Efficient Lane-Level Map Building via Vehicle-Based Crowdsourcing

Jiangang Shu, *Member, IEEE*, Songlei Wang, Xiaohua Jia, *Fellow, IEEE*, Weizhe Zhang, *Senior Member, IEEE*, Ruitao Xie, and Hejiao Huang, *Member, IEEE*

***Abstract*— By providing rich context of lane information on roads, lane-level maps play a vital role in intelligent transportation systems. Since Global Positioning Systems (GPS) have been widely applied to vehicles, vehicle-based crowdsourcing offers an economical way to the lane-level map building by collecting and analyzing the GPS trajectories of vehicles. However, existing works cannot directly extract lane-level road information from raw and interleaved crowdsourcing trajectories, and moreover they are time-consuming and inaccurate. In this article, we propose a lane-level map building scheme, which can directly extract lane-level road information from raw crowdsourcing GPS trajectories with both efficiency and accuracy improvement. Consider the global similarity between trajectories, we design an efficient trajectory segmentation and clustering algorithm based on improved discrete Fréchet distance and entropy theory, which can directly and accurately deal with the interleaved and messy trajectories. To improve the efficiency, we employ the Least Square Estimate (LSE) to constrain Gaussian Mixture Model (GMM) and design an efficient and accurate lane-level road information extraction algorithm. Comprehensive comparative experiments and performance evaluation on a real-world trajectory dataset show that the proposed scheme outperforms the state-of-the-art works in terms of both efficiency and accuracy.**

***Index Terms*— Crowdsourcing, vehicle, lane-level map, trajectory.**

Manuscript received January 22, 2020; revised July 9, 2020 and October 3, 2020; accepted November 23, 2020. Date of publication December 10, 2020; date of current version May 3, 2022. This work was supported in part by the Key-Area Research and Development Program of Guangdong Province under Grant 2019B010136001, in part by the Natural Science Foundation of China under Grant 61732022 and Grant 61672195, in part by the Research Grants Council of Hong Kong under Grant CityU 11208917 and Grant CityU C1008-16G, and in part by the Peng Cheng Laboratory Project of Guangdong Province under Grant PCL2018KP004 and Grant PCL2018KP005. The Associate Editor for this article was M. Zhou. *(Jiangang Shu and Songlei Wang contributed equally to this work.) (Corresponding author: Xiaohua Jia.)*

Jiangang Shu is with the Cyberspace Security Research Center, Peng Cheng Laboratory, Shenzhen 518000, China (e-mail: shujg@pcl.ac.cn).

Songlei Wang and Hejiao Huang are with the School of Computer Science and Technology, Harbin Institute of Technology, Shenzhen 518000, China (e-mail: 18S151524@stu.hit.edu.cn; huanghejiao@hit.edu.cn).

Xiaohua Jia is with the Department of Computer Science, City University of Hong Kong, Hong Kong (e-mail: csjia@cityu.edu.hk).

Weizhe Zhang is with the School of Computer Science and Technology, Harbin Institute of Technology, Harbin 150001, China, and also with the Cyberspace Security Research Center, Peng Cheng Laboratory, Shenzhen 518000, China (e-mail: wzzhang@hit.edu.cn).

Ruitao Xie is with the College of Computer Science and Software Engineering, Shenzhen University, Shenzhen 518000, China (e-mail: drtxie@gmail.com).

Digital Object Identifier 10.1109/TITS.2020.3040728

I. Introduction

LANE-LEVEL maps have been widely applied in intelligent transportation systems such as autonomous driving and advanced driver-assistance systems [1]. Compared with traditional road-level maps with limited and coarse-grained information, lane-level maps are enhanced with detailed lane-level road information including the number, exact locations and geometry lines of lanes on a road. Such fine-grained lane-level maps can offer drivers more accurate road information to avoid fast lane changes and lane departure, which can enhance driving safety, especially in low visibility conditions.

To make high-definition road maps, many digital map makers (e.g., Apple and Baidu) usually rely on smart cars equipped with position sensors (e.g., Global Navigation Satellite Systems and Inertial Navigation Systems) and perception sensors (e.g., laser scanners and cameras) to travel in the target areas and capture the contexts of roads [2]. Although with high accuracy, this approach is costly and time-consuming, and cannot update the maps in real-time when the road conditions change. Since GPS have been widely adopted in vehicles, vehicle-based crowdsourcing is an efficient and economical alternative to extract the lane-level road information from GPS trajectories of vehicles. This approach has been successfully adopted in the road-level map building such as Waze[1] and OpenStreetMap.[2] However, most of the conventional vehicles can only generate the low-precision GPS data with the accuracy of 10m-15m, and it is challenging to build the lane-level maps over the raw low-precision GPS trajectories.

Considering the wide distribution of vehicles' trajectories, probability and statistics is a popular way to build the lane-level maps over the low-precision GPS trajectories [3], [4]. However, existing related solutions need many iterations to fit probabilistic models and are inapplicable to real-world situations where tens of thousands roads and trajectories need to be processed. Moreover, they cannot deal with the sparse or intertwined GPS trajectories, which yet are very common in practice. For example, when passing through the urban canyons or underground passages, due to the low-end GPS equipments, the vehicles may lose their locations or collect the deviated GPS points. In this case, to maintain the high accuracy, high-quality differential GPS (DGPS) data

[1]Waze takes live traffic information from users on the roads to give real-time reports on road conditions. https://www.waze.com/.

[2]OpenStreetMap is created by worldwide voluntary mobile users using their local knowledge and GPS trajectories. https://www.openstreetmap.org/.

is required to extract prior knowledge beforehand [5]. But, it is impossible to obtain such high-quality DGPS data from conventional vehicles. Therefore, it is crucial to extract the lane-level road information with high efficiency and accuracy from the intertwined low-precision GPS trajectories in the vehicle-based crowdsourcing.

In this article, we propose an efficient lane-level map building scheme via vehicle-based crowdsourcing, called *eLane*, which can accurately extract the level-level road information over the low-precision GPS trajectories. The *eLane* consists of two phases: *eLane-PRE* for trajectory preprocessing including segmentation and clustering, and *eLane-EXT* for lane-level road information extraction. In *eLane-PRE*, *eLane* first splits the raw GPS trajectories into segments based on the shape of the real-world roads and the angle variations of trajectories, ensuring that each trajectory segment is only on a single road. Then, it measures the global similarities of trajectory segments with the improved discrete Fréchet distance and classifies the trajectory segments in proximity into a trajectory cluster. In *eLane-EXT*, *eLane* first performs a polynomial curve fitting on the trajectory clusters by LSE [6] to obtain the road features such as centerline and boundaries. These features are then used to optimize the constrained parameters of GMM [7], which reduces the number of iterations of Expectation Maximization (EM) [8]. Finally, combining the road features with the extracted number of lanes by the Constrain Gaussian Mixture Model (CGMM), *eLane* can build an accurate lane-level map. The main contributions of this article can be summarized as follows:

- We propose an efficient trajectory segmentation and clustering algorithm based on global similarity and entropy theory to deal with the interleaved and messy trajectories.
- We explore the constrained Gaussian mixture model to design an efficient and accurate lane-level road information extraction algorithm, which can greatly improve the efficiency while achieving the high accuracy.
- We implement and evaluate our *eLane* on a real-world trajectory dataset comprehensively. Detailed experimental results show that our *eLane* outperforms the state-of-the-art works in terms of both efficiency and accuracy.

The rest of the paper is organized as follows. Related works on trajectory clustering and lane-level map extraction are reviewed in Section II. System model and necessary preliminaries are introduced in Section III. In Section IV, we first give the overview of *eLane*, and then separately describe the detailed constructions of *eLane-PRE* and *eLane-EXT*. The performance evaluation on the real-world dataset is shown in Section V. Finally, we conclude the paper in Section VI.

II. Related Work

Lane-level information extraction over crowdsourcing-based GPS trajectories is essential to build high-definition maps [3]–[5], [34], [35]. Next, we separately discuss existing realted works on mobile crowdsourcing, trajectory clustering and lane-level map extraction.

A. Mobile Crowdsourcing

In mobile crowdsourcing applications, a crowdsourcing platform usually outsources location-based sensing tasks to moving workers equipped with mobile devices, such as smartphones, wearable devices and vehicles. Upon accepting tasks, workers move to specific locations and complete the tasks according to requirements of tasks [42]. To balance worker utilities and platform profit, Sarker *et al*. [43] proposed to formulate task allocation as a multi-objective nonlinear programming problem. To efficiently allocate tasks in dynamic scenarios, where workers' spatiotemporal information is dynamic and cannot be known in advance, Tong *et al*. [44] proposed a global online task allocation mechanism. To protect the privacy during task allocation, Shu *et al*. [45] proposed privacy-preserving task matching scheme while supporting efficient worker revocation.

B. Trajectory Clustering

Due to the unpredictability of users' driving patterns, the GPS trajectories obtained via vehicle-based crowdsourcing are usually complex and diverse [9]. In order to guarantee the extraction accuracy, it is necessary to preprocess these messy trajectories beforehand, i.e., segmentation and clustering.

There are several approaches to trajectory clustering, including distance-based, identifier-based and map-based solutions. In the distance-based solutions, Euclidean distance is a popular metric to measure the similarity of trajectories [10]–[12] and yet it can only evaluate the partial similarity. To capture the global similarity, other metrics, e.g., perpendicular distance, parallel distance or angle distance, have also been adopted [13]–[15]. However, these methods incur a huge computation overhead, because all of three metrics need to be calculated for each pair of trajectories. In the identifier-based methods [16], [17], each participant needs to upload its trajectory, associated with a road identifier indicating to which road the trajectory belongs. This needs all the participants to collaboratively negotiate the road identifiers beforehand and automatically separate the trajectories by themselves. It incurs extra additional communication and computation overhead on the system and the participants. Map-based solutions have also been extensively studied [18]–[20], which realize trajectory clustering by matching a benchmark map with intermediate representations of trajectories. Due to the complexity of road maps, the high clustering accuracy cannot be well achieved.

The comparison of existing trajectory clustering solutions is summarized in Table I. Due to the limitations described above, they can not be directly applied to deal with messy trajectories in the vehicle-based crowdsourcing.

