


Article

A Data Correction Algorithm for Low-Frequency Floating Car Data

Bijun Li ^{1,2} , Yuan Guo ¹, Jian Zhou ^{1,*} and Yi Cai ¹

¹ State Key Laboratory of Information Engineering in Surveying, Mapping, and Remote Sensing, Wuhan University, Wuhan 430079, China; Lee@whu.edu.cn (B.L.); guoyuan@whu.edu.cn (Y.G.); clarify@whu.edu.cn (Y.C.)

² Engineering Research Center for Spatio-Temporal Data Smart Acquisition and Application, Ministry of Education of China, Wuhan University, Wuhan 430079, China

* Correspondence: JianZhou@whu.edu.cn; Tel.: +86-158-7237-3185

Received: 25 September 2018; Accepted: 24 October 2018; Published: 26 October 2018



Abstract: The data collected by floating cars is an important source for lane-level map production. Compared with other data sources, this method is a low-cost but challenging way to generate high-accuracy maps. In this paper, we propose a data correction algorithm for low-frequency floating car data. First, we preprocess the trajectory data by an adaptive density optimizing method to remove the noise points with large mistakes. Then, we match the trajectory data with OpenStreetMap (OSM) using an efficient hierarchical map matching algorithm. Lastly, we correct the floating car data by an OSM-based physical attraction model. Experiments are conducted exploiting the data collected by thousands of taxis over one week in Wuhan City, China. The results show that the accuracy of the data is improved and the proposed algorithm is demonstrated to be practical and effective.

Keywords: data correction; map matching; OpenStreetMap; floating car

1. Introduction

With the development of self-driving vehicles, lane-level maps have drawn much attention from researchers, internet firms, and carmakers. Currently, lane-level map generation methods mainly include the following three approaches: an integrated navigation system (INS) with three-dimensional (3D) Lidar [1,2]; a vision-based approach [3]; and a floating car-based approach [4–6]. The precision of the map can reach the centimeter level using the Lidar approach, but the cost of the devices and production is the most expensive of the three approaches. The vision-based approach captures the map information by monocular camera or stereo camera, however, the quality of the map is influenced by the condition of the light to a great degree; moreover, the cost of production is also high. The cost of the map is low if the data is collected by a floating car. However, the positional accuracy of global positioning system (GPS) traces can only reach 5–30 m because GPS traces are prone to errors due to the multipath effect and the loss of satellite signals. Therefore, it is a challenging task to produce a map with floating car data.

In this paper, we propose a new data correction method for low-frequency floating vehicle data. We developed an adaptive density optimization method to remove a fraction of the noise points by using a Delaunay triangulation network to construct clusters of points. As OpenStreetMap (OSM) has become one of the most successful projects in Volunteered Geographic Information (VGI) project, it is free and has a range of applications. We attempt to match GPS traces with OSM maps and correct the GPS traces by an OSM-based physical attraction model.

In summary, the main contributions of this paper include the following: (1) we improved the spatial-temporal (ST)-matching algorithm through a hierarchical method to match the floating car data and OSM map, by which the accuracy and efficiency of the algorithm have been improved; (2) we used a physical attraction model to correct the GPS points; (3) we proved that it is feasible to improve the accuracy of the GPS points using the OSM map; and (4) we showed that the improvement in accuracy of the GPS traces was helpful for the production of lane-level maps.

The remainder of the paper is structured as follows: In Section 2, we describe the related work of data correction; in Section 3 we discuss the data correction algorithm; in Section 4 we present the experimental result of the algorithm; and finally we present our conclusions in Section 5.

2. Related Work

Methods for correcting floating car data include the following: (1) clusters; (2) filters; and (3) projecting the GPS points onto an existing map by map matching.

Approaches for removing noise points with clusters have been used by many researchers. Among them, kernel density methods have often been used to build the probability function of the GPS points, and the points with a low spatial density are thus eliminated as outliers [7,8]. Biagioni and Eriksson [9] proposed a single sample point density estimate using a kernel density estimator with a Gaussian kernel and extracted an initial road network based on the density estimate. In References [10–12], the authors clustered the GPS data into regions based on the similarity of the position and direction and filtered out the noise by using the average of the clusters. However, the approaches mentioned above can only remove a small number of the noise points; the precision cannot be completely improved.

Various filtering techniques can also be used to smooth the noise of GPS trajectories. In Reference [13], mean and median filters were applied to smooth the noise. These two filters are similar, in that the median filter merely replaces the mean in the mean filter with the median. The mean and median filters are simple but sensitive to outliers. More complex filters have also been used to correct GPS data, such as a Kalman filter or a particle filter [13–16]; however, they are mainly used in high-frequency GPS data processing.

Map matching algorithms for low-frequency data include both local and global algorithms [17]. Local matching algorithms usually match GPS points based on the distance, heading, speed, topology, and shortest path [18,19]. However, the accuracy of local algorithms is generally low and sensitive to the sampling frequency. Global algorithms match the entire trajectory with the road, based on geometric similarity [20,21]. The limitations of global algorithms are highly time consuming and computationally costly. Zhang et al. [22] matched traces with an existing road map based on three features: the distance, direction, and angle between the trace and map. They estimated a new centerline by modeling the traces with a Gaussian distribution, but the precision of their algorithm can only reach to road level.

Different from the methods mentioned above, this paper corrects GPS points in two steps: we first remove part of the noise by using an adaptive density optimization method, and then correct the points through a physical attraction model and a matched OSM map.

3. Data Correction Algorithm

The processes of our proposed algorithm include the following three steps, as shown in Figure 1: data preprocessing, map matching, and data correction. The following subsections detail each of these steps.

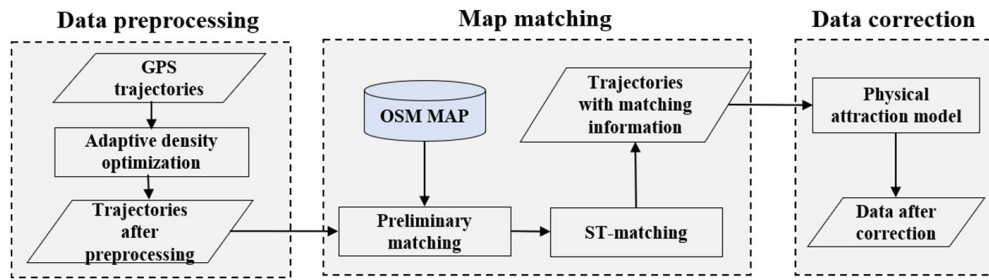


Figure 1. Overview of the proposed algorithm.

3.1. Problem Statement

In the floating car system, the location of a car is recorded by the GPS. A trajectory is a collection of GPS points arranged in a time sequence, $T = \{P_1, P_2, \dots, P_n\}$, as shown in Figure 2. Each of the points P_i has attributes $P_i = \{x_i, y_i, t_i\}$, where (x_i, y_i) are the longitude and latitude of the point, respectively, and t_i is the time the point was collected.

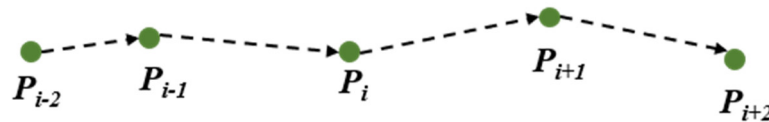


Figure 2. A trajectory of the GPS points. The green dots represent the GPS points and arrows represent the driving direction of floating car. P_i is the ID of the GPS point.

The positional accuracy of GPS traces can reach 5–30 m, but the error will increase when the satellite signals are obstructed by tall buildings, trees, and tunnel. Therefore, it is necessary to remove the points with large error first. However, it is still hard to generate a lane-level map using this GPS data because some points appear on the opposite road, which will influence the results of the lane position. The aim of this paper is to correct the GPS points to the right location by a physical attraction model based on the OSM map information.

