An Improved Direction-preserving Trajectory Simplification Algorithm

Pengfei Hao[a], Chunlong Yao[b, *]

School of Information Science and Engineering, Dalian Polytechnic University, Guangzhou 510000, China

[a]hpf_7883108@163.com

**Corresponding author Email: [b]yaocl@dlpu.edu.cn*

Keywords: trajectory data, direction-preserving, simplification.

Abstract: There is a lot of valuable information in the trajectory data, but the sheer volume of data creates challenges for storing and analyzing data. Therefore, the simplification of the trajectory data is particularly important. The directional information of the trajectory contains a large amount of semantic information, and the trajectory contour can be well maintained based on the directional simplification. The algorithm in this paper, is based on the improvement of the classical direction-preserving simplification algorithm DPTS. The directed weighting graph is used to obtain the unique path according to the shortest path algorithm. The experimental evaluation shows that the average direction error is declined under the same compression ratio.

1. Introduction

The trajectory data is obtained by tracking and positioning the moving object. Nowadays, location-based service devices have been integrated into our lives, leading to the growing popularity of trajectory data [1]. Moreover, these trajectory data contain a large amount of information, which can be used for user behavior analysis [2], traffic analysis [3] [4], route recommendation [5] [6], social relationship analysis [7] [8] and so on.

Typically, these positioning devices periodically collect the position of the moving object. When the sampling period is short, a large amount of data will be generated. Consider Beijing with 67,000 taxis, suppose we collect trajectory data in every 2-3 seconds, the size of the trajectories generated by these taxis for just a single day is as high as 60TB [9]. This will give us a huge challenge in data storage, transmission and processing. In contrast, if the data sampling period is increased in order to reduce the amount of data, the loss of position information will result. In real life, if an object is running at a large speed, its position will vary greatly. For example, a taxi moving at 40 mph would have moved about 100 yards in 5s, whereas another taxi stuck at a traffic signal may not have moved at all[10]. Obviously, we need more frequent observations of the former than of the latter. Similarly, we need more observations to capture a taxi that makes a turn and fewer for one that continues straight. Therefore, standard practice is to oversample initially, and then to simplify by eliminating observations that add little information.

Therefore, compression of the trajectory data is particularly important. The trajectory compression algorithm can be divided into offline compression and online compression according to whether the known overall data is needed [11]. Offline compression is performed under the premise of giving complete data, and the compression ratio is better, but the trajectory performance is poor, which is suitable for the case where the compression performance is not high. Classical offline algorithms includes: DP algorithm [12], TD-TR algorithm [13], optimal algorithm [14] [15] [16], approximate optimal algorithm [17] [18], TS algorithm [19], ESTC -EDP algorithm [20] and so on. Online compression is a kind of timely compression, which can be compressed at any time, so its compression ratio will be reduced, but the compression performance is better, so it is used in the case of high compression performance. Classical online compression algorithms include: OPW-TR algorithm [13], construct security area algorithm [21], uniform sampling algorithm, SQUISH algorithm [22] [23], dead reckoning [24], DOTS algorithm [25] and so on.

The above algorithms all perform trajectory compression based on the position information of the captured trajectory, and the compression algorithm based on the direction-preserving is rare. In a large number of cases, direction-preserving is especially important. When an object moves from position p to p', we defy the direction of this movement to be the angle of an anticlockwise rotation from the positive x-axis to a vector from p to p'. The directions of all movements captured in a trajectory is called the direction information, and is used heavily, both directly and indirectly, in a wide range of applications on trajectory data. Such as: map matching, clustering, direction-based query, contour discovery and so on.

A typical direction-preserving trajectory simplification algorithm (DPTS) [10], by defined an angular threshold, constructs an undirected graph, and then through the breadth-first search BFS traversal, to obtain a shortest path, that is, as the final compressed trajectory. The algorithm ignores the path loss and the final compression trajectory may not be optimal. In this paper, the undirected graph is improved to a directed weighted graph, and the advantages of the proposed algorithm are shown by experimental comparison.

