Map Matching Algorithms: An Experimental Evaluation

Na Ta[1(✉)], Jiuqi Wang[2], and Guoliang Li[3]

[1] School of Journalism and Communication, Renmin University of China, Beijing 100872, China
tanayun@ruc.edu.cn
[2] College of Software, Beihang University, Beijing 100191, China
wangjiuqi1@163.com
[3] Department of Computer Science, Tsinghua University, Beijing 100084, China
liguoliang@tsinghua.edu.cn

Abstract. Map matching is an important operation of location-based services, which matches raw GPS trajectories onto real road networks, and facilitates tasks of urban computing, such as intelligent traffic systems, etc. More than ten algorithms have been proposed to address this problem in the recent decade. However, existing algorithms have not been thoroughly compared under the same experimental framework. For example, some algorithms are tested only on specific datasets. This makes it rather difficult for practitioners to decide which algorithms should be used for various scenarios. To address this problem, in this paper we provide a survey on a wide spectrum of existing map matching algorithms, classify them into different categories based on their main techniques, and compare them through extensive experiments on a variety of real-world and synthetic datasets with different characteristics. We also report comprehensive findings obtained from the experiments and provide new insights about the strengths and weaknesses of existing map matching algorithms which can guide practitioners to select appropriate algorithms for various scenarios.

1 Introduction

Given a set of raw GPS trajectories generated by vehicles on an urban road network, the map matching algorithm is to align each raw trajectory onto underlying road network, where a raw trajectory is a sequence of sampling points of discrete locations at each sampling time, and a road network is a graph of vertices and edges modeling an urban traffic network. The need of such algorithms arises because: (1) the GPS devices have measurement errors, which may incorrectly report the actual location of a vehicle, and (2) sampling rates are not always set to high frequency due to transmission, storage and other costs, making it hard to tell the exact route. Therefore, map matching is an important operation for applications utilizing trajectory data, such as data management for traffic analysis [6], frequent path finder [15], taxi pick-up recommending system [20],

Y. Cai et al. (Eds.): APWeb-WAIM 2018, LNCS 10988, pp. 182–198, 2018.
https://doi.org/10.1007/978-3-319-96893-3_14

discovery of functional urban zones [28], location-aware publish/subscribe framework in digital content communication [13], etc. The basic idea of map matching is to align each sampling point to a "proper" location along some real road, to recover the actual route travelled by the vehicle. To this end, a number of map matching algorithms have been proposed in the past two decades [1,3–12,14–29].

A typical map matching framework includes two steps (after proper preprocessing such as data cleaning and indexing): (1) candidate selection step: road segments or road vertices are selected as candidates of actual locations according to certain measurements, and (2) actual route construction step: the route with the highest matching score is selected as the actual route of the vehicle reporting that particular raw trajectory. The candidate selection algorithms are crucial in terms of the map matching quality, and they vary as different strategies are taken. For example, early algorithms use the closest road segment of each sampling point as their candidate road and connect all candidate segments as the actual route, while later algorithms would employ more sophisticated models (such as the Hidden Markov Model) to address the candidate selection step.

Existing map matching algorithms can be categorized by different perspectives. Algorithms in [18,26] can be used for off-line map matching tasks, and algorithms in [9,22,23] are proper for on-line map matching. Algorithms in [1,14,27] are designed for low sampling rate (no more than one sample point within a minute), while most algorithms can work better on higher sampling rate data sets. According to sampling points used in the candidate selection step, there are incremental [5,8,10,25] and global [5,14,26,29] map matching algorithms. Besides, map matching algorithms can also be classified into geometry-based [11], topology-based [4,5,22,26], probability-based [3,17,19,21], and advanced algorithms such as [16] utilizing the Hidden Markov Model.

However these algorithms have not been thoroughly compared under the same experimental framework. For example, most algorithms are tested only on specific datasets, and there is no uniform quality metrics to demonstrate qualities of these algorithms. This makes it rather difficult for practitioners to decide which algorithms should be used for various scenarios.

To address this problem, in this paper we thoroughly compare existing map matching algorithms on the same experimental framework. We make the following contributions. (1) We provide a comprehensive survey on a wide spectrum of existing map matching algorithms and classify them into different categories based on their techniques. (2) We compare existing algorithms through extensive experiments on a variety of real-world and synthetic datasets with different characteristics. (3) We report comprehensive findings obtained from the experiments and provide new insights about the strengths and weaknesses of existing algorithms which can guide practitioners to select appropriate algorithms for various scenarios.

2 Preliminaries

We introduce following concepts before we formally define the map matching problem.

Definition 1 (Trajectory).[1] *A trajectory T is a sequence of sample points, $T = \{p_1, p_2, \cdots, p_{|T|}\}$, where p_k is a sample point (i.e., a geo-location with a sampling timestamp), and $|T|$ is the number of sample points in T.*

Definition 2 (Road Network). *A road network is a directed graph $G(V, E)$, where $V = \{v_i(x_i, y_i)\}$ is the set of vertices, a vertex v_i is represented by a pair of latitude (x_i) and longitude (y_i); and $E = \{e_j(v_k, v_m)\}$ is the set of edges which are road segments directly connected by vertices in V.*

Thus an actual road is composed by one or more road segments sequentially connected by road vertices.

Definition 3 (Route). *Given two road vertices v_i and v_j, a route R is a sequence of connected road segments starting from v_i and ending at v_j.*

Therefore, the problem of map matching is to align a raw trajectory T to the underlying road network and find a **matching route** R of the highest matching quality to T, where matching quality can be measured by some *matching metrics*. We can broadly classify existing matching quality metrics into several categories: geometry-based, topology-based, probability-based and statistical metrics.

