Hindawi
Mobile Information Systems
Volume 2022, Article ID 5032375, 10 pages
https://doi.org/10.1155/2022/5032375

Research Article

Simplified Algorithm of Moving Object Trajectory Based on Interval Floating

Xuesong Chen, Zhiying Yang, and Yingjie Liu

College of Information Engineering, Shanghai Maritime University, Shanghai 201306, China

Correspondence should be addressed to Zhiying Yang; zyyang@shmtu.edu.cn

Received 30 October 2021; Accepted 23 July 2022; Published 8 November 2022

Academic Editor: Marco Anisetti

This paper proposes an online trajectory simplification algorithm based on interval floating. The accumulated angle deviation is used in the algorithm, and the bounded error theorem of interval floating is presented. First, the accumulated angle deviation starts from the nearest reserved point. Next, the sum of the angle deviations generated by the subsequent trajectory points is continuously calculated. When the simplified threshold is reached for the first time, it is will be judged whether the simplified threshold interval needs to be floated as well as the next reservation in the floating error interval. It is worth noting that the interval between two adjacent reserved points floats only once. The algorithm is tested on real trajectory data, and the experimental results show that the algorithm has an improved simplification rate with a certain simplification error.

1. Introduction

In the 21st century, with the advent of 4G and 5G mobile communication technologies, the higher popularity of the Internet has considerably promoted social progress. Besides, big data-related research becomes the hot spot globally. Also, mobile computing has been constantly evolving along with technologies such as mobile object databases and mobile communication network coverage. The location information of mobile terminal equipment keeps changing with time, and GPS and RFID are currently used as the main data collection equipment. Researchers employ the collected trajectory data to study the characteristics of moving objects and apply them in areas such as smart transportation and location-based services (LBS) [1, 2]. Location-based personalized service technology is a field that has been thriving in recent years. In terms of personalized recommendation, there are location recommendation [3], travel prediction [4], and user behavior analysis [5]. The basis of all LBS research is positioning, and a large amount of user location data needs to be obtained. However, many data in the stored massive trajectory data are of little research significance, which increases the difficulty of the subsequent scientific research analysis, causing huge capital and workload wasted in existing storage technology. In order to reduce the storage cost, more attention has been drawn to the research topic of trajectory simplification of moving objects. Many excellent trajectory simplification algorithms have been reported, but the development of trajectory simplification algorithms is still relatively slow, and each simplification algorithm has its own limitations. Many algorithms have their own specific application scenarios, and it is necessary to develop new simplification algorithms to broaden the spectrum of trajectory simplification so as to cope with various types of trajectory data sets in the future; in addition, most online algorithms use buffers since interval algorithms often have high time complexity. Therefore, it is necessary to explore new low-cost trajectory simplification algorithms.

So far, some related research has focused on the simplification algorithm of moving object trajectory. When the trajectory data simplification algorithm was first applied to computer graphics, the initially processed data only contained spatial information. Therefore, Euclidean distance (PED) was widely used. However, the data collected by the GPS system records time information in addition to space information. If we continue to use PED to study the simplified algorithm of moving object trajectories, we will inevitably lose time information. Therefore, the synchronized

Euclidean distance (SED) is gradually applied to the trajectory simplification algorithm. Many studies on trajectory simplification algorithms show that researchers mainly delve into two dimensions: offline simplification and online simplification.

The advantage of online trajectory simplification is that it supports real-time applications and can compress trajectory data while picking up new trajectory points as they are acquired. Offline trajectory simplification starts compressing only after all points are acquired from the input trajectory and is suitable for analysing historical trajectory data. Offline trajectory simplification usually has fewer errors compared to online trajectory simplification. However, in many applications, the trajectory data of the moving objects arrive in a stream, such as real-time AIS information received by shore-based systems. These applications include real-time trajectory tracking and position monitoring. Therefore, some online trajectory simplification methods have been proposed to handle this situation.

The earliest offline trajectory simplification algorithm based online segments is the Douglas–Peucker algorithm (DP algorithm) proposed in 1973 [6]. The DP algorithm needs to give the threshold of the simplified algorithm in advance. The threshold uses PED, and the algorithm is simplified according to the setting. The threshold recursively selects points greater than the simplified threshold and keeps the algorithm running until all points are less than the threshold set by the simplified algorithm. The TD-TR algorithm (also known as the top-down time ratio algorithm) was proposed by Meratnia and De [7]. Compared with the traditional DP algorithm, the TD-TR algorithm overcomes the shortcomings of the DP algorithm, for example, losing time information. TD-TR uses SED instead in the distance function of the DP algorithm because SED not only retains position but also time information compared with PED distance. Lin et al. [8] proposed the ATS algorithm. The ATS algorithm segments the original trajectory according to the important feature of the trajectory speed, calculates the SED threshold of the small trajectory, and finally uses the DP algorithm to simplify the final trajectory. Ke et al. [9] proposed the Angular algorithm, which uses the accumulated angle deviation to select or reject the trajectory point. By setting the accumulated angle threshold, the angle error before and after the trajectory simplification can be controlled. The time complexity is $O(n)$. Opening Window (OW) is a traditional online trajectory simplification algorithm. The core idea of OW is to initialize a window with a fixed size from the starting point of the trajectory and slide the window over the points on the original trajectory. This process is repeated until the last point of the original trajectory is processed [10]. Opening Window Time Ratio (OW-TR) [7] extended OPW using a synchronized Euclidean distance (SED) error instead of the spatial error. A number of successful online trajectory simplification algorithms have been proposed to simplify the road vehicle trajectories, terrain boundary line, trajectory data mining, and graphics display [11, 12]. Trajcevski et al. [13] proposed an online algorithm called Dead Reckoning, which uses the idea of estimation to estimate subsequent trajectory points. Muckell et al. [14] proposed the SQUISH algorithm, which will add new points directly to the buffer when there is still space in it. When the buffer is full, the algorithm deletes the point with the smallest error. The removal of small information points increases the importance of the left and right points of the discarded points in the area. The trajectory simplified by the algorithm has a reliable error guarantee, and the algorithm can be flexibly adjusted through the simplification rate and error. SQUISH-E [14] can achieve the smallest error under the condition of an artificially given simplification rate. The worst-case time complexity of the algorithm is $O(\text{nlogn}/\beta)$, β represents the artificially set simplification rate. Although it is an online trajectory simplification algorithm, in fact, under the condition of an artificially given simplification rate, the SQUISH algorithm can only perform offline trajectory simplification, and the algorithm needs to repeatedly traverse all points in the original trajectory, which is time-complex. In the most extreme case, the error between the original trajectory and the simplified trajectory will be very considerable.

