IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received February 20, 2020, accepted March 8, 2020, date of publication March 11, 2020, date of current version March 24, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.2980174

Density Adaptive Approach for Generating Road Network From GPS Trajectories

ZHONGLIANG FU[1,2], LIANG FAN[1], YANGJIE SUN[1], AND ZONGSHUN TIAN[3]
[1]School of Remote Sensing and Information Engineering, Wuhan University, Wuhan 430079, China
[2]Collaborative Innovation Centre of Geospatial Technology, Wuhan 430079, China
[3]School of Management Science and Real Estate, Chongqing University, Chongqing 400045, China

Corresponding author: Liang Fan (fanliang@whu.edu.cn)

ABSTRACT Road networks are fundamental parts of intelligent transportation and smart cities. With the emergence of crowdsourcing geographic data, road mapping approaches by using crowdsourcing trajectories have been developed. Existing road map inference algorithms from trajectories can extract relatively accurate road networks, however, these algorithms are not robust to different trajectory datasets and the parameter optimization task is tedious and time-consuming. Therefore, we propose an adaptive approach based on trajectory density. The proposed approach contains two stages. Firstly, the density distribution for each trajectory is adaptively estimated by the Gaussian fitting approach and the density peak points are extracted to construct road centerlines corresponding to each trajectory. Secondly, these extracted road centerlines are incrementally merged by the ''matching-refinement-merging'' process to generate a road network. We compare the proposed approach against four representative methods through trajectory datasets that are completely different in sampling frequency, trajectory density, road density, and noise. The results show that the proposed approach provides better accuracy in terms of precision and integrity and does not require additional parameter adjustment.

INDEX TERMS GPS trajectories, map inference, road networks, spatial data.

I. INTRODUCTION

Road networks are a fundamental part of the National Spatial Data Infrastructure (NSDI). It is widely used in traffic perception [1]–[3], navigation system [4], [5], vehicle behavior analysis [6], [7], and autonomous driving [8], [9]. With the rapid development of urban traffic, the coverage of road networks is getting wider and wider and the structure of road networks is changing frequently. Existing road maps no longer meet the high requirements for the accuracy, timeliness, and completeness of road data in the construction of intelligent transportation and smart cities. Therefore, constructing road network data with high requirements is a major challenge for map production departments.

The producing and updating mode of a road map is still dominated by professional surveying and mapping. Road data of the surveying and mapping method is collected through mobile surveying vehicles, remote sensing satellites, and unmanned aerial vehicles. Then professionals identify the shape of roads, extract the attribute information and construct the topological structure of a road network. The participation of professionals ensures sufficient quality of the generated road network. However, there are also problems such as difficulty in obtaining road network data, low update efficiency, and high production costs.

With the wide application of positioning systems and Internet technology in decades, crowdsourcing geographic data have been produced in daily life. These geographic data have the advantages of large data volume, timeliness, wide-coverage, and low cost, which complement the traditional professional surveying and mapping data. Based on these data, volunteered mapping has been developed. OpenStreetMap is the most extensive and effective volunteered mapping project [10]. It uses vehicle trajectory data, remote sensing images and out-of-copyright maps as references, produces and updates maps through manual editing by volunteers. However, the non-professional nature of volunteers has resulted in the production of road maps inevitably containing many errors, such as misaligned roads, wrong road names, outdated roads and so on. Therefore, additional

The associate editor coordinating the review of this manuscript and approving it for publication was Jing Bi.

quality checks are needed to ensure the reliability of the map. Crowdsourcing trajectories not only contain dynamic information of vehicle behavior but also potentially contain road network information. Therefore, a road map can also be automatically extracted by using crowdsourcing trajectories. The economical, timeliness and automated features of this method have attracted scholars to carry out many studies.

Robustness and adaptability are important indicators to measure the pros and cons of algorithms [11], [12]. Existing map inference algorithms with trajectories have been able to construct relatively complete road networks after researching for a decade. However, these algorithms are not adaptive for different trajectory datasets [13], [14]. Firstly, algorithm parameters need to be adjusted manually to achieve optimal results according to the trajectory density distribution, road width, positioning errors and other factors of different datasets. Improper parameter selection usually has a large impact on the final results. Secondly, trajectory distribution, road distribution, and positioning errors in the same dataset are also uneven. To ensure higher extraction precision, many algorithms directly discard the lower density trajectories as noise, thereby losing the integrity of the road network. On the other hand, although some algorithms can extract a complete road network, they are more sensitive to trajectory noise. The spatial heterogeneity of trajectories makes it difficult to extract road networks accurately and completely. Therefore, robustness, effectiveness, and flexibility are the main limitations of existing map inference algorithms. How to make full use of GPS trajectories with spatial heterogeneity to develop a more powerful and flexible road network generation method is still necessary.

In this paper, we propose a density-based approach to inference road network adaptively. Firstly, we assume that GPS trajectories on the same road follow a Gaussian distribution. Therefore, we can use local density values around each trajectory sample to fit the trajectory distribution on each road. Then peak points of the fitted Gaussian distribution are used to estimate the position of road centerlines corresponding to each trajectory. Finally, the estimated road centerlines are incrementally merged to construct a road network. The rest of the paper is organized as follows. In the next section, related studies will be introduced. Section 3 presents the details of the proposed method. In Section 4, the experimental results and comparative analysis are described. Finally, the conclusion and future work are given in Section 5.

II. RELATED WORK

The methods for automatically extracting road networks with crowdsourced GPS trajectories can be classified as four categories: clustering-based, image-based, intersection linking and trajectory incremental merging [15], [16].

The clustering-based method was first proposed by Edelkamp and Schrödl [17]. In their method, K-means clustering is first performed according to the position of trajectory samples. Each cluster center can be regarded as a node of roads and a complete road network is constructed by connecting the nodes. Since then, Zhu *et al*. proposed a clustering method for low-frequency trajectories based on line segments [18]. Compared with trajectory samples, the trajectory segments contain more features such as position, direction, and length. It can solve the problem of far distance between adjacent low-frequency trajectory samples. Dørum further proposed a direction-constrained clustering method based on a grid. It uses the directional characteristics of trajectory samples to generate a two-way road network [19]. Stanojevic *et al*. combined K-means clustering with network comparison methods and proposed two different road network extraction methods, offline and online [20].

The image-based method converts trajectory data into an image and uses image processing technologies to extract a road network. Davies *et al*. used kernel density estimation to generate a gray-scale image of the roads. Then they used image smoothing, binarization, and contour tracking method to extract road contours and produced Voronoi graph of points to find the centerlines of these road contours [21]. Considering that it is difficult to determine the threshold of image binarization, Biagioni and Eriksson proposed a gray-scale skeletonization approach to find road centerlines and eliminated redundant lines through density-aware map matching [22]. Wang *et al*. introduced the Morse theory to the extraction of road centerlines. Their method achieved good results in large-scale datasets [23].

In urban road networks, intersections have obvious turning characteristics. Therefore, Fathi and Krumm trained a circular descriptor to extract intersections, and then used trajectory segments connecting the intersections to generate road networks [24]. Karagiorgou and Pfoser further proposed an unsupervised intersection extraction method, which identified and clustered trajectory samples at intersections by the direction and speed of samples [25]. After that, they improved the accuracy of road map inference by layering trajectories according to the speed [26]. Wang *et al*. clustered trajectories by physical attraction model and extracted intersections by hotspot analysis. A routable road map was finally generated by connecting intersections [27]. Given the difficulty in extracting complex intersections, Deng *et al*. [28] and Huang *et al*. [29] used trajectory segments and turning angles to extract complex intersections from low-frequency trajectories, respectively.