Geometry-Based Metrics. These metrics quantify the matching quality based on the similarity of geometry characteristics between a trajectory and a route, such as distance, angle between the two curves formed by the trajectory and the route on the digital map. These metrics are fit for high-sampling-rate trajectories with low measurement error. For low-sampling-rate trajectories, the connectivity between sampling points can not be measured properly. In early incremental map matching algorithms (e.g. [11]), nearest road vertices to each trajectory points of T are selected to compose the route of T, and minimal distance between sample points and road vertices are used as the matching metric.

Topology-Based Metrics. The topology information employed by this kind of metrics include connectivity, adjacency, bounding relationship, etc., between curves or polygons. For example, in the global map matching algorithm [5] the Fréchet distance [2] is used to measure the matching quality between a trajectory and a route. The topology-based metrics consider not only the distance between sample points and the potential matching route, but also the topology connectivity inside the route itself, therefore, they are better metrics for noisy low-sampling-rate trajectories than the geometry-based metrics.

Probability-Based Metrics. These metrics use the probability that a trajectory may actually go through a certain route to measure the matching quality. Due to measurement precision, the actual location of each GPS sample point is restricted to an ellipse confidence area, thus the probability that a sample point goes through certain route can be calculated according to the relationship between the point and the part of route within the confidence area [3,17,19,21].

[1] In this paper we use 'trajectory' to represent any raw GPS trajecotry for simplicity.

Fig. 1. Matching route example (Color figure online)

In [9], the Hidden Markov Model is used to measure the possibility that a trajectory may actually go through a certain route, and the route with the highest possibility is selected as the matching route.

Accuracy Metrics. These metrics use statistical knowledge to measure how accurate a route matches a trajectory. For example, given a route, the accuracy metric [14] uses the ratio of correctly matched road segments over total number of segments in the trajectory, to evaluate the accuracy and thus the quality of the matching route. The problem with such metrics is that ground truth has to be provided when evaluating matching quality, while the actual routes are unknown for most raw trajectories, thus restricting the usability of accuracy metrics.

Problem Formulation. Now, we can formalize the problem of map matching.

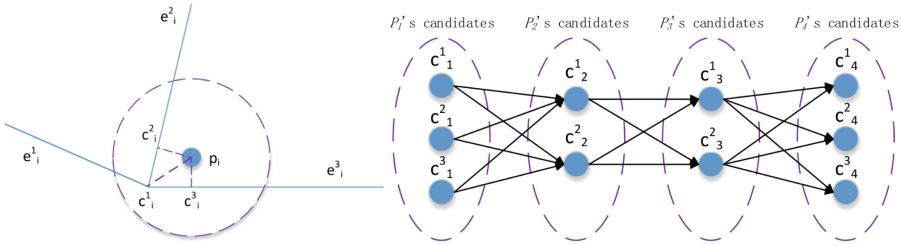
Definition 4 (Map Matching). *Given a raw trajectory T, a road network G and a matching metric M, the map matching of T onto G is to find a route R_{best} in G, so that M is maximized:*

$$R_{best} = arg\ max_{R_k} M(G, T, R_k) \tag{1}$$

In the matching process, more than one possible routes can be generated, they are referred to as **candidate routes**. For example, Fig. 1 visualizes a raw trajectory T (the blue curve) and two candidate routes R_1 (the red dot-curve) and R_2 (the green curve) using Google Map. Suppose the vehicle actually went through route R_1, then the accuracy of R_1 is higher than R_2 because R_1 has more correctly matched road segments for T than R_2 does.

3 ST Methods for the Map-Matching Problem

For the map matching problem, a classic method called `ST-Matching` [14] considers geometric and topological structures of the road network, as well as the temporal/speed constraints of the trajectories. Based on spatial-temporal analysis, a candidate route is concluded, from which the best matching score matrix is identified. The `ST-Matching` algorithm could be divide into two major parts: *Candidate Filtering*, and *Spatial and Temporal Analysis*. We review how to filter the candidate points in Sect. 3.1 and discuss its spatial and temporal analysis in Sect. 3.2. After that we focus on two improved versions of the `ST-Matching` algorithm in Sect. 3.3.



(a) Candidate Road Segments/Points (b) Candidate Graph

Fig. 2. Candidate road segments/points & candidate graph

3.1 Candidate Filtering

Given trajectory $T = \{p_1, p_2, \cdots, p_{|T|}\}$, `ST-Matching` first obtains a set of candidate road segments within radius r of each unmatched trajectory point p_i $(1 \leq i \leq |T|)$. As illustrated in Fig. 2(a), within the circle of radius r, c_i^1, c_i^2, and c_i^3 are candidate points for trajectory point p_i; and e_i^1, e_i^2, and e_i^3 are candidate road edges for p_i.

Once the candidate point sets are proposed for all points in trajectory T, the problem becomes how to choose one candidate from each set in order to make $c_1^{j_1} \rightarrow c_2^{j_2} \rightarrow \cdots \rightarrow c_n^{j_n}$ best matches T.

3.2 Spatial and Temporal Analysis

The spatial analysis function measures the similarity of the unmatched part between two trajectory points and the link with the shortest path between the two corresponding candidate points. First, a candidate graph is constructed (Fig. 2(b)). The distribution of the GPS measurement error is assumed to take the Gaussian distribution $N(\mu, \sigma^2)$. For each candidate point in the candidate point set, its observation probability to p_i is:

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

where x_i^j is the Euclidean distance from candidate c_i^j to unmatched point p_i.

