A Proposal of Online Map-matching Based Trajectory Compression Algorithm Using Road Networks

Shota Iiyama, Masaharu Hirota
Department of Information Science, Okayama University of Science
1-1 Ridaicho, Kita-ku, Okayama-shi, 700-0005, Japan

Tetsuya Oda
Department of Information and Computer Engineering, Okayama University of Science
1-1 Ridaicho, Kita-ku, Okayama-shi, 700-0005, Japan
E-mail: i17i005is@ous.jp, oda@ice.ous.ac.jp, hirota@mis.ous.ac.jp

Abstract

As the data size of GPS logs increases, the amount of data transferred from mobile devices to the server, and the computational cost of analysis of GPS logs increase. One of the solutions to these problems is to compress the GPS logs. However, it is difficult to compress a sparse GPS log while preserving the feature points in the GPS log, such as the user's movement speed, the shape of the GPS log trajectory, and the direction of movement of the GPS log. This study proposes an online compression algorithm for GPS logs that maintains the compression rate while preserving GPS logs' feature points. Our proposed method compresses GPS logs by using information from a road network to identify roads traveled by the user. We evaluate the performance of this method using the GPS data of the bus.

Keywords: GPS, compression algorithm, road network, characteristics of user movement.

1. Introduction

The number of GPS logs that record users' activities increases. Therefore, the data transferred from mobile devices to the server is a critical problem. Another problem is that the processing speed of analysis of GPS logs decreases when analyzing the GPS logs.

One solution to these issues is to compress the GPS logs. Compressing the GPS logs reduces the amount of them transmitted from the mobile device to the server. Also, adequately compressed GPS logs reduce the amount of data processing for analysis.

However, if the GPS log compression lost feature points, the analysis's performance may be worse than the uncompressed. Therefore, it is important to preserve as much of the features as possible. This paper proposes an algorithm for compressing GPS logs moving on the road network while preserving the trajectory's feature points.

2. Related work

The method for GPS log compression without using the road network are Douglas-Peucker Algorithm (DP algorithm)[1] and top-down time-ratio algorithm (TD-TR algorithm)[2]. These methods focus on the shape of the GPS log and compress the GPS log.

Also, the methods for GPS log compression using the road network include Schmid et al.'s method[3]. This method is a compression method for GPS logs of users moving on roads in urban areas. In this method, map-matching is applied to the GPS logs to identify the roads moved by the user. Next, the method uses compression

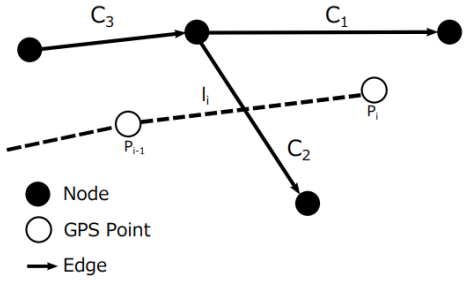

Figure 1 Map-matching.

points the GPS logs obtained at intersections and bus stops on the roads moved by the user.

Our method compresses GPS logs not only by focusing on the shape, but the features of movement.

3. Method

Our compression method employs three methods to preserve the feature points of GPS logs: Opening Window Compression Algorithm (OW)[2], CriticalPoint Algorithm (CP)[4], and StayPoint (SP)[5]. OW is a shape-oriented compression algorithm. CP is a compression algorithm that focuses on changes in velocity and direction of motion. SP is the point at which the user was staying. Our method makes a compression result by combining each of the OW, CP, and SP extracted from the GPS logs.

3.1. *Map-matching*

Our method uses the offline map-matching method proposed by Brakatsoulas et al.[6]. In our method, we extend this method to apply to online data.

Figure 1 shows the example of a network. C_1, C_2, and C_3 in Figure 1 represent the vectors from the start to the end node of the road. A node represents an intersection.

