Physica A 534 (2019) 122318

Contents lists available at ScienceDirect

Physica A

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

An enhanced weight-based real-time map matching algorithm for complex urban networks

Mujun He [a], Linjiang Zheng [a,b,*], Wei Cao [c], Jing Huang [a], Xu Liu [a], Weining Liu [a,b]

[a] *College of Computer Science, Chongqing University, Chongqing, China*
[b] *Key Laboratory of Dependable Service Computing in Cyber Physical Society of Ministry of Education, Chongqing University, Chongqing, China*
[c] *Troops 66139, People's Liberation Army of China, Beijing, China*

HIGHLIGHTS

- An enhanced weight-based real-time map matching algorithm is proposed.
- Dynamical weights of each GPS point are taken in consideration.
- A new method of calculating confidence level on the selected segment is proposed with considering network density and complexity.

ARTICLE INFO

Article history:
Received 26 March 2019
Received in revised form 17 May 2019
Available online 8 August 2019

Keywords:
GPS
Map matching
Dynamic weighted
Vehicle navigation

ABSTRACT

A map-matching algorithm is used to map the inaccurate raw coordinate data to the digital road network. It is an indispensable part of Location Based Service applications and Intelligent Transportation Systems, such as navigation systems. Accuracy and performance (running speed) are usually traded off in traditional algorithms. An enhanced weight-based real-time map matching algorithm only employing GPS data is proposed to guarantee both. The algorithm has two steps: initialization and tracking match, each step is mainly composed of three parts. Firstly, segments near the GPS point are selected as candidate segments. Secondly, four criteria (distance, heading difference, direction difference and segment connectivity) are used to identify the best segment among candidates. Considering the reliability of each criterion, four dynamic weight coefficients are introduced. Finally, before assigning a candidate segment to each GPS point, a confidence level is calculated and considered based on the density and complexity of roads around the point. We evaluate the algorithm with field data collected from the city of Chongqing, China. The results demonstrate that it can identify correct segment from complicated and dense urban road networks, with an average matching accuracy of 97.31% and a latency of 3.20ms per location estimate.

© 2019 Published by Elsevier B.V.

1. Introduction

The number of handheld devices, smart phones and other mobile devices has been increasing rapidly in the past few years. And these devices, equipped with the global positioning system (GPS), become the most widely used geo-positioning sensor. The data generated by such sensors are called GPS data, include basic vehicle telemetry, such as speed, driving direction, and most importantly, the vehicle position [1]. Numerous GPS data, collected from a given spatial area such as

* Correspondence to: College of Computer Science, No.174 Shazhengjie, Shapingba, Chongqing, 400044, China.
E-mail address: zlj_cqu@cqu.edu.cn (L. Zheng).

https://doi.org/10.1016/j.physa.2019.122318
0378-4371/© 2019 Published by Elsevier B.V.

a city, comprise hundreds of thousands of vehicles travel trajectories, containing tremendous traffic information in space and time, which are the fundamental data of Location Based Service (LBS) and Intelligent Transportation System (ITS) applications, such as vehicle navigation, traffic event inspection, travel time prediction, etc.

However, the raw coordinate data obtained from GPS devices cannot be used directly. For most cases, it deviates from the corresponding location of the real road, due to errors in positioning sensors [2] and dense and complex urban road networks [3,4]. Therefore, the first step of using GPS data is to project the raw coordinate data to the digital map correctly, and the process is known as map matching.

Map matching can be performed by either real-time mode [5–8] with low-sampling-rate positioning data [9,10], or post-processing mode [11,12] with high frequency positioning data [3,13]. Low frequency positioning data including cellular mobile data [14,15] or low-sampled GPS [16] can be used. Typically, high frequency positioning data (i.e. 1 Hz or 1 s sampling interval) with many data items, like position, timestamp, speed, heading and positioning accuracy, are accessed and applied to match [17,18]. As presented in [18], real-time algorithms must guarantee not only the accuracy of estimating a user's position on a road segment, but also the efficiency of matching process. In real-time mode, errors in positioning sensors only can be revised by existing GPS points, identified road segments and road topological information.

Accuracy and performance (running speed) are important in real-time map matching, whereas they are usually traded off in traditional algorithms. Some approaches have high matching efficiency but low accuracy, and others present good accuracy with complex calculation and low matching efficiency (discussed in the second section). While weight-based algorithms balance the accuracy and the efficiency, the correct matching road cannot be determined when dealing with intersections and parallel roads [19]. Nevertheless, recent studies [3] have indicated that weight-based map-matching algorithms with appropriate weights have great potential.

An enhanced weight-based real-time map matching algorithm, referred as DW-RMM, is proposed in this article to achieve high success rate and high efficiency using high-frequency GPS positioning data. Compared with previous map matching algorithms, the algorithm has following features:

1. Making full use of spatial–temporal information to build the weight-based model, including speed, position and heading of each GPS point, direction of consecutive GPS points. Topological constraints such as connectivity and turn-restrictions are also considered. Matching of parallel segments and intersections has been improved significantly.
2. Taking a novel approach to calculate the weights of each GPS point for best segment identification. The weights can be changed dynamically, which raises the transportability of the algorithm in different circumstances.
3. Selecting the optimal match for each GPS point based on the confidence level of the assigned segment. Note that segment with the highest comprehensive score is chosen only when the difference between the best segment and the second best segment is greater than the threshold, which reduces the risk of over-fitting.

The rest of this article is organized as follows. The next section provides literature review in existing real-time map matching algorithms. The proposed algorithm is presented and explained in the third section. The fourth section discusses results of the algorithm and the performance is compared with other map-matching algorithms. Finally, the conclusion is drawn in the fifth section by summarizing its achievements, drawback, and future research directions.

2. Related work

The existing real-time map-matching algorithms are generally divided into three steps [2,20,21]: (a) identification of correct segment in the initialization stage, (b) matching at intersection when the vehicle approaches an intersection and (c) matching on the same segment based on the previous points. Hashemi and Karimi [19] categorized existing real-time map-matching algorithms into three groups: simple, advanced, and weigh-based.

Simple algorithms only consider the geometrical relation between vehicle trajectory and the shape of road segment. So, they are also called geometrical map-matching algorithms, which include three different types: point-to-point matching, point-to-curve matching and curve-to-curve matching. Simple algorithms can be applied to any frequency positioning data as the required input data contain position fixes (x- and y- coordinates) and a base road network map [17]. However, they have high matching efficiency but low accuracy for their excessive simplicity, and perform poorly at intersections and parallel roads [22,23]. In general, the models of simple algorithms are too simple to provide accurate solutions in complicated urban road networks.