3.2. Trajectory Preprocessing

To remove the noise points with a large error, the trajectory should be preprocessed first. In general, points with fewer neighboring points are identified as outliers. However, as shown in Figure 3, there is a big difference in density between different roads because of the different grades of the roads and the divisions of urban zoning. The density of the points in a city center is larger than that of the points at the edge of a city. If we were to use the same threshold to distinguish the noise in both the high and low density areas, the correct points would be recognized as noise in the areas with low density. Therefore, it is necessary to choose an adaptive density threshold to preprocess the data.

The density of a GPS point can be described by the null distribution [23]. The null distribution of point P can be defined as follows:

$$P(N(S) = k) = \frac{\lambda^k |B|^k}{k!} \cdot e^{-\lambda |B|} \quad (1)$$

$$P(X < n_i) = \sum_{j=0}^{n_i-1} \frac{e^{-\hat{\lambda}} \hat{\lambda}^j}{j!} \quad (2)$$

$$\hat{\lambda} = \frac{N_D}{|D|} \quad (3)$$

$$P(X \geq n_i) = 1 - P(X < n_i) \quad (4)$$

Equation (1) is the expression of the null distribution, where $N(S)$ is the number of points for a spatial point dataset SD , λ is the intensity of the point dataset, and $|B|$ is the area of SD . In this paper, we calculate the density of GPS point P through the number of points in buffer D . As shown in Figure 2, the probability of the number of points $X < n_i$ in a buffer D can be calculated via Equation (2). The value of $\hat{\lambda}$ can be calculated by Equation (3), where N_D is the number of points in buffer D , and $|D|$ is the area of buffer D . Through Equation (4), we can calculate the probability of $X \geq n_i$ in buffer D .

As already noted, on a different road, the density of GPS points is different. Therefore, we used a Delaunay triangulation network to calculate the radius of the buffer:

$$R_S = \text{Mean}(AT) + \text{Variation}(AT) \quad (5)$$

where $\text{Mean}(AT)$ is the average value of the length of the sides in the Delaunay triangulation network and $\text{Variation}(AT)$ is the variance of the lengths of the sides. In the center of the city, the density of points is large, so the value of R_S is small. In contrast, the value of R_S is large at the edge of the city, as shown in Figure 4.

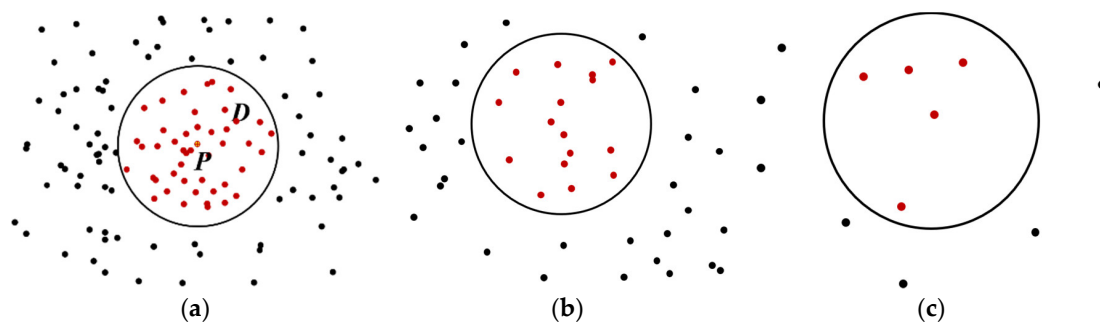


Figure 3. Areas with different point densities. D represents the buffer of central point P . The circle is the range of buffer D . Red dots represent the GPS points inside D , and black dots are the GPS points outside D . (a) The points in the center of a city have more neighboring points; (b) the points out of the center of a city with lower density; and (c) the points at the edge of a city, which might be recognized as noise.

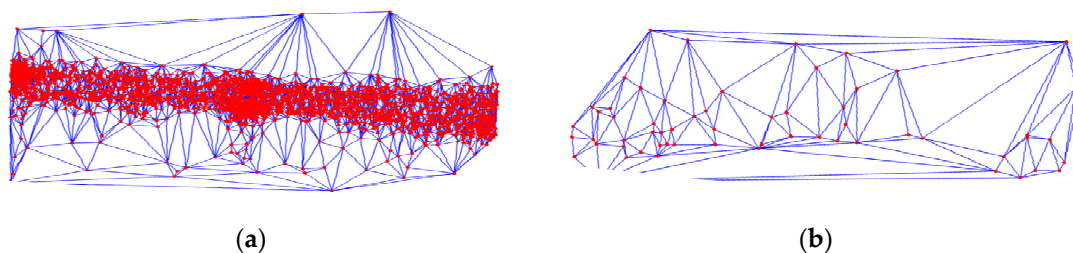


Figure 4. The Delaunay triangulation network of GPS points. The red dots represent the GPS points and the blue lines are the edges of the Delaunay triangulation network. (a) in the center of the city, the length and variance of edges in the Delaunay triangulation network are small, so the value of R_S is small; (b) at the edge of the city, the density of points is small, so the length and variance of the edges in the Delaunay triangulation network is large. As a result, the value of R_S is large.

3.3. Hierarchical Map Matching Algorithm (HST-Matching)

After the preprocessing, the points with large position error have been removed. However, the accuracy of GPS trajectory still cannot meet the requirement of the lane-level map. In this section, we propose hierarchical map matching (HST-Matching) method, by improving the ST-matching algorithm [17], to match low-frequency floating car data with the OSM map. The HST-Matching algorithm consists of two parts: preliminary matching and ST-matching.

The OSM map, as crowdsourced geographic data, is one of the current research trends. The main advantage is its low cost. From the literature [24–26], we know that most of the OSM map data have high accuracy. However, the precision of the GPS points in the floating car measurement is 5–30 m. Therefore, we can attempt to match the GPS points with the OSM map to improve the accuracy of the GPS traces.

The OSM map consists of three parts: nodes, ways, and relations [27]. The nodes can be used to define standalone point features or the shape of a way. The ways are used to represent the linear features, for example, rivers and roads. A way consists of an ordered list of nodes between 2 and 2000 and is defined as a polyline. A relation is used to describe the relationship between two or more data elements, including turn restrictions. The ways in the OSM map not only include the roads, but also rivers, subways, and the boundary river. We only selected the types of roads on which vehicles can drive, including motorway, trunk, primary, secondary, tertiary, service, and residential roads.

A complete road is divided into many segments in the OSM map, $R = \{e_1, e_2, \dots, e_n\}$. Each segment contains a starting point, $e_i.start$, an endpoint, $e_i.end$, and the nodes that control the shape of the road, $e_i.control$, as shown in Figure 5.

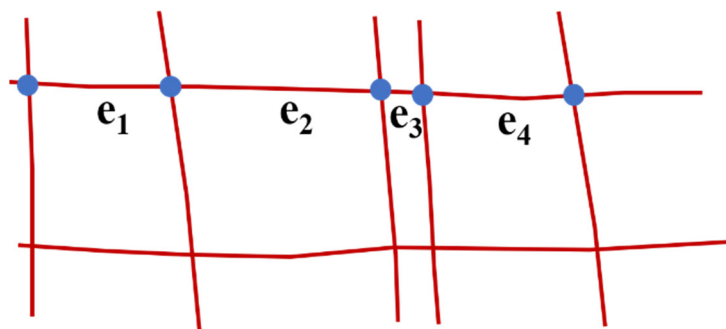


Figure 5. The segments in the OSM map. The red lines represent the roads of OSM, the blue points are the start or end points of segments, and e_i means the segment of road.

