GPS Trajectory Compression Algorithm

Gary Reyes Zambrano(✉)

Ecuador Facultad de Ciencias Matemáticas y Físicas, University of Guayaquil, Cdla. Salvador Allende, Av. Delta y Av. Kennedy, Casilla Postal 471, Guayaquil, Ecuador
gary.reyesz@ug.edu.ec

Abstract. This research is oriented toward the development of a trajectory compression algorithm for global positioning systems. In order to increase the compression ratio of the data, an algorithm is developed based on the algorithm of compression of GPS trajectories Top Down - Time Ratio. The algorithm is composed of a filter for noise reduction and makes use of semantic information to accept or discard relevant points of the trajectory. The experiments of the algorithm were carried out using three trajectory datasets: Mobile Century Data, Geolife Trajectories and T-Drive Data, increasing the compression ratio of the data, which leads to improvements in efficiency. With the results obtained, statistical tests were performed that allowed us to compare the results, compare it with other trajectory compression algorithms and validate the investigation.

Keywords: Compression · GPS data analysis · GPS data simplification

1 Introduction

Data compression process consists on taking a sequence of symbols and transforming them into codes, if the compression is effective, the resulting sequence of codes will be smaller than the original symbols [1]. This process should preserve the statistical purposes and other characteristics of the data while reducing the size [2]. The preservation of data properties and reduction levels depends on the compression algorithm used.

Compression algorithms can be classified into two categories, lossless compression algorithms and lossy compression algorithms. Lossless compression algorithms perform a more accurate reconstruction of the original data without loss of information. In contrast, lossy compression algorithms are inaccurate compared to the original data [3].

A GPS trajectory is represented as a discrete sequence of geographic coordinate points [4]. There are currently active research areas related to GPS trajectories. Among them is the GPS trajectory preprocessing area which studies the techniques and algorithms of trajectory compression. This algorithm remove some sub-traces of the original trajectory [5]; reducing data storage space and data transfer time.

Reducing the data size of a GPS trajectory makes it easier to speed up the information extraction process [6]. There are several methods of trajectory compression that are suitable for different types of data and produce different results, but they all have the same principle in common: compress the data by eliminating redundancy of the data in the source file [7–9].

M. Botto-Tobar et al. (Eds.): ICCCE 2018, CCIS 959, pp. 57–69, 2019.
https://doi.org/10.1007/978-3-030-12018-4_5

The bibliography describes a set of algorithms for the compression or simplification of GPS trajectories and their limitations [10–13] among which are highlighted:

- Douglas-Peucker: Does not compress data in real time.
- Visvalingam: The error rate and processing time are high.
- TD-TR: The error rate of the algorithm is high and does not compress data in real time.
- Opening Window: Its main disadvantage is the frequent removal or distortion of important points such as sharp angles. A secondary limitation is that straight lines are still overrepresented. For its correct functioning requires high hardware performance.
- ST-Trace: The processing time is considerable and requires speed information to characterize the trace.
- None of the algorithms consider the noise present in the trajectory data, which reduces the possibility of eliminating noisy points and improving the process of simplifying points.
- Only Squish and Dots algorithms perform a rigorous analysis of the GPS trajectory decoding procedure, but do not consider the analysis of trajectory noise.
- Douglas Peucker, Visvalingam and Opening Window algorithms only perform spatial analysis of the data. This eliminate temporary information that provides important data for better compression rates.
- The Visvalingam compression algorithm eliminates or distorts points, such as sharp angles, so the resulting trajectory may lack of important points to reconstruct a route.
- None of the algorithms consider the semantic information of the trajectory, wasting the opportunity to make an analysis that allows to discard more points of little meaning from the original trajectory.

This article describes an enhanced algorithm for GPS trajectory compression. In the previous works section, an analysis of the GPS trajectory compression algorithms is done. The GPS trajectory compression algorithm section describes the algorithm developed in this research. In the analysis of the results section the results obtained are shown. In the section on conclusions and future work, the main conclusion and the future work is presented.

2 Background Work

In this section, the algorithms described in the literature are analyzed in order to determine the main elements involved in the compression of GPS trajectory data. As part of the analysis, a review of different behaviors and conditions affecting GPS trajectory compression was conducted using various algorithms proposed in the literature. Table 1 shows the results of previous review.

Table 1. Behavior and conditions that affect GPS trajectory compression

Article	Year	Compression behavior	Conditions affecting the compression ratio
A new perspective on trajectory compression techniques	2003	Improved compression ratio by measuring error distances between synchronized positions	Type of trip (unidirectional, multidirectional), type of transport (taxi, bus)
Compressing trajectories using inter-frame coding	2010	The experimental compression ratio with uninterrupted trajectories is similar to the theoretical compression ratio	Short trajectories, uninterrupted trajectories
A trajectory compression algorithm based on non-uniform quantization	2015	Improves compression rate when processing data from large-scale trajectories in a geographic context	Geographical context (road networks or routes), cars, planes, ships
Improvement of OPW-TR algorithm for compressing GPS trajectory data	2017	Improves compression rate while decreasing data loss	The shape of the trajectory does not consider temporal information

2.1 Line Compression Algorithm

This algorithm, also known as line generalization or Lang's algorithm [14–16], works on the basis of the analysis of three points at the same time. The first three points are chosen and a line is drawn between the first and third points. If the distance between the line and the second point is greater than the defined tolerance, the second point is selected to analyze from the second point to the fourth point. If the distance is smaller than the tolerance, the second point is eliminated and the process is repeated from the third point to the fifth point and so on with the rest of the coordinates.

