Information Sciences 433–434 (2018) 55–69

Contents lists available at ScienceDirect

Information Sciences

journal homepage: www.elsevier.com/locate/ins

Map matching for low-sampling-rate GPS trajectories by exploring real-time moving directions

Yu-Ling Hsueh*, Ho-Chian Chen

Department of Computer Science & Information Engineering, National Chung Cheng University, Taiwan

A R T I C L E   I N F O

Article history:
Received 27 March 2017
Revised 15 November 2017
Accepted 23 December 2017
Available online 25 December 2017

Keywords:
Map matching
GPS data analysis
Location-based services

A B S T R A C T

Map matching is the process of matching a series of recorded geographic coordinates (e.g., a GPS trajectory) to a road network. Due to GPS positioning errors and the sampling constraints, the GPS data collected by the GPS devices are not precise, and the location of a user cannot always be correctly shown on the map. Therefore, map matching is an important preprocessing step for many applications such as navigation systems, traffic flow analysis, and autonomous cars. Unfortunately, most current map-matching algorithms only consider the distance between the GPS points and the road segments, the topology of the road network, and the speed constraint of the road segment to determine the matching results. Moreover, most current map-matching algorithms cannot handle the matching errors at junctions. In this paper, we propose a spatio-temporal based matching algorithm (STD-matching) for low-sampling-rate GPS trajectories. STD-matching considers (1) the spatial features such as the distance information and topology of the road network, (2) the speed constraints of the road network, and (3) the real-time moving direction which shows the movement of the user. Moreover, we also reduce the running time by performing GPS clustering, GPS smoothing, and the A^* shortest path algorithms. In our experiments, we compare STD-matching with three existing algorithms, the ST-matching algorithm, the stMM algorithm, and the HMM-RCM algorithm, using a real data set. The experiment results show that our STD-matching algorithm outperforms the three existing algorithms in terms of matching accuracy.

1. Introduction

With the popularity of mobile devices embedded with GPS (e.g., smartphones, personal navigation devices), a large number of GPS trajectory data are able to be collected by these devices nowadays. An increasing number of applications (such as route planners, traffic flow analysis, geographical social network analysis, and autonomous cars) provide location-based services according to these trajectory data. However, the surrounding environments, especially in urban areas, can influence the GPS data and produce errors that make the data inaccurate. Because of these GPS positioning errors, there may be inaccurate results if these trajectory data are used directly. Therefore, the map-matching technique, which is a process of associating a sequence of positions with the road network in a digital map, is considered as a basic preprocessing step for many applications. After we obtain the graphs transformed from the GPS data through the map-matching process, we can use graph algorithms to efficiently solve various problems.

* Corresponding author.
E-mail addresses: hsueh@cs.ccu.edu.tw (Y.-L. Hsueh), chc103m@cs.ccu.edu.tw (H.-C. Chen).

https://doi.org/10.1016/j.ins.2017.12.031
0020-0255/© 2017 Elsevier Inc. All rights reserved.

There are several factors affecting the accuracy of map-matching algorithms. First, the sampling rate of GPS data has a great influence on matching accuracy. With high-sampling-rate GPS trajectories, more sampling points are given to form the trajectory. Therefore, highly accurate matching results can be produced. However, because of the strict limits of batteries, hand-held devices embedded with GPS are unable to collect high-sampling-rate data for a long period of time. Therefore, matching the low-sampling-rate GPS trajectories on maps has become an important issue. However, when dealing with low-sampling-rate GPS data which fetches a location every minute or even more, the location uncertainty increases as the sampling rate reduces. This uncertainty makes matching a low-sampling-rate trajectory a challenging issue. The second factor is the complexity of the road network. For example, a map of a countryside area is usually a less complicated road network; thus, one can easily match GPS trajectories to the correct road segments. On the contrary, while handling maps in an urban area, the density of the road network is higher, and the road network is thus more complicated. Therefore, the complexity of matching GPS trajectories to the correct road segments becomes challenging, because ambiguous cases of matching are more likely to occur (e.g., ambiguity between a highway and general road when they run parallel to each other).

Although in [12] the issues of low-sampling-rate trajectories and complex road networks have been studied using spatial and temporal features, the proposed method, called ST-matching, still has some problems. First, the ST-matching algorithm cannot handle the matching problem at the beginning of the GPS trajectory. It only uses distance information to address this problem. Second, ST-matching cannot handle the GPS points around the junctions, sometimes matching to a wrong road segment. Third, when a trajectory consists of too many GPS points, the system may incur significant computation time to match the trajectory. In this paper, we propose a novel map-matching algorithm that considers the spatial, temporal, and directional features. The directional analysis function using real-time moving directional data is introduced. With these data, we can solve the matching problem, particularly at junctions and at the beginning of the GPS trajectory. Then, we perform GPS clustering, GPS smoothing, and the A^* shortest path algorithms to reduce the computational cost. The main contributions of this work are summarized as follows:

- We propose a spatio-temporal-based matching algorithm (STD-matching) for low-sampling-rate GPS trajectories by performing a sequence of processes including GPS clustering, GPS smoothing, and the A^* shortest path algorithms for reducing the number of input GPS points on trajectories so as to reduce the computational cost.
- Our STD-matching approach considers various spatio-temporal features such as distance, speed constraints, and the topology of the road network. Furthermore, we adopt the real-time moving direction for map matching. Our experiments indicate that STD-matching achieves up to 92.97% accuracy for high-sampling-rate (under 1 minute) GPS trajectories and up to 81.75% accuracy for low-sampling-rate (one to two minutes) GPS trajectories.
- We collected real-world trajectories using a data collector running on an Android cell phone which records various types of sensor data including longitude, latitude, timestamp, and real-time moving directional data for experiments. We have compared our work with three existing methods [10,12,19]. The experimental results show that our STD-matching approach outperforms the existing approaches on both high and low-sampling-rate GPS trajectories.

The rest of the paper is organized as follows. Section 2 discusses the related work. Section 3 introduces the definitions used in this paper and defines the map-matching problem. Subsequently, the spatio-temporal based matching algorithm (STD-matching) is proposed in Section 4. Section 5 evaluates the performance and presents the experiment results. Finally, we make our conclusions and discuss the future work in Section 6.

2. Background and related work

To solve the map-matching problem, a tremendous number of map-matching algorithms have been proposed lately. In the early stage, the most intuitive idea was to consider the closeness and the shape of the road segments. The basic approach, called point-to-point matching, was proposed in [1] which matches each GPS point to the nearest node in the road network. On the other hand, point-to-curve matching [23] finds the nearest point along the road segment to match with. Finally, Phuyal et al. proposed curve-to-curve matching [18]. Their algorithm firstly identifies the candidate nodes with point-to-point matching. Secondly, it constructs the curve with the GPS trajectory and the curve with the candidate road segments corresponding to the road network, and compares the distance between two curves to find the most similar and the closest road segment as a best match.

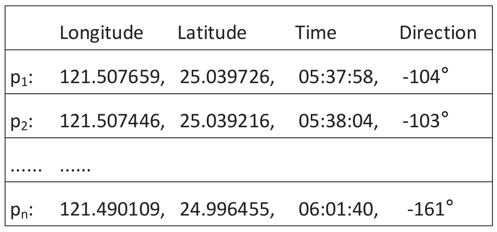
However, the algorithms mentioned above only consider the closeness and the shape of the road segments without considering their connectivity, which has several shortcomings that make the algorithms inappropriate in practice. For example, these algorithms have incorrect results in urban areas because of the high density of the road networks. Moreover, the closest road segment may not always be the correct one to match with. Therefore, to overcome this problem, the approaches proposed in [5,20] make use of the closeness and shape of the road segments as well as the connectivity of the road network. In [5], a weighted topological algorithm using the topology of the road network and the observation position of the vehicle was proposed, while in [20], Quddus et al. developed an enhanced topological map-matching algorithm based on the road network geometry to derive navigation data. The algorithms considering the connectivity of the road network usually model the constraints with speed, travel time, or shortest path distance to filter out outliers to achieve better map-matching results. Therefore, a map-matching algorithm is known as a topological map-matching algorithm when it utilizes the geometry and the connectivity of the road segments.