Before we introduce the map matching algorithm, it is necessary to describe the assumptions used in this paper.

Assumption 1. *The vehicle runs on the roads, so the trajectory can be matched to at least one road.*

Assumption 2. *The path of the car tends to be direct, rather than a roundabout route. This means that the matching path between two GPS points will most likely be the shortest path.*

3.3.1. Preliminary Matching

In the preliminary matching step, we generated the buffer of each point P_i , $1 \leq i \leq n$, with radius r in a trajectory $T = \{P_1, P_2 \dots P_n\}$ to retrieve the candidate segments and candidate points of P_i . As we had already preprocessed the trajectory, the noises with the largest deviations had been removed. The radius of the buffer was set as 30 m.

An example is shown in Figure 6a. In the buffer of point P_i , there are three candidate segments $P_i = \{e_i^1, e_i^2, e_i^3\}$. The distances between P_i and the candidate segments are $\{d_i^1, d_i^2, d_i^3\}$, and the candidate points of P_i are $\{c_i^1, c_i^2, c_i^3\}$. As azimuth information in floating car data is lacking, we calculate the angle differences between the vector connecting points P_i and P_{i+1} and the candidate segments direction $\{e_i^1, e_i^2, e_i^3\}$. As shown in Figure 6b, the angle differences are $\{\theta_i^1, \theta_i^2, \theta_i^3\}$. We use a threshold T_θ to filter out parts of candidate segments. If $\theta_i^j > T_\theta$, the corresponding segment of angle θ_i^j is removed. If only one candidate segment, e_i^1 , remains, point P_i is counted as a high-confidence tracking point (HCTP) according to Assumption 1 and segment e_i^1 is the matched road of point P_i . Algorithm 1 shows the details of the preliminary matching procedure.

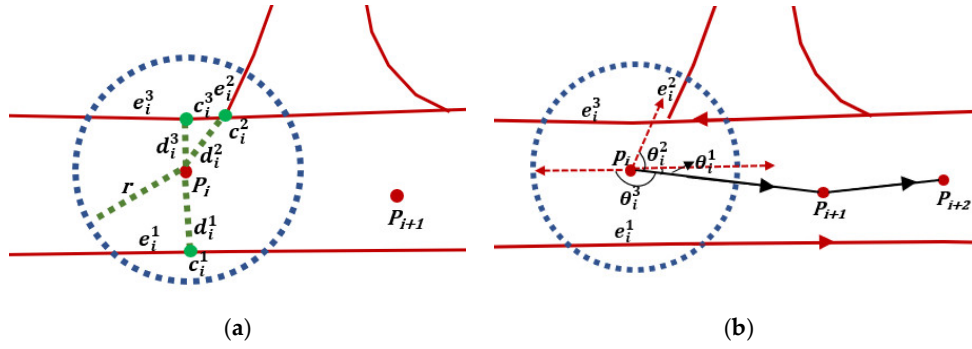


Figure 6. (a) P_i means the ID of GPS point and e_i^j represents the candidate segment of road. d_i^j means the distance between P_i and the candidate segment. c_i^j is ID of the candidate point of P_i . The red lines represent the roads of OSM, and blue dashed lines represent the buffer of center point P_i . Green dashed lines represent the distance between P_i and candidate points. The red dots mean the GPS points, and the green dots are the candidate points of P_i . In the buffer of point P_i , there are three segments with which it intersects. (b) The red dashed lines indicate the direction of the candidate segments; θ_i^j means the angle differences; only segment e_i^1 remains after being filtered out by a threshold.

Algorithm 1 Preliminary Matching Algorithm

Input: Trajectory $P_1 \rightarrow P_2 \dots \rightarrow P_n$; OSM road network R
Output: HCTPlist; Candidate matched points list $c_i^1, c_i^2 \dots c_i^j$

- 1: Initialize HCTPlist and CandidateList as empty list;
- 2: **for** $i = 1$ to n **do**
- 3: $C = \text{GetCandidate}(P_i, R, r)$; // get the candidates within radius r
- 4: **for** $j = 1$ to $C.\text{count}$ **do**
- 5: $\theta_i^j = |\text{azi_}P_i - \text{azi_}c_i^j|$;
- 6: **if** $\theta_i^j < T_\theta$ **then**
- 7: CandidateList.add(c_i^j);
- 8: **end if**
- 9: **end for**
- 10: **if** CandidateList.count == 1 **then**
- 11: HCTPlist.add(P_i);
- 12: **end if**
- 13: **end for**

If a point was counted as high-confidence tracking point (HCTP), we only retained the candidate segment which meets the requirement of threshold T_θ ; other candidate segments will be deleted. There is no need to calculate further. This can reduce the uncertainty and running time of the matching algorithm. However, for other points which do not count as HCTPs, we calculated it by the ST-matching algorithm.

3.3.2. Spatial–Temporal Matching

After the preliminary matching step, some points will have matched with the OSM map. However, there will be many points left to be matched. We choose an ST-matching algorithm to match these points [17,28]. ST-matching is a stable global optimization matching algorithm which can integrate the geometrical, topological, and speed information of traces and map. It includes two steps: spatial analysis and temporal analysis. First, the observation probability is calculated by identifying the shortest distance between the GPS points and the candidate points. Then, the transmission probability is estimated by comparing the shortest path of the GPS points and the candidate points. The temporal probability is calculated by the cosine distance to measure the similarity between the actual speed

of the path and the road speed constraints. Finally, the candidate segment with the largest value is considered to be the final matching result.

We improve this algorithm in three ways. In addition to the HCTPs, we give different weights to the observation and transmission probabilities. Considering that the shortest path is more reliable than the position accuracy of GPS. Therefore, the transmission probability has a higher reliability than the observation probability. Its weight should be set to be larger than the observation probability. Beyond this, the road speed constraint should be a range rather than a simple value. Therefore, we use a new method to calculate the temporal value. This is calculated by comparing the probability of the actual speed and the road speed constraint range.

Spatial Analysis

According to Reference [9], the position error of the GPS can be described with a normal distribution, and the formula of the observation probability can be calculated via:

$$O(c_i^j) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(d_i^j - \mu)^2}{2\sigma^2}} \quad (6)$$

where d_i^j is the distance between point P_i and candidate segment c_i^j , $d_i^j = \text{dist}(c_i^j, P_i)$, and μ and σ are the mean and variance value of normal distribution.

Because of the position error of the GPS, it is not enough to only consider the Euclidean distance between the GPS trace and the segment. For example, as shown in Figure 7, there are two candidate points for P_i , (c_i^1, c_i^2) . The observation probabilities of these two candidate points are equal. Obviously, the correctly matched point should be c_i^2 according to Assumption 2. Hence, topological information is important for map matching, by which we can exclude certain points. The formula of the transmission probability is shown in Equation (7):

$$T(c_{i-1}^k \rightarrow c_i^j) = \frac{\text{dis}(P_{i-1}, P_i)}{S_{(c_{i-1}^k \rightarrow c_i^j)}} \quad (7)$$

where $\text{dis}(P_{i-1}, P_i)$ is the Euclidean distance between tracking points P_{i-1} and P_i and $S_{(c_{i-1}^k \rightarrow c_i^j)}$ represents the shortest path between candidate points c_{i-1}^k and c_i^j . There are many algorithms for the shortest path computation, such as the Dijkstra, Floyd, and A* algorithms [29]. Considering the efficiency of the algorithms, we choose the A* algorithm to calculate the shortest path.

