Feature-based Map Matching for Low-Sampling-Rate GPS Trajectories

YIFANG YIN, National University of Singapore
RAJIV RATN SHAH, Indraprastha Institute of Information Technology Delhi
GUANFENG WANG, Grab Research and Development
ROGER ZIMMERMANN, National University of Singapore

With the increasing availability of GPS-equipped mobile devices, location-based services have become an integral part of everyday life. Among one of the initial steps of positioning data management, map matching aims to reduce the uncertainty in a trajectory by matching the GPS points to the road network on a digital map. Most existing work has focused on estimating the likelihood of a candidate route based on the GPS observations, while neglecting to model the probability of a route choice from the perspective of drivers. In this work, we propose a novel feature-based map matching algorithm that estimates the cost of a candidate route based on both GPS observations and human factors. To take human factors into consideration is highly important, especially when dealing with low sampling rate data where most of the movement details are lost. Additionally, we simultaneously analyze a subsequence of coherent GPS points by utilizing a new segment-based probabilistic map matching strategy, which is less susceptible to the noisiness of the positioning data. We have evaluated both the offline and the online versions of our proposed approach on a public large-scale GPS dataset, which consists of 100 trajectories distributed all over the world. The experimental results show that our method is robust to sparse data with large sampling intervals (e.g., 60s ~ 300s) and challenging track features (e.g., u-turns and loops). Measurements including map matching accuracy and system efficiency have been thoroughly evaluated and discussed. Compared with two state-of-the-art map matching algorithms, our method substantially reduces the route mismatch error by 6.4% ~ 32.3% (either offline or online with the window size set to 360s), with a slight increase in terms of the processing time. The experimental results show that our proposed method obtains the state-of-the-art map matching results in all the different combinations of sampling rates and challenging features.

CCS Concepts: • **Information systems** → **Global positioning systems**; *Location based services*; • **Mathematics of computing** → *Probabilistic algorithms*;

Additional Key Words and Phrases: GIS, worldwide GPS data, feature-based map matching, trajectory simplification

This research was supported in part by the National Natural Science Foundation of China under Grant No. 61472266 and by the National University of Singapore (Suzhou) Research Institute, 377 Lin Quan Street, Suzhou Industrial Park, Jiang Su, People's Republic of China, 215123.
Authors' addresses: Y. Yin, National University of Singapore, 13 Computing Drive, 117417, Singapore; email: yifang@comp.nus.edu.sg; R. Ratn Shah, Indraprastha Institute of Information Technology Delhi, India; email: rajivratn@iiitd.ac.in; G. Wang, Grab Research and Development, 573972, Singapore; email: guanfeng.wang@grab.com; R. Zimmermann, National University of Singapore, 13 Computing Drive, 117417, Singapore; email: rogerz@comp.nus.edu.sg.

https://doi.org/10.1145/3223049

ACM Reference format:
Yifang Yin, Rajiv Ratn Shah, Guanfeng Wang, and Roger Zimmermann. 2018. Feature-based Map Matching for Low-Sampling-Rate GPS Trajectories. *ACM Trans. Spatial Algorithms Syst.* 4, 2, Article 4 (August 2018), 24 pages.
https://doi.org/10.1145/3223049

1 INTRODUCTION

Accurate map matching has been a fundamental but challenging problem that has drawn great research attention in recent years. Given a vehicle track consisting of a sequence of GPS points, map matching algorithms aim to automatically determine the correct route where the driver has traveled on a digital map. The correction of the raw positioning data has been important for many downstream applications, such as navigation, tracking systems, and semantic trajectory detection [8, 11, 12, 25, 35, 37]. Recently, an increasing number of statistics-based map matching algorithms have been proposed to deal with the challenging GPS trajectories that pose difficulties in traditional geometry-based or topology-based methods, e.g., data noise and sparsity. Among the advanced statistics-based algorithms, the Hidden Markov Model (HMM) is one of the most popular and widely used techniques that models the road emission and transition probabilities based on the measurement noise and the road network layout [21]. It has been reported that the HMM-based map matching is highly effective when dealing with trajectories where the GPS sampling interval is less than 30s. However, real positioning data are sometimes collected at a very low sampling rate, e.g., a sample point every 5min [38], due to the high energy consumption of GPS, WiFi, and inertial sensors on mobile devices [18]. A growing number of applications, including energy-efficient localization and cellular provider side localization, depend on only sparse and coarse-grained positioning data [3], posing great difficulties in the development of map matching algorithms.

To reduce the uncertainty in low sampling rate trajectories, hybrid methods have been proposed to estimate the transition probability between two road segments based on a fusion of multiple metrics [3, 4, 9]. For example, Aly and Youssef proposed to detect road semantics with multiple sensors and estimate the transition probability based on both the orientation difference and the skipped road semantics [3]. However, such algorithms mostly assume that the driver has traveled on the shortest path between two road segments, which is not always true especially when dealing with low sampling rate data. To solve the above problem, Zheng et al. proposed to infer the possible routes based on the travel patterns derived from historical data [40]. Osogami and Raymond considered both the number of turns and the travel distance in the cost modeling of a candidate path [22]. However, the performance of such algorithms can be limited due to the requirement of sufficient historical GPS trajectories in the learning phase. Moreover, the original HMM-based map matching algorithm and its extensions mostly retrieve candidate road segments for every GPS point [3, 21, 22], resulting in decreasing effectiveness for trajectories with the existence of large sensor noise.

We therefore present a novel feature-based framework for accurate map matching of challenging GPS trajectories. We first detect key GPS points to segment a trajectory into a list of subsequences. To reduce the method's sensitivity to data noise, we simultaneously consider all the GPS points in one segment to determine the most likely route taken by a user. Figure 1 illustrates a trajectory segment consisting of three GPS points, and two candidate matching routes. The likelihood of a candidate match is computed based on the corresponding cost modeled with a list of pre-defined features. More specifically, we model the cost of a route choice based on two types of features: (1) *trajectory-related features* to estimate the cost of a route from the perspective of GPS observations, and (2) *road-related features* to model the behavior cost of a route choice from the

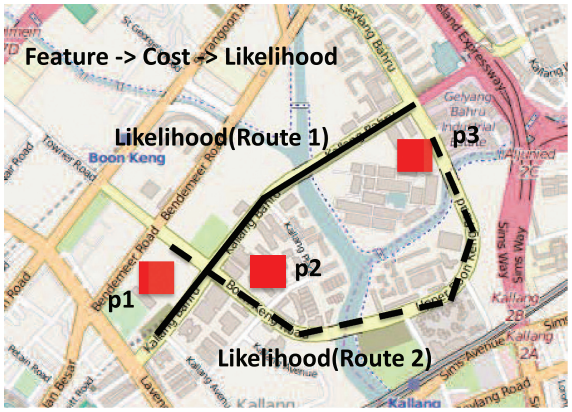


Fig. 1. Route likelihood estimation based on cost modeling with pre-defined features.

perspective of users. We next compute the likelihood of a candidate path based on its cost and determine the correct match as the path with the highest likelihood. From a global perspective, we retrieve candidate road segments for each key GPS point, search for a local optimal path between any two neighboring candidate road segments, and compute the likelihood of a global candidate path by multiplying the likelihood values of the local optimal paths it contains. This problem can be solved efficiently by dynamic programming techniques. Our proposed method is initially offline but can be easily extended into an online version by dividing the trajectory into fixed-sized sequences and process each segment individually. We have evaluated our proposed approach in terms of both accuracy and efficiency by comparing it with two state-of-the-art map matching algorithms. Experiments are conducted on real-world GPS data sampled with different intervals ranging from 1 to 5min. We report the route mismatch fraction, map matching precision, recall, F_1 score, offline processing time, and online output latency as the evaluation measures and the experimental results show that our method is competitively efficient and outperforms its competitors by a mismatch error reduction rate of 6.4% ~ 32.3% on average. With the availability of richer data sources, the map matching results can be further improved. We additionally discuss the utilization and integration of three important types of supplementary data (i.e., GPS accuracy measures, smartphone internal sensors, and historical GPS trajectories) in our proposed system. Here, we summarize the contributions of this article in the following four aspects:

- We present a novel feature-based map matching technique that models the cost of a candidate path with both trajectory-related features (e.g., the distance to the closest GPS point) and road characteristics (e.g., length and transitions).
- We consider more than one GPS point at a time by utilizing a new segment-based probabilistic map matching strategy that searches for shortest path between candidate roads of only the key points detected by trajectory simplification techniques.
- We perform extensive experiments in terms of accuracy and efficiency on a large-scale real dataset consisting of trajectories with features (e.g., u-turns and loops) that pose difficulties to map matching algorithms.
- We evaluate the proposed technique with varying sampling intervals (1~5min). The experimental results show that our method works consistently well and outperforms the state-of-the-art map matching algorithms.

The rest of the article is organized as follows. We first report the important related work in Section 2 and present the system overview in Section 3. Next, we introduce the technical details

of the proposed feature-based map matching framework and discuss the integration of potential supplementary data sources in Section 4. Finally, we evaluate the effectiveness of our proposed approach by comparing with the state-of-the-art map matching techniques in Section 5. Section 6 concludes and suggests future work.

2 RELATED WORK

Over the past decades, extensive research has been conducted on matching GPS points on a digital map. With simple road network information such as the locations and the shape of the roads, early map matching techniques can be generally classified into two categories: geometry-based matching and topology-based analysis. Geometry-based algorithms match a single GPS point [34] or a segment of GPS trajectories [6, 41] to the closest road arc based on geometric calculations. However, without considering the constraints induced by a map topology, these methods suffer from one significant drawback of being sensitive to measurement errors. However, topology-based map matching algorithms utilize not only the shapes but also topological information such as connectivity and contiguity of the road network [4, 10]. Quddus et al. reported improved results by leveraging vehicle information of heading and speed in the topological analysis [24]. However, such algorithms are still vulnerable to sensor noise and unsuitable for highly erroneous and sparse positioning data [3].

