International Journal of
Geo-Information

Article

Low-Frequency Trajectory Map Matching Method Based on Vehicle Heading Segmentation

Qingying Yu [1,2], Fan Hu [1,2], Chuanming Chen [1,2,*], Liping Sun [1,2] and Xiaoyao Zheng [1,2]

[1] School of Computer and Information, Anhui Normal University, Wuhu 241002, China; ahnuyuq@ahnu.edu.cn (Q.Y.); hufan@ahnu.edu.cn (F.H.); slp620@ahnu.edu.cn (L.S.); zxiaoyao@ahnu.edu.cn (X.Z.)

[2] Anhui Provincial Key Laboratory of Network and Information Security, Anhui Normal University, Wuhu 241002, China

* Correspondence: ccm1981@ahnu.edu.cn

Abstract: Numerous Global Positioning System connected vehicles are collecting extensive data remotely in cities, enabling data-driven infrastructure planning. To truly benefit from this emerging technology, it is important to combine telematics and map data to make it easier to extract and mine useful information from the data. By performing map matching, data points that cannot be accurately located on the road network can be projected onto the correct road segment. As an important means of remote data processing, it has become an important pre-processing step in the field of data mining. However, due to the various errors of location devices and the complexity of road networks, map matching technology also faces great challenges. In order to improve the efficiency and accuracy of the map matching algorithm, this study proposes an offline method for low-frequency trajectory data map matching based on vehicle trajectory segmentation. First, the trajectory is segmented based on the vehicle's travel direction. Then, the comprehensive probability of the corresponding road segment is calculated based on the spatial probability and the directional probability of each road segment around the location. Third, the *k* candidate matching paths under consideration are selected based on the comprehensive probability evaluation. Finally, the shortest path planning and the probability calculation of the different candidate paths are combined to find the optimal matching path. The experimental results on the real trajectory dataset in Shanghai and the road network environment show that the proposed algorithm has better accuracy, efficiency, and robustness than other algorithms.

Keywords: low-frequency trajectory; map matching; comprehensive probability; shortest-path planning

check for updates

Citation: Yu, Q.; Hu, F.; Chen, C.; Sun, L.; Zheng, X. Low-Frequency Trajectory Map Matching Method Based on Vehicle Heading Segmentation. *ISPRS Int. J. Geo-Inf.* **2022**, *11*, 355. https://doi.org/10.3390/ijgi11070355

Academic Editors: Wolfgang Kainz and Hartwig H. Hochmair

Received: 23 April 2022
Accepted: 20 June 2022
Published: 23 June 2022

1. Introduction

In recent years, the global positioning system (GPS) has been widely used in vehicles, which will facilitate the collection of a large amount of trajectory data by city management and traffic services to reflect information such as vehicle position, speed, and direction. In intelligent transportation systems, these GPS trajectory data are usually integrated with geographic information system (GIS) to provide various services such as vehicle navigation, tracking, and mapping of new roads to individuals or businesses for GIS [1–6]. In the trajectory sampling process, objects such as trees and buildings affect the accuracy of the trajectory data, so the acquired trajectory points may not fall accurately on the road network. Therefore, an important step in integration with geographic information is to match the trajectory points with the road network to determine the road segment of the vehicle's travel path and the location of the vehicle on the road network, i.e., map matching. If the GPS trajectory data and the road network data are accurate enough, effective map matching is easy to achieve. However, due to the complex structure of the road network, more accurate and efficient map matching methods are required to overcome the difficulties and challenges in map matching.

According to the different sampling frequency of trajectory points, we can distinguish between high-frequency methods for trajectory map matching and low-frequency methods for trajectory map matching. Using high-frequency trajectory data, road segments matched by the location points can be better found, resulting in higher-precision matching results. However, due to the strict limitations of the battery, the GPS-embedded handheld device cannot collect high-frequency trajectory data for long periods. Therefore, map matching for low-frequency trajectory data becomes an important issue. Achieving low-frequency trajectory map matching in a complex urban road network is the subject of this study. Existing map matching methods for low-frequency trajectory data mainly include the following three categories.

(1) Methods based on machine learning

The methods based on machine learning mainly include Kalman filter and artificial neural network techniques [7]. Toledo-Moreo et al. [8] proposed a particle filter-based algorithm that combines measurements from global navigation satellite system (GNSS) receivers, gyroscopes, and orometers to solve the map matching problem for parallel roads. Similarly, Szottka et al. [9] proposed a particle filter-based algorithm, using camera detection data with road network markers and commercial map data. Tao et al. [10] developed a Kalman filter-based localization solver for map matching using GPS data, vehicle data, camera observations, and road network marker data embedded in digital navigation maps. Gu et al. [11] combined multisensor measurements and 3D mapping techniques to develop a robust localization system in urban canyons. The 3D map is used to perform signal transmission and track positions to correct vehicle positioning. Shunsuke et al. [12] proposed a particle-filtered vehicle positioning system for autonomous driving that integrates global navigation system, an inertial navigation system, and camera observation data. Zheng et al. [13] proposed a machine learning segmentation and classification algorithm for lane change detection using steering angle and vehicle data in CAN-buses. This type of method uses various sensor data acquired from multiple sensors to improve the matching accuracy [14,15]. However, these sensor data, which requires additional acquisition usually cannot be obtained from ordinary vehicle navigation devices. In addition, most machine learning based map matching methods rely on the process of parameter learning. In summary, the above machine learning based methods all need to collect, learn, and calculate a large amount of data. Compared with the probabilistic and weighted methods, more complex calculations are required.

(2) Methods based on probability calculation

In general, probability-based map-matching methods use a hidden Markov model to find the most likely path. In this process, the different combinations of roads that the vehicle could take are evaluated to find the best matching path. The sequence of projected points is the latent state of the Markov model, and the original GPS points are the observations. Different algorithms propose different transition probability distributions to determine the probability of passing through a candidate road segment. Most algorithms use a Gaussian distribution to describe the emission probability. Newson et al. [16] first proposed a hidden Markov model for fitting GPS data based on a probabilistic map matching framework. Zeng et al. [17] proposed the application of GPS trajectory curvature integral in map matching, and used the curvature feature to find the best matching path. However, when the sampling rate of input GPS points is increased, the performance of map matching is poor. Luo et al. [18] constructed a map matching method based on the hidden Markov chain model, using the aggregated geometric data of the road network, the topology matrix of the road segments and the refined quadtree structure, and combined the Viterbi algorithm to find the optimal order of the road segments. Chen et al. [19] proposed a three-stage map matching method based on driving direction. In this method, the directional probability and spatial probability of road segments around GPS points are calculated to obtain the corresponding matching path. However, when searching for the path, the best matching path is not further found by calculation.

(3) Methods based on weight assignment

Unlike probabilistic methods, weighted map matching methods assign costs to different candidate paths and use different selection methods to find potential paths. Lin et al. [20] proposed a Dijkstra-based selection mapping algorithm to evaluate the correctness of road sequences. Based on map data, a virtual directed graph is defined, whose nodes are a set of candidate points, and the weights of the edges represent the transition probability from one candidate point to another, to find a suitable path. However, the approximation between the candidate point and the GPS point is not considered. Ptošek et al. [21] proposed a map matching algorithm that combines geometric mapping with Dijkstra's shortest path algorithm to solve the problems of inaccurate vehicle position data and too short road segments in the matching process. However, when the frequency of trajectory point sampling is continuously reduced, the matching accuracy is also continuously reduced.

In addition, relevant researchers have also made significant efforts to improve the execution efficiency of map matching. Koller et al. [22] combined the computation of probabilities and assignment of weights by using a map matching method based on the HMM, but replaced the Viterbi algorithm with the Dijkstra algorithm. The method of mixing probabilities and weights reduces the running time of the map matching process by reducing the overhead in computing the transition probabilities. Traditional map matching methods for low-frequency trajectory data often need to find the shortest path between adjacent sampling points. As the sampling frequency continues to decrease, the distance between adjacent sampling points becomes larger and larger, and the shortest path selection method can no longer find the best matching path for the trajectory points. Therefore, it is necessary to match the low-frequency trajectory points to the corresponding road segments in a more precise way. To overcome the limitations of existing map matching algorithms, this study proposes a segmented map matching method based on the travel direction of trajectory points, combined with the hidden Markov model and the shortest path algorithm to ensure the matching accuracy and path topology feasibility.

The rest of the study is organized as follows. Section 2 presents the basic concepts and descriptions of related problems, and describes the framework and process of the proposed algorithm in detail. Section 3 verifies the performance of the proposed algorithm through experiments. The contribution description and associated analysis of the proposed method are discussed in Section 4. The conclusions and future prospects are presented in Section 5.

2. Methods

2.1. Basic Concepts and Problem Description

This section defines the relevant concepts used in this paper.

Definition 1. *(road segment): A road segment is a directed edge, defined as* $e = \left(e_{id}, e^{id}_{start}, e^{id}_{end}\right)$*, where* e_{id} *represents the road segment identifier,* e^{id}_{start} *represents the road segment start point* e_{id}*, and* e^{id}_{end} *represents the road segment end point* e_{id}*. In this study,* $e^{id}_{start} \to e^{id}_{end}$ *and* $e^{id}_{end} \to e^{id}_{start}$ *are treated as different directions of the same road segment.*

Definition 2. *(road segment node): The road segment node refers to the start point or end point of the road segment, expressed as* $v = (v_{id}, v_{lat}, v_{lon})$*, where* v_{id} *is the unique identifier of the road segment node and* v_{lat} *and* v_{lon} *are the latitude and longitude coordinates of the location of the road segment node location, respectively.*

Definition 3. *(road network): The road network is represented by a single directed graph G = (V, E), where V is the set of road segment nodes, expressed as* $V = \{v_1, v_2, \ldots, v_n\}$*, E is the set of all road segments, expressed as* $E = \{e_1, e_2, \ldots, e_n\}$*.*

Definition 4. *(location point): The location point (also called trajectory point) represents the location information acquired by the GPS device within a certain time interval, defined as* $p_i = \{lat_i, lon_i, t_i, s_i, h_i\}$*, where* lat_i *and* lon_i *represent the latitude and longitude coordinates of the* *current location point, respectively, t_i represents the time of the current location point, s_i represents the current speed of the vehicle, and h_i represents the current driving direction of the vehicle.*

Definition 5. *(original trajectory segment): The original trajectory segment consists of a set of location points ordered by timestamps, defined as $\tau = \langle p_i, p_{i+1}, \ldots, p_{i+u-1} \rangle$, where $p_j (i \le j \le i+u-1)$ is the location point, and the length of τ is the number of location points contained in τ. That is, $|\tau| = u$.*

Definition 6. *(matched trajectory segment): Given an original trajectory segment, its matched trajectory segment τ_m represents the actual driving segment of the vehicle in the road network, defined as $\tau_m = \langle e_i, e_{i+1}, \ldots, e_n \rangle$, where the road segments e_i and e_{i+1} have a common road segment node. That is, they are connected to each other.*

Definition 7. *(travel direction angle): The travel direction refers to the direction in which the vehicle is moving at the location point. The angle formed by the counterclockwise rotation of the forward direction to the true north direction is called the travel direction angle of the location point. The travel direction angle of the location point p_i is recorded as h_i, $0^\circ \le h_i \le 360^\circ$.*

The heading direction angle can be obtained from the trajectory dataset.

