan University
weiwang1@fudan.edu.cn

Yan Huang
University of North Texas
huangyan@unt.edu

ABSTRACT

Map-matching is the process of aligning a sequence of observed user positions with the road network on a digital map. It is a fundamental pre-processing step for many applications, such as moving object management, traffic flow analysis, and driving directions. In practice there exists huge amount of low-sampling-rate (e.g., one point every 2-5 minutes) GPS trajectories. Unfortunately, most current map-matching approaches only deal with high-sampling-rate (typically one point every 10-30s) GPS data, and become less effective for low-sampling-rate points as the uncertainty in data increases. In this paper, we propose a novel global map-matching algorithm called *ST-Matching* for low-sampling-rate GPS trajectories. ST-Matching considers (1) the spatial geometric and topological structures of the road network and (2) the temporal/speed constraints of the trajectories. Based on spatio-temporal analysis, a *candidate graph* is constructed from which the best matching path sequence is identified. We compare ST-Matching with the incremental algorithm and Average-Fréchet-Distance (AFD) based global map-matching algorithm. The experiments are performed both on synthetic and real dataset. The results show that our ST-matching algorithm significantly outperform incremental algorithm in terms of matching accuracy for low-sampling trajectories. Meanwhile, when compared with AFD-based global algorithm, ST-Matching also improves accuracy as well as running time.

Categories and Subject Descriptors

H.2.8 [**Database Applications**]: Spatial Databases and GIS.

General Terms

Algorithms, Design

Keywords

Map-matching, GPS, trajectory, road network

1. INTRODUCTION

The past years have seen a dramatic increase of handheld or dashboard-mounted travel guidance systems and GPS-embedded PDAs and smart phones. The proliferation of these devices has enabled the collection of huge amount of GPS trajectories. More and more applications, such as route planner [7], hot route finder [16], traffic flow analysis [15], geographical social network [23], have started to use information from GPS data to achieve better quality of services.

Typically a GPS trajectory consists of a sequence of points with latitude, longitude, and timestamp information. However, this data is not precise due to measurement errors caused by the limitation of GPS devices and sampling error caused by the sampling rate [17]. Therefore the observed GPS positions often need to be aligned with the road network on a given digital map. This process is called *map-matching*. Map-matching is a fundamental pre-processing step for many trajectory-based applications, such as moving object management, traffic flow analysis, and driving directions. The difficulty of map-matching can greatly differ depending on GPS accuracy and the sampling rate.

This paper addresses the problem of *sampling error* in particular. In practice there exists large amount of *low-sampling-rate* (e.g., one point every 2 minutes) GPS trajectories. They are either application-logged data collected from ad-hoc location-based queries, or generated in the scenarios where saving of energy cost and communication cost are desired. For example, there are 60,000+ taxies in Beijing, among which many are GPS-embedded. Since taxi drivers travel very frequently, sampling rate has to be reduced in order to save energy consumption and achieve reasonable response time. Unfortunately, current map-matching approaches only deal with high-sampling-rate (typically one point every 10-30s) GPS data, and become less effective for low-sampling-rate points as the uncertainty in data increases.

Most existing map-matching approaches employ *local* or *incremental* algorithms that map *current* or *neighboring* positions onto vector road segments on a map. For an approach that only considers current positions, the result is greatly affected by measurement errors. The accuracy is generally low because the correlation of neighboring points is completely overlooked. The incremental matching algorithm in [8][7] pursues the local matching of a small portion of the trajectory. When matching a new position, its previous position and last matched edge are considered. Although fast in computation, this approach's performance is sensitive to the decrease of sampling frequency.

On the other hand, a *global* algorithm aligns *entire trajectory* with the road network. Generally speaking, a global approach achieves better accuracy at a higher computational cost. Existing global

ACM GIS '09 , November 4-6, 2009. Seattle, WA, USA (c) 2009 ACM
ISBN 978-1-60558-649-6/09/11...$10.00

matching algorithms are all based on certain distance/similarity measures (such as Average Fréchet distance proposed in [3]). The lack of ground truth (i.e. the "true" path of the moving object) makes it difficult to evaluate the real matching quality. Moreover, current global matching algorithms only employ spatial analysis, while neglect the temporal/speed constraints of the trajectories. This makes them vulnerable to the decrease of sampling rate as well.

In this paper we refer to *low-sampling-rate* as one point every 2 minutes or above. With such sampling rate, the distance between two points may reach over 1300m even a vehicle's speed is only 40km/*h*! This poses a big challenge for the map-matching problem because as the distance between two neighboring points increases, less information can be used to deduce the precise locations of the object. The problem is aggravated when a moving object is travelling with high speed, or there are many intersections between two observed neighboring points.

To address the challenge, we provide two key observations that lead to our approach, as illustrated in the following examples.

Observation 1: True paths tend to be direct, rather than roundabout.

Example 1: Consider the GPS trajectory of a taxi visualized in Figure 1. The taxi travels from north to south. p_a, p_b and p_c are three consecutive sampling points.

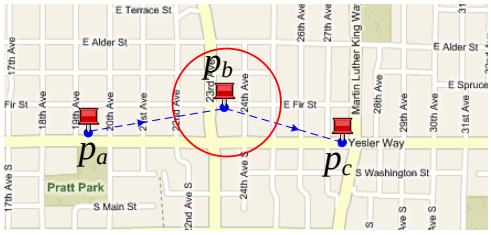


Figure 1. Illustration of observation 1

Most map-matching algorithms would match the circled GPS observation (p_b) to its nearest road segment, i.e., the vertical one. However, from its previous position p_a and future position p_c, we may determine that p_b should be matched to the horizontal road segment because it is unlikely for this taxi to take a roundabout trip to the horizontal road first, and then return to the vertical road. This implies that topological information of the road network can be combined with the position context to provide better matching results.

Observation 2: True paths tend to follow the speed constraints of the road.

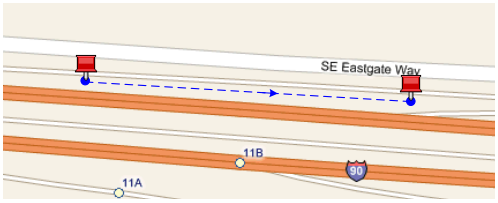


Figure 2. Illustration of observation 2

Example 2: Consider another taxi GPS trajectory visualized in Figure 2. Without speed information, it is nearly impossible to tell whether these two points belong to the highway or the service road. However, if we compute the average speed of this path as 80k*m/h* based on distance of the two points and their timestamps, we may determine that the two points are very likely on the highway. This implies that temporal/speed information is also useful in the matching process.