The trajectory incremental merging approach assumes that there is a blank initial road map, and then uses a map matching method to merge trajectories with the road map one by one. After all the trajectories are processed, a complete road network can be generated [30]. Aiming at the problem of GPS error, Cao and Krumm proposed the physical attraction model to cluster trajectories to reduce errors [31]. Tang *et al*. used Delaunay triangulation to extract road network in a weighted merging manner, which further improved the accuracy of road network generation [32]. He *et al*. proposed a trajectory flow tracking approach to generate high precision road network under complex road scenes such as overpasses and parallel sections [33].

The above methods can effectively extract the road networks from trajectory datasets. However, the clustering-based and the image-based methods ignored the connectivity between observations in the same trajectory. The intersection linking method needs to extract accurate location and coverage of intersections. However, how to ensure the extraction accuracy in the face of complex scenarios is still challenging. In the method of trajectory incremental merging, the error generated by each merging process will gradually accumulate, which may affect the extraction accuracy of the final generated road network. Besides, all the above methods need to manually adjust parameters for different datasets and wrong parameter settings will have a great impact on the extraction results.

III. METHODOLOGY

In general, there is a certain deviation between a GPS trajectory and the corresponding road centerline even without location errors. On the other hand, the peak points of trajectory density tend to be distributed in the center of the road surface and the position of road centerline can be estimated by these density peak points. Because of this, unlike the common method for directly merging GPS trajectories to generate road network, we propose to use trajectory density to estimate road centerlines corresponding to each trajectory firstly and then merge the road centerlines to generate road network. The map inference architecture of our method consists of two stages: road centerline extraction and road centerline incremental merging.

Road Centerline Extraction: Firstly, the kernel density analysis is used to estimate trajectory density. Then based on the assumption that the trajectory density on the same road follows a Gaussian distribution, we use each trajectory sample and the density value of their neighborhood to fit a Gaussian distribution. The peak point of the Gaussian distribution corresponding to each trajectory samples can be linked to constructing a road centerline. Since the trajectory noise may result in wrong fitted peak points, we use a hidden Markov model (HMM) to eliminate these wrong peak points and construct road centerline segments corresponding to each trajectory.

Road Centerline Incremental Merging: The road network is generated by incrementally merging the extracted road centerlines with an initial blank road map. Firstly, each road centerline is matched with the road map to find the unmatched parts. Then we refine the unmatched road centerlines by the other road centerlines on the same road. After road centerline refinement, these refined road centerlines are merged with the road map in geometry and topology. When all of the road centerlines have been matched, refined and merged, the final road network is generated.

A. ROAD CENTERLINE EXTRACTION

1) TRAJECTORY DENSITY ESTIMATION

The trajectory density can be calculated by kernel density estimation (KDE). Similar to the KDE method for point data, the overall density distribution of trajectories is estimated by superposing the probability density distribution of each trajectory. The probability density value of a certain point on each trajectory will decrease as the distance between that point and the trajectory increases. We choose a Gaussian kernel function to represent the probability density distribution of each trajectory. The input trajectories can be represented by a series of polylines $T = \{T_1, T_2, \ldots, T_n\}$ and each node on T_i is the GPS observation of a trajectory. After dividing the area covered by trajectories into a grid, the kernel density of each cell can be calculated according to the definition of kernel density [34]:

$$D(x) = \sum_{i=1}^{n} K(d_i/h)/(nh) \tag{1}$$

where d_i is the distance from the center of the cell to trajectory T_i, $K(\cdot)$ is the Gaussian kernel function, and h is the bandwidth of Gaussian kernel function. After calculating the kernel density of all cells, a rasterized kernel density distribution image can be obtained. Fig. 1a shows the original trajectory data, and the corresponding kernel density estimate is shown in Fig. 1b. The density value for each cell is mapped to the color band. Higher values are colored with red while lower values are colored with green. Besides, blue lines in Fig. 1b are ground truth of the road network. It can be found that the peaks of the trajectory kernel density on each road coincide with the position of road centerlines. In the next section, we will extract road centerlines based on the trajectory kernel density distribution calculated here.

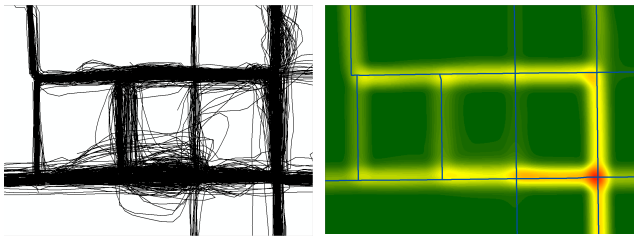


FIGURE 1. Kernel density estimate: (a) original GPS trajectories, (b) kernel density of the trajectories.

2) ROAD CENTERLINE ESTIMATION

Winden *et al.* conducted a statistical analysis and found that the distance from trajectory samples on the same road to the road centerline approximately follows a Gaussian distribution [35]. Therefore, when the density of a point on the trajectory and the density of its neighbors are known, the position of road centerline can be estimated by fitting a Gaussian distribution. Fig. 2 shows the road centerline estimation for a given trajectory by Gaussian distribution fitting. As shown in Fig. 2a, the black vertical lines are trajectories on the same road. The black point P is a sample on the trajectory. Blue points are the neighbors of P. Each blue point locates on the center of the cell which is constructed in trajectory density estimation. If the density at point P is D_p and the distance

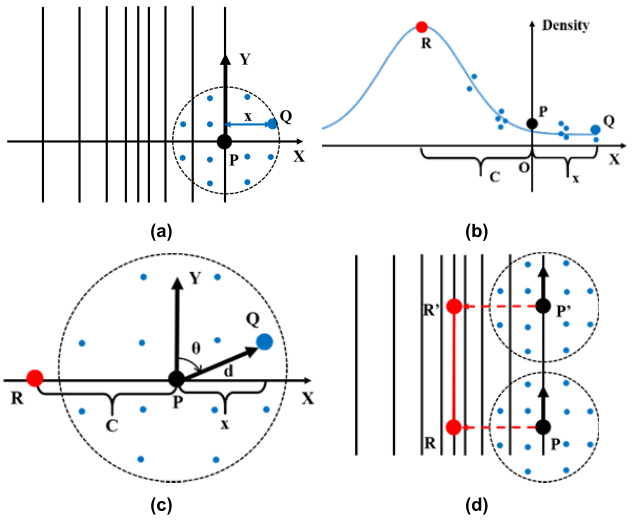


FIGURE 2. The road centerline estimation: (a) a trajectory sample and its neighbors, (b) Gaussian distribution fitting for trajectory density, (c) location of the density peak point, (b) road centerline construction for the corresponding trajectory.

from P to road centerline is C, D_p and C should satisfy the Gaussian distribution function (Fig. 2b). It can be represented as follows:

$$D_p = A \cdot exp(-C^2/B^2) \tag{2}$$

where A and B are parameters of a Gaussian distribution function. For a point Q in the neighborhood of P, its density can be expressed as:

$$D_q = A \cdot exp(-(x+C)^2/B^2) \tag{3}$$

where, $x = d \cdot sin\theta$, d is the distance between P and Q, θ is the angle between the trajectory direction of P and the vector PQ(Fig. 2c). We take clockwise rotation as the positive angle, and counterclockwise as the negative angle.

Take the logarithm of equations (2) and (3) and subtract them to eliminate A:

$$B^2\left(ln\, D_p - ln\, D_q\right) = 2Cx + x^2 \tag{4}$$

D_p and D_q in equation (4) can be obtained from kernel density distribution of trajectories, x is calculated from the positions and directions of PQ. Therefore, the only unknown variables in equation (4) are B and C. Using more than two neighboring points, B and C can be estimated. Then the parameter A of the Gaussian distribution function can be calculated by equation (2).