C. Lane-Level Map Extraction

Existing lane-level map extraction solutions can be generally classified into three categories, according to the kind of data that they deal with: LiDAR data [21]–[24], [46], [47], vision data [25]–[32] and GPS trajectory data [3], [4], [33]–[35].

In the LiDAR-based solutions, LiDAR data is usually generated by 2D or 3D LiDAR scanners. Gwon *et al*. [21]

TABLE I
COMPARISON OF EXISTING TRAJECTORY CLUSTERING SOLUTIONS

Solutions	Limitations
Distance-based [10]–[15]	Partial similarity or huge computation overhead
Identifier-based [16], [17]	Uniform road identifier
Map-based [18]–[20]	Low accuracy

proposed to extract lane-level information from the piecewise parametric polynomial set of 3D LiDAR data. LiDAR data can be also directly segmented into lane-level maps by using deep neural networks [22]. Lang *et al.* [24] proposed a EM-based algorithm to detect parameters of parallel and 3D lines from the 3D point cloud [23], and these line parameters can be integrated to build the lane-level map. However, these LiDAR-based solutions cannot be applied in a large scale, because it is impossible to enable each conventional vehicle to be equipped with a professional LiDAR scanner.

Vision-based solutions can be roughly classified into two categories [25]: feature-based schemes [26], [27] and Deep Learning (DL)-based schemes [28]–[32]. In [26], lane points can be detected by aligning scanning lines along the time axis and the Hough transform. Yoo *et al.* [27] proposed to detect lanes over vanishing points from projected 2D images. Modular Bayesian network [28] and fully convolutional network [29], [30] were also introduced to infer ego-lane positions from monocular camera images. Seo *et al.* [31] combined Markov random fields with GMM to produce lane-level highway maps from ortho-images. To extract lane boundaries from satellite imagery, pixel-wise segmentation and hypothesis linking were respectively applied to generate and connect line candidates [32]. However, vision-based solutions are sensitive to image quality, which yet cannot be well guaranteed by various vehicles. Moreover, they are mostly supervised learning and require large-scale training data which needs to be manually annotated for different scenarios. Besides, they generally have high requirements on computation power and memory space.

Since GPS sensors have been universally equipped by vehicles, trajectory-based solutions are the most suitable to be applied in a large scale. Non-parametric Kernel Density Estimation (KDE) was utilized to analyze the probability density distribution of trajectory points [4]. To mine lane-level information from low-precision GPS trajectories, naïve Bayesian classification was introduced in [33]. To improve the extraction accuracy, Zhang *et al.* [34] proposed to calculate the topological representation of road by cubic Hermite spline and stepwise rarefy the topological representation to obtain the accurate locations of lanes. GMM-EM was firstly introduced to fit trajectory points by GMM and estimate GMM parameters by EM [3]. Based on [3], Tang *et al.* [5] designed a new regularization term to confirm the number of lanes and [35] introduced a new constrained condition to simplify the process of EM. However, since they don't set reasonable initial parameters of estimation and appropriately restrict evaluation parameters, their computation costs are huge and a large number of trajectories must be prepared to achieve the claimed accuracy.

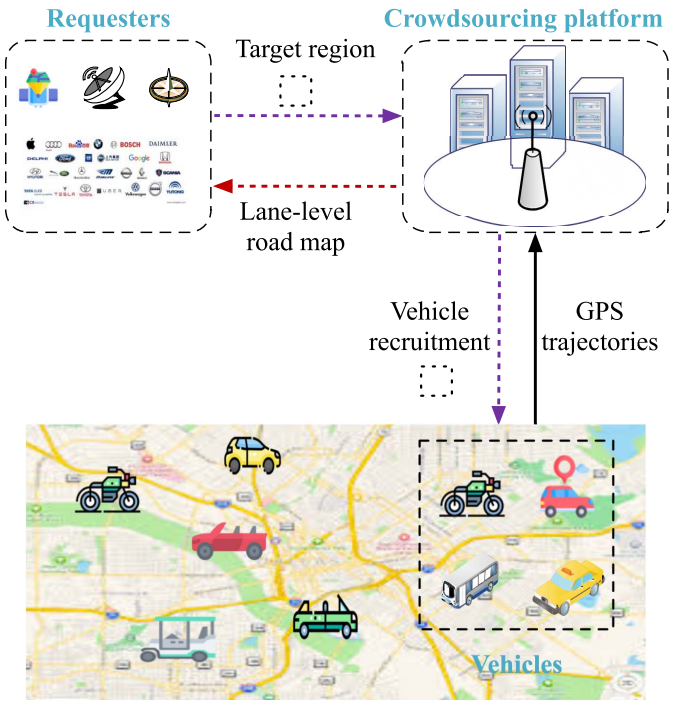


Fig. 1. System model.

III. MODEL AND PRELIMINARIES

A. System Model

As illustrated in Fig. 1, we consider a vehicle-based crowdsourcing system for lane-level map building, which consists of vehicles, a crowdsourcing platform and requesters. Their roles are defined as follows:

- *Requesters*. A requester specifies a geographical region to collect the lane-level road information as a task, and publishes it onto the crowdsourcing platform. The requester could be a map provider or a self-driving company.
- *Crowdsourcing platform*. Upon receiving a task from a requester, the crowdsourcing platform recruits some vehicles and lets the vehicles move in the target region, record and upload their GPS trajectories. After collecting all the vehicles' GPS trajectories, the crowdsourcing platform analyzes the trajectories, builds a lane-level road map and sends the map to the requester.
- *Vehicles*. The vehicles recruited are the workers that collect and upload their GPS trajectories of driving routes to the crowdsourcing platform.

The focus of the paper is to efficiently build the lane-level road map by analyzing the vehicles' raw GPS trajectories. Note that how to publish a task and how to recruit the vehicles are out of scope of the paper.

B. Preliminaries

GPS trajectory. Each trajectory uploaded by a vehicle with identity v can be represented as

$$Tr_v = \{loc_1, loc_2, \cdots, loc_{n_v}\},$$

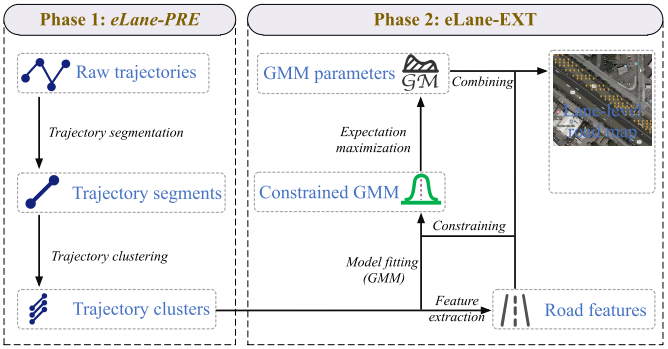


Fig. 2. The framework of *eLane*.

where n_v is the number of collected locations in the trajectory Tr_v. Each location $loc_{i\in[1,n_v]}$ is a GPS sampled spatial-temporal point, denoted as a tuple $< x_i, y_i, t_i >$ where x_i and y_i represent the longitude and latitude of geographic location, respectively, and t_i is the collecting timestamp of location. And note that all the locations in Tr_v are arranged in chronological order, which determines the driving direction of the vehicle.

Discrete Fréchet distance. Fréchet distance is a popular metric to measure the similarity between two curves, and its discrete variation, called discrete Fréchet distance [36], is widely adopted because of its good approximation to the Fréchet distance and computational efficiency. Given two discrete location sequences $P = \{p_1, \cdots, p_i\}, i \in [1, m]$ and $Q = \{q_1, \cdots, q_j\}, j \in [1, n]$, a formal definition of the discrete Fréchet distance $\Psi_{dF}(P, Q)$ is set as follows:

$$\begin{aligned}&\Psi_{dF}(P, Q)\\ &\quad = \max\{d_i | d_i\\ &\quad = \min\{d_i^j | d_i^j = d(p_i, q_j), j \in [1, n]\}, i \in [1, m]\},\end{aligned} \quad (1)$$

where $d(p_i, q_j)$ is the Euclidean distance between p_i and q_j.

IV. The Design of *eLane*

A. Overview

As illustrated in Fig. 2, *eLane* is composed of two phases, *eLane-PRE* for trajectory preprocessing including trajectory segmentation and clustering, and *eLane-EXT* for lane-level road information extraction, which are described as follows:

- *eLane-PRE*. In this phase, each raw GPS trajectory is split into multiple segments based on its angle variation, ensuring that each separated segment is on a relatively straight road. Then, distance-based trajectory clustering is performed on the segments of different trajectories, to ensure that the segments of different trajectories on the same road are put into a trajectory cluster.
- *eLane-EXT*. In this phase, road features such as centerline and boundaries are first determined by LSE. Then these road features can be used as the constraints of GMM and the initialization parameters of EM. Finally, the lane-level road information can be extracted with the EM algorithm on the constrained GMM.

The notations used in our *eLane* are summarized in Table II.

TABLE II
Notations

Notation	Description
T_{angle}	Threshold for trajectory segmentation
T_{dis}	Threshold for clustering distance
k	Number of Gaussian components, one for each lane
$\{\omega_{j\in[1,k]}\}$	Weight of each component, corresponding to the relative traffic volume in each lane, and $\sum_{j=1}^{k} \omega_j = 1$
$\{\mu_{j\in[1,k]}\}$	Mean of the trajectories for each component, equal to the centerline of each lane
$\{\sigma_{j\in[1,k]}\}$	Standard deviation of the trajectories for each component, representing the spread of the trajectories of each lane

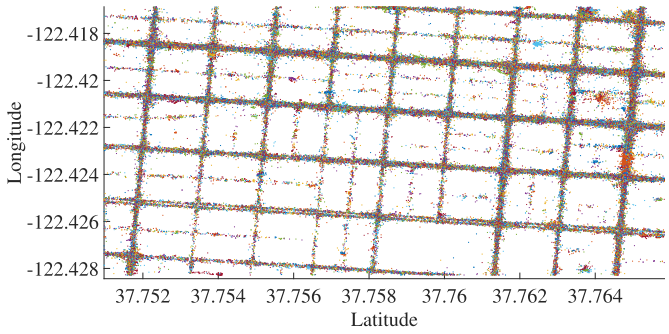


Fig. 3. Raw vehicle trajectories.