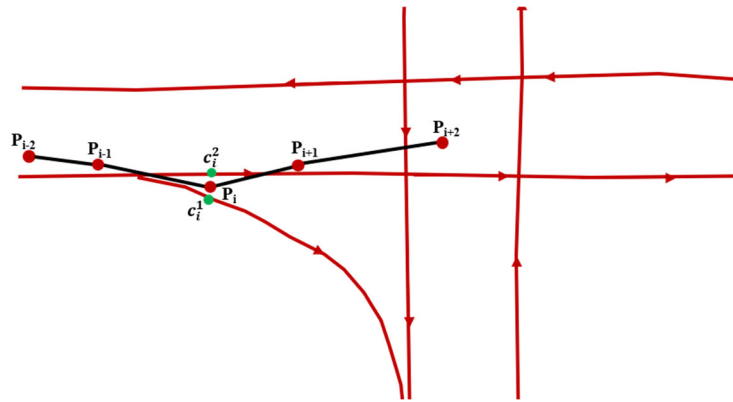


Figure 7. An example of the transmission probability. P_i means the ID of GPS point and c_i^j represent the ID of candidate point. Red dots mean the GPS points of the trajectory and the black lines are the trajectory of floating car. The red lines represent the OSM roads and the green dots indicate the candidate points of P_i .

Combining Equations (6) and (7), the spatial analysis function is calculated as follows:

$$F_s(c_{i-1}^k \rightarrow c_i^j) = w_1 O(c_i^j) + w_2 T(c_{i-1}^k \rightarrow c_i^j), 2 \leq i \leq n \quad (8)$$

where w_1 and w_2 represent the weight of the observation and the transmission probabilities, respectively, and $w_1 + w_2 = 1$.

Temporal Analysis

The spatial analysis can match the GPS trace to the OSM map in most cases. However, there are lots of elevated roads in China, and the shape and location of the elevated roads are similar to the roads underneath them. Therefore, it is difficult to match the GPS trace using only spatial analysis. However, as shown in Table 1, the road speed constraints are different for the elevated roads than for other roads. Therefore, it is feasible to use the road speed constraint information to refine our analysis. We calculate the probability of the actual speed of a vehicle in different road speed constraint ranges, as shown in the equations below:

$$F_t(c_{i-1}^k \rightarrow c_i^j) = \int_{\bar{V}.min}^{\bar{V}.max} \frac{1}{\sqrt{2\pi}R.\sigma} e^{-\frac{(\bar{V}(c_{i-1}^k \rightarrow c_i^j) - R.u)^2}{2R.\sigma^2}} \quad (9)$$

$$\bar{V}(c_{i-1}^k \rightarrow c_i^j) = [\bar{V}.min, \bar{V}.max] \quad (10)$$

$$\bar{V}.min = \frac{S_{(c_{i-1}^k \rightarrow c_i^j)}}{\Delta t_{(i-1 \rightarrow i)}} - \tau \quad (11)$$

$$\bar{V}.max = \frac{S_{(c_{i-1}^k \rightarrow c_i^j)}}{\Delta t_{(i-1 \rightarrow i)}} + \tau \quad (12)$$

where $R.u$ and $R.\sigma$ represent the mean value and variance of the candidate road speed, respectively, and $\bar{V}(c_{i-1}^k \rightarrow c_i^j)$ is the average speed of the car from point c_{i-1}^k to c_i^j , $\Delta t_{(i-1 \rightarrow i)}$ is the time interval from point c_{i-1}^k to c_i^j , and τ is the calculation error of the speed.

Table 1. The road speed constraint ranges in China.

Value	Motorway	Trunk	Primary	Secondary	Tertiary	Service	Residential
Min-speed (km/h)	90	60	40	30	20	0	0
Max-speed (km/h)	120	100	60	50	40	20	15

Combining Equations (8) and (9), the final ST-matching function is:

$$F(c_{i-1}^k \rightarrow c_i^j) = F_s(c_{i-1}^k \rightarrow c_i^j) \times F_t(c_{i-1}^k \rightarrow c_i^j) 2 \leq i \leq n. \quad (13)$$

After the HST-matching steps, the value of $F(c_{i-1}^k \rightarrow c_i^j)$ should be calculated for each candidate point. We select the highest score as the matching result between two HCTPs, as shown by the red line in Figure 8. Algorithm 2 shows the process of ST-matching.

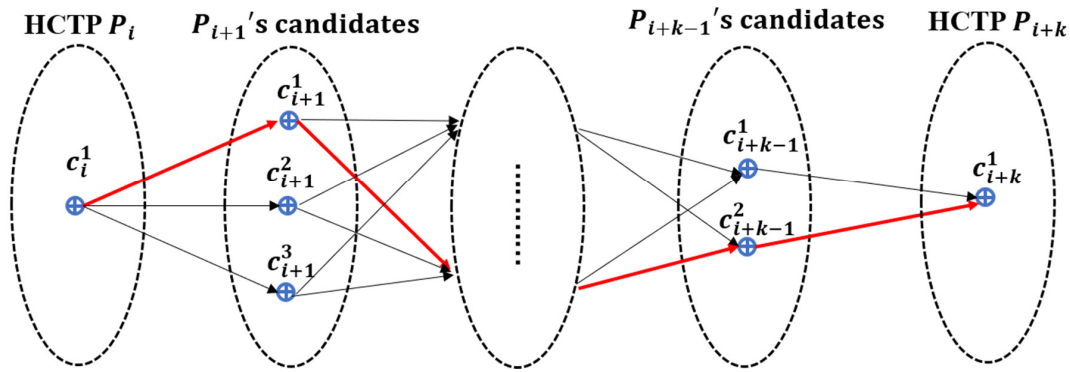


Figure 8. The matching result of the ST-algorithm. P_i means the ID of GPS point and c_i^j is the ID of candidate point. The points P_i and P_{i+k} are the HCTP points, so there is only one candidate point. The black arrows represent the candidate matching path from P_i to P_{i+1} , and the red arrows represent the final matching results.

Algorithm 2 Spatial and Temporal Matching Algorithm

Input: HCTPlist P_i, P_{i+k} ; CandidateList $c_i^1, c_i^2 \dots c_i^j$; Trajectory $P_{i+1} \rightarrow P_{i+2} \dots \rightarrow P_{i+k}$
Output: OSM-WayID-List;
1: Initialize OSM-WayID-List as empty list;
2: **for** each c_i^1 and c_{i+k}^1 **do**
3: $F(c_i^1) = 1$;
4: $F(c_{i+k}^1) = 1$;
5: **end for**
6: **for** $t = i + 1$ to $i + k - 1$ **do**
7: $\max = -\infty$;
8: **for** $s = 1$ to $\text{candidateList}(P_t).\text{count}$ **do**
9: $F(c_t^s) = F(c_{t-1}^j) + F(c_{t-1}^j \rightarrow c_t^s)$;
10: $\text{Alt} = F(c_t^s)$;
11: **if** ($\text{Alt} > \max$) **then**
12: $\max = \text{Alt}$;
13: $C = \max. c_t^s$;
14: **end if**
15: **end for**
16: OSM-WayID-List.add($C.\text{id}$);
17: **end for**

3.4. Trajectory Correction Algorithm

The accuracy of the trajectory improves after the optimization approach; however, it is not enough for the accurate generation of a lane-level map. As shown in Figure 9, the red points are matched to the yellow road which should locate at the road from right to left. However, some points appear on the opposite road because of the multipath effect, which will decrease the accuracy for generating the lane-level map. Thus, to address this issue, this paper proposes a physical attraction model based on the matched OSM map.

According to [7,30], in the physical attraction model two types of forces act on the trajectories. One is an attractive force from the other traces in the same direction and on the same road. The other is a spring force to prevent the trace from moving away from its original position, as shown in Figure 10. All the traces in the same direction will be grouped together by these forces. The accuracy of this approach can reach the road level, but the original information of the trace is lost. Moreover, this approach makes mistakes at crossings, and it is time-consuming to calculate the distance between one

trace and all the other traces. Thus, this paper introduces the matched OSM map algorithm to address these concerns.