Definition 8. *(road segment direction angle): The angle formed by the counterclockwise rotation of the road segment direction to the north direction is defined as the road segment direction angle. The calculation method is shown in Equation (1).*

$$\beta = \begin{cases} \left(\frac{\pi}{2} - \arctan\left(\frac{y_2 - y_1}{x_2 - x_1}\right)\right) * \frac{180}{\pi}, & x_2 > x_1 \\ \left(\frac{3*\pi}{2} - \arctan\left(\frac{y_2 - y_1}{x_2 - x_1}\right)\right) * \frac{180}{\pi}, & x_2 < x_1 \\ 0, & x_2 = x_1 \wedge y_2 \ge y_1 \\ 180, & x_2 = x_1 \wedge y_2 < y_1 \end{cases} \tag{1}$$

Here, (x_1, y_1) and (x_2, y_2) are the latitude and longitude coordinates of the two points, respectively, and $0^\circ \le \beta \le 360^\circ$.

As shown in Figure 1, $E = \{e_1, e_2, e_3, \ldots, e_{18}\}$ represents the set of all road segments, and $V = \{v_1, v_2, v_3, \ldots, v_{15}\}$ represents the set of all road segment nodes. V and E together form an instance of a road network. The sequence of location points $\langle p_1, p_2, p_3, \ldots, p_7 \rangle$ form the original trajectory segment τ, where each location point contains latitude, longitude, timestamp, instantaneous speed, and travel direction information. For example, taking true north as the base direction, the travel direction angle of p_1 is represented as h_1. The matching trajectory segment of τ is $\tau_m = \langle e_9, e_{14}, e_6, e_7, e_{17} \rangle$.

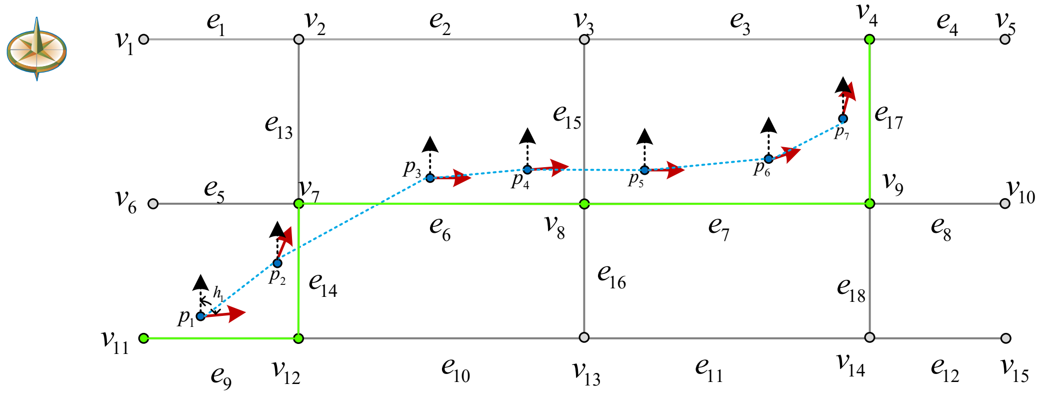


Figure 1. Example of trajectory data and road network model.

2.2. Low-Frequency Trajectory Map Matching Algorithm

2.2.1. Algorithm Overview

The proposed algorithm first divides the complete trajectory into several consecutive sub-trajectories according to the travel direction of the location point. Second, it calculates the directional probability for each sub-trajectory after segmentation, and obtains the spatial probability according to the distance between the location point and the road segment. Based on the directional probability and the spatial probability, the corresponding comprehensive probability is then obtained. Finally, the shortest path algorithm and the result of calculating the comprehensive probability are combined to obtain the matching paths under different probabilities. The optimal path is selected based on the similarity factor between the trajectory and the road segments, so that the trajectory points are matched to the road network. Figure 2 shows the schematic diagram of the proposed algorithm, which mainly includes the following three steps.

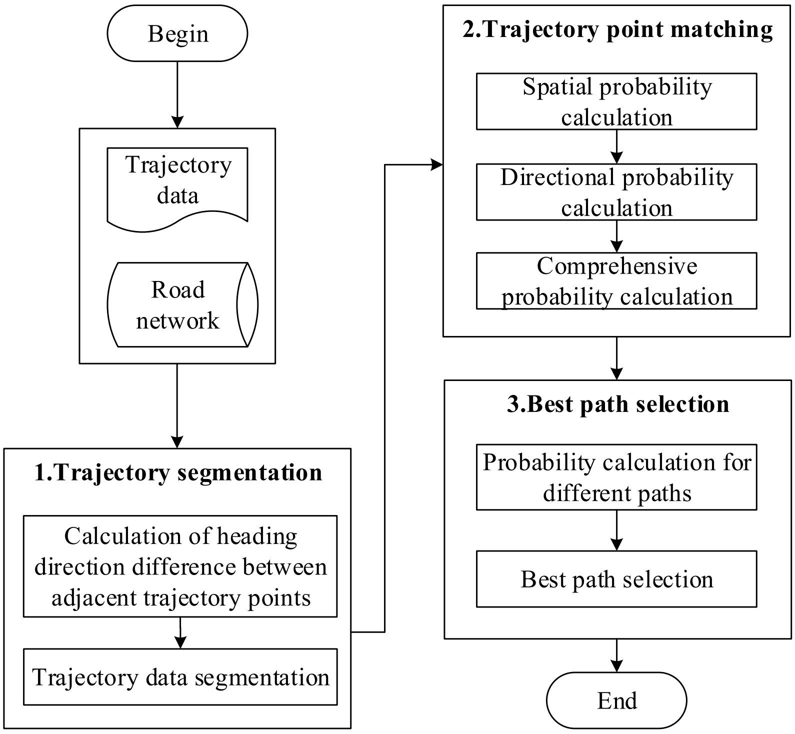

Begin
Trajectory data
Road network
1.Trajectory segmentation
Calculation of heading direction difference between adjacent trajectory points
Trajectory data segmentation
2.Trajectory point matching
Spatial probability calculation
Directional probability calculation
Comprehensive probability calculation
3.Best path selection
Probability calculation for different paths
Best path selection
End

Figure 2. Algorithmic framework.

(1) Trajectory segmentation

Consider an original trajectory segment $\tau = \{p_1, p_2, p_3, \ldots, p_n\}$, in which each location point p_i has its direction information during the driving process. The whole trajectory is segmented according to the travel direction of the location point. The first point in the subsequent segmented sub-trajectory is the last point in the previous segmented sub-trajectory.

(2) Trajectory point matching

For each trajectory point p_i, there are corresponding candidate road segments around it. Based on the distance between the trajectory point and the candidate road segment, the corresponding spatial probability G_1 is obtained, and the directional probability G_2 is obtained from the travel direction angle and the road segment direction angle. Finally, combining the spatial probability and the directional probability, the comprehensive probability G_w between the trajectory point and the road segment under consideration is obtained. The road segments under consideration are sorted according to their values for comprehensive probability.

(3) Best path selection

For each segmented sub-trajectory τ_i, we first find the path from the first point to the last point of τ_i using the shortest path algorithm. For each trajectory point p_i there are several candidate road segments with different probabilities, so in the path search, we obtain several matching sequences consisting of road segments with different probabilities. The path that best matches the original trajectory segment is selected based on the distance between the trajectory segment and each candidate path. After matching the first segmented sub-trajectory, the candidate road segment that matches the last point of the previous sub-trajectory is recorded, then the second sub-trajectory is matched, and so on, to finally obtain the best matching path.

2.2.2. Trajectory Segmentation

In an original trajectory segment, each trajectory point p_i has a corresponding travel direction angle $h_i(2 \leq i \leq n)$. The direction rotation angle θ_i between adjacent trajectory points is $\min(|h_i - h_{i-1}|, 360° - |h_i - h_{i-1}|)$. When $\theta_i \geq \theta_c$ is satisfied, we segment the trajectory, where θ_c is the angle threshold. In the experiments, the matching accuracy is tested under different angle thresholds, which determine the value of θ_c.

As shown in Figure 3, there is an initial trajectory segment $\tau = \{p_1, p_2, p_3, p_4, p_5, p_6\}$, p_1 corresponds to the travel direction angle h_1, p_2 corresponds to the travel direction angle h_2, and so on. In the segmentation process, the travel direction difference between p_1 and p_2 is calculated first, and the result is less than θ_c. Therefore, they belong to one segment. Then, we calculate the travel direction difference between p_2 and p_3 and find that it is greater than θ_c, so they are divided into different segments. According to this rule, the trajectory is segmented, and the segmented sub-trajectories are finally obtained as $\tau_1 = \{p_1, p_2\}$ and $\tau_2 = \{p_2, p_3, p_4, p_5, p_6\}$. To ensure the connectivity between the segmented sub-trajectories, the last point in τ_1 is the first point in τ_2. The pseudocode for trajectory segmentation is shown in Algorithm 1.