To pursue improved matching accuracy, probabilistic map matching algorithms have been proposed to take advantage of statistical models such as Kalman Filter [23], particle filters [11, 16], and HMM [7, 9, 21]. Wang et al. proposed a novel statistics-based online map matching algorithm called Eddy with a solid error- and delay-bound analysis [33]. To work with mobile devices, Liu et al. presented a novel technique termed Passby that maintains high matching accuracies while working with the most simplified road network [17]. Situations that pose difficulties in map matching, e.g., dealing with low sampling rate GPS tracks [20, 39] and matching to incomplete map data [31], have also been studied recently but still remain challenging problems. Newson and Krumm proposed an elegant HMM-based map matching algorithm for relatively noisy and sparse GPS trajectories [21]. However, experiments show that the accuracy decreases significantly when the sampling period grows larger than 30s. Zheng et al. proposed a history-based route inference system that derives the travel pattern from historical data to reduce the uncertainty of GPS trajectories [40]. However, the inference process requires a large quantity of historical trajectories with good coverage and high density, which greatly limits the applicability of such algorithms. However, trajectory segmentation algorithms have been proposed for spatio-temporal data management [5, 26, 27]. Buchin et al. presented an efficient trajectory segmentation algorithm based on different spatiotemporal criteria including location, heading, speed, velocity, curvature, shape, and so on [5]. Song et al. proposed a novel road-network-based trajectory compression algorithm that outperformed existing approaches in terms of saving storage cost with bounded errors [27]. However, only a few efforts have been made on incorporating trajectory simplification algorithms to improve map matching results [15].

Recently, a number of map matching algorithms started to utilize other sensors equipped on smartphones such as WiFi and cellular fingerprinting [29, 30]. Aly and Youssef proposed to detect road semantics (e.g., speed bumps and tunnels) by leveraging smartphone's inertial sensors and presented an improved HMM with a semantics-enriched digital map [3]. Furthermore, one interesting direction that emerged recently is to perform map matching with the assistance of driver behavior analysis. Drivers attempt to reach some destination while optimizing some trade-off between time, safety, and other factors that can be modeled by a list of path features such as road type and speed limit [42]. Osogami and Raymond proposed to integrate the number of turns in the transition probability calculation of a HMM to favor a more "natural" path in the decision-making

process [22]. Promising results have been reported on GPS points taken during a single trip in Seattle. However, the generality of such methods remains unclear without conducting extensive experimental studies on large-scale positioning data.

3 SYSTEM OVERVIEW

As an overview, we first provide formal definitions of the map matching problem in our proposed feature-based framework. Next, we briefly discuss the functionality of each system component and introduce the technical details in the next section.

3.1 Problem Statement

Given a GPS trajectory and a digital map, our goal is to find the most likely route that has been traveled by the user. Here, we model a map as a simple directed graph where the nodes have assigned geo-coordinates on Earth and the edges represent linear road segments between two nodes:

Definition 1 (Road Segment). A road segment e is a directed edge that is associated with an id $e.eid$, a length value $e.l$, a starting point $e.startp$, and an ending point $e.endp$. It represents a linear road segment between the two nodes $e.startp$ and $e.endp$.

Definition 2 (Road Network). A road network is a directed graph $G(V, E)$, where E is a set of edges representing the road segments and V is the vertex set consisting of the starting and ending points of the road segments.

Definition 3 (GPS Trajectory). A GPS trajectory is a sequence of GPS points $T = \{p_1, p_2, \ldots, p_n\}$. Each point p_i is associated with a geo-coordinate $< p_i.lat, p_i.lon >$ and a timestamp $p_i.t$.

To incorporate user behavior analysis, we also introduce the concept of *action*, which describes the user behavior of traveling from one road segment to another. Note that the two road segments are required to be directly connected to form a legal action. The formal definition is given as below:

Definition 4 (Action). An action a is a directed edge that is associated with a starting road segment $a.start$, an ending road segment $a.end$, the angle between the two road segments $a.angle$, and the cost of taking this action $a.cost$. It models a turning action from $a.start$ to $a.end$.

Definition 5 (Action Graph). An action graph is a directed graph $G_A = (E, A)$, where the vertices E are the set of the road segments in $G(V, E)$, and the edges $A = \{a_1, a_2, \ldots, a_m\}$ are formed by the actions of traveling between two directly connected road segments.

Definition 6 (Action Sequence). An action sequence AS is a path connecting two road segments in the action graph $G_A(E, A)$. Let $AS = \{\tilde{a}_1, \tilde{a}_2, \ldots \tilde{a}_{\tilde{m}}\}$ where $\tilde{a}_i \in A$, then for each i in range $[1, \tilde{m})$, we have $\tilde{a}_i.end = \tilde{a}_{i+1}.start$.

Definition 7 (Path). A path P is a sequence of connected road segments. Given an action sequence AS, the corresponding path can be recovered by concatenating the road vertices traversed by AS.

Now, we define the map matching problem as follows: Provided with a raw GPS trajectory T and a road network $G(V, E)$, generate the action graph $G_A(E, A)$ and estimate the cost and probability of taking each action in A. Find the most probable sequence of actions AS and recover the optimal path P accordingly.

3.2 Architecture Overview

The architecture of the proposed feature-based map matching system is illustrated in Figure 2. It consists of three major components: *Road Candidate Preparation*, *Action Graph Generation*, and *Path Recovery*.

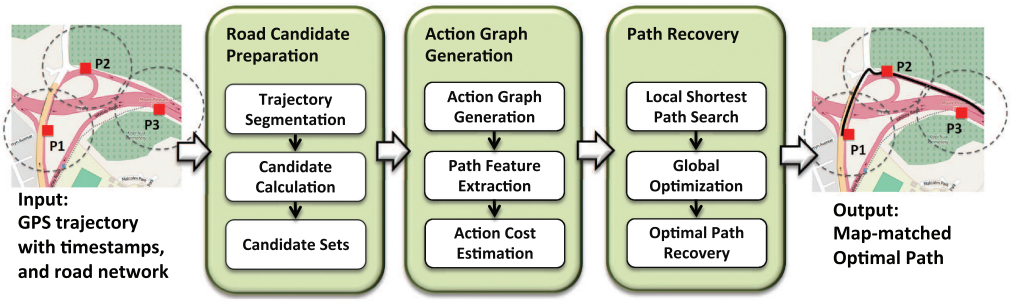


Fig. 2. Overview of the proposed feature-based map matching system architecture.

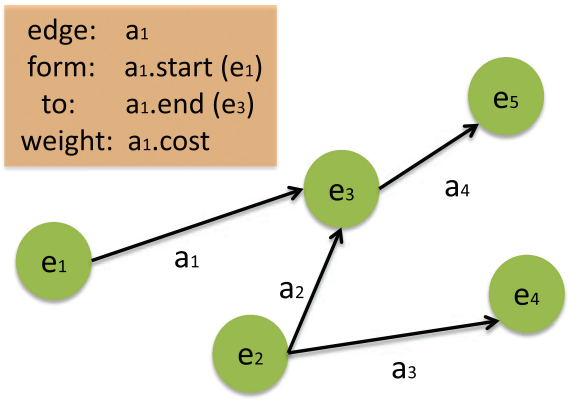


Fig. 3. The generated action graph where nodes are road segments and edges are actions.

***Road Candidate Preparation*:** Given a raw GPS trajectory and the corresponding road network information, this component retrieves possible candidate roads for a number of key GPS points detected from the trajectory. The key points are obtained by trajectory simplification while preserving the shape of the curve within a given tolerance [13, 19]. Trajectory simplification helps remove noisy and stop points, so that enhanced map matching results can be obtained [15]. While previous statistics-based methods mostly retrieve candidate roads for every GPS point, we alternatively adopt a more effective segment-based map matching strategy that considers a subsequence of points at a time in the probabilistic modeling. Moreover, we incorporate user behavior analysis in finding the local optimal path between the candidate roads of two neighboring key GPS points, which significantly improves the map matching accuracy especially when dealing with low sampling rate positioning data.

***Action Graph Generation*:** This component generates the action graph $G_A = (E, A)$ where the action sequence that matches the real route can be detected based on a simple shortest path search between any two neighboring candidate roads. As illustrated in Figure 3, the nodes in the action graph are formed by the road segments and the edges are formed by the actions A where each element a represents a directed edge from node $a.start$ to node $a.end$ with the edge weight $a.cost$. Please note that the cost here models not only the likelihood of taking one action given the raw GPS observations but also the trade-off made by users (drivers) among a list of factors such as time, safety, and stress. More specifically, we model the cost of a user's route choice behavior based on a list of raod features. Two basic road features (length and transitions) that are available

from all digital maps are leveraged in this work. If provided with more semantic information such as road type and speed limit, then user behavior cost can be better modeled by advanced feature fusion techniques [42]. Additionally, in areas where dense historical GPS trajectories are available, it is also possible to automatically infer road semantics, e.g., road popularity, based on data-driven approaches [40].

***Path Recovery*:** Based on the action graph generated, this component computes the shortest path and the corresponding cost between any two neighboring candidate road segments in the retrieved candidate set. The local shortest paths are next concatenated at the starting and ending road segments to form a set of global candidate paths for the whole input trajectory. Thereafter, the probability of a global candidate path being the correct match is modeled based on the cost of the local shortest paths it contains. Finally, the candidate path with the highest probability score is returned as the predicted map matching result, which can be efficiently solved by a dynamic programming technique.

4 FEATURE-BASED MAP MATCHING

In traditional map matching algorithms, the candidate path scoring mostly relies on the distance modeling between the input GPS observations and the road network database based on spatial and temporal constraints [20, 21]. In this work, we refer to the aforementioned aspect as trajectory-related path features and further propose a general framework that enables effective feature fusion with road characteristics in the decision-making process. Our proposed map matching algorithm can effectively reduce the uncertainty of low sampling rate GPS data with possibly challenging patterns such as u-turns and loops. The technical details of each system component are introduced as below.

4.1 Action Graph Generation

The *Action Graph Generation* is the core component of the proposed system. Recall that the action graph is created based on the road network. The nodes in the graph are road segments and an edge a describes the action of traveling from one road segment $a.start$ to a neighboring road segment $a.end$. The edge weight is set to $a.cost$. Next, we introduce how to extract path features and estimate action costs based on an input trajectory segment s.

4.1.1 Path Feature Extraction. We extract two types of path features to estimate the traveling cost, taking both the GPS observations and the human factors into consideration.

