Chapter 11
Automatic Urban Road Network Extraction From Massive GPS Trajectories of Taxis

Song Gao, Mingxiao Li, Jinmeng Rao, Gengchen Mai, Timothy Prestby, Joseph Marks, and Yingjie Hu

11.1 Introduction

Increasingly, mobile Internet, ubiquitous sensors (Hancke and Hancke Jr. 2013) and growing Volunteering Geographic Information (VGI) (Goodchild 2007) altogether boost the construction of transportation information infrastructures (Shaw 2010). Urban road networks, as the important carriers of transportation in cities, provide basic support for human & goods transportation and various Location-Based Services (LBS) such as vehicle route planning and navigation, which are the keys to smart transportation. However, how to build and update the road network in a rapid and cost-effective way still remains to be a challenging problem.

Traditional methods such as field surveying and map digitalization are usually costly and cannot produce up-to-date urban road networks in time (Tao 2000). With the rapid development of information and communication technologies (ICTs) and positioning technologies such as the Global Positioning System (GPS), huge amounts of vehicle movement trajectory data have been accumulated (Liu et al.

S. Gao (✉) · M. Li · J. Rao · T. Prestby · J. Marks
GeoDS Lab, Department of Geography, University of Wisconsin-Madison, Madison, WI, USA
e-mail: song.gao@wisc.edu

M. Li
GeoDS Lab, Department of Geography, University of Wisconsin-Madison, Madison, WI, USA

State Key Laboratory of Resources and Environmental Information System, Institute of Geographic Sciences and Natural Resources Research, Chinese Academy of Sciences, Beijing, China

G. Mai
STKO Lab, Department of Geography, University of California, Santa Barbara, CA, USA

Y. Hu
GeoAI Lab, Department of Geography, University at Buffalo, Buffalo, NY, USA

M. Werner, Y.-Y. Chiang (eds.), *Handbook of Big Geospatial Data*,
https://doi.org/10.1007/978-3-030-55462-0_11

2012). By leveraging these massive GPS trajectories, automatic construction and updates to road networks can be achieved in near real-time (Li et al. 2012). Although the idea for the extraction of urban road networks from GPS trajectory data is intuitive, there is still a considerable gap between the raw GPS traces and the road network structure. On the one hand, the GPS trajectories have non-negligible errors due to the inherent noise in GPS, which makes it difficult to distinguish two closely located road segments in some cases (Cao and Krumm 2009). On the other hand, complex urban environment such as “urban canyon” often leads to the deterioration in the GPS precision, making the trajectories less accurate to represent the road segments. In addition, the diversity and complexity of the road network structure in some places (e.g., roundabouts, parking lots) also cause challenges for road network extraction from raw GPS trajectories. Besides the GPS precision and the complex road network structure, the preprocessing of trajectory data is another challenge. A large proportion of the GPS data on straight road segments are redundant since fewer points are already enough to reconstruct the linear road segments, whereas the curved roads required more points. Also, when there is traffic congestion, more redundant data (e.g., stay points) will be produced (Zheng 2015). These situations may cause high computational costs and limit the road network extraction efficiency. In addition, trajectory outliers (i.e., anomalies) will make it non-trivial to reconstruct the road network and need to be addressed during the preprocessing step (Zheng 2015; Wang et al. 2019).

To this end, in this chapter, we focus on the trajectory sampling, compression, and clustering techniques to update road geometry information with regard to the spatial coverage and topological connectivity. We conduct a literature review in the following section and then propose a geospatial-big-data-driven framework to achieve an automatic road network extraction. Specifically, we first introduce a trajectory compression approach to reduce redundant trajectory data and avoid the unnecessary computational cost. Then we present an anisotropic density-based trajectory clustering with noise (ADCN) algorithm (Mai et al. 2018) for identifying the trajectory points on the road segments, and finally, a kernel density estimation and vectorization approach is utilized for road network extraction.

11.2 Literature Review

Existing literature on road network extraction methods can be classified into two categories: density-based approaches and cluster-based approaches.

11.2.1 Density-Based Approaches

The first category mainly relies on density estimation and raster processing techniques. It converts trajectory data into raster data based on density and extracts the road network using morphological methods (Davies et al. 2006; Wu et al. 2007; Shi et al. 2009; Zhao et al. 2011; Biagioni and Eriksson 2012; Jiang et al. 2012; Wang et al. 2015b; Kuntzsch et al. 2016; Tang et al. 2017). For example, Davies et al. (2006) first generated a 2D histogram based on the GPS trajectories, then applied a global density threshold on the cells to find potential road areas, and finally computed road centerlines based on the Voronoi graph. Shi et al. (2009) converted vehicle GPS trajectories into a road network bitmap, then computed the road network skeleton on the bitmap, and finally extracted the vector road network map data from the skeleton. Biagioni and Eriksson (2012) generated a road network skeleton based on a kernel-density method and use a map-matching method to achieve topology reconstruction. Kuntzsch et al. (2016) formulated an explicit intersection model which integrated consistency measurements with the raw trajectory data to better perform geometry and topology reconstruction of the network; Tang et al. (2017) employed Delaunay triangulation with the trajectory stream fusion to improve the map generation accuracy. However, the difference in trajectory density has a great influence on the extraction effect, which could make these methods unreliable in cases with heterogenous trajectory density.

11.2.2 Cluster-Based Approaches

The second category adopted clustering methods to generate road networks (Edelkamp and Schrödl 2003; Lee et al. 2007; Worrall et al. 2007; Wu et al. 2013; Cao and Krumm 2009; Wang et al. 2015a; Aronov et al. 2016; Stanojevic et al. 2018). Trajectory clustering is usually used to find representative trajectories shared by different objects such as individuals or vehicles (Zheng 2015). For example, Gaffney and Smyth (1999) and Cadez et al. (2000) used a regression mixture model and an Expectation-Maximization (EM) model to cluster trajectories according to the overall distance between two trajectories. Lee et al. (2007) proposed TRACLUS, a modified density-based trajectory clustering algorithm for grouping close trajectory line segments into clusters, which is based on the original point-based DBSCAN algorithm. Li et al. (2010) further introduced an incremental clustering algorithm that reduces the computational and storage cost. In practice, trajectory clustering can be naturally used for road network extraction. Edelkamp and Schrödl (2003) applied the K-means algorithm to cluster the trajectories and then fit the road centerline with the spline curve. This approach is suitable for data with small density difference, low noise, and high frequency sampling. Worrall et al. (2007) used clustering to extract the skeleton points of the road network and used the least squares regression method to connect the skeleton points to generate the road network. Cao and Krumm (2009) classified the GPS traces using simulations of physical forces among the traces, and then merged the classified traces into a graph representation of the road network structure. Wang et al. (2015a) determined a proper circle boundary to cluster trajectory data into intersections and used the core points to build the road networks. Stanojevic et al. (2018) formulated

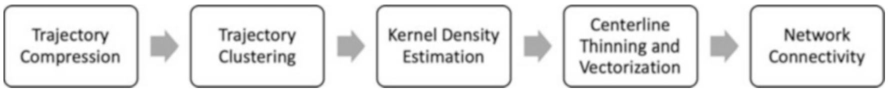

Fig. 11.1 The trajectory big data processing workflow in this study

the road network generation task as a network alignment optimization task and proposed an offline algorithm that clustered GPS points for graph construction as well as an online algorithm that can create and update the road network. However, the trajectory points along the road networks include linear features with a continuously changing density which makes current clustering methods tend to either create an increasing number of small clusters or add noise points into large clusters. Therefore, incorporating directional information into clustering methods has become an efficient way to cluster anisotropic distributed points and enhance extraction performance (Mai et al. 2018).