2. Related definition

Section Headings

Definition 1 Trajectory: A track of length n is a collection of a series of time-ordered anchor points, $T = \{P_1, P_2, ...,P_n\}$. Each of the positioning points Pi in T is composed of a triplet $<x_i, y_i, t_i>$, where x_i, y_i, represents the position coordinates of the moving object at time t_i.

Definition 1 Trajectory: A track of length n is a collection of a series of time-ordered anchor points, $T = \{P_1, P_2, ...,P_n\}$. Each of the positioning points Pi in T is composed of a triplet $<x_i, y_i, t_i>$, where x_i, y_i, represents the position coordinates of the moving object at time t_i.

Definition 2 Trajectory compression: Given a trajectory $T = \{P_1, P_2, ...,P_n\}$, trajectory compression is to find a set of chronologically ordered sets of locating points T' (a subset of T), $T' = \{P_{i1},P_{i2},...,P_{im}\}$, where $i_1 <... <i_m$, $i_1=1$, $i_m = n$.

Definition 3 Compression ratio: The compression ratio is given by the original trajectory $T = \{P_1, P_2, ...,P_n\}$, and the compressed trajectory $T' = \{P_{i1},P_{i2},...,P_{im}\}$. Compression ratio :

$$CR = n/m , n \geq m \tag{1}$$

Definition 4 Direction-based Error Measurement Ed [10] . Given a segment p_ip_{i+1} in T , the direction of p_ip_{i+1}, denoted by $\theta(p_ip_{i+1})$, is defined to be the angle of an anticlockwise rotation from the positive x-axis to a vector from pi to pi+1. Thus, each direction falls in $[0, 2\pi)$. The angular difference between two directions θ1 and θ2, denoted by $\triangle(\theta 1, \theta 2)$, is defined to be the minimum of the angle of the anticlockwise rotation from θ1 to θ2 and that from θ2 to θ1, i.e.,

$$\triangle(\theta 1, \theta 2) = \min\{|\theta 1 - \theta 2|, 2\pi - |\theta 1 - \theta 2|\} \tag{2}$$

Note that the angular difference between any two directions falls in [0, π].

Let $T' = \{P_{i1}, P_{i2}, ..., P_{im}\}$ be a simplification of *T* The simplification error of *T ′* under Ed, denoted by $\epsilon(T')$, is defined as follows. Given a segment p_ikp_{ik+1} in *T ′*, the simplification error of p_ikp_{ik+1}, denoted by $\epsilon(p_ikp_{ik+1})$, is defined to be the greatest angular difference between the direction of pskpsk+1 and the direction of a segment in T approximated by pskpsk+1. That is,

$$\epsilon\ (p_ikp_{ik+1}) = \text{maxik} \leq \text{h} < \text{ik} + 1\Delta(\theta(p_ikp_{ik+1}), \theta(\text{phph} + 1)) \tag{3}$$

Then, the simplification error of *T ′* under Ed is defined to be the greatest simplification error of a segment in *T ′*. That is,

$$\epsilon(T') = \text{max}1 \leq \text{k} < \text{m}\ \epsilon(p_ikp_{ik+1}) \tag{4}$$

In the following, when we write ϵ (pipj) ($0 \leq i < j \leq n$), we mean the simplification error of pipj when it is used to approximate the line segments between pi and pj in T . Let T be a trajectory and ϵ t be the error tolerance ($\epsilon_t < \pi$). Trajectory T ′ is said to be an ϵ t-simplification of T if T ′ is a simplification of T and $\epsilon\ (T') \leq \epsilon_t$.The DPTS problem is formalized as follows.

Definition 5 spatial error: Given a trajectory *T* and its compressed representation *T'*, the spatial error of *T'* with respect to a point P_i in *T* is define as the distance between $P_i\ (x_i, y_i, t_i)$ and its estimation $P'_i\ (x'_i, y'_i, t_i)$. If *T'* contains P_i, then $P'_i = P_i$. Otherwise, the closest point to P_i is defined as P'_i which is along the line between $\text{pred}_{T'}(P_i)$ and $\text{succ}_{T'}(P_i)$, where $\text{pred}_{T'}(P_i)$ and $\text{succ}_{T'}(P_i)$ denote P_i's closest predecessor and successor among the points in *T'* [22].