Algorithm 1: Trajectory segmentation

```
Input: Trajectory $T$, $\theta_c$
Output: Segmented trajectory set $\mathbf{\Gamma}$
Initialize $\mathbf{\Gamma} = \phi$, $ind = 2$, $cnt = 1$;
$\mathbf{\Gamma}_1^1 = p_1$;
$\mathbf{\Gamma}_1^2 = p_2$;
for $i = 3$ to $|T|$ do
  if $|h(p_i) - h(p_{i-1})| < \theta_c$ then
    $ind = ind + 1$;
    $\mathbf{\Gamma}_{cnt}^{ind} = p_i$;
  else
    $cnt = cnt + 1$;
    $\mathbf{\Gamma}_{cnt}^1 = p_{i-1}$;
    $\mathbf{\Gamma}_{cnt}^2 = p_i$;
    $ind = 2$;
  end if
end for
return $\mathbf{\Gamma}$;
```

The segmented sub-trajectories are stored in $\mathbf{\Gamma}$, which is the basis for the next trajectory point matching.

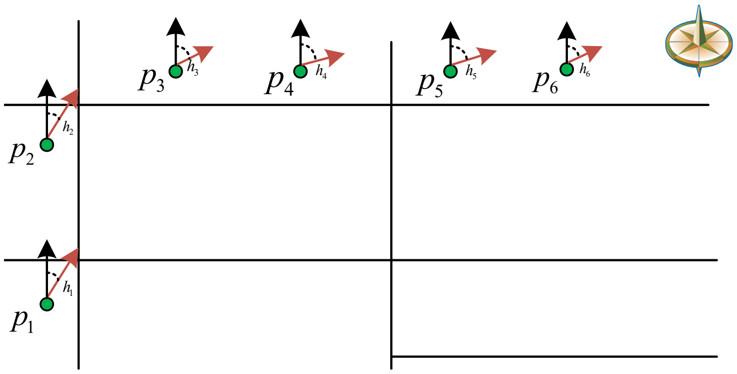


Figure 3. Trajectory segmentation example.

2.2.3. Trajectory Point Matching

After the trajectory is segmented, the candidate road segments are found with correspondingly different probabilities by analyzing the probabilities between the trajectory point and each road segment. As shown in Figure 4, a circle is drawn with the trajectory point p_1 as the center and r as the radius, and the road segments $\{e_1, e_2, e_3, e_4, e_5\}$ within the circle are selected as candidate road segments. In order to find the correct road segment that matches the trajectory point, we set the radius r to a relatively large value (such as 100 m). By calculating the spatial probability and the directional probability, we obtain m candidate road segments corresponding to the trajectory point for further processing.

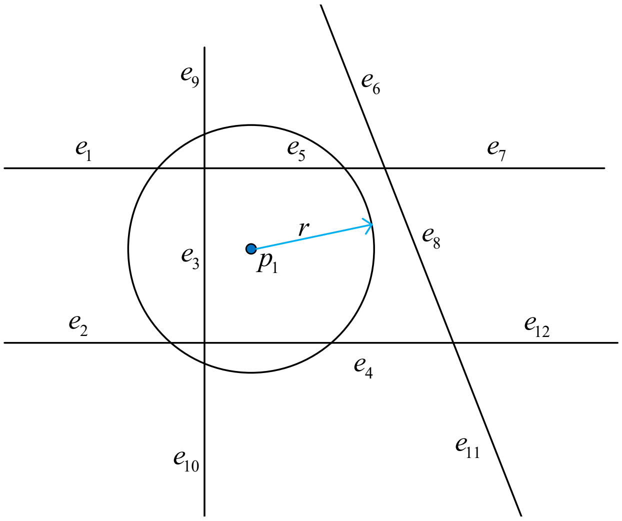


Figure 4. Selection of candidate road segments.

Definition 9. *(Spatial probability)* [19]: *Let $H_{p_i}^{e_j}$ be the shortest distance from the trajectory point p_i to the road segment e_j. We define the spatial probability as a measure on the possibility of spatial matching between the p_i and e_j, denoted as $G_1\left(H_{p_i}^{e_j}\right)$. The calculation methods of $H_{p_i}^{e_j}$ and $G_1\left(H_{p_i}^{e_j}\right)$ are shown in Equations (2) and (3), respectively:*

$$H_{p_i}^{e_j} = \min(disp(p_i, c_1), disp(p_i, c_2), disp(p_i, c_3)) \tag{2}$$

$$G_1\left(H_{p_i}^{e_j}\right) = \frac{1}{\sqrt{2\pi}\sigma_1} e^{-\frac{(H_{p_i}^{e_j})^2}{2(\sigma_1)^2}} \tag{3}$$

where c_1 and c_2 are the two endpoints of the road segment e_j respectively, and c_3 is the vertical projection point from p_i to e_j. If c_3 is not located in e_j, then $disp(p_i, c_3)$ is set to $+\infty$; σ_1 is *the standard deviation of location measurement, which is generally set to 20 m* [23]*. In general, the distance between the trajectory point and the road segment can be simulated as the normal distribution of the distance $H_{p_i}^{e_j}$.*

However, matched road segments determined only by spatial probability will have corresponding errors. As shown in Figure 5, for the trajectory point p_2 (p_3), e_3 (e_4) with the maximum spatial probability is an incorrect matched road segment. According to the direction of travel of the trajectory $p_1 \rightarrow p_2 \rightarrow p_3 \rightarrow p_4$, the correct matching road segments should be $e_1 \rightarrow e_2 \rightarrow e_5$. Therefore, the directional probability is proposed to improve the matching accuracy.

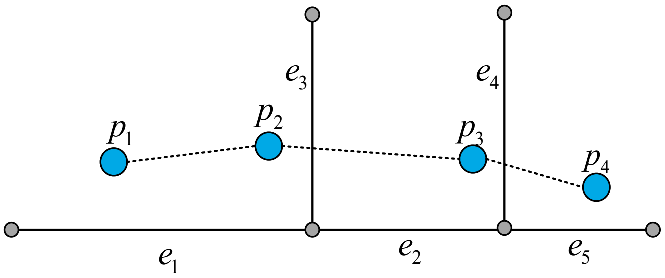


Figure 5. Disadvantages of matching only by spatial probability.

Definition 10. *(Directional probability)* [19]*: Let $A_{h_i}^{\theta_j}$ be the angular difference between the travel direction angle of p_i and the road segment direction angle. Define the directional probability as a measure on the possibility of the direction matching between the trajectory point p_i and the candidate road segment e_j, denoted as $G_2\left(A_{h_i}^{\theta_j}\right)$. The calculation methods of $A_{h_i}^{\theta_j}$ and $G_2\left(A_{h_i}^{\theta_j}\right)$ are shown in Equations (4) and (5), respectively:*

$$A_{h_i}^{\theta_j} = \min(|h_i - \theta_j^1)|, |h_i - \theta_j^2|) \tag{4}$$

$$G_2\left(A_{h_i}^{\theta_j}\right) = \frac{1}{\sqrt{2\pi}\sigma_2} e^{-\frac{(A_{h_i}^{\theta_j})^2}{2(\sigma_2)^2}} \tag{5}$$

where h_i represents the travel direction angle of p_i, θ_j^1 and θ_j^2 represent the angles in two different directions of the road segment; σ_2 is the standard deviation of the directional measurement, and is also set to 20 m.

Based on Equation (5), we obtain different directional probabilities between each trajectory point and the different candidate road segments. As for the spatial probability, the directional probability can also be simulated as a normal distribution of the travel direction angle difference $A_{h_i}^{\theta_j}$.

Definition 11. *(Comprehensive probability)* [19]*: By combining the spatial probability and directional probability, this study defines the comprehensive probability as G_w, which is calculated by Equation (6).*

$$G_w = \sqrt{G_1 \times G_2} \tag{6}$$

According to the above calculation and discussion, there is at least one eligible road segment around each trajectory point, so there is at least one comprehensive probability. Based on the comprehensive probability value, we select the *m* candidate road segments with the highest probability for the selection of the next best path and the evaluation of the map matching algorithm. The pseudocode for trajectory point matching is shown in Algorithm 2.

Algorithm 2: Trajectory point matching

```
Input: Segmented trajectory set $\mathbf{\Gamma}$, Road Network $G$, Radius threshold $r$, Candidate value $k$
Output: Candidate road segment set $\mathbf{Cr}$, Comprehensive probability set $\mathbf{Gw}$
1: Initialize $\mathbf{Cr} = \varnothing$, $\mathbf{N} = \varnothing$, $\mathbf{H} = \varnothing$;
2: for each $p_i \in \mathbf{\Gamma}$ do
3:    Construct a circle $C_i$ with $p_i$ as center and $r$ as radius;
4:    $\mathbf{Cr} \leftarrow \mathbf{Cr} \cup$ {the candidate road segments around $C_i$};
5:    for each $c_j \in \mathbf{Cr}$ do
6:      $\mathbf{N}_{ij} \leftarrow$ the spatial probability of $c_j$ calculated by Equation (3);
7:      $\mathbf{H}_{ij} \leftarrow$ the heading probability of $c_j$ calculated by Equation (5);
8:      $\mathbf{Gw}_{ij} \leftarrow$ the comprehensive probability of $c_j$ calculated by Equation (6);
9:    end for
10: end for
11: return $\mathbf{Gw}$, $\mathbf{Cr}$;
```

After matching the trajectory points with Algorithm 2, **Gw** stores the comprehensive probabilities for each trajectory point, and **Cr** stores the eligible road segments around each trajectory point. The next step is to find the optimal path through the segmented sub-trajectories and the comprehensive probabilities.

2.2.4. Best Path Selection

In each segmented sub-trajectory $\tau_i^k = \left\{p_i^a, p_i^{a+1}, \ldots, p_i^b\right\}$, each trajectory point has its corresponding candidate road segments with comprehensive probability G_w values. Since the research object of this paper is low-frequency trajectory data, the road segments coinciding with the adjacent trajectory points are generally not connected. Therefore, we find the best road segment matching the trajectory point through three steps.