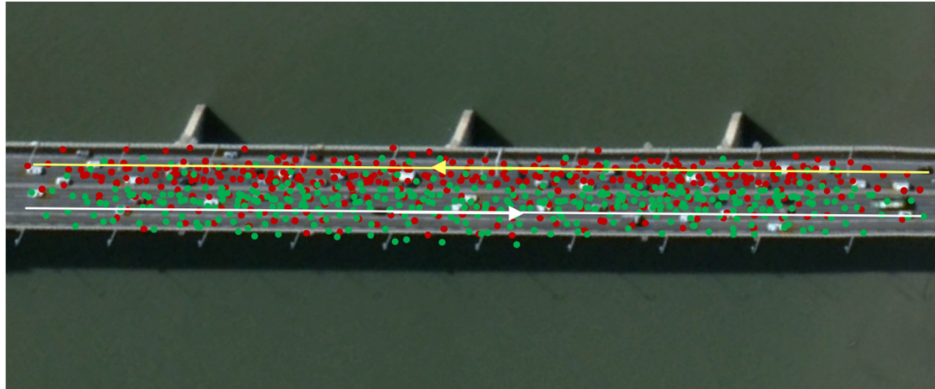


Figure 9. An example of the multipath effect. The yellow line is the OSM road from right to left and the white line is the OSM road from left to right. The red dots are matched to the yellow road and the green dots are matched to the white road. Some points appear on the wrong road because of the GPS error.

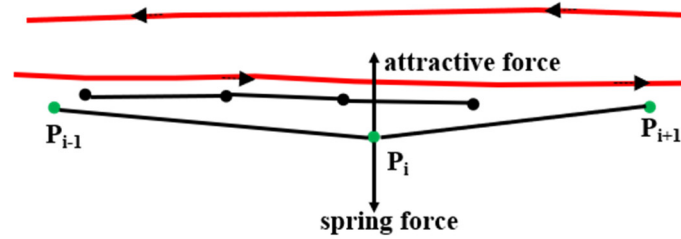


Figure 10. Two types of forces acting on the trajectories. The green dots are the GPS points on a trajectory and the black dots are the points on another trace.

In general, when a car drives on the road, it tends to keep driving in the same lane, unless an emergency occurs or it drives to an intersection. Therefore, the traces tend to keep the same distance with the matched OSM map. We calculated the attractive force by the relative distance between the trace and OSM road. Compared to calculating the distance between one trace and all the other traces, this method can greatly decrease the running time of the algorithm. At the intersection, our method is more reliable because the OSM map constrains the direction of the force. The equations are shown below:

$$F_1(P_i) = -\frac{M}{\sigma^3\sqrt{2\pi}}e^{-\frac{(d_i-\bar{d})^2}{2\sigma^2}}(d_i-\bar{d}) \quad (14)$$

$$F_2(P_i) = k(y-d_i) \quad (15)$$

$$F_1(P_i) = F_2(P_i) \quad (16)$$

$$\bar{d} = \frac{\sum_{i=1}^n d_i}{n}, \quad (17)$$

where M and σ are the two experimental parameters that determine the potential energy of the attractive force, K is the spring constant, d_i is the distance from P_i to the matched OSM map points, \bar{d} is the average distance of d_i , and $(y-d_i)$ is the difference of the distance between the new and the original position of point P_i in order to keep $F_1(P_i)$ equal to $F_2(P_i)$. As shown in Figure 11, we set the direction of the OSM road as the x-axis and the left side of the road as the y-axis. P'_i is the new position of P_i . The details are shown in Algorithm 3.

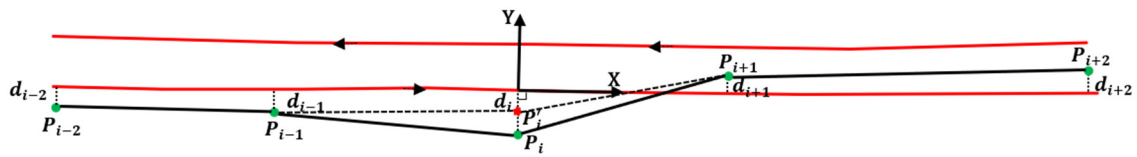


Figure 11. An example of the physical attraction model for point P_i . P_i means the ID of GPS points, and d_i represents the distance from GPS points to the matched OSM road. P'_i is the new position of P_i . The green dots are the GPS points, and the red dot is the new point. The red lines are the OSM roads, the black lines represent the original trajectory, and the black dashed lines indicate the new trajectory.

Algorithm 3 Physical Attraction Model

Input: Trajectory $P_1 \rightarrow P_2 \rightarrow \dots \rightarrow P_n$; OSM-WayID-List;
Output: New Trajectory $P'_1 \rightarrow P'_2 \rightarrow \dots \rightarrow P'_n$;

- 1: **for** $t = 1$ to n **do**
- 2: $T = 0$; $K = \infty$;
- 3: $\bar{d} = \text{meandistance}(d_1, d_2, \dots, d_n)$;
- 4: $K = d_i - \bar{d}$;
- 5: **While** $T \leq 20$ && $K > 0.5$ **do**
- 6: $F_1(P_i) = F_2(P_i)$;
- 7: $\bar{d}' = \text{meandistance}()$;
- 8: $K = d'_i - \bar{d}'$;
- 9: $T = T + 1$;
- 10: **end while**
- 11: **end for**

4. Experimental Tests of the Proposed Approach

4.1. Experimental Data

To test the algorithms proposed in this paper, we collected about 40 million GPS points from thousands of taxis within one week in Wuhan, China. The sampling frequency ranged from 1 s to 10 min, as shown in Figure 12. Over 60% of the points are included in the sampling frequency range of 1–40 s.

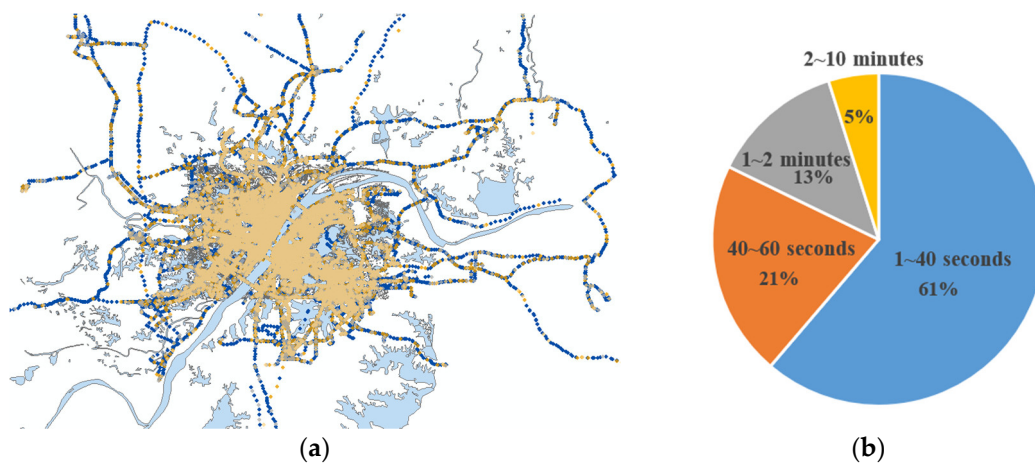


Figure 12. The experimental data. (a) GPS points collected from taxis; the yellow dots represent the GPS points whose sampling frequency ranged from 2–10 min; the gray dots are the points who sampling frequency ranged from 1–2 min; the orange dots are the points whose sampling frequency ranged from 40–60 s; and the blue dots are the points whose sampling frequency ranged from 1–40 s. (b) Distribution of the sampling frequencies.