From candidate point c_{i-1}^t to c_i^s, the spatial analysis is defined as:

$$F_s(c_{i-1}^t \rightarrow c_i^s) = N(c_i^s) * V(c_{i-1}^t \rightarrow c_i^s), 2 \leq i \leq n. \tag{3}$$

where $V(c_{i-1}^t \rightarrow c_i^s)$ is the transition probability:

$$V(c_{i-1}^t \rightarrow c_i^s) = \frac{d(i-1, i)}{w(c_{i-1}^t, c_i^s)}. \tag{4}$$

where $d(i-1, i)$ is the Euclidian distance from p_{i-1} to p_i, and $w(c_{i-1}^t, c_i^s)$ is the length of the shortest path between c_{i-1}^t to c_i^s.

The temporal analysis of `ST-Matching` considers the speed information:

$$F_t(c_{i-1}^t \to c_i^s) = \frac{\sum_{u=1}^{k}(e'_u v * \overline{v}_{c_{i-1}^t \to c_i^s})}{\sqrt{\sum_{u=1}^{k}(e'_u v)^2} * \sqrt{\sum_{u=1}^{k} \overline{v}^2_{c_{i-1}^t \to c_i^s}}}. \tag{5}$$

where point set e' is the shortest path connecting c_{i-1}^t and c_i^s.

Combining the spatial and temporal analysis, the `ST-Matching` function to score the route between two candidate points can be achieved:

$$F(c_{i-1}^t \to c_i^s) = F_s(c_{i-1}^t \to c_i^s) * F_t(c_{i-1}^t \to c_i^s). \tag{6}$$

Therefore, for trajectory T, the route with the best score from one candidate point of the starting point of T to one candidate point of the end point of T is identified as the matching route for T. However, `ST-Matching` is based on an assumption that a driver always chooses the shortest route, which may not be consistent with the real world.

3.3 Improvements of ST-Matching

The `GridST` [7] tries to improve the first part of `ST-Matching`, and the `IVMM` [27] algorithm aims to improve the second part of `ST-Matching`.

(1) **The ST-Matching Based on the Locality of Road Networks**

The `GridST` algorithm ameliorates the candidate filtering of `ST-Matching`. The error circle radius and the maximum number of selected candidate points are dynamically adjusted according to the locality of the road network. Subsequently, the number of shortest path computations is reduced, shortening the overall running time. In order to generate the locality of road network, `GridST` splits the road network graph into grids. Before the map-matching process, all grids' information are calculated and organized to ensure the running time of this algorithm. If a grid has a higher density of road segments, the candidate filtering will have higher possibilities to select enough number of candidate point in a smaller error circle, vice versa. Therefore, `GridST` reduces the number of shortest path computations and the overall running time of map-matching process.

(2) **The Interactive Voting-Based Map-Matching algorithm (`IVMM`)**

This algorithm utilizes a voting process among all sampling points to reflect their interactive influence after spatial and temporal analysis of candidate points. For each sampling point, `IVMM` will repeatedly select an optimal route which passes through it. Every candidate point will get one vote when the optimal path includes this candidate point. Then the global optimal route will be chosen according to the vote result.

Given the spatial and temporal result of `ST-Matching`: $F(c_{i-1}^t \to c_i^s) = F_s(c_{i-1}^t \to c_i^s) * F_t(c_{i-1}^t \to c_i^s)$, a Static Score Matrix $M = diag(M^1, M^2, \ldots, M^n)$ is built, $M^i = (F(c_{i-1}^t \to c_i^s))_{a_{i-1} \times a_i}$. Each item in this matrix represents the possibility of a candidate point to be a correct match point. However, this possibility only considers the information of two adjacent points.

To model the weighted influence of candidate points, a $(n-1)$-dimension Matrix $\mathbf{W_i}$ is created for each sampling point p_i. And these matrix only have items in diagonal line: $w_i^j = f(dist(p_i, p_j)), (j = 1, 2, \ldots, n)$, where j is the sequence number of diagonal line in $\mathbf{W_i}$. And $dist()$ is the Euclidean distance. $f(x) = \mathrm{e}^{-\frac{x^2}{\beta^2}}$, where β is a parameter related to the road network. Then M is recalculated with the weighted influence in $\mathbf{W_i}$, and every item in M is multiplied by their weighted score. Matrix M becomes weighted score matrix Φ.

Next, voting based on the interaction of candidate points starts. For each candidate point c_i^k, `IVMM` attempts to find an optimal route using the weighted score matrix Φ. If a candidate point c_i^k is included in an optimal route, this candidate point c_i^k gains one vote. Then the candidate point with the largest number of votes for each sampling point p_i is identified. Finally, the best route which passes through every corresponding candidate point is selected as the matching route.

4 Other Algorithms

4.1 The Fuzzylogic Algorithm

The `FuzzyLogic` algorithm [12] is different from afore-mentioned algorithms: it exploits fuzzy logic to construct the degree of similarity between a matched route and a raw trajectory. The matching route is selected based on its possibility to achieve the best similarity.

(1) **Candidate Filtering**

In `FuzzyLogic`, it first plots the candidate area of an ellipse around the current trajectory point whose radius is the GPS positioning error. `FuzzyLogic` checks all roads in the candidate area and connects them with the already-matched road. If the candidate area can not satisfy the conditions, then `FuzzyLogic` directly gives up this matching. Otherwise, each sampling point has a candidate set including all candidate roads within the candidate area.

(2) **Fuzzy Analysis**

`FuzzyLogic` uses the fuzzy comprehensive judgement and constructs the set of fuzzy factors $F = \{F_x, F_y, F_z\}$, representing three aspects: car running direction, the distance between candidate road and sampling point and comparability of unmatched trajectory with candidate roads.

(2.1) The Membership Factor of Direction

Let $\theta'_{(j,k)}$ denote the direction angle between the j^{th} sampling point and the k^{th} candidate point for each sampling point, θ_j denote the direction angle of each sampling point. Their difference $\Delta\theta_j$ denotes direction angle factor set F_x, and it represents the angle between the vehicle's running direction and the candidate road direction (Fig. 3). Five classes of degree are identified: "very small", "small", "medium", "big"and "very big" for fuzzy reasoning.