Based on above observations, we propose a novel global map-matching algorithm called *ST-Matching* for low-sampling-rate GPS trajectories in this paper. To the best of our knowledge, our work is the first map-matching algorithm that incorporates both (1) the spatial geometric and topological structures of the road network and (2) the temporal/speed constraints of the trajectories. By combing spatial and temporal analysis, a candidate graph is constructed from which ST-matching algorithm tries to find a global matching path with highest score.

In summary, this paper makes the following contributions:

- We propose a novel global matching algorithm called ST-matching for low-sampling-rate GPS trajectories. It combines both spatial and temporal features that are not explored by previous methods.
- We perform extensive experiments on both synthetic and real dataset. The real dataset is collected from the physical world and labeled by real people. Therefore, it can better evaluate the performance of map-matching algorithms than synthetic data used in most existing work.
- Our algorithm is evaluated in terms of running time and matching accuracy. The results show that our ST-matching algorithm significantly outperform incremental algorithm in terms of matching accuracy for low-sampling-rate trajectories. Meanwhile, when compared with Average-Fréchet-Distance based global algorithm, ST-Matching also improves accuracy as well as running time.

The rest of the paper is organized as follows. Section 2 discusses the related work. Preliminaries and problem definition are presented in Section 3. System overview is given in Section 4. Then ST-matching algorithm is proposed in Section 5 with detailed discussion and analysis in both Section 5 and 6. Section 7 presents the experimental evaluation results. Finally, we conclude the paper with Section 8.

2. RELATED WORK

In this section, we present the related work of the map-matching problem. Since our algorithm relies on shortest path computation, we will also briefly review the shortest path problem.

2.1 Map-matching Problem

There are a number of studies on matching GPS observations on a digital map. These approaches can be generally classified into three classes: local/incremental method [4][8][20], global method [1][3][21], and statistical method [11][18].

The local/incremental methods try to find *local* match of geometries. The incremental method in [8] uses two similarity measures to evaluate the candidate edges, one for distance similarity and the other for orientation similarity. The combined similarity measure is computed as the sum of individual scores. The time complexity is $O(n)$ once we find adjacent edges for each sample, where n is the number of GPS points to be matched. The "adaptive clipping" method in [20] uses Dijkstra algorithm to construct shortest path on local free space graph. It runs in

$O(mn \log m)$ time, where m and n are number of edges in the road network and number of GPS points respectively. On top of incremental matching, [4] proposes a segment-based matching method to assign confidence values for different sampling points. It matches high-confidence segments first, and then match low-confidence segments using previously matched edges. In general, when match a new position, a local/incremental method only considers a small portion of the trajectory that is close to the position. It runs fast and performs well when sampling frequency is very high (e.g. 2-5 seconds). However as the sampling rate decreases, the problem of "arc-skipping" [8] becomes prominent, causing significant degrade of accuracy. By contrast, with reasonable increase of time complexity, our global ST-matching algorithm is more robust to the decrease of sampling rate.

The global methods aim to match the *entire trajectory* with the road network. Paper [21] employs an offline snapping method that aims to find a minimum weight path based on edit distance. Other methods [1][3] are based on Fréchet distance or its variants. Fréchet distance takes the continuity of curves into account and is therefore suitable for comparing trajectories. In [1], the algorithm applies parametric search over all critical values. Then it solves decision problem by finding a monotone path in the free space from the lower left corner to the upper right corner. It runs in $O(mn \log^2 mn)$ time, where *m* and *n* are the number of edges and number of nodes in the road network respectively. This work is extended in [3] with average Fréchet distance to reduce the effect of outliers. Paper [3] also uses weak Fréchet distance that runs in $O(mn \log mn)$ time with similar matching quality. Global methods aim to minimize the Fréchet distance between the trajectory and the matched road segments. However, current methods have not tried using true paths to evaluate the actual matching accuracy. Meanwhile, the temporal/speed information in the trajectories is generally overlooked. As our experiments demonstrate, when applied to low-sampling rate real data with human labeled true paths, ST-matching outperforms the methods based on Fréchet distance both in terms of matching accuracy and running time.

Statistical models are also used for matching GPS observations. In [18], a map-matching method based on Bayesian classifier is presented that incorporates a Hidden Markov Model to model topological constraints of the road network. An enhanced method based on extended Kalman filter and cubic spline interpolation is proposed in [11]. The statistical approaches seem particularly effective to handle GPS measurement errors, as demonstrated by their experiments. They may be combined with our ST-matching method to achieve better results.

2.2 Shortest Path Computation

Now we briefly review related work on shortest path computation since it is used in our ST-Matching algorithm. This is an area that has received extensive research attention over decades. A good survey can be found in [5].

The basic algorithm in shortest path computation is Dijkstra's algorithm. In practice, A* algorithm [10] is often used as a more efficient alternative. A* algorithm uses heuristic function to guide the search toward the destination. Other strategies such as bidirectional search [12], search decomposition [13], and hierarchical search [14] are often used in real applications as well.

Some pre-processing steps can be added to speed up the shortest path search. For example, ALT algorithms in [6] employ the combination of landmarks and triangular inequality to reach a tighter lower bound than Euclidean distance used in A* algorithm. Reach-based-pruning [9] is another method to compute lower bound for pruning purpose.

3. PROBLEM STATEMENT

In this section, we will give the preliminaries and formally define the problem of map-matching for low-sampling-rate GPS trajectories.

Definition 1 (*GPS Log*): A GPS log is a collection of GPS points $L = \{p_1, p_2, \dots, p_n\}$. Each GPS point $p_i \in L$ contains latitude $p_i.lat$, longitude $p_i.lng$ and timestamp $p_i.t$, as illustrated in the left part of Figure 3.

Definition 2 (*GPS Trajectory*): A GPS Trajectory T is a sequence of GPS points with the time interval between any consecutive GPS points not exceeding a certain threshold ΔT, i.e. $T: p_1 \rightarrow p_2 \rightarrow \cdots \rightarrow p_n$, where $p_i \in L$, and $0 < p_{i+1}.t - p_i.t < \Delta T$ $(1 \leq i < n)$. Figure 3 shows an example of GPS trajectory. ΔT is the sampling interval. In this paper, we focus on low sampling rate GPS trajectories with $\Delta T \geq 2min$.

	Latitude,	Longitude,	Time
p_1:	39.9848,	116.3319,	13:11
p_2:	39.9841,	116.3472,	13:13,
……	……		
p_n:	39.9858,	116.3558,	13:25

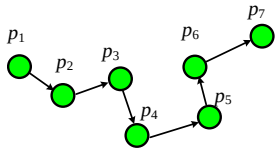


Figure 3. Illustration of GPS log and GPS trajectory

Definition 3 (*Road Segment*): A road segment e is a directed edge that is associated with an id $e.eid$, a typical travel speed $e.v$, a length value $e.l$, a starting point $e.start$, an ending point $e.end$ and a list of intermediate points that describes the road using a polyline. Figure 4 shows several real road segments in Bing Map Search [2]. Note that a road may contain several road segments.