Step 1 (Establish trajectory buffer): In the selection of the best path, we need to find the path between the trajectory points using the shortest path algorithm. Due to the size of the road network, it is impossible to find the path in the entire road network. Therefore, we filter the road network by the travel directions of the trajectory points before matching. As shown in Figure 6, there are four trajectory points. We construct a circle with the first trajectory point as the center and r' as the radius, and then translate continuously along the trajectory travel direction. After one continuous iteration, we obtain the corresponding road buffer. In this study, the trajectory buffer is constructed by determining the road segments and nodes contained in the road buffer.

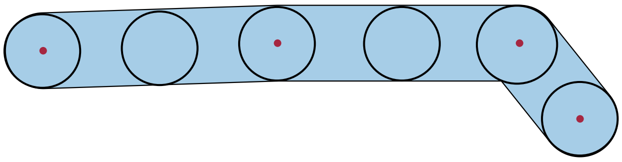


Figure 6. Creating the trajectory buffer.

Step 2 (Find matching paths): Each segmented sub-trajectory $\tau_i^k = \left\{p_i^a, p_i^{a+1}, \ldots, p_i^b\right\}$ corresponds to a trajectory buffer. Due to the large distance between two location points in the low-frequency trajectory, two continuous points are usually matched to two unconnected road segments in the road network, which may be far apart. Thus, we need to find possible paths between them. Let us take τ_i^k as an example, for p_i^a, there are m possible road segments, and the same is true for p_i^b. Each trajectory point corresponds to several possible road segments with different probabilities. In this study, potential paths with different probabilities are found by finding the paths between the candidate road segments with different probabilities of the first point and the last point in each segmented sub-trajectory. When finding paths, the candidate road segment with the highest probability is not necessarily the matching road segment. Based on the above considerations, the following operations are performed in this study.

As shown in Figure 7, there is a segmented sub-trajectory $\tau\prime = \{p_1, p_2, p_3\}$ in the trajectory buffer. We first determine the start and end points of the path according to the travel direction of the trajectory. The candidate road segments of the first trajectory point p_1 are $\{e_{10}, e_7, e_8\}$, sorted by to the comprehensive probability, and the sorted candidate road segments of the last trajectory point p_3 are $\{e_3, e_2, e_{12}\}$. Starting from the road segment with the highest probability of p_1 (i.e., e_{10}), we first find the start point of the path corresponding to the angle between the travel direction of p_1 and the direction of the road segment, which is denoted as v_6 in the figure. Then, we calculate the difference between the road segment direction angle of the maximum probability candidate road segment e_3 of the last point p_3 and the travel direction angle of p_3, and find that the end point of the path is v_7. Therefore, we find the shortest path from v_6 to the node v_7 is $e_{10} \rightarrow e_{11} \rightarrow e_2$.

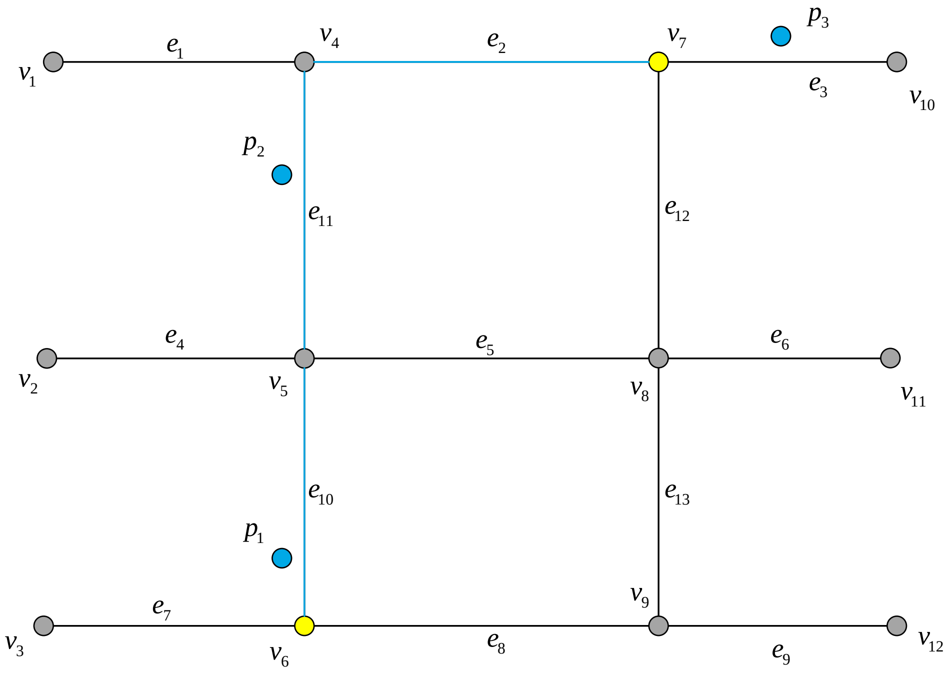


Figure 7. Finding matching paths.

However, the path found in the maximum comprehensive probability way is not necessarily the final matching path. As shown in Figure 8, the sub-trajectories segmented by the trajectory direction are $\tau_1 = \{p_1, p_2\}$ and $\tau_2 = \{p_2, p_3\}$. For the last point of τ_1 (i.e., p_2), the road segment with the highest comprehensive probability is e_4, but we can see in Figure 8 that p_2 should match to e_5. Therefore, Step 3 is used to determine the best matching path.

Step 3 (Find the best matching path): For each trajectory point in the segmented sub-trajectory, there are candidate road segments with different comprehensive probabilities. Therefore, in this step, k candidate paths are found from the start point to the end point in the segmented sub-trajectories, as shown in Table 1. (1) When k = 1, it is the path from the candidate road segment with the highest comprehensive probability of the start point to the candidate road segment with the highest comprehensive probability of the end point; (2) When k = 2, it is the path from the candidate road segment with the highest comprehensive probability of the start point to the candidate road segment with the second highest probability of the end point; (3) When k = 3, it means the path from the second-highest-probability candidate road segment of the start point to the highest-probability road segment of the end point; (4) When k = 4, it means the path from the second-highest-probability candidate road segment of the start point to the second-highest-probability candidate road segment of the end point; (5) When k = 5, it means the path from the candidate road segment with the highest comprehensive probability of the start point to the candidate road segment with the third highest probability of the end point. In the experiments, different *k* values (*k* = 1, 2, 3, 4, 5) are set to evaluate the effect of *k* different paths and find the best path.

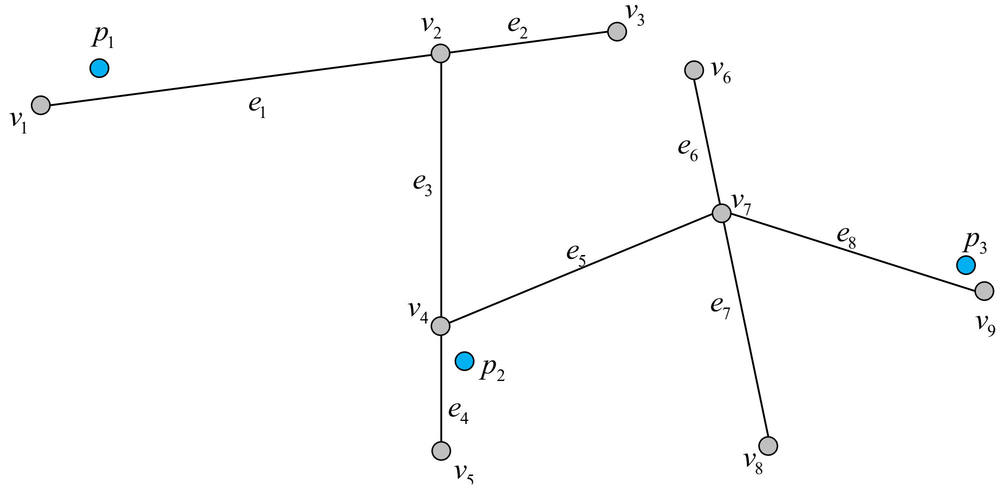


Figure 8. Problems in the search for matching paths.

Table 1. *k* cases of candidate paths.

k	Comprehensive Probability Ranking of the Candidate Road Segment	
	Corresponding to the Start Point	Corresponding to the End Point
1	1	1
2	1	2
3	2	1
4	2	2
5	1	3

By the above analysis, we can find the relevant *k* candidate paths for each segmented sub-trajectory of the original trajectory, and combine the candidate paths of the different sub-trajectories to get *k* complete candidate paths. Figure 9 shows the trajectory points p_1, p_2, p_3 and the corresponding matching road segments e_1 and e_3. In the path finding process, $e_1 \to e_2 \to e_3$ is one of the candidate paths. To find the best matching path, the distance between the original trajectory $p_1 \to p_2 \to p_3$ and each candidate path must be calculated. First, calculate the shortest distance from each trajectory point to each road segment in the path. That is, the projected distance from the trajectory point to the road segment. If the projected point is not on the road segment, calculate the distance between the trajectory point and the projected point on the extended line of the road segment. As shown in Figure 9, the distances from p_1 to road segments e_1, e_2, and e_3 are d_1, d_2, and d_3, respectively; the distances from p_2 to road segments e_1, e_2, and e_3 are d_4, d_5, and d_6, respectively; and the distances from p_3 to road segments e_1, e_2, and e_3 are d_7, d_8, and d_9 respectively. Second, add all the shortest distances to get the distance between the trajectory and the current candidate path. Since there are *k* candidate paths connected by road segments with different probabilities, the distance between any trajectory and its *k*-th candidate path is denoted as D_k $(k = 1, 2, 3, 4, 5)$. Using Equation (7), the path with the largest probability value $P(D_k)$ is selected from the *k* candidate paths as the best matching path.

$$P(D_k) = \frac{1}{\sqrt{2\pi}\sigma_3} e^{-\frac{(D_k)^2}{2(\sigma_3)^2}}, \tag{7}$$

where σ_3 is set to 20 m, representing the same meaning as σ_1 and σ_2. The pseudocode for the best path selection is shown in Algorithm 3.