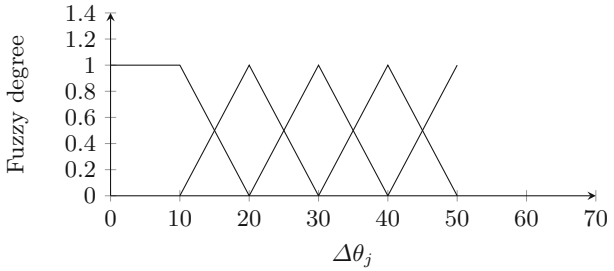


Fig. 3. Membership function of direction

(2.2) The Membership Factor of Distance

Let $\Delta d_{(j,k)}$ represent the projection distance from the j^{th} sampling point to the k^{th} candidate road. $\Delta d_{(j,k)}$ is regarded as the distance factor set: F_y. This distance can be classified to five fuzzy degrees: "very small", "small", "medium", "big"and "very big", as shown in Fig. 4.

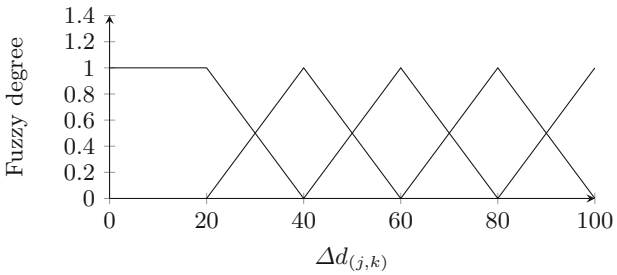


Fig. 4. Membership function of distance

(2.3) The Membership Factor of Comparability of Positioning Trajectory

The handling for this factor is similar to the previous two factors. Candidate roads are resembled to an assumption point with the computing rules used by foregone sampling point and use the distance between the assumption point and sampling point as the third factor in this fuzzy model.

With all three fuzzy factors, `FuzzyLogic` performs fuzzy transform. The fuzzy vector Q would be the result set aimed at $F = \{F_x, F_y, F_z\}$, where each element within Q denotes the possibility degree of candidate road for each sampling point. The candidate road with the largest matching degree is the matched road [12].

4.2 The Statistic Algorithm

The `Statistic` algorithm [24] is based on multiple hypothesis technique. For one unmatched trajectory, `Statistic` first selects all nodes within the radius r around the sampling point. After that, it adds all roads in the network which connect to at least one of these selected nodes to the candidate road set. For each

road candidate, the sampling point is assigned to the road, and the matching score is calculated and stored in the list of current road candidates. The matching score is calculated by combining the heading of sampling point compared to the heading of road, the current speed of sampling point and the free-flow speed on the road, which is shown below:

$$Score_{road} = d(p_i, l_j) + ((v(p_i) - v_{ff}(l_j))^2 \theta_{ij}) \quad (7)$$

where p_i represents the i^{th} sampling point and l_j is the j^{th} candidate road for p_i, $v(p_i)$ stands for current speed of p_i and $v_{ff}(l_j)$ represents the free-flow speed of l_j road. The parameter θ equals 1 if $v(p_i) > v_{ff}(l_j)$, and 0 otherwise.

When all candidates road have been processed for the current sampling point, this algorithm selects the road which got the highest score to be the matched road. But if the number of candidate roads is not enough, it repeatedly increase radius r until there are enough candidates for each sampling point.

5 Experimental Study

We experimentally compare existing map matching algorithms. Our experimental goal is to evaluate the matching quality, running time, and impacts of parameters to the performance of different algorithms. The matching quality is measured by accuracy-based metrics to reveal how close the matching results are to the actual routes. The running time is the total time to match a given set of raw trajectories. The parameters in question are (1) number of candidate points, and (2) sampling rate, as these two are crucial to the algorithms' performance.

5.1 Experimental Settings

Algorithms. We compare the following algorithms: `ST-Matching` [14], `IVMM` [27], `GridST` [7], `FuzzyLogic` [12] and `Statistic` [24].

Data Sets

Road Network. We use the road network of Beijing which has 1,285,215 vertices and 2,690,296 edges.

Real Trajectory Data. We use two real datasets: `Taxi` (www.datatang.com/data/45888) and `UCar` (www.10101111.com/). `Taxi` contains trajectories generated by more than 8,000 public taxicabs in Beijing of one month; `UCar` contains trajectories of nearly 2,000 cars registered in the platform of ShenZhou Zhuanche(like Uber) within one week in Beijing.

Synthetic Trajectory Data. We implement a simulator to generate synthetic data as follows. First, a starting point v_s and a destination point v_d are randomly selected from the vertex set of the road network. Then, a connected path from v_s to v_d is generated (this path does not have to be the shortest path between v_s and v_d). Next, assuming the vehicle is moving at some fixed speed (e.g.,

Table 1. Trajectory data sets

Data set	Num. of traj.	Avg point num.	Max point num.	Min point num.	Avg sample rate
`Taxi`	200,000	27	50	5	x
`UCar`	120,000	16	20	3	x
`Syn`	10,000	388.6	1333	10	20 s

60 km/h), the simulator selects a set of sample points along the path for a given sampling rate (e.g., 1 min), and randomly deviates the sampling point (which is originally on the road) to a location within an error range of latitude and longitude. Our default settings are: the vehicular speed is 45 km/h, the sampling rate is 20 s, and the latitude and longitude deviations are both $\pm 0.0002^{o}$.

Table 1 shows the statistics of the three datasets.