Definition 6 speed error: Given a trajectory *T* and its compressed representation *T'*, the speed error of *T'* with respect to a point $P_i(x_i, y_i, t_i)$ in *T* is define as the absolute difference value between *Speed* (P_i) and *AverageSpeed* (P_sP_e), where $P_s(x_s, y_s, t_s) = \text{pred}_{T'}(P_{i+1})$, $P_e(x_e, y_e, t_e) = \text{succ}_{T'}(P_i)$. P_i's speed and average speed of segment P_sP_e are defined as follow

$$Speed(P_i) = Distance(P_i, P_{i+1})/(t_{i+1} - t_i)\ ,\ P_i \neq P_n \tag{5}$$

$$AverageSpeed(P_sP_e) = Distance(P_s, P_e)/(t_e - t_s) \tag{6}$$

Definition 7 heading error: Given a trajectory *T* and its compressed representation *T'*, the heading error of *T'* with respect to a point P_i in *T* is define as heading change between *Heading* (P_i) and *Heading* (P_sP_e), where $P_s = \text{pred}_{T'}(P_{i+1})$, $P_e = \text{succ}_{T'}(P_i)$. P_i's heading, heading of segment P_sP_e and *HeadingChange* (h_1,h_2) are defined as follow

$$Heading(Pi) = \overrightarrow{PiPi+1}\ ,\ P_i \neq P_n \tag{7}$$

$$Heading(PsPe) = \overrightarrow{PsPe} \tag{8}$$

$$HeadingChange(h1, h2) = \begin{cases} 360° - |h1 - h2| & , |h1 - h2| > 180° \\ |h1 - h2| & , |h1 - h2| \leq 180° \end{cases} \tag{9}$$

3. Algorithm

3.1 Empowerment Ideas

The basic idea of the algorithm is consistent with the DPTS algorithm. It is main change that replace the DPTS algorithm undirected graph with a directed weighted graph, so that traversing graph obtains the only shortest path.

Have to be aware of, in order to ensure that the path with the smallest weight is also the least number of location points, there is no case where the shortest weight of the path is large or the maximum weight of the path is the smallest. The specific empowerment ideas are as follows:

We define both edge weights and direction weights, which are represented by W_E and W_H respectively. For the edge weights, we set a fixed value, $W_E=n*\pi$, where n is the number of original location points. When setting W_E, it is necessary to ensure that the shortest track path obtained is also the path with the fewest points after compression. There is no path with fewer points but greater weight. For the direction weights, set it to the sum of all angular changes compared to the threshold during the formation of this edge. For example, if there is an edge between P_1 and P_4 in the original trajectories, then $W_H=\beta(P_1, P_2)+\beta(P_2, P_3)+\beta(P_3, P_4)$, where $\beta(P_1, P_2)$ represents Angle change between $\theta(P_1, P_2)$ and $\theta(P_1, P_4)$, $\beta(P_2, P_3)$ represents Angle change between $\theta(P_2, P_3)$ and $\theta(P_1, P_4)$, $\beta(P_3, P_4)$ represents Angle change between $\theta(P_3, P_4)$ and $\theta(P_1, P_4)$.

Proof: If there is such a case, let the number of points be m after compression, then the total weight of these points is $W_{total\,1} = m * W_E + W_{H\,total\,1}$; assume that there are m-1 points but the weight is greater than $W_{total\,1}$, the weight of these points is: $W_{total\,2} = (m-1) * W_E + W_{H\,total\,2}$; Equation subtraction $W_{total\,1} - W_{total\,2} = W_E + (W_{H\,total\,1} - W_{H\,total\,2}) < 0$; if this assumption is true, when $W_E=n*\pi$ is defined, such a situation does not occur. Since all angle changes are within 180 degrees, when all the number of track points is multiplied by 180 degrees as the fixed edge weight, there is no case that $W_{total\,1}-W_{total\,2}<0$, which is contradictory to the assumption, and the assumption is not true.

3.2 Algorithm Description

The algorithm in this paper is mainly divided into the following three steps.