However, most existing algorithms do not work well when matching on the junctions, because it is challenging to determine which road segment to match with at a junction by only considering the closeness and topology of the road network. In these situations, the system takes the next GPS point and historical trajectory into consideration to identify the correct road segments. Therefore, a number of algorithms which require a reasonable amount of data (such as geometric data, the topology of the road network, and other sensor data) to model the probabilities have been proposed. However, the accuracy of these probabilistic algorithms relies on how to model realistic uncertainty. In [8], Honey et al. introduced the technique of defining an elliptical or rectangular confidence region around a position. In [24], Zhao et al. used this technique in the case of GPS and suggested that the error variance is associated with the map-matching solution of the GPS position. In the case of the distance between the candidate point and the GPS point being considered, Gaussian distribution is always adopted to model the distance error region with different probabilities. However, there are still many other data such as the connectivity of the road network and the speed of the vehicle that can be modelled into probability to improve the map-matching algorithms.

Later, the mathematical models such as the Kalman Filter, the particle filter, fuzzy logic, belief theory, and the Hidden Markov Model (HMM) were utilized to improve the matching accuracy. In particular, the HMM algorithm has been widely adopted for map matching recently because it considers all the states from the first point to the end point of the GPS trajectory. In [15], Newson et al. proposed a map-matching algorithm based on HMM. In their paper, emission probabilities which model GPS noise as zero-mean Gaussian were utilized, while transition probabilities were modelled as this distance difference probability (i.e., the distance difference between GPS displacement and the match route distance). The authors found that the correct pairs of matched points typically result in a very short route. Finally, the Viterbi algorithm was used to compute the optimal path. To simplify the complexity of the HMM model, in [13], Mohamed et al. proposed an algorithm which uses three filters (i.e., a speed filter, a direction filter, and an α-trimmed mean filter) to reduce the candidate sets for improving the efficiency of the map-matching process. In [6], Han et al. combined the road segments to reduce the capacity of the road network, and used a Hilbert R-tree to index the road network to improve the running time of searching the road network. In [10], Jagadeesh et al. designed an HMM-based map-matching method to generate a partial path with a high certainty by utilizing the Viterbi algorithm. The partial path is then defined as the pre-identified first alternative path alone with other $k-1$ alternative paths obtained by inflating the travel time on each link of the paths in the route choice set for map matching. The best matched path is selected from these alternative paths based on the route choice and observation generation probabilities, which take the real drive data into account. However, the matched results are not output until the convergence state is detected. As a result, the system incurs high latency and fails to provide responsive results. Mohamed et al. [14] proposed a map-matching SnapNet system for cellular-based trajectories which might be far from the actual road segments. An incremental HMM algorithm was utilized to provide real-time matches after a series of filters was applied to reduce the noise in the row data. Subsequently, the Viterbi algorithm calculates the most possible road segment for map matching.

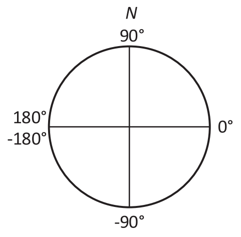
In [2,7,12,17,21], in order to improve the accuracy of the map-matching algorithm, various parameters and algorithms were used to enhance the scoring function. In [7], for example, He et al. used distance weight, direction weight, and connection weight to model the scoring function, while in [21], Rahmani et al. considered the number of traffic lights and number of left turns which a vehicle meets to enhance the accuracy of the matching results. Rahmani et al. found that these two parameters affect the travel time of the GPS trajectory, and they enhanced the scoring function with this observation. In [17], in order to deal with the GPS data which were collected with a low-sampling rate and with a high level of noise (i.e., high GPS errors), the idea of cumulative proximity-weight formulation was proposed to improve the accuracy of the map-matching algorithm. However, the authors only considered the candidate positions of the previous point to perform spatial analysis and temporal analysis for matching. In [2], Goh et al. trained the transition probabilities with a Support Vector Machine (SVM) to enhance its accuracy. In [12], Lou et al. proposed a map-matching algorithm for low-sampling-rate GPS trajectories. The authors modelled the temporal analysis with speed and travel time data to improve its accuracy. Nikoli et al. [16] utilized a genetic algorithm that uses path cost and the shape similarity between the observed routes and GPS trajectories for the fitness function. The path cost is computed using A^* Algorithm, while the shape similarity value is computed based on the dynamic time wrapping technique. Hu et al. [9] fused multiple types of information including speed and direction when selecting a road. The method considers the global effect that moving objects which travel together should have a similar moving speed. Additionally, a historical model was built to estimate the historical speed and current surrounding speed used by spatial and temporal analyses. The ST-CRF map-matching method [11] firstly utilizes the influencing factor based upon the spatial distribution of the middle points between two consecutive GPS points for map matching on low-sampling-rate GPS trajectories. Similar to most of the existing work, both spatial and temporal accessibility between two GPS points in addition to the consistency of driving direction are considered in the ST-CRF model, which is basically an HMM-based model with a matching procedure based on the Viterbi algorithm.

The approaches in [2,7,21,22] were desgined mainly to improve the best path search step to reduce the computation time. In [22], Wang et al. proposed an algorithm based on the Ski-rental model, called Eddy, which includes solid error and delay-bound analysis. In different environments, this algorithm considers the tradeoff between accuracy and latency. Therefore, the algorithm dynamically changes the size of the sliding window to retrieve the best match from the candidate graph. In [7], He et al. introduced the confidence points (i.e., the GPS point which has only one candidate road segment) and the Maximum Delay Constraint Dynamic Time Window (MDCDTW) (i.e., a sliding window with a time constraint) to reduce the computation time. In [2], Goh et al. proposed an online map-matching algorithm based on HMM, which uses a variable



	Longitude	Latitude	Time	Direction
p_1:	121.507659,	25.039726,	05:37:58,	-104°
p_2:	121.507446,	25.039216,	05:38:04,	-103°
……	……			
p_n:	121.490109,	24.996455,	06:01:40,	-161°

(a) GPS trajectory



(b) Real-time moving direction data

Fig. 1. An illustration of GPS trajectory and direction data.

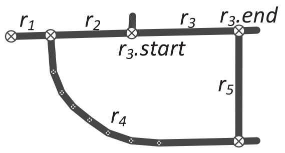


Fig. 2. An example of a road segment.

sliding window (VSW) and a precomputed SVM-training probability function to improve the running time. AntMapper was proposed in [4], which considers both local topological information and global shape similarity measures. The main concept is to use the ant colony optimization algorithm to find the best path that matches the GPS trajectory on the map. In [21], Rahmani et al. improved the computation time in [12] by using a local search tree and topological ordering to realize on-line map matching. However, the approaches [2,7,22] which use a sliding window to speed up the running time do not take the time of waiting for GPS records into account. Therefore, these algorithms are not real-time algorithms.

In the previous research, numerous methods with very different approaches to map matching have been proposed. Most of these map-matching approaches consider the distance, topology and speed characteristics of a GPS point, such as the distance between the GPS point and the selected candidate point, the shortest path between two neighbouring GPS points in a road network, and the similarity between the road speed constraint and the user speed. However, there are still many unresolved issues such as improving the accuracy of map matching for low-sampling-rate GPS trajectories, developing efficient on-line map-matching algorithms, and matching errors at junctions. In this paper, we combine all the distance, topology and speed characteristics, and additionally use a new feature, real-time moving directional data, to improve the accuracy of map matching. Our approach has better performance in terms of accuracy than previous works. Moreover, we also improve the running time by using GPS clustering, GPS smoothing, and the A^* shortest path algorithm.