Algorithm 3: Best path selection

```
Input: Trajectory segment set $\mathbf{\Gamma}$, Road Network $G$, Candidate value $k$, Comprehensive probability set Gw
Output: Final path set Path
Initialize $k \leftarrow \{1,2,3,4,5\}$, Path= $\varnothing$;
Sort the candidate road segments according to Gw;
dist←the adjacency matrix of the link nodes in the sorted set of candidate road segments;
for each $\tau_i \in \mathbf{\Gamma}$ do
    $\mathbf{Path}_{tem}$←the different shortest paths obtained based on dist and different $k$ values;
    $\mathbf{Path}_i$←the best path obtained based on Equation (7);
end for
return Path;
```

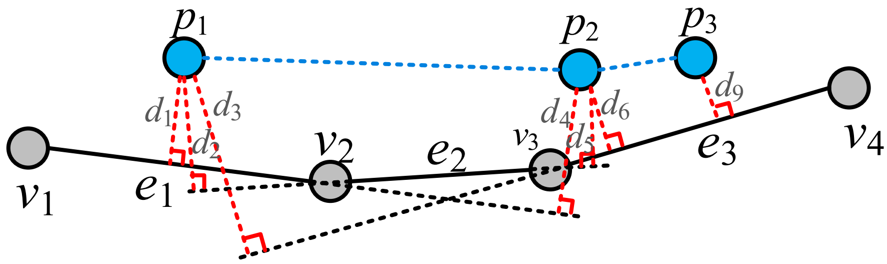


Figure 9. Find the best matching path.

After the optimal path is selected using Algorithm 3, the final trajectory matching path is stored in **Path**.

2.2.5. Algorithm Complexity Analysis

The time complexity of Algorithm 1 mainly depends on the computational cost of the segment angle difference. The time complexity of this part is $O(n)$, where n is the number of location points in the trajectory, so the time complexity of Algorithm 1 is $O(n)$.

The time complexity of Algorithm 2 mainly depends on: (1) the cost of searching for possible road segments around each trajectory point; (2) the cost of computing spatial probability, directional probability, and comprehensive probability. The time complexity of scanning the trajectory point set and scanning the candidate road segment set are both $O(n \times p)$, where p is the number of candidate road segments, $p \ll n$, so the time complexity of Algorithm 2 is $O(n)$.

The time complexity of Algorithm 3 mainly depends on: (1) the cost of finding the shortest path of each segmented sub-trajectory, the time complexity of this part is $O(m)$, where m is the number of segmented sub-trajectories; (2) the cost of finding the best matching path from the shortest path set, the time complexity of this part is $O(k)$, where k is the number of candidate paths. $k \ll m$, so the time complexity of Algorithm 3 is $O(m)$.

3. Results

In this section, the performance of the proposed algorithm is evaluated under various aspects with real trajectory data and road network data.

The experimental environment is an Intel Corei5 processor with 2.4 GHz CPU. The operating platform is Windows 10, and the proposed algorithm is implemented using MATLAB 2018a. The relevant datasets, evaluation indicators, parameter settings, experimental results, and analysis are presented below.

3.1. Datasets

3.1.1. Road Network Dataset

In this study, the road network of Shanghai is imported via ArcGIS [24]. The road network data include major urban roads, urban secondary roads, urban branch roads, residential roads, expressways, tunnels, elevated expressways, suburban trails, sidewalks, driveways, and more. The data contain more than 100,000 road segments and more than 70,000 nodes.

3.1.2. Trajectory Dataset

The trajectory dataset is from the trajectory data of more than 10,000 taxis in Shanghai, China, on 20 February 2007, collected by the Smart City Research Group of the University of Hong Kong. The trajectory data fields include the taxi number, timestamp, latitude and longitude, the vehicle's travel direction, instantaneous speed, and whether the vehicle is carrying passengers. To illustrate the relevant features of taxi trajectory data, we used the trajectory data from 6:00 a.m. to 10:00 p.m. The number of sampling points and the average distance between adjacent points under different sampling frequencies are shown in Table 2. Following the literature [25], we use ArcGIS software to put the vehicle trajectory data and Shanghai road network data in the same layer, and find out the correct road segments matching the trajectory points by manual tagging. In this study, the sampling rate of the trajectory data is reduced to obtain low-frequency trajectory data. Based on this, experiments are conducted to evaluate the performance of the algorithm.

Table 2. Trajectory data information at different sampling frequencies.

Sampling Interval (s)	Number of GPS Points	Average Distance (m)
15	110,972	377.81
30	60,575	631.01
60	59,941	964.72
90	39,220	1721.07
120	16,970	2295.23

3.2. Evaluation Indicators

In this study, three evaluation indicators are used to measure the superiority of the algorithm, including two evaluation indicators to measure the matching accuracy and one to measure the matching time.

3.2.1. Trajectory Point Matching Accuracy

The trajectory point matching accuracy, denoted as R_c, refers to the ratio between the number of trajectory points correctly matched with the road network and the number of trajectory points to be matched, calculated as follows:

$$R_c = \frac{P}{R} \times 100\%, \tag{8}$$

where P represents the number of trajectory points correctly matched to the road segment, and R represents the number of trajectory points used for matching.

3.2.2. Road Segment Matching Accuracy

The road segment matching accuracy, denoted as R_m, refers to the ratio between the number of road segments correctly matched to the road network to the total number of actual road segments of the trajectory, calculated as follows:

$$R_m = \frac{l}{L} \times 100\%, \tag{9}$$

where l represents the number of correctly matched road segments, and L represents the total number of actual road segments of the trajectory.

Since this algorithm is used for matching low-frequency trajectory data, the distance between trajectory points is relatively large, and there are more road segments between each two consecutive trajectory points. Therefore, the R_m indicator is used to measure the accuracy of road segment matching.

3.2.3. Trajectory Point Matching Time

Use N_s to denote the average runtime required for each trajectory point to complete the matching.

$$N_s = \frac{Trt}{R}, \tag{10}$$

where Trt represents the total time required for the entire trajectory to complete the map matching process, and R represents the number of trajectory points used for matching.

Due to the complexity of road networks, the accuracy of trajectory map matching changes accordingly for road networks with different topologies. Therefore, in this study, a visual comparison of the map matching results of different algorithms in road networks with different topologies is performed, which shows the superiority of the proposed algorithm.

3.3. Parameter Setting

3.3.1. Segmentation Angle

In the first stage of the algorithm, we perform the trajectory segmentation operation according to the specified segmentation angle (θ_c) value. Based on trajectory feature analysis and experimental results, we found that the best segmentation angle was 20°. As different datasets have different characteristics, we set the angle to different values to check its effect on accuracy and time. In particluar, five different angle values of 0°, 20°, 40°, 60°, and 80° are used to compare the matching accuracy and matching time of the same trajectory.

3.3.2. Number of Candidate Paths

In the second stage of the algorithm, we determine the candidate road segments around each trajectory point according to the comprehensive probability obtained from the spatial probability and the directional probability, and sort the road segments by comprehensive probability in descending order. In the third stage, we find that the final matched path is not necessarily the path between the candidate road segments with the highest comprehensive probabilities. Therefore, by fixing the number of candidate paths (k), we can find potential paths between candidate road segments with different probabilities. To check the effect of different number of paths k on matching accuracy, we set the value of k to 1, 2, 3, 4, and 5, respectively, and use the same trajectory for experimental verification, to find the effects of the number of paths on the map matching accuracy and matching time.

3.3.3. Sampling Frequency

The sampling frequency reflects the time interval between two adjacent trajectory points. To verify the superiority of the proposed algorithm, trajectory data with different sampling intervals, which are 15 s, 30 s, 60 s, 90 s, and 120 s, respectively, are collected. The matching accuracy and matching time of the algorithm at different sampling frequencies are compared and analyzed, which shows the superiority of the proposed algorithm.

3.4. Experimental Results

3.4.1. Experimental Comparison Results Based on Different Parameters

(1) Experimental results at different segmentation angles

Figure 10 shows the results of comparing the matching accuracy R_c, R_m and matching time N_s under different segmentation angles θ_c. The "(s)" in the symbol "N_s (s)" indicates that the time unit of N_s is seconds. The sampling interval between trajectory points is 60 s. It can be seen that when the angle is in the range of [0°, 40°], the time required is almost the same, and when the segmentation angle is in the range of [60°, 80°], the matching time is reduced. In terms of accuracy, we find that for segmentation angles of 20° and 40°, the matching accuracy reaches a peak, and the matching accuracy is the same under the two segmentation angles. When the angle is 60° and 80°, the time loss decreases, the matching accuracy also decreases significantly.

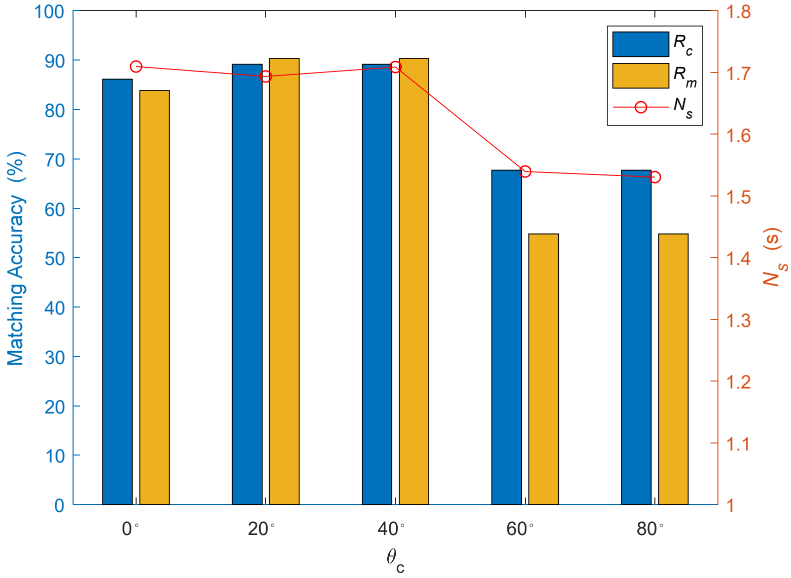


Figure 10. Matching results under different segmentation angles.