11.3 Methodology

In this section, we present the details of our road map generation method using GPS trajectories. As shown in Fig. 11.1, the trajectory big data processing workflow can be divided into the following steps. First, we utilized a trajectory compression approach to simplify the trajectory data and reduce unnecessary computational cost. Second, we applied the Anisotropic Density-based Clustering with Noise (ADCN) algorithm (Mai et al. 2018) to identify the trajectory points that were along the road networks with high confidence. Third, a kernel density estimation (KDE) approach was used to generate a continuous surface. Fourth, high-density areas were selected as candidates to further extract the road centerlines using thinning and vectorization operators. Finally, the extracted road network connectivity should be evaluated.

11.3.1 Trajectory Compression

With the rapid development of ICT technologies and positioning devices, the spatiotemporal resolution of trajectory data has unprecedentedly increased. However, a large amount of trajectory data lead to high computational costs, and a large storage space is necessary to support the data processing and management. With regard to road network extraction, it is straightforward to reconstruct the road segments as long as the trajectory points at the intersections can be obtained and then connected. Thus, a trajectory compression method was applied to simplify the trajectory data points and improve extraction efficiency.

The main purpose of trajectory compression in road network extraction is to maintain the shape of the trajectories. Several popular algorithms for line simplification include Douglas-Peucker algorithm (Douglas and Peucker 1973), Reumann-Witkam algorithm (Reumann and Witkam 1974), Lang simplification (Lang 1969), Opheim simplification (Opheim 1982), etc. The evaluation and comparison of these line simplification algorithms for vector generalization were conducted by Shi and Cheung (2006). McMaster (1989) proposed a conceptual model including a sequential set of five procedures for processing linear data with focuses on *geometric simplification* and *smoothing*. In our method, we applied the widely used Douglas-Peucker simplification algorithm for trajectory simplification.[1] A graphic illustration of the Douglas-Peucker algorithm is shown in Fig. 11.2. A sequence of points (P_1, P_2, P_3, P_4, P_5, P_6) represent a trajectory in Fig. 11.2a. To simplify the trajectory, we first mark the first point P_1 and the last point P_6 as endpoints and added them to the reserved point set. Then, the point that is furthest from the line segment with the endpoints is found, and the distance between the point and the line segment is calculated. If the distance is larger than a compression threshold α, the point is marked as an endpoint and added to the reserved point set (e.g. P_4 in in Fig. 11.2b and P_5 in Fig. 11.2c). Otherwise, the points between the endpoints would be discarded (e.g. P_2 and P_3 in Fig. 11.2c). The same process is iteratively performed until all the trajectory points are marked as an endpoint or discarded (Fig. 11.2d). Finally, the points in the reserved point set are sorted according to the original trajectory sequence and linked to generate the compressed trajectory.

11.3.2 Identification of the Trajectory Points Along the Road

Density-based clustering algorithms such as DBSCAN have been widely used for spatial knowledge discovery such as the detection of urban areas of interest (Hu et al. 2015) and vague cognitive regions (Gao et al. 2017). The DBSCAN algorithm offers several key advantages compared with other clustering algorithms such as K-Means. DBSCAN can discover clusters with arbitrary shapes, are robust to noise, and do not require prior knowledge (Ester et al. 1996). However, the trajectory points demonstrate clear anisotropic spatial processes, which makes these methods tend to either create an increasing number of small clusters or add noise points into large clusters. Therefore, in this section, we apply a novel anisotropic density-based clustering algorithm with noise (ADCN) (Mai et al. 2018) to cluster anisotropic points for identifying the trajectory points along the roads. The codes for implementing the ADCN algorithm are available in both Javascript[2] and Python[3] programming languages.

[1]Python code for the Douglas-Peucker algorithm https://pypi.org/project/simplification/.

[2]Javascript implementation of the ADCN algorithm at: https://github.com/gengchenmai/adcn.

[3]Python implementation of the ADCN algorithm at: https://github.com/gissong/ADCN.

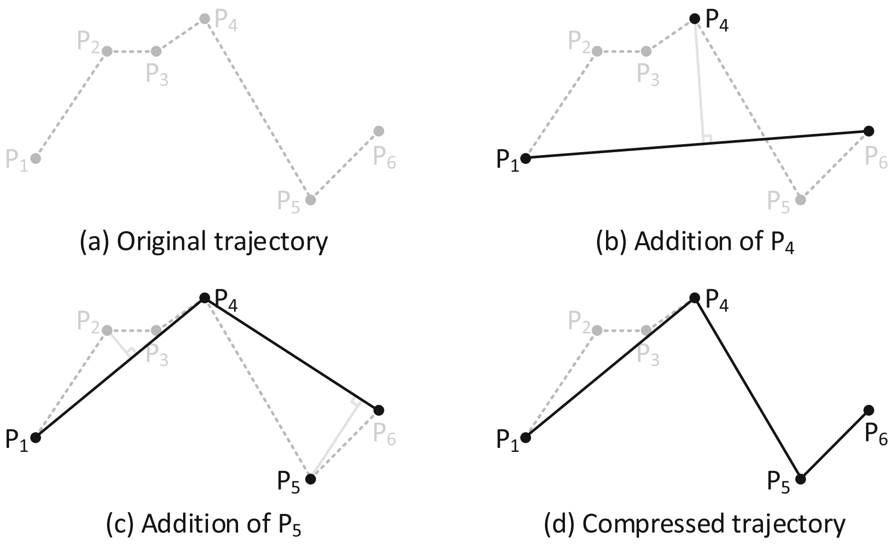

Fig. 11.2 Illustration of the Douglas-Peucker trajectory compression algorithm

11.3.2.1 Density-Based Spatial Clustering of Applications with Noise (DBSCAN)

The used ADCN algorithm was modified based on the DBSCAN algorithm. Thus, before we describe the ADCN algorithm in detail, we first introduce some fundamental concepts of the DBSCAN algorithm (Ester et al. 1996). The key idea of the DBSCAN algorithm is that: given a set of points, it groups nearby points (i.e., the points in high-density areas) together and marks the points in low-density areas as outliers. In order to group the points based on the density, the *Eps-neighborhood* of a point is defined (see Definition 1).

Definition 1 (Eps-neighborhood of a point) The *Eps-neighborhood* $N_{Eps}(p_i)$ of point p_i in a dataset D is defined as all the points within the scan circle centered at p_i with a radius *Eps*, which is expressed as follows:

$$N_{Eps}\left(p_i\right) = \left\{p_j\left(x_j, y_j\right) \in D | dist\left(p_i, p_j\right) \leq Eps\right\}$$

where $dist(p_i, p_j)$ is the distance between point p_i and point p_j.

There are two kinds of points in a cluster: *core points* (i.e., points inside of the cluster) and *border points* (i.e., points on the border of the cluster). One intuition is that for each point of a cluster, an Eps-neighborhood should contain at least a minimum number of points (*MinPts*). However, an Eps-neighborhood of a border point usually contains much fewer points than an Eps-neighborhood of a core point, and it is hard to choose the representative *MinPts* for all points. Thus, DBSCAN introduces three basic concepts: directly density-reachable, density-reachable and density-connected (see Definitions 2, 3, and 4).

Definition 2 (directly density-reachable) A point p is *directly density-reachable* from a point q wrt. *Eps* and *MinPts* if:

1. $p \in N_{Eps}(q)$ and
2. $|N_{Eps}(q)| \geq MinPts$ (core point condition).