- *Trajectory-related features*: the distance between a road and a trajectory segment, which will next be used to estimate the cost of a road segment given the GPS observations.
- *Road-related features*: the road characteristics such as length and transitions, which will next be used to model the behavior cost of a route choice made by users.

Figure 4 illustrates the *trajectory-related feature* extraction with the input being a trajectory segment consisting of four GPS points. Intuitively, the road segments that are farther from the trajectory are less likely to be the correct match. Therefore, we formulate the distance between a road segment e and a trajectory segment s as

$$dist(e, s) = \min_{p \in s} dist(e, p), \tag{1}$$

where $p \in s$ denotes a GPS point p in trajectory segment s, and $dist(e, p)$ represents the distance between point p and road segment e, which is defined to be the great circle distance between point p and the point on road segment e that is the closest to p. For example, in Figure 4, $dist(e, s) = dist(e, p_4)$ as point p_4 is the closest GPS measurement to road segment e.

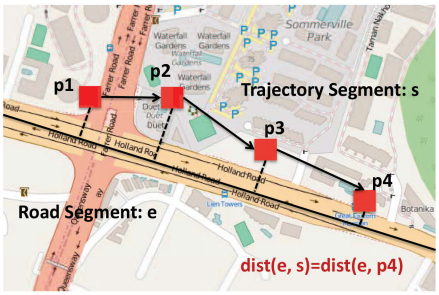


Fig. 4. Illustration of the *trajectory-related feature* extraction.

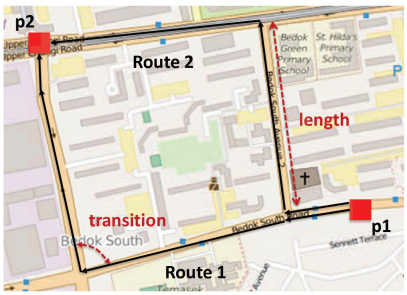


Fig. 5. Illustration of the *road-related feature* extraction.

Figure 5 illustrates the extraction of the two *road-related features* that are leveraged in this work to model a user's behavior cost. Due to concerns about the time, people usually favor shorter paths over longer ones. Moreover, the type and the number of road transitions also play an important role in users' route choices [22]. For example, in Figure 5, even though *Route 2* is the shorter path between the two GPS points, it is more likely that people would choose *Route 1* due to safety concerns as this path contains only one 90-degree-turn while the former path contains two. Based on the above observations, we compute the length of a road segment and the transition angle between any two connected road segments. Each length feature is associated with a road segment e and denoted as $e.l$. Similarly, each transition angle is associated with an action a and denoted as $a.angle$. Next, we discuss how to formulate the action cost based on the extracted features.

4.1.2 Action Cost Estimation. Considering that the probability of driver to turn on a road segment that is farther away from the GPS observations is small, we model the cost of an action a based on the input trajectory segment s as

$$C_{traj} = \min\{dist(e, s), maxC_{traj}\}, \tag{2}$$

where $e = a.end$ is the ending road segment of action a and $maxC_{traj}$ is a threshold that limits the maximum value of C_{traj}. We set the cost of the road segments that are far away from the input trajectory segment to a constant value $maxC_{traj}$. This is because that the trajectory segment s provides little information when the distance $dist(e, s)$ grows much larger than the GPS accuracy, i.e., GPS observations are only effective for the cost estimation of the nearby road segments. Therefore, we set $maxC_{traj} = 100$ meters in our experiments, which we consider to be reasonable according to the GPS uncertainty.

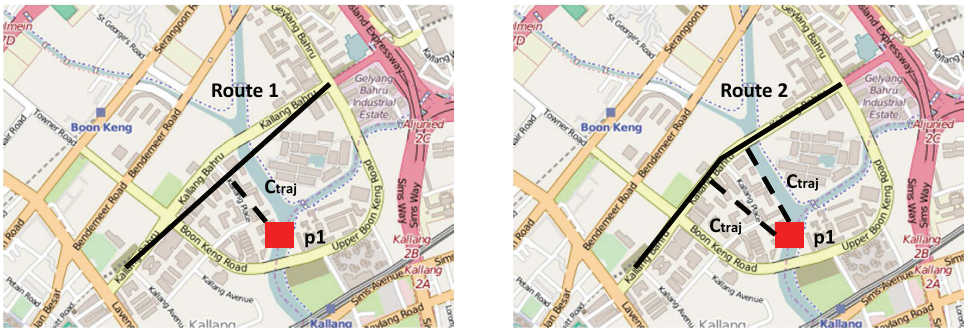
Based on *road-related features*, we estimate C_{len} of action a as the length of the ending road segment $a.end$ in meters,

$$C_{len} = e.l, \tag{3}$$

where $e = a.end$ and $e.l$ is the length attribute of road segment e. Thereafter, we model the cost of a transition C_{turn} as proposed by Osogami and Raymond [22],

$$C_{turn} = \begin{cases} 0 & |a.angle| < \pi/4 \\ 1 & \pi/4 \le |a.angle| < 3\pi/4 \\ 2 & 3\pi/4 \le |a.angle| \le \pi \end{cases}, \tag{4}$$

where $a.angle$ is the transition angle between the two road segments $a.start$ and $a.end$. It is worth mentioning that currently we do not distinguish between left and right turns, therefore we have $0 \le a.angle \le \pi$. Later in the experiments, we will see the setting of the relative weights of angles



(a) A route consists of one road segment. (b) A route consists of two road segments.

Fig. 6. An example to illustrate the proposed cost function.

shown in Equation (4) works generally well on trajectories from different regions all over the world. Moreover, it is also possible to fine-tune the weights using machine learning techniques with the presence of large GPS data.

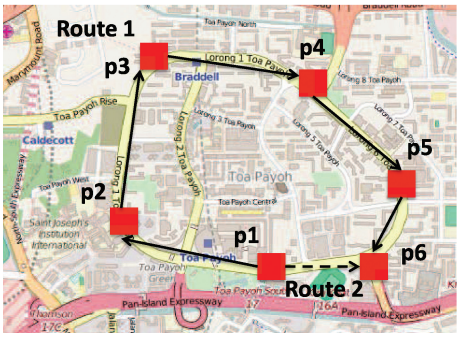
Finally, we fuse C_{traj}, C_{len}, and C_{turn} into the overall cost of an action as

$$C = C_{traj} \cdot (C_{len} + \omega C_{turn}), \tag{5}$$

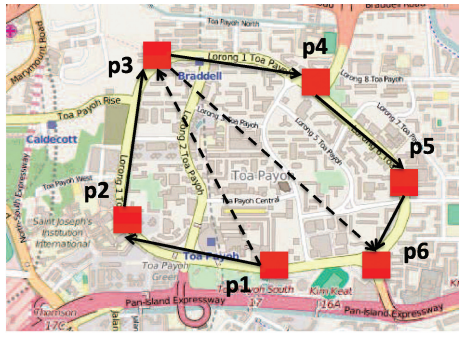
where ω is a balancing factor between road length and transition angle. The intuition of Equation (5) is that small C_{traj} values promote user actions of turning onto road segments that are closer to the GPS observations, while small C_{len} and C_{turn} promote shorter and straighter routes that are more likely to be chosen by users. We show an example in Figure 6 to illustrate the reason that we use the product instead of the sum in the late fusion between the trajectory-related cost and the road-related cost in Equation (5). Figures 6(a) and 6(b) show two routes between the same source and destination but consist of different numbers of road segments. Point p_1 is a GPS observation, and we also plot the corresponding C_{traj} (marked in dotted lines) in the two figures. If we perform the late fusion as the sum of the trajectory-related cost and the road-related cost, then the number of road segments in the route will have a great impact on the sum of C_{traj} even when the shape of the route remains approximately unchanged. Take Figure 6 as an example, the total trajectory-related cost of *Route 2* will be more than twice as that of *Route 1*, which can be an issue as intuitively the total trajectory-related cost of *Route 2* should only be slightly greater. To remove the impact caused by the number of road segments in a route, we consider C_{traj} to be the weight of the corresponding road segment, and adopt the product as our late fusion strategy. Thereafter, we compute cost C for every action a and set $a.cost = C$ accordingly. Next, we present a segment-based map matching strategy that models the probability of a route choice based on the action cost.

4.2 Segment-based Probabilistic Modeling

Based on the action cost estimated using Equation (5), we are able to retrieve the most likely action sequence that connects any two candidate road segments in the action graph by shortest path search. Figure 7(a) shows a trajectory consisting of six GPS points with the feature of *loop*. The shortest-path-search strategy based on our proposed cost estimation works well for relatively straight trajectory segments such as $\{p_1, p_2, p_3\}$ or $\{p_3, p_4, p_5, p_6\}$. However, it is highly difficult to recover the real path between p_1 and p_6 directly as *Route 2* is more likely to be mistakenly retrieved due to the small values of C_{len} and C_{turn}. To solve this problem, we propose a segment-based

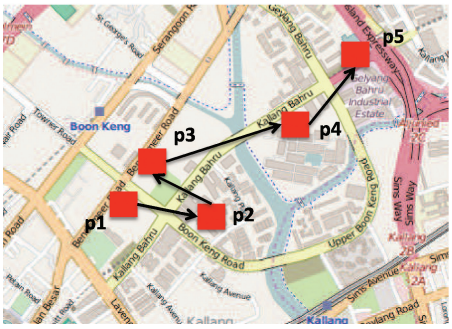


(a) Problems caused by loops in GPS trajectories.

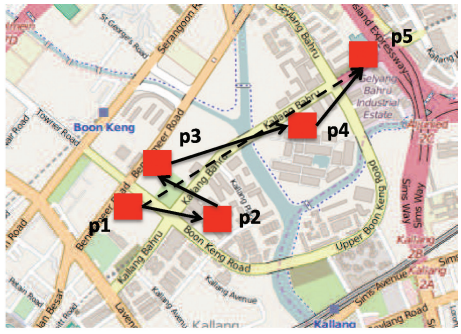


(b) Simplified trajectory after applying the Douglas-Peucker algorithm.

Fig. 7. Segment-based map matching for trajectories with loops.



(a) Problems caused by hives in GPS trajectories.



(b) Simplified trajectory after applying the Douglas-Peucker algorithm.