From the experiments, it can be seen that at $\theta_c = 20°$, the matching accuracy peaks and the time loss is relatively small. This is because at a segmentation angle of 0°, all trajectory points are considered to be segmented. When the road segments in the road network are dense, using a segmentation angle of 0° during the matching process causes the number of segmented sub-trajectories to be larger, and the number of trajectory points in each sub-trajectory to be smaller. The result cannot be corrected in the path finding process, leading to corresponding errors. When the segmentation angle is $\theta_c = 40°$, the accuracy remains the same as $\theta_c = 20°$, but more time is required in the matching process. For angles of 60° and 80°, the trajectory that should be segmented is not segmented, the number of segments in a trajectory is reduced, and the number of times a path is found is correspondingly reduced. However, the trajectories are not segmented correctly, resulting in a large error. That is, both the matching accuracy and the matching time decrease. Therefore, we set θ_c to 20° in the following experiments.

(2) Experimental results under different candidate path values

As can be seen in Figure 11, as the number of candidates increases, the matching accuracy also increases, and the time peaks when k = 5. This is because the path search selects the best matching path by comparing the candidate road segments with different probabilities. When k = 1, the path searched is the path between the maximum probability matching road segments that correspond to the trajectory points, so there is a large matching error. As the k value increases, the matching accuracy improves due to the continuous correction of the path. In the following experiments for comparison with other algorithms, we set k = 5.

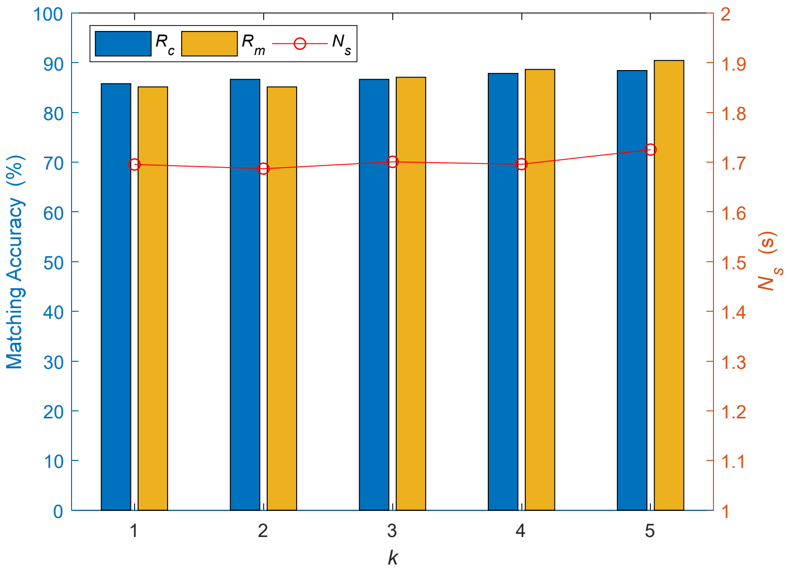


Figure 11. Matching results under different candidate values for *k*.

(3) Experimental results at different sampling frequencies

Table 3 shows the corresponding results for the trajectory map matching accuracy R_c, R_m and matching time N_s at different sampling frequencies and different path candidate values. From the table, we can see that with the continuous increase of the sampling interval, the trajectory map matching accuracy continues to decrease. This is because with the continuous reduction of the sampling frequency, the relevant information between the trajectory points also becomes less, and the distance between two adjacent trajectory points continues to increase, which increases the possibility of matching road segments. The algorithm has to make calculations in different situations. Therefore, the accuracy will continue to decrease and the matching time will continue to increase.

Table 3. Comparison results under different sampling frequencies and path candidate values.

Sampling Interval (s)	*k*	R_c (%)	R_m (%)	N_s (s)
30	1	86.95	88.41	0.369709
	2	89.86	92.31	0.370341
	3	89.86	92.31	0.371870
	4	92.75	94.87	0.375194
	5	92.75	94.87	0.377814
60	1	83.30	85.9	0.598502
	2	88.09	91.03	0.633812
	3	92.85	91.03	0.617812
	4	92.85	91.03	0.609079
	5	92.85	91.03	0.621662
120	1	68.26	70.06	0.952200
	2	73.23	73.21	0.962600
	3	73.23	73.21	0.996057
	4	73.23	73.21	1.014783
	5	78.26	78.20	1.027100

3.4.2. Comparison Results with Other Algorithms

Figure 12a shows the comparison results of ST-Matching [23], SD-Matching [19], and the proposed algorithm in terms of trajectory point matching accuracy R_c. In this study, the trajectory is segmented by setting the threshold value for the segmentation angle θ_c, and the path results are corrected in path finding to better match the trajectory points to the road network. It can be seen that the proposed algorithm has about 5% higher matching accuracy compared with the SD-Matching algorithm. This is because after the trajectory segmentation, we only need to analyze the path between the first and the last trajectory points in the segmented sub-trajectories. In the path search, the paths between candidate road segments with different probabilities are compared to correct the matching result, to improve the matching accuracy. The ST-Matching algorithm assigns the location points to the road segments with a short distance, and does not consider the relationship with the neighboring points, resulting in a large matching error.

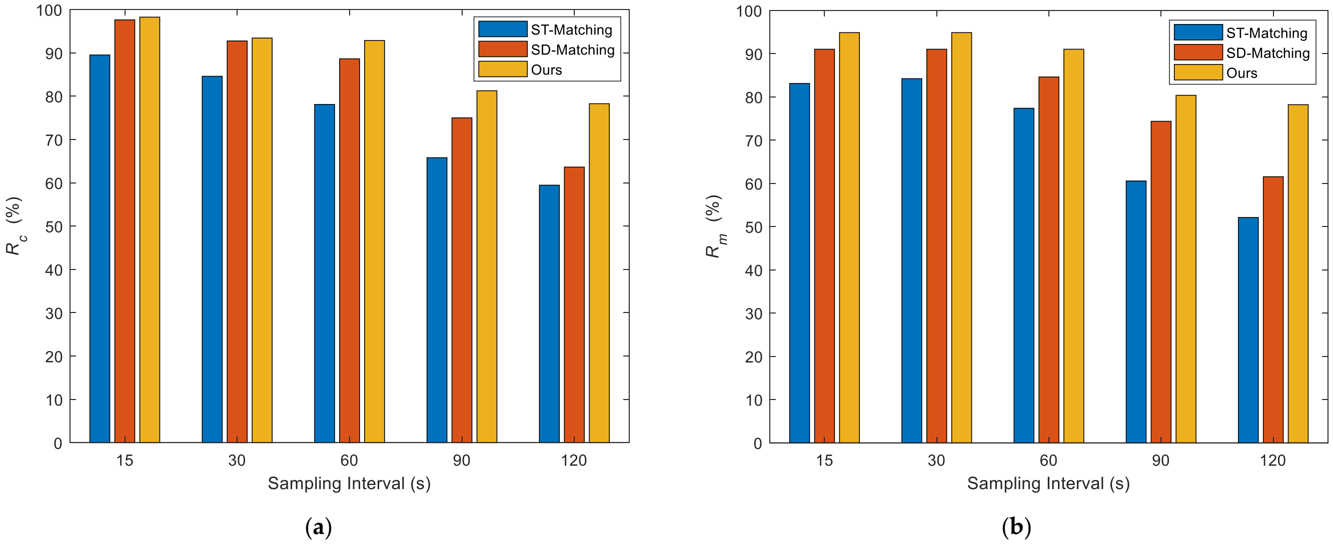


Figure 12. Comparison of the accuracy of the algorithms under different sampling frequencies. (**a**) Comparison of the R_c indicators. (**b**) Comparison of the R_m indicators.

Figure 12b shows the comparison results of ST-Matching, SD-Matching, and the proposed algorithm in terms of road segment matching accuracy R_m. In this study, the best path is selected by comparing and screening the paths between candidate road segments with different probabilities. It can be seen that the proposed algorithm based on the trajectory data of different sampling frequencies has about 8% improved matching accuracy compared to ST-Matching and about 5% compared to SD-Matching. From the above analysis, we can see that after calculating and analyzing the paths between the candidate road segments with different probabilities, the road segment matching accuracy is further improved.

Figure 13 shows the comparison results of ST-Matching, SD-Matching and the proposed algorithm in the running time N_s. It can be seen that the running time of the proposed algorithm is reduced by about 10–20% compared to the other two algorithms. The larger the sampling interval between the trajectory points, the smaller the difference in running time between this algorithm and the other two algorithms. The proposed algorithm segments the trajectory based on the segmentation angle threshold. The matching time of the trajectory data mainly depends on the number of segmented sub-trajectories. The road network was reduced and optimized based on the feature of the trajectory point travel direction. Compared to SD-Matching, a semicircular buffer is not built, but a trajectory buffer corresponding to the driving direction between the trajectory points is built, which improves the matching efficiency. The ST-Matching algorithm does not build a buffer during the path search, so the scope of the path includes the entire road network, which takes a lot of time.

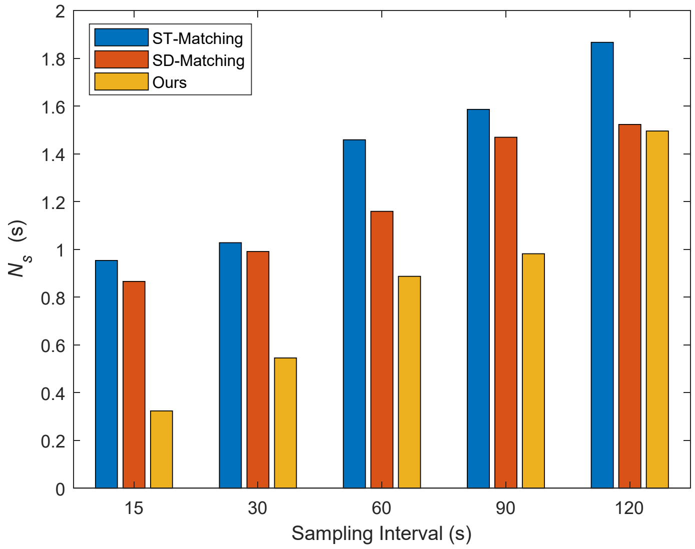


Figure 13. Comparison of the running times of the algorithms at different sampling frequencies.