3. Problem definition

In this section, we define the terms and the notations used throughout this paper and provide the definitions for the map-matching problem.

Definition 1. GPS point: A GPS point is a pair of longitude and latitude coordinates with a timestamp collected by an embedded GPS device. In this paper, we additionally collect the GPS real-time moving direction of each GPS point. Therefore, each GPS point p_i consists of longitude $p_i.lng$, latitude $p_i.lat$, timestamp $p_i.t$, and GPS real-time moving direction $p_i.b$, as in the example shown in Fig. 1. In Fig. 1(a), each row represents a GPS point. Fig. 1(b) illustrates the quantified real-time moving directions.

Definition 2. GPS trajectory: A GPS trajectory T consists of a series of continuous GPS points, and each time interval between any two adjacent GPS points does not exceed a certain value ΔT, which represents the sampling interval; i.e., $T : p_1 \rightarrow p_2 \rightarrow \ldots \rightarrow p_n$, and $0 < p_{i+1}.t - p_i.t \leq \Delta T \quad (1 \leq i \leq n)$.

Definition 3. Road segment: A road segment r is a directed edge. Each road segment is associated with a unique id $r.id$, a travelling speed constraint $r.v$, a length value $r.l$, a starting point $r.start$, an ending point $r.end$, a Boolean value indicating whether the road segment is a two-way street, and a list of intermediate points on the road segment. Fig. 2 shows an example of a 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 terminal points for intersections of the road segments, and E is a set of edges which represents the road segments.

Definition 5. Path: A path P is a set of connected road segments that start at a vertex v_i and end at another vertex v_j, i.e., $P : r_1 \rightarrow r_2 \rightarrow \ldots \rightarrow r_n$, where $r_1.start = v_i$, $r_n.end = v_j$, $r_k.end = r_{k+1}.start$, $1 \leq k < n$, and $v_i, v_j \in V$.

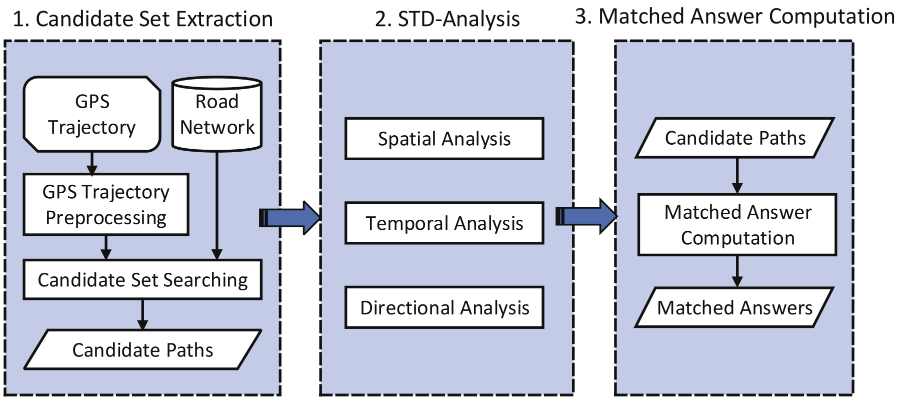


Fig. 3. An overview of the system architecture.

A ground truth, $GT_i : r_1 \to r_2 \to \ldots \to r_n$, is a path representing the real path passed by a user during the trajectory, $T_i : p_1 \to p_2 \to \ldots \to p_n$. Each GPS point p_j, in GPS trajectory T_i has a real matched road segment $GT_i.r_j$, i.e., the actual road segment where the user is located at $p_j.t$ is $GT_i.r_j$, $1 \le j \le n$. A matched answer, $MA_i : r_1 \to r_2 \to \ldots \to r_n$, is a set of road segments of a trajectory T_i produced by a map-matching algorithm. Each GPS point p_j in trajectory T_i has a corresponding matched road segment $MA_i.r_j$, $1 \le j \le n$. In this paper, we use S_i^j to represent the correctness of the matched result $MA_i.r_j$ and TS_i to indicate the total score of the matched result of T_i. The equation is shown as follows:

$$\begin{cases} S_i^j = 1, & if\ GT_i.r_j = MA_i.r_j \\ S_i^j = 0, & otherwise \end{cases}$$

$$TS_i = \sum_{j=1}^{n} S_i^j \tag{1}$$

where $n = |T_i|$ is the number of GPS points.

Assuming that there are m trajectories, the *accuracy* is calculated by the following equation:

$$Accuracy = \sum_{i=1}^{m} \frac{TS_i}{|T_i|} \tag{2}$$

The problem statement of the map matching is defined as: Given raw GPS trajectories T and a road network $G(V, E)$, find all the matched answers MA from G that match T with the objective of maximizing the accuracy defined in Eq. (2).

4. The STD-matching algorithm

In this section, we outline an overview of our map-matching algorithm. The architecture is shown in Fig. 3, where three major components are composed: *Candidate Set Extraction*, *STD-Analysis*, and *Matched Answer Computation*. First, we extract a candidate point set of each GPS point from the road network. Second, we perform STD-analysis which considers spatial, temporal, and directional information. Next, each candidate path which is the shortest path between two neighbouring candidate points is assigned the three weightings obtained from the STD-analysis. Finally, we combine the STD-analysis to retrieve the matched answer of the GPS trajectory from the candidate paths. In the following sections, we describe each component of our STD-Matching algorithm in detail.

4.1. Candidate set extraction

In this section, we give the preliminaries before performing our map-matching algorithm. This section includes two small components: GPS trajectory preprocessing (Section 4.1.1) and candidate set searching (Section 4.1.2). First, given a raw GPS trajectory from the user that contains consecutive GPS points with longitude, latitude, timestamps, and real-time moving directions, the preprocessing component first processes the raw trajectory. Subsequently, given a preprocessed GPS trajectory and the corresponding road network, the searching component of the candidate sets retrieves all the possible candidate points for each GPS point on the trajectory. The output of this section is a set of candidate points and their corresponding candidate road segments. After this step, the system can narrow down the search area of each GPS point for matching to the ground truth.

Algorithm 1 *GPSPreprocessing* (T).

```
Input: a raw GPS trajectory $T : p_1 \to p_2 \to \ldots \to p_n$
Output: a list of the processed GPS trajectory $T' : p_1 \to \ldots \to p_m$
1: let $pre[]$ denote the previous point of a given GPS point
2: initialize $groupList$   /*store the GPS points in the same group*/
3: initialize $cList$   /*store the GPS trajectory after clustering*/
4: for each GPS point $p \in T$ do
5:     Let $d = distance(p, pre[p])$ be the moving distance
6:     Let $v = \frac{d}{(p.t - pre[p].t)}$ be the moving speed
7:     if $v < v_c$ then
8:         if $d \le D_c$ then
9:             $groupList.\text{add}(p)$
10:        else
11:            $p_r = findREP(groupList)$ and $cList.\text{add}(p_r)$
12:            empty $groupList$ and $groupList.\text{add}(p)$
13:        end if
14:        if $p$ is the last point in $T$ then
15:            $p_r = findREP(groupList)$ and $cList.\text{add}(p_r)$
16:        end if
17:    end if
18: end for
19: $T' = \text{Smooth}(cList, M_d)$
20: return $T'$
```

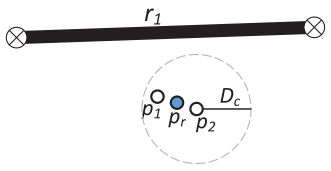


Fig. 4. An example of GPS point clustering.