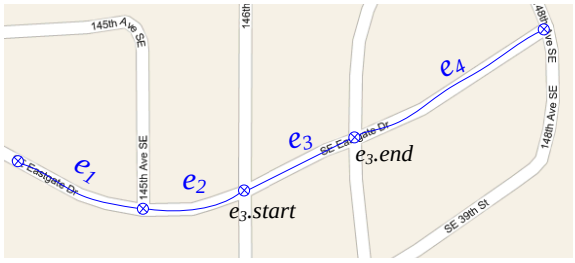


Figure 4. Illustration of road segment

Definition 4 (*Road Network*): A road network is a directed graph $G(V, E)$, where V is a set of vertices representing the intersections and terminal points of the road segments, and E is a set of edges representing road segments.

Definition 5 (*Path*): Given two vertices V_i, V_j in a road network G, a path P is a set of connected road segments that start at V_i and end at V_j, i.e. $P: e_1 \rightarrow e_2 \rightarrow \cdots \rightarrow e_n$, where $e_1.start = V_i$, $e_n.end = V_j$, $e_k.end = e_{k+1}.start$, $1 \leq k < n$.

Now the problem of map-matching is defined as:

Given a raw GPS trajectory T and a road network $G(V, E)$, find the path P from G that matches T with its real path.

4. SYSTEM OVERVIEW

The architecture of our proposed map-matching system is shown in Figure 5. It is composed of three major components: *Candidate Preparation*, *Spatial and Temporal Analysis*, and *Result Matching*.

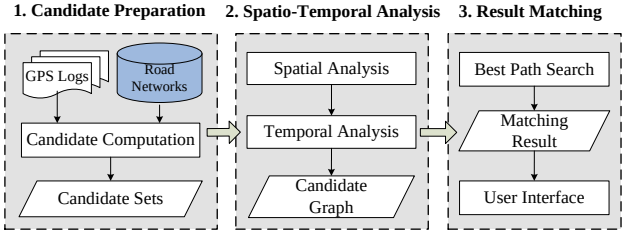


Figure 5. Overview of system architecture

Candidate Preparation This component contains a road network database with indexed edge and vertex information. It accepts given raw GPS trajectory from the user, and then retrieves all the possible candidate points for each sampling point on the trajectory. This step can be efficiently performed with the built-in grid-based spatial index. The output of this component is a set of candidate points and the candidate road segments they lie on.

Spatial and Temporal Analysis This component performs spatial analysis followed by temporal analysis on the retrieved candidate sets and the trajectory to be matched.

- Spatial analysis not only considers the distance between a single GPS point and the candidate road segments for this point, but also takes into account the topological information of the road network. To avoid roundabout paths, we employ shortest path to measure the similarity between each candidate path and the "true" path.
- Temporal analysis measures the actual average travel speed between any neighboring points. It then compares the average speed with the typical speed constraints on each candidate path. The information can later be used to match the trajectory to the candidate path with most similar speed conditions during that time interval.

After spatial and temporal analysis, a candidate graph is constructed as the output of this component. The nodes of the graph are the set of candidate points for each GPS observation, and the edges of the graph are set of shortest paths between any two neighboring candidate points. The nodes and edges are all assigned weight values based on the results of spatial/temporal analysis.

Result Matching This component evaluates the candidate graph using the weight information assigned during spatial/temporal analysis. It matches given trajectory to the path with highest score in the candidate graph. The results are then visualized on an interface that can be tailored towards different end-user devices. The results can also be stored in a traffic database to support external applications such as traffic management or driving directions.

5. The ST-MATCHING ALGORITHM

In this section, we describe our ST-Matching algorithm in details.

5.1 Candidate Preparation

Given trajectory $T = p_1 \rightarrow p_2 \rightarrow \cdots \rightarrow p_n$, we first retrieve a set of candidate road segments within radius r of each point $p_i, 1 \leq i \leq n$. Then we compute candidate points, which are line segment projections of p_i to these road segments defined as follows.

Definition 6 (*Line Segment Projection*): The line segment projection of a point p to a road segment e is the point c on e such that $c = \arg min_{\forall\, c_i \in e}\, dist(c_i, p)$, where $dist(c_i, p)$ returns the distance between p and any point c_i on e.

In the rest of this paper, we use e_i^j and c_i^j respectively to denote the jth candidate edge and candidate point of p_i. As shown in Figure 6, p_i's candidate points are c_i^1,c_i^2 and c_i^3.

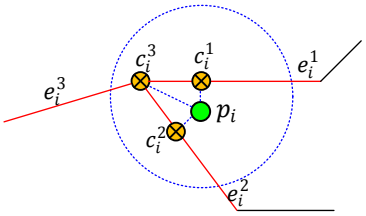


Figure 6 Candidate points for a sampling point p_i

To facilitate the search of candidate points, the road network is indexed using a grid. Once the candidate point sets are retrieved for all the sampling points on the trajectory T, the problem becomes how to choose one candidate from each set so that P: $c_1^{j_1} \rightarrow c_2^{j_2} \rightarrow \cdots \rightarrow c_n^{j_n}$ best matches T: $p_1 \rightarrow p_2 \rightarrow \cdots \rightarrow p_n$.

5.2 Spatial Analysis

In spatial analysis, we use both geometric and topological information of the road network to evaluate the candidate points found in the previous step. The geometric information is incorporated using *observation probability*, and the topological information is expressed using *transmission probability*.

Definition 7 (*Observation Probability*): The observation probability is defined as the likelihood that a GPS sampling point p_i matches a candidate point c_i^j computed based on the distance between the two points $dist(c_i^j,\ p_i)$.

Generally speaking, the error in a GPS measurement can be reasonably described as a normal distribution $N(\mu, \sigma^2)$ of the distance between p_i and c_i^j. It indicates how likely a GPS observation p_i can be matched to a candidate point c_i^j on the real road without considering its neighboring points. Formally, we define observation probability $N(c_i^j)$ of c_i^j w.r.t. p_i as:

$$N(c_i^j) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(x_i^j - \mu)^2}{2\sigma^2}} \qquad (1)$$

where $x_i^j = dist(c_i^j,\ p_i)$ is the distance between p_i and c_i^j. In this paper, we use a zero-mean normal distribution with a standard deviation of 20 meters based on empirical evaluation.

Observation probability does not take into account a GPS point's position context. This sometimes leads to wrong matching results. Figure 7 shows such an example. The thick lines represent a highway and the thin vertical line represents a local road. Although p_i is closer to c_i^1 on the local road, we should

match p_i to c_i^2 on the highway if we already know that its neighbors p_{i-1} and p_{i+1} are on the highway. This is based on the observation that a vehicle is unlikely to take a roundabout path (*Observation 1*).