This method finds the candidate roads C_1 and C_2 and selects one of them using equation (1) for the point P_i.

$$S_{p_i} = \mu_d - a \cdot D(P_i, C_i)^{n_d} + \mu_\alpha - \cos(\alpha_{i,j})^{n_\alpha} \quad (1)$$

μ_d, μ_α, n_d, and n_α are constants. C_j is a candidate road in P_i. $D(P_i, C_j)$ is the geographical distance between P_i and C_j. $\alpha_{i,j}$ is the difference between the angles of l_i and C_j. l_i is the vector from P_{i-1} to P_i. $\cos(\alpha_{i,j})$ is the cosine value of $\alpha_{i,j}$. Among the candidate roads, the road with the highest value of S_{p_i} is the destination of P_i.

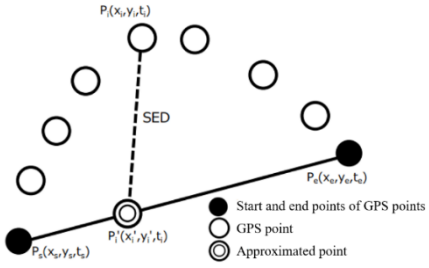

Figure 2 Synchronized Euclidean Distance.

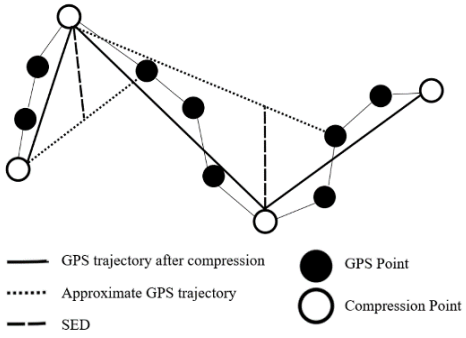

Figure 3 Opening Window Compression Algorithm.

3.2. *Opening Window Compression Algorithms*

First, we describe the SED that we uses in OW. The SED is the value of the distance between point P'_i of the GPS log after compression and point P of the GPS log before compression, considering the time information.

We use the method of Meratnia et al.[2] to obtain the SED. Figure 2 shows an example of SED. The P_i shows one of the GPS points in the GPS log. P'_i is the point at which P_i is approximated on the line P_s to P_e. x'_i and y'_i are found by the following equations.

$$\Delta e = t_e - t_s, \quad \Delta i = t_i - t_s \quad (2)$$

$$x'_i = x_e + \frac{\Delta i}{\Delta e}(x_e - x_s) \quad (3)$$

$$y'_i = y_e + \frac{\Delta i}{\Delta e}(y_e - y_s) \quad (4)$$

OW defines the $anchor$ and $float$. The $anchor$ p_s is the first point in the GPS log. The $float$ p_e is the third point from the beginning of the GPS log. The SED is calculated for the GPS point between the $anchor$ and the $float$, respectively. If there are no points at which the SEDs are above the threshold, the $float$ is changed to the next GPS point. Otherwise, the point with the highest SED among the points whose SED are above the threshold is the compressed point and the next $anchor$.

The process of computing the SED of a GPS point between each $anchor$ and $float$ is repeated. This process is repeated until the OW terminates when GPS positioning is finished. Figure 3 shows an example of a compressed GPS log with OW applied.

The difference between our proposed OW and the original OW is how to choose $anchor$ and $float$. In original OW, where an $anchor$ is the first point in the GPS log, and $float$ is the third point from the GPS log's beginning. On the other hand, our method uses the results of map-matching to select $anchor$ and $float$.