Definition 3 (density-reachable) A point p is *density-reachable* from a point q wrt. *Eps* and *MinPts* if there is a chain of points $p_1, \ldots, p_n, p_1 = q, p_n = p$ such that p_{i+1} is directly density-reachable from p_i.

Definition 4 (density-connected) A point p is *density-connected* to a point q wrt. *Eps* and *MinPts* if there is a point O such that p and q are density-reachable from O wrt. *Eps* and *MinPts*.

Figure 11.3 illustrates core points, border points, density-reachability, and density-connectivity in DBSCAN. These definitions can be used to further define the density-based notion of a cluster or a noise in DBSCAN. Specifically, the core points and border points are grouped into clustered points while the points that do not belong to any cluster are the noise points (as shown in Fig. 11.4).

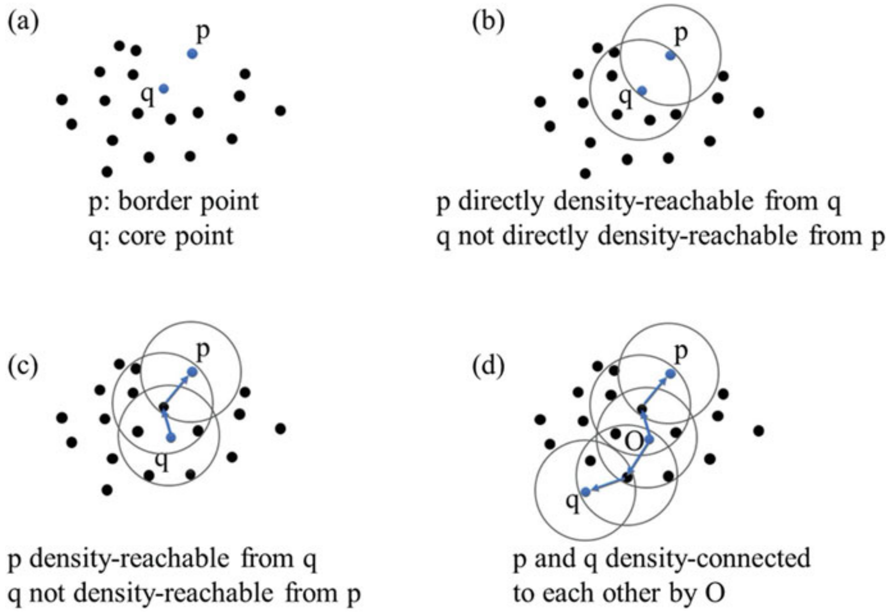
(a)
p
q
p: border point
q: core point
(b)
p
q
p directly density-reachable from q
q not directly density-reachable from p
(c)
p
q
p density-reachable from q
q not density-reachable from p
(d)
p
O
q
p and q density-connected
to each other by O

Fig. 11.3 Illustration of core points, border points, density-reachability and density-connectivity in the DBSCAN algorithm (Ester et al. 1996)

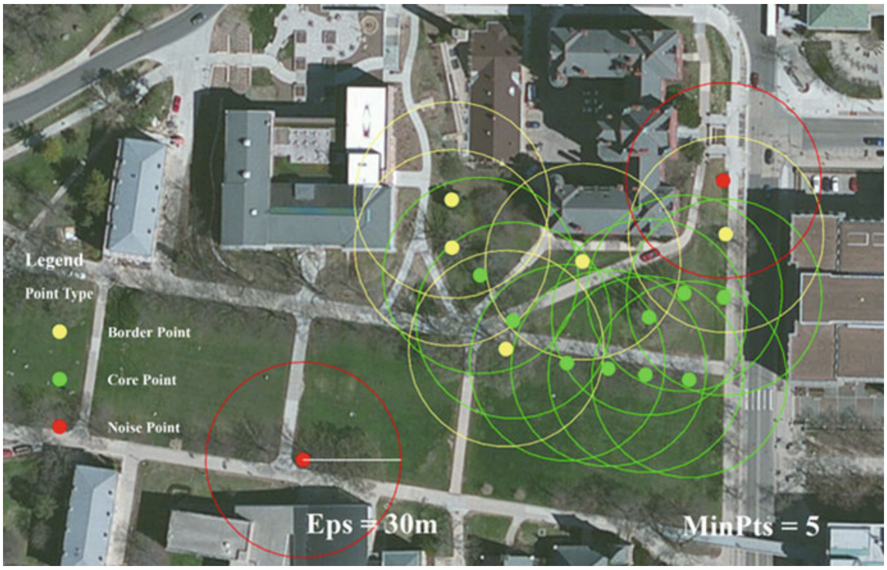

Fig. 11.4 Illustration of core points, border points, and noise points in the DBSCAN algorithm

11.3.2.2 Anisotropic Perspective on Local Point Density

One key consideration in the ADCN algorithm is the anisotropic perspective on local point density in different directions. Without predefined direction information from spatial point datasets, one has to compute the local major direction for each point based on the spatial distribution of neighboring points. The standard deviation ellipse (SDE) (Yuill 1971) is a suitable method to get the major direction of a point set. Given *n* points, the SDE constructs an ellipse to represent the orientation and arrangement of these points. The center of this ellipse is defined as the geometric center of these *n* points and is calculated as:

$$\overline{X} = \sum_{i=1}^{n} x_i \Big/ n, \quad \overline{Y} = \sum_{i=1}^{n} y_i \Big/ n$$

The coordinates (x_i,y_i) of each point are normalized to the deviation from the mean center point:

$$\widetilde{x_i} = x_i - \overline{X}, \;\; \widetilde{y_i} = y_i - \overline{Y}$$

Thus, the semi-major axes σ_x and the semi-minor axes σ_y of SDE are calculated as:

$$\sigma_x = \sqrt{\sum_{i=1}^{n} \left(\widetilde{x_i} \cos\theta + \widetilde{y_i} \sin\theta\right)^2 \Big/ n}, \;\; \sigma_y = \sqrt{\sum_{i=1}^{n} \left(\widetilde{y_i} \cos\theta - \widetilde{x_i} \sin\theta\right)^2 \Big/ n}$$

where θ is the rotation angle and the SDE will be further used as the search polygon for clustering neighboring points (Mai et al. 2018).

11.3.2.3 Anisotropic Density-Based Clusters with Noise (ADCN) Algorithm

In order to introduce an anisotropic perspective to density-based clustering algorithms, we have to revise some definitions. First, the original *Eps-neighborhood* of a point in a dataset D is defined by DBSCAN, as given in Definition 1. Such a scan circle results in an isotropic perspective on clustering. However, an anisotropic assumption will be more appropriate for trajectory points along the roads. Intuitively, in order to introduce the anisotropic perspective into DBSCAN, we can employ a scan ellipse instead of a circle to define the Eps-neighborhood of each point. Before that, we defined a set of points around a point to derive the scan ellipse;

Definition 5 (Search-neighborhood of a point) The $\mathrm{k_{th}}$ nearest neighbor $KNN(p_i)$ of point p_i. Here $k = MinPts$ and $KNN(p_i)$ does not include p_i itself.

After determining the search-neighborhood of a point, it is possible to define the *Eps-ellipse-neighborhood region* (see Definition 6) and the *Eps-ellipse-neighborhood* (see Definition 7) of each point.