In step 1, it constructs a graph based on the threshold ϵt, represented by $G\epsilon t$(V, E，W). For each position p_i of T, where $1 \le i \le n$, it creates a vertex of pi in v. For the position i <j of each pair (pi, pj), if ϵ(pipj) $\le \epsilon t$, it creates an edge (pi, pj) in E and assigns a specific weight to the edge. In step 2, it finds the shortest path from p1 to pn in $G\epsilon t$ by the shortest path algorithm (ie.Dijkstra). In step 3, the solution resulting from the shortest path, this shortest path corresponds to the smallest track point in all locations in the T simplification. Therefore, if the shortest path is sorted ("p_{s1}-p_{s2} -...- p_{sm}"), the result T' is returned as (p_{s1}, p_{s2}, ..., p_{sm}).

Pseudo code description:

```
Input:T={p1,p2,p3….pn}    // Original track
      ϵt                  // Angle threshold
Output:T’={pi1,pi2,…pim} // Compressed track，pi1=p1,pim=pn,m<n;
1)define G（E,V,W）       // Defining a directed weighting graph
2)while true do
3)   for each pi in T and for each pj in T where i<j do
4)   define edgeWheight=n*π  // Defining edge rights
5)   define directionWheight=0  // Define direction rights
6)      if ϵ（pipk）≤ϵt and for each pk between pi and pj
```

```
7)    directionWheight= ∑ϵ（pipk）   i<k<=j // Directional right is the sum of all angle changes
8)    W= directionWheight+ edgeWheight  // Calculate total weight
9)get prenode[] on the basis of shortestPath algorithm// Get the array of precursor nodes
10)return the trajectory correspoinding to the shortest path from prenode[]
```

In the first step of the composition weighting process, the construction graph $G\epsilon t$ will traverse and check all (p_i, p_j), where $1 < i < j < n$, whether the threshold requirement $\epsilon(p_i p_j) \le \epsilon_t$ is satisfied, so the time complexity is $O(n)*O(n^2) = O(n^3)$. The second step is to traverse the graph with the shortest path search, so the time complexity is $O(n^2)$. In the third step, the shortest path time complexity is $O(n)$. Therefore, the time complexity of the algorithm in the DPTS algorithm is consistent.

4. Conclusion

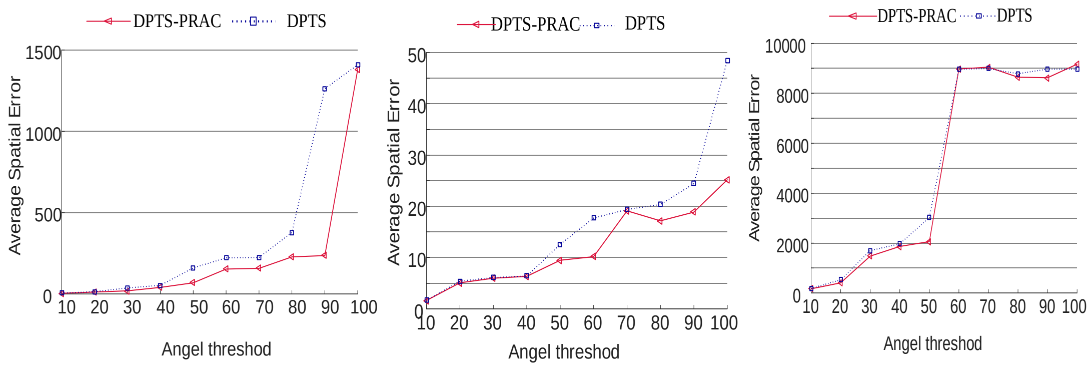
In order to verify the performance of the algorithm, the proposed algorithm and DPTS algorithm are implemented in java language. And Geolife dataset [26, 27] was used for algorithms evaluating. This dataset can support transportation mode learning, 73 users have labeled their trajectories with transportation mode, such as driving, taking a bus, riding a bike and walking. In our evaluations, three labeled trajectories were used. Trajectory one is a multi-modal trajectory, it contains three transportation mode (walk, bus, train), 5911 positioning points, and a total duration of 3 hours 49 minutes (from 2008-06-18,12:10:33 to 2008-06-18,15:59:59). Trajectory two is a bus track, it contains 2045 positioning points. Trajectory three is a taxi track in motorway, it contains 2167 positioning points