The main innovations and contributions of this paper include as follows: (1) The bounded error theorem of interval floating is proposed, which can fully simplify the trajectory with a certain simplification error. (2) An online trajectory simplification algorithm is presented and implemented, which can simplify trajectory data online. (3) Various experiments were conducted from different data set sizes and different angle thresholds to evaluate the time performance and simplification rate performance of the algorithm. Through experimental comparison, in the face of large-scale trajectory data, the proposed algorithm has better time complexity and simplification rate.

2. Related Definitions and Lemmas of Algorithms

2.1. Definition 1: Angle Difference of Trajectory Segment [9]. Define the direction angle θ of the trajectory segment as follows: the directions of the trajectory segment $\overline{p_i p_j}$ and $\overline{p_m p_k}$ are denoted by $\theta(\overline{p_i p_j})$ and $\theta(\overline{p_m p_k})$, respectively, and the constraints are

$$\theta \in (0, 2\pi), (1 \le i < j \le n, 1 \le m < k \le n). \tag{1}$$

The angle difference formula is

$$\Delta\theta = \theta\left(\overline{p_i p_j}\right) - \theta\left(\overline{p_m p_k}\right). \tag{2}$$

For two given angles θ_1 and θ_2, the magnitude of their angle difference is

$$\min\left\{\left|\theta_1 - \theta_2\right|, 2\pi - \left|\theta_1 - \theta_2\right|\right\}. \tag{3}$$

The angle difference $\Delta(\theta_1, \theta_2)$ is divided into two cases, $|\theta_1 - \theta_2|$ and $2\pi - |\theta_1 - \theta_2|$ (see Figure 1). Easy to get by definition, the range of angle difference is $[0, \pi]$.

2.2. Definition 2: Angular Deviation of Trajectory Segment [9]. The angle of the moving object at point p_i and the angle change between the points before and after it is the angle

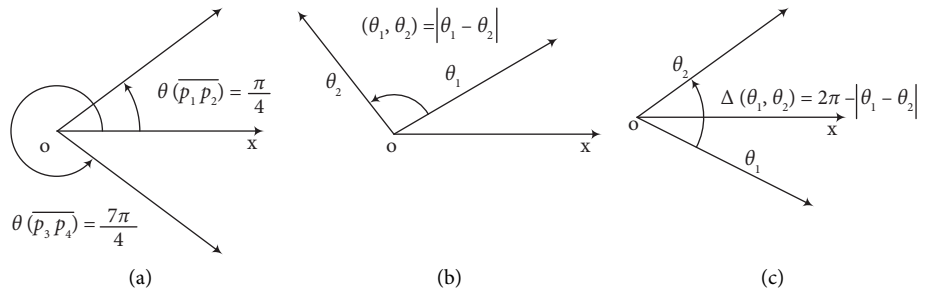

Figure 1: (a), (b), and (c), respectively, show how the angle difference of the trajectory segment is calculated in different situations.

deviation of the point, which is represented by the symbol $p_{i.\epsilon_d}$, and the details are as follows:

$$\Delta\theta = \theta\left(\overline{p_i p_{i+1}}\right) - \theta\left(\overline{p_{i-1} p_i}\right),$$
$$p_{i.\epsilon_d} = \begin{cases} \Delta\theta + 2\pi, & \Delta\theta \leq -\pi, \\ \Delta\theta, & -\pi < \Delta\theta \leq \pi, \\ \Delta\theta - 2\pi, & \Delta\theta > \pi. \end{cases} \tag{4}$$

$p_{i.\epsilon_d}$ has a positive value and a negative value (see Figure 2), $p_{2.\epsilon_d}$ and $p_{3.\epsilon_d}$ represent the angle deviation of point p_2 and point p_3, respectively.

2.3. Definition 3: Cumulative Angular Deviation [9]. The meaning of the cumulative angle deviation is the cumulative sum of the angle deflection of all points from the count point to the current point, which is defined as follows:

$$p_{i.\epsilon_d} = \begin{cases} 0, & i = s_k, \\ \sum_{m=s_k+1}^{i} p_{m.\epsilon_d}, & s_k < i < s_k + 1. \end{cases} \tag{5}$$

It should be noted that p_{s_k} represents the starting point of the simplified trajectory segment $\overline{p_{s_k} p_{s_{k+1}}}$, $p_{s_{k+1}}$ is the end point of the trajectory segment, and the subscript of the trajectory segment needs to satisfy the constraint: $s_k < i < s_{k+1}$.

2.4. Lemma 1: Bounded Error of Position Information [15]. The trajectory simplification algorithm that retains the direction information generally achieves the purpose of simplification by constraining the direction of the trajectory, which can ensure a certain direction error. In fact, the algorithm also retains the position information while retaining the direction information. Long et al. [15] proved that the direction-preserving algorithm can maintain the position characteristics. The following introduces the bounded error lemma of position information:

In the simplified algorithm that preserves direction information, if the simplified error is within ε, then the shortest vertical Euclidean distance d_{minped} and ε of the original trajectory and the simplified trajectory must satisfy the following relationship:

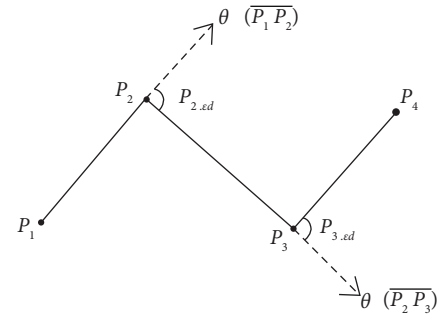

Figure 2: Schematic diagram of the calculation method of the angle deviation of each point.

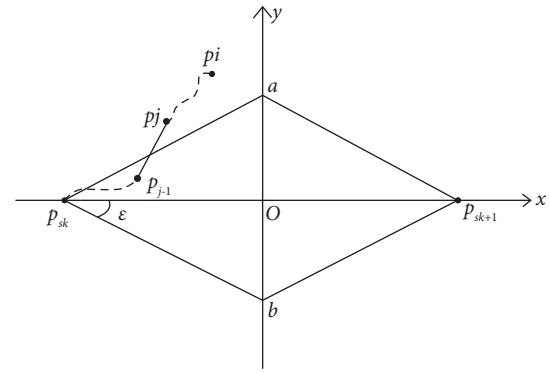

Figure 3: Rhombuses constructed from trajectory segments.

$$d_{\text{minpe d}} \leq \frac{1}{2} l_{\max} \tan \varepsilon. \tag{6}$$

Among them, $l_{\max}$ represents the maximum length of the original trajectory segment corresponding to the simplified trajectory segment in the simplified trajectory.