B. eLane-PRE: Trajectory Segmentation and Clustering

Since the starting and ending points of vehicles are different and meanwhile a trajectory usually covers several road sections, the collected trajectories from different vehicles may intertwine with each other, as shown in Fig. 3. To extract the accurate lane-level road information, we need to preprocess the raw trajectories beforehand, mainly including trajectory segmentation and trajectory clustering. *eLane-PRE* first divides the raw trajectories into multiple trajectory segments based on the angular variations of trajectory vectors. Then the trajectory segments are classified into different clusters based on the improved discrete Fréchet distance.

1) Trajectory Segmentation: A spatial-temporal trajectory usually reflects the moving tendency of a vehicle. Specifically, any two adjacent locations loc_i and loc_{i+1} can form a trajectory vector $\boldsymbol{v}_i$, which reflects the moving direction at loc_i. The angular variation ξ_i of any two adjacent trajectory vectors $\boldsymbol{v}_{i-1}$ and $\boldsymbol{v}_i$ reflects the change of driving direction at loc_i. ξ_i at loc_i can be calculated with its adjacent two locations loc_{i-1} and loc_{i+1}:

$$\begin{cases}\boldsymbol{v}_{i-1} &= (x_i - x_{i-1}, y_i - y_{i-1})\\ \boldsymbol{v}_i &= (x_{i+1} - x_i, y_{i+1} - y_i)\\ \xi_i &= \arccos \dfrac{\boldsymbol{v}_{i-1} \cdot \boldsymbol{v}_i}{|\boldsymbol{v}_{i-1}||\boldsymbol{v}_i|}.\end{cases} \quad (2)$$

It is known that the angle of two adjacent trajectory vectors in a trajectory will not change a lot unless they are at an

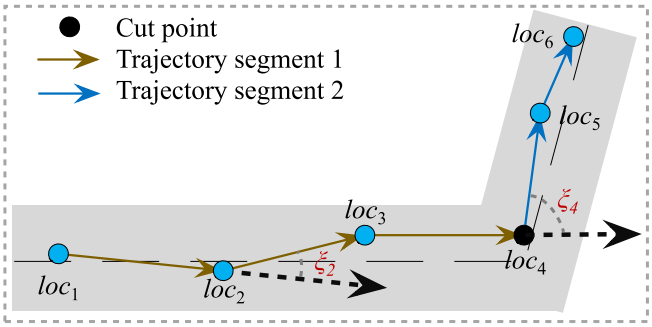


Fig. 4. Trajectory segmentation.

intersection or changing lanes [37]. For example in Fig. 4, in a trajectory piece $Tr_v = \{loc_1, \cdots, loc_6\}$, the angle ξ_2 at loc_2 changes a little, while the angle ξ_4 at the inflection point loc_4 changes obviously. Based on these observations, we can divide Tr_v into two segments $Ts_v^1 = \{loc_1, loc_2, loc_3, loc_4\}$ and $Ts_v^2 = \{loc_4, loc_5, loc_6\}$ by comparing each angle ξ_i with a preset angle threshold T_{angle}. The value of T_{angle} is determined based on specific intersection design rules. For instance in China, the minimum turning angle at an intersection is about 60° [38], so T_{angle} can be set to 60°. The details of trajectory segmentation are described as Algorithm 1. Through the trajectory segmentation, trajectories can be divided into multiple trajectory segments to ensure that all locations of a trajectory segment are on the same road section.

Algorithm 1 Trajectory Segmentation

Input:
Trajectories $TR =< Tr_1, Tr_2, \cdots, Tr_V >$
Angle threshold T_{angle}
Output:
Trajectory segments $TS =< Ts_1, \cdots, Ts_L >$

```
Initialize TS = Φ
for all Tr_{v∈[1,V]} in TR do
  i = 1
  j = 2
  for all loc_{j∈[2,n_v−1]} in Tr_v do
    Compute ξ_j as formula (2)
    if ξ_j > T_angle then
      Add < loc_i, ··· , loc_j > as a segment into TS
      i = j
    end if
    j = j + 1
  end for
end for
```

2) Trajectory Clustering: Although each trajectory is transformed into a sequence of separated segments after trajectory segmentation, the trajectory segments on different roads may still be intertwined, which hinders the accurate extraction of lane-level road information. In this regard, segment clustering still needs to be conducted beforehand.

Discrete Fréchet distance is a popular metric to measure the spatial relationships of discrete points between two curves, and it can be adopted to realize the trajectory clustering by measuring the distance between any two trajectory segments. For example, there are three trajectory segments $< Tr_1, Tr_2, Tr_3 >$ in Fig. 5(a), among which Tr_1 and Tr_2 are on the same road, and Tr_3 is on the other road. The discrete Fréchet distance between Tr_1 and Tr_2 is $\Psi_{dF}(Tr_1, Tr_2) = d_1$, which is obviously less than the road width. The discrete Fréchet distance between Tr_1 and Tr_3 is $\Psi_{dF}(Tr_1, Tr_3) = d_2$, which is significantly more than the road width. Therefore, Tr_1 and Tr_2 can be classified into the same cluster of trajectory segment, while Tr_3 will be classified into another cluster. However, a trajectory on the same road may be divided into different trajectory segments due to the noise points inside, or the starting and ending points of trajectory segments on the same road may be far apart due to the different length of trajectory segments. These cases will make the designed trajectory clustering ineffective. As shown in Fig. 5(b), $\Psi_{dF}(Tr_4, Tr_5)$ is actually the distance d_3 between the ending points of two trajectory segments, which is obviously more than the road width. This will classify them into different clusters by mistake. Due to the cluttered characteristic of crowdsourcing trajectory data, this case occurs frequently and will greatly affect the performance of clustering. Therefore, we need to improve the discrete Fréchet distance to make it more suitable for trajectory clustering in crowdsourcing.

Improved discrete Fréchet distance. To solve the above drawback, we improve discrete Fréchet distance by padding the starting and ending points of trajectory segments before distance computation. As shown in Fig. 5(c), *eLane-PRE* compares the positional relationship of sequences $< loc_1, loc_4 >$ and $< loc_3, loc_7 >$, where (loc_1, loc_3) and (loc_4, loc_7) are the starting points and ending points of Tr_4 and Tr_5, respectively. Since the position of loc_1 is before loc_4, *eLane-PRE* adds loc_1 as a new starting point of Tr_5. Likewise, *eLane-PRE* adds loc_7 as a new ending point of Tr_4. Through padding, Tr_4 and Tr_5 can be updated as:

$$\begin{cases} \widetilde{Tr_4} = \{loc_1, loc_2, loc_3, loc_7\} \\ \widetilde{Tr_5} = \{loc_1, loc_4, loc_5, loc_6, loc_7\}. \end{cases}$$

Then the discrete Fréchet distance between $\widetilde{Tr_4}$ and $\widetilde{Tr_5}$ is calculated as d_4, which is less than the road width and thus Tr_4 and Tr_5 are classified into the same cluster. At this point, given two trajectory segments $P = \{p_1, \cdots, p_{n_1}\}$ and $Q = \{q_1, \cdots, q_{n_2}\}$, the improved discrete Fréchet distance is calculated as follows:

$$\begin{cases} \widetilde{P} = \{o_{min}(q_1, p_1), p_1, \cdots, p_{n_1}, o_{max}(q_{n_1}, p_{n_2})\} \\ \widetilde{Q} = \{o_{min}(p_1, q_1), q_1, \cdots, q_{n_2}, o_{max}(p_{n_1}, q_{n_2})\} \\ \widetilde{\Psi_{dF}}(P, Q) = \Psi_{dF}(\widetilde{P}, \widetilde{Q}), \end{cases} \tag{3}$$

where $o_{min}(p_i, q_j) = p_i$ if $p_i.x < q_j.x$, otherwise $o_{min}(p_i, q_j) = \phi$, and $o_{max}(p_i, q_j) = p_i$ if $p_i.x > q_j.x$, otherwise $o_{max}(a, b) = \phi$. The details of trajectory clustering are described as Algorithm 2.

Heuristic search for T_{dis}. Since the value of threshold T_{dis} is crucial to the performance of trajectory clustering, we design a heuristic search based on entropy theory [39] to determine its optimum value. The entropy is usually used to measure the

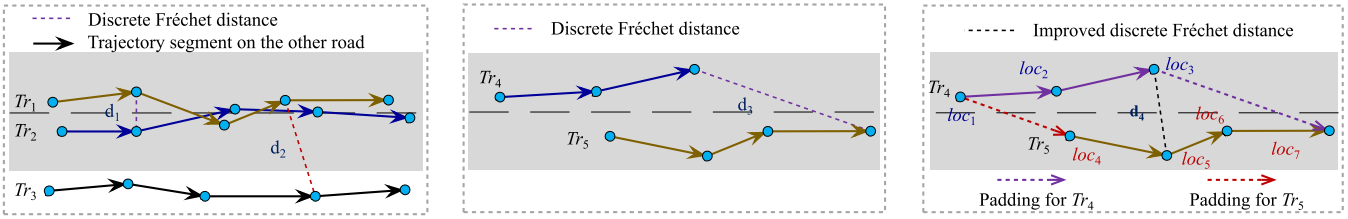


Fig. 5. Trajectory clustering. (a) discrete Fréchet distance; (b) discrete Fréchet distance failure; (c) improved discrete Fréchet distance.