3.3. *StayPoint*

The StayPoint represents a point where the user was in a range for a amount of time. A GPS log that becomes a StayPoint satisfies the following conditions.

$$m < i \leq n, Int(P_m, P_n) \geq T_r \quad (5)$$

$$D(P_m, P_i) \leq D_r, Dist(P_m, P_{n+1}) > D_r \quad (6)$$

P_i is a point of trajectory in the GPS log. $D(P_m, P_i)$ is the distance between the two points P_m and P_i based on the Hubeny formula. D_r is the threshold for the distance between P_m and P_i. $Int(P_m, P_n)$ is the difference in time between P_m and P_n. T_r is the threshold for the difference in time between P_m and P_n. From the GPS log that satisfies these equations, we calculate the StayPoint. Let StayPoint be s, then $s = (x, y, t_a, t_l)$. $s.x$ and $s.y$ represent the coordinates of s. $s.t_a$ represents the start time of the user's stay at the StayPoint. $s.t_l$ represents the time when the user ends his stay at the StayPoint. $s.x$, $s.y$, $s.t_a$, and $s.t_l$ are computed by following equations.

$$s.x = \sum_{i=m}^{n} \frac{P_i.x}{|P|}, s.y = \sum_{i=m}^{n} \frac{P_i.y}{|P|} \quad (7)$$

$$s.t_a = P_m.t_m, s.t_a = P_n.t_n \quad (8)$$

Our method extracts the StayPoint from the GPS log and preserves the state of staying at one point in the GPS log before compression. In our method, s is represented by $sa = (x, y, t_a)$ and $sl = (x, y, t_l)$ to describe the same format as the traditional GPS log.

3.4. *CriticalPoint Algorithm*

CP is a compression method that focuses on the speed and direction of the user's movement. At two consecutive points P_i, P_{i-1} in the GPS log, the compression point satisfies either of the following conditions.

$$SD(P_i, P_{i-1}) > Sr, AD(P_i, P_{i-1}) > Ar \quad (9)$$

$SD(P_i, P_{i-1})$ calculates the difference in speed between P_i and P_{i-1}.$AD(P_i, P_{i-1})$ calculates the angle difference between P_i and P_{i-1}. Sr is the threshold for the difference in velocity between P_i and P_{i-1}. Ar is the threshold for the angle difference between P_i and P_{i-1}.

3.5. *Integrated processing of compression points*

If there is SP or CP compression point within a certain distance from the OW compression points, our method deletes the OW compression points. This reason is compression points by OW preserve only the shape features, but the SP or CP preserve velocity or state of staying in addition to the shape features. The proposed method regards the points obtained as a result of this process as compressed points.

4. Experiment

We evaluate the performance of our method for the GPS logs compression preserving the features of the GPS logs while maintaining the Compression Rate. We used the GPS log of buses in Okayama Prefecture, Japan. The number of GPS logs is 297. The average number of GPS points in the GPS log is about 185.9. The average positioning interval of the GPS logs is about 18.3 seconds. Also, map-matching uses the road network of Okayama Prefecture obtained from OpenStreetMap.

This experiment uses three evaluation criteria: SED Error, Speed Error, and Compression Rate. The SED Error is the difference between the shape of the GPS log before and after compression. This value is the median and average of the SED calculated from the GPS logs before and after compression. Speed Error is the difference between the speed of the GPS log before and after compression. Compression Rate is the compression rate of the GPS logs

The proposed method parameter is the threshold of the SED, which is used to find the compression point in the OW. We changed the threshold of the SED by ten from 30 to 200. For the GPS logs compressed by the proposed method, we evaluate them by each evaluation criterion.

4.1. *Experiment Results and Discussion*

Figure 4, Figure 5, and Figure 6 show the evaluation results of SED Error, Speed Error, and Compression Rate of the GPS logs compressed by our method. Their

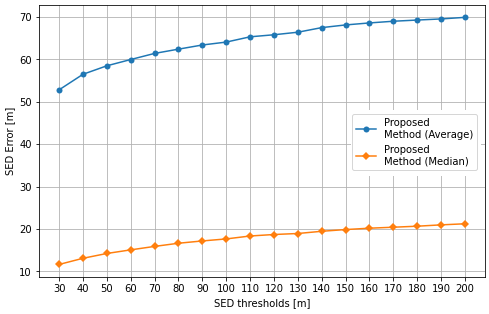

Figure 4 SED Error.