Proof. Let $\overline{p_{s_k} p_{s_{k+1}}}$ be a simplified trajectory segment selected arbitrarily. Among them, we stipulate that p_{s_k} is the starting point of the simplified trajectory section, and $p_{s_{k+1}}$ is the end point of the simplified trajectory section. Then, the simplification of the current trajectory section must satisfy the directional error within ε. For a simplified trajectory segment $\overline{p_{s_k} p_{s_{k+1}}}$ selected at random, we can construct a

rhombus according to $\overline{p_{s_k}p_{s_{k+1}}}$ (see Figure 3), and prove the theorem in the rhombus.

In the rhombus in the figure above, there is a relationship: $\Delta(\overline{p_{s_k}a}, \overline{p_{s_k}p_{s_{k+1}}}) = \varepsilon$. Assuming that the position of p_i is outside the rhombus, then there must be a situation where p_i is below $\overline{p_{s_k}a}$. Therefore, there is a difference between the unsimplified trajectory segment $\overline{p_{j-1}p_j}$ and the simplified trajectory segment $\overline{p_{s_k}a}$. There must be an intersection point between them. If the position of p_i is outside the rhombus, the direction error between the original trajectory segment $\overline{p_{j-1}p_j}$ and the simplified trajectory segment $\overline{p_{s_k}p_{s_{k+1}}}$ must be greater than ε. So far, the conclusion drawn is in contradiction with our original definition of the direction angle error. So it is deduced that p_i must exist within the rhombus constructed by the simplified trajectory segment $\overline{p_{s_k}p_{s_{k+1}}}$. So, it is concluded that even the point p_i farthest from the simplified trajectory segment $\overline{p_{s_k}p_{s_{k+1}}}$ must satisfy the following constraints:

$$d_{\text{minped}} \leq \frac{1}{2} l_{\max} \tan \varepsilon. \tag{7}$$

Among them, distance represents the vertical Euclidean distance between two elements in the calculation plane, which can be the distance from point to point or point to straight line. □

2.5. Lemma 2: Bounded Error of Direction [9]. The results and discussion may be presented separately, or in one combined section and may optionally be divided into headed subsections. Ke et al. [9] proposed the bounded error theorem of direction in the A algorithm and proved the theorem. The accumulated angle deviation used by the A algorithm is based on the bounded error theorem of direction. The following will prove that the accumulated angular deviation is a bounded error in direction.

Assuming that the artificially prescribed direction threshold $p_{i.\varepsilon_d}$ is ε_t, for each original trajectory point p_i, if the cumulative angle deviation $p_{i.\varepsilon_d}$ of point p_i is greater than the given ε_t, then p_i will be retained. The direction error between the final simplified trajectory and the original trajectory must obey the following constraints:

$$\varepsilon(T') < 2\varepsilon_t. \tag{8}$$

Proof. According to the relevant definition introduced above, for a random segment of trajectory segment $\overline{p_{s_k}p_{s_{k+1}}}$, the direction of the trajectory segment $\overline{p_ip_{i+1}}$ composed of connected trajectory points between p_{s_k} and $p_{s_{k+1}}$ can be derived. The specific expression is as follows:

$$\psi = \theta\left(\overline{p_{s_k}p_{s_{k+1}}}\right) + \sum_{m=s_k+1}^{i} p_{m.\varepsilon_d}. \tag{9}$$

Among them, the p_{s_k} is the starting point, $p_{s_{k+1}}$ is the end point, and p_{s_k+1}, p_{s_k+2}, p_{s_k+3}... etc. represent the trajectory points between p_{s_k} and $p_{s_{k+1}}$.

For the random trajectory segment $\overline{p_{s_k}p_{s_{k+1}}}$, the trajectory segment $\overline{p_ip_{i+1}}$ composed of all the adjacent trajectory points in $\overline{p_{s_k}p_{s_{k+1}}}$ satisfies the angle constraint $[\theta(\overline{p_{s_k}p_{s_{k+1}}}) - \varepsilon_t, \theta(\overline{p_{s_k}p_{s_{k+1}}}) + \varepsilon_t]$ (see Figure 4). In other words, for all the trajectory segments $\overline{p_ip_{i+1}}$, the direction of $\overline{p_ip_{i+1}}$ must be within the ε_t error in the direction of the first small trajectory segment $p_{s_k}p_{s_k+1}$.

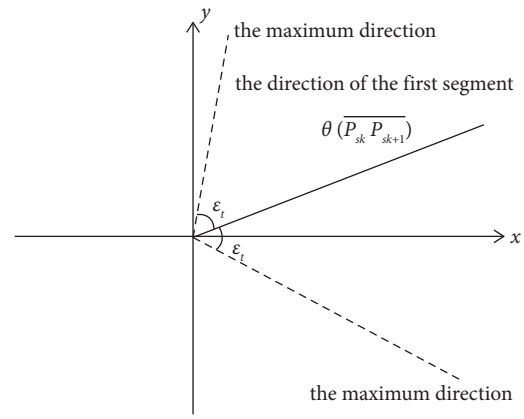

Figure 4: Directional error range chart.

The vector $\overrightarrow{p_{s_k}p_{s_{k+1}}}$ in the rhombus (see Figure 3) can be expressed as follows:

$$\overrightarrow{p_{s_k}p_{s_{k+1}}} = \overrightarrow{p_{s_k}a} + \overrightarrow{ap_{s_{k+1}}}. \tag{10}$$

According to (11), it can be seen that the vector sum of the geometric vectors formed by all two adjacent trajectory points in $\overrightarrow{p_{s_k}p_{s_{k+1}}}$ can be finally expressed as $\overrightarrow{p_{s_k}p_{s_{k+1}}}$, then the following constraints must be derived:

$$\theta\left(\overline{p_{s_k}p_{s_{k+1}}}\right) - \varepsilon_t < \theta\left(\overline{p_{s_k}p_{s_{k+1}}}\right) < \theta\left(\overline{p_{s_k}p_{s_{k+1}}}\right) + \varepsilon_t. \tag{11}$$

According to the above proofs, the direction angle error of any deleted track segment in the original trajectory is controlled within the range of ε from the original trajectory.