Ground Truth. As stated before, our synthetic trajectories are generated by first selecting a route from a starting location to a destination, and then adding some noises to simulate real trajectories. Therefore, the correct routes are known and can be used as ground truth. In addition to our synthetic data, we also provide a set of 30 real trajectories manually labeled as ground truth, denoted as `HL-30`. These trajectories are selected from `Taxi` and `UCar` datasets and manually labelled with the true routes. Trajectory lengths varies from 5.090 km to 23.933 km, averaging at 10.568 km; number of points set to 30.

Settings. All the algorithms are implemented by C++, compiled by Visual C++. All the experiments are conducted on a Windows Server 2012 with an Intel Xeon E52682 CPU (two cores, 2.5 GHz) and 4 GB memory.

5.2 Evaluating Accuracy

Accuracy Metrics. Given a trajectory T whose ground truth is denoted as $\overline{T}$, we measure the matching quality of a route R to T as follows:

$$N_{Acc} = \frac{\text{num. of road segments in } R \bigcap \overline{T}}{\text{num. of all road segments in } R} \tag{8}$$

$$L_{Acc} = \frac{\sum \text{length of road segments in } R \bigcap \overline{T}}{\text{length of } R} \tag{9}$$

Parameter Selection and Default Values. For `ST-Matching` algorithm, we set $k = 5$, r=100 m, $\mu = 0$, and δ=20 m. For `IVMM` algorithm, we set $k = 5$, r =100 m, $\mu = 0$, and δ=20 m. For `GridST` algorithm, we set $\mu = 0$, and δ=20 m. These settings are used as default values through out our experiments.

Figures 5 and 6 show the results on `HL-30` and `Syn` datasets respectively. We have the following observations.

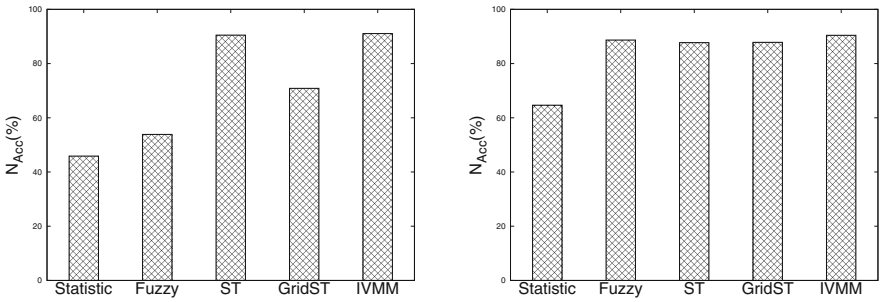
First, for `HL-30` dataset (sampling rate $\geq$ 1 min), the `IVMM` algorithm and `ST-Matching` algorithm achieve top N_{Acc} and L_{Acc} accuracy. For `IVMM`, this is because the voting step strengthens scores of candidate points which have

higher possibility to be on the real route. For `ST-Matching`, the spatial and temporal analysis can return high quality candidates especially for low-sampling-rate trajectories, explaining the top accuracy achieved by `ST-Matching`. `GridST` achieves the third accuracy, demonstrating that the policy to divide the road network into grids and adjust candidate numbers dynamically can not beat the original `ST-Matching` algorithm in case of low-sampling-rate. For `FuzzyLogic`, it chooses the best route based on the similarity between the trajectory and the route, when the sampling rate decreases, the similarity between the trajectory and the route is discounted, resulting in the fourth accuracy among all the algorithms tested. The `Statistic` algorithm is inferior to other algorithms on accuracy because its scoring model for candidate route sometimes could not select the "right"candidate.

Second, for synthetic dataset `Syn` (sampling rate 20 s), the `IVMM` algorithm and `FuzzyLogic` algorithm achieve top N_{Acc} and L_{Acc} accuracy. For `IVMM`, the voting step provides stable functionality despite the sampling rate as just analyzed. For `FuzzyLogic`, this is because that it chooses the best route based on the similarity between the trajectory and the route, the geometry and topology factors can filter high quality candidate in case of high-sampling-rate trajectories. The `ST-Matching` and `GridST` algorithms can also achieve 80%+ N_{Acc} and L_{Acc} accuracy because the spatial and temporal analysis can return high quality candidates. The `Statistic` algorithm is inferior to other algorithms on accuracy because its scoring model for candidate route sometimes could not select the "right" candidate, despite the sampling rate.

Third, for a given dataset, all five algorithms have similar ranking for both accuracy metrics. Although N_{Acc} focuses on the number of correctly matched road segments, and L_{Acc} focuses on the length of correctly matched road segments, on average, the number of road segments in a trajectory is proportional to the length of a trajectory, because lengths of road segments vary within a limited range (e.g., 20 m–50 m).

Fourth, for the two datasets, the `ST-Matching` and `IVMM` algorithms report similar accuracy, demonstrating stable matching quality on different sampling rates. The `FuzzyLogic`, `GridST` and `Statistic` algorithms work better for high-sampling-rate trajectories.



(a) `HL-30` (b) `Syn`

Fig. 5. Evaluating accuracy: N_{Acc}

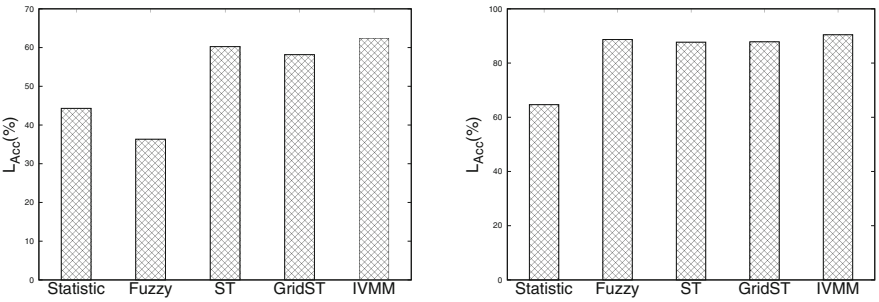


Fig. 6. Evaluating accuracy: L_{Acc}

5.3 Evaluating Running Time

We compare the running time of all the algorithms by varying the number of trajectories tested from the `Taxi`, `UCar` and `Syn` datasets. Figure 7(a), (b) and (c) show the respective results. We have the following observations.