Advanced map-matching algorithms turn map matching problem into complicated mathematic problem, and apply complex mathematical models to improve the accuracy. Fuzzy logic [21,24] and hidden Markov model (HMM) [25,26] are frequently used, the other advanced algorithms are multiple-hypothesis [27], probability theory [5], Dempster–Shafer theory of evidence [28], and Kalman filter [8], etc. Most of advanced map-matching algorithms are designed for use with high frequency positioning data. They utilize additional information, such as positioning data quality and vehicle speed, to achieve higher success rates than other map-matching algorithms [17]. A comprehensive review of different map-matching algorithms showed that their logic is much more complex, which may lead to slower performance [19]. Their average running time for each trajectory point ranges from seconds to minutes. For example, the average output delay of an online map-matching algorithm based on HMM is 82 s [26].

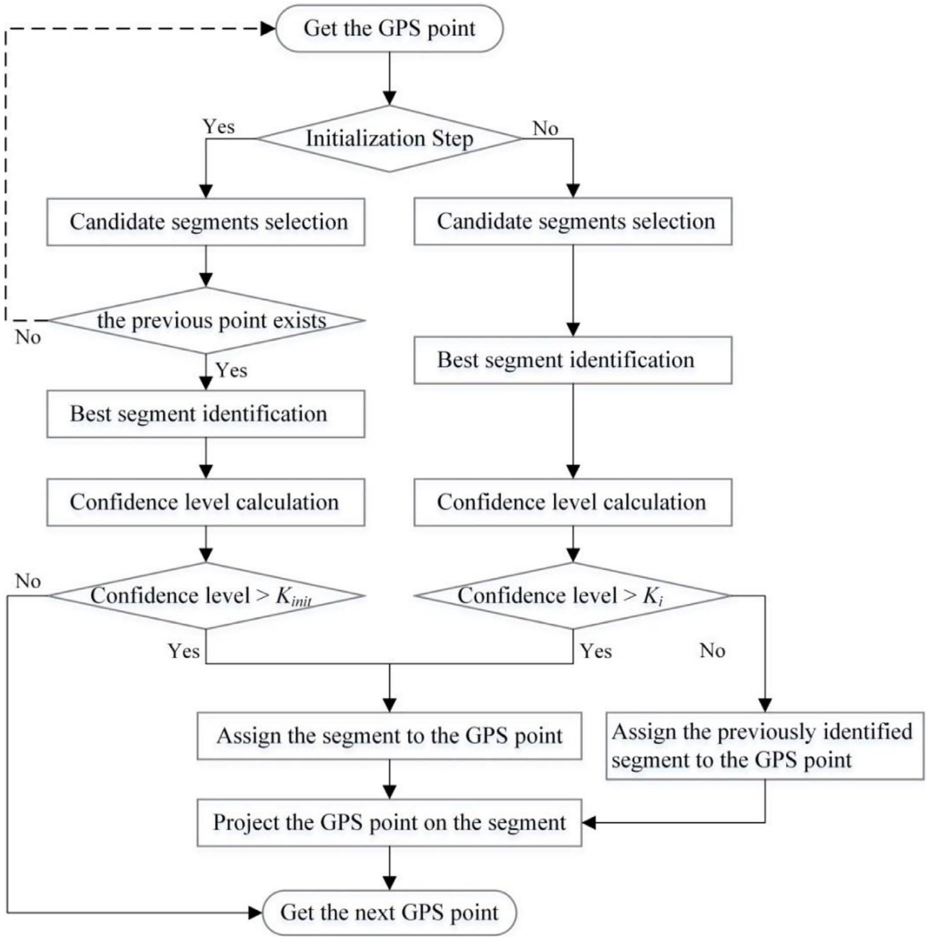


Fig. 1. A Flowchart presenting DW-RMM algorithm.

Weight-based algorithms balance the accuracy and the performance, and take the weighted average scores of specific criteria. Score is assigned to each candidate segment, and choosing the correct segment with the best score. Greenfield [20] developed a weighted map-matching process based on proximity, direction difference, intersecting angle between line connecting GPS points and segment. But heading and speed information obtained from GPS device are ignored, and the accuracy in urban roads was 85.6%. Subsequently, the weight-based map-matching was further enhanced by introducing more topological information such as turn-restriction and segment connectivity (e.g. Quddus et al. [2]). In the algorithm of Velaga et al. [13], the accuracy reached 96.36% in urban roads, and 1/180 s was taken to match one GPS point on average. Nevertheless, direction difference is not considered as a criterion. Some criteria are somewhat subjective and not always efficient. The criteria's weights for each GPS point are also constant. Li et al. [29] proposed a topological algorithm, and the accuracy was enhanced by employing two network features, including grade separation and the direction of traffic flow. The accuracy of their algorithm was 97.7% at a cost of using DR (low-cost Dead-Reckoning) sensors and DEM (Digital Elevation Model) in addition to GPS. However, DEM may be unavailable or not in a suitable scale for other area.

Compared to the aforementioned algorithms, the recent weight-based map-matching algorithm [3] set up three functions of weights (distance, direction difference and heading difference), modified the default direction of candidate segments and considered segment connectivity and turn-restriction. A confidence level is defined to identify the correct segment, no longer a simple choice of the best score [2,13,20]. The average accuracy of the algorithm reached 92.2%. Yet topological constraints such as connectivity and turn-restrictions are not always checked. The algorithm failed to find correct segment in two roads with an acute angle (Y-shaped intersection), and could not deal with high positional error in GPS points and parallel roads. One of the primary features of weight-based algorithms is that they are very effective in for matching high frequency positioning data because the segment connectivity is not useful and segment skipping problem may occur when matching low frequency positioning data [17]. Given discussion above, weight-based algorithms show great potential in matching high frequency positioning data.

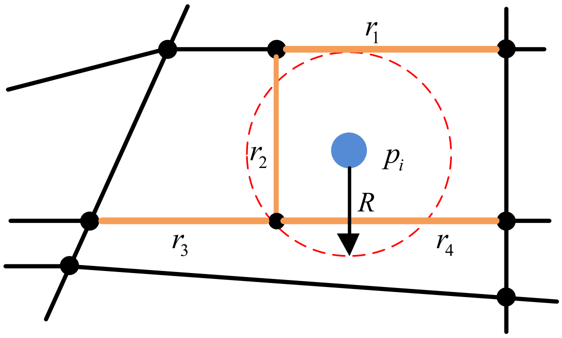


Fig. 2. An example of candidate segment selection.

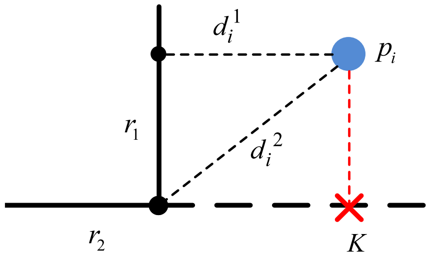


Fig. 3. Distance between the GPS point and each candidate segment.

3. An enhanced real-time map-matching algorithm

A flowchart of the proposed algorithm is showed in Fig. 1. The map-matching process has two steps: initialization and tracking match. And each step is mainly composed of three parts, candidate segments selection, best segment identification, and confidence level calculation. Segments near the GPS point are selected as candidate segments, weighted scores of candidate segments are calculated based on four criteria, and identifying the best segment with the highest score. Before projecting the point on the segment, the confidence level is calculated and taken into account.