Definition 6 (Eps-ellipse-neighborhood region of a point) An ellipse ER_i is called an Eps-ellipse-neighborhood region of a point p_i if:

1. Ellipse ER_i is centered at point p_i.
2. Ellipse ER_i is scaled from the standard deviation ellipse SDE_i computed from the search-neighborhood $S(p)_i$ of point p_i.
3. $\frac{\sigma'_{max}}{\sigma'_{min}} = \frac{\sigma_{max}}{\sigma_{min}}$ where σ'_{max}, σ'_{min}, σ_{max}, σ_{min} are the length of the semi-long and semi-short axes of ellipse ER_i and ellipse SDE_i.
4. $Area(ER_i) = \pi Eps^2$.

According to Definition 6, the Eps-ellipse-neighborhood region of a point is computed based on the search-neighborhood of a point. Each point should have a unique *MinPts*, as long as the search-neighborhood of the current point has at least two points for the computation of the standard deviation ellipse.

Definition 7 (Eps-ellipse-neighborhood of a point) An Eps-ellipse-neighborhood $EN_{Eps}(p_i)$ of point p_i is defined as all the points inside the ellipse ER_i, which can be expressed as:

$$EN_{Eps}(p_i) = \left\{ p_j\left(x_j, y_j\right) \in D \middle| \frac{\left(\left(y_j - y_i\right)\sin\theta_{max} + \left(x_j - x_i\right)\cos\theta_{max}\right)^2}{a^2} + \frac{\left(\left(y_j - y_i\right)\cos\theta_{max} - \left(x_j - x_i\right)\sin\theta_{max}\right)^2}{b^2} \le 1 \right\}$$

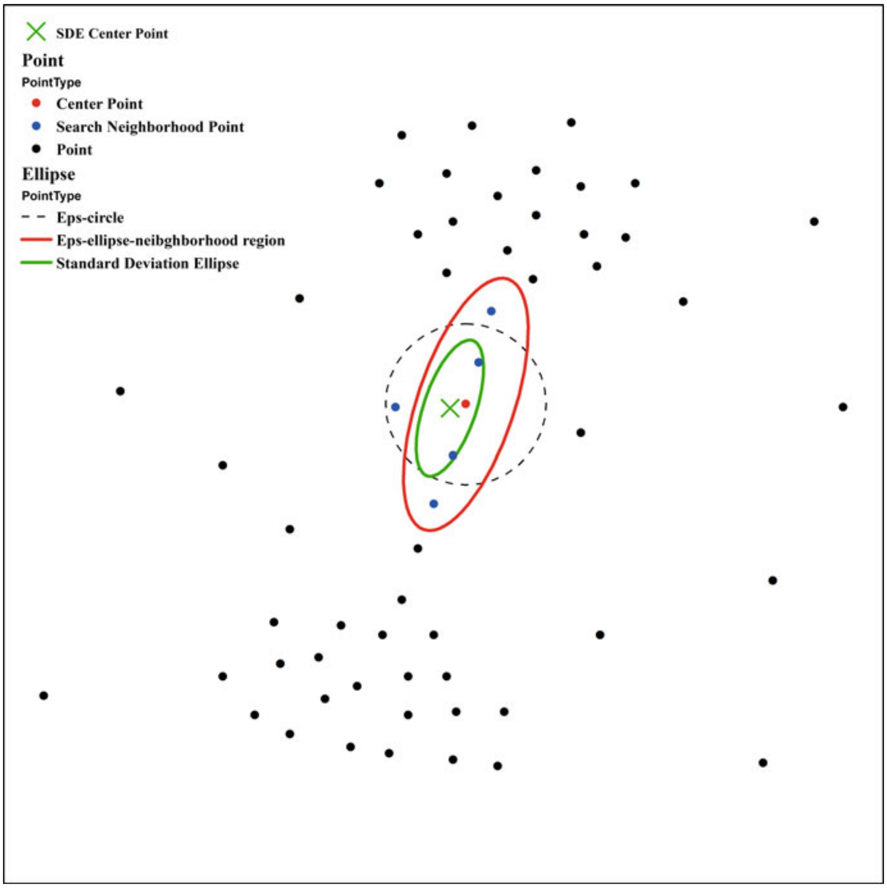
SDE Center Point
Point
PointType
Center Point
Search Neighborhood Point
Point
Ellipse
PointType
Eps-circle
Eps-ellipse-neibghborhood region
Standard Deviation Ellipse

Fig. 11.5 Illustration of the ADCN algorithm (Mai et al. 2018)

Equipped with Definitions 1, 5, 6, and 7, we can introduce the anisotropic perspective to density-based clustering algorithms. The definitions of directly density reachable, density reachable, cluster, and noise in ADCN are similar to DBSCAN, which will not be repeated here. Figure 11.5 illustrates the related definitions for ADCN. The red point in the figure represents the current center point. The blue points indicate the search-neighborhood of the corresponding center point according to Definition 5. The green ellipse and the green cross stand for the standard deviation ellipse constructed from the corresponding search-neighborhood and the center point. The red ellipse is the scale-transformed Eps-ellipse-neighborhood region according to Definition 6, whereas the dashed-line circle indicates a traditional scan circle in DBSCAN. As can be seen in Fig. 11.5, ADCN could exclude the point to the left of the linear bridge pattern in the clustering process, whereas DBSCAN still includes it.

11.3.2.4 ADCN Algorithm in Road Network Extraction

The abovementioned ADCN algorithm takes the same parameters (*MinPts* and *Eps*) as the DBSCAN algorithm which must be decided before clustering. This is for good reasons, as the proper selection of DBSCAN parameters has been well studied, and ADCN can easily replace DBSCAN without any changes to established workflows.

The ADCN method starts with an arbitrary point p_i in a point dataset D and discovers all the core points which are density reachable from point p_i along the major direction. The result of the points in the clusters will be regarded as the trajectory points on the road. In order to take care of situations where all points of the search-neighborhood $S(p_i)$ of point p_i are strictly on the same line, the short axis of the Eps-ellipse-neighborhood region ER_i becomes zero, and its long axis becomes infinity. This means that $EN_{Eps}(p_i)$ is reduced to a straight line. The process of constructing the Eps-ellipse-neighborhood $EN_{Eps}(p_i)$ of point p_i becomes a point-on-line query. Furthermore, the ADCN method uses a $\mathrm{k_{th}}$ nearest neighborhood of point p_i as the search-neighborhood. Here, the center point p_i will not be included in its $\mathrm{k_{th}}$ nearest neighborhood. The runtimes of ADCN are heavily dominated by the search-neighborhood query which is executed on each point. Hence, the time complexities of ADCN, DBSCAN, and OPTICS are $O(n^2)$ without a spatial index and $O(n \log n)$otherwise (Kolatch 2001; Mai et al. 2018).