First, the runtime efficiencies of all five algorithms present similar trends on all of the three datasets tested, demonstrating stability despite the underlying trajectory data.

Second, the `Statistic` and `FuzzyLogic` algorithms have top runtime efficiency, which is linear to the dataset size. This is because the logic of these two algorithms does not involve time-consuming matrix calculation as in the `ST-Matching` and `GridST` algorithms.

Third, the `ST-Matching` and `GridST` algorithms also present linear runtime efficiency in terms of dataset size. The reason that these two algorithms is less efficient than the `Statistic` and `FuzzyLogic` algorithms, as just stated, is because their logic involves matrix calculation which is time-consuming.

Fourth, the `IVMM` algorithm does not scale well as the data size increases. As indicated in [27], this algorithm has to be parallel-programming in order to achieve satisfying efficiency, which is non-trivial work.

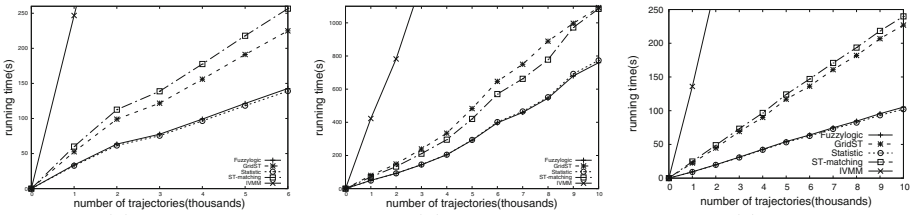


Fig. 7. Evaluating running time

5.4 Evaluating Impact on Accuracy and Running Time by Candidate Point Numbers

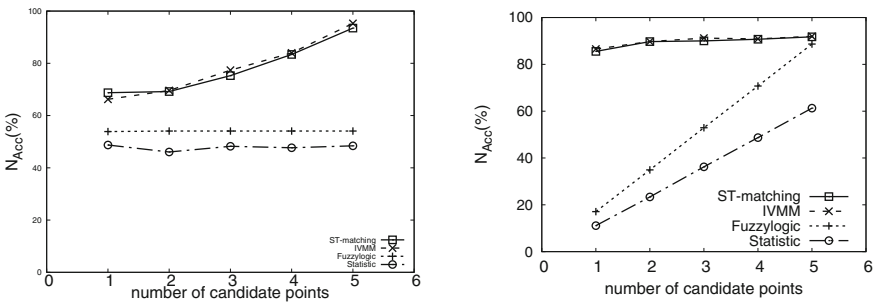
Both the `ST-Matching` and `IVMM` algorithms have an explicit parameter in terms of the maximum candidate points for each trajectory point. For the `FuzzyLogic` and `Statistic` algorithms, they also select a number of possible matching road points in early state of each algorithm, therefore we treat this parameter as maximum candidate points in this section as well. We evaluate the impact on accuracy by varying this parameter from 1 to 5. Figures 8, 9 and 10 show the corresponding results of N_{Acc}, L_{Acc} and runtime efficiency on `HL-30` dataset and a 1000-trajectory `Syn` dataset. We have the following observations.

First, on each dataset, the algorithms compared exhibit similar matching quality variation and runtime efficiency trends as the number of candidate points increases, demonstrating stability despite the underlying trajectory data.

Second, for the `ST-Matching` and `IVMM` algorithm, the accuracy improvement over the number of candidate points is more obvious on the `HL-30` dataset than on the `Syn` dataset. This is because `HL-30` contains low-sampling trajectories, as the number of candidate points increases, the possibility that the "right" road segments are taken into consideration is increased, therefore, increasing accuracy.

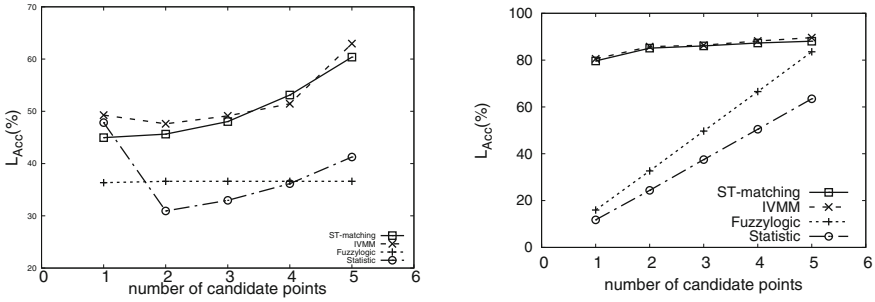
Third, for the `FuzzyLogic` and `Statistic` algorithms, their matching quality is not comparable to the `ST-Matching` and `IVMM` algorithm. But since the `Syn` dataset contains high-sampling-rate trajectories, when the number of candidate points is big enough (e.g., 4 and above), the matching accuracy can be improved. Note that for `FuzzyLogic`, it is almost as good as `ST-Matching` and `IVMM` when the number of candidate points is 5.

Fourth, the `Statistic` algorithm has the best runtime efficiency, the `FuzzyLogic` algorithm has comparable efficiency, and the `ST-Matching` consumes more time as the number of candidate points increases. This indicates that the `ST-Matching` is not suitable for more than 5 candidate points. Besides, the runtime efficiency of `IVMM` is not plotted in Fig. 10 since it explodes as the number of candidate points increases, proving again that it can not scale well unless parallel programming is used.