The aim of initial process is to find the correct segment for the first GPS point. Since a wrong assignment of first GPS point will lead to a sequence of mismatches, a relatively conservative delay matching approach is used in initialization step. For given two consecutive GPS points, calculating scores of their all possible matched segments, and choosing the match with the highest score. If the confident level is less than K_{init}, the algorithm will discard the point, wait for the coming GPS point, and continues to process in the same way until the condition is satisfied.

When the desired confidence level is reached, the algorithm proceeds to next step, tracking match. This step regards current GPS point as the connected point to the previous one. The segment with the highest score is selected only when the confidence level exceeds K_i. Otherwise, the algorithm chooses the previously identified segment. After identifying the optimal segment, the GPS point is perpendicularly projected onto the segment, updating the position of the vehicle. If there is only one candidate segment, the point is directly projected. Moreover, if the projection is on the extension of the segment, the near node to the GPS point will be treated as the position of the vehicle.

3.1. Candidate segments selection

A limited number of segments near the GPS point are selected as candidate segments, which are taken as possible road for the vehicle. The aim is to improve the efficiency of the algorithm by reducing the searching space before map matching. The segments falling in or crossing the error circle are extracted. The circle is centered at the location of GPS point, and the radius is defined by the horizontal delusion of precision (HDOP). Compared with the error ellipse [13,24] and the gridding method [23,30], it is simple and easy for the module to implement, without losing efficiency. Fig. 2 shows an example of candidate segments selection for a GPS point. It centers at the location of GPS point $\boldsymbol{p}_i$ with radius R, which is defined by the horizontal delusion of precision (HDOP). The smaller the value of HDOP indicates higher position accuracy of $\boldsymbol{p}_i$ and smaller the value of radius. And the candidate segments are $\{r_1, r_2, r_3, r_4\}$.

3.2. Best segment identification

The algorithm mainly considers four criteria to establish the model, (a) distance to segment, (b) heading difference, (c) direction difference and (d) segment connectivity. After the candidate segments are obtained, the score ($\boldsymbol{Score}_i^j$) of the

jth candidate segment at point $\boldsymbol{p}_i$ is calculated from Eqs. (1) and (2), where $\boldsymbol{S}_{dist_i^j}$, $\boldsymbol{S}_{head_i^j}$, $\boldsymbol{S}_{dir_i^j}$ and $\boldsymbol{S}_{con_i^j}$ represent the scores of closeness, heading difference, direction difference, and connectivity, respectively. $\boldsymbol{W}_{dist}$, $\boldsymbol{W}_{head}$, $\boldsymbol{W}_{dir}$ and $\boldsymbol{W}_{con}$ are the corresponding weights to each criterion. Inspired by Hashemi and Karimi [3], weights of four criteria in this article are established based on the parameters, which affect the reliability of each weight, such as HDOP, instantaneous velocity and traveled distance from the previous GPS point (x_{i-1}, y_{i-1}) to the current GPS point (x_i, y_i).

$$\boldsymbol{Score}_i^j = \boldsymbol{S}_{dist_i^j} + \boldsymbol{S}_{head_i^j} + \boldsymbol{S}_{dir_i^j} + \boldsymbol{S}_{con_i^j} \tag{1}$$

$$\begin{cases} \boldsymbol{S}_{dist_i^j} = \boldsymbol{W}_{dist} f\left(d_i^j\right) \\ \boldsymbol{S}_{head_i^j} = \boldsymbol{W}_{head} f\left(\theta_i^j, \Delta\theta\right) \\ \boldsymbol{S}_{dir_i^j} = \boldsymbol{W}_{dir} f\left(\beta_j\right) \\ \boldsymbol{S}_{con_i^j} = \boldsymbol{W}_{con} f\left(\Delta l\right) \end{cases} \tag{2}$$

3.2.1. Distance to segment

Intuitively, distance generally refers to the vertical distance from the GPS point to the road segment, but the projected location of the point is not always on the road segment. As shown in Fig. 3, the projection K is on the extension of the segment r_2. If the projection is on the extension of the segment, the near node to the GPS point will be treated as the position of the vehicle. d_i^1 and d_i^2 denote the distance between the GPS point $\boldsymbol{p}_i$ and segment r_1 and segment r_2, respectively.

It has been shown that the distance between the GPS point and candidate segments can be modeled accurately using a Gaussian distribution [28]. As we know, if the number of events is very large, then the Gaussian distribution function may be used to describe physical events. The Gaussian distribution is a continuous function which approximates the exact binomial distribution of events.

We assume that the mean of Gaussian noise influencing d_i^j is zero, standard deviation is σ_g, and the distance is modeled as Eq. (3), where d_i^j is the distance between GPS point $\boldsymbol{p}_i$ and the candidate segment r_j. This equation is based on the fact that the smaller the distance is, the greater the possibility of segment r_j being the correct road. The standard deviation σ_g is determined through Eq. (4) based on Median Absolute Deviation [31]. In statistics, the Median Absolute Deviation (MAD) is a measure of statistical dispersion and it is a robust measure of the variability of a univariate sample of quantitative data. The variance and standard deviation are also measuring of spread, but they are more affected by extremely high or extremely low values and non-normality. If your data is normal, the standard deviation is usually the best choice for assessing spread. However, if your data is not normal, the MAD is one statistic you can use instead.

$$f\left(d_i^j\right) = \frac{1}{\sqrt{2\pi}\sigma_g} e^{-\frac{\left(d_i^j\right)^2}{2\sigma_g{}^2}} \tag{3}$$

Because the reliability of calculated distance d_i^j depends on the horizontal precision of the GPS point, weight of distance is determined through Eq. (4) based on HDOP, where $\boldsymbol{p}_i.hdop$ denotes the HDOP of current GPS point $\boldsymbol{p}_i$. $HDOP_1$ and $HDOP_2$ are the empirical values of the lower and upper thresholds, respectively. The lesser is $\boldsymbol{p}_i.hdop$, the more accurate is the location information of GPS point, and the higher is $\boldsymbol{W}_{dist}$.

$$\boldsymbol{W}_{dist} = \begin{cases} 1, & \boldsymbol{p}_i.hdop \le HDOP_1 \\ \dfrac{\boldsymbol{p}_i.hdop - HDOP_1}{HDOP_2 - HDOP_1}, & HDOP_1 < \boldsymbol{p}_i.hdop < HDOP_2 \\ 0, & \boldsymbol{p}_i.hdop \ge HDOP_2 \end{cases} \tag{4}$$

3.2.2. Heading difference

The heading information of a GPS point from the GPS receiver reflects the instantaneous direction of the vehicle. It should be noted that the heading direction should be the same as the orientation of segment when a vehicle is traveling on a road at a speed. Fig. 4 illustrates definitions of heading difference in different steps. In the initialization, heading difference is defined as the angle between heading of a point and direction of the road segment, shown as θ_i^j in Fig. 4(a). To improve the ability to handle intersections, the heading of previous GPS point is considered in tracking match step. As shown by Fig. 4(b), heading difference $\Delta\theta$ is the difference between θ and φ_j according to Eq. (5), where θ represents the angle between the heading of the current point and the heading of the previous point, φ_j denotes the difference between the orientation of candidate segment r_j and that of the last identified segment.