Fig. 8. Segment-based map matching for trajectories with hives.

probabilistic model by trajectory simplification. As illustrated in Figure 7(b), we first segment the input trajectory and obtain a list of key GPS points (e.g., p_1, p_3, and p_6). Next, we retrieve possible candidate road segments for each of the key GPS points, compute the shortest path between any two neighboring candidates, and recover the real path by global optimization.

Moreover, trajectory simplification can help enhance map matching results for trajectories with challenging features by removing noisy and stop points [15]. If vehicles get stuck in traffic jams or stop at intersections, then GPS points are most likely to be randomly distributed around the stop point due to the general GPS noise. As illustrated in Figure 8(a), traditional map matching algorithms process each GPS point individually where stop points can cause significant errors in the map matching results. Figure 8(b) shows that by applying trajectory simplification, noisy points and oversampling at stop points can be effectively reduced. Though the number of GPS points available in a simplified trajectory decreases, the shape of the curve can still be well preserved while setting the error tolerance to a reasonable value. The technical details of the proposed segment-based probabilistic modeling are introduced as below.

4.2.1 Trajectory Segmentation. In our implementation, we obtain the list of key GPS points by applying the Douglas-Peucker algorithm [13]. Given a trajectory composed of line segments, this

algorithm simplifies the trajectory by finding a similar curve with fewer points. For example, in Figure 7(b), a trajectory consisting of six GPS points is approximated by curve $\{p_1, p_3, p_6\}$ after simplification. Subsequently, the trajectory is divided into two segments $\{p_1, p_2, p_3\}$ and $\{p_3, p_4, p_5, p_6\}$, with the key GPS points being p_1, p_3, and p_6.

The Douglas-Peucker algorithm initially keeps the first and the last points in the trajectory (i.e., p_1 and p_6), and then finds the point that is the farthest from the line segment between the first and the last points (i.e., p_3). This point will only be kept when its distance to the line segment is greater than a pre-defined threshold ϵ. If this point is kept, then the algorithm will recursively process the segment from the first point to this point and the segment from the this point to the last point. Otherwise, any points other than the first and the last points can be discarded with the simplified curve being no worse than ϵ.

4.2.2 Path Recovery. For an input trajectory $T = \{p_1, p_2, \ldots, p_n\}$, we first detect the key GPS points, denoted as $\tilde{T} = \{\tilde{p}_1, \tilde{p}_2, \ldots, \tilde{p}_{\tilde{n}}\}$, based on the Douglas-Peucker algorithm. Next, we retrieve the nearest ten road segments and form the candidate set $E^i = \{e_1^i, e_2^i, \ldots, e_{10}^i\}$ for each key point $\tilde{p}_i$. We model the GPS noise with a Gaussian kernel and estimate the probability of candidate e_k^i being the correct match of key point $\tilde{p}_i$ as [21]

$$p(e_k^i) = \frac{1}{\sqrt{2\pi}\sigma} e^{\frac{dist(e_k^i, \tilde{p}_i)^2}{2\sigma^2}}, \tag{6}$$

where σ is the standard deviation of GPS measurements, $dist(e_k^i, \tilde{p}_i)$ represents the minimum great circle distance between candidate e_k^i and point $\tilde{p}_i$.

The probability of a route being the correct match of the real path between two neighboring key points $\tilde{p}_i$ and $\tilde{p}_{i+1}$ is calculated as follows. For any two candidate road segments e_f^i and e_t^{i+1} ($1 \le f, t \le 10$, $1 \le i < \tilde{n}$), we obtain the optimal action sequence that is connecting e_f^i and e_t^{i+1}, denoted as $AS^*(e_f^i, e_t^{i+1})$, based on the shortest path search in the action graph. Subsequently, we compute the sum of the action cost in $AS^*(e_f^i, e_t^{i+1})$ as

$$C^*\left(e_f^i, e_t^{i+1}\right) = \sum_{a \in AS^*(e_f^i, e_t^{i+1})} a.cost. \tag{7}$$

Osogami and Raymond [22] assumed that a route with total cost C matches the real path with a probability proportional to $\exp(-C)$. However, as the length of the trajectory segment between $\tilde{p}_i$ and $\tilde{p}_{i+1}$ has a great impact on the route cost $C^*(e_f^i, e_t^{i+1})$, we further normalize the cost by the great circle distance between $\tilde{p}_i$ and $\tilde{p}_{i+1}$, and the fraction of the number of GPS points in this segment over the total number of GPS points in trajectory T,

$$p\left(e_f^i, e_t^{i+1}\right) = \exp\left(-\frac{N(\tilde{p}_i, \tilde{p}_{i+1})}{n \cdot dist(\tilde{p}_i, \tilde{p}_{i+1})} \cdot C^*\left(e_f^i, e_t^{i+1}\right)\right), \tag{8}$$

where $N(\tilde{p}_i, \tilde{p}_{i+1})$ represents the number of GPS points in the segment between key points $\tilde{p}_i$ and $\tilde{p}_{i+1}$, n is the total number of GPS points in the input trajectory T, and $dist(\tilde{p}_i, \tilde{p}_{i+1})$ represents the great circle distance between points $\tilde{p}_i$ and $\tilde{p}_{i+1}$.

A global candidate path for the entire trajectory T goes through the candidate road segments of every key point $\tilde{p}_i$ in temporal order: $e_{k_1}^1 \to e_{k_2}^2 \to \cdots \to e_{k_{\tilde{n}}}^{\tilde{n}}$. We estimate the likelihood of a global candidate path by combining Equations (6) and (8):

$$\begin{aligned} &l\left(e_{k_1}^1 \to e_{k_2}^2 \to \cdots \to e_{k_{\tilde{n}}}^{\tilde{n}}\right) \\ &\quad = p\left(e_{k_1}^1\right) \cdot p\left(e_{k_1}^1, e_{k_2}^2\right) \cdot p\left(e_{k_2}^2\right) \cdots p\left(e_{k_{\tilde{n}}}^{\tilde{n}}\right). \end{aligned} \tag{9}$$

ALGORITHM 1: Feature-based Map Matching.

Input: trajectory T and road network $G(V, E)$
Output: the matched path P

Find a set of key GPS points $\tilde{T} = \{\tilde{p}_1, \tilde{p}_2, \ldots, \tilde{p}_{\tilde{n}}\}$ in T by Douglas-Peucker algorithm
for *each segment s between* $\tilde{p}_i$ *and* $\tilde{p}_{i+1}$ **do**
  Extract path features based on s and $G(V, E)$
  Update action cost by Equation (5)
  Retrieve candidate road segments E^i and E^{i+1} for key points $\tilde{p}_i$ and $\tilde{p}_{i+1}$
  for $e_f^i \in E^i$, $e_t^{i+1} \in E^{i+1}$ **do**
    compute $p(e_f^i)$, $p(e_t^{i+1})$ by Equation (6)
    compute $p(e_f^i, e_t^{i+1})$ by Equation (8)
  end
end
Initialize $f[e_k^1] = p(e_k^1)$, $k = 1, 2, \ldots, 10$
for $i = 2$ *to* $\tilde{n}$ **do**
  for $e_t^i \in E^i$ **do**
    for $e_f^{i-1} \in E^{i-1}$ **do**
      $l = f[e_f^{i-1}] \cdot p(e_f^{i-1}, e_t^i) \cdot p(e_t^i)$
      if $l > f[e_t^i]$ **then**
        $f[e_t^i] = l$
        $pre[e_t^i] = e_f^{i-1}$
      end
    end
  end
end
$P = \arg\max_{e_{k_1}^1 \to e_{k_2}^2 \to \cdots \to e_{k_{\tilde{n}}}^{\tilde{n}}} f[e_{k_{\tilde{n}}}^{\tilde{n}}]$
return P

The candidate path with the highest likelihood is returned as the final map matching results, which can be efficiently solved by a dynamic programming technique:

$$P = \arg\max l\left(e_{k_1}^1 \to e_{k_2}^2 \to \cdots \to e_{k_{\tilde{n}}}^{\tilde{n}}\right). \tag{10}$$

Algorithm 1 outlines our feature-based map matching technique. As aforementioned, it detects a set of key GPS points and processes each segment in-between to calculate the likelihood values $p(e_f^i)$, $p(e_t^{i+1})$, and $p(e_f^i, e_t^{i+1})$. In terms of the global optimization, $f[e_k^i]$ records the highest likelihood of a candidate path ending at road segment e_k^i. $pre[e_k^i]$ caches a candidate road segment of the previous key point $\tilde{p}_{i-1}$ from which the highest likelihood $f[e_k^i]$ is obtained. It is easy to see that the match of the ending point $\tilde{p}_{\tilde{n}}$ is $\arg\max_{e_{k_{\tilde{n}}}^{\tilde{n}}} f[e_{k_{\tilde{n}}}^{\tilde{n}}]$, and $pre[e_{k_{\tilde{n}}}^{\tilde{n}}]$ records the match of the previous point $\tilde{p}_{\tilde{n}-1}$. Therefore, the rest of the path can be reversely recovered from $pre[]$, and so on.

4.3 Online Updating and Decoding

We have introduced in detail how our method processes trajectories in an offline manner. However, to meet the needs of real-time applications such as traffic sensing, online solution is always

desirable that performs map matching incrementally while future data points are yet to come. One straightforward strategy is to divide the trajectory into fixed-sized sequences and handle them independently [33]; i.e., the system waits and decodes only the new observations every D seconds where D is the window size. This results in the following changes compared with the offline map matching algorithm: (1) The trajectory input changes to a sub-trajectory that only contains the GPS points in the current window; (2) The parameter n in Equation (8) is revised to the number of GPS points in the current window, since the total number of GPS points in the whole trajectory is unknown; (3) The cost associated with the *road-related features*, C_{len} and C_{turn} in Equation (5), is pre-computed and cached to improve the system efficiency; (4) The cost associated with the *trajectory-related features*, C_{traj} in Equation (5), is calculated online once a new window of observations arrives. More specifically, talking about the action cost updating, C_{len} and C_{turn} remain unchanged when new observations arrive as the *road-related features* are independent of GPS points. Therefore, it is preferable to pre-compute C_{len} and C_{turn} for system efficiency concerns. However, the calculation of C_{traj} relates to the trajectory input and thus has to be computed online. Similar to traditional HMM-based map matching, our method computes the distance from a new observation to the nearby roads and updates the *trajectory-related features* on-the-go, but with a slight overhead to perform trajectory segmentation as introduced in Section 4.2.1.