Algorithm 2 Trajectory Clustering

Input:
Trajectory segments $TS =< Ts_1, \cdots, Ts_L >$
Distance threshold T_{dis}
Output:
Trajectory clusters $TC =< Tc_1, \cdots, Tc_m >$
Initialize $label(i) = 0, i \in [1, L]$, and $m = 0$
while $\exists\ label(i) = 0,\ i \in [1, L]$, **do**
  $m = m + 1$
  $Tc[m] = \Phi$
  for all $label(i) = 0,\ i \in [1, L]$ **do**
    if $\nexists\ Ts_j \in Tc_m$ s.t. $\Psi_{dF}(Ts_i, Ts_j) > T_{dis}$ **then**
      $label(i) = m$
      Add Ts_i to Tc_m
    end if
  end for
end while

uncertainty about a given probability distribution, and it can be defined as:

$$H(X) = \sum_{i=1}^{n} p(x_i) * \log_2 \frac{1}{p(x_i)}. \tag{4}$$

In a bad clustering case, the number of trajectories in different trajectory clusters, $|Tc_i|$, tends to be uniform. That is, $|Tc_i| = 1$ for almost all trajectory clusters when T_{dis} is too small, and the number of trajectory clusters $m \approx 1$ when T_{dis} is too large. In this case, the entropy $H(X)$ will be large. In contrast, in a good clustering case, $|Tc_i|$ tends to be skewed and $H(X)$ becomes smaller. Based on the above observation, we can obtain an optimal T_{dis} by minimizing the objective function H_T:

$$H_T = \frac{H(X)}{\sqrt{m}} = \frac{\sum_{i=1}^{n} p(x_i) * \log_2 \frac{1}{p(x_i)}}{\sqrt{m}}, \tag{5}$$

where $p(x_i) = \frac{|Tc_i|}{L}$, L is the total number of trajectory segments and m is the total number of clusters. The optimal T_{dis} can be efficiently obtained by simulated annealing [40], [48].

C. eLane-EXT: Lane-Level Road Information Extraction

In this phase, *eLane-EXT* first utilizes LSE to extract road features (e.g., centerline and boundaries) from the trajectory segment clusters. Then, these features can be used as the constraints of GMM and the initialization parameters of EM. Finally, combining the features extracted by LSE with the concrete parameters extracted by EM, *eLane-EXT* determines the exact positions of each lane.

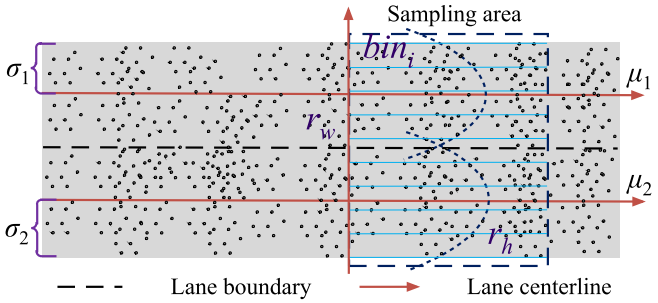


Fig. 6. Trajectory fitting by GMM.

Extraction by GMM-EM. To fit the GPS trajectories across the roads, GMM is widely adopted based on the common assumption that the trajectories tend to cluster near the center of each lane with some spread due to inaccuracy of GPS [3]. In the GMM fitting, two-dimensional GPS data is converted into single-dimensional data by using an $[r_h, r_w]$ rectangular window to sample the vertical sections of trajectories, as shown in Fig. 6. Through the rectangular-based sampling, the road is divided into multiple bins, where each bin is denoted by bin_i. The probability density of bin_i can be calculated by

$$p(bin_i) = \frac{\#(bin_i)}{N}, \tag{6}$$

where $\#(bin_i)$ is the number of points in the bin bin_i, and N is the number of points in the sampling area. At this point, the GMM can be defined as follows:

$$p(bin_i) = \sum_{j=1}^{k} \omega_j \frac{\exp\left\{-\frac{(bin_i - \mu_j)^2}{2\sigma_j^2}\right\}}{\sqrt{2\pi\sigma_j^2}}. \tag{7}$$

To determine the GMM, the EM algorithm is commonly adopted to figure out the unknown parameter $\theta_k(\omega_j, \mu_j, \sigma_j)$ [3]. The basic idea is to estimate $\theta_k\{\omega_j, \mu_j, \sigma_j\}^k$ for each $j \in [1, k]$, and then select the k by solving

$$k = \min\left\{-\frac{1}{Bn}\sum_{i=1}^{Bn} \log p(bin_i \mid \theta_k) + \lambda R(\mathcal{D}, \theta_k)\right\}, \tag{8}$$

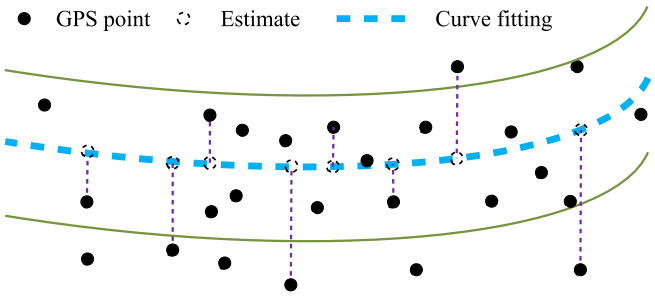


Fig. 7. Curve fitting the road centerline.

where Bn is the number of bins. The first term in equation (8) is the negative mean log-likelihood, which assesses how well the model fits the observed data. In the second term, $R(\mathcal{D}, \theta_k)$ is a regularization term that penalizes complex models, and $\lambda > 0$ is the regularization parameter [5].

Disadvantages. Since the generic GMM-EM method in existing solutions [3], [5] don't put any constraints on the parameters of θ_k by considering actual road conditions, parameter estimation and optimization takes too many number of iterations. This unconstrained GMM-EM approach causes huge computational overhead, especially when estimating numerous roads. Moreover, the initialization parameters $(\omega_j^{(0)}, \mu_j^{(0)}, \sigma_j^{(0)})$ are regarded as fixed values, which does not conform to the diverse road conditions in the real world and will also reduce the extraction accuracy.

Extraction by constrained GMM-EM. In order to improve efficiency, we design a constrained GMM-EM in *eLane-EXT*. *eLane-EXT* contains two steps: road feature extraction and lane-level road information extraction.

(1) Road feature extraction. In this step, the road centerline is first extracted by fitting the trajectory clusters with LSE, and then the road boundaries are obtained by the prediction interval under appropriate confidence coefficient. As shown in Fig. 7, for each location point $loc_i =< x_i, y_i, t_i >$, $i \in [1, n]$ where n is the number of points in a trajectory segment cluster, LSE provides an estimate $\widehat{y_i}$ to minimize the following function [6]:

$$\min \sum_{i=1}^{n} (\widehat{y_i} - y_i)^2. \tag{9}$$

Through LSE, the estimated curve $\widehat{center_i} =< x_i, \widehat{y_i} >$ is closest to the real centerline of road.

On the other hand, the two boundaries $\overline{B_i}$ and $\underline{B_i}$ of loc_i in a trajectory segment cluster can be extracted by prediction interval as

$$\overline{B_i} = \widehat{y_i} + t_{n-2}(s)\sqrt{1 + \frac{1}{n} + \frac{(x_i - \bar{x})^2}{\sum_{j=1}^{n}(x_j - \bar{x})^2}}, \tag{10}$$

$$\underline{B_i} = \widehat{y_i} - t_{n-2}(s)\sqrt{1 + \frac{1}{n} + \frac{(x_i - \bar{x})^2}{\sum_{j=1}^{n}(x_j - \bar{x})^2}}, \tag{11}$$

where $\bar{x}$ is the mean, t_{n-2} is the multiplier associated with sampling area size and confidence level, s is the standard error of the estimate. Through the detailed experiments, we find that when the confidence coefficient is set as 95%, the prediction

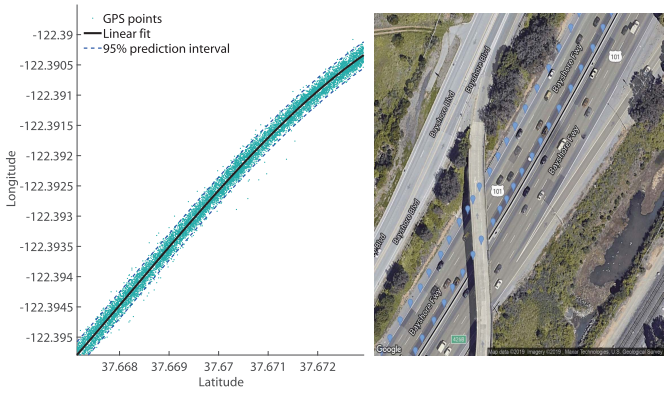


Fig. 8. Prediction interval and marking of road boundaries on Google map.

interval of the trajectory segment cluster can accurately identify the boundaries of roads. As shown in Fig. 8, the extracted road boundaries are marked on the Google satellite map.

(2) Lane-level road information extraction. Some prior knowledge about lane shapes in reality can be used to constrain the GMM:

- The width of each lane in the same sampling area is nearly the same in reality, then $\{\mu_j\}$ can be assumed as equally spaced. Based on this assumption, each μ_j can be constrained as

$$\mu_j = B + (j - \frac{1}{2})\Delta\mu_k, j \in [1, k], \tag{12}$$

where B is the starting boundary of the sampling area and it can be calculated by

$$B = Bin(\frac{1}{N}\sum_{i=1}^{N}\underline{B_i}), \tag{13}$$

where the function $Bin()$ determines which bin the boundary is located in, and N denotes the number of points in the sampling area. The average of lower bounds is set as the location of the starting boundary of the sampling area. $\Delta\mu_k$ is the distance between the centerline of adjacent lanes, and it equals to the lane width. When k is known, $\Delta\mu_k$ can be calculated by

$$\Delta\mu_k = \frac{1}{k}[Bin(\frac{1}{N}\sum_{i=1}^{N}\overline{B_i}) - Bin(\frac{1}{N}\sum_{i=1}^{N}\underline{B_i})]. \tag{14}$$

- The lanes of a road are usually of equal width, and the spread of trajectories for each lane remains approximately the same, then all the Gaussian components can be assumed to share the same variance:

$$\sigma_1 = \cdots = \sigma_j = \cdots = \sigma_k = \sigma, j \in [1, k]. \tag{15}$$

Based on the above two observations, we revise the GMM and propose the Constrained GMM (CGMM) as

$$p(bin_i) = \sum_{j=1}^{k}\omega_j \frac{\exp\left\{-\dfrac{[bin_i - B - (j - \frac{1}{2})\Delta\mu_k]^2}{2\sigma^2}\right\}}{\sqrt{2\pi\sigma^2}}. \tag{16}$$

TABLE III
METHODOLOGIES OF STATE-OF-THE-ART WORKS

Scheme	Methodology
[3] 2010 [5] 2016	Fit data by GMM; estimate params by EM; select result with regularization term
[4] 2013	Identify locations of lanes by KDE
[34] 2016	Estimate approximate lane curves by cubic Hermite spline; rarefy lane parameters by Hausdorff distance
[35] 2020	Identify the centerline of road by dynamic time warping; identify locations of lanes by GMM
eLane	Identify the centerline and boundaries of road by LSE; constrain the parameters of GMM-EM

At this point, the model parameters to be estimated in EM are reduced as $\{k, \omega_1, \cdots, \omega_k, \sigma\}$, which will largely reduce the number of iterations.