Therefore, when running the trajectory simplification algorithm, setting the algorithm's ε_t to half of ε can realize that the error between the simplified trajectory and the original trajectory is within ε. □

3. Trajectory Simplification Algorithm Based on Interval Floating

3.1. Theorem: Bounded Error of Interval Float. This paper proposes the theorem: Bounded Error of Interval Float.

If the current trajectory point $p_{i.\varepsilon_a}$ is in the interval $\varepsilon_t < p_{i.\varepsilon_a} < 2\varepsilon_t$, the algorithm performs a floating operation, that is, the accumulated angle deviation interval that needs to be discarded to simplify the trajectory floats from $[-\varepsilon_t, \varepsilon_t]$ to $[\min\_\varepsilon_a, 2\varepsilon_t + \min\_\varepsilon_a]$; if the current trajectory point satisfies the accumulated angle deviation is within the interval $-\varepsilon_t > p_{i.\varepsilon_a} > -2\varepsilon_t$, the accumulated angle deviation If the interval floats from $[-\varepsilon_t, \varepsilon_t]$ to $[-2\varepsilon_t + \max\_\varepsilon_a, \max\_\varepsilon_a]$, the simplified error $\varepsilon(T')$ must satisfy the constraints:

$$\varepsilon(T') < 2\varepsilon_t = \varepsilon. \tag{12}$$

Among them, $\min\_\varepsilon_a$ and $\max\_\varepsilon_a$ respectively represent the minimum and maximum accumulated angle deviation values from the last retained point to the current trajectory point.

Proof. For the simplified trajectory segment $\overline{p_{s_k}p_{s_{k+1}}}$ in Lemma 2, if the cumulative angle deviation of the end point $p_{s_{k+1}}$ is in the interval $\varepsilon_t < p_{s_{k+1}.\varepsilon_a} < 2\varepsilon_t$, there must be a small value Ω (see Figure 5), so that the current point $p_{s_{k+1}.\varepsilon_a}$ satisfies the following formula:

$$\Omega = 2\varepsilon_t - p_{s_{k+1}\bullet\varepsilon_a}, \tag{13}$$

and the minimum accumulated angle deviation $\min\_\varepsilon_a$ in the trajectory section $\overline{p_{s_k}p_{s_{k+1}}}$ must satisfy the following formula:

$$\min\_\varepsilon_a > -\varepsilon_t. \tag{14}$$

Therefore, point $p_{s_{k+1}}$ is not necessarily the first reserved point encountered from point p_{s_k}, and the interval of the error threshold needs to be floated up: from $[-\varepsilon_t, \varepsilon_t]$ to $[\min\_\varepsilon_a, 2\varepsilon_t + \min\_\varepsilon_a]$. After that, continue to search for the first reserved point encountered from point p_{s_k}, and the point $p_{ss_{k+1}}$ whose cumulative angle deviation is greater than $2\varepsilon_t + \min\_\varepsilon_t$ for the first time is reserved; in the same way, when $-\varepsilon_t > p_{s_{k+1}.\varepsilon_a} > -2\varepsilon_t$, the interval needs to be floated down to $[-2\varepsilon_t + \max\_\varepsilon_a, \max\_\varepsilon_a]$.

The direction of the trajectory segment $\overline{p_i p_{i+1}}$ existing between the first and last points of the simplified trajectory $\overline{p_{s_k}p_{ss_{k+1}}}$ that floats through the interval satisfies the formula:

$$\psi = \theta\left(\overline{p_{s_k}p_{ss_{k+1}}}\right) + \sum_{m=s_k+1}^{i} p_{m.\varepsilon_d}. \tag{15}$$

For the floating interval, the random trajectory segment $\overline{p_i p_{i+1}}$ satisfies the angle constraint in its trajectory segment $\overline{p_{s_k}p_{s_{k+1}}}$:

$$\left[\theta\left(\overline{p_{s_k}p_{ss_{k+1}}}\right) - \min\_\varepsilon_a, \theta\left(\overline{p_{s_k}p_{ss_{k+1}}}\right) + 2\varepsilon_t + \min\_\varepsilon_a\right]. \tag{16}$$

That is, the directions of the small trajectory sections $\overline{p_i p_{i+1}}$ between the simplified trajectory sections $\overline{p_{s_k}p_{s_{k+1}}}$ are all within the direction error interval of the floating interval $[\min\_\varepsilon_a, 2\varepsilon_t + \min\_\varepsilon_a]$ of the first small trajectory section $p_{s_k}p_{s_{k+1}}$ of the simplified trajectory section $\overline{p_{s_k}p_{s_{k+1}}}$.

After floating, the simplified trajectory segment $\overline{p_{s_k}p_{ss_{k+1}}}$ of the new end point will be obtained, and the rhombus is reconstructed according to $\overrightarrow{p_{s_k}p_{ss_{k+1}}}$ (see Figure 6). The vector B in the rhombus can be expressed as follows:

$$\overrightarrow{p_{s_k}p_{ss_{k+1}}} = \overrightarrow{p_{ss_k}a} + \overrightarrow{ap_{ss_{k+1}}}. \tag{17}$$

According to the abovementioned formula, the vector sum of the geometric vectors composed of all the two adjacent trajectory points between $\overline{p_{s_k}p_{ss_{k+1}}}$ can be finally expressed as $\overrightarrow{p_{s_k}p_{ss_{k+1}}}$, and the satisfied constraints must be derived:

$$\theta\left(\overline{p_{s_k}p_{ss_{k+1}}}\right) - \min\_\varepsilon_a < \theta\left(\overline{p_{s_k}p_{ss_{k+1}}}\right) < \theta\left(\overline{p_{s_k}p_{ss_{k+1}}}\right) + 2\varepsilon_t + \min\_\varepsilon_a. \tag{18}$$

In the same way, the descending provable interval must satisfy the constraint:

$$\theta\left(\overline{p_{s_k}p_{ss_{k+1}}}\right) + \max\_\varepsilon_a > \theta\left(\overline{p_{s_k}p_{ss_{k+1}}}\right) > \theta\left(\overline{p_{s_k}p_{ss_{k+1}}}\right) - 2\varepsilon_t + \max\_\varepsilon_a. \tag{19}$$

According to the above proofs, the error between a random small track segment deleted in the middle of the simplified track segment and the original track can be guaranteed to be $[\min\_\varepsilon_a, 2\varepsilon_t + \min\_\varepsilon_a]$ or $[-2\varepsilon_t + \max\_\varepsilon_a, \max\_\varepsilon_a]$.