11.3.3 Road Network Generation

11.3.3.1 Road Density Surface Generation

After identifying the compressed and clustered GPS trajectory points on the roads, the kernel density estimation (KDE) is introduced to fit a smooth surface. To fit the surface, we used 30 m*30 m square grid cells to divide the space and calculated the trajectory point density of each cell. Let $d_1, d_2, \ldots, d_n$ be a given set of trajectory point densities. The kernel density estimator is defined as:

$$\hat{f}_{(d)} = \frac{1}{nh^d} \sum_{i=1}^{n} K\left(\frac{d - d_i}{h}\right)$$

where n is the number of density sets, h denotes the bandwidth parameter, and K is a kernel function. The kernel surface value is highest at the location of the center and decreases with increasing distance from the center until reaching zero at the search radius. The density at each grid cell is calculated by adding the weighted values under the kernel surface where it overlays the raster cell center. The kernel function used in this work is based on the quadratic kernel function (Silverman 1986) described as follows:

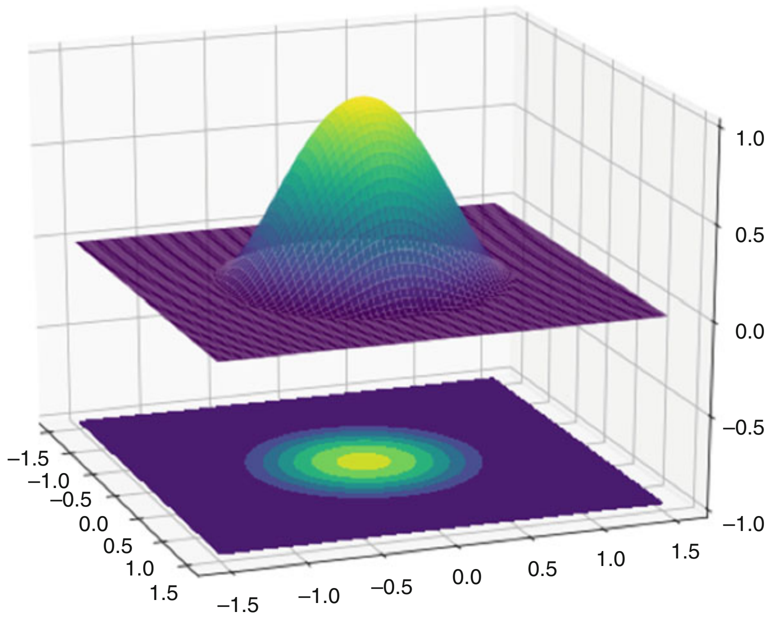

Fig. 11.6 A 3D Visualization of the selected quadratic kernel function. The vertical axis represents the derived K function value

$$K_{(d)} = \begin{cases} 3\pi^{-1}\left(1 - d^T d\right)^2 & if\ d^T d < 1 \\ 0 & otherwise \end{cases}$$

Figure 11.6 shows a 3D plot of the selected quadratic kernel function. Note that there are two reasons we choose the quadratic kernel function. First, the quadratic kernel function's density estimates have higher differentiability properties. Second, it can be calculated more quickly than the Gaussian kernel, which is suitable for massive-scale trajectory points with fine spatial resolution.

11.3.3.2 Collapse Surface to Centerline

After the previous steps, the areas whose density is above a threshold β are selected as the candidates. In this step, the main purpose is to extract the centerline of the candidate areas while keeping the road network topology. The thinning operator proposed by Yuan et al. (2012) was performed to remove certain grid cells from the candidate areas. For a given grid cell in the candidate area, whether it should be removed depends on its 8-neighboring cells. This method first divided the binary image (i.e., is a road pixel or not) into two disjointed subfields in a checkerboard pattern. Then, iterations were performed to remove redundant neighboring cells. The algorithm ensured that the connectivity of the cells was preserved when a cell was deleted. A combination of the thinning operator and the raster-to-vector operation converted the KDE surface of compressed trajectories into road centerlines. The quality measures of the extracted road networks including accuracy (correctness), coverage completeness, redundancy, and connectivity will be discussed with case study experiments in Sect. 11.4.2.

11.4 Case Study

11.4.1 Data

We applied the above introduced trajectory compression and clustering workflow in the DiDi research open data "November 2016, Chengdu City Second Ring Road Regional Trajectory Data Set"[4] to extract the road information. Figure 11.7 shows the KDE visualization of over 1 million e-hailing trip origins and destinations and part of the extracted road geometry density map from 181,172 trip trajectories in one day. The date range of this dataset is from November 1 to November 30, 2016 and the temporal sampling resolution is about 2~5 seconds. The original data is about 50 GB, and about 10 GB after compression.

11.4.2 Experiment

11.4.2.1 Evaluation Metrics

In order to determine the effectiveness of our algorithm that generates big-data-driven road networks, we performed vector-based quality measures (Wiedemann et al. 1998). These quality measures were done by comparing a reference road network data layer to our extracted road network data layers (in Fig. 11.8). To begin, the road networks were divided into small pieces of the same length. Next, a buffer was created as a zone with a consistent width that encircles each line segment for a given analysis. To determine if the extracted network roads match the reference roads, we constructed a buffer around the reference road data and determined if the portion of extracted roads inside of the buffer met our requirements. This is the *correctness*. Similarly, we constructed a buffer around the extracted road data and evaluated the portion of reference roads inside the buffer to further analyze the results. This is called the *completeness*. Together, *correctness* and *completeness* constitute a comprehensive metric known as *quality*. Another evaluation metric known as *redundancy* determines the degree of overlap in the road extraction methods.

[4]https://gaia.didichuxing.com.

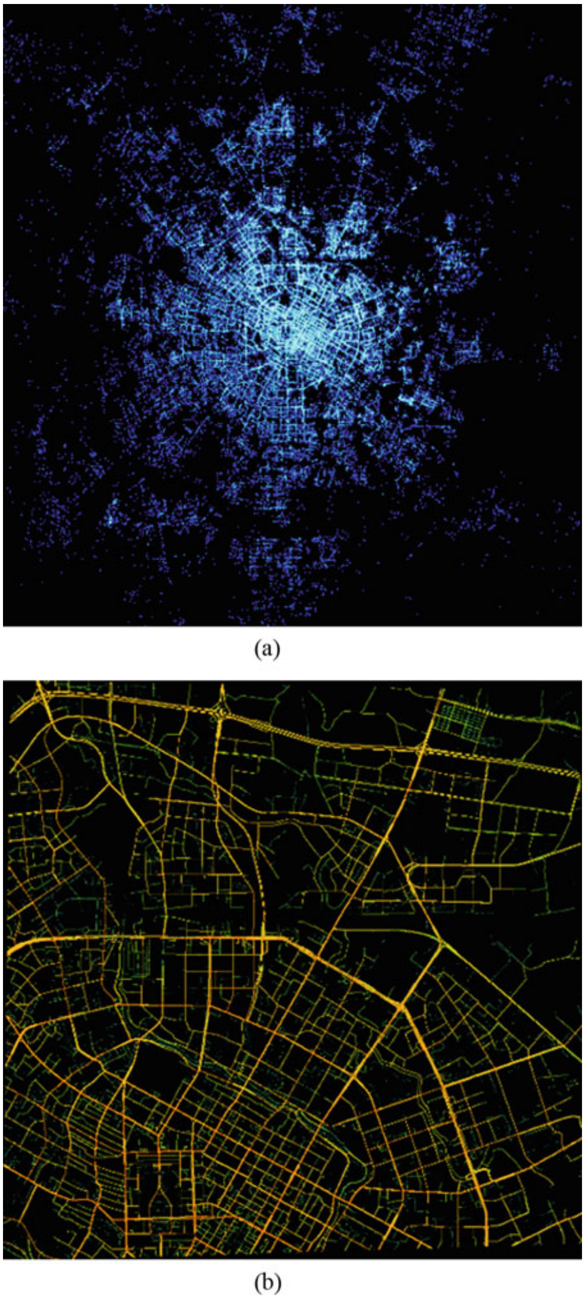

Fig. 11.7 A map visualization of the case-study DiDi GPS data: (**a**) over 1 million trip origins and destinations per day; (**b**) part of the extracted road geometry density from 181,172 trip trajectories