4.1.1. GPS trajectory preprocessing

We conduct the following procedures as shown in Algorithm 1 to preprocess the GPS points from the raw GPS trajectories before the main map-matching procedure is performed. First, we identify the outliers in the GPS trajectories (Lines 5–7). Because of the GPS errors in urban areas, GPS records are sometimes very far from the actual location of the user. Therefore, the GPS points that suddenly jump to physically unreachable locations are considered as outliers. In Line 6, we compute the travelling speed to determine the outliers based on the Euclidean distance and time interval between two consecutive GPS points. In this paper, we use the travelling speed constraint v_c and remove those GPS points for which the travelling speed exceeds v_c. Second, we cluster the GPS points in the raw GPS trajectory ((Lines 8–16). The GPS trajectory of a user driving in an urban area would pass through many traffic lights. In the trajectory data sets, when encountering a traffic light, the user stops and waits for one to two minutes. In this case, the difference in the coordinates between the GPS points where the user waits for the traffic light is within a very small range. Therefore, we cluster the GPS points that are close to their neighbouring GPS points in order to reduce the computational cost and to obtain a set of trajectories with less noise. Furthermore, we merge the GPS points for which the moving distance between the current point and the previous point is less than or equal to the cluster distance threshold D_c into a group. In Lines 11 and 15, we perform the *findREP* function to obtain a representative GPS point p_r by computing the average longitude, latitude and timestamp among all GPS points in the same group to represent the points of the group. Furthermore, we use the matched result of the representative GPS to also be the matched result of the GPS points that are in the same group. As in the example of GPS points clustering shown in Fig. 4, the distance between GPS points p_1 and p_2 is less than the cluster distance threshold D_c. Therefore, p_r is the representational GPS point of p_1 and p_2. Road segment r_1 is the matched result of p_r, and the matched answers of p_1 and p_2 are r_1 as well.

After clustering the GPS points in the trajectory, we perform a smoothing procedure to reduce the noise of the trajectory. In Line 19, we smooth the input trajectory *cList* with a slope difference constraint M_d as in the example shown in Fig. 5. For every five consecutive GPS points, we compute the slope m_1 of the first two GPS points and the slope m_2 of the last two GPS points. If the difference between m_1 and m_2 is less than or equal to the slope difference constraint M_d, we replace p_i with p_i' which is the intermediate point of p_{i-1} and p_{i+1}. For simplicity, in the following section, we regard a GPS trajectory T as a pre-processed GPS trajectory T'.

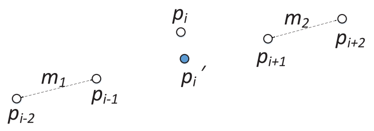


Fig. 5. An example of GPS trajectory smoothing.

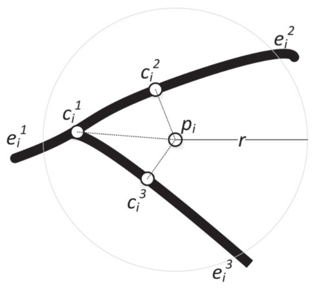


Fig. 6. The candidate set $\{c_i^1, c_i^2, c_i^3\}$ for a GPS point p_i within radius r.

4.1.2. Candidate set searching

Searching for a matched road segment is time consuming when a road network is large. Hence, we retrieve a set of candidate points for each GPS point before finding the matched answers. Given a GPS trajectory T: $p_1 \to p_2 \to \cdots \to p_n$, we first search for a set of candidate road segments e_i^j for GPS point p_i within radius r, where $1 \le i \le n$ and j is an index of candidate road segments. Next, we compute the candidate points which have the minimum distance from the candidate road segments e_i^j to p_i with projection. The candidate points are chosen by the following principles: if the projection of p_i onto the candidate road segment is located between its start point and end point, we choose the projection point as the candidate point; otherwise, we choose the endpoint that is nearest to p_i. In this paper, we use e_i^j and c_i^j to respectively denote the jth candidate road segment and the candidate point of p_i. As in the example shown in Fig. 6, c_i^1, c_i^2, and c_i^3 are the candidate points of the GPS point p_i by a range query within radius r.

To improve the efficiency of searching for candidate points, we use a grid to index the road network. After retrieving all the candidate point sets of the GPS points on trajectory T, we choose one candidate from each set to compute the best matched answer. That is to say, the map-matching problem can be simplified as $P: c_1^{j_1} \to c_2^{j_2} \to \cdots \to c_n^{j_n}$ best matches T: $p_1 \to p_2 \cdots \to p_n$.

4.2. STD analysis

After candidate set extraction is performed, we obtain the candidate set of each GPS point and the candidate path between any two adjacent candidate points. In order to choose the best matched answer from these candidate paths, we consider the spatial, temporal, and directional information to model the score function for these candidate paths. This section is composed of three analyses: spatial analysis, temporal analysis, and directional analysis. First, spatial analysis takes the distance information and the topology of the road network into consideration. Subsequently, temporal analysis considers the speed constraint of the road segments. Finally, directional analysis utilizes the real-time moving direction and models the directional analysis function. With these three analyses, the output of this component is a candidate graph, and each node and edge of the graph is assigned a score.

4.2.1. Spatial analysis

In spatial analysis, we consider both the geometrical and topological information of the road network to determine the matching probability of the candidate points found in the previous step. For the geometrical information, we use *observation probability* (Definition 6); for the topological information, we use *transition probability* (Definition 7).

Definition 6. Observation probability: Observation probability is used to represent the likelihood that a GPS point p_i matches to a candidate point c_i^j based on the distance between the two points $distance(p_i, c_i^j)$.

Despite the error of GPS measurement, the real position of a GPS point can still be considered within a certain range around the measured GPS point. If the candidate point is far from the measured GPS point, the likelihood of the measured GPS point matching to the candidate point is lower. Therefore, the observation probability can be reasonably formalized as a zero-mean normal distribution $N(\mu, \sigma^2)$ with a standard deviation σ. Then, we define observation probability $O(c_i^j)$ as the following equation:

$$O(c_i^j) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(d_i^j-\mu)^2}{2\sigma^2}}, \tag{3}$$

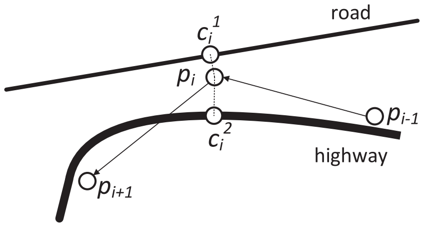


Fig. 7. An example of using transition probability.

where $d_i^j = distance(p_i, c_i^j)$ is the distance between p_i and c_i^j.

Besides considering the geometrical information, we also consider the topological information of the road network to increase the accuracy of the matching results. For example, in Fig. 7, the thin line is a local road segment and the thick line is a highway road segment. These two types of road segment are not connected to each other. $p_{i-1} \to p_i \to p_{i+1}$, which is a user's GPS trajectory, needs to be matched on the map. If we only consider the observation probability, p_i would be matched to c_i^1. However, a vehicle is unlikely to take a long distance path from a highway to a local road and then go back to the highway later. Therefore, given that p_{i-1} and p_{i+1} are matched to the highway, p_i should be matched to c_i^2 rather than to c_i^1.

To solve the problem in Fig. 7, we calculate the shortest path between two neighbouring candidate points c_{i-1}^t and c_i^s. Subsequently, we define *transition probability* as follows:

Definition 7. Transition probability: Transition probability is defined as the possibility that the moving path of a vehicle from p_{i-1} to p_i follows the shortest path from c_{i-1}^t to c_i^s. With the distance ratio of $distance(p_{i-1}, p_i)$ and the shortest path from c_{i-1}^t to c_i^s, we formalize the transition probability as the following equation:

$$T(c_{i-1}^t, c_i^s) = \frac{distance(p_{i-1}, p_i)}{SP(c_{i-1}^t, c_i^s)}, \tag{4}$$

where $SP(c_{i-1}^t, c_i^s)$ is the length of the shortest path from c_{i-1}^t to c_i^s. In this paper, we perform the A^* search algorithm [3] to reduce the computational cost of searching for the shortest path between two neighbouring candidate points.