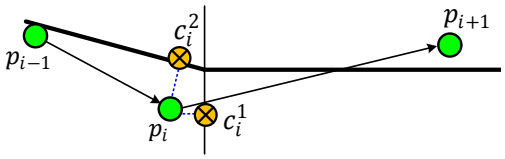


Figure 7. An example that needs transmission probability

To characterize the above intuition, we compute shortest path between two neighboring candidate points c_{i-1}^t and c_i^s. Then we define t*ransmission probability* as follows:

Definition 8 (*Transmission Probability*): Given two candidate points c_{i-1}^t and c_i^s for two neighboring GPS sampling points p_{i-1} and p_i respectively, the transmission probability from c_{i-1}^t to c_i^s is defined as the likelihood that the "true" path from p_{i-1} to p_i follows the shortest path from c_{i-1}^t to c_i^s.

We compute transmission probability as

$$V(c_{i-1}^t \to c_i^s) = \frac{d_{i-1\to i}}{w_{(i-1,t)\to(i,s)}} \qquad (2)$$

where $d_{i-1\to i} = dist(p_i, p_{i-1})$ is the Euclidean distance between p_i and p_{i-1}, and $w_{(i-1,t)\to(i,s)}$ is the length of shortest path from c_{i-1}^t to c_i^s.

Combining Equation (1) and (2), we define the **spatial analysis function** $F_s(c_{i-1}^t \to c_i^s)$ as the product of observation probability and transmission probability:

$$F_s(c_{i-1}^t \to c_i^s) = N(c_i^s) * V(c_{i-1}^t \to c_i^s),\ 2 \le i \le n \qquad (3)$$

where c_{i-1}^t and c_i^s are any two candidate points for two neighboring GPS points p_{i-1} and p_i respectively.

Equation (3) computes the likelihood that an object moves from c_{i-1}^t to c_i^s using the product of two probability functions, thus geometric and topological information are both taken into consideration. Note that in practice it is unlikely for a moving object to always follow the strict shortest path. Therefore the observation probability $N(c_i^s)$ cannot be omitted from (3).

With spatial analysis, for any two neighboring GPS points p_{i-1} and p_i, a set of candidate paths $c_{i-1}^t \to c_i^s$ are generated. Each path is assigned a spatial measurement value computed from Equation (3).

5.3 Temporal Analysis

Spatial analysis can distinguish the actual path from the other candidate paths in most cases. However, there are situations that spatial analysis alone could not handle.

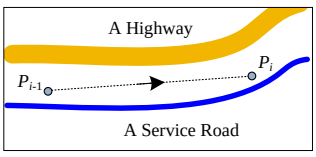


Figure 8. An example that needs temporal analysis

Consider the example shown in Figure 8. The thick line is a highway, and the thin lines are service roads close to the highway. The spatial analysis function may produce the same value whether two points p_{i-1} and p_i are matched to the highway or the service road. However, if we calculate the average speed from p_{i-1} to p_i as $80\text{km}/h$, we would match them to the highway considering the speed limits of the service road (*Observation 2*).

More formally, given two candidate points c_{i-1}^t and c_i^s for two neighboring GPS sampling points p_{i-1} and p_i respectively, the shortest path from c_{i-1}^t to c_i^s is denoted as a list of road segments $[e_1', e_2', \dots, e_k']$. The average speed $\bar{v}_{(i-1,t)\to(i,s)}$ of the shortest path is computed as follows:

$$\bar{v}_{(i-1,t)\to(i,s)} = \frac{\sum_{u=1}^{k} l_u}{\Delta t_{i-1\to i}} \qquad (4)$$

.

where $l_u = e_u'.l$ is the length of e_u', and $\Delta t_{i-1\to i} = p_i.t - p_{i-1}.t$ is the time interval between two sampling points p_i and p_{i-1}. Note that each road segment e_u' is also associated with a typical speed value $e_u'.v$. We employ cosine distance to measure the similarity between the actual average speed from c_{i-1}^t to c_i^s and the speed constraints of the path. Consider the vector that contains k elements of the same value $\bar{v}_{(i-1,t)\to(i,s)}$ and the vector $(e_1'.v, e_{21}'.v, \dots, e_k'.v)^{\mathrm{T}}$. The **temporal analysis function** is defined as follows.

$$F_t(c_{i-1}^t \to c_i^s) = \frac{\sum_{u=1}^{k}(e_u'.v \times \bar{v}_{(i-1,t)\to(i,s)})}{\sqrt{\sum_{u=1}^{k}(e_u'.v)^2} \times \sqrt{\sum_{u=1}^{k} \bar{v}_{(i-1,t)\to(i,s)}^2}} \qquad (5)$$

As in spatial analysis function, c_{i-1}^t and c_i^s are any two candidate points for p_{i-1} and p_i respectively.

5.4 Result Matching

With the spatial and temporal analysis above, we are ready to describe our *ST-Matching* algorithm used in the result matching component.

In general, after spatial and temporal analysis, we are able to generate a candidate graph $G_T'(V_T', E_T')$ for trajectory $T: p_1 \to p_2 \to \cdots \to p_n$. V_T' is a set of candidate points for each GPS sampling point, and E_T' is a set of edges representing the shortest paths between any two neighboring candidate points, as depicted in Figure 9. Each node in G' is associated with $N(c_i^s)$. Each edge is associated with $V(c_{i-1}^t \to c_i^s)$ and $F_t(c_{i-1}^t \to c_i^s)$.

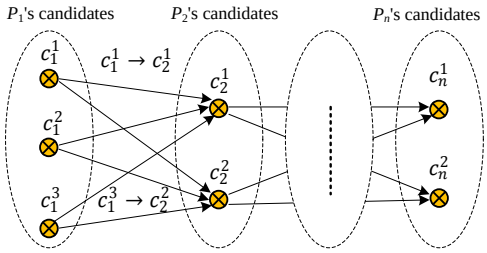


Figure 9. The candidate graph $G_T'(V_T', E_T')$

Combining Equation (3) and (5), we define **ST function** for $c_{i-1}^t \to c_i^s$ as:

$$F(c_{i-1}^t \to c_i^s) = F_s(c_{i-1}^t \to c_i^s) * F_t(c_{i-1}^t \to c_i^s), 2 \le i \le n \qquad (6)$$

A candidate path sequence P_c for the entire trajectory T is a path in the candidate graph, denoted as $P_c: c_1^{s_1} \rightarrow c_2^{s_2} \rightarrow \cdots \rightarrow c_n^{s_n}$. The overall score for such a candidate sequence is $F(P_c) = \sum_{i=2}^{n} F\left(c_{i-1}^{s_{i-1}} \rightarrow c_i^{s_i}\right)$. From all the candidate sequences we aim to find the one with the highest overall score as the best matching path for the trajectory. More formally, the best matching path P for a trajectory T is selected as:

$$P = arg\,max_{P_c} F(P_c), \qquad \forall P_c \in G_T'(V_T', E_T') \qquad (7)$$

Algorithm 1 outlines the framework of ST-Matching algorithm. We first compute the candidate point sets for each GPS sampling point on T. Then we construct the candidate graph $G_T'(V_T', E_T')$ based on spatial and temporal analysis. Finally, the algorithm reports the path sequence P with highest ST-function value from G_T' as the result.