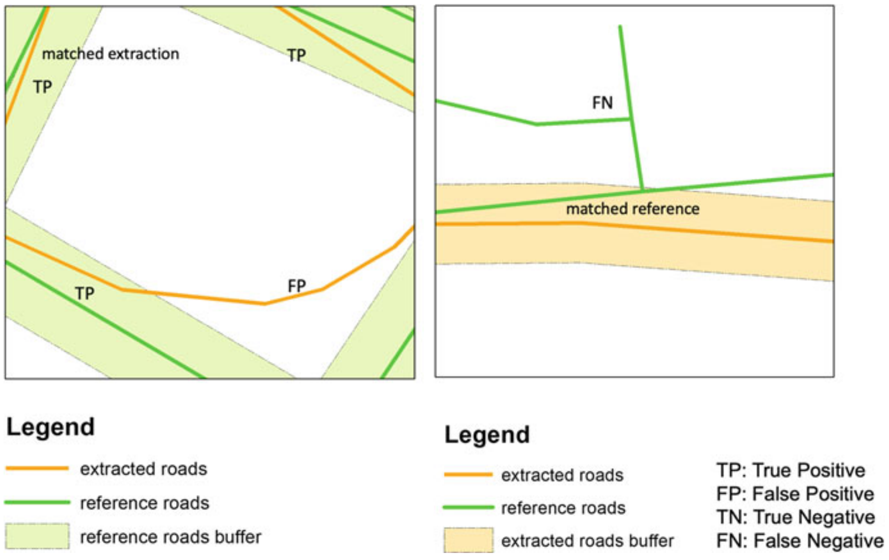

Fig. 11.8 The matching principle between extracted roads and reference roads using buffers

- Completeness

  The completeness constitutes the percentage of reference roads delineated by the extracted road network. That is, the percentage of the reference road data located within the buffer encircling the extracted roads. The closer completeness is to 1, the better the performance of the algorithm is. Let L_{mr} be the length of matched reference roads and L_r be the length of all reference roads.

$$Completeness = \frac{L_{mr}}{L_r}$$

- Correctness

  The correctness signifies the percentage of extracted road data that actually corresponds to the reference road. In other words, it is the percentage of the extracted road network that falls within the buffer surrounding the reference roads. The closer correctness is to 1, the better. Let L_{me} be the length of matched extraction roads and L_e be the length of all extracted roads.

$$Correctness = \frac{L_{me}}{L_e}$$

- Quality

  The quality summarizes the road extraction method by factoring both completeness and correctness into one value. The closer quality is to 1, the better. Let L_{ur} be the length of unmatched reference roads.

$$Quality = \frac{L_{me}}{L_e + L_{ur}}$$

- Redundancy

  The redundancy measures the percentage of the matched extraction road network which overlaps itself. The closer redundancy is to 0, the better.

$$Redundancy = \frac{L_{me} - L_{mr}}{L_{me}}$$

11.4.2.2 Results

In order to evaluate the extracted road network quality, we downloaded the OpenStreetMap (OSM) data as the reference road network layer. Moreover, we also compared our approach to the baseline approach that is directly based on the KDE surface of raw trajectory GPS points and the vectorization operation. The one-day outcomes of the correctness, completeness, quality, and redundancy using our proposed method with different buffer radiuses (10 m, 20 m, and 40 m) are shown in Table 11.1 and the results using the baseline approach are shown in Table 11.2. One can conclude that both methods achieved good performance in extracting roads which fall within the vicinity of the reference roads. Overall, a larger buffer distance gets better results but it may encounter the problem of large redundancy for nearby roads. However, the methods scored poorly in completeness as an inadequate amount of the reference roads were situated closely to the extracted roads. As seen in Fig. 11.9, many of the reference roads were simply not extracted at all which made these unsuccessfully extracted reference roads isolated from the extracted network. With no extracted roads close by, the completeness value falls. As a whole, the quality metric scored poorly due to the shortcoming of many reference road segments not being extracted at all. We would expect the completeness score

Table 11.1 The extracted road quality evaluation results using our proposed approach

Buffer distance	Completeness	Correctness	Quality	Redundancy
40 m	0.603534	0.995467	0.430439	1.00737
20 m	0.502849	0.983940	0.371855	0.692080
10 m	0.330325	0.854347	0.265552	0.280146

Table 11.2 The extracted road quality evaluation results using the baseline approach

Buffer distance	Completeness	Correctness	Quality	Redundancy
40 m	0.597233	0.986477	0.429040	0.952992
20 m	0.469571	0.925763	0.341474	0.636230
10 m	0.302925	0.768156	0.236453	0.272123

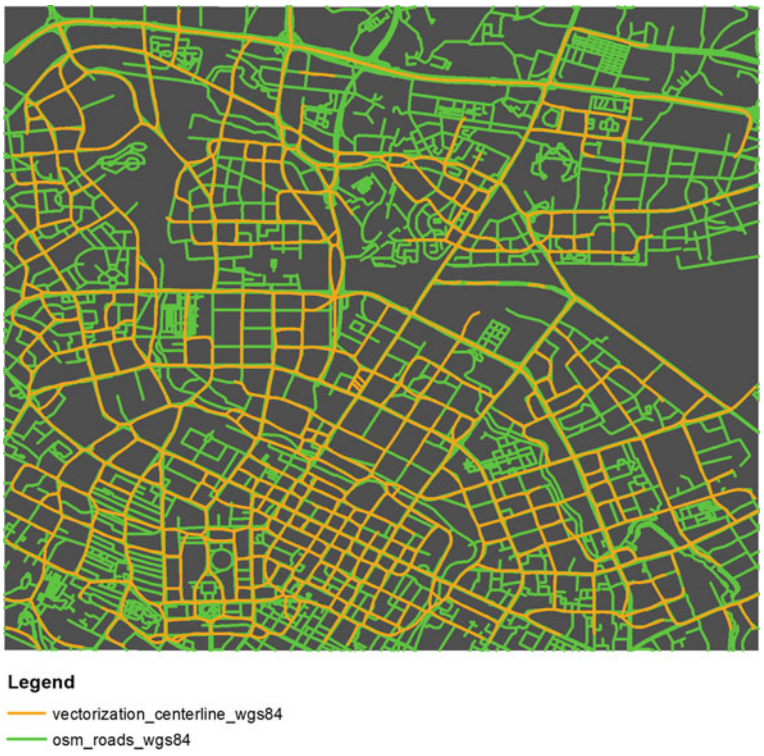

Fig. 11.9 A visual comparison of the extracted road network and the OSM road reference layer

could increase with larger spatial coverage of the datasets especially in those (non-major) tributary roads. The redundancy value scored the worst since the length of the matched extraction was much larger than the length of matched reference. Again, the failure to extract many reference road segments resulted in a low evaluation metric. However, Tables 11.1 and 11.2 showed that our approach scored best in completeness, correctness, and quality whereas the baseline approach scored best in redundancy only. Our approach outperformed the baseline approach with a small margin as it compressed clusters and could limit noise points outside the road network.

In addition, in order to determine the connectivity of the extracted road network, we applied the shortest-path based approach using the average path length similarity (APLS) metric to evaluate the extracted road network quality. The average path length was calculated for both the proposal layer (the extracted roads) and the ground-truth reference layer (the downloaded OSM road data) using the Dijkstra's shortest path algorithm (Dijkstra 1959). Then, we examined the similarity between the average path lengths to formulate an overall score. The code to run such an analysis was made available by the CosmiQ Works.[5] This measurement was performed on both approaches.

To perform the measurement, the road layers were converted to graphs, on which nodes are placed at intersections, endpoints, and midpoints. The shortest path was then calculated from each node to each other node for each graph. The differences in path length were used to calculate a metric. In order to quantify these differences in distance between the proposal and ground truth graphs, the APLS metric was computed, which sums the differences in optimal path lengths between nodes in the ground truth graph and the proposal graph. Missing paths are given the maximum proportional distance of 1.0.