The heading difference is modeled as Eq. (5), introducing a constant to reduce the sensitivity. In the initialization step, the heading difference is calculated based on θ_i^j, which is the difference between heading of the point $\boldsymbol{p}_i$ and the direction

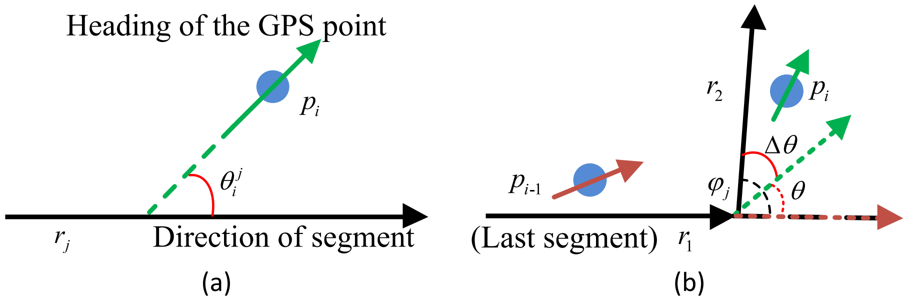


Fig. 4. Heading difference in different steps. (a) Heading difference in initialization. (b) Heading difference in tracking match.

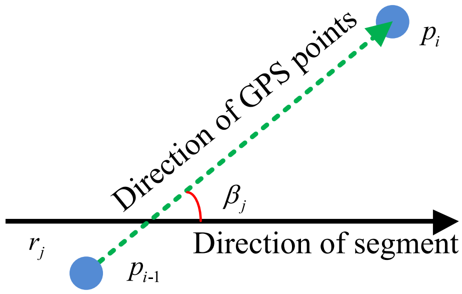


Fig. 5. Direction difference.

of candidate segment r_j. So A equals 1 and B is 0. And in next step, the heading difference is determined by $\Delta\theta$, A is 0 and B equals 1. The value of θ_i^j and $\Delta\theta$ are in the range of [0, 180°].

$$\Delta\theta = |\theta - \varphi_j| \tag{5}$$

$$f\left(\theta_i^j, \Delta\theta\right) = \frac{1 + A\cos\theta_i^j + B\cos\Delta\theta}{2} \tag{6}$$

The reliability of vehicle's heading is greatly affected by the instantaneous velocity $\boldsymbol{p}_i.v$, the weight of heading difference is determined through Eq. (7). The faster the vehicle's speed, the more reliable the heading of the GPS point is [2], and the more precise the calculated heading difference is. In addition, the impact of the previous point's velocity $\boldsymbol{p}_{i-1}.v$ is also taken into account. V_1 and V_2 are the empirical values of the lower and upper limits of speed, respectively. If $\boldsymbol{p}_i.v$ or $\boldsymbol{p}_{i-1}.v$ is less than V_1, then $\boldsymbol{W}_{head}$ will be considered 0 and heading difference will not be considered as a criterion in the best segment identification process. On the contrary, if $\boldsymbol{p}_i.v$ and $\boldsymbol{p}_{i-1}.v$ exceed V_2, then $\boldsymbol{W}_{head}$ will be 1. For any other values of $\boldsymbol{p}_i.v$ and $\boldsymbol{p}_{i-1}.v$, the heading weight will be between 0 and 1.

$$\boldsymbol{W}_{head} = \begin{cases} 0, & \boldsymbol{p}_i.v < V_1 \text{ or } \boldsymbol{p}_{i-1}.v < V_1 \\ \dfrac{\boldsymbol{p}_i.v + \boldsymbol{p}_{i-1}.v - 2V_1}{2\left(V_2 - V_1\right)}, & \text{others} \\ 1, & \boldsymbol{p}_i.v > V_2 \text{ and } \boldsymbol{p}_{i-1}.v > V_2 \end{cases} \tag{7}$$

3.2.3. Direction difference

GPS heading is sometimes very inaccurate, especially when the vehicle's speed is close to zero [30], and it takes a toll on the reliability of $\boldsymbol{W}_{head}$. Another criterion, direction difference, is used to find the correct segment in complex urban road networks. The direction difference is defined as the difference between direction of a segment and direction of the connecting two consecutive GPS points. As shown by Fig. 5, β_j denotes the direction difference between GPS points and the candidate segment r_j. The direction difference is modeled as Eq. (8).

The weight of direction is calculated from Eq. (9) based on the traveled distance between two GPS points. $l_{p_{i-1}\to p_i}$ denotes the Euclidean distance between the previous GPS point $\boldsymbol{p}_{i-1}$ and the current point $\boldsymbol{p}_i$, l_{mean} is the average traveled distance between two adjacent GPS points. The longer traveled distance is, the lesser the direction of GPS points is affected by GPS errors, and the higher the $\boldsymbol{W}_{dir}$ will be.

$$f\left(\beta_j\right) = \frac{1 + \cos\beta_j}{2} \tag{8}$$

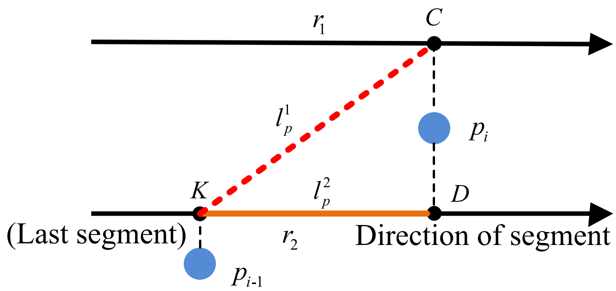


Fig. 6. An example of correct segment identification based on segment connectivity.

$$\boldsymbol{W}_{dir} = \begin{cases} 1, & l_{p_{i-1}\to p_i} > l_{mean} \\ \dfrac{l_{p_{i-1}\to p_i}}{l_{mean}}, & l_{p_{i-1}\to p_i} \le l_{mean} \end{cases} \tag{9}$$

3.2.4. Segment connectivity

Based on three criteria mentioned above, error matching still occurs on some special segments, such as parallel roads. Since the identified segments of adjacent GPS points must be connected, the connectivity criterion describes the relationship between the candidate segment and the last match. Fig. 6 illustrates an example of identifying the correct segment in parallel roads based on segment connectivity.