Algorithm 1 ST-Matching Algorithm

Input: Road network G, a trajectory $T: p_1 \rightarrow p_2 \rightarrow \cdots \rightarrow p_n$.

Output: The matched sequence $P: c_1^{j_1} \rightarrow c_2^{j_2} \rightarrow \cdots \rightarrow c_n^{j_n}$ in G

1: Initialize $tList$ as an empty list; // a list of candidate points
2: for $i = 1$ to n do
3: s = GetCandidates(p_i, G, r); // candidates within radius r
4: $tList$.add(s);
5: G_T' = ConstructGraph($tList$); // constructs graph G_T'
6: return FindMatchedSequence(G_T');

The *FindMatchedSequence* procedure aims to find the longest path in G_T'. In general, finding longest path in a graph is an NPC problem. However, G_T' is a directed acyclic graph (DAG) and the answer can be computed efficiently using the topological order of the graph. It is easy to see that such topological order can be obtained from the construction of G_T'. Algorithm 2 shows the details of the procedure *FindMatchedSequence*.

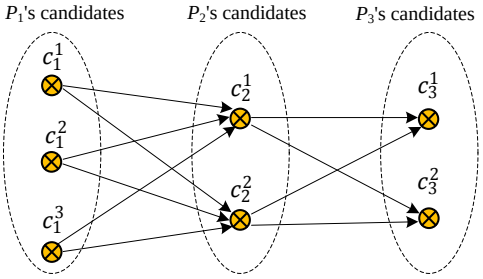
Algorithm 2 FindMatchedSequence

Input: Candidate graph $G_T'(V_T', E_T')$

Output: The longest sequence $P: c_1^{s_1} \rightarrow c_2^{s_2} \rightarrow \cdots \rightarrow c_n^{s_n}$ in G_T'

1: Let $f[\,]$ denote the highest score computed so far;
2: Let $pre[\,]$ denote the parent of current candidate;
3: **for each** c_1^s **do**
4: $f[c_1^s] = N(c_1^s)$;
5: **for** $i = 2$ to n **do**
6: **for each** c_i^s **do**
7: $max = -\infty$;
8: **for each** c_{i-1}^t **do**
9: $alt = f[c_{i-1}^t] + F(c_{i-1}^t \rightarrow c_i^s)$;
10: **if** ($alt > max$) **then**
11: $max = alt$;
12: $pre[c_i^s] = c_{i-1}^t$;
13: $f[c_i^s] = max$;
14: Initialize $rList$ as an empty list; // a list of matched points in reverse order
15: $c = arg\,max_{c_n^s}(f[c_n^s])$;
16: **for** $i = n$ to 2 **do**
17: $rList$.add(c);
18: $c = pre[c]$;
19: $rList$.add(c);
20: **return** $rList.reverse()$;

Figure 10 shows a running example of the above procedure. The value of observation probability, transmission probability and temporal measurement function are also listed in the figure.



N:

c_1^1	c_1^2	c_1^3	c_2^1	c_2^2	c_3^1	c_3^2
0.8	0.2	0.5	0.6	0.6	0.4	0.3

(V, F_t):

	$\rightarrow c_2^1$	$\rightarrow c_2^2$
c_1^1	(0.5,0.5)	(0.8,0.5)
c_1^2	(0.3,0.4)	(0.1,0.9)
c_1^3	(0.4,0.6)	(0.9,0.9)

	$\rightarrow c_3^1$	$\rightarrow c_3^2$
c_2^1	(0.2,0.6)	(0.1,0.7)
c_2^2	(0.3,0.3)	(0.5,0.9)

Figure 10. An example of *FindMatchedSequence* procedure

The algorithm first fills out the table $f[\,]$ as follows.

$f[\,]$:

c_1^1	c_1^2	c_1^3	c_2^1	c_2^2	c_3^1	c_3^2
0.8	0.2	0.5				

Next consider candidate c_2^1. The algorithm first evaluates the path $c_1^1 \rightarrow c_2^1$. We have $F(c_1^1 \rightarrow c_2^1) = 0.6 \times 0.5 \times 0.5 = 0.15$. Similarly, $F(c_1^2 \rightarrow c_2^1) = 0.07$ and $F(c_1^3 \rightarrow c_2^1) = 0.144$. Therefore $f[c_2^1] = \max\{0.8 + 0.15, 0.2 + 0.072, 0.5 + 0.144\} = 0.95$, and c_2^1 's parent is c_1^1. We repeat the above process for all the candidates and complete the table $f[\,]$.

$f[\,]$:

c_1^1	c_1^2	c_1^3	c_2^1	c_2^2	c_3^1	c_3^2
0.8	0.2	0.5	0.95	1.04	1.076	1.175

We find that c_3^2 has the highest overall score, and we choose c_3^2 as the matching result for the last GPS point. Based on the stored parent information, the algorithm finally reports the matching result as $c_1^1 \rightarrow c_2^2 \rightarrow c_3^2$.

5.5 Localizing ST-Matching Strategies

The proposed ST-Matching algorithm is a global algorithm as we can only identify the best path sequence after computing overall score for the entire trajectory. In practice, when a trajectory has too many points (i.e. n is very large), or online processing is desired, we may localize the ST-Matching algorithm by construct a partial candidate graph $G_{T[w]}'$ over a sliding window of the trajectory, where $w (w < n)$ is the window size.

As illustrated in Figure 11, each partial candidate graph $G_{T[w]}'$ is constructed from a subset of trajectory T. We find the best matching path sequence for this subset similarly as in global ST-matching. Then we slide the window and repeat the process.

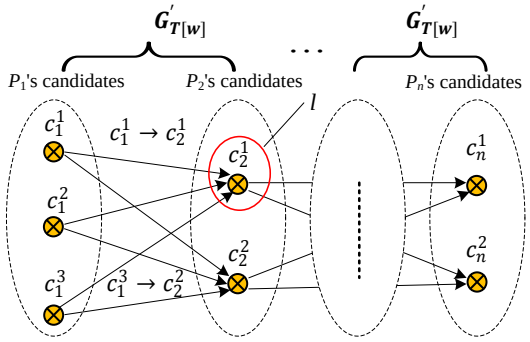


Figure 11. Localizing ST-Matching ($w = 2, l = 1$)

Partitioning trajectory into windows can reduce the average delay and save storage space for online processing, but does not necessarily speed up the overall processing time because the most expensive part of ST-Matching algorithm is shortest path computation. To reduce computation complexity, we may heuristically retain candidate points with top l score so far, thus reducing the number of shortest-paths that need to be computed for next sampling point. When $l = 1$, ST-Matching degenerates into an incremental algorithm.