4.2. Trajectory Preprocessing

The trajectory preprocessing algorithm is an adaptive density optimization method. The result is shown in Figure 13. The red dots and black dots represent the selected GPS points and outliers, respectively; the noise points that are removed by this method.

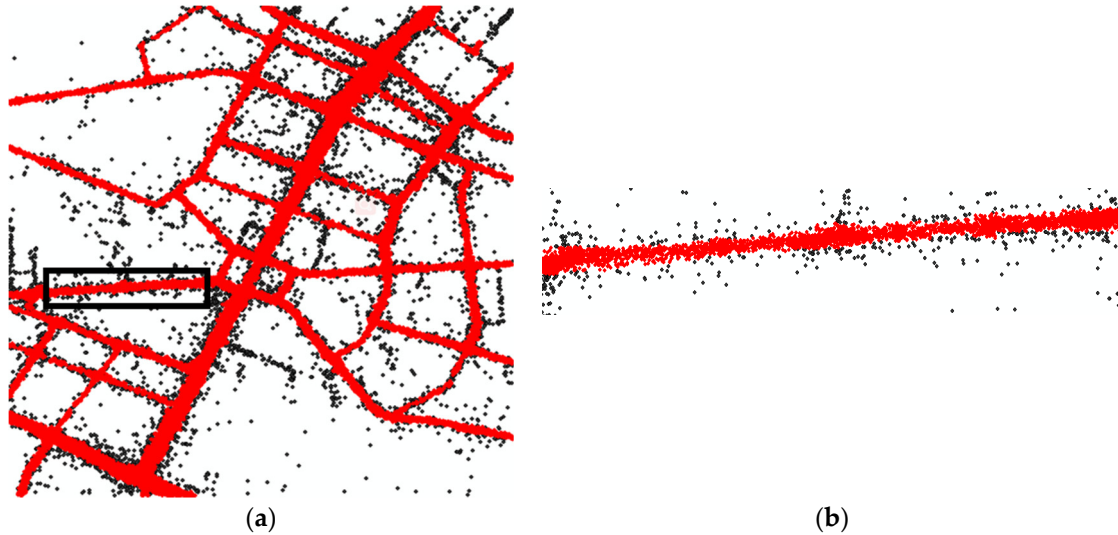


Figure 13. The results of preprocessing. The red dots are the preserved GPS points and the black dots are noise points (a) An example on a large scale. (b) The detailed results of the black rectangle in (a).

4.3. Map Matching

The matching data was labeled by real people. Compared to the synthetic trajectory data used in reference [17], it is more reliable. The labeled data contains 34 traces covering about 494 km, as represented in Figure 14.



Figure 14. The labeled trajectories. The teal lines represent the labeled traces.

4.3.1. Evaluation Approach

To evaluate the matching quality, we calculated the accuracy and recall, as shown in the following equations:

$$accuracy = \frac{\text{Number of correctly matched points}}{\text{Total number of matched points}} \times 100\% \quad (18)$$

$$recall = \frac{\text{Number of correctly matched points}}{\text{Number of relevant points in labeled points}} \times 100\% \quad (19)$$

4.3.2. Parameter Selection

In Section 3.3.1, we used the threshold T_θ to screen out the candidate segments. The accuracy and recall are different, depending on the choice of T_θ , as shown in Figure 15. The recall of the HCTPs is about 86–92%, and the points matched as HCTP do not need to be matched further. This reduces the running time of the algorithm. The accuracy of HCTP is about 90%. This means that Assumption 1, as proposed above, is dependable. According to these results, when we set $T_\theta = 90^\circ$ the accuracy and recall reach their maximums.

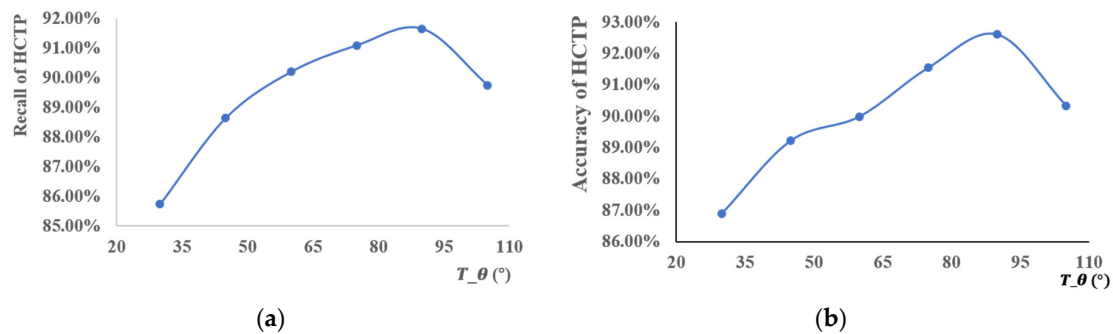


Figure 15. The (a) recall and (b) accuracy of map matching for different T_θ . The blue lines represent the value of recall and accuracy of HCTP for different T_θ .

In the ST-matching step, the parameters included the following: μ , σ , w_1 , w_2 , τ , $R.\sigma$, $R.u$. μ and σ are the mean value and variance of the normal distribution, respectively. In this paper, we set $\mu = 0$ and $\sigma = 10$. w_1 and w_2 are the weight of the observation and the transmission probabilities, respectively. As explained above, the transmission probability is more reliable than the observation probability; therefore, we set $w_1 = 0.3$ and $w_2 = 0.7$. The value of τ was 10 km/h. $R.u$ and $R.\sigma$ are the mean value and variance of the road speed constraints, respectively. The values for the different roads are shown in Table 2.

Table 2. The mean ($R.u$) and variance ($R.\sigma$) of the road speed constraints for the different roads.

Threshold	Motorway	Trunk	Primary	Secondary	Tertiary	Service	Residential
$R.u$	105	80	50	40	30	10	10
$R.\sigma$	5	7	3	3	3	3	1.5

4.3.3. Matching Result

We compared the HST-matching results with those of the ST-matching algorithm. To evaluate the quality and efficiency of the two algorithms, we compared the accuracy and running time. Figure 16 represents the accuracy comparison results. When the number of points on a trajectory was in the range 5–15, the HST-matching algorithm significantly outperformed the ST-matching algorithm; the accuracy of the HST-matching algorithm showed about a 15% improvement. With an increasing number of points in a trajectory, the performance of these two algorithms became more similar, but the HST-matching algorithm still showed about an 8% improvement over the ST-matching algorithm.

4.3.4. Running Time

As shown in Figure 17, it is clear that the HST-matching algorithm is faster than the ST-matching algorithm, especially when the number of points is in the range 5–15. This is because we calculate the HCTPs first. This method greatly reduces the number of unnecessary calculations, especially when there are fewer points on the trajectory. As the number of points increases, the time cost of the two methods increases quickly and tends to become more similar. This is because the algorithms need more time to calculate the shortest path with an increased number of points.

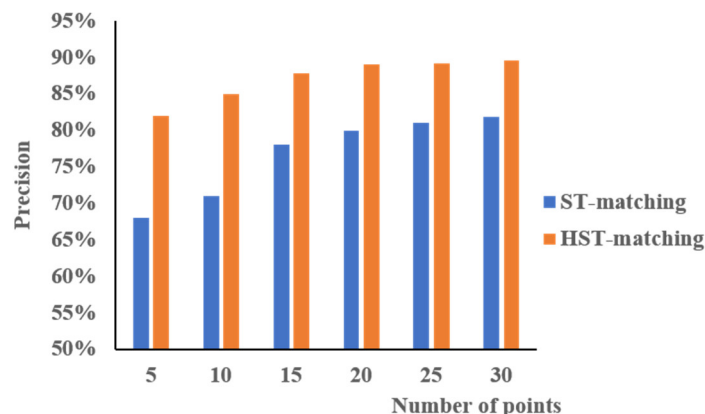


Figure 16. The accuracy of map matching using spatial-temporal (ST)-matching (blue) vs. hierarchical ST (HST)-matching (orange).