The segment connectivity is modeled as Eq. (10), which represents the probability of each candidate segment being the correct. A factor α is used to adjust the sensitivity of the connectivity weight. Δl computed by Eq. (11), is the difference between the traveled distance and the distance between the projections, where l_{tr} is the traveled distance between $\boldsymbol{p}_{i-1}$ and $\boldsymbol{p}_i$, l_p^j denotes the distance between the projection of current GPS point on candidate segment r_j and the projection on last segment. As shown in Fig. 6, $l_p^1 = |KC|$, $l_p^2 = |KD|$. K is the projection of previous point $\boldsymbol{p}_{i-1}$ on the last identified segment, C and D are projections of $\boldsymbol{p}_i$ on candidate segment r_1 and r_2, respectively. The traveled distance l_{tr} can be calculated from the speed and timestamp of GPS point according to Eq. (12) where $\overline{v_m}$ is the average speed of last m consecutive points' speed, including GPS point $\boldsymbol{p}_i$, and $m \ge 2$. t_{i-m+1} and t_i are the timestamp of point $\boldsymbol{p}_{i-m+1}$ and $\boldsymbol{p}_i$.

$$f(\Delta l) = e^{-\alpha \Delta l} \tag{10}$$

$$\Delta l = |l_{tr} - l_p^j| \tag{11}$$

$$l_{tr} = \max(\overline{v_m} \times (t_i - t_{i-m+1}), l_{p_{i-m+1}\to p_i}) \tag{12}$$

The weight of connectivity is determined through Eq. (13) based on the obtained information from GPS data. And Δl_{mean} is the mean value of Δl. The smaller the value of Δl is, the higher is $\boldsymbol{W}_{con}$. When the value of Δl is no more than Δl_{mean}, the $\boldsymbol{W}_{con}$ will be equal to 1. For other values, $\boldsymbol{W}_{con}$ is between 0 and 1.

$$\boldsymbol{W}_{con} = \begin{cases} 1, & \Delta l \le \Delta l_{\text{mean}} \\ f(\Delta l), & \Delta l > \Delta l_{\text{mean}} \end{cases} \tag{13}$$

3.3. Confidence level calculation

Most of the existing algorithms select the best segment with the highest score. However, sometimes the scores between the best segment and the second best segment are very close. Simply selecting the segment with the highest score leads to a wrong match. Uncertainty in a map-matched location stems from three sources: uncertainty in positioning sensor, digital map, and map-matching algorithm [32]. This uncertainty can be formulated in a confidence level and assigned to the map-matched location [3]. Therefore, a new method, considering network density and complexity around the GPS point, is proposed to formulate the confidence level on the selected segment. The range of confidence level is between 0 and 1, the same as the range of all the criteria, which $\boldsymbol{W}_{dist}$, $\boldsymbol{W}_{head}$, $\boldsymbol{W}_{dir}$ and $\boldsymbol{W}_{con}$ are calculated using Eqs. (4), (7), (9) and (13). The contribution of each parameter in determining the confidence level using Eq. (14) is the same as its contribution in scoring candidate segments using Eq. (2). In this equation, $Dist_i^f$, $Head_i^f$, Dir_i^f and Con_i^f are the scores of four criteria for the best segment r_i^f. Equally, $Dist_i^s$, $Head_i^s$, Dir_i^s and Con_i^s are the scores of the second best segment.

$$K_i = \frac{1}{\boldsymbol{W}_{dist} + \boldsymbol{W}_{head} + \boldsymbol{W}_{dir} + \boldsymbol{W}_{con}} \begin{pmatrix} \boldsymbol{W}_{dist} \dfrac{|Dist_i^f - Dist_i^s|}{Dist_i^f + Dist_i^s} + \boldsymbol{W}_{head} \dfrac{|Head_i^f - Head_i^s|}{Head_i^f + Head_i^s} + \\ \boldsymbol{W}_{dir} \dfrac{|Dir_i^f - Dir_i^s|}{Dir_i^f + Dir_i^s} + \boldsymbol{W}_{con} \dfrac{|Con_i^f - Con_i^s|}{Con_i^f + Con_i^s} \end{pmatrix} \tag{14}$$

The first GPS point is a challenge for all map-matching algorithms, because a wrong segment assigned to the first GPS point will lead to a sequence of mistaken matches. Therefore, a relatively conservative delay matching method is used to ensure that there must be enough confidence on the identified segment before switching from the initialization step to the next step [3]. For example, $\boldsymbol{p}_1$ and $\boldsymbol{p}_2$ are two GPS points in the initialization step, the candidate segments are $C_1 = \{r_1^1, r_1^2, r_1^3\}$ and $C_2 = \{r_2^1, r_2^2\}$. Calculating the scores of each candidate segment in C_2 under all possible matches of $\boldsymbol{p}_1$ in C_1. There are six possible combinations, choosing the two highest scores, such as $\boldsymbol{Score}_2^1|(\boldsymbol{p}_1 \to r_1^2)$ and $\boldsymbol{Score}_2^1|(\boldsymbol{p}_1 \to r_1^3)$. If the condition of Eq. (15) is satisfied, then the segment r_1^2 is matched to $\boldsymbol{p}_1$, and r_2^1 is assigned to $\boldsymbol{p}_2$. Otherwise, $\boldsymbol{p}_1$ is discarded as an outlier, $\boldsymbol{p}_2$ and $\boldsymbol{p}_3$ are processed in the same way and remains in the initialization step. K_{init} is the confidence level in the initialization step by taking the empirical value.

$$\boldsymbol{Score}_2^1 \left|\left(\boldsymbol{p}_1 \to r_1^2\right) - \boldsymbol{Score}_2^1\right| \left(\boldsymbol{p}_1 \to r_1^3\right) > K_{init} \tag{15}$$

We formulate the pseudo code of our algorithm as follows. The algorithm deals with trajectory points incrementally, it calculates score of each candidate segment, and outputs the optimal solution. The input data is a set of GPS points generated in a real-time scenario. The output is a set of matched roads for all the trajectory points. Trajectory T the algorithm input data is a dynamic set of GPS points in a real-time scenario. The output is a matched road set M for all the trajectory points.

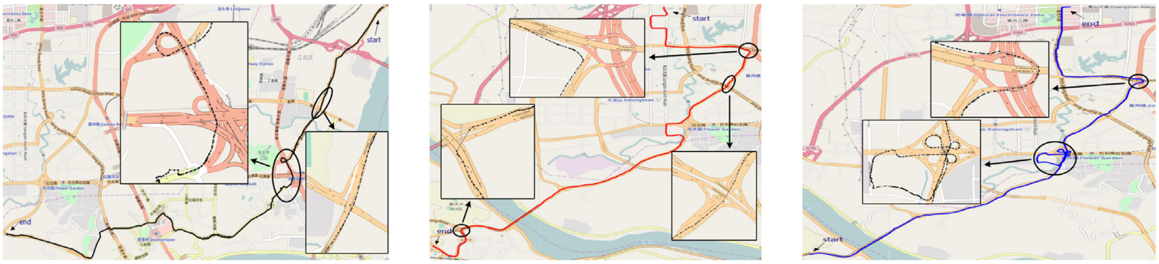
Algorithm: DW-RMM Algorithm

Input: GPS trajectory $T = \{p_1, p_2, \cdots, p_n\}$; road network G.
Output: matched road $M = \{r_1, r_2, \cdots, r_k\}$ and vehicle's current position.

```
Let scoresInit[c][l] store the score of p_i's candidate segment l when p_{i-1} is assigned to c
Let scores[i][l] store the score of p_i's candidate segment l
Let s store the optimal segment of the current point p_i
for i =2 to n do
  Find the set of candidate segments C nearest to the previous point p_{i-1}
  Find the set of candidate segments L nearest to the current point p_i
  if isInitStep then /* initialization  step */
    for each c in C do
      for each l in L do
        scoresInit[c][l] = CalcScores(p_{i-1}, p_i, c, l)
    [first, second] = findTop2Candidate(scoresInit)
    if first – second > K_init then
      s = l_f /* l_f is the candidate segment with the highest score */
  else  /* tracking matching  step */
    r = M.getTail() /* get matched road segment of last point p_{i-1} */
    for each l in L do
      scores[i][l] = CalcScores(p_{i-1}, p_i,r, l)
    [first, second] = findTop2Candidate(scores[i])
    K_i = CalcConfidenceLevel(p_i)
    if first – second > K_i then
      s = l_f
    else
      s = r
  M.add(s)
  projectToRoad(p_i, s)
return M
```