To ensure the accuracy of the fitting process, we select all cell centers that are less than ε from the point P as neighbor points in the kernel density image. The Gaussian distribution is fitted from the density and position of these neighbor points by the Ransac algorithm [36]. Firstly, we randomly select two neighbor points to calculate parameters A, B, and C by equation (2) and (4). The calculated parameters of Gaussian distribution are used to estimate the density of all cell centers that are selected. The cell centers with the estimation error less than O_c are regarded as inliers. Then, we repeat the above fitting process for N times and the Gaussian distribution model with the most inliers is selected as the final fitted model. Reference [36] shows the details of setting parameters O_c and N.

Since the density peak point R of the Gaussian distribution corresponds to a point on the road centerline, the spatial location of R can be calculated by the fitted distance C. It can be calculated by moving P along the vertical direction of the trajectory direction of P in a distance C (Fig. 2c). A series of trajectory density peak points can be extracted by using each trajectory sample and its trajectory direction to fit a Gaussian distribution. Each trajectory density peak point corresponds to a node on the road centerline. By connecting all of the density peak points extracted from samples on a trajectory in sequence, a road centerline corresponding to the trajectory can be extracted (Fig. 2d).

Because sparsely sampled trajectories show large geometric inconsistency, we resample trajectory points for trajectory datasets with low sampling frequency. Then, the resampled trajectory points are fitted with Gaussian distribution and road centerlines are extracted. Besides, when there is noise in the trajectory data, it is easy to extract wrong road centerlines. The reason is that noise trajectory samples may have wrong trajectory directions. Trajectory peak points estimated by the noise trajectory samples deviate from the real trajectory density peaks, which lead to wrong road centerline extraction. In the next section, these wrong density peaks will be eliminated.

3) WRONGLY EXTRACTED ROAD CENTERLINE ELIMINATION

There will be a certain error between the density of peak point estimated by the Gaussian distribution and the kernel density at that point. False density peak points can be eliminated by using a fixed error threshold, but the value of the error threshold is difficult to determine. If the error threshold is too small, some correctly fitted peaks will be eliminated, resulting in a large number of shorter road centerlines. On the other hand, if the error threshold is too large, some false density peak points will be missed. To eliminate false peak points as accurately as possible, the accuracy of each density peak point estimation is regarded as a hidden variable, and a hidden Markov model is used for modeling.

If the density of a node on a road centerline is accurately estimated, the error between the estimated density and the kernel density should be small. Meanwhile, the direction of the extracted road centerline should be consistent with the direction of the corresponding trajectory. The direction of a current node on a road centerline is related to the estimated position of the next node. Therefore, the accuracy of the density estimation of all nodes on the road centerline constitutes a Markov process.

HMM is a Markov process with a set of hidden states and observations [37]. The state-to-state transition is defined by the transition probability. Each state has an observation probability over possible observations. Given a series of

observations, corresponding hidden states can be generated by maximizing the overall probability. We take the accuracy of density estimation of each node on a road centerline as the hidden state, which can be expressed as the set $S = \{S_1, S_2\}$, where S_1 represents the accurate estimation and S_2 represents the wrong estimation. Observations corresponding to the hidden states are the density estimation error of each node, which is expressed by $O = \{O_1, O_2, ..., O_n\}$, where n is the total number of nodes on the road centerline. The observation probability corresponding to each hidden state can be expressed as:

$$\begin{cases} b_{1i} = P(o_i = O_i \mid s_i = S_1) = exp(-o_i^2/\delta^2) \\ b_{2i} = P(o_i = O_i \mid s_i = S_2) = 1 - exp(-o_i^2/\delta^2) \end{cases} \tag{5}$$

where o_i represents the density estimation error corresponding to the *ith* node, s_i represents the estimation accuracy of the *ith* node, and δ is a parameter for the observation probability distribution. δ can be determined by using the critical value of the interior fitting error in the Ransac fitting results. When the density estimation error is at the critical value O_c, the probability that it belongs to two different states should be equal. Therefore, δ can be calculated as follows:

$$\delta = \sqrt{o_c^2/ln2} \tag{6}$$

The transition probability between hidden states can be calculated using the difference between the direction of the road centerline and the direction of the trajectory. If T represents a GPS trajectory, and t_i, t_{i+1} represent adjacent trajectory samples on the trajectory. C represents the extracted road centerline corresponding to the trajectory, and c_i, c_{i+1} are adjacent nodes on the road centerline corresponding to t_i and t_{i+1}. Then the trajectory direction of t_i can be expressed as a vector t_it_{i+1}. The road centerline direction of c_i is related to the estimation accuracy of c_{i+1}. When c_{i+1} is accurately estimated, the road centerline direction of c_i is the vector c_ic_{i+1}. The transition probability can be expressed as:

$$\begin{cases} a_{11} = P(q_i = S_1 \mid q_{i+1} = S_1) = 1 - |cos(\theta_{tc})| \\ a_{12} = P(q_i = S_2 \mid q_{i+1} = S_1) = |cos(\theta_{tc})| \end{cases} \tag{7}$$

where θ_{tc} represents the angle between the vectors t_it_{i+1} and c_ic_{i+1}. When c_{i+1} is incorrectly estimated, we cannot use c_{i+1} to calculate the road centerline direction of c_i. Therefore, we use the Gaussian distribution parameter corresponding to c_i and the trajectory sample t_{i+1} to re-estimate the node c_{i+1} which can be represented as c'_{i+1} for convenience. Then, the road centerline direction of c_i is the vector $c_ic'_{i+1}$ and the state transition probability is:

$$\begin{cases} a_{21} = P(q_i = S_1 \mid q_{i+1} = S_2) = 1 - |cos(\theta_{tc'})| \\ a_{22} = P(q_i = S_2 \mid q_{i+1} = S_2) = |cos(\theta_{tc'})| \end{cases} \tag{8}$$

where θ'_{tc} represents the angle between the vectors t_it_{i+1} and $c_ic'_{i+1}$. When the density estimation error and the Gaussian distribution parameters corresponding to each node are known, hidden states can be solved by using the Viterbi algorithm. Thus, the estimation accuracy of each node on the road centerline can be calculated and wrongly estimated nodes are eliminated. After eliminating wrongly estimated nodes, a road centerline is divided into several sub road centerlines according to the eliminated nodes. We still refer these sub road centerlines as road centerlines for convenience in the following sections.

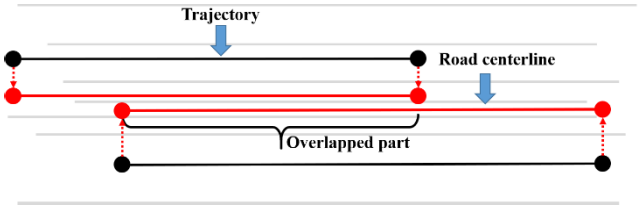


FIGURE 3. Road centerlines constructed by two trajectories.

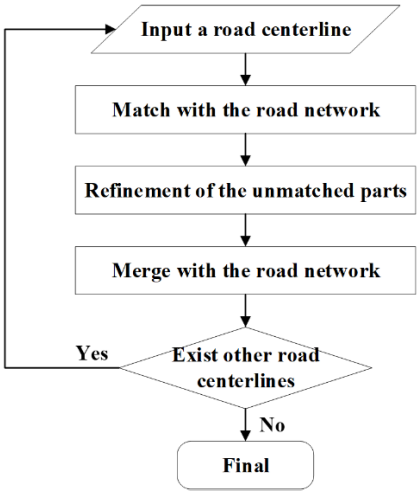


FIGURE 4. Flowchart of the road centerline incremental merge.