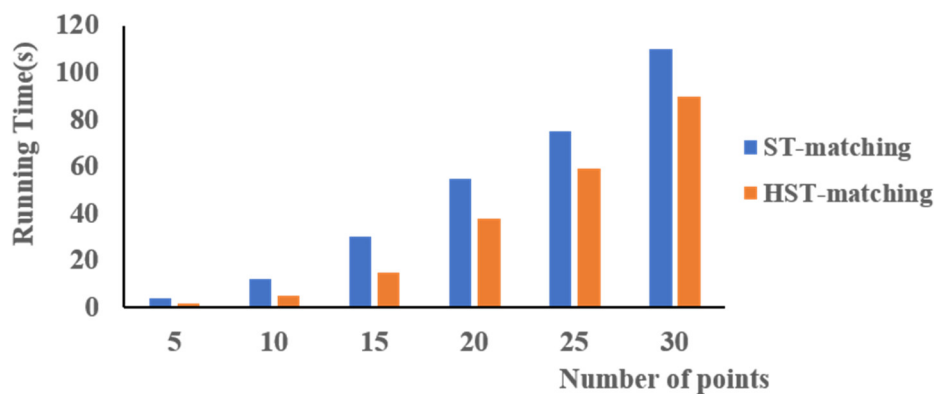


Figure 17. Running time of the map matching using ST-matching (blue) vs. HST-matching (orange).

4.4. Data Correction

4.4.1. Parameter Selection

In this part, there are three main parameters that need to be set: M , σ , and k . According to reference [30], we set $M = 1$, $\sigma = 10$, and $k = 0.005$.

4.4.2. Correction Result

Figure 18 shows the original data. The data is messy, and there are many points that appear on the wrong side of the road, against the traffic regulations. After the data correction algorithm proposed in this paper is used, the position accuracy of the data is improved. The trajectory no longer appears in the opposite lane; the points are corrected to the right position, as shown in Figure 19.

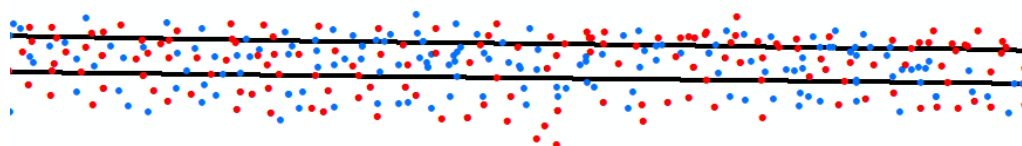


Figure 18. Original data from a road. The black lines represent the OSM road, the red dots represent the points matched to the upper black line, and the green dots are the points matched to the lower line. The points from different directions are mixed together.

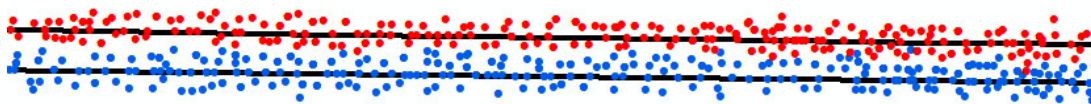


Figure 19. The results of the correction algorithm represented in this paper. The points from different directions separate well.

The algorithm proposed in reference [30] groups the traces with the same direction together; the gap between them is less than 0.5 m, as represented in Figure 20. By this approach, the accuracy of the data can only reach the road level, beyond which a lane-level map cannot be generated. Additionally, original information of the floating car traces is lost. Moreover, there are still some mistakes at the intersections when using this approach, as shown in Figure 21b; there are some incorrect edges that need to be removed. In contrast, in our method points are previously matched to the OSM, so they can be correctly clustered especially at the intersection, as shown in Figure 21a.



Figure 20. The results of the algorithm represented in reference [30]. The data with the same direction clustered together, and the maximum gap was 0.5 m.

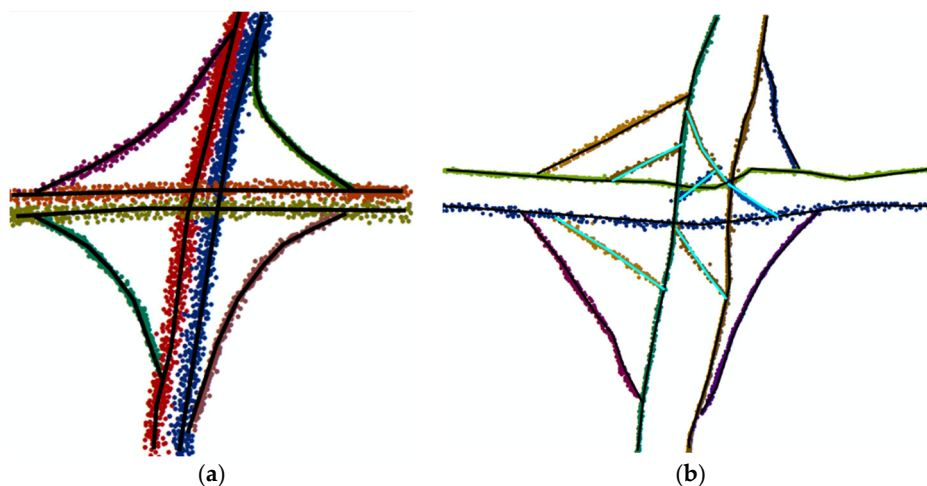


Figure 21. An example of an intersection. (a) The result of the algorithm represented in this paper. Various colors represent different directions of GPS points. (b) The result of the algorithm represented in reference [30], in which there are some incorrect edges as represented by the teal lines.

The time complexity of the algorithm in reference [30] is (M^2) , where M is the number of nodes in the GPS dataset. For each node, a dataset was a square (100×100 m) centered at the node. It took at least 15 s to calculate the data for each node. However, the time complexity of the algorithm proposed in this paper is (M) . Our algorithm only needs 150 ms to calculate the data for each point—a marked improvement on previous algorithms.

5. Conclusions

In this paper, we proposed a data correction algorithm for low-frequency floating car data. After preprocessing the data, we employed an HST-matching algorithm to match the GPS trajectories with the OSM map. The accuracy and running time of this algorithm were compared with those of the ST-matching algorithm. The accuracy of the HST-matching algorithm was higher; the accuracy of the HST-matching algorithm was always 8–15% higher than that of the ST-matching algorithm. Moreover, we needed less time to calculate the results because we adopted a hierarchical algorithm to calculate

the HCTP first. Next, the data was corrected by a physical attraction model based on the matched OSM map. A verification experiment was conducted based on the data of actual taxi trajectories. The results showed that the accuracy of the data after the correction was improved, especially at the crossroads. Moreover, we improved the time efficiency by 150 times.

This paper proved that OSM can be used to improve the accuracy of low-floating car data. This study was also useful for increasing the precision of the production of lane-level maps, which were generated by the corrected data.

However, although we greatly improved the time efficiency, it still took a long time to calculate all the data because of the huge quantity of floating car data. In the future, we will continue to improve the calculation efficiency of this algorithm and to research the production of lane-level maps.

Author Contributions: Methodology, Y.G.; Software, J.Z.; Validation, Y.G., Y.C., and B.L.; Resources, B.L.; Original draft preparation, Y.G.; and review and editing of manuscript, Y.G.

Funding: This research was funded by the National Natural Science Foundation of China (41671441, 41531177, U1764262).