The algorithm in this paper is mainly based on the improvement of DPTS, so this evaluation is only for the direction preservation algorithm DPTS. Since the algorithm does not set the distance threshold, the error of the distance is too large. Various error metrics include average spatial error, average speed error and average heading error were used in our evaluation. Given a trajectory $T = \{P_1, P_2, P_3, ..., P_{n-1}, P_n\}$ and its compressed representation $T' = \{P_{i1}, P_{i2}, P_{i3}, \dots, P_{im-1}, P_{im}\}$, these error metrics are defined as follows:

$$AverageSpatialError(T,T') = \frac{1}{n}\sum_{i=1}^{n} SpatialError(T,T',P_i) \qquad (12)$$

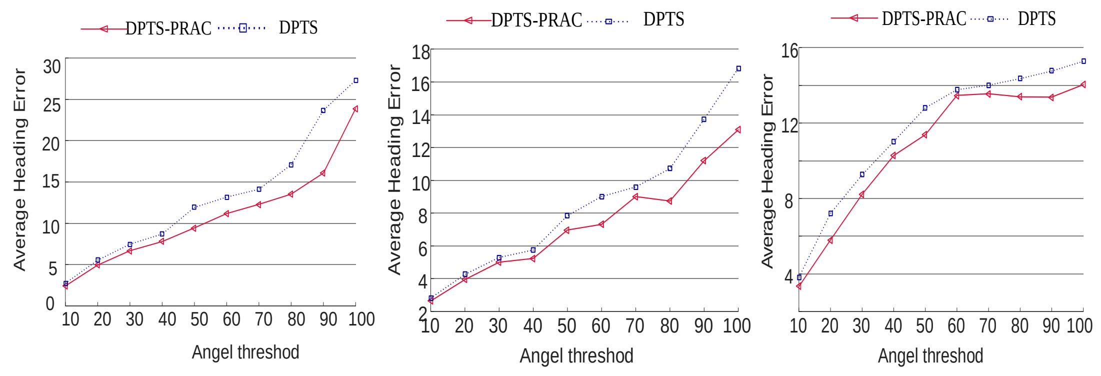
$$AverageSpeedError(T,T') = \frac{1}{n}\sum_{i=1}^{n} SpeedError(T,T',P_i) \qquad (13)$$

$$AverageHeadingError(T,T') = \frac{1}{n}\sum_{i=1}^{n} HeadingError(T,T',P_i) \qquad (14)$$



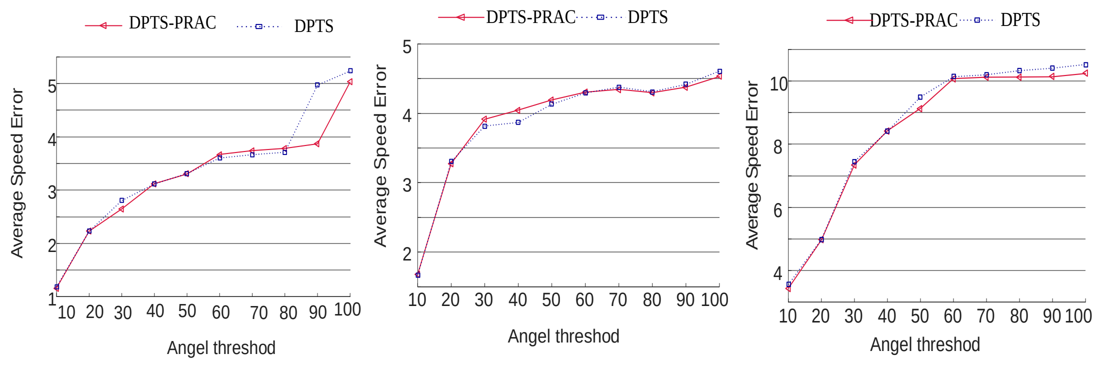
Trajectory One (multi-modal)   Trajectory two (bus)   Trajectory three (taxi)