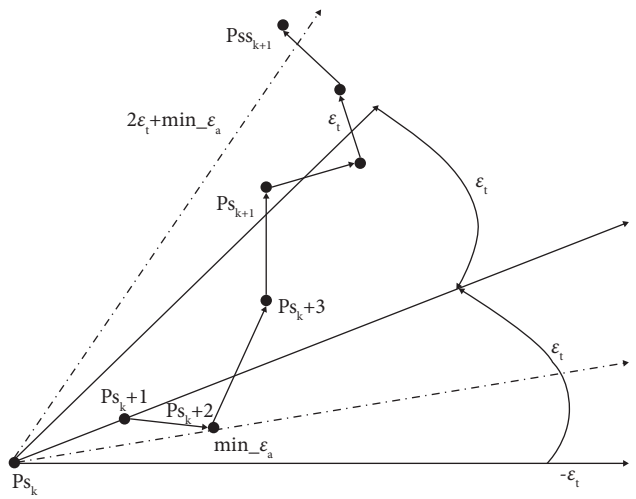

Figure 5: The direction range diagram after the interval is floated, the point $p_{ss_{k+1}}$ marked in the figure is the point.

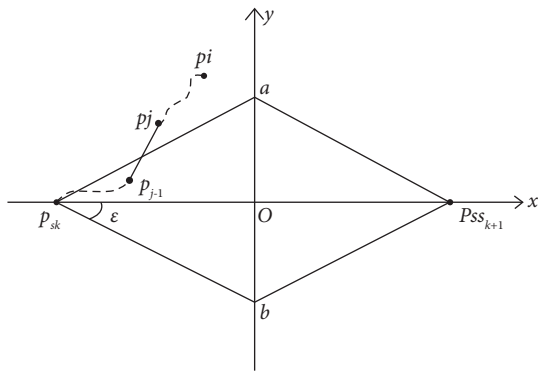

Figure 6: Rhombuses constructed from trajectory segments.

Therefore, when running the trajectory simplification algorithm, when the cumulative angle deviation of the current point is greater than the given threshold ε, refinding the first reserved point after an interval float must ensure the directional error between the simplified trajectory and the original trajectory in $2\varepsilon_t$. □

3.2. Description of Interval Floating Algorithm. According to related lemmas and theorems, this paper proposes an interval floating-based trajectory simplification algorithm for moving objects. As long as the simplified threshold is set, for each trajectory point collected, the search started from the most recently retained point each time, if there is a trajectory. If the point satisfies $p_{i.\varepsilon_a} > 2\varepsilon_t$, the point will be reserved, which can avoid the situation that the angle deviation of the reserved point is extremely small when its accumulated angle deviation reaches the threshold. For example, we set the direction error to 0.6, and the simplified threshold value when the algorithm is running 0.3, point p_3

will be retained instead of point p_2, but the contribution of point p_2 is much greater than that of p_3, so we cannot use $p_{i.\varepsilon_a} > \varepsilon_t$ as the judgment condition (see Figure 7). However, algorithm Angular uses $p_{i.\varepsilon_a} > \varepsilon_t$ as the judgment condition; if it does not exist, it will continue to find the point where the accumulated angle deviation is greater than the simplified threshold from the current point, perform an interval float from the current point, and continue to find the first point after the floating operation. If the point $p_{ss_{k+1}}$ accumulated angle deviation exceeds the floating interval, this point $p_{ss_{k+1}}$ must be retained. At this time, the direction interval is updated to the initial threshold interval again, that is, for each simplified trajectory, they pass the floating interval no more than once. If more than once, the bounded error between the simplified trajectory and the original trajectory cannot be guaranteed (see Algorithm 1).

4. Experiment and Discussion

In order to verify the trajectory simplification algorithm STIF based on interval floats, this paper uses the Geolife dataset in [16–18]. The GPS trajectory dataset was collected in (Microsoft Research Asia) Geolife project by 182 users in a period of over five years (from April 2007 to August 2012). A GPS trajectory of this dataset is represented by a sequence of time-stamped points, each of which contains the information of latitude, longitude, and altitude. Each file in the data selected for this experiment is larger than 30 KB. The experiment in this paper uses data sets of different sizes and different angle thresholds for experiments and analyzes the time performance and simplification rate performance of the algorithm through the experimental results.

The simplification rate of the algorithm is defined as follows:

$$\text{rate} = \frac{T'}{T} \times 100\%. \tag{20}$$

Among them, T' represents the simplified trajectory, and T represents the original trajectory. What we hope is that the algorithm can guarantee a low simplification rate within a certain error.

4.1. Performance Evaluation Based on Simplified Time. According to the experimental results (see Figure 8), when the data size is constant and the threshold is set very small, the simplification time of STIF algorithm is slightly larger than that of Angular. As the threshold increases, the simplification time of STIF algorithm tends to decrease and gradually approach Angular. From the perspective of the two algorithms, whether it is STIF or Angular, as the simplification threshold increases, the simplification time of the algorithm tends to decrease.

The size of the data set ranges from 1000 to 5000. According to the experimental results, for the two algorithms, the average simplification time (see Figures 9 and 10) of the simplification thresholds of different sizes decreases with the increase of the data set.

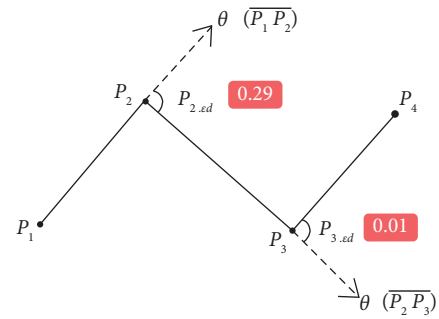

Figure 7: Cumulative angle deviation.

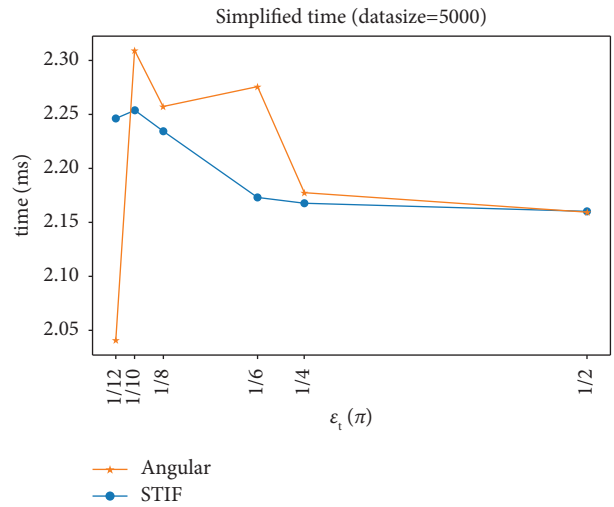

Figure 8: The performance of the time performance of the Angular algorithm and the STIF algorithm under 5000 trajectories as the direction error increases.