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Gwon, G.P.; Hur, W.S.; Kim, S.W.; Seo, S.W. Generation of a Precise and Efficient Lane-Level Road Map for Intelligent Vehicle Systems. *IEEE Trans. Veh. Technol.* **2017**, *66*, 4517–4533. [\[CrossRef\]](#)
2. Li, Y.; Hua, L.; Tan, J.; Zan, L.; Hong, X.; Chen, C. Scan Line Based Road Marking Extraction from Mobile LiDAR Point Clouds. *Sensors* **2016**, *16*, 903. [\[CrossRef\]](#)
3. Guo, C.; Kidono, K.; Meguro, J.; Kojima, Y.; Ogawa, M.; Naito, T. A Low-Cost Solution for Automatic Lane-Level Map Generation Using Conventional In-Car Sensors. *IEEE Trans. Intell. Transp. Syst.* **2016**, *17*, 2355–2366. [\[CrossRef\]](#)
4. Tang, L.; Yang, X.; Kan, Z.; Li, Q. Lane-Level Road Information Mining from Vehicle GPS Trajectories Based on Naïve Bayesian Classification. *ISPRS Int. J. Geo-Inf.* **2015**, *4*, 2660–2680. [\[CrossRef\]](#)
5. Tang, L.; Yang, X.; Dong, Z.; Li, Q. CLRIC: Collecting Lane-Based Road Information via Crowdsourcing. *IEEE Trans. Intell. Transp. Syst.* **2016**, *17*, 2552–2562. [\[CrossRef\]](#)
6. Li, J.; Qin, Q.; Xie, C.; Zhao, Y. Integrated use of spatial and semantic relationships for extracting road networks from floating car data. *Int. J. Appl. Earth Obs. Geoinf.* **2012**, *19*, 238–247. [\[CrossRef\]](#)
7. Wang, J.; Rui, X.; Song, X.; Tan, X. A novel approach for generating routable road maps from vehicle GPS traces. *Int. J. Geogr. Inf. Syst.* **2015**, *29*, 69–91. [\[CrossRef\]](#)
8. Liu, X.; Biagioni, J.; Eriksson, J.; Wang, Y.; Forman, G.; Zhu, Y. Mining large-scale, sparse GPS traces for map inference: Comparison of approaches. In Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Beijing, China, 12–16 August 2012; pp. 669–677.
9. Biagioni, J.; Eriksson, J. Map inference in the face of noise and disparity. In Proceedings of the International Conference on Advances in Geographic Information Systems, Redondo Beach, CA, USA, 6–9 November 2012; pp. 79–88.
10. Worrall, S.; Nebot, E. *Automated Process for Generating Digitised Maps through GPS Data Compression*; University of Sydney: Sydney, Australia, 2007.
11. Schroedl, S.; Wagstaff, K.; Rogers, S.; Langley, P.; Wilson, C. Mining GPS Traces for Map Refinement. *Data Min. Knowl. Discov.* **2004**, *9*, 59–87. [\[CrossRef\]](#)
12. Lee, J.G.; Han, J.; Whang, K.Y. Trajectory clustering: a partition-and-group framework. In Proceedings of the ACM SIGMOD International Conference on Management of Data, Beijing, China, 12–14 June 2007; pp. 593–604.
13. Lee, W.-C.; Krumm, J. Trajectory Preprocessing. In *Computing with Spatial Trajectories*; Springer: New York, NY, USA, 2011; pp. 3–33, ISBN 978-1-4614-1628-9.
14. Fox, D. Adapting the Sample Size in Particle Filters Through KLD-Sampling, Adapting the Sample Size in Particle Filters Through KLD-Sampling. *Int. J. Robot. Res.* **2003**, *22*, 985–1003. [\[CrossRef\]](#)

15. Hightower, J.; Borriello, G. Particle Filters for Location Estimation in Ubiquitous Computing: A Case Study. In *UbiComp 2004: Ubiquitous Computing; Lecture Notes in Computer Science*; Springer: Berlin/Heidelberg, Germany, 2004; pp. 88–106.
16. Murphy, K.; Russell, S. Rao-Blackwellised Particle Filtering for Dynamic Bayesian Networks. In *Sequential Monte Carlo Methods in Practice*; Statistics for Engineering and Information Science; Springer: New York, NY, USA, 2001; pp. 499–515, ISBN 978-1-4419-2887-0.
17. Lou, Y.; Zhang, C.; Zheng, Y.; Xie, X.; Wang, W.; Huang, Y. Map-matching for Low-sampling-rate GPS Trajectories. In Proceedings of the 17th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems, Seattle, WA, USA, 4–6 November 2009; ACM: New York, NY, USA, 2009; pp. 352–361.
18. Greenfeld, J.S. Matching GPS Observations to Locations on a Digital Map. In Proceedings of the 81th Annual Meeting of the Transportation Research Board, Washington, DC, USA, 14 January 2002.
19. Qingquan, L.I.; Lian, H. A Map Matching Algorithm for GPS Tracking Data. *Acta Geod. Cartogr. Sin.* **2010**, *39*, 207–212.
20. Zhe, Z.; Qingquan, L.I.; Zou, H.; Wan, J.; University, S.; University, W. Curvature Integration Constrained Map Matching Method for GPS Floating Car Data. *Acta Geod. Cartogr. Sin.* **2015**, *44*, 1167–1176.
21. Marchal, F.; Hackney, J.; Axhausen, K. Efficient Map Matching of Large Global Positioning System Data Sets: Tests on Speed-Monitoring Experiment in Zürich. *Trans. Res. Rec. J. Transp. Res. Board* **2005**, *1935*, 93–100. [[CrossRef](#)]
22. Zhang, L.; Thiemann, F.; Sester, M. Integration of GPS traces with road map. In Proceedings of the International Workshop on Computational Transportation Science, San Jose, CA, USA, 2 November 2010; pp. 17–22.
23. Liu, Q.; Tang, J.; Deng, M.; Shi, Y. An Iterative Detection and Removal Method for Detecting Spatial Clusters of Different Densities. *Trans. GIS* **2015**, *19*, 82–106. [[CrossRef](#)]
24. Barron, C.; Neis, P.; Zipf, A. A Comprehensive Framework for Intrinsic OpenStreetMap Quality Analysis. *Trans. GIS* **2015**, *18*, 877–895. [[CrossRef](#)]
25. Zhang, H.; Malczewski, J. Accuracy Evaluation of the Canadian OpenStreetMap Road Networks. *Int. J. Geospat. Environ. Res.* **2017**, *5*, 347.
26. Wang, M.; Li, Q.; Hu, Q.; Zhou, M. Quality Analysis of Open Street Map Data. *ISPRS Int. Arch. Photogramm. Remote Sens. Spat. Inf. Sci.* **2013**, *XL-2/W1*, 155–158. [[CrossRef](#)]
27. OpenStreetMap Wiki. Available online: https://wiki.openstreetmap.org/wiki/Main_Page (accessed on 7 September 2018).
28. Yuan, J.; Zheng, Y.; Zhang, C.; Xie, X.; Sun, G.Z. An Interactive-Voting Based Map Matching Algorithm. In Proceedings of the Eleventh International Conference on Mobile Data Management, Kansas City, MI, USA, 23–26 May 2010; pp. 517–520.
29. Fu, L.; Sun, D.; Rilett, L.R. Heuristic shortest path algorithms for transportation applications: State of the art. *Comput. Oper. Res.* **2006**, *33*, 3324–3343. [[CrossRef](#)]
30. Cao, L.; Krumm, J. From GPS traces to a routable road map. In Proceedings of the 17th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems, Seattle, WA, USA, 4–6 November 2009; pp. 3–12.