Generally speaking, there is a tradeoff between the map matching accuracy and the output delay. A larger window size D usually leads to a more accurate matching results but a longer output delay, and vice versa [20, 28]. With more observations available (e.g., in the extreme case where the whole trajectory has been revealed before processing), the Douglas-Peucker algorithm can generate better trajectory simplification results and subsequently provide more accurate information that benefits the decoding process. The window size D should be chosen according to system requirements. Please note that it is also feasible to refine the predicted matches in previous windows by incorporating new observations to perform map matching repeatedly. By doing so, increasingly improved map matching results can be obtained for each GPS point until the whole trajectory has been revealed. We have evaluated both the offline execution time and the online output delay of our proposed method. Experimental results show that our method always achieves the best map matching accuracy with competitive computational cost when comparing to the state-of-the-art map matching algorithms.

4.4 Discussion on Supplementary Data Sources

Map matching problems have been studied for decades. But due to the lack of a public large-scale information-rich testing dataset, researchers mostly collect their own dataset for algorithm design and evaluation [1, 3]. Additional information other than GPS can help significantly improve map matching results. Here, we introduce three types of such important supplementary information and briefly discuss how to integrate each of them into our proposed system.

4.4.1 GPS Accuracy Measures. GPS devices can provide users an accuracy measure associated with each latitude-longitude coordinate pair. This number indicates the degree of closeness between the GPS reading and the true location. While the GPS accuracy measure can be useful for automatic positioning data correction [32, 36], which serves as a preprocessing step before applying map matching algorithms, it is also possible to integrate the GPS accuracy measure in map matching algorithms directly. For example, in our proposed system, parameter σ in Equation (6) models the GPS noise. Instead of using a global constant value, local σ settings per GPS point can be estimated based on the GPS accuracy measures. Moreover, in our system's trajectory segmentation module, GPS accuracy measure can be used to avoid selecting key points with low accuracy by modeling the weight of each GPS point in trajectory simplification algorithms [15].

Table 1. Number of Tracks in the Worldwide GPS Dataset Tagged with Different Features

U-turns	Hives	Loops
25	3	24
Gaps	Congruence issues	No tags
73	20	19

4.4.2 Smartphone Internal Sensors. With the ubiquity of sensor-equipped smartphones, it has become increasingly feasible to leverage other sensors (e.g., WiFi and inertial sensors) in addition to GPS to enrich digital maps [2, 29]. Current public maps only provide users the basic road networks without road features. But with the availability of various smartphone's sensor data, rich road semantics (e.g., tunnels, curves, bridges, speed bumps) can be automatically detected using a decision-tree classifier [3]. Road features have impacts on drivers' route choices, and therefore can be taken into consideration in the action cost modeling in our proposed system.

4.4.3 Large Historical GPS Trajectories. Dense and large users' historical GPS trajectories can be very helpful in developing map matching algorithms, but at the same time much more difficult to acquire compared with smartphone's sensor data. Similar to the discussions in Section 4.4.2, historical GPS trajectories can be used to detect road semantics, model action cost, and derive travel patterns [22, 40]. Moreover, the historical trajectories can also be used to detect missing roads in digital maps [31]. In most map matching systems, researchers assume that the map data is complete, which is unfortunately not always true in real-life scenarios. Roads get built and removed, an inaccurate map will result in failed or incorrect matches between GPS points and road segments. This problem can be (partially) solved if large historical GPS trajectories are available, and thus obtaining improved map matching results.

5 EVALUATION

We implemented the proposed map matching algorithm and evaluated both its effectiveness and efficiency. The evaluation consists of three steps. The first part introduces the experimental setup and the details of the testing dataset. The second part estimates the value of the key parameter in our proposed model. And the third part verifies the offline and online version of our proposed approach by comparing it to the state-of-the-art map matching methodologies.

5.1 Dataset and Experimental Setup

The dataset we used for evaluation is a large-scale real dataset that consists of 100 GPS tracks all over the world [14]. Each track is associated with a map and a correctly map-matched route. Moreover, the tracks are labeled with a selection of features that may pose difficulties to map matching algorithms including:

- *u-turns*: the vehicle turned 180° and reversed the direction of travel
- *hives*: large numbers of points packed in a small area
- *loops*: the vehicle was traveling in circles
- *gaps*: temporal gaps existing in the track
- *severe congruence issues*: situations where the map and the track are incongruent or dissimilar

Table 2. Trajectory Statistics of the Worldwide GPS Dataset

Statistics	Track length (m)	Track duration (s)	Number of GPS points
Min.	5,005.9	265.0	262.0
Max.	94,172.4	15,4030.0	19,313.0
Average	26,754.2	4,950.7	2,472.5

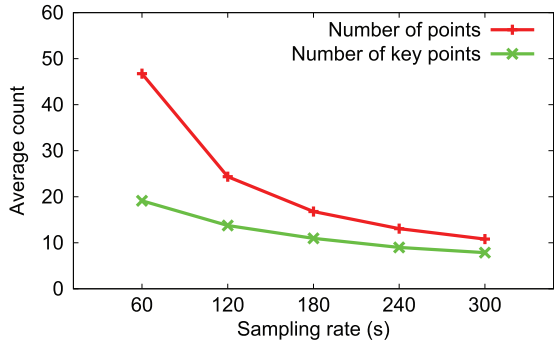


Fig. 9. Average number of GPS points and segments before and after trajectory simplification.

The number of tracks that are labeled with each of the five features is reported in Table 1. Additionally, there are 19 tracks formed by high quality GPS data with no tags associated. As illustrated in Table 2, the length of the tracks varies from 5 to 100 kilometers, and the dataset contains 247,251 points in total with a sampling rate of 1Hz. The average length and the average duration of the 100 tracks are 26.8km and 4950.7s, respectively. For more details, please refer to the dataset paper [14].

The sampling interval of real GPS data varies significantly from less than 1min to more than 5min [38]. Therefore, it is important to evaluate the robustness of the proposed map matching algorithm when applied to low sampling rate GPS data. To achieve this goal, we generate five datasets by subsampling the original data with different sampling intervals of 60s, 120s, 180s, 240s, and 300s, respectively. The average number of GPS points per track versus the sampling rate is illustrated in Figure 9. Moreover, we also report the average number of the key points per track detected based on the trajectory simplification technique as introduced in Section 4.2.1.

From the results, we can see the shape of a trajectory can be described with much fewer points especially when the sampling rate is relatively high. For example, with a sampling interval of 60s, the number of key GPS points detected, which is 19.12, is less than half of the total number of GPS points, which is 46.74, per track on average. The shape of a trajectory can be better preserved with more key points by setting the distance threshold ϵ required by the Douglas-Peucker algorithm to a smaller value. However, this is not necessary, because our method works well as long as the segments between key points do not contain any loops. Thereby, we intuitively set the distance threshold $\epsilon = 0.001$ throughout the experiments, which obtains excellent map matching accuracies when comparing to other existing methods.

5.2 Parameter Estimation

The balancing factor ω in Equation (5) is a key parameter in our model. It controls the weights of road length and transition angle in the action cost estimation. Here, we examine the map matching results obtained by setting ω to different values. The matching accuracy is measured by the Route

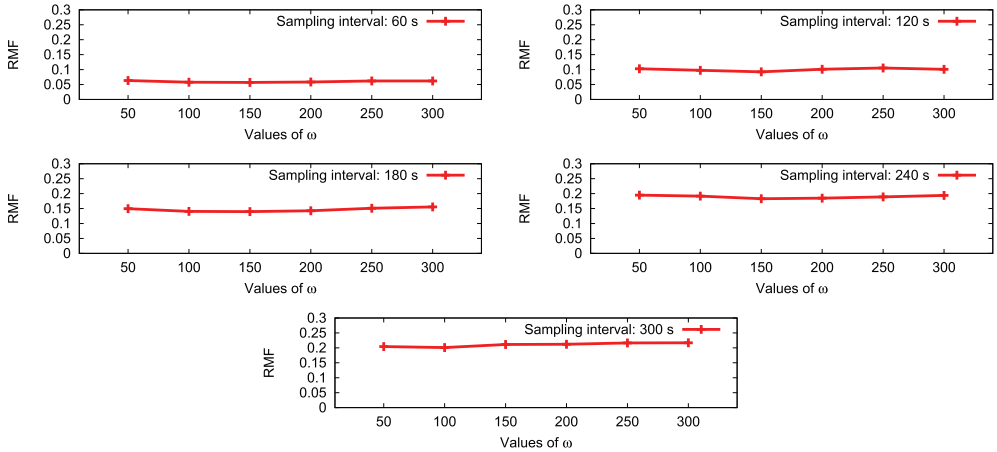


Fig. 10. Route mismatch fraction plot based on different combinations of parameter ω and sampling intervals.

Mismatch Fraction (RMF) proposed by Newson and Krumm [21]. This measure computes the total length of the false positive road segments, denoted as d_+, and the total length of the false negative road segments, denoted as d_-. Thus, the summation of d_+ and d_- is the total mismatched distance. Let d_0 represent the total length of the real path. RMF quantifies the map matching error by the fraction of $(d_+ + d_-)/d_0$, so that a small RMF value indicates the map matching results are more similar to the real path:

$$RMF = (d_+ + d_-)/d_0. \tag{11}$$

Figure 10 shows the average route mismatch fraction plotted against ω on trajectories sampled with different intervals, which are 60s, 120s, 180s, 240s, 300s, respectively. As can be seen, the best map matching results are obtained with ω setting to 100 or 150 in all the five subfigures. The accuracy of the predicted path slightly decreases with smaller or larger ω values, but the variations are not significant within a range of ω settings (e.g., $50 < \omega < 300$). Additionally, the variation trend of the route mismatch fraction against ω is similar in all the five subfigures with different sampling intervals, which indicates that our method is robust to trajectories with various sampling intervals or even temporal gaps.

Generally speaking, improved map matching results can be obtained with a range of ω values. But if large historical GPS data are available, such parameters can be better tuned by machine learning techniques such as the maximum entropy inverse reinforcement learning [42]. Regional factors can also be considered in the user behavior modeling based on training data with good geospatial coverage all over the world.