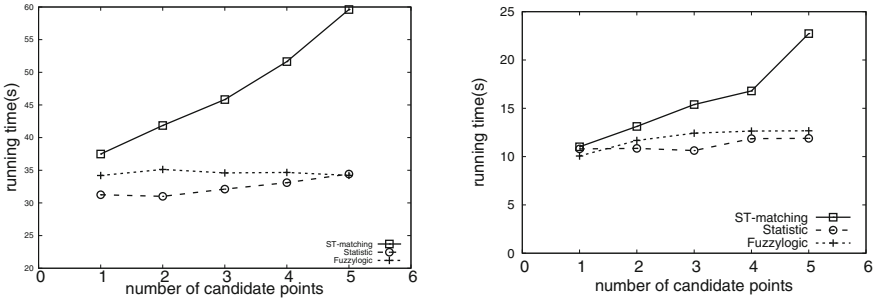
(a) `HL-30` (b) `Syn` (1000 trajectories)

Fig. 8. Evaluating impact by max. candidate point num. on accuracy: N_{Acc}



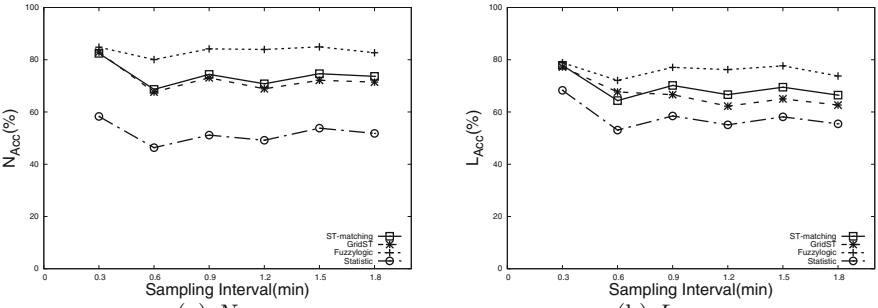
(a) HL-30 (b) Syn (1000 trajectories)

Fig. 9. Evaluating impact by max. candidate point num. on accuracy: L_{Acc}



(a) HL-30 (b) Syn (1000 trajectories)

Fig. 10. Evaluating impact by max. candidate point num. on running time



(a) N_{Acc} (b) L_{Acc}

Fig. 11. Evaluating impact by sampling rate on accuracy: N_{Acc} and L_{Acc} (Syn: 1000 trajectories for each sampling interval)

5.5 Evaluating Impact on Accuracy by Sampling Rates

In this section, we compare the matching quality in terms of accuracy with respect to the sampling rate on our synthetic dataset Syn. Figure 11(a) and (b) show the corresponding results on N_{Acc} and L_{Acc}. The result of IVMM is not

reported because the running time is more than one order of magnitude to that of the other algorithms. We have the following observations.

First, the four tested algorithms exhibit similar matching quality as measured by both N_{Acc} and L_{Acc}. Second, the `FuzzyLogic` algorithm is the most insensitive to the sampling rate variation, and has the best matching quality. Third, the `ST-Matching` and `GridST` algorithms have similar matching quality, as their basic logic consent. Last, the `Statistic` has the worst matching quality, as demonstrated in above experiments.

6 Conclusion

This paper provides an experimental survey on existing map matching algorithms, including, `ST-Matching`, `GridST`, `IVMM`, `FuzzyLogic`, and `Statistic`, and compares them through extensive experiments on both real-world and synthetic datasets with different characteristics. We provide the following experimental findings.

(1) For better matching quality (measured by N_{Acc} and L_{Acc} accuracy), the `ST-Matching` and `IVMM` algorithms are the best choice on low-sampling-rate trajectory datasets as they outperform other algorithms; and the `FuzzyLogic` algorithm is also a good choice on high-sampling-rate trajectory datasets.
(2) The `FuzzyLogic` and `Statistic` algorithms always achieve better efficiency on both high-sampling-rate and low-sampling-rate trajectory datasets.
(3) Generally speaking, as the sampling rate increases, the matching quality of all tested algorithms increases.
(4) Among all tested algorithms, the `Statistic` algorithm reports the worst matching quality.

Acknowledgement. This research is supported in part by the Key Grant Project on Humanities and Social Sciences of MOE of China (16JJD860008), the 2018 RUC Special Fund for First-Class Universities (Majors) of Central Universites, and RUC Start-up Fund (2018030119).

References

1. Aly, H., Youssef, M.: SemMatch: road semantics-based accurate map matching for challenging positioning data. In: Proceedings of the 23rd SIGSPATIAL International Conference on Advances in Geographic Information Systems, Bellevue, WA, USA, 3–6 November 2015, pp. 5:1–5:10 (2015)
2. Alt, H., Godau, M.: Computing the fréchet distance between two polygonal curves. Int. J. Comput. Geometry Appl. **5**, 75–91 (1995)
3. Bierlaire, M., Chen, J., Newman, J.: A probabilistic map matching method for smartphone GPS data. Transp. Res. Part C-emerg. Technol. **26**, 78–98 (2013)
4. Blazquez, C., Vonderohe, A.P.: Simple map-matching algorithm applied to intelligent winter maintenance vehicle data. Transp. Res. Rec. **1935**, 68–76 (2005)