B. ROAD CENTERLINE INCREMENTAL MERGING

Since we can extract a corresponding road centerline from each trajectory, road centerlines extracted by multiple trajectories will overlap (Fig. 3). In this section, the road network is generated by merging road centerlines. We modify the incremental merging approach for trajectories to merge the constructed road centerlines. Fig. 4 shows the flowchart of the incremental merging process. The process begins with an initial blank road network and merges each road centerline with the road network iteratively. The difference from original incremental merging methods is that we add the step of road centerline refinement before merging. The refinement process can ensure the accuracy of road centerlines to be merged and can reduce the cumulative error during the incremental merging process.

1) ROAD CENTERLINE MATCHING

Map matching method can be used to extract the road sections that have been merged and the road sections to be merged. The matching of road centerlines is similar to the

matching of trajectories. Thus, existing map matching methods for trajectories can also be used for road centerlines. We select the progressive matching method proposed by Brakatsoulas [38]. This method can simultaneously use geometric and topological features of a road network and has high matching efficiency and accuracy for massive data. To adapt to the situation that there may be several unmatched segments on a road centerline, we adjust the node matching process in the method of Brakatsoulas. A more formal description is shown in Algorithm 1 below.

Given a road centerline and a road network represented by straight road segments, the type of current node to be matched is determined firstly according to the matching result of the previous node. Then node matching is divided into two cases according to different node types: initial node matching and successor node matching.

a: INITIAL NODE MATCHING

If the current node to be matched is the first node on a road centerline or the previous node is an unmatched node, the current node is an initial node. We use a distance threshold Dt to select road segments whose distance from the initial node is less than Dt as candidate matching segments. If there are multiple candidates matching segments, the segment closest to the initial node is selected as the matching segment. Then the projection point of the initial node on the matching segment is calculated as the matching point. If no candidate matching segment exists, the node is unmatched. We use the maximum distance between the road centerline and the corresponding trajectory as the matching distance threshold for all nodes on the road centerline.

b: SUCCESSOR NODES MATCHING

If the previous node is a matching node, the current node is a successor node. Matching segments of the current node can be dynamically searched by using the matching segment of the previous node.

As shown in Fig. 6a, the connected solid lines are road segments, and the dashed line represents a road centerline to be matched. Among them, p_1 is the initial node, p_2 and p_3 are the successor nodes that have been matched, and p_4 is the successor node to be matched. Then we can match p_4 by the following steps.

a) The candidate road segment set corresponding to the node p_3 is taken as the candidate road segment set of p_4: $G = \{e_2, e_3\}$.

b) A matching path $L = \{e_1\}$ is constructed using matching segments from node p_1 to node p_2. Then a candidate matching path set $E = \{e_1e_2, e_1e_3\}$ is composed of a matching path L and each candidate matching segments, as shown in Fig. 6b and Fig. 6c.

c) Calculate the Fréchet distance between each candidate matching path and the road centerline $C = \{p_1, p_2, p_3, p_4\}$. Then the path with the smallest Fréchet distance is selected as S. If the Fréchet distance between S and C is larger than Dt, p_4 is unmatched. Then continue to match the next node on the road centerline. Otherwise, go to step d. In Fig. 6b and Fig. 6c, the candidate matching path e_1e_3 is selected.

d) Calculate the closest point from the matching path to p_4. Decide whether it is necessary to continue searching for candidate matching road segments by using the position of the closest point on the matching path. If the closest point is at the tail node of the matching path, go to step e. Otherwise, go to step f.

Algorithm 1 Road centerline matching

Input: a road centerline *Rc,* road network represented by line segments *Rn*
Output: matched line segments on the road network *Mc*

```
Pn ← previous node of the current matching node
Dt ← distance threshold
Mc ← Ø
for each node Cn on Rc do
    Nt ← getNodeType(Cn, Pn)
    if Nt is an initial node, then
        Cs ← getCandidateSegments(Cn, Rn, Dt)
        if count(Cs) > 0, then
            Ms ← getClosestSegments(Cn, Cs)
            Mc.addSegment(Ms)
        end if
    else
        Cs ← getCandidateSegmentsOfPrevNode(Pn)
        Cp ← constructCandidatePaths(Cs, Rn)
        Mp ← getClosestPath(Cp, Rc)
        if fréchetDistance(Mp, Rc) < Dt, then
            Pt ← getClosestPoint(Cn, Mp)
            if Pt is at the tail of Mp, then
                Cs ← updateCandidateSegments(Mp)
                go to line 14
            else
                Ms ← getSegment(Pt, Mp)
                Mc.addSegment(Ms)
                updatePrevMatchSegment(Pn, Mp, Mc)
            end if
        end if
    end if
end for
```

FIGURE 5. Road centerline matching algorithm.

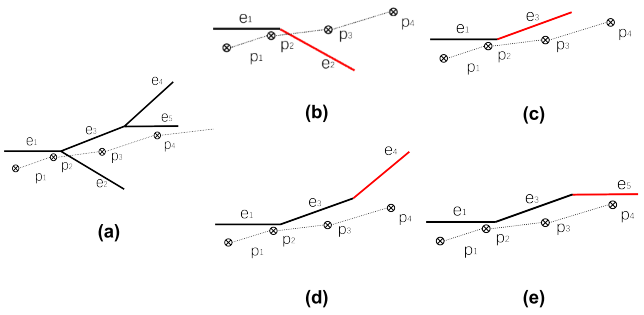


FIGURE 6. The process of road centerline matching.

e) The candidate road segment set of p_4 is updated to the segments connected to the matching path: $G = \{e_4, e_5\}$, as shown in Fig. 6d and Fig. 6e. If the candidate segment set is empty, p_4 is unmatched. Otherwise, steps b-d are repeated.

f) If the closest point is not at the tail of the matching path, the closest point is regarded as the matching point of p_4 and the road segment where the closest point is located is regarded as the matching road segment. In Fig. 6d and Fig. 6e, the candidate matching path $e_1e_3e_5$ is selected and e_5 is the matching segment of p_4. Then we check whether the matching segment of node p_3 is on the selected matching path. If it is not located on the matching path, the matching point and matching road segment of p_3 are updated by the matching path.

After all the nodes on the road centerline are matched, we connect the successive unmatched nodes to construct road centerlines to be merged in the next section.

2) REFINEMENT OF UNMATCHED ROAD CENTERLINES

After matching road centerlines, we can merge the unmatched parts with the road map. Considering the estimation error of extracted road centerlines, we do not directly merge the unmatched road centerline with the road map. By using all of the extracted road centerlines on the same road, we can move the unmatched road centerline to the optimal position. In this way, the positional error accumulation can be reduced during the road centerline incremental merging process.

The mean shift algorithm proposed by Comaniciu and Meer [39] is a non-parametric feature space analysis method for finding the maximum density. This method iteratively updates the position of the centroid in the neighborhood. When the position of the centroid is stable, the stable centroid is the density maximum point:

$$p_{i+1} = \sum_{p_j \in N} p_j exp(-\frac{1}{2}(\frac{p_j - p_i}{h})^2) / \sum_{p_j \in N} exp(-\frac{1}{2}(\frac{p_j - p_i}{h})^2) \quad (9)$$

where N is the neighborhood of the centroid point, p_i represents the position of the centroid point after the ith iteration, p_j is one of the samples in the neighborhood of p_i, and h is the neighborhood radius. We choose the mean shift algorithm to move unmatched road centerlines to the position with a maximum density which is regarded as the optimal position of road centerlines. Details of the refinement process are shown in Algorithm 2.