By integrating Eqs. (3) and (4), we define the spatial analysis function $F_s(c_{i-1}^t \to c_i^s)$ as the following equation:

$$F_s(c_{i-1}^t \to c_i^s) = O(c_i^s) * T(c_{i-1}^t \to c_i^s), \quad 2 \le i \le n, \tag{5}$$

where c_{i-1}^t and c_i^s are two neighbouring candidate points of GPS points p_{i-1} and p_i. When dealing with the first point p_1 of GPS trajectory T, Eq. (5) would be changed to $F_s(c_1^s) = O(c_1^s)$. Eq. (5) computes the probability of a vehicle moving from c_{i-1}^t to c_i^s based on the geometrical and topological information. After spatial analysis, we can obtain a set of candidate paths $c_{i-1}^t \to c_i^s$ between any two adjacent GPS points p_{i-1} and p_i, and finally, each candidate path is given a spatial analysis score by Eq. (5).

4.2.2. Temporal analysis

In spite of the spatial analysis considering the geometrical and topological information, there are some situations that spatial analysis could not handle. For example, a local road segment and a highway road segment are close to each other and have a similar shape. In this situation, the spatial analysis function would produce the same probability of whether the trajectory matches the local road or the highway. However, the travelling speeds of these two types of road segment are different. If a vehicle is moving at a speed of 90 km/h, we would match it to the highway.

In general, the average travel speed is similar to the road speed constraint. The observation of this behavior can be used to identify the candidate path with the most similar speed condition during the time interval. Therefore, we take the speed constraints of the road segments into consideration to solve this problem. Given two candidate points c_{i-1}^t and c_i^s of the GPS points p_{i-1} and p_i, the shortest path from c_{i-1}^t to c_i^s can be represented as a list of road segments $[r_1', r_2', \cdots, r_k']$. First, we compute the average speed of the shortest path by the following equation:

$$\bar{v}_{c_{i-1}^t \to c_i^s} = \frac{\Sigma_{u=1}^k r_u'.l}{\Delta t_{p_{i-1} \to p_i}}, \tag{6}$$

where $\Sigma_{u=1}^k r_u'.l$ is the total length of the shortest path from c_{i-1}^t to c_i^s, and $\Delta t_{p_{i-1} \to p_i} = p_i.t - p_{i-1}.t$ is the time interval between GPS points p_{i-1} and p_{i-1}. Additionally, each road segment r_u' in the road network is assigned a speed constraint $r_u'.v$. Then, we can compute the similarity between the average speed and the speed constraint of the shortest path based

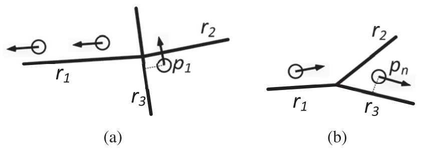


Fig. 8. Two examples of matching near a junction with directional analysis.

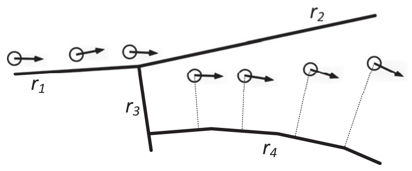


Fig. 9. An illustration of directional analysis.

on cosine similarity. Finally, the temporal analysis function can be formulated as the following equation:

$$F_t(c^t_{i-1} \to c^s_i) = \frac{\Sigma^k_{u=1}(r'_u.v \times \overline{v}_{c^t_{i-1}\to c^s_i})}{\sqrt{\Sigma^k_{u=1}(r'_u.v)^2} \times \sqrt{\Sigma^k_{u=1}\overline{v}^2_{c^t_{i-1}\to c^s_i}}}, \quad 2 \le i \le n, \tag{7}$$

where c^t_{i-1} and c^s_i are the two neighbouring candidate points of GPS points p_{i-1} and p_i. With the temporal analysis, we can reduce the wrong matching result between two types of road segment associated with different speed constraints.

4.2.3. Directional analysis

After the spatial analysis and temporal analysis, most map-matching problems can be correctly solved. However, it is still difficult to determine the matching result at junctions. In this section, we introduce the directional analysis to address this problem. Most of the previous approaches do not leverage the directional data, or just obtain the direction by calculating the link between two adjacent GPS points. However, the direction of two adjacent GPS points linking together does not represent the true moving direction of the GPS points. Therefore, we collect the real-time moving direction $p_i.b$, which is the acceleration direction of the GPS point p_i at timestamp $p_i.t$, from the magnetic field sensor and accelerometer which are embedded in the smartphone. As a result, we are able to compute the similarity between the real-time moving direction and the road segment direction.

Two examples of map matching near a junction are shown in Fig. 8. The arrow line represents the real-time moving direction of each GPS point. Considering the example shown in Fig. 8(a), p_1 is the first point of a GPS trajectory, and has no previous GPS point. Without real-time moving directional data, there is confusion for matching p_1 to road segment r_2 or r_3. In Fig. 8(b), p_n is the last point of a GPS trajectory, and has no next GPS point. Similar to the previous example, it is hard to match p_n. Moreover, transition probability using topological information cannot be used in these two examples. Consequently, p_1 and p_n can only utilize the observation probability using geometrical information to determine the matching results. However, if we take the real-time moving direction of each GPS point into consideration, we could match p_1 to r_3 in Fig. 8(b) and p_n to r_3 in Fig. 8(b).

As such, we further compute the angular difference between the real-time moving direction of a GPS point and the angle of a candidate road segment. Given a candidate point c^j_i for a GPS point p_i, e^j_i is the candidate road segment where c^j_i is located. Therefore, the road angle is computed as the following Eq. (8):

$$\theta_{e^j_i} = arctan(\frac{e^j_i.end.lat - e^j_i.start.lat}{e^j_i.end.lng - e^j_i.start.lng}), \tag{8}$$

where $e^j_i.end.lat$ is the latitude coordinate of the end point on e^j_i, and $e^j_i.start.lng$ is the longitude coordinate of the start point on e^j_i. The angular difference is defined as follows:

$$\Delta\theta(c^j_i) = |\theta_{e^j_i} - p_i.b|, \tag{9}$$

where $p_i.b$ is the real-time moving direction of the GPS point p_i. For example, in Fig. 9, when considering just the spatial and temporal analysis function alone, it is very likely to match the trajectory to the path $r_1 \to r_2$. However, if we calculate the angular difference $\Delta\theta(c^j_i)$ between the real-time moving direction $p_i.b$ and the angle of road segment $\theta_{e^j_i}$, we would match the trajectory to the path $r_1 \to r_3 \to r_4$.

In order to normalize the angular difference, we exploit normal distribution $N(\mu, \sigma^2)$ with a standard deviation σ. The normal distribution indicates how likely it is that a GPS point p_i can be matched to a candidate point c^j_i with directional

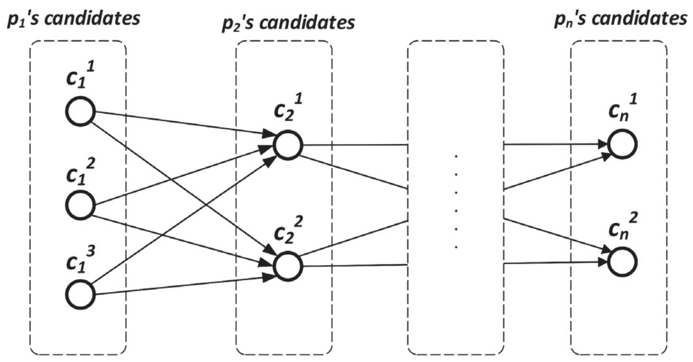


Fig. 10. An illustration of candidate points and candidate paths.

information. Formally, we define the directional analysis function as the following equation:

$$F_b(c_i^j) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(\Delta\theta(c_i^j)-\mu)^2}{2\sigma^2}} \tag{10}$$

With directional analysis, the real-time moving direction and the direction of the road segment are taken into consideration. Each candidate point c_i^j is assigned a directional analysis score computed based on Eq. (10).