5.3 Comparison with the State-of-the-art

We evaluated the effectiveness and efficiency of the proposed map matching algorithm. In addition to the Route Mismatch Fraction (RMF) [21], we also report the map matching precision, recall, and F_1 score [3] for accuracy comparison. The map matching precision is defined as the ratio between the distance of the matching sequence and the total distance of the predicted trace, and recall is defined as the ratio between the distance of the matching sequence and the total distance of the

Table 3. Average Route Mismatch Fraction Comparison of Offline Map Matching Over the 100 Test Trajectories

Sampling rate (s)	60	120	180	240	300
HMM-based	0.0657	0.1189	0.1773	0.2610	0.2856
IRL-based	0.0854	0.1157	0.1621	0.2085	0.2147
Feature-based	0.0578	0.0975	0.1406	0.1917	0.2009

ground truth trace:

$$Precision = \frac{Total\ distance\ of\ matching\ sequence}{Total\ distance\ of\ predicted\ trace}, \tag{12}$$

$$Recall = \frac{Total\ distance\ of\ matching\ sequence}{Total\ distance\ of\ ground\ truth\ trace}. \tag{13}$$

We also calculate the F_1 score, which is defined as

$$F_1 = 2 \times \frac{Precision \times Recall}{Precision + Recall}. \tag{14}$$

Next, we evaluate the effectiveness of the proposed method using low sampling rate GPS data with various challenging features. We compare our method with the following two state-of-the-art algorithms and report route mismatch fraction based on different combinations of track features and sampling rates:

- **HMM-based Map Matching:** It leverages a Hidden Markov Model (HMM) to find the most likely sequence of road segments by conjunctively considering the measurement noise and the road network topology [21].
- **IRL-based Map Matching:** It extends the HMM-based map matching approach and estimates the transition probability between two road segments by a fusion of transition angle and travel distance, which is trained by Inverse Reinforcement Learning (IRL) [22].

Throughout the experiments, the standard deviation of GPS measurements σ in Equation (6) is set to 10 in all methods. The scaling factor β, which is used to estimate the transition probabilities in HMM-based and IRL-based methods, is also set to 10, but qualitative findings hold for a range of parameter settings [22]. The balancing factor ω in Equation (5) is set to 100 in both the IRL-based method and our proposed feature-based method. For efficiency concerns, we only retrieve road segments that are within 200m of each GPS point to construct the candidate set in HMM-based and IRL-based methods. We implemented all the three methods with Python and conducted all the experiments on a server with two Intel Xeon E5-2680 v3 2.5GHz CPUs and 512GB RAM. PostgreSQL and pgRouting were installed and utilized as the database for efficient geospatial search.

5.3.1 Offline Map Matching Results. We first evaluate the performance of our offline map matching method. We conduct experiments on the worldwide testing dataset and report the average RMF, precision, recall, F_1 score, and the processing time per GPS point over the 100 GPS trajectories in Tables 3, 4, and 5, respectively.

We first compare the RMF based on five different sampling rates, and our proposed method obtain the best map matching accuracy in all cases. As illustrated in Table 3, our algorithm outperformed the HMM-based method by 12.0%, 18.0%, 20.7%, 26.6%, and 29.7% with sampling intervals 60s, 120s, 180s, 240s, and 300s, respectively. Traditional HMM-based methods simply assume that drivers would always take the shortest path in terms of travel distance between two candidate

Table 4. Average Precision, Recall, and F_1 Score Comparison of Offline Map Matching over the 100 Test Trajectories

(a) Precision

Sampling rate (s)	60	120	180	240	300
HMM-based	0.9696	0.9514	0.9236	0.8867	0.8875
IRL-based	0.9610	0.9524	0.9331	0.9103	0.9095
Feature-based	0.9710	0.9567	0.9372	0.9149	0.9131

(b) Recall

Sampling rate (s)	60	120	180	240	300
HMM-based	0.9669	0.9347	0.9018	0.8577	0.8517
IRL-based	0.9555	0.9316	0.9036	0.8740	0.8667
Feature-based	0.9747	0.9458	0.9223	0.8877	0.8789

(c) F_1 Score

Sampling rate (s)	60	120	180	240	300
HMM-based	0.9676	0.9420	0.9114	0.8704	0.8668
IRL-based	0.9576	0.9407	0.9162	0.8900	0.8849
Feature-based	0.9721	0.9507	0.9286	0.8998	0.8937

Table 5. Average Processing Time (in Seconds) per GPS Point of Offline Map Matching over the 100 Test Trajectories

Sampling rate (s)	60	120	180	240	300	Mean
HMM-based	0.25	0.23	0.22	0.22	0.21	0.23
IRL-based	0.61	0.60	0.59	0.61	0.57	0.60
Feature-based	0.67	0.86	0.96	0.90	0.96	0.87

road segments, which is not necessarily true especially for low sampling rate GPS data. With a growing sampling interval, the uncertainty in GPS trajectories is also increasing, because most of the movement details are lost. Although the HMM-based method is able to handle data sparsity to some extent (e.g., 30s [21]), its effectiveness decreases dramatically as the sampling interval grows much larger. To obtain improved results, Osogami and Raymond proposed an IRL-based method based on the assumption that drivers would take the more "natural" path instead of the shortest path [22]. From Table 3, we can see that the IRL-based technique outperformed the HMM-based method for sampling periods larger than two minutes, but it only achieved similar or even worse results for the cases reported in the first two columns. It indicates that the IRL-based method is more susceptible to data noise as it tries to find a matching road segment for every GPS point. Our method, however, segments a trajectory and aims to find a matching route for every segment. Therefore, the robustness of the proposed feature-based method has been greatly improved by simultaneously considering all the points in one segment while performing map matching. Compared with the IRL-based method, our approach achieved improvements of 32.3%, 15.7%, 13.3%, 8.1%, and 6.4% in the five groups with different sampling intervals, respectively. In terms of precision, recall, and F_1 score, Tables 4(a) and 4(b) show that our proposed method effectively improved both precision and recall at all the five sampling rates. Tables 4(c) reports the F_1 score, which

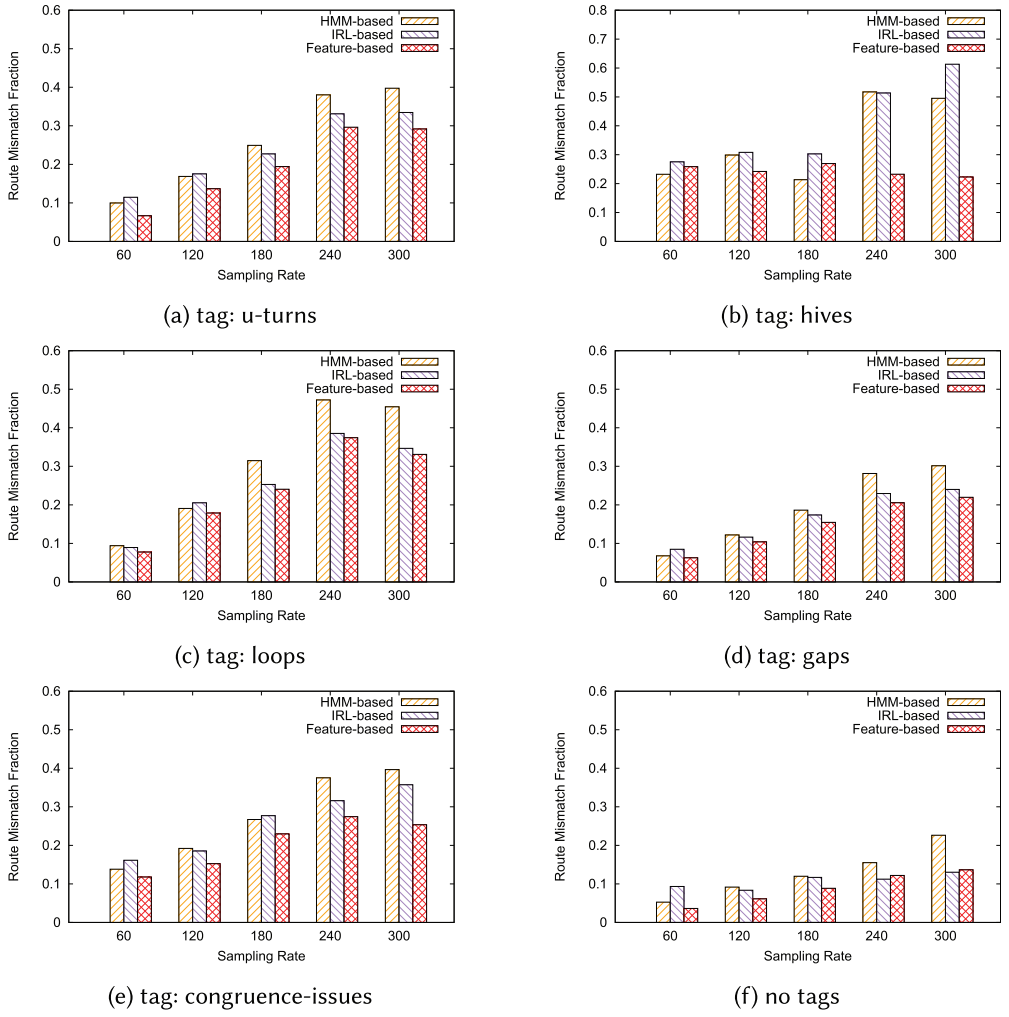


Fig. 11. Route mismatch error comparison of offline map matching with different sampling rate.

is the harmonic mean of precision and recall, and provides a single measurement for the system. In terms of efficiency, we report the processing time per GPS point of the three methods in Table 5. Except for the algorithm complexity, a number of other factors also have an impact on the processing time including the road network complexity, GPS sampling rate, the existence of gaps in trajectories, and so on. We report the mean average value in the last column, and as we can see that the HMM-based method is the most efficient but works less satisfactory in terms of the map matching accuracy. Our proposed method additionally considers the shape of the trajectory and the transition angles between roads, and it obtains the best map matching accuracy with a reasonable increase in the processing time.