For each node on the unmatched road centerline, it is regarded as a centroid point and iteratively shifted using the mean shift algorithm. The iteration stops when the shift distance of the centroid point is less than 1 meter. To find neighbor nodes in the mean shift algorithm, we choose the distance threshold in the road centerline matching process as the neighborhood radius. Since the road centerlines with different directions at an intersection will affect each other, we only select similar nodes in the neighbor nodes to participate in the shift step during each iteration. The nodes with a directional difference less than 45 degrees from the direction of the current node to move are selected as similar nodes. Besides, to ensure that the node of a road centerline can iterate to the optimal position quickly, we only shift the centroid point along the perpendicular direction of the direction of a road centerline during each iteration.

Algorithm 2 Refinement of an unmatched road centerline

Input: an unmatched road centerline *UC*, all of the nodes on extracted road centerlines *AN*
Output: refined road centerline *RC*

```
h ← neighborhood radius
RC ← Ø
for each node ND on UC do
    i ← 0
    p_i ← position of ND
    d ← direction of ND
    do while distance between p_{i-1} and p_i > 1 meter
        NP ← getNeighborPoints(p_i, h, AN)
        SP ← getSimilarNeighborPoints(d, NP)
        p_i' ← calMeanPosition(p_i, h, SP)
        p_{i+1} ← shiftPoint(p_i, p_i', d)
        i ← i + 1
    end while
    RC.addNode(p_i)
end for
```

FIGURE 7. Algorithm of unmatched road centerline refinement.

3) GEOMETRIC MERGING AND TOPOLOGY MODIFICATION

After the refinement of unmatched road centerlines, these unmatched parts have high positional accuracy. Therefore, we merge these unmatched parts with the road map in this section. The merging process mainly includes two steps: geometric merging and topology node modification.

a: GEOMETRIC MERGING

Firstly, we select matching nodes adjacent to the head and tail nodes of the unmatched road centerline. Then the head and tail nodes of the unmatched road centerline are projected onto the matching road segments corresponding to the selected matching nodes. We connect the head and tail nodes with their corresponding projection points respectively to merge the unmatched road centerline with the road network. Fig. 8a shows the process of geometric merging. The road segment indicated by the red line is the segment integrated into the road network.

b: TOPOLOGY NODE MODIFICATION

After the geometric merging of unmatched road centerlines, redundant nodes are easily generated at a road intersection. As shown in the road network indicated by the black line in Fig. 8b, the dot and the square in the dashed circle are nodes representing the same intersection. To improve the location accuracy of intersections and eliminate redundant nodes, the

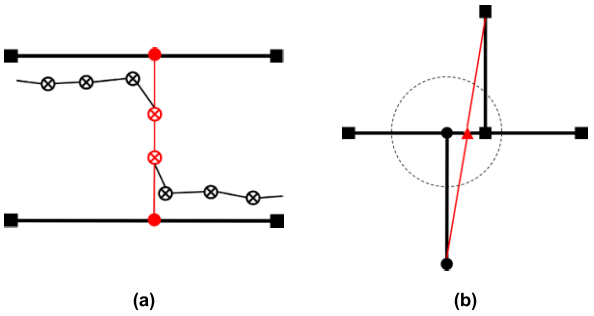


FIGURE 8. Unmatched road centerline merging: (a) geometric merging, (b) topology node modification.

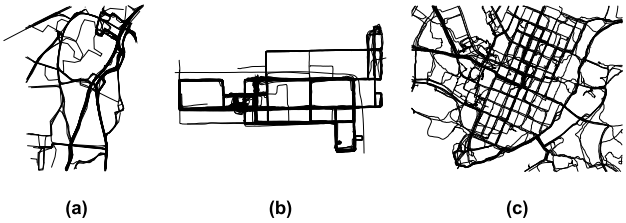


FIGURE 9. Trajectory datasets: (a) Athens, (b) Chicago, (c) Joensuu.

TABLE 1. Statistics of trajectory datasets.

	Athens	Chicago	Joensuu
Routes	129	889	109
Points	2840	118237	43632
Sampling Rate(s)	34.07	3.61	2
Trajectory Length(km)	443	2869	250
Speed(km/h)	19.55	33.14	9.1
Area(km^2)	2.6×6	4.5×2.5	2.8×2.2

topological relationship of the road network at intersections needs to be modified. Firstly, we check whether there is a topology node in the neighborhood of the connection node which connects the newly added road segment with the original road network. The nodes with a degree of connectivity not equal to 2 are considered as topological nodes, and the distance threshold during road centerline matching is selected as the neighborhood radius. If there are topology nodes, the median position of the topology nodes is calculated. Then we move the connection node and the topology nodes to the median position and delete redundant nodes that overlay in the same position. As shown in Fig. 8b, the red triangle is the updated topology node, and the red lines are the adjusted road segments.

IV. EXPERIMENTAL ANALYSIS

A. DATASETS

We select three different trajectory datasets from Athens, Chicago, and Joensuu (Fig. 9) to evaluate the proposed road network generation algorithm. Table 1 shows the statistics of three trajectory datasets. Among them, the Athens and Chicago datasets are two different campus shuttle trajectories with low sampling frequency and high sampling frequency, respectively. These two datasets can be obtained through the website http://mapconstruction.org. The Joensuu dataset is jogging trajectories collected at high frequencies from smartphone applications, which is provided by Mariescu-Istodor and Cellnet [40] (http://cs.uef.fi/mopsi/routes/network). The coverage areas of three trajectory datasets are similar. Besides, these trajectory datasets are often used for verification of road network generation algorithms.

The roads covered by the trajectories in Athens dataset are irregularly distributed, whereas Chicago and Joensuu datasets both contain visible urban block structures. On the other hand, compared with Athens and Chicago datasets, Joensuu dataset contains more road intersections and the roads are more densely distributed. From the perspective of the trajectory density distribution, Chicago dataset has a higher trajectory density, whereas Athens and Joensuu datasets cover relatively few trajectories on each road. Besides, due to the influence of high-rise buildings in Chicago dataset, some areas contain large trajectory noise. Because of the differences in the road distribution, road structure, road density, and trajectory density of the three datasets, whether the proposed approach can adaptively generate the complete road network from trajectories will be the main verification goal.

B. ROAD NETWORK GENERATION RESULTS AND ANALYSIS

This section compares the proposed method with four road network generation methods, i.e. Stanojevic's method [20], Biagioni's method [22], Huang's method [29], and Ahmed's method [30]. The four comparison methods represent the clustering-based method, image-based method, intersection linking method, and trajectory incremental merging method, respectively. To adjust to different datasets, we vary the parameters based on the original parameter settings in the four comparison methods and referring to the parameter settings in Ahmed's review literature [16]. Then we select the parameters with the best results for each dataset. Since this parameter setting process is manual, better quality may be achieved; however, the optimization task is tedious and time-consuming. Different from the comparison algorithms, the proposed algorithm used the same parameter settings for each dataset. Fig. 10 shows the experimental results of the proposed method and comparative methods. The parameter settings of each algorithm are shown in Table 2.