6. ALGORITHM ANALYSIS

Now we analyze the time complexity of our proposed ST-Matching algorithm. Let n denote the number of sampling points on the given trajectory, and m denote the number of road segments in the road network. We further assume that the maximum number of candidates of one sampling point is k.

Notice that the number of shortest paths in G'_T is $(n-1)k^2$. The time complexity for constructing the candidate graph is therefore $O(nk^2 m \log m)$. For the *FindMatchedSequence* procedure, each edge in G'_T is visited exactly once. The time complexity is $O(nk^2)$. Therefore our global ST-Matching algorithm has the time complexity of $O(nk^2 m \log m + nk^2)$.

Note that for any given sampling point, we can practically choose a small value of k. Thus the time complexity is close to $O(nm \log m)$. By contrast, the global algorithm in [7] has the complexity of $O(nm\,(log\,mn)^2)$ using Fréchet Distance and $O(nmlogmn)$ using Weak Fréchet Distance.

In practice, Dijkstra is often not a good choice to compute shortest path. We may choose A* [10] or ALT algorithm [6] to achieve a better running time. A* algorithm uses estimates on distances to the destination to guide vertex selection in a search from the source. To achieve even faster running time, ALT algorithm employs the combination of landmarks and triangular inequality to get a tighter lower bound, but it requires more disk space for pre-processing. Moreover, the localizing strategies proposed in Section 5.5 are also useful to reduce the response time and improve computation efficiency.

7. EXPERIMENTS

In this section, we first present the experimental settings, including the dataset we used and some parameters we selected in the experiment. Next we describe the evaluation approaches that we applied. Then we report the major results with some discussions. Our experimental results are compared with both incremental algorithm and Average-Fréchet-Distance (AFD) based global map-matching algorithm. The results are mainly evaluated in terms of running time and matching accuracy.

7.1 Experimental Settings

7.1.1 Dataset Description

Road Network: In our experiments, we use the road network of Beijing as visualized in Figure 12. The network graph contains 58,624 vertices and 130,714 road segments.

Figure 12. Road network of Beijing

Synthetic Trajectory Data: The synthetic data used in our experiments contains trajectories randomly generated by our own simulator. The simulator works as follows. It first randomly selects two vertices in the road network and compute top K shortest paths between them. Then it randomly select a trajectory from the K paths as the ground truth, denoted as G: $e_1, e_2, \dots, e_n$. The motivation behind this is that moving objects generally follow the direction from source to destination, but not necessarily follow the shortest path strictly. Note that the time interval between any two neighboring points is not uniform. To retrieve a trajectory with desired sampling interval, the simulator select one road segment from every k' segments on G, i.e., it select $e_1, e_{1+k'}, \dots, e_{1+mk'}$, where $m = \left\lfloor \frac{n}{i} \right\rfloor$. The adjustment of sampling rate is therefore achieved by changing the value of k'. The simulator generates one GPS point with estimated timestamp information for each selected road segment. The points are produced to follow the zero-mean normal distribution with the standard deviation of 20 meters.

Real Trajectory Data: Different from existing approaches, we also use human labeled *true path* data as the **ground truth** in our experiments. From hundreds of real trajectories collected from GeoLife system [22], we select 28 trajectories that diverse in spatial coverage, length, time interval and number of road segments. They are manually labeled with the true paths. Figure 13 shows the histogram of the real dataset characteristics.

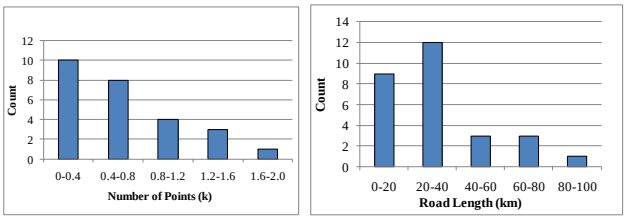


Figure 13. Histogram of real dataset

Figure 14 shows an example of real trajectory visualized using Bing Map API. The blue lines connected by green markers represent the GPS trajectory, and the red lines connected by red markers represent the true paths of the user.

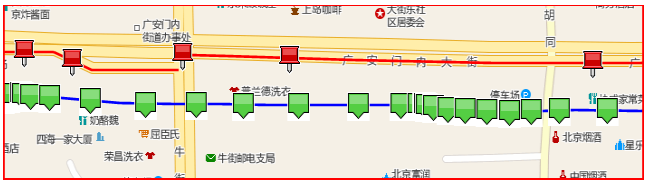


Figure 14. An example of labeled trajectory data in GeoLife[22]

7.1.2 Parameter Selection

Data Sampling Rate: This paper focuses on low-sampling-rate GPS trajectories. The sampling interval ranges from 2 minutes to nearly 6 minutes for synthetic data, and ranges from 30 seconds to 5 minutes for real trajectory data.

Parameters for ST-Matching Algorithm: In our experiments, we choose $k = 5$ empirically as the maximum number of candidates of any sampling points. The candidate search radius is selected as $r = 100m$. For observation probability estimation, we use a normal distribution with $\mu = 0$ and $\delta = 20\,m$. When evaluating localized strategies, the window size ranges from 1 to 40, and we choose to retain top *1, 3, 5* nodes.

Parameters for Baseline Algorithms: We compare our experiments with two baseline algorithms in this paper. For the incremental algorithm, we use the empirical settings $\mu_d = 10$, $a = 0.17$, $n_d = 1.4$, $\mu_\alpha = 10$, and $n_\alpha = 4$ as suggested in [3]. For AFD-based algorithm, we set $\lambda = 10\ meters$.

7.2 Evaluation Approaches

Ground Truth: In synthetic data, the ground truth is randomly selected among the top *K* shortest paths from the source to the destination. In real data collected from GeoLife, the ground truth is established using human labeled true paths. During the labeling process, if a user cannot identify the actual road segment that the sampling point resides, such point is labeled with a special tag so that later on it will not be counted in the evaluation.

Evaluation Criteria: Our ST-Matching algorithm is evaluated both in terms of running time and matching quality. The running time is measured using the actual program execution time. The matching quality is measured using two accuracy metrics *Accuracy by Number* (A_N) and *Accuracy by Length* (A_L) defined as follows.

$$A_N = \frac{\#\ correctly\ matched\ road\ segments}{\#\ all\ road\ segments\ of\ the\ trajectory}$$

$$A_L = \frac{\sum The\ length\ of\ matched\ road\ segments}{The\ length\ of\ the\ trajectory}$$

Baselines: We mainly compare our ST-Matching algorithm with two baselines: an incremental algorithm that performs matching based on a point's previous matching result; and a global matching algorithm that aims to minimize Average-Fréchet-Distance (AFD). To evaluate the effects of temporal analysis, we developed an "S-Matching" algorithm by removing temporal analysis from ST-Matching. S-Matching is also compared with the original ST-Matching method.