Next, we evaluate the effectiveness of our proposed approach for trajectories with different challenging features. The average route mismatch fractions are reported in Figure 11. Generally speaking, the qualitative findings are mostly the same as the results reported in Table 3. By segmenting an input trajectory into relatively straight subsequences, our method is able to obtain the highest

Table 6. Average Route Mismatch Error Comparison of Online Map Matching with Different Window Sizes

Window size (s)	120	180	240	300	360
HMM-based	0.0814	0.0754	0.0750	0.0705	0.0708
IRL-based	0.1077	0.0888	0.0947	0.0932	0.0872
Feature-based	0.0851	0.0760	0.0755	0.0679	0.0660

accuracy even when applying to trajectories with *u-turns* and *loops*. For trajectories labeled by *hives* where a large volume of GPS points are packed in a small area, the IRL-based method performed much worse than the rest of the cases due to its sensitivity to data noise. However, Figure 11(f) shows the comparison when dealing with good quality data without any challenging features associated. Although the IRL-based method performed comparatively well or even slightly better than the proposed approach in some cases, it failed to maintain the good performance in other groups with decreasing sampling intervals. While in most of the cases, it is easy for humans to manually judge which is the correct route of a track on a map, there are situations where the correct matching is not clear even to us. Kubička et al. labeled such situations as *severe congruence issues* when the map and the track are incongruent or dissimilar [14]. Please note that although the hand-correction made by humans cannot be considered as the ground truth data for trajectories annotated with *congruence-issues*, the results shown in Figure 11(e) indicate that the route predicted by our proposed algorithm is more similar to human intuition. This is also important, because it is reasonable to assume that a human mind is usually superior in matching GPS tracks on a digital map.

5.3.2 Online Map Matching Results. As our method focuses on low-sampling-rate trajectories, we evaluate the route mismatch fraction and the online output latency with the sampling interval set to 60s. We compare the RMF, precision, recall, F_1 score, and the latency based on five different window sizes, which are 120s, 180s, 240s, 300s, and 360s, and report the average results over the 100 test GPS trajectories in Tables 6, 7, and 8, respectively.

As can be seen, all three methods achieve high precision, recall, and F_1 scores. By comparing Tables 6 and 8, we observe that a larger window size D results in a longer output delay but more accurate map matching results. The HMM-based method outperformed our proposed approach when the window size was smaller than 240s. This is because the HMM-based method processed each GPS point individually and was less susceptible to the number of GPS observations available before running the algorithm. Comparatively, our proposed method reduces the data noise by performing trajectory segmentation and processes one segment at one time. With more GPS observations, the shape of the trajectory can be better revealed and thus benefit our proposed decoding process. As in this experiment, our proposed method obtained the best map matching accuracy with a window size larger than 240s. In the case of $D = 360$s, where enough GPS observations are available in a temporal window for processing, our method successfully reduced the RMF by 6.8% and 24.3% compared with the HMM-based method and the IRL-based method, respectively. In terms of the online latency, we can see from Table 8 that the HMM-based method achieves the highest efficiency. As both our method and the IRL-based method consider the transition angles between two road segments, a slight overhead has been introduced due to the increase of the shortest path search space. Moreover, our method performs an additional trajectory simplification step. Fortunately, the Douglas-Peucker algorithm we adopted is highly efficient. Overall, the computational cost of our proposed method is competitive with the other two state-of-the-art map matching algorithms. The latency increase in our method is less than 3s, which is small compared with the window size.

Table 7. Average Precision, Recall, and F_1 Score Comparison of Online Map Matching with Different Window Sizes

(a) Precision

Window size (s)	120	180	240	300	360
HMM-based	0.9573	0.9615	0.9634	0.9654	0.9660
IRL-based	0.9425	0.9542	0.9525	0.9535	0.9576
Feature-based	0.9572	0.9620	0.9642	0.9674	0.9685

(b) Recall

Window size (s)	120	180	240	300	360
HMM-based	0.9639	0.9661	0.9642	0.9667	0.9659
IRL-based	0.9530	0.9602	0.9556	0.9562	0.9578
Feature-based	0.9634	0.9674	0.9632	0.9677	0.9680

(c) F_1 Score

Window size (s)	120	180	240	300	360
HMM-based	0.9600	0.9631	0.9631	0.9654	0.9653
IRL-based	0.9470	0.9565	0.9533	0.9541	0.9570
Feature-based	0.9592	0.9637	0.9629	0.9668	0.9676

Table 8. Average Latency (in Seconds) Comparison of Online Map Matching with Different Window Sizes

Window size (s)	120	180	240	300	360
HMM-based	51.90	79.99	109.16	139.12	168.42
IRL-based	52.39	81.02	110.35	140.98	170.32
Feature-based	52.86	81.52	110.99	141.47	171.07

To understand the impact of trajectory characteristics on map matching, we illustrate the RMF versus the corresponding latency for trajectories with different challenging features in Figure 12. Generally, the qualitative findings remain the same as the average results reported in Tables 6 and 8. A larger window size leads to more accurate map matching results and our proposed method outperforms its competitors when the window size is greater than 240. But, we also observe two interesting phenomena from the results shown in Figure 12. First, for trajectories labeled by *hives*, a larger window size surprisingly resulted in an accuracy decrease in the HMM-based and the IRL-based methods. This is because these two methods processed each GPS point individually and got confused when the position change in GPS was mostly caused by data noise. Our proposed method, however, was able to successfully group the GPS points into one segment and effectively reduce the impact of data noise in this situation. The second observation concerns trajectories labeled by *loops*. As we can see from Figure 12(c), our proposed method performed less effectively in this case. Trajectories with loops usually contain a large number of turnings, thus the shape of such trajectories become more difficult to be preserved when performing the trajectory simplification. To decrease the distance threshold ϵ required by the Douglas-Peucker algorithm may help in this case. To wait until the whole trajectory to be revealed before applying the Douglas-Peucker algorithm also helps generate better trajectory simplification results. As shown in Figure 11(c) in

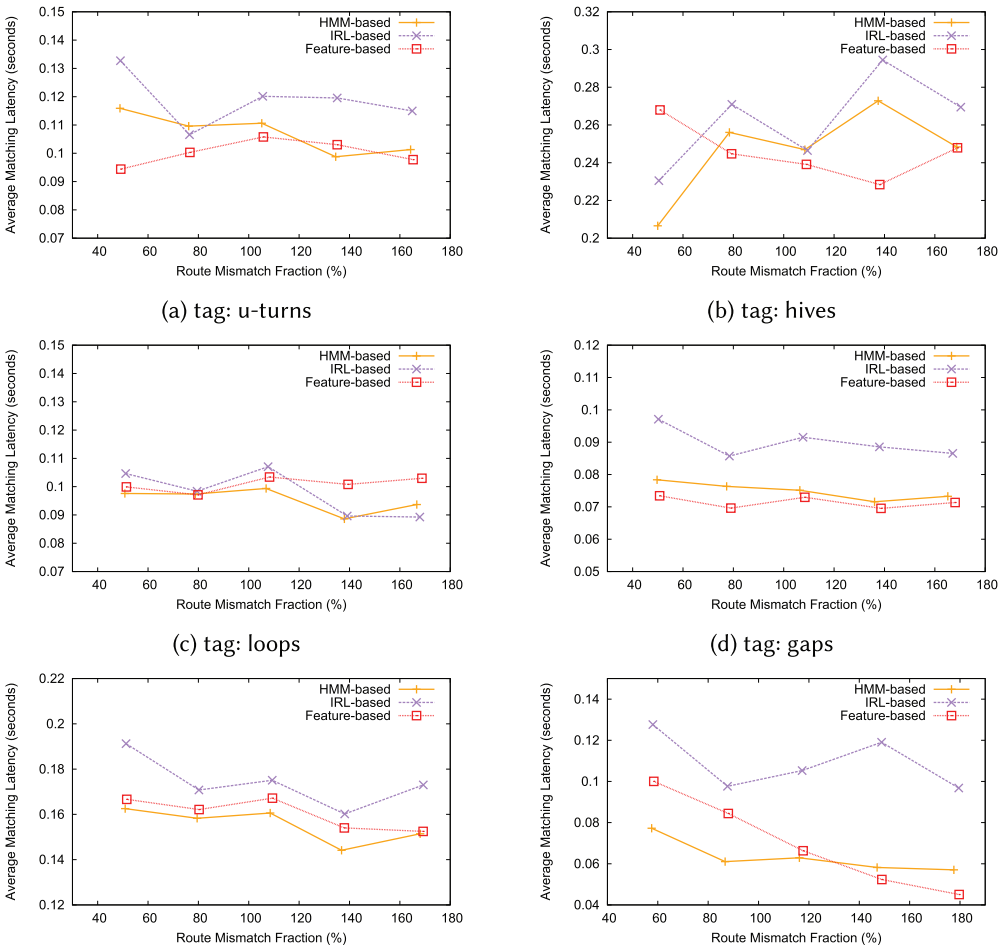


Fig. 12. The route mismatch fraction (in percentage) and latency (in seconds) of map matching results for trajectories with different challenging features.

the offline map matching evaluation, our method outperforms its competitors even when handling trajectories with *loops*.

6 CONCLUSION AND FUTURE WORK

We have presented a novel feature-based map matching framework that models the likelihood of a candidate route based on both GPS observations and human factors. To improve the system robustness to data noise, we simultaneously process multiple GPS points at a time by segmenting the input trajectory into coherent subsequences. Road characteristics such as length and transition angles are incorporated as these factors have a major affect on a driver's route choice. We conduct extensive experiments on a challenging real-world dataset to evaluate both the offline and the online versions of our proposed approach. We compare our method to two state-of-the-art map matching algorithms in terms of effectiveness and efficiency, and the experimental results show

that our proposed method is able to achieve the best map matching accuracy with a reasonable tradeoff in the processing time.

In the future, we plan to explore the fusion of additional features for user behavior cost estimation. With a digital map that contains more semantic information such as road type and speed limit, improved results can be obtained by integrating these road characteristics in the action cost estimation. As discussed in Section 4.4, rich road semantics can also be detected and inferred when more supplementary data sources become available. Moreover, the online version of our proposed method can be further optimized. For example, we currently divide the trajectory into fixed-sized sequences and process each segment individually. This part can be improved by developing more advanced online map matching strategies that perform tradeoff analysis and decide the optimal timing for decoding.