4.3. Matched answer computation

In this section, we combine the spatial, temporal, and directional analysis that have been described in the previous section to retrieve the best matched answers for GPS trajectories. After the candidate set is extracted, we generate a candidate graph with the candidate points c_i^s and candidate paths $c_{i-1}^t \to c_i^s$ as shown in Fig. 10. After STD-analysis, each candidate point c_i^s is assigned a score based on the observation probability $O(c_i^s)$ and the directional analysis function $F_b(c_i^s)$. Each candidate path $c_{i-1}^t \to c_i^s$ is assigned a score based on the transition probability $T(c_{i-1}^t \to c_i^s)$ and the temporal analysis function $F_t(c_{i-1}^t \to c_i^s)$.

Finally, we combine the Eqs. (5), (7) and (10) to form the STD-function (defined in Eq. (11)) to find the matched answer for candidate path $c_{i-1}^t \to c_i^s$.

$$F(c_{i-1}^t \to c_i^s) = F_s(c_{i-1}^t \to c_i^s) * F_t(c_{i-1}^t \to c_i^s) * F_b(c_i^s) \quad 2 \le i \le n \tag{11}$$

In Fig. 10, we choose one candidate path between each two neighbouring candidate point sets to obtain a matched path $P : c_1^{j_1} \to c_2^{j_2} \to \cdots \to c_n^{j_n}$ for the GPS trajectory T. The overall score of a matched path $F(P)$ can be presented as $\Sigma_{i=2}^{n} F(c_{i-1}^t \to c_i^s)$. The matched path with the highest overall score is regarded as the best matched answer.

5. Experiments

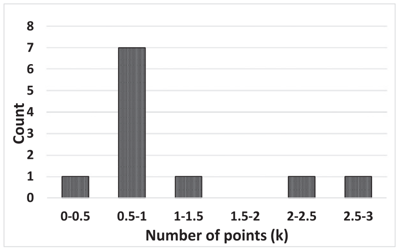
In this section, we first introduce the data set we use and the procedures of data collection. Subsequently, we introduce the parameters used in the experiment. We then show the experimental results. Our approach is compared with three existing algorithms: the ST-matching algorithm [12], the stMM algorithm [19], and the HMM-RCM algorithm [10].

5.1. Dataset description

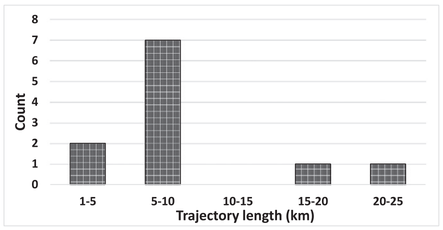
In our experiment, we utilize the road network of Taipei city obtained from Open Street Map. This road network contains 200,236 vertices and 90,709 road segments. We collected real-world trajectories while riding on the streets, using a data collector running on an Android system. In addition to longitude, latitude, and timestamp, we also collected the real-time moving directional data for each GPS point by the magnetic field sensor and accelerometer in the cell phone. We collected 11 trajectories and manually labelled the ground truth for each trajectory. Fig. 11(a) shows the number of GPS points of the trajectories in the data set, and Fig. 11(b) shows the length of the trajectories in the dataset.

5.2. Parameter settings

In this paper, the sampling interval of the GPS trajectory is in the range from one second to two minutes. In the pre-processing phase, we remove the GPS outliers based on the travelling speed constraint of $v_c = 50$ m/s (meter per second). We set $d_c = 4$ m (meter) as the cluster distance threshold, and set the slope difference constraint $s_d = 0.01$ to smooth the trajectory. In our approach, we search the candidate set for a GPS point within radius $r = 200$ m, but only choose k candidate

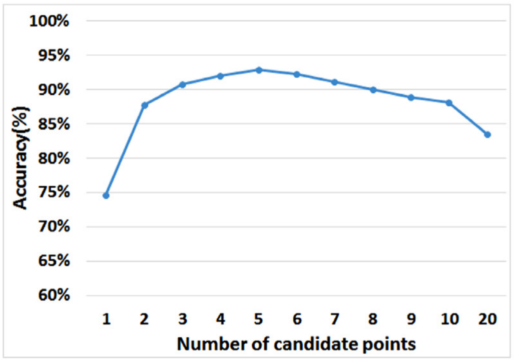


(a) Number of GPS points on each trajectory

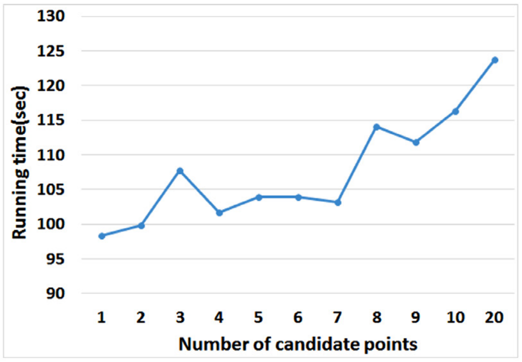


(b) The length of trajectories

Fig. 11. Statistics of the trajectory dataset.

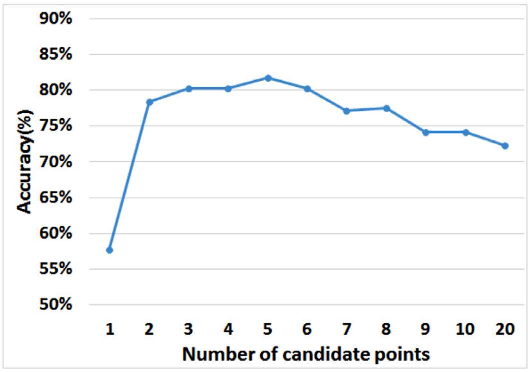


(a) Accuracy

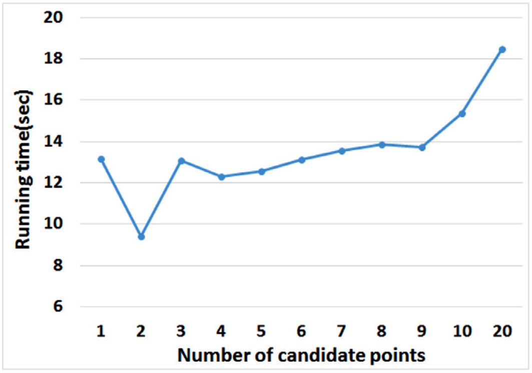


(b) Running time

Fig. 12. Impact of number of k candidate points w.r.t. a high sampling rate.



(a) Accuracy



(b) Running time

Fig. 13. Impact of number of k candidate points w.r.t. a low-sampling rate.

points for a GPS point. In the next section, we discuss the impact of different k and r values in detail. For observation probability, we use a normal distribution with $\mu = 0$ and $\sigma = 20$ m. For the directional analysis function, we use a normal distribution with $\mu = 0$ and $\sigma = 30^\circ$.

Table 1 shows all the parameters used in our approach. We compare our approach with three other algorithms. For the ST-matching algorithm [12], the candidate search range $r = 200$ m, and the parameters of observation probability are the same as in our approach. For the stMM algorithm [19], we set the weight coefficient values $W_d = 32$, $W_b = 21$, $W_s = 35$, $W_h = 12$ as suggested by their algorithm.