7.3 Experimental Results

7.3.1 Results on Synthetic Data

Impact of Max. Number of Candidate Points: Figure 15 and 16 shows the impact of maximum number of candidate points on matching accuracy and running time of ST-Matching. In the figures, "Inf" means that we do not limit the number of candidates.

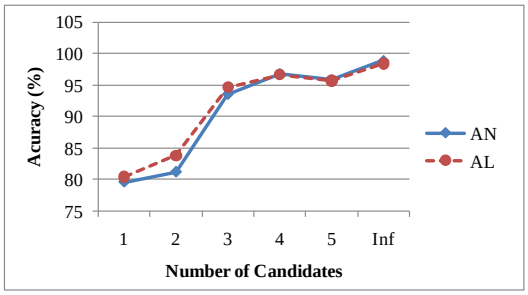


Figure 15. Accuracy w.r.t. number of candidates

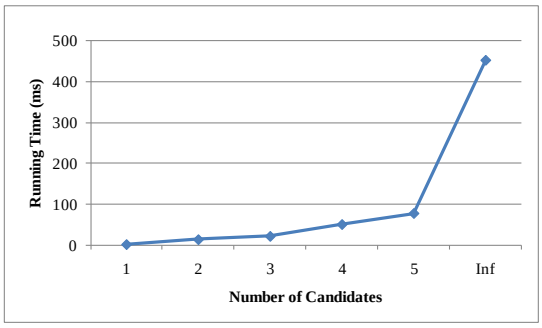


Figure 16. Running time w.r.t. number of candidates

As shown in Figure 15, the accuracy generally increases as the algorithm takes more candidate points into consideration. However, a large number of candidate points for one GPS point would lead to a huge amount of shortest path computation, which will significantly increase running time. Figure 16 shows that if we limit the number of candidates to 5, the average running time of our map-matching algorithm is acceptable.

ST-Matching vs. S-Matching:
Next we evaluate the effects of temporal analysis by comparing the matching quality of ST-Matching and S-Matching w.r.t to sampling interval.

In synthetic GPS trajectory generation, the change of sampling interval is achieved through the adjustment of parameter k' as discussed in Section 7.1.1. We choose k' as 9, 11, 13, 15, and 17 respectively. The value of sampling interval for each value of i is listed in Table 1.

Table 1. Sampling interval in synthetic data

k'	9	11	13	15	17
Interval (min)	2.91	3.42	4.14	5.12	5.77

We compute top 5 shortest paths. For each k', we randomly generate 10 GPS trajectories and test our map-matching algorithms. We then take the average accuracy.

As shown in Table 2, the accuracy generally goes down as sampling rate decreases. This is because the shortest path assumption will no longer hold. We also observe that ST-Matching outperform S-Matching most of the time, indicating that temporal constraints are indeed useful in most cases.

Table 2. ST-Matching vs. S-Matching (accuracy)

k'	A_N		A_L	
	S-Matching	ST-Matching	S-Matching	ST-Matching
9	0.917	**0.935**	0.938	**0.954**
11	0.893	**0.913**	0.920	**0.944**
13	0.895	**0.891**	0.928	**0.926**
15	0.837	**0.855**	0.890	**0.896**
17	0.803	**0.823**	0.843	**0.863**

ST-Matching v.s. Incremental v.s. AFD-based global algorithm

Next we compare our ST-Matching algorithm with incremental and AFD-based global map-matching algorithm. As Figure 17 and 18 shows, ST-Matching outperforms both incremental and ADF-based algorithm significantly. Meanwhile we notice that the performance of incremental algorithm degenerates sharply when sampling rate decreases while global algorithms are more robust to the change of sampling rate.

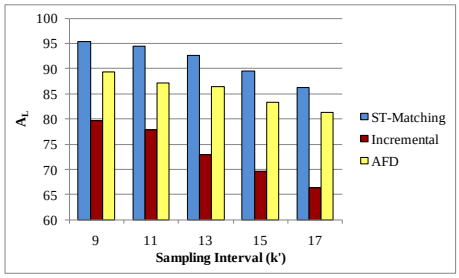


Figure 17. A_L(%) w.r.t sampling interval

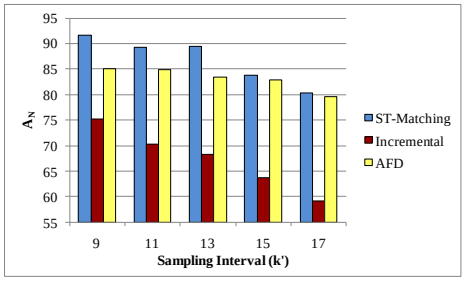


Figure 18. A_N (%) w.r.t sampling interval

7.3.2 Results on Real Data

Now we evaluate ST-Matching algorithm on real data collected from GeoLife.

ST-Matching v.s. Incremental v.s. AFD-based global algorithm

Figure 19 and 20 shows the change of matching quality w.r.t to the sampling rate on real data. It can be seen clearly that our ST-Matching algorithm outperforms both incremental and AFD-based algorithm on real data. As the sampling rate decreases, the accuracy of incremental algorithm drops sharply, while two global algorithms again exhibit more consistent performance. We also notice that the matching accuracy of two global algorithms tends to be very close when the sampling interval is high. This implies that temporal constraints become less effective when neighboring points are far away from each other.

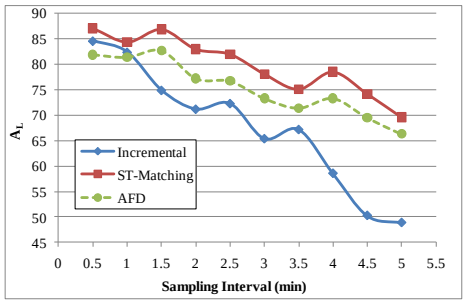


Figure 19. A_L w.r.t sampling interval on real data

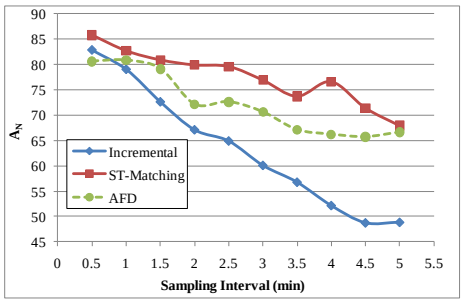


Figure 20. A_N w.r.t sampling interval on real data

Running Time

In order to evaluate the efficiency of ST-Matching algorithm, we compare the running time of our algorithm with the AFD-based global method. As demonstrated by Figure 21, when number of points is small, the two methods exhibit similar running time. However, as the number of points increase, the running time of ADF-based approach increases dramatically.