On the other hand, through the experiments in Section V-C, the regularization term in formula (8) can be ignored. That is, k can be determined by solving

$$k = \max\left\{\frac{1}{Bn}\sum_{i=1}^{Bn}\log p(bin_i \mid \theta_k)\right\}. \tag{17}$$

In most cases, the above method can achieve the correct result. However, due to the uncertain traffic flows in each lane or the GPS inaccuracy, incorrect estimations on the number of lanes still exist occasionally. Based on this, we optimize the results of the estimated number of lanes by comparing $< Nline_{\mathcal{A}_{i-1}}, Nline_{\mathcal{A}_i}, Nline_{\mathcal{A}_{i+1}} >$, where $Nline_{\mathcal{A}_i}$ is the estimated number of lanes in the sampling area $\mathcal{A}_i$. If there exists the following case:

$$\begin{cases} Nline_{\mathcal{A}_{i-1}} & = Nline_{\mathcal{A}_{i+1}} \\ Nline_{\mathcal{A}_i} & \neq Nline_{\mathcal{A}_{i-1}}, \end{cases} \tag{18}$$

we can set $Nline_{\mathcal{A}_i} = Nline_{\mathcal{A}_{i-1}}$. This holds based on the real-world observation that the number of lanes cannot frequently change in a short distance. Such optimization can significantly increase the extraction accuracy of the number of lanes.

V. PERFORMANCE EVALUATION

In this section, we first independently evaluate the performance of two phases in *eLane*: *eLane-PRE* and *eLane-EXT*, and then measure the efficiency and accuracy of *eLane-EXT* in comparison with the state-of-the-art lane-level map building schemes: [3]–[5], [34], [35]. Their methodologies are briefly summarized in Table III.

In the performance evaluation, we adopt a real-world dataset obtained from [41], which contains about ten million GPS coordinates collected by approximately 500 taxis over 30 days in the San Francisco Bay Area, USA. Our *eLane*[3] is implemented by Matlab and all the experiments are conducted on a PC with an Intel Core I7 CPU at 2.6 GHz and 16 GB RAM. In the experiments, T_{angle} is set as 60°, T_{dis} is set as $20m$, the confidence coefficient is set as 95%, the iterative convergence threshold is set as 10^{-4}, and the length r_h and width r_w of sampling area are respectively set as $5m$ and $50m$.

[3]https://github.com/jgshu/eLane

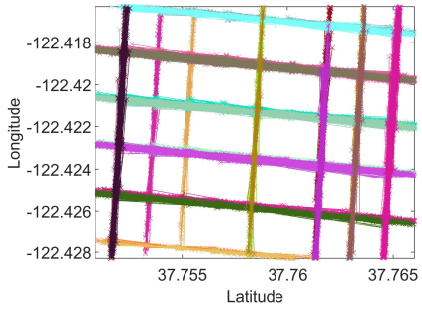


Fig. 9. Trajectory segment clusters when $T_{angle} = 60°$ and $T_{dis} = 20m$, and different colors means different clusters.

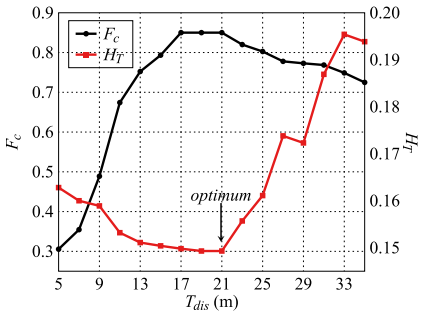


Fig. 10. Objective function H_T as formula (5) and clustering metric F_c as formula (19) under different clustering distance threshold T_{dis}.

A. eLane-PRE

In order to quantitatively evaluate the performance of *eLane-PRE* and the determination of clustering distance threshold T_{dis}, we design a metric F_c inspired by $F1 - score$ as

$$F_c = \frac{2 \times precision_c \times recall_c}{precision_c + recall_c}, \tag{19}$$

where $precision_c = \frac{cv_c}{M_c}$ and $recall_c = \frac{cv_c}{Ground_c}$. M_c is the number of extracted road clusters, $Ground_c$ is the actual number of roads in the sub-region, and cv_c is the number of actual roads matched by extracted trajectory clusters.

To easily process the large dataset, we split it into multiple sub-regions, as shown in Fig. 3. For each sub-region, we link the raw GPS points of each vehicle in chronological order as a trajectory. Then based on the prior knowledge of real-world road conditions, we split the trajectory into multiple trajectory segments based the minimum turning angle at an intersection, i.e., $T_{angle} = 60°$. Finally, we cluster the trajectory segments based on the improved discrete Fréchet distance. Fig. 9 shows the snapshot of trajectory segment clusters in a sub-region, where lines with different colors represent trajectory segment clusters on different roads. Fig. 10 shows the variation of F_c and H_T under different T_{dis}. We can see the trend of H_T is opposite to that of F_c, and the maximum F_c and the minimum H_T can be simultaneously achieved

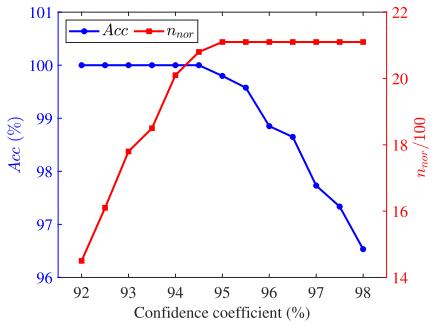


Fig. 11. Accuracy of road feature extraction by LSE under different confidence coefficients.

when T_{dis} is an optimum value. That means, when T_{dis} is less than the optimum, the trajectory segments on the same road will be incorrectly classified into different trajectory clusters, so the number of clusters m is larger than the correct one and H_T is larger than the minimum; when T_{dis} is larger than the optimum, the trajectory segments on different roads will be classified into the same trajectory cluster, so m is smaller than the correct one and H_T is larger than the minimum.

B. eLane-EXT

1) Road Feature Extraction: We randomly select 10 trajectory segment clusters from the dataset processed by *eLane-PRE*, and then divide the trajectory clusters into two parts according to the driving directions. To evaluate the effectiveness of road feature extraction, we define a metric called average accuracy

$$Acc = \frac{n_{nor}}{n_{nor} + n_{out}} \times 100\%, \tag{20}$$

where n_{nor} denotes the average number of points located inside the road area, and n_{out} denotes the average number of points located outside the road area. Finally, we measure Acc and n_{nor} on the selected 10 clusters under different confidence coefficients within $[92\%, 98\%]$, as shown in Fig. 11. When the confidence coefficient is within $[92\%, 95\%]$, the prediction interval is within the road boundaries and all points between the prediction interval are inside the road area, and thus $Acc \approx 100\%$. As the confidence coefficient increases, the road width covered by the prediction interval gradually expands, and N_{nor} also increases. But when the confidence coefficient is greater than 95%, the prediction interval will exceed the road width and outliers will appear. With the increasing confidence coefficient in $[95\%, 98\%]$, Acc decreases while n_{nor} tends to be stable. Therefore, we can accurately identify the road boundaries with the prediction interval under the 95% confidence coefficient.

2) Lane-Level Road Information Extraction: According to the road construction standards in America, we set the maximum number of lanes as 6. The length r_h and width r_w of sampling area are set as $5m$ and $50m$, respectively, and each sampling area is divided into 100 bins. The initialization parameters $\theta_k^{(0)}\{\omega_1^{(0)}, \cdots, \omega_k^{(0)}, \sigma^{(0)}\}$ for each k are set as following:

$$\theta_k^{(0)} \begin{cases} \omega_1^{(0)} = \ldots = \omega_k^{(0)} = \dfrac{1}{k}, \\ \sigma^{(0)} = \dfrac{\Delta\mu}{2}. \end{cases} \tag{21}$$

The criterion for iterative convergence of CGMM is that the absolute value of difference between the mean log-likelihood values of two adjacent iterations is less than 10^{-4}. Fig. 12 shows the training process of CGMM for a 4-lane road under different $k \in [1, 6]$. It can be seen that when k is set as the correct number of lanes, the CGMM generated by EM can fit the dataset best with the largest mean log-likelihood, which validates the effectiveness of *eLane-EXT*.

To measure the accuracy of *eLane-EXT*, we take 10 consecutive samplings on the randomly selected roads under 4 types of lanes: 2-lane, 3-lane, 4-lane and 5-lane, and perform the quantitative evaluations on both the accuracy of lane number identification and the accuracy of lane position extraction. The quantitative evaluation on the accuracy of lane number identification is done by marking the identified results by *eLane-EXT* on the Google satellite map and manually counting the correct number of lanes, then comparing the human-interpreted results with the identified results. Fig. 13 (a) and Fig. 13 (b) respectively show the identified number of lanes before and after the optimization mentioned in section IV-C, and it proves that our proposed optimization method can greatly improve the accuracy of lane number identification. The experiments show that our proposed *eLane-EXT* achieves an overall accuracy of 83.3% in the lane number identification, which is superior to 76.9% by [3] but slightly inferior to 85.2% by [5]. However, the higher accuracy in [5] depends highly upon the additional DGPS data, which is not easy to obtain in crowdsourcing scenarios.