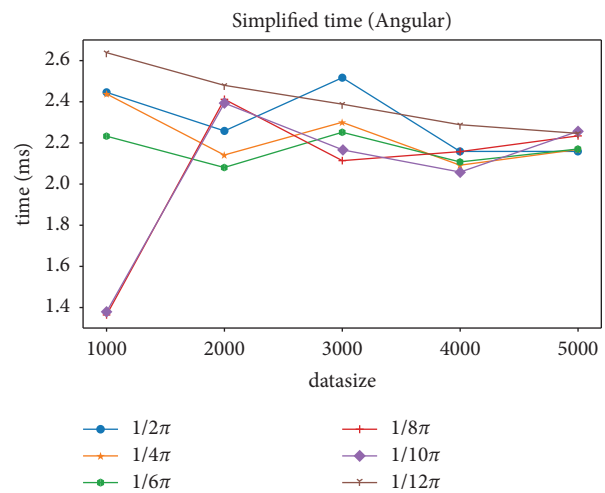

Figure 9: The performance of the Angular algorithm in different directions error with the increase of the trajectory data set.

4.2. Performance Evaluation Based on Simplification Rate. According to the experimental results (see Figure 11), it can be concluded that under 5000 trajectories, the average

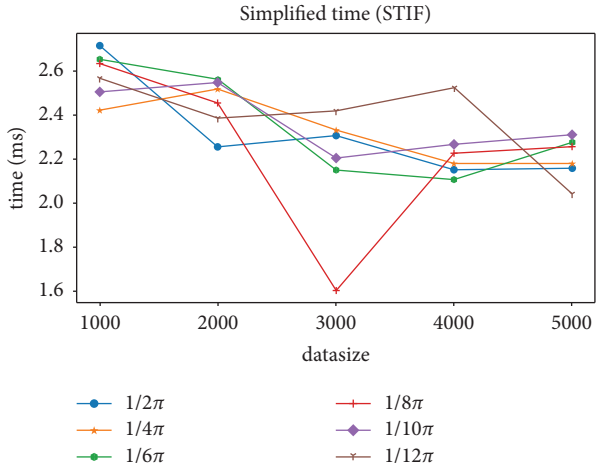

Figure 10: The performance of the STIF algorithm in different directions error with the increase of the trajectory data set.

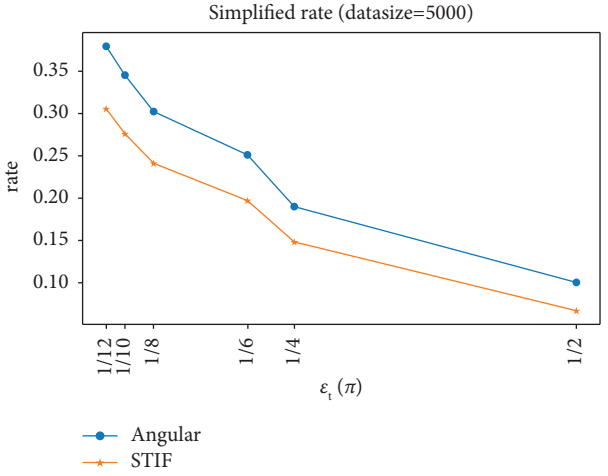

Figure 11: The performance of simplification rate performance of the Angular algorithm and the STIF algorithm under 5000 trajectories as the direction error increases.

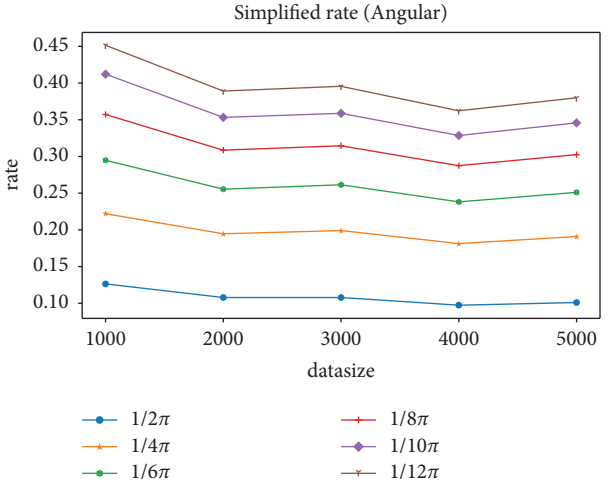

Figure 12: The performance of the Angular algorithm in different directions error with the increase of the trajectory data set.

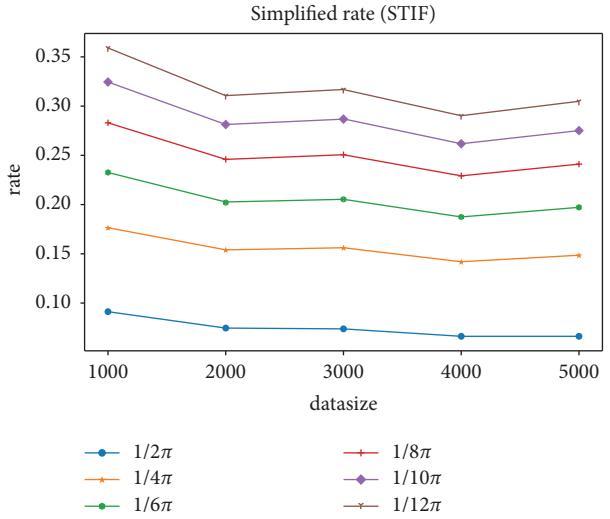

Figure 13: The performance of the STIF algorithm in different directions error with the increase of the trajectory data set.