Figure 14 shows the visualization results of ST-Matching, SD-Matching, and the proposed algorithm under the topology of the bus-type road network. It can be seen that the three algorithms have good matching effects in the uncomplicated road network dominated by the bus topology, and all of them can match the trajectory points with the correct road network. Figure 15 shows three visualization results with multiple parallel road topologies. We can see that ST-Matching mainly matches the trajectory points to the nearest road segments, and finds the path between the corresponding road segments of each trajectory point. Although the SD-Matching algorithm also determines the relevant number of candidates in the matching process, it does not correct the road segments between different candidate paths. If there is a path between the candidate road segments with the highest probability corresponding to the trajectory points, it is considered as the final matching path, which leads to large errors. The proposed algorithm makes appropriate corrections to the matching results, and this improves the matching accuracy.

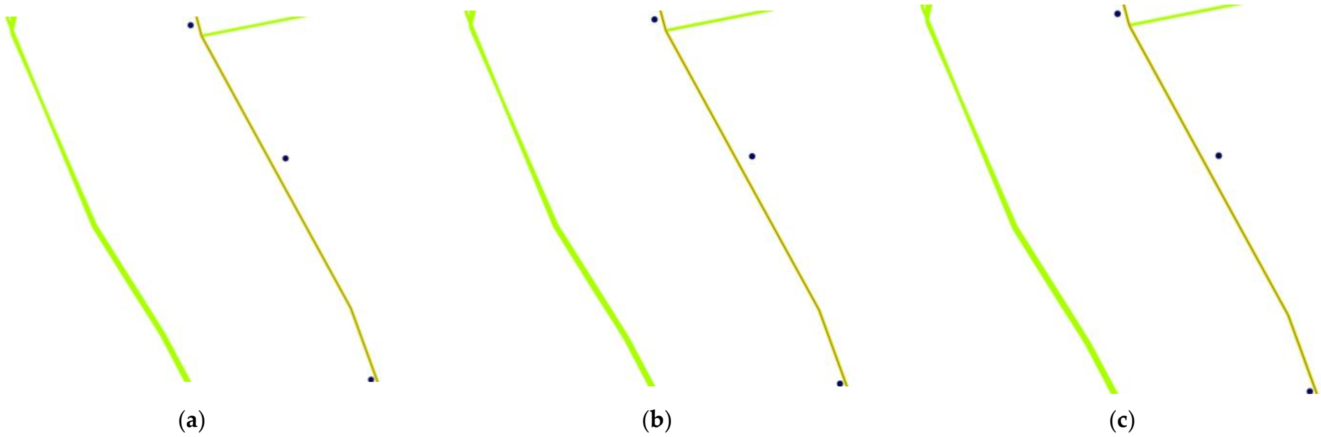
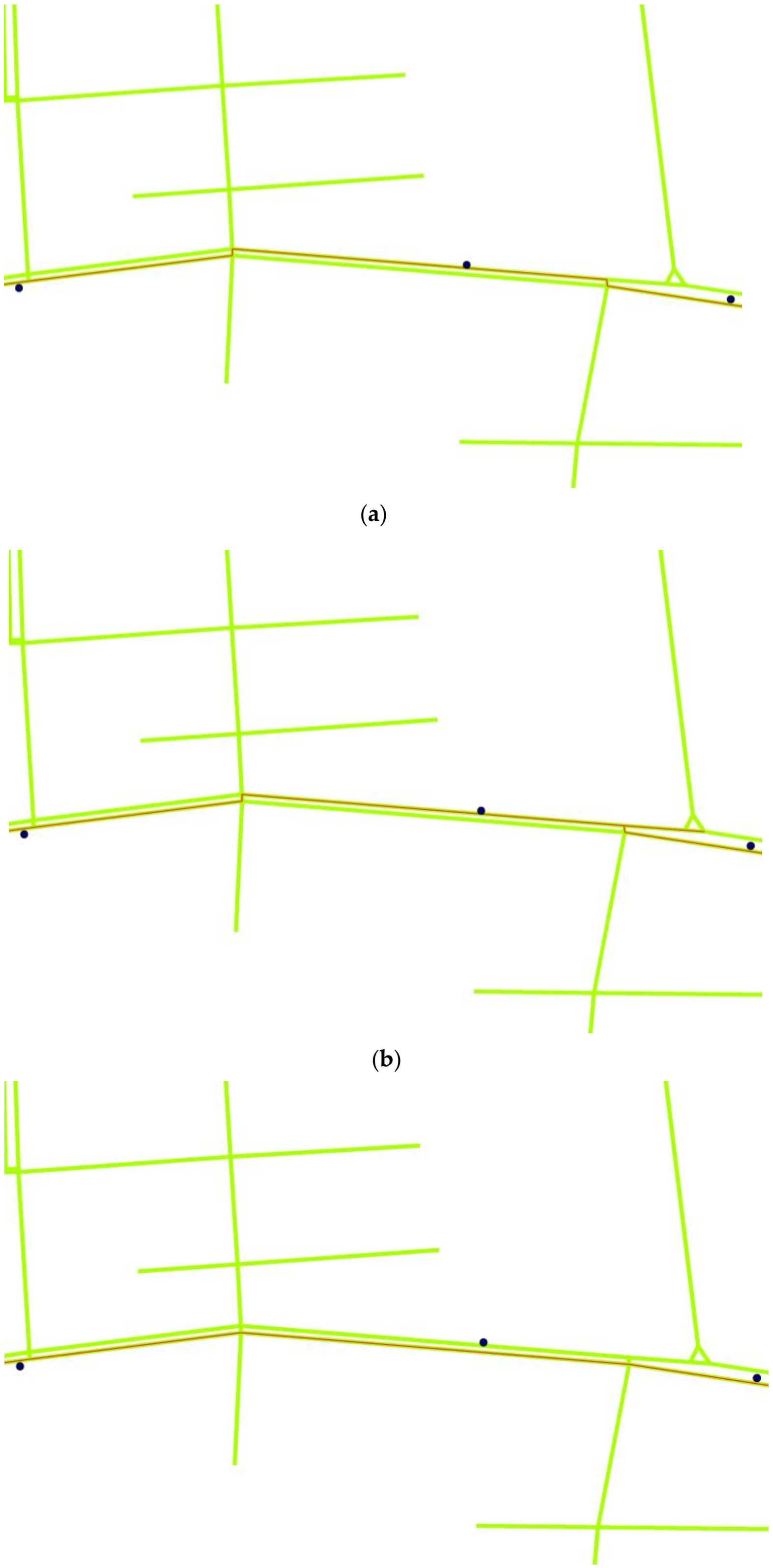

(**a**) (**b**) (**c**)

Figure 14. Visual comparison results of the algorithms under a single road topology. (**a**) ST-Matching algorithm. (**b**) SD-Matching algorithm. (**c**) The proposed algorithm.



(**a**)

(**b**)

(**c**)

Figure 15. Visual comparison results of the algorithms under parallel road topology. (**a**) ST-Matching algorithm. (**b**) SD-Matching algorithm. (**c**) The proposed algorithm.

4. Discussion

There is a certain range of errors in the data collected by the positioning device. If these data are directly used in traffic detection, urban planning, and other applications, it will lead to a large degree of error, resulting in waste of manpower and material resources. The map matching method plays an important role in the preprocessing step of trajectory data mining. In practice, vehicles usually report their GPS positions to the dispatch center at a lower sampling rate in order to save communication and energy costs [26–28]. If the processing object is high-frequency trajectory data, where the distance between adjacent location points is short, the road network topology can be used for analysis and matching [29]. By contrast, the low-frequency trajectory map matching method generally cannot guarantee a feasible matching path, due to the long time interval between two adjacent trajectory points, i.e., there may be several road segments between the segments matched by two adjacent trajectory points.

For low-frequency trajectory data, related researchers have proposed several map matching algorithms [30–32], which usually combine the hidden Markov model (HMM) and the shortest path algorithm to determine the shortest path between two consecutive trajectory points. Although these algorithms can achieve high matching accuracy, the time required to calculate the shortest path is significantly high. Lengthy computations impose a huge computational burden on the algorithms and limit the application scenarios of the algorithms. To overcome this challenge, this paper proposes an efficient map-matching algorithm for low-sampling-rate trajectory data. The proposed algorithm firstly segments the complete trajectory according to the travel direction of the location points; secondly, the candidate road segments around the segmented trajectory points are selected according to the comprehensive probability obtained from the spatial probability and the directional probability; thirdly, based on the obtained candidate road segments and the shortest path algorithm, each segmented sub-trajectory is matched with the road network, and the candidate paths are obtained; finally, the best matching path is found by the correlation calculation. Experiments with real trajectory datasets and road network data show that the proposed algorithm has higher matching accuracy and matching efficiency.

The main contributions of this study are as follows:

(1) A new low-frequency trajectory map matching algorithm is proposed, which matches the segmentation of trajectory data according to the driving direction of location points and improves the matching accuracy;
(2) The segmentation matching method, hidden Markov model and shortest path algorithm are combined to improve the matching accuracy and matching efficiency;
(3) Experiments are performed using real trajectory datasets and Shanghai road network data, and the results show that the proposed vehicle direction segmentation-based map matching algorithm achieves higher accuracy and requires less running time. In particular, the proposed method shortens the running time by almost 10−20%, and the matching accuracy is improved by almost 5%. Simultaneously, it has good adaptability under the parallel road topology.

The algorithm proposed in this paper improves the map matching ability of trajectory data to a certain extent, and improves the matching accuracy and matching efficiency of trajectory data with low sampling rate. The results of this study are helpful for vehicle navigation, tracking, and mapping of new roads, because to truly benefit from the emerging technology of data-driven infrastructure planning, it is important to combine telematics and map data to make it easier to extract and mine useful information from the GPS data to assist in designing traffic systems in large urban areas to prevent congestion. Traffic jams cause large economic losses, so anything that can assist in improving travelling time will likely be welcomed by logistics professionals and civil engineers.