4. Evaluation

We comprehensively evaluate the effectiveness and efficiency of the proposed algorithm. In this section, we first present the data description and the experimental environment, followed by a detailed analysis of the evaluation results.



#1 #2 #3

Fig. 7. Raw GPS trajectories in Chongqing.

Table 1
Characteristics of the GPS trajectories.

ID	Number of points	Accuracy range (m)	Mean accuracy (m)	Speed (m/s)	Mean speed (m/s)	Points with speed (%)	Length (km)
#1	2064	3.5∼68	13	0∼14	2.83	54.8	12.4
#2	1764	4.0∼70	7.5	0∼12	3.76	68.2	11.8
#3	1832	4.5∼70	10.9	0∼12.8	3.52	61	12.3

Table 2
Evaluation results by proposed map-matching algorithm.

ID	Number of points	Percentage of projection (%)	Correctly assigned segment	Success rate (%)	Process time (s)	Average time (ms)
#1	2064	91.4	1817	96.3	7.19	3.48
#2	1764	98.3	1699	97.98	5.49	3.11
#3	1832	97.2	1738	97.64	5.48	2.99
Total	5660	95.6	5254	97.31	18.16	3.20

The identification accuracy and computational performance of the algorithm are compared with different map-matching algorithms.

4.1. Dataset and experimental environment

The proposed algorithm is tested in an urban environment, and the data is from GPS, the sole positioning sensor. We have collected three different bus trajectories in Chongqing, with a total length of 36 km, and a sampling rate of 1 s to 3 s. Since the map data from OpenStreeMap (OSM; www.openstreetmap.org) is not complete, some non-main road's data is missing, which results in part of GPS points cannot find the right match. These GPS points are removed in advance in data preprocessing.

The raw GPS trajectories are presented in Fig. 7, and Table 1 shows their characteristics. As shown by Fig. 7, trajectories involve parallel roads, intersections, overpasses and roundabouts, which are amplified to show the errors in original track points, and the complexity and density of roads. The algorithm was coded in Java and tested on a 32-bit operating system with 2.10 GHz Intel Core i3-2310M central processing unit (CPU) and 4G memory (Mobile devices will be considered in future work).

4.2. Results

The empirical values used to calculate the coefficients of distance and heading difference are determined through the HDOP and speed of all the trajectory points, $V_1 = 2.35$ m/s, $V_2 = 3.35$ m/s, $HDOP_1 = 4$, $HDOP_2 = 7$. And the achieved evaluation results of the algorithm for each GPS trajectory are listed in Table 2. On average, 95.6% of the GPS points are projected, accuracy is 97.31%, and the delay is about 3.20 ms per point.

In previous work, real-time map-matching algorithms were poor at dealing with parallel roads and intersections [3]. However, our algorithm has solved these problems to a large extent. Fig. 8(a) shows the performance of the proposed algorithm while the car passes through a Y-shaped intersection. Despite original track points deviate the real road, the algorithm finds the correct segment. There is another example of matching at an overpass in Fig. 8(b), the candidate segments are parallel and in the same direction, which are regard as parallel roads. The wrong segment identification is resolved by the algorithm.

Additionally, the matching result of special roads is presented in Fig. 9(b). Fig. 9(a) shows the original trajectory of a vehicle which has passed through a complex cloverleaf intersection. It comprises parallel roads, fly-overs, roundabouts, T-junctions, and auxiliary roads. As we can see, the proposed algorithm performs perfectly and finds correct segment for each point.

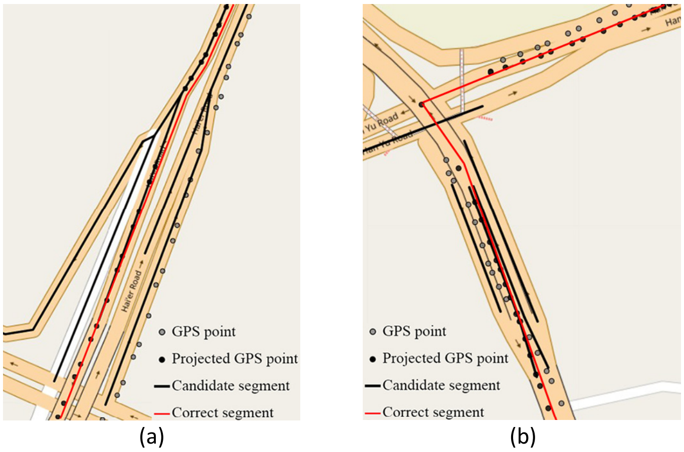


Fig. 8. Correct performance at Y-junction and parallel roads. (a) Correct performance while the car crosses Y-junction; (b) The map-matching algorithm handles parallel segments well.

Fig. 9. Contrast between the original trajectory and the trajectory after processing. (a) Original trajectory. (b) Projection trajectory.

The algorithm soundly copied with most of the cases as long as not all characteristics of a GPS point, as distance, heading, direction and connectivity lead the algorithm to the wrong segment. But wrong segment identification still occurs sometimes. Fig. 10 illustrates a failure in correct segment identification while the car waits at the traffic light.

The vehicle has just passed a U-shaped intersection, marked with a circle in Fig. 10. The speeds of successive GPS points are close to zero when the vehicle waits for the green light, the direction and heading of points are inactive to identify the correct segment. Besides, the two candidate segments are connected to the last matching segment, the connectivity is out of use, too. Only the distance criterion works, and selects the wrong match eventually, which is caused by the error in GPS positioning. There is no way for the algorithm to figure the correct segment out when the algorithm is fed with wrong data [3]. The confidence level may help prevent projecting the GPS points on the wrong segment, but the right segment cannot be identified until the data associated with the GPS point are more consistent with the characteristics of the correct segment than other segments.

4.3. Comparison and discussion

When comparing the performance of different map-matching algorithms, three points must be also considered. One is related to the observation that employing additional positioning sensors, besides GPS, could compensate for the low accuracy of GPS and improve the success rate. Another point is concerned with the road complexity in environment within which a map-matching algorithm is conducted. The final point is related to the response time, which is important for real-time applications.