The quantitative evaluation on the accuracy of lane position extraction is usually performed through the comparison with a lane-level map in high-accuracy GPS data (e.g., DGPS). However, due to the lack of such data, we can only perform the quantitative evaluation through the above human understanding method. We design a metric F_p inspired by $F1 - score$ as

$$F_p = \frac{2 \times precision_p \times recall_p}{precision_p + recall_p}, \tag{22}$$

where $recall_p = \dfrac{cv_p}{Ground_p}$ and $precision_p = \dfrac{cv_p}{M_p}$. M_p is the number of extracted lanes, $Ground_p$ is the actual number of lanes, and cv_p is the number of extracted lanes matched by the actual road conditions. Fig. 14 marks the extracted results of 4 types of lanes on the Google satellite map where yellow markers depict the extracted centerline of each lane. According to the experiments, F_p is 0.87 on average for 40 sampling areas under 4 types of lanes, and it indicates that most of extracted lane-level road information by *eLane-EXT* can accurately identify the actual lane positions.

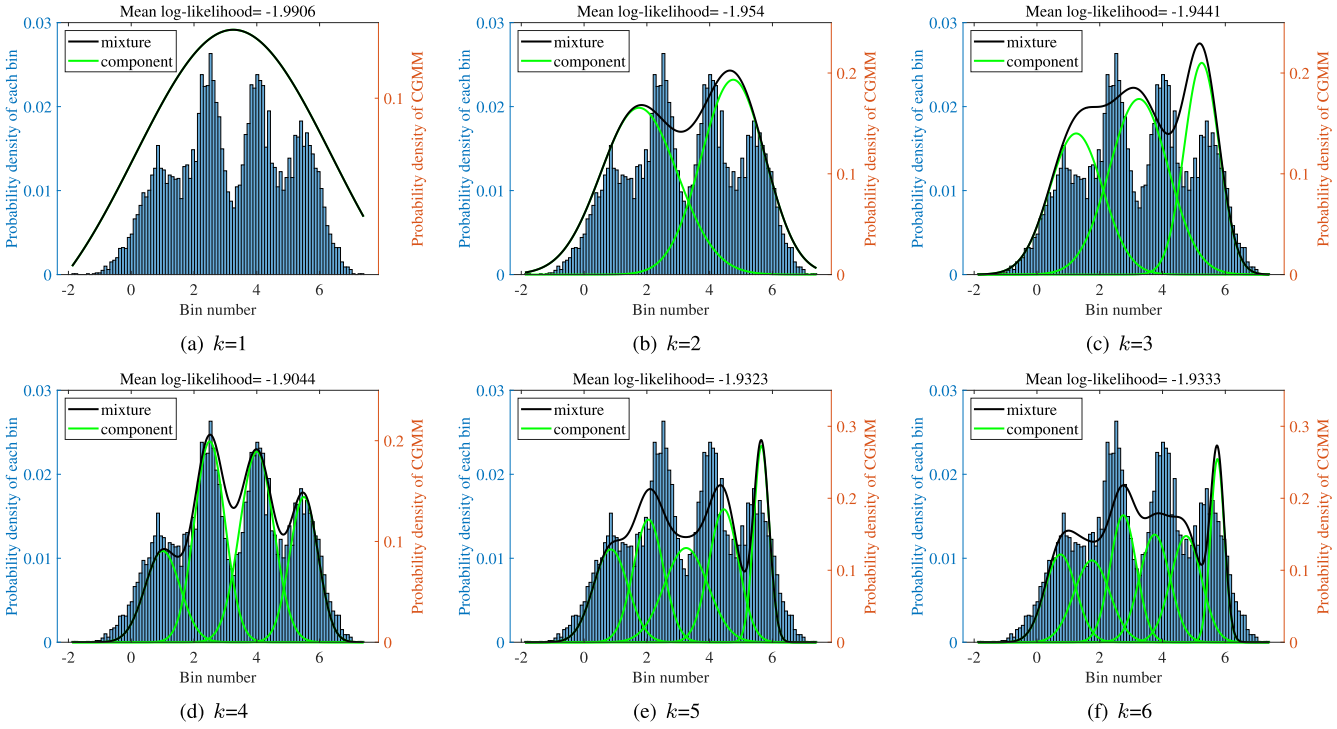


Fig. 12. Training process of CGMM on a 4-lane road when the number of lanes k is estimated in $[1, 6]$.

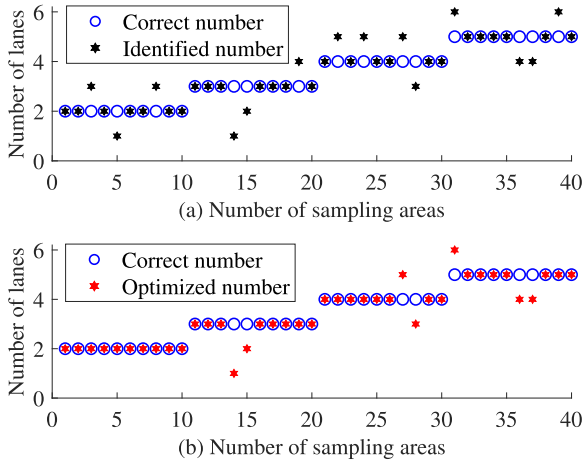


Fig. 13. Lane number identification and optimization. (a) before optimization; (b) after optimization.

C. Comparative Analysis

In this subsection, we will compare our *eLane* with the state-of-the-art lane-level map building schemes in Table III in the following three aspects: 1) the visualization of final extracted results; 2) the time efficiency in terms of the number of EM iterations and run time; 3) the extraction accuracy of different data volumes.

1) Result Visualization: Fig. 15 marks the lane centerlines extracted by all the comparison schemes on the Google satellite map. In Fig. 15 (b) and Fig. 15 (c), we see that the lane centerlines extracted by [3]–[5] are not smooth and consecutive. The reason is that they all separately set rectangular sampling areas and the predicted results of each sampling area is independent. In Fig. 15 (e) for [35], we see that the interval between adjacent lane centerlines are not equal, and it is because that it doesn't take into account the actual road information, i.e., the width of lanes on a road is equal and lane centerlines are equidistant. [34] solves the above defects by utilizing cubic Hermite spline to extract the skeleton of roads, as shown in Fig. 15 (d). However, it doesn't achieve good efficiency and accuracy, as shown in Fig. 17 and Fig. 18. In contrast, our *eLane* achieves higher efficiency and accuracy while ensuring the continuity of lane centerlines, as shown in Fig. 15 (a). The reasons are as follows: (1) the skeleton (i.e., boundaries) of roads are first extracted by LSE and the number of lanes in each sampling area is accurately estimated by CGMM;(2) Wrong estimations are finally adjusted based on the background information: the number of lanes in adjacent sampling areas is ordinarily equal. Therefore, we can obtain the exact number of lanes on a road, and then obtain the accurate lane centerlines by combining the road boundaries and the number of lanes.

2) Efficiency: Fig. 16 shows the mean log-likelihood of *eLane*, [3], [5] and [35], which represents the process of EM algorithm for different types of lane. From the three subgraphs for *eLane*, we see that when k is set as the correct number of lanes, the estimated model parameters will produce the maximum mean log-likelihood compared with other incorrect k. That is because if k is correct, *eLane* can obtain the exact locations $\{\mu_1, \cdots, \mu_k\}$ of lanes by LSE, and the initial estimate of σ^0 of each Gaussian component is also close to a half of the actual lane width. Therefore, *eLane* can directly determine the most appropriate k without the additional regularization term in formula (8). In contrast, from the remaining six subgraphs

(a) 2-lane (b) 3-lane (c) 4-lane (d) 5-lane

Fig. 14. Marking of lane centerline on Google map when the confidence coefficient of road boundaries is set as 95%.

(a) *eLane* (b) [3] and [5] (c) [4] (d) [34] (e) [35]

Fig. 15. Marking of lane centerline extracted by different schemes on the Google map.

for [3], [5] and [35], we can see that there is no significant difference between the obtained mean log-likelihood under different k. For this reason, the correct number of lanes cannot be directly determined, and the additional result selection is required. Moreover, from Fig. 16, we can see that the number of iterations of our *eLane* is much less than that of them. That is because in each iteration of *eLane*, the centerline locations $\{\mu_1, \cdots, \mu_k\}$ are fixed. Thereby the iteration can converge quickly when the parameter k does not match the trajectory cluster. We conduct experiments in different sampling areas and find that the generic GMM method in [3] and [5] requires an average of 228 iterations to obtain the estimated results and [35] requires an average of 179 iterations, while our *eLane* only needs an average of 71 iterations.

To comprehensively evaluate the time efficiency of *eLane-EXT*, we randomly select 25 sampling areas as Section V-B and measure its time cost. The time cost of *eLane-EXT* in each sampling area contains the time cost of road feature exaction by LSE and the time cost of estimation by EM. As shown in Fig. 17, our *eLane* only takes about one-third of the time cost of [3], [5] and [34]. That is because [3] and [5] don't constrain the parameters of GMM and [34] requires to calculate the Hausdorff distance between each pair of points. The time cost of KDE-based solution [4] is similar to our *eLane* and while its accuracy is extremely low as shown in Fig. 18.

3) Accuracy: Since the GPS data from vehicles are widely distributed in the real-world crowdsourcing, data on some roads may be missing or the amount of data may be small. To evaluate the robustness of our *eLane*, we need to measure the accuracy of lane-level road information extraction on the sparse dataset. We select 10 sampling areas as Section V-B, and equally divide the data of each sampling area into ten parts, and compare the lane position accuracy F_p in *eLane* with those in [3]–[5], [34], [35] under different data volumes.

As shown in Fig. 18, it can be seen that the F_p of [4] is the lowest. The reason is that trajectories from a single vehicle are usually spread out over different lanes due to the inaccuracy of GPS or lane-changing, while the scheme of [4] regards each "kernel" as a lane and thus is seriously affected by the trajectory spread. The schemes of [3], [5] and [35] achieve similar accuracy, because their methodologies are all based on the GMM. Their differences are as follows: the scheme of [5] only optimizes the final estimates of [3], and the scheme of [35] simplifies the GMM of [3] and [5]. However, since the parameters of GMM are not be constrained properly, their F_ps are relatively low. We can also see that the increasing data volume even reduces F_p (e.g., 20%, 70% and 90%) in [34] and it proves that [34] is sensitive to outliers. That is because, it requires to calculate the Hausdorff distance between each pair of points, and the error of outliers will be accumulated in the process of extraction. We can conclude that our *eLane* achieves higher accuracy than the other schemes under the same volume of data. Moreover, to achieve the same accuracy, our *eLane* only needs about one-second of the data volume required by them. It indicates that our *eLane* still works effectively in the crowdsourcing scenario with sparse data.