5. Conclusions

In this study, we propose a new algorithm for low-frequency trajectory map matching that considers the vehicle's direction of travel and the hidden Markov model. The proposed algorithm considers the situation that the path found by the shortest path matching algorithm has errors due to the large distance between adjacent location points in the low-frequency trajectory data, calculates candidate paths with different probabilities by similarity measure, and finally obtains the optimal matching path. We compare the proposed algorithm with two similar algorithms on real trajectory and road network datasets, indicating that the proposed algorithm is excellent in matching accuracy and efficiency. It provides a good data foundation for applications in the data mining field. These matched GPS trajectory data are typically integrated with geographic information systems in intelligent transportation systems to provide various services to individuals or companies, such as vehicle navigation, tracking, and map drawing.

The proposed algorithm has a good effect on obtaining the complete matched path, but it can only be applied to offline trajectory data. In the future research, we will extend the proposed algorithm for online applications.

Author Contributions: Conceptualization, Qingying Yu; Data curation, Fan Hu; Funding acquisition, Qingying Yu, Chuanming Chen, Liping Sun and Xiaoyao Zheng; Methodology, Fan Hu and Qingying Yu; Supervision, Chuanming Chen; Validation, Chuanming Chen; Visualization, Qingying Yu; Writing—original draft, Fan Hu; Writing—review and editing, Qingying Yu, Liping Sun and Xiaoyao Zheng. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the National Natural Science Foundation of China, grant number 61702010, 61972439, 61672039, the University Natural Science Research Program of Anhui Province, grant number KJ2021A0125, the Anhui Provincial Natural Science Foundation of China, grant number 2108085MF214, and the Key Program in the Youth Elite Support Plan in Universities of Anhui Province, grant number gxyqZD2020004.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Acknowledgments: The authors would like to thank the reviewers for their useful comments and suggestions for this paper.

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

References

1. Kiani, A.; Liu, G.; Shi, H.; Khreishah, A.; Ansari, N.; Lee, J.Y.; Liu, C. A two-tier edge computing based model for advanced traffic detection. In Proceedings of the 5th International Conference on Internet of Things: Systems, Management and Security, Valencia, Spain, 15–18 October 2018; pp. 208–215.
2. Wali, B.; Khattak, A.J.; Bozdogan, H.; Kamrani, M. How is driving volatility related to intersection safety? A Bayesian heterogeneity-based analysis of instrumented vehicles data. *Transp. Res. Part C Emerg. Technol.* **2018**, *92*, 504–524. [CrossRef]
3. Azimjonov, J.; Özmen, A. A real-time vehicle detection and a novel vehicle tracking systems for estimating and monitoring traffic flow on highways. *Adv. Eng. Inform.* **2021**, *50*, 101393. [CrossRef]
4. Tang, J.; Bi, W.; Liu, F.; Zhang, W. Exploring urban travel patterns using density-based clustering with multi-attributes from large-scaled vehicle trajectories. *Phys. A Stat. Mech. Its Appl.* **2021**, *561*, 125301. [CrossRef]
5. Dandala, T.T.; Krishnamurthy, V.; Alwan, R. Internet of Vehicles (IoV) for traffic management. In Proceedings of the 2017 International Conference on Computer, Communication and Signal Processing, Chennai, India, 10–11 January 2017; pp. 1–4.
6. Lee, H.Y.; Ho, H.W.; Zhou, Y. Deep learning-based monocular obstacle avoidance for unmanned aerial vehicle navigation in tree plantations. *J. Intell. Robot. Syst.* **2021**, *101*, 5. [CrossRef]
7. Velaga, N.R.; Quddus, M.A.; Bristow, A.L. Developing an enhanced weight-based topological map-matching algorithm for intelligent transport systems. *Transp. Res. Part C Emerg. Technol.* **2009**, *17*, 672–683. [CrossRef]
8. Toledo-Moreo, R.; Bétaille, D.; Peyret, F. Lane-level integrity provision for navigation and map matching with GNSS, dead reckoning, and enhanced maps. *IEEE Trans. Intell. Transp. Syst.* **2009**, *11*, 100–112. [CrossRef]
9. Szottka, I. Particle filtering for lane-level map-matching at road bifurcations. In Proceedings of the 16th International IEEE Conference on Intelligent Transportation Systems, Hague, The Netherlands, 6–9 October 2013; pp. 154–159.
10. Tao, Z.; Bonnifait, P.; Fremont, V.; Ibanez-Guzman, J. Lane marking aided vehicle localization. In Proceedings of the 16th International IEEE Conference on Intelligent Transportation Systems, Hague, The Netherlands, 6–9 October 2013; pp. 1509–1515.
11. Gu, Y.; Wada, Y.; Hsu, L.; Kamijo, S. Vehicle self-localization in urban canyon using 3D map based GPS positioning and vehicle sensors. In Proceedings of the 2014 International Conference on Connected Vehicles and Expo, Vienna, Austria, 3–7 November 2014; pp. 792–798.

12. Shunsuke, K.; Yanlei, G.; Hsu, L.T. GNSS/INS/on-board camera integration for vehicle self-localization in urban canyon. In Proceedings of the 2015 IEEE 18th International Conference on Intelligent Transportation Systems, Gran Canaria, Spain, 15–18 September 2015; pp. 2533–2538.
13. Zheng, Y.; Hansen, J.H.L. Lane-change detection from steering signal using spectral segmentation and learning-based classification. *IEEE Trans. Intell. Veh.* **2017**, *2*, 14–24. [CrossRef]
14. Kassas, Z.Z.M.; Maaref, M.; Morales, J.J.; Khalife, J.J.; Shamei, K. Robust vehicular localization and map matching in urban environments through IMU, GNSS, and cellular signals. *IEEE Intell. Transp. Syst. Mag.* **2020**, *12*, 36–52. [CrossRef]
15. Maaref, M.; Kassas, Z.M. Ground vehicle navigation in GNSS-challenged environments using signals of opportunity and a closed-loop map-matching approach. *IEEE Trans. Intell. Transp. Syst.* **2019**, *21*, 2723–2738. [CrossRef]
16. Newson, P.; Krumm, J. Hidden Markov map matching through noise and sparseness. In Proceedings of the 17th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems, Seattle, WA, USA, 4–6 November 2009; pp. 336–343.
17. Zeng, Z.; Zhang, T.; Li, Q.; Wu, Z.; Zou, H.; Gao, C. Curvedness feature constrained map matching for low-frequency probe vehicle data. *Int. J. Geogr. Inf. Sci.* **2016**, *30*, 660–690. [CrossRef]
18. Luo, A.; Chen, S.; Xv, B. Enhanced map-matching algorithm with a hidden Markov model for mobile phone positioning. *ISPRS Int. J. Geo-Inf.* **2017**, *6*, 327. [CrossRef]
19. Chen, C.; Ding, Y.; Xie, X.; Zhang, S. A three-stage online map-matching algorithm by fully using vehicle heading direction. *J. Ambient. Intell. Humaniz. Comput.* **2018**, *9*, 1623–1633. [CrossRef]
20. Lin, M.C.-H.; Huang, F.-M.; Liu, P.-C.; Huang, Y.-H.; Chung, Y.-S. Dijkstra-Based Selection for Parallel Multi-lanes Map-Matching and an Actual Path Tagging System. In Proceedings of the Asian Conference on Intelligent Information and Database Systems, Da Nang, Vietnam, 14–16 March 2016; pp. 499–508.
21. Ptošek, V.; Rapant, L.; Martinovič, J. Floating car data map-matching utilizing the Dijkstra's algorithm. In *Data Management, Analytics and Innovation*; Springer: Singapore, 2020; pp. 115–130.
22. Koller, H.; Widhalm, P.; Dragaschnig, M.; Graser, A. Fast hidden Markov model map-matching for sparse and noisy trajectories. In Proceedings of the 2015 IEEE 18th International Conference on Intelligent Transportation Systems, Gran Canaria, Spain, 15–18 September 2015; pp. 2557–2561.
23. 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; pp. 352–361.
24. Karduni, A.; Kermanshah, A.; Derrible, S. A protocol to convert spatial polyline data to network formats and applications to world urban road networks. *Sci. Data* **2016**, *3*, 160046. [CrossRef]
25. Nikolić, M.; Jović, J. Implementation of generic algorithm in map-matching model. *Expert Syst. Appl.* **2017**, *72*, 283–292. [CrossRef]
26. Yu, J.; Yang, Q.; Lu, J.; Han, J.; Peng, H. Advanced Map Matching Algorithms: A Survey and Trends. *Acta Electron. Sin.* **2021**, *49*, 1818–1829.
27. Huang, Z.; Qiao, S.; Han, N.; Yuan, C.A.; Song, X.; Xiao, Y. Survey on vehicle map matching techniques. *CAAI Trans. Intell. Technol.* **2021**, *6*, 55–71. [CrossRef]
28. Chao, P.; Xu, Y.; Hua, W.; Zhou, X. A survey on map-matching algorithms. In Proceedings of the Australasian Database Conference, Melbourne, VIC, Australia, 3–7 February 2020; Springer: Cham, Switzerland, 2020; pp. 121–133.
29. Yu, Q.; Hu, F.; Ye, Z.; Chen, C.; Sun, L.; Luo, Y. High-frequency trajectory map matching algorithm based on road network topology. *IEEE Trans. Intell. Transp. Syst.* **2022**, *3*, 1–16. [CrossRef]
30. Hsueh, Y.L.; Chen, H.C. Map matching for low-sampling-rate GPS trajectories by exploring real-time moving directions. *Inf. Sci.* **2018**, *433*, 55–69. [CrossRef]
31. Tanaka, A.; Tateiwa, N.; Hata, N.; Yoshida, A.; Wakamatsu, T.; Osafune, S.; Fujisawa, K. Offline map matching using time-expanded graph for low-frequency data. *Transp. Res. Part C Emerg. Technol.* **2021**, *130*, 103265. [CrossRef]
32. Chen, R.; Yuan, S.; Ma, C.; Zhao, H.; Feng, Z.-Y. THMM: A tailored hidden markov model optimized for cellular-based map matching. *IEEE Trans. Ind. Electron.* **2021**, *12*, 1–10. [CrossRef]