Table 3 summarizes real-time map-matching algorithms with criteria, positioning technology, sampling rate, method used for correct segment identification and average running time, which is the average output latency incurred by the algorithm for each trajectory point. They are tested on different data sets with diverse sampling rate, but only in urban

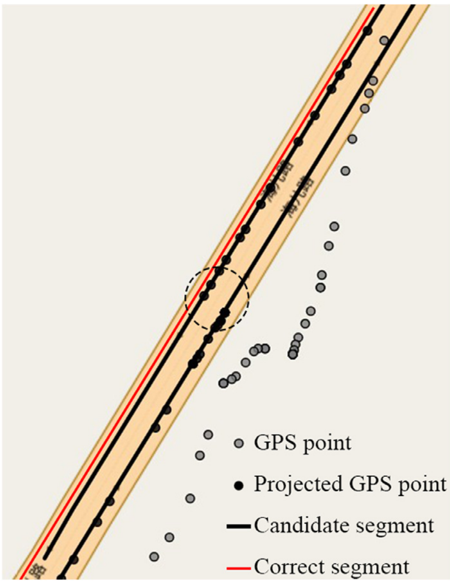


Fig. 10. Failure in correct segment identification while the car waits at the traffic light.

Table 3
Comparison between real-time map-matching algorithms.

Algorithm	Criteria for segment identification	Sensors	Sampling rate	Method	Correct segment assignment (%)	Time (ms)
Our algorithm	Closeness Heading difference Direction difference Segment connectivity Turn restriction	GPS	1–3 s	Weight-based	97.31	3.20
Velaga et al. (2009) [13]	Closeness Heading difference Turn-restrictions Segment connectivity	GPS	1 s	Weight-based	96.8	5.56
Goh et al. (2012) [26]	Closeness Segment connectivity	DR GPS	1–3 s 10 s–5 min	HMM	97 92.1	82000
Li et al. (2013) [29]	Closeness Heading difference Segment connectivity	GPS DR DEM	Not specified	Weight-based	97.7	248.85
Quddus and Washing (2015) [17]	Closeness Bearing difference Shortest path distance Heading difference	GPS	1 s 5 s 30 s 60 s	Weight-based	98.6 98.4 98.9 98.5	<12.59 <18.89 <56.39 <74.07
Hashemi and Karimi (2016) [3]	Closeness Heading difference Direction difference Segment connectivity Turn restriction	GPS	1 s	Weight-based	92.19	1.18

areas. It reveals that the proposed algorithm has better accuracy than the most of list methods. By looking the accuracy which was obtained exclusively with the data from GPS devices in high sampling rate, only the algorithm developed by the Quddus and Washing [17] attained 98.6% of accurately identified segments. Besides this, the success rate of our algorithm reached 97.31%, which is more than success rates achieved by Goh and Dauwels [26] HMM algorithm and the weight-based algorithm recently proposed by Hashemi and Karimi [3], and it is almost the same as the accuracy that weight-based algorithms obtained through additional data, such as DR and DEM.

Our initial work does not consider optimization of the map-matching process, resulting in more than 18 ms processing for a single trajectory point. After optimizing the calculation of segment direction, the average running time is reduced to 3.20 ms. However, it is not easy to compare running times of different algorithms, because they are not usually presented or they are provided without other detail data. Additionally, they are not tested in the same experimental environment.

Running times of these methods range from milliseconds to minutes. Processing time of Li et al. [29] algorithm required to process 2399 fixes (i.e. London data) is 597 s, which is achieved using a PC with 4 GB RAM and 3.4 GHz processing speed. Quddus and Washing [17] spent roughly 100 s to cope with 7940 fixes. The algorithm developed by Velaga et al. [13] carried out the map-matching of 180 positioning fixes per sec (with a laptop of 1 GB RAM and 1.46 processor speed). Furthermore, the method presented by Hashemi and Karimi [3] took about 1.18 ms to process per fix. By comparison, our proposed algorithm is a bit slower, because it has to do intensive calculations in order to compute the score of each candidate segment. It is obvious that algorithms have good efficiencies to meet the request in real-time application. To summarize, considering accuracy, simplicity, and performance together, our algorithm outstrips all the aforementioned algorithms, which presents a good accuracy and high matching efficiency.

5. Conclusion

This study showed that weight-based map-matching algorithm employed with appropriate weights can achieve a high accuracy, even better than the advanced algorithms. The average matching accuracy of the proposed algorithm reached 97.31%, and the delay of single point is about 3.20 ms, which is preferable to the most real-time map-matching algorithms. Furthermore, the algorithm not only can correctly process the match of Y-junction and parallel roads, but also properly handle the curve of the overpass and special roads. The results demonstrate that the algorithm is capable of identifying correct segment from complicated and dense urban road networks. In summary, the algorithm has good accuracy and performance to meet the demand for real-time applications.

The algorithm, however, failed to find the correct segment when the vehicle is waiting at the traffic light. It means that it is sensitive to the low speed in complex matching environment, which is the direction for further research. Besides, as discussed in the introduction, low frequency GPS data is another research direction. Finally, we plan to implement the algorithm on a distributed collaborative high-performance computing framework to improve performance further. At the same time, porting to mobile platforms will also be considered.

Acknowledgments

This work is supported by the National Key Research and Development Program of China under grant NO. 2017YFC0212100, Key Research and Development Program of Chongqing, China under grant NO. cstc2018jszx-cyztzxX0019, University Research Program of Ford, U.S.A. under grant NO. DEPT2018-J030.1.

References

[1] E. Bouillet, A. Ranganathan, Scalable, real-time map-matching using IBM's system S, in: 17th International Conference on Mobile Data Management, 2010, pp. 249–257, http://dx.doi.org/10.1109/MDM.2010.36.

[2] M.A. Quddus, W.Y. Ochieng, Z. Lin, R.B. Noland, A general map matching algorithm for transport telematics applications, GPS Solut. 7 (2003) 157–167, http://dx.doi.org/10.1007/s10291-003-0069-z.

[3] M. Hashemi, H.A. Karimi, A weight-based map-matching algorithm for vehicle navigation in complex urban networks, J. Intell. Transp. Syst. 20 (2016) 573–590, http://dx.doi.org/10.1080/15472450.2016.1166058.

[4] Guanghan Peng, Hongzhuan Zhao, Xiaoqin Li, The impact of self-stabilization on traffic stability considering the current lattice's historic flux for two-lane freeway, Physica A (2019) 31–37, http://dx.doi.org/10.1016/j.physa.2018.09.173.

[5] M.A. Quddus, W.Y. Ochieng, R.B. Noland, Current map-matching algorithms for transport applications: State-of-the art and future research directions, Transp. Res. C 15 (2007) 312–328, http://dx.doi.org/10.1016/j.trc.2007.05.002.