Figure 1 Comparison of average spatial errors under various trajectories



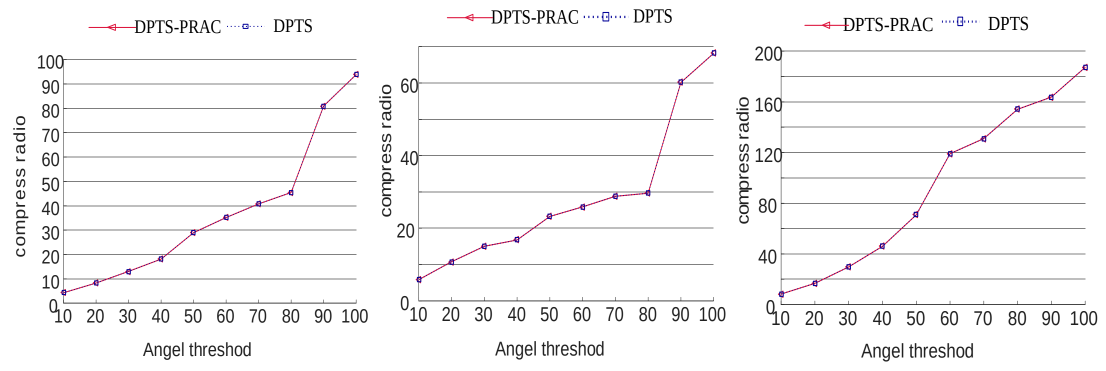
Trajectory One (multi-modal) Trajectory two (bus) Trajectory three (taxi)

Figure 2 Comparison of average direction errors in each trajectory mode



Trajectory One (multi-modal) Trajectory two (bus) Trajectory three (taxi)

Figure 3 Comparison of average speed errors in various trajectory modes



Trajectory One (multi-modal) Trajectory two (bus) Trajectory three (taxi)

Figure 4 Comparison of compression ratios in various trajectory modes

The above experimental results show that the evaluation error of the algorithm is compared with the DPTS algorithm, in which the red line represents the algorithm and the blue line represents the DPTS. The average distance error in Figure 1 is either fall or flat in each trajectory pattern. In Figure 2, the average direction error is significantly reduced, and the advantages of the algorithm in this paper can be seen. The average speed error in Figure 3 is also basically a flat effect. Figure 4 shows that the compression ratios of the two algorithms are consistent with the theory, the algorithm does not change the compression ratio.

5. Conclusion

In this paper, we consider that the classical direction preserving simplification algorithm DPTS has unit loss, and the obtained path is not optimal. Therefore, the DPTS algorithm is improved, the undirected graph is replaced to the directed graph, and the effective weight is given ,to ensure that the shortest path traversed by the shortest path algorithm is unique and let it be the compressed trajectory. It is known from the experimental evaluation that the algorithm reduces the directional error while maintaining the same compression ratio. However, the algorithm in this paper is insufficient, ignore the situation that the angle change between the track points is greater than the threshold, and the distance threshold is not considered, so the following will be studied.

Acknowledgement

In this paper, Corresponding Author is Chunlong Yao.