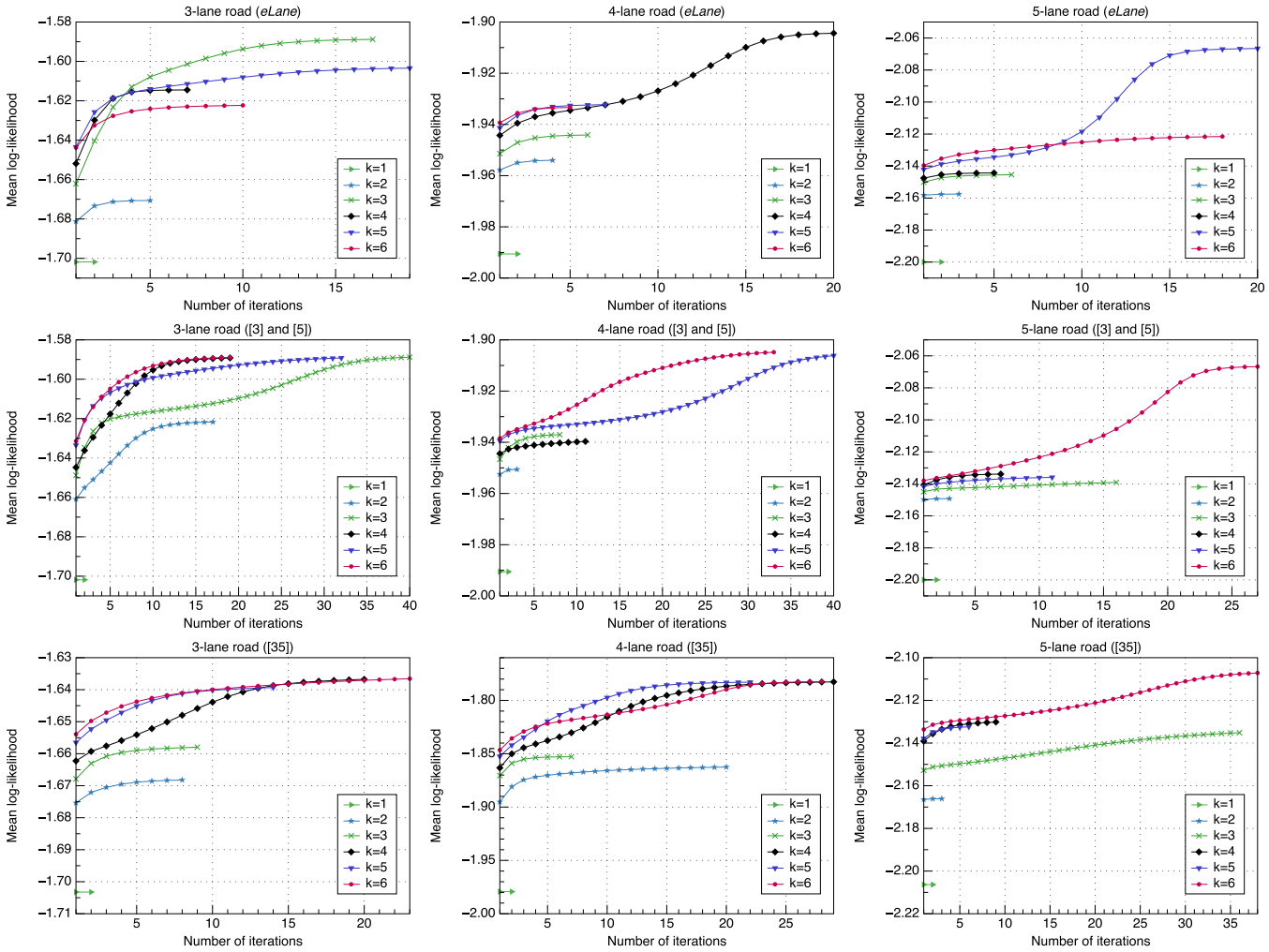


Fig. 16. Estimation process of *eLane*, [3], [5] and [35] when the iterative convergence threshold of EM algorithm is 10^{-4}.

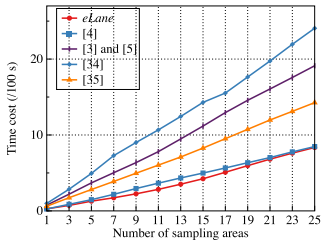


Fig. 17. Time cost of lane-level road information extraction.

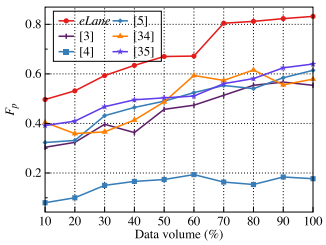


Fig. 18. Average accuracy of lane position extraction.

VI. Conclusion

In this article, we proposed an efficient and accurate lane-level map building scheme over the low-precision GPS trajectories via vehicle-based crowdsourcing. We designed an efficient segmentation and clustering algorithm to preprocess interleaved trajectories by considering the global similarity between trajectories, in which we also designed an entropy-based heuristic search method to determine the optimum clustering threshold. We explored the constrained GMM-EM to design an efficient and accurate lane-level road information extraction algorithm, which can greatly improve the extraction efficiency while achieving the high accuracy. We conducted comprehensive experiments on our *eLane* and evaluated its performance in comparison with the state-of-the-art works. Detailed experimental results show that our *eLane* has better efficiency and accuracy.

References

[1] K. Jiang, D. Yang, C. Liu, T. Zhang, and Z. Xiao, "A flexible multi-layer map model designed for lane-level route planning in autonomous vehicles," *Engineering*, vol. 5, no. 2, pp. 305–318, 2019.

[2] C. Ye, J. Li, H. Jiang, H. Zhao, L. Ma, and M. Chapman, "Semi-automated generation of road transition lines using mobile laser scanning data," *IEEE Trans. Intell. Transp. Syst.*, vol. 21, no. 5, pp. 1877–1890, May 2020.

[3] Y. Chen and J. Krumm, "Probabilistic modeling of traffic lanes from GPS traces," in *Proc. 18th SIGSPATIAL Int. Conf. Adv. Geographic Inf. Syst. GIS*, 2010, pp. 81–88.

[4] E. R. I. A. C. M. Uduwaragoda, A. S. Perera, and S. A. D. Dias, "Generating lane level road data from vehicle trajectories using kernel density estimation," in *Proc. 16th Int. IEEE Conf. Intell. Transp. Syst. (ITSC)*, Oct. 2013, pp. 384–391.

[5] L. Tang, X. Yang, Z. Dong, and Q. Li, "CLRIC: Collecting lane-based road information via crowdsourcing," *IEEE Trans. Intell. Transp. Syst.*, vol. 17, no. 9, pp. 2552–2562, Sep. 2016.

[6] G. Wang, Y. Wei, S. Qiao, P. Lin, and Y. Chen, *Generalized Inverses: Theory and Computations*. Berlin, Germany: Springer, vol. 53, 2018.

[7] P. D. McNicholas and T. B. Murphy, "Parsimonious Gaussian mixture models," *Statist. Comput.*, vol. 18, no. 3, pp. 285–296, Sep. 2008.

[8] M. R. Gupta, "Theory and use of the EM algorithm," *Found. Trends Signal Process.*, vol. 4, no. 3, pp. 223–296, 2010.

[9] X. Yang, L. Tang, X. Zhang, and Q. Li, "A data cleaning method for big trace data using movement consistency," *Sensors*, vol. 18, no. 3, p. 824, Mar. 2018.

[10] M. Vlachos, M. Hadjieleftheriou, D. Gunopulos, and E. Keogh, "Indexing multidimensional time-series," *VLDB J.*, vol. 15, no. 1, pp. 1–20, Jan. 2006.

[11] B. Lin and J. Su, "One way distance: For shape based similarity search of moving object trajectories," *GeoInformatica*, vol. 12, no. 2, pp. 117–142, Jun. 2008.

[12] L. Chen, M. T. Özsu, and V. Oria, "Robust and fast similarity search for moving object trajectories," in *Proc. ACM SIGMOD Int. Conf. Manage. Data - SIGMOD*, 2005, pp. 491–502.

[13] J.-G. Lee, J. Han, and K.-Y. Whang, "Trajectory clustering: A partition-and-group framework," in *Proc. ACM SIGMOD Int. Conf. Manage. Data - SIGMOD*, 2007, pp. 593–604.

[14] J.-G. Lee, J. Han, X. Li, and H. Gonzalez, "TraClass: Trajectory classification using hierarchical region-based and trajectory-based clustering," *Proc. VLDB Endowment*, vol. 1, no. 1, pp. 1081–1094, 2008.

[15] N. Wang, S. Gao, X. Peng, and M. Wang, "Research on fast and parallel clustering method for trajectory data," in *Proc. IEEE 24th Int. Conf. Parallel Distrib. Syst. (ICPADS)*, Dec. 2018, pp. 252–258.

[16] B. Han, L. Liu, and E. Omiecinski, "Road-network aware trajectory clustering: Integrating locality, flow, and density," *IEEE Trans. Mobile Comput.*, vol. 14, no. 2, pp. 416–429, Feb. 2015.

[17] X. Niu, T. Chen, C. Q. Wu, J. Niu, and Y. Li, "Label-based trajectory clustering in complex road networks," *IEEE Trans. Intell. Transp. Syst.*, vol. 21, no. 10, pp. 4098–4110, Oct. 2020.

[18] Y. Lou, C. Zhang, Y. Zheng, X. Xie, W. Wang, and Y. Huang, "Map-matching for low-sampling-rate GPS trajectories," in *Proc. 17th ACM SIGSPATIAL Int. Conf. Adv. Geographic Inf. Syst. GIS*, 2009, pp. 352–361.