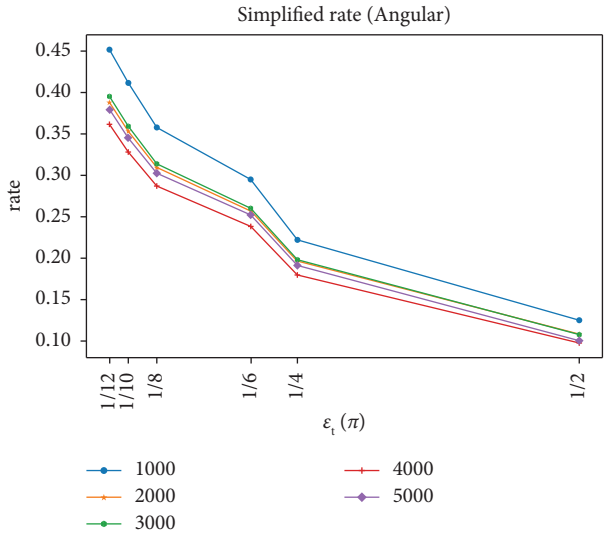

Figure 14: The change of the simplification rate of different data scales with the increase of the simplification threshold of the Angular algorithm.

simplification rate of the two algorithms tends to decrease with the increase of the simplification threshold, and the simplification rate of STIF algorithm is lower than Angular.

By observing the experimental results (see Figures 12 and 13), we can conclude that no matter what the simplification threshold is, the relationship between the simplification rate of the two algorithms and the data scale is that the larger the data scale is, the lower is the simplification rate.

It can be concluded from the experimental results (see Figures 14 and 15) that for the two algorithms, the simplification rate decreases with the increase of the simplification threshold among the five data sets of different sizes. Also, compared to algorithm Angular, the highest simplification rate of algorithm STIF is only a data size of 1000. When the simplification threshold is set to $1/12\pi$, it is

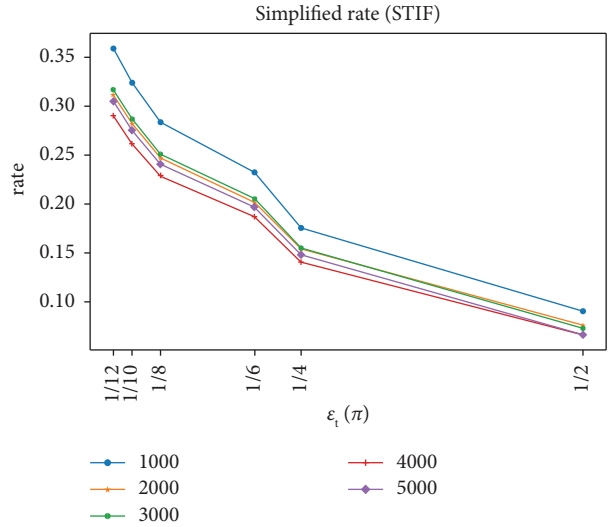

Figure 15: The change of the simplification rate of different data scales with the increase of the simplification threshold of the STIF algorithm.

Input: The starting point P_s, ε, $\varepsilon_t = \varepsilon/2$, Angle deviation of current point $p_{i.\varepsilon_d}$, Cumulative angular deviation of current point $p_{i.\varepsilon_a}$, Upper interval positive_$\varepsilon_t = \varepsilon_t$, Lower interval negative_$\varepsilon_t = -\varepsilon_t$, Current maximum and minimum angle deviation max_$\varepsilon_a = 0$, min_$\varepsilon_a = 0$, Float limit flag
Output: Simplified trajectory T'
for (int $i = 1$; $i < P$.size() $- 1$; i++){
$p_{i.\varepsilon_a} = p_{i.\varepsilon_a} + p_{i.\varepsilon_d}$
if (max_$\varepsilon_a < p_{i.\varepsilon_d}$)
max_$\varepsilon_a = p_{i.\varepsilon_d}$;
if (min_$\varepsilon_a > p_{i.\varepsilon_d}$)
min_$\varepsilon_a = p_{i.\varepsilon_d}$
//If the angle deviation of the current point is greater than $2 * \varepsilon_t$
if (Math.abs($p_{i.\varepsilon_d}$) $> 2 * \varepsilon_t$){
add (P.get(ii));
$p_{i.\varepsilon_a} = 0$;
flag = 0;
positive_$\varepsilon_t = \varepsilon_t$;//"Zero adjustment" in the upper section"
negative_$\varepsilon_t = -\varepsilon_t$; //"Zero adjustment" in the lower interval
max_$\varepsilon_a = 0$;
min_$\varepsilon_a = 0$;
}
if (Math.abs($p_{i.\varepsilon_a}$)> ε_t){
//The upper interval meets the floating condition
if ($p_{i.\varepsilon_a}$ > positive_ε_t && $p_{i.\varepsilon_a} < 2 *$ positive_ε_t && flag == 0){
//Float in the lower interval
negative_ε_t = min_ε_a;
//Float up the upper interval
positive_$\varepsilon_t = 2 * \varepsilon_t$ + min_ε_a;
//Calculate the subsequent points after the interval has floated once. When //the simplified condition is reached again, "zero"
flag = 1;
} else if($p_{i.\varepsilon_a}$ < negative_ε_t && $p_{i.\varepsilon_a} > 2 *$ negative_ε_t && flag = = 0){
//The lower interval meets the floating condition
//Float in the upper interval
positive_ε_t = max_ε_a;
//Float down in the lower interval
negative_$\varepsilon_t = -2 * \varepsilon_t$ + max_ε_a;
flag = 1;
}else if(flag = = 1){

Algorithm 1: Continued.

```
        //Already floated once, cannot float again
        add (P.get(ii));
        p_{i.ε_a} = 0;
        flag = 0;
        positive_ε_t = ε_t; //"Zero adjustment" in the upper section
        negative_ε_t = -ε_t;//"Zero adjustment" in the lower interval
max _ε_a = 0;
min _ε_a = 0;
        } else if(Math.abs (p_{i.ε_a}) >2 * ε_t) { add(P.get(ii));
           p_{i.ε_a} = 0;
           flag = 0;
           positive_ε_t = ε_t;//"Zero adjustment" in the upper section
           negative_ε_t = −ε_t; //"Zero adjustment" in the lower interval
max _ε_a = 0;
min _ε_a = 0;
           }
           if (p_{i.ε_a} < negative_ε_t || p_{i.ε_a} > positive_ε_t){
              add (P.get(ii));
              p_{i.ε_a} = 0;
              flag = 0;
              positive_ε_t = ε_t; //"Zero adjustment" in the upper section
              negative_ε_t = −ε_t; //"Zero adjustment" in the lower interval
max _ε_a = 0;
min _ε_a = 0;
              }
           } //if (Math.abs(deviation) >error_t)
        }
        add (P.get(P.size()-1)); //The last point is added
}
```

Algorithm 1: Simplified Algorithm of Moving Object Trajectory Based on Interval Floating (STIF).

slightly higher than 0.35, but for algorithm Angular, when the simplification threshold is $1/12\pi$, the highest simplification rate is greater than 0.45, and the lowest simplification rate is also greater than 0.35. Speaking of massive trajectory data, the number of acquisition points for each trajectory is more than tens of thousands of points. In this case, the Angular algorithm retains too many trajectories data are not conducive to efficient data storage. On the contrary, the STIF algorithm can ensure the accurate retention of points while reducing the simplification rate.