References

[1] Long C, Wong C W, Jagadish H V. Trajectory simplification: on minimizing the direction-based error [M]. VLDB Endowment, 2014.
[2] Liu W, Zheng Y, Chawla S, et al. Discovering spatio-temporal causal interactions in traffic data streams[C]// ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. ACM, 2011:1010-1018.
[3] Luo W, Tan H, Chen L, et al. Finding time period-based most frequent path in big trajectory data[C]// ACM SIGMOD International Conference on Management of Data. ACM, 2013:713-724.
[4] Wei L Y, Zheng Y, Peng W C. Constructing popular routes from uncertain trajectories[C]// ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. ACM, 2012:195-203.
[5] Pham H, Shahabi C, Liu Y. EBM: an entropy-based model to infer social strength from spatiotemporal data[C]// ACM SIGMOD International Conference on Management of Data. ACM, 2013:265-276.
[6] Xiao X, Zheng Y, Luo Q, et al. Inferring social ties between users with human location history [J]. Journal of Ambient Intelligence & Humanized Computing, 2014, 5(1):3-19.
[7] Zheng Y, Yuan N J, Zheng K, et al. On discovery of gathering patterns from trajectories[C]// IEEE International Conference on Data Engineering. IEEE Computer Society, 2013:242-253.
[8] Tang L A, Zheng Y, Yuan J, et al. On Discovery of Traveling Companions from Streaming Trajectories[C]// IEEE, International Conference on Data Engineering. IEEE Computer Society, 2012:186-197.
[9] Jing Yuan. "Querying, Mining with Applications on Large-Scale Trajectory Data"[D].University of Science and Technology of China, 2012.
[10] Cheng L, Wong C W, Jagadish H V. Direction-preserving trajectory simplification [M]. VLDB Endowment, 2013.
[11] JIANG Junwen, WANG Xiaoling. Review on trajectory data compression [J].Journal of East China Normal University: Natural Science, 2015(5):61-76.(in Chinese).
[12] Poiker T, Douglas D H. Reflection Essay: Algorithms for the Reduction of the Number of Points Required to Represent a Digitized Line or its Caricature [M]// Classics in Cartography. Wiley-Blackwell, 1973:112-122.
[13] Meratnia N, By R A D. Spatiotemporal Compression Techniques for Moving Point Objects[C]// Advances in Database Technology - EDBT 2004, International Conference on Extending Database Technology, Heraklion, Crete, Greece, March 14-18, 2004, Proceedings. DBLP, 2004:765-782.
[14] Perez J C, Vidal E. Optimum polygonal approximation of digitized curves [J]. Pattern Recognition Letters, 1994, 15(8):743-750.
[15] Salotti M. Improvement of Perez and Vidal Algorithm for the Decomposition of Digitized Curves into Line Segments[C]// Pattern Recognition, 2000. Proceedings. 15th International Conference on. IEEE, 2000:878-882 vol.2.
[16] Salotti M. An efficient algorithm for the optimal polygonal approximation of digitized curves [J]. Pattern Recognition Letters, 2001, 22(2):215-221.
[17] Kolesnikov A, Fränti P. A fast near-optimal min-# polygonal approximation of digitized curves [J]. International Journal of Pediatric Otorhinolaryngology, 1997, 39(3):248.
[18] Kolesnikov A, Nti P. Reduced-search dynamic programming for approximation of polygonal curves [J]. Pattern Recognition Letters, 2003, 24(14):2243-2254.
[19] Chen Y, Jiang K, Zheng Y, et al. Trajectory simplification method for location-based social networking services[C]//

International Workshop on Location Based Social Networks, Lbsn 2009, November 3, 2009, Seattle, Washington, Usa, Proceedings. DBLP, 2009:33-40.
[20] Qian H, Lu Y. Simplifying GPS Trajectory Data with Enhanced Spatial-Temporal Constraints [J]. International Journal of Geo-Information, 2017, 6(11):329.
[21] Potamias M, Patroumpas K, Sellis T. Sampling Trajectory Streams with Spatiotemporal Criteria[C]// International Conference on Scientific and Statistical Database Management. IEEE Computer Society, 2006:275-284.
[22] Muckell J, Hwang J H, Lawson C T, et al. Compression of trajectory data: a comprehensive evaluation and new approach[J]. Geoinformatica, 2014, 18(3):435-460.
[23] Muckell J, Hwang J H, Patil V, et al. SQUISH: an online approach for GPS trajectory compression[C]// International Conference on Computing for Geospatial Research & Applications. ACM, 2011:13.
[24] Trajcevski G, Cao H, Scheuermanny P, et al. On-line data reduction and the quality of history in moving objects databases[C]// ACM International Workshop on Data Engineering for Wireless and Mobile Access. ACM, 2006:19-26.
[25] Cao W, Li Y. DOTS: An online and near-optimal trajectory simplification algorithm [J]. Journal of Systems & Software, 2017, 126:34-44.
[26] Zheng Y, Li Q, Chen Y, et al. Understanding mobility based on GPS data[C]// International Conference on Ubiquitous Computing. ACM, 2008:312-321.
[27] Zheng Y, Xie X, Ma W Y. GeoLife: A Collaborative Social Networking Service among User, Location and Trajectory [J]. Bulletin of the Technical Committee on Data Engineering, 2010, 33(2):32-39.