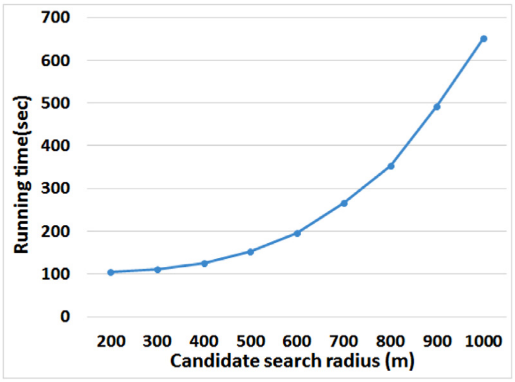


Fig. 14. Running time w.r.t various candidate search ranges.

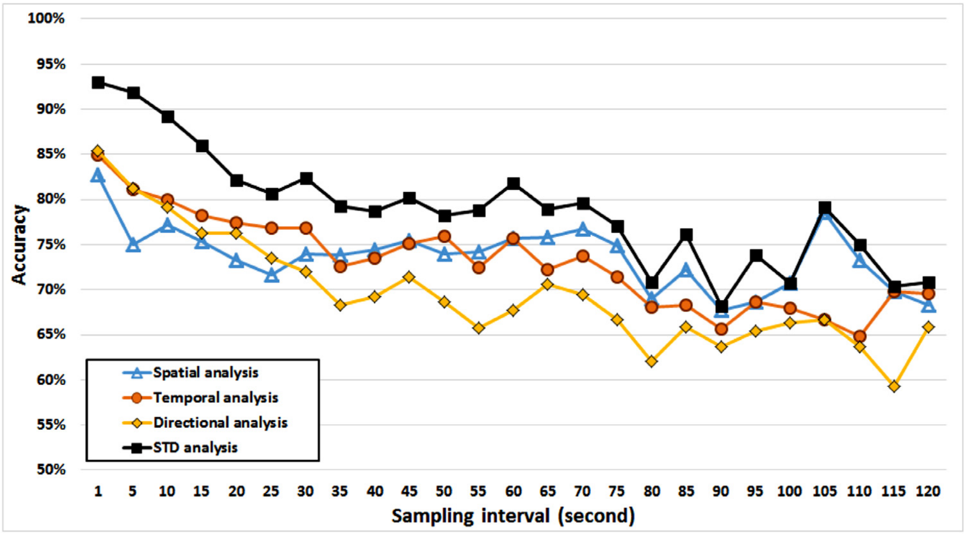


Fig. 15. Accuracy of using one analysis function.

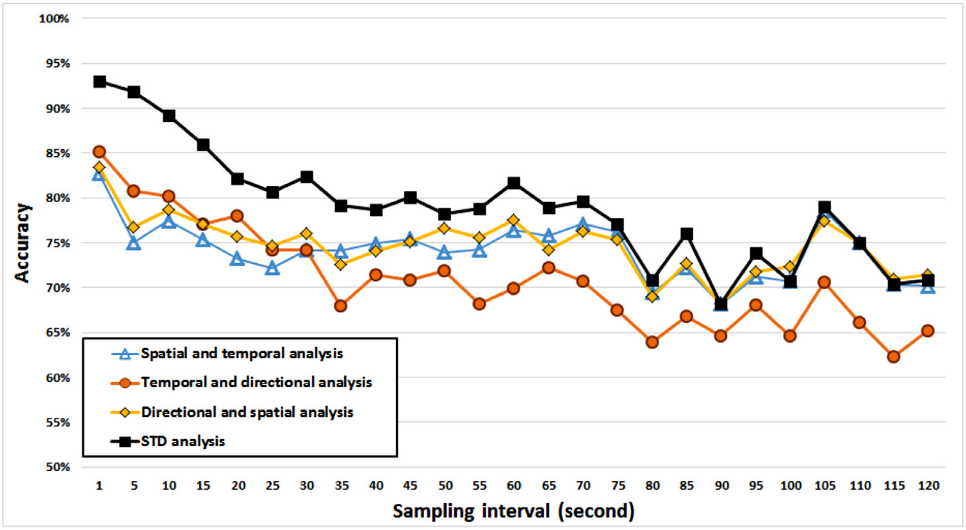


Fig. 16. Accuracy of using two analysis functions.

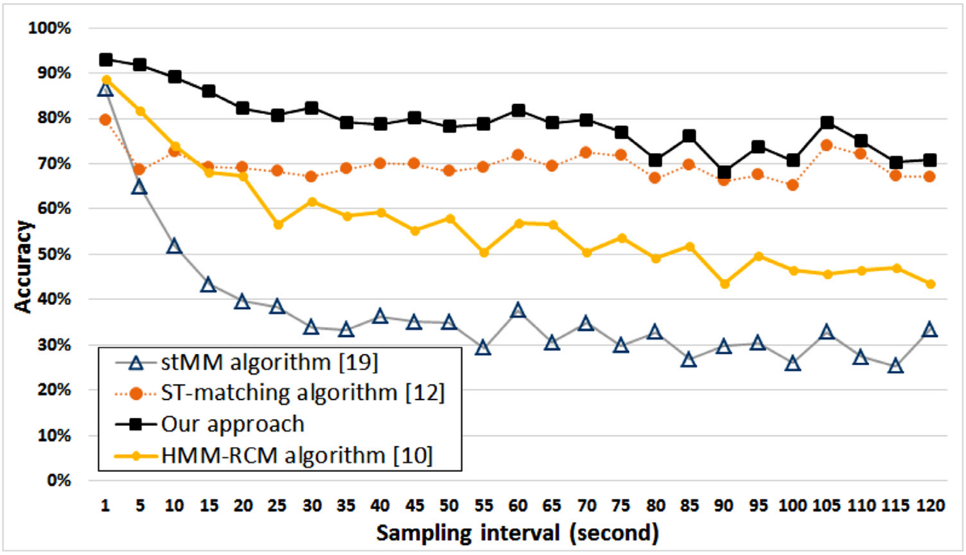


Fig. 17. Accuracy w.r.t sampling intervals.

Table 1
Simulation parameters.

Parameter	Symbol	Value
Travelling speed constraint	v_c	50 m/s
Cluster distance threshold	d_c	4 m
Slope constraint	s_d	0.01
Standard deviation for observation probability and directional analysis function	σ	20 m, 30°

5.3. Experimental results

5.3.1. Impact of number of k candidate points

In our experiment, we choose the top-k candidate points which are first k candidate points close to the GPS point. Figs. 12 and 13 show the impact of different k with respect to the high and low-sampling rates, respectively. The range of k is set from 1 to 10. In Fig. 12(a), the matching accuracy increases when $k = 1$ to $k = 5$. This means that the accuracy increases when the algorithm utilizes more candidate points, and our algorithm does not only consider the distance information to match a GPS point to the closest candidate point. The accuracy decreases when $k = 5$ to $k = 10$, because more irrelevant candidate points might be included. Furthermore, the parallel candidate road segments which are far from the GPS point may have a higher directional analysis score so as to incur inaccurate matching results. As a result, we set k to 5 for the rest of our experiments. Fig. 12(b) shows that the running time increases as the number of candidate points increases, since a large number of candidate points for one GPS point may lead to a large amount of shortest path computation. On the other hand, in Fig. 13, we show the results using the low-sampling-rate (60 s sampling interval) trajectory data by varying the k value. From the figure, we can see a similar trend to the experiments in Fig. 12(a) and (b).

5.3.2. Impact of the candidate search range

We report the impact of the candidate search range in this section. As shown in Fig. 14, running time is proportional to the range of the candidate search radius r. In this paper, we set r to 200 m to search for the top-k candidate points. In the case that k cannot be completely retried within this range, the system then increases the range until all k candidate points are found.