5. Brakatsoulas, S., Pfoser, D., Salas, R., Wenk, C.: On map-matching vehicle tracking data. In: Proceedings of the 31st International Conference on Very Large Data Bases, Trondheim, Norway, 30 August–2 September 2005, pp. 853–864 (2005)
6. Brakatsoulas, S., Pfoser, D., Tryfona, N.: Practical data management techniques for vehicle tracking data. In: Proceedings of the 21st International Conference on Data Engineering, ICDE 2005, 5–8 April 2005, Tokyo, Japan, pp. 324–325 (2005)
7. Chandio, A.A., Tziritas, N., Zhang, F., Xu, C.-Z.: An approach for map-matching strategy of GPS-trajectories based on the locality of road networks. In: Hsu, C.-H., Xia, F., Liu, X., Wang, S. (eds.) IOV 2015. LNCS, vol. 9502, pp. 234–246. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-27293-1_21
8. Chawathe, S.S.: Segment-based map matching. In: Intelligent Vehicles Symposium, pp. 1190–1197 (2007)
9. Goh, C.Y., Dauwels, J., Mitrovic, N., Asif, M.T., Oran, A., Jaillet, P.: Online map-matching based on hidden markov model for real-time traffic sensing applications. In: 15th International IEEE Conference on Intelligent Transportation Systems, ITSC 2012, Anchorage, AK, USA, 16–19 September 2012, pp. 776–781 (2012)
10. Gonzalez, H., Han, J., Li, X., Myslinska, M., Sondag, J.P.: Adaptive fastest path computation on a road network: a traffic mining approach. In: Proceedings of the 33rd International Conference on Very Large Data Bases, University of Vienna, Austria, 23–27 September 2007, pp. 794–805 (2007)
11. Greenfeld, J.: Matching GPS observations to locations on a digital map. In: Proceedings of TRB (2002)
12. Haibin, S., Jiansheng, T., Chaozhen, H.: A integrated map matching algorithm based on fuzzy theory for vehicle navigation system, vol. 1, pp. 916–919 (2006)
13. Hu, H., Liu, Y., Li, G., Feng, J., Tan, K.: A location-aware publish/subscribe framework for parameterized spatio-textual subscriptions. In: 31st IEEE International Conference on Data Engineering, ICDE 2015, Seoul, South Korea, 13–17 April 2015, pp. 711–722 (2015)
14. Lou, Y., Zhang, C., Zheng, Y., Xie, X., Wang, W., Huang, Y.: Map-matching for low-sampling-rate GPS trajectories. In: Proceedings of 17th ACM SIGSPATIAL International Symposium on Advances in Geographic Information Systems, ACM-GIS 2009, 4–6 November 2009, Seattle, Washington, USA, pp. 352–361 (2009)
15. Luo, W., Tan, H., Chen, L., Ni, L.M.: Finding time period-based most frequent path in big trajectory data. In: Proceedings of the ACM SIGMOD International Conference on Management of Data, SIGMOD 2013, New York, NY, USA, 22–27 June 2013, pp. 713–724 (2013)
16. Newson, P., Krumm, J.: Hidden Markov Map matching through noise and sparseness. In: Proceedings of 17th ACM SIGSPATIAL International Symposium on Advances in Geographic Information Systems, ACM-GIS 2009, 4–6 November 2009, Seattle, Washington, USA, pp. 336–343 (2009)
17. Ochieng, W.Y., Quddus, M.A., Noland, R.B.: Map-matching in complex urban road networks. Revista Brasileira de Cartografia **2**(55), 1–14 (2003)
18. Pereira, F.C., Costa, H., Pereira, N.M.: An off-line map-matching algorithm for incomplete map databases. Eur. Transp. Res. Rev. **1**(3), 107–124 (2009)
19. Pink, O., Hummel, B.: A statistical approach to map matching using road network geometry, topology and vehicular motion constraints, pp. 862–867 (2008)
20. Qu, M., Zhu, H., Liu, J., Liu, G., Xiong, H.: A cost-effective recommender system for taxi drivers. In: The 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD 2014, New York, NY, USA, 24–27 August 2014, pp. 45–54 (2014)

21. Quddus, M.A., Noland, R.B., Ochieng, W.Y.: A high accuracy fuzzy logic based map matching algorithm for road transport. J. Intell. Transp. Syst. **10**(3), 103–115 (2006)
22. Quddus, M.A., Ochieng, W.Y., Zhao, L., Noland, R.B.: A general map matching algorithm for transport telematics applications. GPS Solut. **7**(3), 157–167 (2003)
23. Rohani, M., Gingras, D., Gruyer, D.: A novel approach for improved vehicular positioning using cooperative map matching and dynamic base station DGPS concept. IEEE Trans. Intell. Transp. Syst. **17**(1), 230–239 (2016)
24. Schuessler, N., Axhausen, K., Zurich, E.: Map-matching of GPS traces on high-resolution navigation networks using the multiple hypothesis technique (MHT), vol. 01 (2009)
25. Wenk, C., Salas, R., Pfoser, D.: Addressing the need for map-matching speed: localizing globalb curve-matching algorithms. In: Proceedings 18th International Conference on Scientific and Statistical Database Management, SSDBM 2006, 3–5 July 2006, Vienna, Austria, pp. 379–388 (2006)
26. Yin, H., Wolfson, O.: A weight-based map matching method in moving objects databases. In: Proceedings of the 16th International Conference on Scientific and Statistical Database Management (SSDBM 2004), 21–23 June 2004, Santorini Island, Greece, pp. 437–438 (2004)
27. Yuan, J., Zheng, Y., Zhang, C., Xie, X., Sun, G.: An interactive-voting based map matching algorithm. In: Eleventh International Conference on Mobile Data Management, MDM 2010, Kanas City, Missouri, USA, 23–26 May 2010, pp. 43–52 (2010)
28. Yuan, N.J., Zheng, Y., Xie, X., Wang, Y., Zheng, K., Xiong, H.: Discovering urban functional zones using latent activity trajectories. IEEE Trans. Knowl. Data Eng. **27**(3), 712–725 (2015)
29. Zheng, K., Zheng, Y., Xie, X., Zhou, X.: Reducing uncertainty of low-sampling-rate trajectories. In: IEEE 28th International Conference on Data Engineering (ICDE 2012), Washington, DC, USA (Arlington, Virginia), 1–5 April 2012, pp. 1144–1155 (2012)