REFERENCES

[1] Essam Algizawy, Tetsuji Ogawa, and Ahmed El-Mahdy. 2017. Real-time large-scale map matching using mobile phone data. *ACM Trans. Knowl. Discov. Data* 11, 4 (2017), 52:1–52:38.

[2] H. Aly, A. Basalamah, and M. Youssef. 2017. Automatic rich map semantics identification through smartphone-based crowd-sensing. *IEEE Trans. Mobile Comput.* 16, 10 (2017), 2712–2725.

[3] Heba Aly and Moustafa Youssef. 2015. semMatch: Road semantics-based accurate map matching for challenging positioning data. In *Proceedings of the ACM Special Interest Group on Spatial Information (SIGSPATIAL'15)*. 5:1–5:10.

[4] Sotiris Brakatsoulas, Dieter Pfoser, Randall Salas, and Carola Wenk. 2005. On map-matching vehicle tracking data. In *Proceedings of the Conference on Very Large Data gases (VLDB'05)*. 853–864.

[5] Maike Buchin, Anne Driemel, Marc van Kreveld, and Vera Sacristán. 2011. Segmenting trajectories: A framework and algorithms using spatiotemporal criteria. *J. Spatial Info. Sci.* 2011, 3 (2011), 33–63.

[6] S. S. Chawathe. 2007. Segment-based map matching. In *Proceedings of the IEEE Intelligent Vehicles Symposium*. 1190–1197.

[7] Shunkai Fang and Roger Zimmermann. 2011. EnAcq: Energy-efficient GPS trajectory data acquisition based on improved map matching. In *Proceedings of the ACM Special Interest Group on Spatial Information (SIGSPATIAL'11)*. 221–230.

[8] Preeti Goel, Lars Kulik, and Kotagiri Ramamohanarao. 2016. Privacy-aware dynamic ride sharing. *ACM Trans. Spatial Algor. Syst.* 2, 1 (2016), 4:1–4:41.

[9] C. Y. Goh, J. Dauwels, N. Mitrovic, M. T. Asif, A. Oran, and P. Jaillet. 2012. Online map-matching based on hidden Markov model for real-time traffic sensing applications. In *IEEE Intell. Transport. Syst.* 776–781.

[10] Joshua S. Greenfeld. 2002. Matching GPS observations to locations on a digital map. In *Proceedings of the Transportation Research Board Annual Meeting*.

[11] F. Gustafsson, F. Gunnarsson, N. Bergman, U. Forssell, J. Jansson, R. Karlsson, and P. J. Nordlund. 2002. Particle filters for positioning, navigation, and tracking. *IEEE Trans. Signal Process.* 50, 2 (2002), 425–437.

[12] Ralf Hartmut Güting, Fabio Valdés, and Maria Luisa Damiani. 2015. Symbolic trajectories. *ACM Trans. Spatial Algor. Syst.* 1, 2 (2015), 7:1–7:51.

[13] John Hershberger and Jack Snoeyink. 1992. *Speeding Up the Douglas-Peucker Line-Simplification Algorithm*. Technical Report. University of British Columbia.

[14] M. Kubička, A. Cela, P. Moulin, H. Mounier, and S. I. Niculescu. 2015. Dataset for testing and training of map-matching algorithms. In *Proceedings of the IEEE Intelligent Vehicles Symposium*. 1088–1093.

[15] Hengfeng Li, Lars Kulik, and Kotagiri Ramamohanarao. 2014. Spatio-temporal trajectory simplification for inferring travel paths. *ACM SIGSPATIAL*. 63–72.

[16] Lin Liao, Donald J. Patterson, Dieter Fox, and Henry Kautz. 2007. Learning and inferring transportation routines. *Artificial Intell.* 171, 5–6 (2007), 311–331.

[17] Kuien Liu, Yaguang Li, Fengcheng He, Jiajie Xu, and Zhiming Ding. 2012. Effective map-matching on the most simplified road network. In *Proceedings of the ACM Special Interest Group on Spatial Information (SIGSPATIAL'12)*. 609–612.

[18] Yankai Liu and Zhuo Li. 2017. A novel algorithm of low sampling rate GPS trajectories on map-matching. *EURASIP J. Wireless Commun. Netw.* 2017, 1 (2017), 30.

[19] Cheng Long, Raymond Chi-Wing Wong, and H. V. Jagadish. 2013. Direction-preserving trajectory simplification. *VLDB Endow.* 6, 10 (2013), 949–960.

[20] Yin Lou, Chengyang Zhang, Yu Zheng, Xing Xie, Wei Wang, and Yan Huang. 2009. Map-matching for low-sampling-rate GPS trajectories. In *Proceedings of the ACM Special Interest Group on Spatial Information (SIGSPATIAL'09)*. 352–361.

[21] Paul Newson and John Krumm. 2009. Hidden Markov map matching through noise and sparseness. In *Proceedings of the ACM Special Interest Group on Spatial Information (SIGSPATIAL'09)*. 336–343.
[22] Takayuki Osogami and Rudy Raymond. 2013. Map matching with inverse reinforcement learning. In *Proceedings of the International Joint Conferences on Artificial Intelligence (IJCAI'13)*. 2547–2553.
[23] O. Pink and B. Hummel. 2008. A statistical approach to map matching using road network geometry, topology and vehicular motion constraints. In *11th International IEEE Conference on Intelligent Transportation Systems*. 862–867.
[24] Mohammed A. Quddus, Washington Yotto Ochieng, Lin Zhao, and Robert B. Noland. 2003. A general map matching algorithm for transport telematics applications. *GPS Solutions* 7, 3 (2003), 157–167.
[25] Mohammed A. Quddus, Robert B. Noland, and Washington Y. Ochieng. 2006. A high accuracy fuzzy logic based map matching algorithm for road transport. *J. Intell. Transport. Syst.* (2006), 103–115.
[26] Iulian Sandu Popa, Karine Zeitouni, Vincent Oria, and Ahmed Kharrat. 2015. Spatio-temporal compression of trajectories in road networks. *Geoinformatica* 19, 1 (2015), 117–145.
[27] Renchu Song, Weiwei Sun, Baihua Zheng, and Yu Zheng. 2014. PRESS: A novel framework of trajectory compression in road networks. *Proc. VLDB Endow.* 7, 9 (2014), 661–672.
[28] Rastislav Srámek, Brona Brejová, and Tomás Vinar. 2007. On-Line viterbi algorithm for analysis of long biological sequences. In *Proceedings of the Workshop on Algorithms in Bioinformatics (WABI'07)*.
[29] Arvind Thiagarajan, Lenin Ravindranath, Hari Balakrishnan, Samuel Madden, and Lewis Girod. 2011. Accurate, low-energy trajectory mapping for mobile devices. In *Proceedings of the Conference on Networked Systems Design and Implementation*. 267–280.
[30] Arvind Thiagarajan, Lenin Ravindranath, Katrina LaCurts, Samuel Madden, Hari Balakrishnan, Sivan Toledo, and Jakob Eriksson. 2009. VTrack: Accurate, energy-aware road traffic delay estimation using mobile phones. In *Proceedings of the Conference on Embedded Networked Sensor Systems*. 85–98.
[31] Fernando Torre, David Pitchford, Phil Brown, and Loren Terveen. 2012. Matching GPS traces to (possibly) incomplete map data: Bridging map building and map matching. In *Proceedings of the ACM Special Interest Group on Spatial Information (SIGSPATIAL'12)*. 546–549.
[32] Guanfeng Wang, Beomjoo Seo, and Roger Zimmermann. 2012. Automatic positioning data correction for sensor-annotated mobile videos. *ACM SIGSPATIAL*. 470–473.
[33] Guanfeng Wang and Roger Zimmermann. 2014. Eddy: An error-bounded delay-bounded real-time map matching algorithm using HMM and online viterbi decoder. In *Proceedings of the ACM Special Interest Group on Spatial Information (SIGSPATIAL'14)*. 33–42.
[34] Christopher E. White, David Bernstein, and Alain L. Kornhauser. 2000. Some map matching algorithms for personal navigation assistants. *Transport. Res. Part C: Emerg. Technol.* (2000), 91–108.
[35] Yifang Yin, Beomjoo Seo, and Roger Zimmermann. 2015. Content vs. context: Visual and geographic information use in video landmark retrieval. *ACM Trans. Multimedia Comput. Commun. Appl.* 11, 3 (2015), 39:1–39:21.
[36] Yifang Yin, Guanfeng Wang, and Roger Zimmermann. 2016. Automatic geographic metadata correction for sensor-rich video sequences. *ACM SIGSPATIAL*. 24:1–24:10.
[37] Yifang Yin, Luming Zhang, and Roger Zimmermann. 2015. Exploiting spatial relationship between scenes for hierarchical video geotagging. In *Proceedings of the 5th ACM on International Conference on Multimedia Retrieval (ICMR'15)*. 363–370.
[38] Jing Yuan, Yu Zheng, Chengyang Zhang, Wenlei Xie, Xing Xie, Guangzhong Sun, and Yan Huang. 2010. T-drive: Driving directions based on taxi trajectories. In *Proceedings of the ACM Special Interest Group on Spatial Information (SIGSPATIAL'10)*. 99–108.
[39] J. Yuan, Y. Zheng, C. Zhang, X. Xie, and G. Z. Sun. 2010. An interactive-voting based map matching algorithm. In *Proceedings of the International Conference on Mobile Data Management (MDM'10)*. 43–52.
[40] Kai Zheng, Yu Zheng, Xing Xie, and Xiaofang Zhou. 2012. Reducing uncertainty of low-sampling-rate trajectories. In *Proceedings of the IEEE International Conference on Data Engineering (ICDE'12)*. 1144–1155.
[41] Yu Zheng. 2015. Trajectory data mining: An overview. *ACM Trans. Intell. Syst. Technol.* 6, 3 (2015), 29:1–29:41.
[42] Brian D. Ziebart, Andrew L. Maas, J. Andrew Bagnell, and Anind K. Dey. 2008. Maximum entropy inverse reinforcement learning. In *Proceedings of the Innovative Applications of Artificial Intelligence Conference (AAAI'08)*. 1433–1438.

Received January 2017; revised March 2018; accepted April 2018