By comparing the road network generation results of each method in Fig. 10, it can be found that Stanojevic's method can extract relatively complete road networks in different datasets. However, it is easy to generate redundant road segments, i.e. the generated road network in the Joensuu dataset. The reason is that the distribution of trajectory samples in the Joensuu dataset is relatively scattered. It is difficult to cluster all trajectory samples accurately using the K-means clustering method. Biagioni's method performs well in the

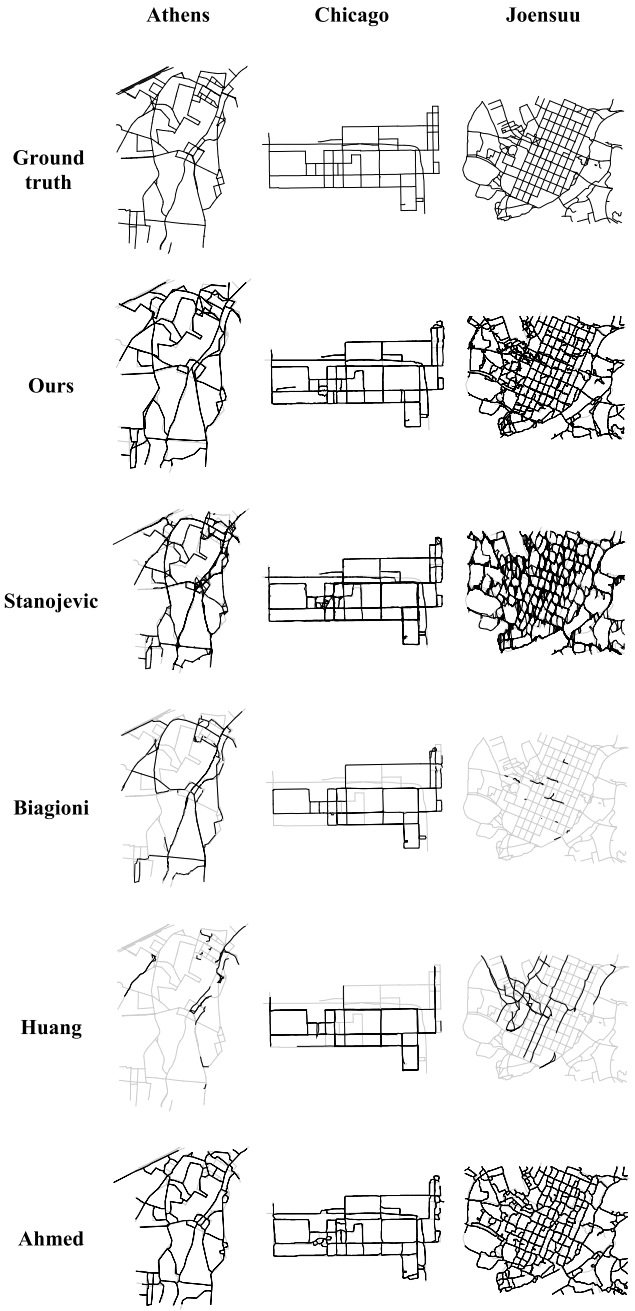


FIGURE 10. Results of road network extraction.

Athens and Chicago datasets. However, it is almost impossible to extract roads from the Joensuu dataset. The main reason is that trajectory density of adjacent roads in Athens and Chicago datasets are highly differentiated, whereas the road distribution is dense and the differentiation of trajectory density on adjacent roads is low in Joensuu dataset. Therefore, the kernel density of trajectories on adjacent roads is connected. Using a skeleton extraction approach can hardly generate accurate road centerlines. Huang's method should ensure that the turning characteristics of trajectories at intersections are obvious, and the turning points need to be clustered accurately to extract the location and coverage

TABLE 2. Parameter setting of the algorithms.

Method	Parameter	Athens	Chicago	Joensuu
Ours	cell size	1	1	1
	epsilon	25	25	25
Stanojevic (2018)	heading tolerance	45	60	45
	densification distance	20	20	20
	clustering radius	75	100	150
Biagioni (2012)	cell size	1	1	1
	kernel bandwidth	15	17	25
Huang (2019)	epsilon for major intersections	50	50	80
	minimum samples for major intersections	10	20	10
	epsilon for minor intersections	30	30	50
	minimum samples for minor intersections	3	10	5
Ahmed (2012)	epsilon	90	80	100

of intersections. Therefore, it generated a relatively complete road network in the Chicago dataset, while fewer road segments can be extracted in Athens and Chicago datasets. Ahmed's method has well precision and completeness in extracting road networks, but it is sensitive to trajectory noise. For example, it generated more incorrect road segments in the area with densely distributed trajectories in Chicago dataset. The proposed method can also extract accurate and complete road networks. Compared with the results of Ahmed's method, the proposed method has a better tolerance to trajectory noise. Meanwhile, the road networks extracted by the proposed method contains few redundant road segments.

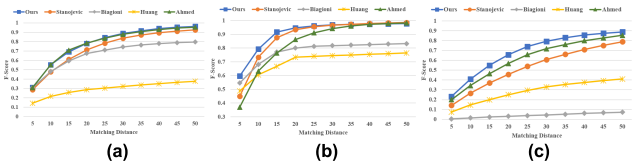
To compare the road network extraction results of various methods quantitatively, the TOPO method proposed by Biagioni is used to evaluate the accuracy of road network extraction [22]. The TOPO method takes both geometric accuracy and topological accuracy of road network extraction into account. This method first generates a certain number of random sampling points in the study area. Then it calculates all the paths that can be reached in the neighborhood of the sampling points in both the extracted road network and the real road network. These paths are resampled with an equal distance. The matching precision and recall are calculated by matching the resampled path nodes between the extracted road network and the real road network. Then F-score of road network extraction is further calculated by using equation (10). Fig. 11 shows the relationship between F-score and the matching distance in Athens, Chicago, and Joensuu datasets for each road network generation method. Table 3 shows the F-score, recall, and precision of each method when the matching distance is 20 meters.

$$FScore = 2 \cdot (precision \cdot recall)/(precision + recall) \quad (10)$$

Comparing the F-score of each method, it can be seen that the proposed method and Ahmed's method are the most accurate and followed by Stanojevic's method and Biagioni's

TABLE 3. Evaluation of the algorithms with the matching distance of 20 meters.

	Athens			Chicago			Joensuu		
	Recall	Precision	F-Score	Recall	Precision	F-Score	Recall	Precision	F-Score
Ours	**0.765**	0.797	0.781	**0.952**	0.942	**0.947**	**0.756**	**0.581**	**0.657**
Stanojevic	0.682	0.747	0.713	0.933	0.936	0.934	0.499	0.422	0.458
Biagioni	0.543	0.892	0.675	0.678	0.977	0.800	0.017	0.504	0.032
Huang	0.169	**0.937**	0.287	0.580	**0.997**	0.733	0.162	0.552	0.250
Ahmed	0.741	0.833	**0.785**	0.868	0.856	0.862	0.604	0.537	0.568



(a) (b) (c)

FIGURE 11. F-score of the proposed method and comparative methods: (a) Athens, (b) Chicago, (c) Joensuu.

TABLE 4. Algorithm running time (minutes).

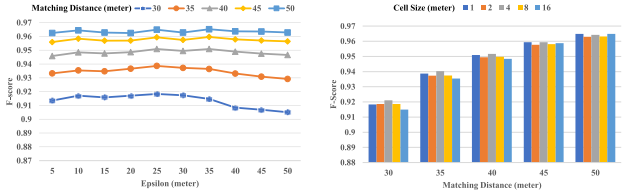
Method	Athens	Chicago	Joensuu
Ours (C++)	10.34	31.26	14.34
Stanojevic (Python)	0.22	9.08	3.35
Biagioni (Python)	16.19	33.15	16.84
Huang (Python)	1.55	4.84	2.67
Ahmed (Java)	0.17	17.12	8.62

method in Athens dataset. Huang's method extracts the most precise road segments, but the F-score is lower due to incompletion of the extracted road network. In Chicago dataset, all the methods have high accuracy. The road network extracted by the proposed method and Stanojevic's method has better integrity and road segments extracted by Huang's method and Biagioni's method are more precise. In Joensuu dataset, the proposed method has significant road network extraction accuracy compared with other methods. This improvement benefits from the high recall of the proposed method, especially in the densely distributed road regions. However, the road networks in the Athens and Chicago datasets are sparsely distributed. Thus, the improvements of the recall are not significant enough compared with other methods in Athens and Chicago datasets.