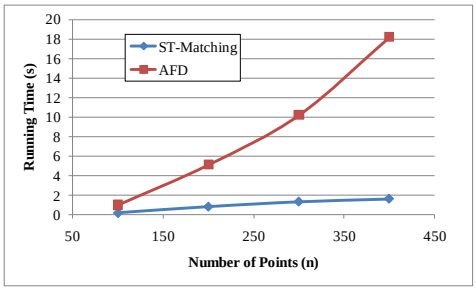


Figure 21. Running time comparison on real data

Localizing Strategies

Finally we evaluate the effect of localizing strategies proposed in Section 5.5 with one real trajectory of over 400 points. From Figure 22 it can be seen that in general when the window size reaches a certain value, the further increase of window size has little effect in improving the accuracy. In some cases, the increase of window size may even have adverse effect on matching accuracy. This implies that localized ST-Matching can be practically used as long as we select an appropriate window size. On the other hand, retaining top *l* nodes will significantly reduce the matching accuracy especially when window size is large.

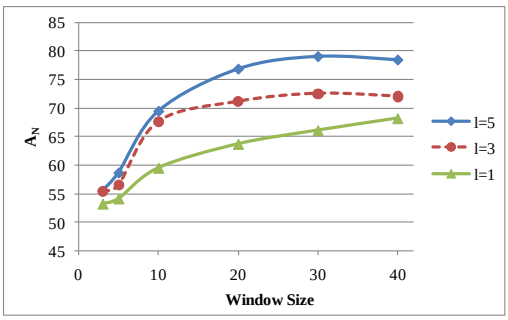


Figure 22. Evaluation of localizing Strategies

8. CONCLUSIONS

In this paper, we propose a new global map-matching algorithm called ST-Matching to match low-sampling-rate GPS data onto a digital map. The algorithm employs spatial and temporal analysis to generate a candidate graph, from which a sequence of matched results with highest sum of score is identified as the matching result. The experiment results demonstrate that our ST-matching algorithm significantly outperforms incremental algorithm in terms of matching accuracy for low-sampling trajectories. Meanwhile, when compared with ADF-based global algorithm, ST-Matching also improves accuracy as well as running time.

In our future work, we plan to build a full-fledged map-matching system based on ST-Matching algorithm. The first application of such system will be taxi trajectory matching that can help in driving directions based on taxi-driver's knowledge.

9. REFERENCES

[1] Alt, H., Efrat, A., Rote, G., and Wenk, C. 2003. Matching planar maps. In Journal of Algorithms 49, 2, 262-283, 2003.

[2] Bing maps search (formerly Live maps search). http://cn.bing.com/ditu/. Microsoft Corporation.

[3] Brakatsoulas, S., Pfoser, D., Salas, R., and Wenk, C. 2005. On map-matching vehicle tracking data. In Proceedings of the 31st international Conference on Very Large Data Bases, 853-864, 2005.

[4] Chawathe, S.S. Segment-based Map Matching. In IEEE Symposium on Intelligent Vehicles, 2007.

[5] Fu, L., Sun, D., and Rilett, L. R. Heuristic shortest path algorithms for transportation applications: state of the art. In Comput. Oper. Res. 33, 11, 3324-3343, 2006.

[6] Goldberg, A. V. and Harrelson, C. 2005. Computing the shortest path: A* search meets graph theory. In Proceedings of the Sixteenth Annual ACM-SIAM Symposium on Discrete, 156-165, 2005.

[7] Gonzalez, H., Han, J., Li, X., Myslinska, M., and Sondag, J. P. 2007. Adaptive fastest path computation on a road network: a traffic mining approach. In Proceedings of the 33rd international Conference on Very Large Data Bases 794-805, 2007.

[8] Greenfeld, J. Matching GPS observations to locations on a digital map. In Proceedings of 81th Annual Meeting of the Transportation Research Board, 2002.

[9] Gutman, R. Reach-based routing: a new approach to shortest path algorithms optimized for road networks. In Proceedings of 6th ALENEX, 100–111, 2004.

[10] Hart, N. Nilsson, and B. Raphael. A formal basis for the heuristic determination of minimum cost paths. In IEEE Transactions on system science and cybernetics, 4:100-107, 1968.

[11] Hummel, B., and Tischler, K. GPS-only map matching: exploiting vehicle position history, driving restriction information and road network topology in a statistical framework. In Proceedings of the GIS Research UK Conference, 68–77, 2005.

[12] JAT, Nicholson. Finding the shortest route between two points in a network. In Computer Journal, 9, 275-80, 1996.

[13] J.F., Dillenburg, P.C., Nelson. Improving search efficiency using possible subgoals. In Mathematical and Computer Modeling, 22(2-7):379-414, 1995.

[14] Jing, N., Huang, Y., and Rundensteiner, E. A. Hierarchical optimization of optimal path finding for transportation applications. In Proceedings of the Fifth international Conference on information and Knowledge Management , 261-268, 1996.

[15] Kuehne, R. et al. New approaches for traffic management in metropolitan areas. In 10th IFAC Symposium on Control in Transportation Systems, 2003.

[16] Li, X., Han, J., Lee, J., and Gonzalez, H. Traffic density-based discovery of hot routes in road networks. In 10th International Symposium on Spatial and Temporal Databases, 2007.

[17] Pfoser, D. and Jensen, C. S. Capturing the uncertainty of moving-object representations. In Proceedings of the 6th international Symposium on Advances in Spatial Databases, 111-132, 1999.

[18] Pink, O. Hummel, B. A statistical approach to map matching using road network geometry, topology and vehicular motion constraints. In the 11th International IEEE Conference on Intelligent Transportation Systems, 2008.

[19] Samet, H., Sankaranarayanan, J., and Alborzi, H. Scalable network distance browsing in spatial databases. In Proceedings of the 2008 ACM SIGMOD international Conference on Management of Data, 43-54, 2008.

[20] Wenk, C., Salas, R., and Pfoser, D. 2006. Addressing the need for map-matching speed: localizing global curve-matching algorithms. In Proceedings of the 18th international Conference on Scientific and Statistical Database Management, 2006.

[21] Yin, H. and Wolfson, O. A Weight-based map matching method in moving objects databases. In Proceedings of the 16th international Conference on Scientific and Statistical Database Management, 437, 2004.

[22] Zheng, Y., Chen, Y., Xie, X., and Ma, W. GeoLife2.0: a location-based social networking service. In proceedings of International Conference on Mobile Data Management, 2009.

[23] Zheng, Y., Wang, L., Xie, X., and Ma, W. GeoLife-managing and understanding your past life over maps. In 9th International Conference on Mobile Data Management, 2008.