Due to the nature of this metric, any missing nodes with high centrality will be penalized much more heavily than those with low centrality as high betweenness centrality roads account for larger traffic flow (Gao et al. 2013). It is important to consider how the nodes are generated and which nodes are important to the results when considering larger graphs. Accordingly, it may be necessary to exclude or alter the generation of nodes if calculating every possible path becomes infeasible—which was the case for our road extraction result.

Figure 11.10 shows the proposal graph for our road centerline extraction with nodes for intersections in sky-blue, as well as a buffer for the visible ground truth graph in yellow.

The graph was then run through multiple iterations of the APLS measurement code in order to determine a semi-optimal distance between the generated midpoints in order to approximate the best result. As shown in Table 11.3, it was determined that a midpoint distance of 300 meters gave the best suitable result with the highest APLS score.

A histogram (in Fig. 11.11) of the results was also created for the path differences when comparing the ground truth graph to the proposal graph, as well as when comparing the path differences from the proposal graph to the ground truth graph. The connectivity results were not good given the aforementioned low coverage completeness of the extracted road networks.

The same process was repeated to calculate the APLS metric for the baseline approach. Also, a distance of 300 meters between midpoints gave the best result for the total score (as shown in Table 11.4). However, the total APLS score for the baseline approach was lower than that of our proposed centerline road extraction approach, concluding that our approach provided slightly better connectivity in its resulting road network.

[5]https://github.com/CosmiQ/apls.

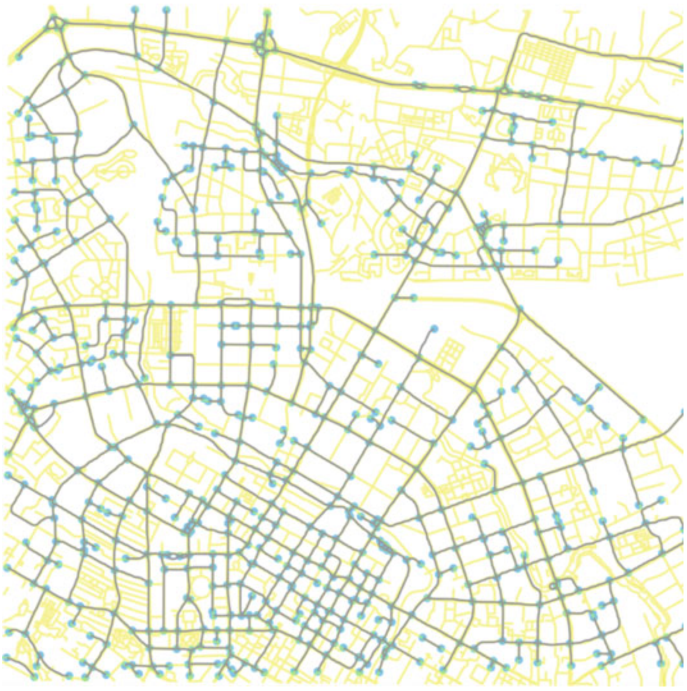

Fig. 11.10 The extracted road graph with ground-truth reference graph (yellow)

Table 11.3 Connectivity evaluation results with different distance settings using our proposed approach (The bold number highlights the best score)

Distance between midpoints (in meters)	Ground truth nodes snapped to proposal graph	Proposal nodes snapped onto ground truth graph	Total score
100	0.032315	0.275312	0.057841
150	0.032516	0.271229	0.058071
200	0.033787	0.287925	0.060478
250	0.034572	0.280139	0.061549
300	0.036189	0.281458	**0.064132**
400	0.034554	0.276131	0.061422
450	0.033946	0.277987	0.060503
500	0.034808	0.271562	0.061707
550	0.034434	0.265163	0.060953
600	0.034333	0.269076	0.060897
700	0.034470	0.259051	0.060844
1000	0.029633	0.243184	0.052829

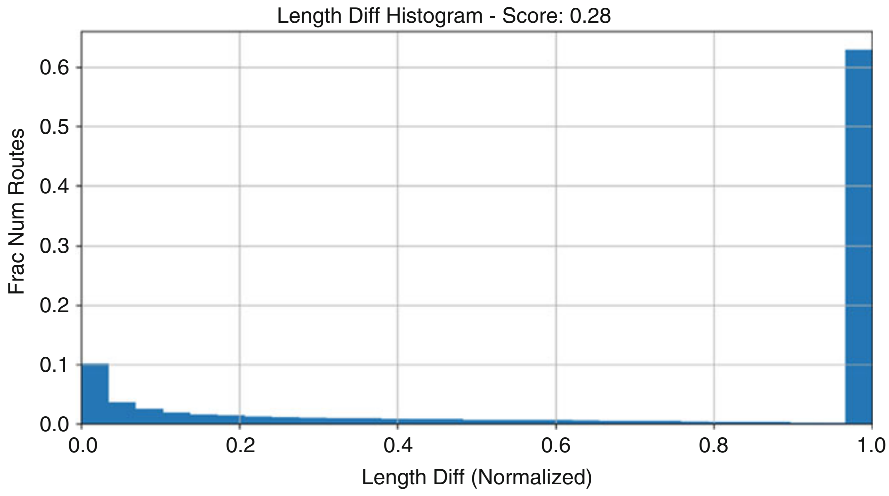

Fig. 11.11 A histogram showing the path differences from proposal (our approach) to ground truth road network

Table 11.4 Connectivity evaluation results with different distances settings using the baseline approach (The bold number highlights the best score)

Distance between midpoints (in meters)	Ground truth nodes snapped onto proposal graph	Proposal nodes snapped onto ground truth graph	Total score
250	0.030692	0.286749	0.055450
300	0.032063	0.289250	**0.057727**
328	0.031514	0.283418	0.056722
350	0.031102	0.293708	0.056248
400	0.031444	0.284132	0.056621
450	0.030946	0.285617	0.055842
500	0.031358	0.276147	0.056321
1000	0.029496	0.242499	0.052595
2000	0.029709	0.238963	0.052849

11.5 Conclusion and Future Work

In this chapter, we present a data-driven approach to extracting road centerline geometry information using large-scale GPS trajectory data. The introduced road extraction framework utilizes trajectory compression (Douglas-Peucker algorithm), an anisotropic density-based trajectory clustering (ADCN algorithm), a kernel density estimation, and vectorization approach. Compared with remote sensing-based approach, the ride-hailing service GPS trajectory data has a higher spatiotemporal resolution but a smaller geographical coverage. A case study using the DiDi open trajectory dataset in Chengdu, China demonstrates the effectiveness of our proposed approach for extracting road networks. It performed well with regard to the correctness of extracted road networks. However, the connectivity quality is bad due to the large incomplete coverage of the tributary roads. Future work needs to further improve the completeness. In addition, road networks include not only geometry information but also attributes (e.g., number of lanes, one-way restriction, speed limit). Such information may also be extracted from large-scale trajectory datasets with additional attributes (e.g., direction, speed), which require more attention in the road network generation pipeline.

Acknowledgments We thank the support of the DiDi Chuxing GAIA Open Dataset Initiative. Dr. Song Gao would like to thank the support for this research provided by the University of Wisconsin - Madison Office of the Vice Chancellor for Research and Graduate Education with funding from the Wisconsin Alumni Research Foundation (WARF).