Due to these algorithms being implemented based on different coding languages (i.e., Java, Python, and C++), the algorithms' running times are not comparable in theory. However, to at least give an impression, Table 4 shows the respective running times of these algorithms on the three trajectory datasets. All these algorithms were run on Intel Core i7 CPUs running at 2.6 GHz with 8 GB of RAM using a Windows 10 operating system.

C. PARAMETER SENSITIVITY TEST

Parameters involved in the proposed algorithm mainly include the neighborhood radius ε in the density distribution



(a) (b)

FIGURE 12. Parameter sensitivity test on the Athens dataset: (a) the sensitivity of neighborhood radius, (b) the sensitivity of cell size.

fitting process and the cell size in the kernel density estimation process. To analyze the effect of different parameter values on the performance of our map inference algorithm, we used the Athens dataset for a parameter sensitivity test. Some results of this analysis are shown in Fig. 12.

When the neighborhood radius ε ranges from 5 meters to 50 meters at intervals of 5 meters, we calculate the F-score of the generated road networks with the matching threshold ranges from 35 meters to 50 meters. As shown in Fig. 12a, the curves in the figure represent the F-score with the matching distances of 30, 35, 40, 45, and 50 meters, respectively. It can be found that the F-score of the generated road networks is relatively high and stable. Fig. 12b shows the F-score of the generated road networks at different cell sizes. It can be found that the proposed algorithm is robustness even if the value of cell size changes greatly. Although the geometry of the extracted road centerlines is not accurate when the cell size is large, the road centerline refinement process can improve the positional accuracy during road centerline merging.

V. CONCLUSION AND FUTURE WORK

Based on the assumption that trajectories follow a Gaussian distribution on the road, we propose a two-stage method that extracting road centerlines and generating road network by incrementally merging road centerlines. This method can adaptively extract road centerlines corresponding to each trajectory according to the trajectory density. Furthermore, it evaluates the extraction results of road centerlines through the Hidden Markov model and eliminates the wrongly extracted road centerline segments. Then the map matching approach and the road centerline refinement approach are used to incrementally merge road centerlines to generate a road network. Compared with the existing road network generation methods, the proposed method is more robust and

does not require additional parameter adjustment. We evaluate our method by three trajectory datasets that are completely different in sampling frequency, trajectory density, road density, and trajectory type. Compared with four representative methods, it can be found that our method has advantages in the precision and integrity of road network extraction.

Although the proposed method can extract relatively complete road networks in different trajectory datasets, there are still many redundant road segments in the extracted road networks. In terms of road geometry and topological connections, the road network generated by our algorithm still has a large gap compared with the real road network, and it is difficult to accurately extract details of road networks such as highway interchanges and overpasses. Therefore, generating lane-level road networks and highly detailed road networks will be further researched in future work.

REFERENCES

[1] B. Piccoli, K. Han, T. L. Friesz, T. Yao, and J. Tang, "Second-order models and traffic data from mobile sensors," *Transp. Res. C, Emerg. Technol.*, vol. 52, pp. 32–56, Mar. 2015.

[2] Z. Li, Y. Zhu, H. Zhu, and M. Li, "Compressive sensing approach to urban traffic sensing," in *Proc. 31st Int. Conf. Distrib. Comput. Syst.*, Jun. 2011, pp. 889–898.

[3] S. Abdul Rahman, A. Mourad, M. El Barachi, and W. A. Orabi, "A novel on-demand vehicular sensing framework for traffic condition monitoring," *Veh. Commun.*, vol. 12, pp. 165–178, Apr. 2018.

[4] A. Dijkstra, "Assessing the safety of routes in a regional network," *Transp. Res. C, Emerg. Technol.*, vol. 32, pp. 103–115, Jul. 2013.

[5] X. Xie, K. Wong, H. Aghajan, "Smart route recommendations based on historical GPS trajectories and weather information," Mobile Ghent, Tech. Rep., 2013.

[6] J. Kim and H. S. Mahmassani, "Spatial and temporal characterization of travel patterns in a traffic network using vehicle trajectories," *Transp. Res. Procedia*, vol. 9, pp. 164–184, Jan. 2015.

[7] V. Havyarimana, Z. Xiao, A. Sibomana, D. Wu, and J. Bai, "A fusion framework based on sparse Gaussian–Wigner prediction for vehicle localization using GDOP of GPS satellites," *IEEE Trans. Intell. Transp. Syst.*, vol. 21, no. 2, pp. 680–689, Feb. 2020.

[8] P. Bender, J. Ziegler, and C. Stiller, "Lanelets: Efficient map representation for autonomous driving," in *Proc. IEEE Intell. Vehicles Symp. Proc.*, Jun. 2014, pp. 420–425.

[9] G.-P. Gwon, W.-S. Hur, S.-W. Kim, and S.-W. Seo, "Generation of a precise and efficient lane-level road map for intelligent vehicle systems," *IEEE Trans. Veh. Technol.*, vol. 66, no. 6, pp. 4517–4533, Jun. 2017.

[10] M. Haklay and P. Weber, "OpenStreetMap: User-generated street maps," *IEEE Pervas. Comput.*, vol. 7, no. 4, pp. 12–18, Oct. 2008.

[11] H. Liu, J. Lu, M. Guo, S. Wu, and J. Zhou, "Learning reasoning-decision networks for robust face alignment," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 42, no. 3, pp. 679–693, Mar. 2020.

[12] G. Wang, J. Qiao, J. Bi, Q.-S. Jia, and M. Zhou, "An adaptive deep belief network with sparse restricted boltzmann machines," *IEEE Trans. Neural Netw. Learn. Syst.*, to be published, doi: 10.1109/TNNLS.2019.2952864.

[13] K. Buchin, M. Buchin, D. Duran, B. T. Fasy, R. Jacobs, V. Sacristan, R. I. Silveira, F. Staals, and C. Wenk, "Clustering trajectories for map construction," in *Proc. 25th ACM SIGSPATIAL Int. Conf. Adv. Geographic Inf. Syst. (SIGSPATIAL)*. New York, NY, USA: ACM, 2017, p. 14.

[14] D. Duran, V. Sacristán, and R. I. Silveira, "Map construction algorithms: An evaluation through hiking data," in *Proc. 5th ACM SIGSPATIAL Int. Workshop Mobile Geographic Inf. Syst.* New York, NY, USA: ACM, 2016, pp. 74–83.

[15] J. Biagioni and J. Eriksson, "Inferring road maps from global positioning system traces: Survey and comparative evaluation," *Transp. Res. Rec., J. Transp. Res. Board*, vol. 2291, no. 1, pp. 61–71, Jan. 2012.

[16] M. Ahmed, S. Karagiorgou, D. Pfoser, and C. Wenk, "A comparison and evaluation of map construction algorithms using vehicle tracking data," *GeoInformatica*, vol. 19, no. 3, pp. 601–632, Jul. 2015.

[17] S. Edelkamp and S. Schrödl, "Route planning and map inference with global positioning traces," in *Computer Science in Perspective*. Berlin, Germany: Springer, 2003, pp. 128–151.