[6] Sharath, M.N. Velaga, R. Nagendra, Quddus, A. Mohammed, A dynamic two-dimensional (D2D) weight-based map-matching algorithm, Transp. Res. C 98 (2019) 409–432, http://dx.doi.org/10.1016/j.trc.2018.12.009.

[7] Mingliang Che, Yingli Wang, Chi Zhang, Xinliang Cao, An enhanced hidden Markov map matching model for floating car data, Sensors 18 (2018) http://dx.doi.org/10.3390/s18061758.

[8] H. Wang, L. Jin, Z. Hou, R. Fang, W. Mei, H. Jian, Research on parallelized real-time map matching algorithm for massive GPS data, Cluster Comput. 20 (2017) 1123–1134, http://dx.doi.org/10.1007/s10586-017-0869-5.

[9] M. Nikolić, J. Jović, Implementation of generic algorithm in map-matching model, Expert Syst. Appl. 72 (2017) 283–292, http://dx.doi.org/10.1016/j.eswa.2016.10.061.

[10] M. Rahmani, H.N. Koutsopoulos, Path inference from sparse floating car data for urban networks, Transp. Res. C 30 (2013) 41–54, http://dx.doi.org/10.1016/j.trc.2013.02.002.

[11] MIWA, Tomio, KIUCHI, Daisuke, YAMAMOTO, Toshiyuki, MORIKAWA, Takayuki, Development of map matching algorithm for low frequency probe data, Transp. Res. C 22 (2012) 132–145, http://dx.doi.org/10.1016/j.trc.2012.01.005.

[12] B.Y. Chen, H. Yuan, Q. Li, W.H.K. Lam, S.-L. Shaw, K. Yan, Map-matching algorithm for large-scale low-frequency floating car data, Int. J. Geogr. Inf. Sci. 28 (2014) 22–38, http://dx.doi.org/10.1080/13658816.2013.816427.

[13] N.R. Velaga, M.A. Quddus, A.L. Bristow, Developing an enhanced weight-based topological map-matching algorithm for intelligent transport systems, Transp. Res. C 17 (2009) 672–683, http://dx.doi.org/10.1016/j.trc.2009.05.008.

[14] R. Mohamed, H. Aly, M. Youssef, Accurate real-time map matching for challenging environments, IEEE Trans. Intell. Transp. Syst. 18 (2017) 847–857, http://dx.doi.org/10.1109/TITS.2016.2591958.

[15] E. Algizawy, T. Ogawa, A. El-Mahdy, Real-time large-scale map matching using mobile phone data, ACM Trans. Knowl. Discov. Data 11 (2017) 1–38, http://dx.doi.org/10.1145/3046945.

[16] Y. Liu, Z. Li, Erratum to: A novel algorithm of low sampling rate GPS trajectories on map-matching, EURASIP J. Wireless Commun. Networking 2017 (2017) 150, http://dx.doi.org/10.1186/s13638-017-0814-6.

[17] M. Quddus, S. Washington, Shortest path and vehicle trajectory aided map-matching for low frequency GPS data, Transp. Res. C 55 (2015) 328–339, http://dx.doi.org/10.1016/j.trc.2015.02.017.

[18] Chen Chao, Ding Yan, Xie Xuefeng, Zhang Shu, A three-stage online map-matching algorithm by fully using vehicle heading direction, J. Ambient Intell. Hum. Comput. 9 (2018) 1623–1633, http://dx.doi.org/10.1007/s12652-018-0760-0.
[19] M. Hashemi, H.A. Karimi, A critical review of real-time map-matching algorithms: Current issues and future directions, Comput. Environ. Urban Syst. 48 (2014) 153–165, http://dx.doi.org/10.1016/j.compenvurbsys.2014.07.009.
[20] J.S. Greenfeld, Matching GPS observations to locations on a digital map, Transp. Res. Board (2012) 13, http://dx.doi.org/10.4236/pos.2013.43023.
[21] M. Ren, H.A. Karimi, A fuzzy logic map matching for wheelchair navigation, GPS Solut. 16 (2012) 273–282, http://dx.doi.org/10.1007/s10291-011-0229-5.
[22] J.S. Kim, J.H. Lee, T.H. Kang, W.Y. Lee, Y.G. Kim, Node based map matching algorithm for car navigation system, in: Proceedings of the 29th International Symposium on Automotive Technology and Automation, 1996, pp. 121–126.
[23] G.E. Taylor, G. Blewitt, D. Steup, S. Corbett, Road reduction filtering for GPS-GIS navigation, Trans. GIS 5 (2010) 193–207, http://dx.doi.org/10.1111/1467-9671.00077.
[24] M.A. Quddus, R. Noland, W.Y. Ochieng, A high accuracy fuzzy logic based map matching algorithm for road transport, J. Intell. Transp. Syst. 10 (2006) 103–115, http://dx.doi.org/10.1080/15472450600793560.
[25] Y. Lou, C. Zhang, Y. Zheng, X. Xie, W. Wang, Y. Huang, Map-matching for low-sampling-rate GPS trajectories, in: Proceedings of the 17th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems (GIS '09), 2009, pp. 352–361, http://dx.doi.org/10.1145/1653771.1653820.
[26] C.Y. Goh, J. Dauwels, N. Mitrovic, M.T. Asif, A. Oran, P. Jaillet, Online map-matching based on hidden markov model for real-time traffic sensing applications, in: Proceedings of the 15th International IEEE Conference on Intelligent Transportation Systems, 2012, pp. 776–781, http://dx.doi.org/10.1109/ITSC.2012.6338627.
[27] M. Jabbour, P. Bonnifait, V. Cherfaoui, Map-matching integrity using multihypothesis road-tracking, J. Intell. Transp. Syst. 12 (2008) 189–201, http://dx.doi.org/10.1080/15472450802448179.
[28] J. Bi, L. Wang, Y. Liu, Traffic data collection system for floating car based on GPS/GPRS/MM, in: Proceedings of the 2nd WRI Global Congress on Intelligent Systems, Vol. 3, 2010, pp. 66–69, http://dx.doi.org/10.1109/GCIS.2010.68.
[29] L. Li, M. Quddus, L. Zhao, High accuracy tightly-coupled integrity monitoring algorithm for map-matching, Transp. Res. C 36 (2013) 13–26, http://dx.doi.org/10.1016/j.trc.2013.07.009.
[30] Y. Tang, A.D. Zhu, X. Xiao, An efficient algorithm for mapping vehicle trajectories onto road networks, in: Proceedings of the 20th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems, 2012, pp. 601–604, http://dx.doi.org/10.1145/2424321.2424427.
[31] P. Newson, J. Krumm, Hidden Markov map matching through noise and sparseness, in: Proceedings of the 17th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems, 2009, pp. 336–343, http://dx.doi.org/10.1145/1653771.1653818.
[32] H.A. Karimi, T. Conahan, D. Roongpiboonsopit, A methodology for predicting performances of map-matching algorithms, in: Proceedings of the 6th International Symposium on Web and Wireless Geographical Information Systems, 2016, pp. 202–213, http://dx.doi.org/10.1007/11935148_19.