References

Aronov B, Driemel A, Kreveld MV, Löffler M, Staals F (2016) Segmentation of trajectories on nonmonotone criteria. ACM Trans Algorithms (TALG) 12(2):26

Biagioni J, Eriksson J (2012, November) Map inference in the face of noise and disparity. In: Proceedings of the 20th international conference on advances in geographic information systems. ACM, pp 79–88

Cadez I, Gaffney S, Smyth P (2000, March) A general probabilistic framework for clustering individuals. In: Proc of ACM SIGKDD, vol 2000

Cao L, Krumm J (2009, November) From GPS traces to a routable road map. In: Proceedings of the 17th ACM SIGSPATIAL international conference on advances in geographic information systems. ACM, pp 3–12

Davies JJ, Beresford AR, Hopper A (2006) Scalable, distributed, real-time map generation. IEEE Pervasive Comput 5(4):47–54

Dijkstra EW (1959) A note on two problems in connexion with graphs. Numer Math 1(1):269–271

Douglas DH, Peucker TK (1973) Algorithms for the reduction of the number of points required to represent a digitized line or its caricature. Cartographica Int J Geogr Inf Geovis 10(2):112–122

Edelkamp S, Schrödl S (2003) Route planning and map inference with global positioning traces. In: Computer science in perspective. Springer, Berlin, Heidelberg, pp 128–151

Ester M, Kriegel HP, Sander J, Xu X (1996) A density-based algorithm for discovering clusters in large spatial databases with noise. KDD 96(34):226–231

Gaffney S, Smyth P (1999) Trajectory clustering with mixtures of regression models. KDD 99:63–72

Gao S, Janowicz K, Montello DR, Hu Y, Yang JA, McKenzie G, Ju Y, Gong L, Adams B, Yan B (2017) A data-synthesis-driven method for detecting and extracting vague cognitive regions. Int J Geogr Inf Sci 31(6):1245–1271

Gao S, Wang Y, Gao Y, Liu Y (2013) Understanding urban traffic-flow characteristics: a rethinking of betweenness centrality. Environment and Planning B: Planning and Design. 40(1):135–53

Goodchild MF (2007) Citizens as sensors: web 2.0 and the volunteering of geographic information. GeoFocus Revista Internacional de Ciencia y Tecnología de la Información Geográfica 7:8–10

Hancke GP, Hancke GP Jr (2013) The role of advanced sensing in smart cities. Sensors 13(1):393–425

Hu Y, Gao S, Janowicz K, Yu B, Li W, Prasad S (2015) Extracting and understanding urban areas of interest using geotagged photos. Comput Environ Urban Syst 54:240–254

Jiang Y, Li X, Li XJ et al (2012) Geometrical characteristics extraction and accuracy analysis of road network based on vehicle trajectory data[J]. J Geo-Inf Sci 14(2):165–170

Kolatch E (2001) Clustering algorithms for spatial databases: a survey, pp 1–22

Kuntzsch C, Sester M, Brenner C (2016) Generative models for road network reconstruction. Int J Geogr Inf Sci 30(5):1012–1039

Lang T (1969) Rules for the robot draughtsmen. Geogr Mag 42(1):50–51

Lee JG, Han J, Whang KY (2007, June) Trajectory clustering: a partition-and-group framework. In: Proceedings of the 2007 ACM SIGMOD international conference on Management of data. ACM, pp 593–604

Li Z, Lee JG, Li X, Han J (2010) Incremental clustering for trajectories. In: International conference on database systems for advanced applications. Springer, Berlin, Heidelberg, pp 32–46

Li J, Qin Q, Xie C, Zhao Y (2012) Integrated use of spatial and semantic relationships for extracting road networks from floating car data. Int J Appl Earth Obs Geoinf 19:238–247

Liu Y, Wang F, Xiao Y, Gao S (2012) Urban land uses and traffic 'source-sink areas': evidence from GPS-enabled taxi data in Shanghai. Landsc Urban Plan 106(1):73–87

Mai G, Janowicz K, Hu Y, Gao S (2018) ADCN: an anisotropic density-based clustering algorithm for discovering spatial point patterns with noise. Trans GIS 22(1):348–369

McMaster RB (1989) The integration of simplification and smoothing algorithms in line generalization. Cartographica Int J Geogr Inf Geovis 26(1):101–121

Opheim H (1982) Fast data reduction of a digitized curve. GeoProcessing 2:33–40

Reumann K, Witkam APM (1974) Optimizing curve segmentation in computer graphics. International Comp Sympo 1973. Amsterdam, pp 467–472

Shaw SL (2010) Geographic information systems for transportation: from a static past to a dynamic future. Ann GIS 16(3):129–140

Shi W, Cheung C (2006) Performance evaluation of line simplification algorithms for vector generalization. Cartogr J 43(1):27–44

Shi W, Shen S, Liu Y (2009, October) Automatic generation of road network map from massive GPS, vehicle trajectories. In: 2009 12th international IEEE conference on intelligent transportation systems. IEEE, pp 1–6

Silverman BW (1986) Density estimation for statistics and data analysis. Chapman & Hall, London

Stanojevic R, Abbar S, Thirumuruganathan S, Chawla S, Filali F, Aleimat A (2018, May) Robust road map inference through network alignment of trajectories. In: Proceedings of the 2018 SIAM International conference on data mining. Society for Industrial and Applied Mathematics, pp 135–143

Tang L, Ren C, Liu Z, Li Q (2017) A road map refinement method using Delaunay triangulation for big trace data. ISPRS Int J Geo Inf 6(2):45

Tao CV (2000) Mobile mapping technology for road network data acquisition. J Geospat Eng 2(2):1–14

Wang J, Rui X, Song X, Tan X, Wang C, Raghavan V (2015a) A novel approach for generating routable road maps from vehicle GPS traces. Int J Geogr Inf Sci 29(1):69–91

Wang S, Wang Y, Li Y (2015b, November) Efficient map reconstruction and augmentation via topological methods. In: Proceedings of the 23rd SIGSPATIAL international conference on advances in geographic information systems, pp 1–10

Wang H, Li Y, Liu G, Wen X, Qie X (2019) Accurate detection of road network anomaly by understanding Crowd's driving strategies from human mobility. ACM Trans Spatial Algorithms Sys (TSAS) 5(2):11

Wiedemann C, Heipke C, Mayer H, Jamet O (1998) Empirical evaluation of automatically extracted road axes. In: Empirical evaluation techniques in computer vision, vol 12. IEEE Computer Society Press, Los Alamitos, pp 172–187

Worrall S, Nebot E (2007) Automated process for generating digitised maps through GPS data compression. In: Australasian conference on robotics and automation, vol 6. ACRA, Brisbane

Wu C, Ayers PD, Anderson AB (2007) Validating a GIS-based multi-criteria method for potential road identification. J Terrramech 44(3):255–263

Wu J, Zhu Y, Ku T, Wang L (2013) Detecting road intersections from coarse-gained GPS traces based on clustering. JCP 8(11):2959–2965

Yuan NJ, Zheng Y, Xie X (2012) Segmentation of urban areas using road networks. MSR-TR-2012–65, Tech Rep

Zhao Y, Liu J, Chen R, Li J, Xie C, Niu W, Geng D, Qin Q (2011, July). A new method of road network updating based on floating car data. In: 2011 IEEE international geoscience and remote sensing symposium. IEEE, pp 1878–1881

Zheng Y (2015) Trajectory data mining: an overview. ACM Trans Intell Sys Technol (TIST) 6(3):29

Yuill R. S (1971) The standard deviational ellipse; an updated tool for spatial description. Geografiska Annaler: Series B, Human Geography. 53(1):28–39