[18] Y. Zhu, X. Liu, and P. Wang, "Road recognition using big data of coarse-grained vehicular footprints," in *Proc. IEEE 23rd Int. Conf. Parallel Distrib. Syst. (ICPADS)*, Dec. 2017, pp. 570–577.

[19] O. H. Dørum, "Deriving double-digitized road network geometry from probe data," in *Proc. 25th ACM SIGSPATIAL Int. Conf. Adv. Geographic Inf. Syst. (SIGSPATIAL)*. New York, NY, USA: ACM, 2017, p. 15.

[20] R. Stanojevic, S. Abbar, and S. Thirumuruganathan, "Robust road map inference through network alignment of trajectories," in *Proc. SIAM Int. Conf. Data Mining. Soc. Ind. Appl. Math.*, 2018, pp. 135–143.

[21] J. J. Davies, A. R. Beresford, and A. Hopper, "Scalable, distributed, real-time map generation," *IEEE Pervas. Comput.*, vol. 5, no. 4, pp. 47–54, Oct. 2006.

[22] J. Biagioni and J. Eriksson, "Map inference in the face of noise and disparity," in *Proc. 20th Int. Conf. Adv. Geographic Inf. Syst. (SIGSPATIAL)*, New York, NY, USA: ACM, 2012, pp. 79–88.

[23] S. Wang, Y. Wang, and Y. Li, "Efficient map reconstruction and augmentation via topological methods," in *Proc. 23rd SIGSPATIAL Int. Conf. Adv. Geographic Inf. Syst. (GIS)*, New York, NY, USA: ACM, 2015, p. 25.

[24] A. Fathi and J. Krumm, "Detecting road intersections from GPS traces," in *Proc. Int. Conf. Geographic Inf. Sci.* Berlin, Germany: Springer, 2010, pp. 56–69.

[25] S. Karagiorgou and D. Pfoser, "On vehicle tracking data-based road network generation," in *Proc. 20th Int. Conf. Adv. Geographic Inf. Syst. (SIGSPATIAL)*, New York, NY, USA: ACM, 2012, pp. 89–98.

[26] S. Karagiorgou, D. Pfoser, and D. Skoutas, "A layered approach for more robust generation of road network maps from vehicle tracking data," *ACM Trans. Spatial Algorithms Syst.*, vol. 3, no. 1, pp. 1–21, May 2017.

[27] J. Wang, X. Rui, X. Song, X. Tan, C. Wang, and V. Raghavan, "A novel approach for generating routable road maps from vehicle GPS traces," *Int. J. Geograph. Inf. Sci.*, vol. 29, no. 1, pp. 69–91, Jan. 2015.

[28] M. Deng, J. Huang, Y. Zhang, H. Liu, L. Tang, J. Tang, and X. Yang, "Generating urban road intersection models from low-frequency GPS trajectory data," *Int. J. Geograph. Inf. Sci.*, vol. 32, no. 12, pp. 2337–2361, Dec. 2018.

[29] Y. Huang, Z. Xiao, X. Yu, D. Wang, V. Havyarimana, and J. Bai, "Road network construction with complex intersections based on sparsely sampled private car trajectory data," *ACM Trans. Knowl. Discovery from Data*, vol. 13, no. 3, pp. 1–28, Jun. 2019.

[30] M. Ahmed and C. Wenk, "Constructing street networks from GPS trajectories," in *Proc. Eur. Symp. Algorithms*. Berlin, Germany: Springer, 2012, pp. 60–71.

[31] L. Cao and J. Krumm, "From GPS traces to a routable road map," in *Proc. 17th ACM SIGSPATIAL Int. Conf. Adv. Geographic Inf. Syst. (GIS)*. New York, NY, USA: ACM, 2009, pp. 3–12.

[32] L. Tang, C. Ren, Z. Liu, and Q. Li, "A road map refinement method using delaunay triangulation for big trace data," *ISPRS Int. J. Geo-Inf.*, vol. 6, no. 2, p. 45, 2017.

[33] S. He, F. Bastani, S. Abbar, M. Alizadeh, H. Balakrishnan, S. Chawla, and S. Madden, "RoadRunner: Improving the precision of road network inference from GPS trajectories," in *Proc. 26th ACM SIGSPATIAL Int. Conf. Adv. Geographic Inf. Syst. (SIGSPATIAL)*, New York, NY, USA: ACM, 2018, pp. 3–12.

[34] P. J. Green, A. H. Seheult, and B. W. Silverman, "Density estimation for statistics and data analysis," *Appl. Statist.*, vol. 37, no. 1, p. 120, 1988.

[35] K. van Winden, F. Biljecki, and S. van der Spek, "Automatic update of road attributes by mining GPS tracks," *Trans. GIS*, vol. 20, no. 5, pp. 664–683, Oct. 2016.

[36] M. A. Fischler and R. C. Bolles, "Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography," *Commun. ACM*, vol. 24, no. 6, pp. 381–395, 1981.

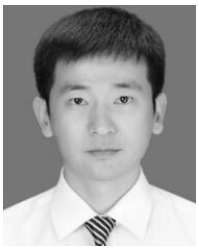
[37] P. Newson and J. Krumm, "Hidden Markov map matching through noise and sparseness," in *Proc. 17th ACM SIGSPATIAL Int. Conf. Adv. Geographic Inf. Syst. (GIS)*. New York, NY, USA: ACM, 2009, pp. 336–343.

[38] S. Brakatsoulas, D. Pfoser, and R. Salas, "On map-matching vehicle tracking data," in *Proc. 31st Int. Conf. Very Large Data Bases. VLDB Endowment*, 2005, pp. 853–864.

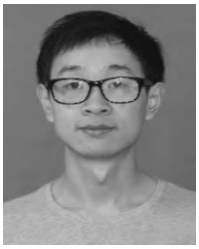
[39] D. Comaniciu and P. Meer, "Mean shift: A robust approach toward feature space analysis," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 24, no. 5, pp. 603–619, May 2002.
[40] R. Mariescu-Istodor and P. Fränti, "CellNet: Inferring road networks from GPS trajectories," *ACM Trans. Spatial Algorithms Syst.*, vol. 4, no. 3, pp. 1–22, Sep. 2018.

ZHONGLIANG FU received the B.S., M.S., and Ph.D. degrees in photogrammetry and remote sensing from the Wuhan Technical University of Survey and Mapping, Wuhan, China, in 1985, 1988, and 1996, respectively. He is currently a Professor and a Ph.D. Advisor with the School of Remote Sensing and Information Engineering, Wuhan University. He is also the Director of the Geographic Information System Department. His interests include spatial data management and update, remote sensing image processing and analysis, map scanning image recognition, vehicle license plate recognition, and geographic information engineering technology.

LIANG FAN received the B.S. and M.S. degrees in remote sensing science and technology from Wuhan University, Wuhan, China, in 2013 and 2015, respectively, where he is currently pursuing the Ph.D. degree in photogrammetry and remote sensing with the School of Remote Sensing and Information Engineering. His research interests include spatial data management and update, multisource spatial data matching and fusion, and spatial data analysis.



YANGJIE SUN received the M.S. degree in cartography and geography information system from Central South University, China, in 2016. He is currently pursuing the Ph.D. degree in photogrammetry and remote sensing with the School of Remote Sensing and Information Engineering, Wuhan University. His research interest includes remote sensing 3D information processing and analysis.



ZONGSHUN TIAN received the B.S. degree in spatial-informatics and digitalized technology and the Ph.D. degree in cartography and geographical information engineering from Wuhan University, China, in 2011 and 2017, respectively. He is currently a Lecturer with the School of Management Science and Real Estate, Chongqing University. His research interests include trajectory data mining and location-based service.