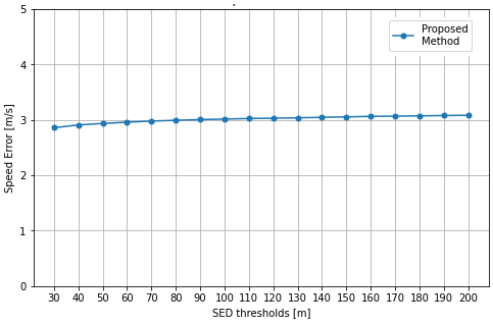

Figure 5 Speed Error.

horizontal axis is the threshold value of the SED. The vertical axis is the value of each evaluation criterion.
In Figure 4, the average value of SED Error is large more than the median. This result shows that our method can compress almost GPS logs preserving the feature, but the method when compressing some GPS logs failed.

The reason why the proposed method has tremendous SED Error values in some GPS logs is due to calculating the pseudo arrival time of the $float$ used in the OW of the proposed method. The pseudo arrival time assumes that the user moves straight from the GPS point to the $float$. However, because roads are not always straight but have curves, there is a difference between the expected and the actual arrival time.

The Speed Error in Figure 5 shows that the Speed Errors are about 3m/s in all SED thresholds. When the threshold of SED was changed, the value of Speed Error did not change much. This result indicates that the proposed method is not much affected by the threshold of SED and can preserve velocity features.

Figure 6 shows the Compression Rate results of the proposed method. As the threshold of SED increases, the Compression Rate also increases. This result shows that the proposed method can change the Compression Rate by varying SED's threshold value. Figure 4 and Figure 5 show that the proposed method can preserve velocity and shape features, even when the Compression Rate is

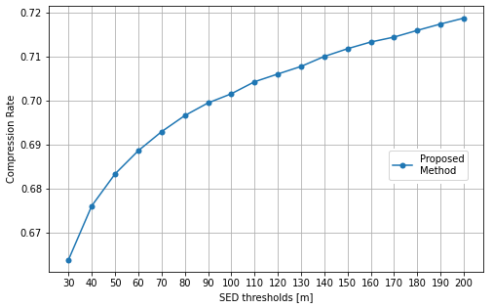

Figure 6 Compression Rate

increased. Therefore, our method can compress GPS logs while considering the trade-off between the shape and velocity information and the Compression Rate.

5. Conclusion and future work

This paper proposes an online compression algorithm for GPS logs that uses three methods to preserve the feature points. Our experiments show our method can compress GPS logs preserving velocity and shape features.

Future work will improve map-matching performance because the compression of our proposed method depends on the performance of the map-matching.

Acknowledgements

This work was supported by JSPS KAKENHI Grant Number JP19K20418 and Grant for Promotion of OUS Research Project (OUS-RP-20-3).

References

1. Douglas, D. H. and Peucker, T. K.: Algorithms for the reduction of the number of points required to represent a line of its caricature, IJGIG, Vol. 10, No 2, pp. 112–122, 1973.
2. Meratnia, N. and de By, R. A.: Spatiotemporal compression techniques for moving point objects, EDBT, pp. 765–782, 2004.
3. Schmid, F., Richter, K. F. and Laube. P.: Semantic trajectory compression, SSTD, Vol.5644, pp. 411–416, 2009.
4. Barbeau, S., Labrador, M. A., Perez, A., Winters, P. Georggi, N., Aguilar, D. and Perez, R.: Dynamic management of real-time location data on GPS-enabled mobile phones, UBICOMM, pp. 343– 348, 2008.
5. Zheng, V. W., Zheng, Yu., Xie, X. and Yang, Q.: Collaborative location and activity recommendations with GPS history data, WWW, pp. 1029–1038, 2010.
6. Brakatsoulas, S., Pfoser, D., Salas, R. and Wenk, C.: On map-matching vehicle tracking data, VLDB, pp. 853–864, 2005