[19] X. Zhou, F. Miao, H. Ma, H. Zhang, and H. Gong, "A trajectory regression clustering technique combining a novel fuzzy C-Means clustering algorithm with the least squares method," *ISPRS Int. J. Geo-Inf.*, vol. 7, no. 5, p. 164, Apr. 2018.

[20] S. Wang, Z. Bao, J. S. Culpepper, T. Sellis, and X. Qin, "Fast large-scale trajectory clustering," *Proc. VLDB Endowment*, vol. 13, no. 1, pp. 29–42, Sep. 2019.

[21] 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.

[22] R. V. Carneiro *et al.*, "Mapping road lanes using laser remission and deep neural networks," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, Jul. 2018, pp. 1–8.

[23] J. Jung and S. H. Bae, "Real-time road lane detection in urban areas using LiDAR data," *Electronics*, vol. 7, no. 11, p. 276, 2018.

[24] A. H. Lang, S. Vora, H. Caesar, L. Zhou, J. Yang, and O. Beijbom, "PointPillars: Fast encoders for object detection from point clouds," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 12697–12705.

[25] Y. Xing *et al.*, "Advances in vision-based lane detection: Algorithms, integration, assessment, and perspectives on ACP-based parallel vision," *IEEE/CAA J. Automatica Sinica*, vol. 5, no. 3, pp. 645–661, May 2018.

[26] S. Jung, J. Youn, and S. Sull, "Efficient lane detection based on spatiotemporal images," *IEEE Trans. Intell. Transp. Syst.*, vol. 17, no. 1, pp. 289–295, Jan. 2016.

[27] J. H. Yoo, S.-W. Lee, S.-K. Park, and D. H. Kim, "A robust lane detection method based on vanishing point estimation using the relevance of line segments," *IEEE Trans. Intell. Transp. Syst.*, vol. 18, no. 12, pp. 3254–3266, Dec. 2017.

[28] A. Kasmi, D. Denis, R. Aufrere, and R. Chapuis, "Probabilistic framework for ego-lane determination," in *Proc. IEEE Intell. Vehicles Symp.*, Jun. 2019, pp. 1746–1752.

[29] W. Jang, J. An, S. Lee, M. Cho, M. Sun, and E. Kim, "Road lane semantic segmentation for high definition map," in *Proc. IEEE Intell. Vehicles Symp. (IV)*, Jun. 2018, pp. 1001–1006.

[30] Q. Zou, H. Jiang, Q. Dai, Y. Yue, L. Chen, and Q. Wang, "Robust lane detection from continuous driving scenes using deep neural networks," *IEEE Trans. Veh. Technol.*, vol. 69, no. 1, pp. 41–54, Jan. 2020.

[31] Y.-W. Seo, C. Urmson, and D. Wettergreen, "Ortho-image analysis for producing lane-level highway maps," in *Proc. 20th Int. Conf. Adv. Geographic Inf. Syst. - SIGSPATIAL*, 2012, pp. 506–509.

[32] A. Zang, R. Xu, Z. Li, and D. Doria, "Lane boundary extraction from satellite imagery," in *Proc. 1st ACM SIGSPATIAL Workshop High-Precis. Maps Intell. Appl. Auto. Vehicles - AutonomousGIS*, 2017, pp. 1–8.

[33] L. Tang, X. Yang, Z. Kan, and Q. Li, "Lane-level road information mining from vehicle GPS trajectories based on naïve Bayesian classification," *ISPRS Int. J. Geo-Inf.*, vol. 4, no. 4, pp. 2660–2680, 2015.

[34] T. Zhang, S. Arrigoni, M. Garozzo, D.-G. Yang, and F. Cheli, "A lane-level road network model with global continuity," *Transp. Res. C, Emerg. Technol.*, vol. 71, pp. 32–50, Oct. 2016.

[35] M. A. Arman and C. M. J. Tampère, "Road centreline and lane reconstruction from pervasive GPS tracking on motorways," *Procedia Comput. Sci.*, vol. 170, pp. 434–441, Jan. 2020.

[36] T. Eiter and H. Mannila, "Computing discrete fréchet distance," Citeseer, Christian Doppler Lab. Expert Syst., TU Vienna, Vienna, Austria, Tech. Rep. CD-TR 94/64, 1994.

[37] S. Karagiorgou and D. Pfoser, "On vehicle tracking data-based road network generation," in *Proc. 20th Int. Conf. Adv. Geographic Inf. Syst. - SIGSPATIAL*, 2012, pp. 89–98.

[38] X. Yang, L. Tang, L. Niu, X. Zhang, and Q. Li, "Generating lane-based intersection maps from crowdsourcing big trace data," *Transp. Res. Part C: Emerg. Technol.*, vol. 89, pp. 168–187, 2018.

[39] C. E. Shannon, "A mathematical theory of communication," *Bell Syst. Tech. J.*, vol. 27, no. 3, pp. 379–423, Jul./Oct. 1948.

[40] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi, "Optimization by simulated annealing," *Science*, vol. 220, no. 4598, pp. 671–680, 1983.

[41] M. Piorkowski, N. Sarafijanovic-Djukic, and M. Grossglauser. (2009). *CRAWDAD Dataset Epfl*. Downloaded From. [Online]. Available: https://crawdad.org/epfl/mobility/20090224

[42] N. Eagle, "Txteagle: Mobile crowdsourcing," in *Proc. Int. Conf. Internationalization, Design Global Develop.*, Jul. 2009, pp. 447–456.

[43] S. Sarker, M. A. Razzaque, M. M. Hassan, A. Almogren, G. Fortino, and M. Zhou, "Optimal selection of crowdsourcing workers balancing their utilities and platform profit," *IEEE Internet Things J.*, vol. 6, no. 5, pp. 8602–8614, Oct. 2019.

[44] Y. Tong, J. She, B. Ding, L. Wang, and L. Chen, "Online mobile micro-task allocation in spatial crowdsourcing," in *Proc. IEEE 32nd Int. Conf. Data Eng. (ICDE)*, May 2016, pp. 49–60.

[45] J. Shu, K. Yang, X. Jia, X. Liu, C. Wang, and R. Deng, "Proxy-free privacy-preserving task matching with efficient revocation in crowdsourcing," *IEEE Trans. Depend. Sec. Comput.*, early access, Oct. 12, 2019, doi: 10.1109/TDSC.2018.2875682.

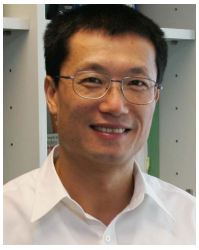
[46] Z. Chen, J. Zhang, and D. Tao, "Progressive LiDAR adaptation for road detection," *IEEE/CAA J. Automatica Sinica*, vol. 6, no. 3, pp. 693–702, May 2019.

[47] C. Sun *et al.*, "Proximity based automatic data annotation for autonomous driving," *IEEE/CAA J. Automatica Sinica*, vol. 7, no. 2, pp. 395–404, Mar. 2020.

[48] J. Bi *et al.*, "Application-aware dynamic fine-grained resource provisioning in a virtualized cloud data center," *IEEE Trans. Autom. Sci. Eng.*, vol. 14, no. 2, pp. 1172–1184, Apr. 2017.

Jiangang Shu (Member, IEEE) received the Ph.D. degree in computer science from the City University of Hong Kong (CityU), Hong Kong, in 2019. He was a postgraduate Visiting Student with the Secure Mobile Centre, Singapore Management School, Singapore, and a Post-Doctoral Fellow with CityU. He is currently a Research Scientist with the Cyberspace Security Research Center, Peng Cheng Laboratory, Shenzhen, China. His research interests include AI privacy, crowdsourcing and cloud security, the IoT security, applied cryptography, data security and privacy, and searchable encryption.

Songlei Wang received the B.E. degree from the China University of Petroleum (East China), Qingdao, China, in 2018. He is currently pursuing the M.A.Eng. degree in computer science with the Department of Computer Science, Harbin Institute of Technology, Shenzhen, China. His research interests include security and privacy in crowdsourcing, and data mining.



Xiaohua Jia (Fellow, IEEE) received the B.Sc. and M.Eng. degrees from the University of Science and Technology of China in 1984 and 1987, respectively, and the D.Sc. degree in information science from The University of Tokyo in 1991. He is currently the Chair Professor of the Department of Computer Science, City University of Hong Kong. His research interests include cloud computing and distributed systems, computer networks, and mobile computing. He is the General Chair of ACM MobiHoc 2008, the TPC Co-Chair of IEEE GlobeCom 2010 Ad Hoc and Sensor Networking Symposium, and the Area-Chair of IEEE INFOCOM 2010 and 2015. He was an Editor of IEEE TRANSACTIONS ON PARALLEL AND DISTRIBUTED SYSTEMS from 2006 to 2009. He is an Editor of IEEE INTERNET OF THINGS, *Wireless Networks*, *World Wide Web Journal*, and *Journal of Combinatorial Optimization*.

Weizhe Zhang (Senior Member, IEEE) is currently a Professor with the School of Computer Science and Technology, Harbin Institute of Technology, China. He has authored more than 100 academic papers in journals, books, and conference proceedings. His research interests include parallel computing, distributed computing, cloud and grid computing, and computer networks. He serves on a number of journal editorial boards. He has edited more than ten international journal special issues as a guest editor and has served for many international conferences as the chair or a committee member.

Ruitao Xie received the B.Eng. degree from the Beijing University of Posts and Telecommunications in 2008 and the Ph.D. degree in computer science from the City University of Hong Kong in 2014. She is currently an Assistant Professor with the College of Computer Science and Software Engineering, Shenzhen University. Her research interests include AI networking and mobile computing, distributed systems, and cloud computing.

Hejiao Huang (Member, IEEE) received the Ph.D. degree in computer science from the City University of Hong Kong, Hong Kong, in 2004. She was an Invited Professor with the Institut National de Recherche en Informatique et Automatique, Rennes, France. She is currently a Professor with the Shenzhen Graduate School, Harbin Institute of Technology, Shenzhen, China. Her research interests include cloud computing, trustworthy computing, formal methods for system design, and wireless networks.