5.3.3. Impact of spatial, temporal, and directional analysis functions

We compare the map matching accuracy of our proposed analysis functions by varying the sampling interval of the data collection in this section. Fig. 15 shows the accuracy of map matching with only one analysis function. Next, we also compare our approach with different combinations of analysis functions in terms of accuracy. Fig. 16 shows the accuracy of map matching with two analysis functions. As a result, our approach that combines all three analysis functions outperforms other combinations of analysis functions.

5.3.4. Our approach vs. the existing algorithms

We compare our approach with the ST-Matching algorithm [12], the stMM algorithm [19], and the HMM-RCM algorithm [10]. Fig. 17 shows the accuracy with respect to the different sampling intervals. Please note that a short sampling interval refers to a high sampling rate, while a long sampling interval refers to a low-sampling rate. We can see that our approach outperforms the three existing algorithms. When matching trajectory data with a sampling interval of up to 10 seconds, the HMM-RCM algorithm performs better than the stMM and ST-matching algorithm. As the sampling interval increases, the accuracy of the stMM algorithm drops dramatically, while our approach and the ST-matching algorithm have more stable performance. We also notice that the average matching accuracy of our approach is still higher than that of the ST-matching algorithm when the sampling interval is over 80 seconds. This implies that the directional analysis improves the matching accuracy when dealing with low-sampling-rate trajectory data. The accuracy of the HMM-RCM approach is worse than that of our approach, because the attributes that influence drivers' route preferences described in the paper are either not significant or are not widely available for use. For example, our trajectories were collected from local streets, so the attributes of *average road class* and *number of class changes* might be unchanging from road segment to segment. Furthermore, the attribute of *number of traffic signals* is not always of concern to a driver. For example, a taxi or a bus driver would drive on the busy streets to pick up passengers.

6. Conclusions

In this paper, we deal with the problem of map matching for low-sampling-rate GPS trajectories. A map-matching algorithm called the STD-matching algorithm is proposed and analyzed in this paper. The algorithm employs the real-time moving direction with the directional analysis function to reflect the influence of a user's true movement over the GPS trajectories. We construct the STD-matching algorithm which considers all of the spatial, temporal, and directional information. Furthermore, we reduce the computational cost by conducting GPS clustering, GPS smoothing, and the A^* shortest path search algorithms. We finally evaluate our approach with three existing algorithms. The experiment results show that our STD-matching algorithm outperforms the ST-matching algorithm, the stMM algorithm, and the HMM-RCM algorithm in terms of matching accuracy for both high-sampling-rate and low-sampling-rate trajectories and concludes the utility of our novel approach.

Acknowledgement

This research has been funded in part by the National Science Council under the Grants NSC 104-2221-E-194-043-MY2 and 105-3011-E-194-001.

References

[1] D. Bernstein, A. Kornhauser, An introduction to map matching for personal navigation assistants (1998).
[2] C. Goh, J. Dauwels, N. Mitrovic, M. Asif, A. Oran, P. Jaillet, Online map-matching based on hidden markov model for real-time traffic sensing applications, in: Intelligent Transportation Systems (ITSC), 2012 15th International IEEE Conference on, IEEE, 2012, pp. 776–781.
[3] A.V. Goldberg, C. Harrelson, Computing the shortest path: A search meets graph theory, in: Proceedings of the Sixteenth Annual ACM-SIAM Symposium on Discrete Algorithms, Society for Industrial and Applied Mathematics, 2005, pp. 156–165.
[4] Y.J. Gong, E. Chen, X. Zhang, L.M. Ni, J. Zhang, Antmapper: an ant colony-based map matching approach for trajectory-based applications, IEEE Trans. Intell. Transp. Syst. PP (99) (2017) 1–12, doi:10.1109/TITS.2017.2697439.
[5] J.S. Greenfeld, Matching gps observations to locations on a digital map, Transportation Research Board 81st Annual Meeting, 2002.
[6] J. Han, X. Fu, L. Liu, D. Jiang, Online map matching by indexing approximate road segments, in: Software Engineering and Service Science (ICSESS), 2011 IEEE 2nd International Conference on, IEEE, 2011, pp. 299–303.
[7] Z.-c. He, S. Xi-wei, L.-j. Zhuang, P.-l. Nie, On-line map-matching framework for floating car data with low sampling rate in urban road networks, Intell. Transp. Syst., IET 7 (4) (2013) 404–414.
[8] S.K. Honey, W.B. Zavoli, K.A. Milnes, A.C. Phillips, M.S. White Jr, G.E. Loughmiller Jr, Vehicle Navigational System and Method, 1989 US Patent 4,796,191.
[9] G. Hu, J. Shao, F. Liu, Y. Wang, H.T. Shen, If-matching: towards accurate map-matching with information fusion, IEEE Trans. Knowl. Data Eng. 29 (1) (2017) 114–127, doi:10.1109/TKDE.2016.2617326.
[10] G.R. Jagadeesh, T. Srikanthan, Online map-matching of noisy and sparse location data with hidden Markov and route choice models, IEEE Trans. Intell. Transp. Syst. PP (99) (2017) 1–12, doi:10.1109/TITS.2017.2647967.
[11] X. Liu, K. Liu, M. Li, F. Lu, A st-crf map-matching method for low-frequency floating car data, IEEE Trans. Intell. Transp. Syst. 18 (5) (2017) 1241–1254, doi:10.1109/TITS.2016.2604484.
[12] Y. Lou, C. Zhang, Y. Zheng, X. Xie, W. Wang, Y. Huang, Map-matching for low-sampling-rate gps trajectories, in: Proceedings of the 17th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems, ACM, 2009, pp. 352–361.
[13] R. Mohamed, H. Aly, M. Youssef, Accurate and efficient map matching for challenging environments, in: Proceedings of the 22nd ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems, ACM, 2014, pp. 401–404.
[14] R. Mohamed, H. Aly, M. Youssef, Accurate real-time map matching for challenging environments, IEEE Trans. Intell. Transp. Syst. 18 (4) (2017) 847–857, doi:10.1109/TITS.2016.2591958.
[15] P. Newson, J. Krumm, Hidden markov map matching through noise and sparseness, in: Proceedings of the 17th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems, ACM, 2009, pp. 336–343.
[16] M. Nikoli, J. Jovi, Implementation of generic algorithm in map-matching model, Expert Syst. Appl. 72 (C) (2017) 283–292, doi:10.1016/j.eswa.2016.10.061.
[17] A. Oran, P. Jaillet, An hmm-based map matching method with cumulative proximity-weight formulation, in: Connected Vehicles and Expo (ICCVE), 2013 International Conference on, IEEE, 2013, pp. 480–485.
[18] B.P. Phuyal, Method and use of aggregated dead reckoning sensor and gps data for map matching, in: Proceedings of the 15th International Technical Meeting of the Satellite Division of The Institute of Navigation (ION GPS 2002), 2002, pp. 430–437.
[19] M. Quddus, S. Washington, Shortest path and vehicle trajectory aided map-matching for low frequency gps data, Transp. Res. Part C 55 (2015) 328–339.

[20] M.A. Quddus, W.Y. Ochieng, L. Zhao, R.B. Noland, A general map matching algorithm for transport telematics applications, GPS Sol. 7 (3) (2003) 157–167.
[21] M. Rahmani, H.N. Koutsopoulos, Path inference from sparse floating car data for urban networks, Transp. Res. Part C 30 (2013) 41–54.
[22] G. Wang, R. Zimmermann, Eddy: an error-bounded delay-bounded real-time map matching algorithm using hmm and online viterbi decoder, in: Proceedings of the 22nd ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems, ACM, 2014, pp. 33–42.
[23] C.E. White, D. Bernstein, A.L. Kornhauser, Some map matching algorithms for personal navigation assistants, Transp. Res. Part C 8 (1) (2000) 91–108.
[24] Y. Zhao, Vehicle location and navigation systems, 1997.