5. Conclusions

This article mainly introduces a new trajectory simplification algorithm based on interval floating. Various experiments were conducted from different data set sizes and different angle thresholds to evaluate the time performance and simplification rate performance of the algorithm. Through experimental comparison, algorithm STIF has outperformed algorithm Angular in simplification rate; in addition, as the data set increases, the average simplification time of algorithm STIF is slightly longer than that of algorithm Angular when the simplification threshold is smaller. With the increase of the threshold, the simplification time of the STIF algorithm is significantly reduced. Therefore, in the face of large-scale trajectory data, the STIF algorithm can show a better simplification rate and simplification time. For future works, it is planned to assess the algorithms with other datasets, considering other transportation modes and trajectories' characteristics as well as different application scenarios.

Data Availability

The dataset used to support the results of this experiment is the GPS trajectory dataset, which is collected by Microsoft Research Asia. In more than three years (from April 2007 to August 2012), 182 users participated in the Geolife project. The data used in the algorithm experiment can be obtained through the following website: https://research.microsoft.com/en-us/projects/geolife/.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

References

[1] X. He and D. Kempe, "Stability of inuence maximization," in *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, Washington, DC, USA, August 2014.

[2] M. Lichman and P. Smyth, "Modeling human location data with mixtures of kernel densities," *Proceedings of the 20th*

ACM SIGKDD international conference on Knowledge discovery and data mining, , New York, NY, US, August 2014.

[3] Y.-H. Lin, C.-H. Lai, and P.-R. Lei, *Mining top-K relevant stay regions from historical trajectories*PAKDD, Chengdu, China, 2014.

[4] A. Xue, R. Zhang, Y. Zheng, X. Xie, J. Huang, and Z. Xu, "Destination Prediction by Sub-trajectory Synthesis and Privacy protection against Such Prediction," in *Proceedings of the 2013 IEEE 29th International Conference on Data Engineering (ICDE)*, Brisbane, QLD, Australia, April 2013.

[5] R. Sowmya and K. R. Suneetha, "Data Mining with Big Data," in *Proceedings of the International Conference on Intelligent Systems & Control. IEEE*, Coimbatore, India, January 2017.

[6] D. H Douglas, T. K. Peucker, and T. K. Peucker, "Algorithms for the reduction of the number of points required to represent a digitized line or its caricature," *Cartographica: The International Journal for Geographic Information and Geovisualization*, vol. 10, no. 2, pp. 112–122, 1973.

[7] N. Meratnia and B. R. De, "Spatiotemporal compression techniques for moving point objects," in *Proceedings of the Advances in Database Technology - EDBT 2004, International Conference on Extending Database Technology*, pp. 765–782, Heraklion, Crete, Greece, March 2004.

[8] C. Y. Lin, C. C. Hung, and P. R. Lei, "A velocity—preserving trajectory simplifi—cation approach," in *Proceedings of the of 2016 Conference on Technologies and Applications of Artificial Intelligence (TAAI)*, p. 58—65, IEEE Press, Hsinchu, Taiwan, November 2016.

[9] B. Ke, J. Shao, Y. Zhang, and Y. Yang, "An online approach for direction-based trajectory compression with error bound guarantee," in *Proceedings of the Asia-Pacific Web Conference*, pp. 79–91, Suzhou, China, September 2016.

[10] E. Keogh, S. Chu, D. Hart, and M. Pazzani, "'An online algorithm for segmenting time series," in *Proceedings of the IEEE Int. Conf. Data Mining*, pp. 289–296, San Jose, CA, USA, November 2002.

[11] R. W. Liu, J. Nie, S. Garg, Z. Xiong, Y. Zhang, and M. S. Hossain, "Data-driven trajectory quality improvement for promoting intelligent vessel traffic services in 6G-enabled maritime IoT systems," *IEEE Internet of Things Journal*, vol. 8, no. 7, pp. 5374–5385, 2021.

[12] R. W. Liu, M. Liang, J. Nie, W. Y. B. Lim, Y. Zhang, and M. Guizani, "Deep learning-powered vessel trajectory prediction for improving smart traffic services in maritime Internet of things," *IEEE Transactions on Network Science and Engineering*, vol. 9, no. 5, pp. 3080–3094, 2022.

[13] G. Trajcevski, H. Cao, P. Scheuermanny, and O. Wolfson, "On-line data reduction and the quality of history in moving objects databases," in *Proceedings of the 5th ACM International Workshop on Data Engineering for Wireless and Mobile Access*, pp. 19–26, DBLP, Chicago, Il, Usa, June 2006.

[14] J Muckell, P. W. Olsen, J. H Hwang et al., "Compression of trajectory data: a comprehensive evaluation and new approach," *GeoInformatica*, vol. 18, no. 3, pp. 435–460, 2014.

[15] C Long, R. C. W. Wong, H. V. C. W. Jagadish, and H. V. Jagadish, "Direction-preserving trajectory simplification," *Proceedings of the VLDB Endowment*, vol. 6, no. 10, pp. 949–960, 2013.

[16] Z. Yu, L. Zhang, X. Xing, and W. Y. Ma, "Mining interesting locations and travel sequences from GPS trajectories," in *Proceedings of the 18th International Conference on World Wide Web*, WWW, Madrid, Spain, April 2009.

[17] Y. Zheng, Q. Li, Y. Chen, and W. Y. Ma, "Understanding mobility based on GPS data[C]," in *Proceedings of the Ubiquitous Computing, 10th International Conference*, pp. 21–24, Seoul, Korea, September 2008.

[18] Y. Zheng, X. Xing, and W. Y. Ma, "GeoLife: A collaborative social networking service among user, location and trajectory," *Bulletin of the Technical Committee on Data Engineering*, pp. 32–40, 2010.