For the calculation of the given straight line, the formula 1 is used:

$$(x - x_1)/(x_2 - x_1) = (y - y_1)/(y_2 - y_1) \tag{1}$$

For the calculation of the distance between a line and a point the formula 2 is used:

$$|A * xp + B * yp + C|A2 + B^2 \tag{2}$$

2.2 Douglas-Peucker GPS Trajectory Compression Algorithm

Douglas-Peucker (DP) is a GPS trajectory compression algorithm based on the top-down method for data analysis. It is used to remove a series of line segments from a curve, which reduces the amount of data present in a GPS trajectory [17]. It's a line generalization algorithm. Recursively select points from the original series of GPS trajectory points [18–21].

Douglas-Peucker implements a divide and conquer strategy and is computed in four steps [22, 23]:

- The first and last node of a polygonal string are taken as the end points of a line.
- For all intermediate points, the shortest distance to this line is determined. If it is greater than any distance, the polygonal chain is separated by the point with the greatest distance, forming two new segments.
- The new segments formed are analyzed using the same procedure.
- The algorithm ends when it does not exceed any point line distance.

The computational complexity of the algorithm in the worst case is $O(n^2)$, where n is the number of original points. The computational complexity of the algorithm in the worst case can be improved to $O(n \log_n)$ using an approach involving convex hulls [6].

2.3 Top Down Time Ratio Line Simplification Algorithm

This algorithm is a modification of the Douglas-Peucker algorithm where the time variable is added. To do this, the coordinates of the point P_i' in time are calculated using the ratio of two time intervals.

$$\Delta_e = t_e - t_s \tag{3}$$

The difference between the time of the point to be analyzed and the time of the starting point is calculated using the formula 4:

$$\Delta_i = t_i - t_s \tag{4}$$

To obtain the coordinates of P_i', formulas 5 and 6 are applied:

$$x_i' = x_s + \frac{\Delta_i}{\Delta_e}(x_e - x_s) \tag{5}$$

$$y_i' = y_s + \frac{\Delta_i}{\Delta_e}(y_e - y_s) \tag{6}$$

After obtaining the coordinates, the synchronous Euclidean distance between P_i' and P_i, is calculated. If the distance is greater than the tolerance, this reference point is taken and the calculation of the intervals is performed again. The computational complexity in the worst case is $O(n^2)$. $O(n \log_n)$ implementation enhancement for Douglas-Peucker that takes advantage of geometric properties cannot be applied to TD-TR [24].

2.4 Visvalingam-Whyatt Algorithm

The Visvalingam-Whyatt algorithm use the concept of effective area, which is defined as the area of the triangle formed by a point and its two neighbors. The algorithm takes a poly-line P as the sequence of points, and the spatial displacement error is defined by

the user. For each set of three consecutive points a triangle is formed, this being the effective area. Iteratively, the point that produces the least displacement of the area is selected to form an approximation. This process stops when the effective area is larger than ε [10, 21].

2.5 Experimental Analysis of the Algorithms

As part of the research process, an experiment was conducted with the analyzed algorithms. The results can be seen in Table 2.

Table 2. Comparative table of the analyzed algorithms

Algorithms	Execution time (seg)	Compression ratio (%)	Error rate
Douglas-Peucker	50,06	19,00	0,0225
Line simplification algorithm	154,69	99,49	0,0491
Visvalingam	280,99	32,84	0,0333
TD-TR	1512,57	99,60	0,0391

The experiment consists of running all the algorithms using the same database. The table shows the results obtained, from which it is decided to use the TD-TR line simplification algorithm as base for the selection of points in the compression algorithm proposed in this research.

3 GPS Trajectory Compression Algorithm

In this research, a GPS trajectory compression algorithm called GR-B is proposed. The algorithm consists in three stages.

1. Noise reduction
2. Line simplification
   - Simplification of semantic points
   - Point simplification based on TD-TR
3. Data compression

The algorithm has as input the GPS trajectory dataset to be compressed and as output the compressed GPS trajectory dataset. It starts with the application of an algorithm based on the noise reduction logic of the Kalman algorithm to eliminate all points that are considered noise. The output of this step is the input for the point simplification logic used in the TD-TR algorithm, in which the spatial and temporal elements of the data are taken into account. To improve the simplification of points in the TD-TR algorithm, semantic analysis of the trajectory is used.

Once the data has been simplified, compression is carried out using the Brotli algorithm. The data is divided into three independent vectors to compress each vector,

resulting in three smaller compressed files. As a result of applying these three steps you have a set of compressed GPS data.

3.1 Noise Reduction Using Kalman's Algorithm Logic

This algorithm initially builds a model, closely related to the trajectory to be analyzed, in order to adjust the filter.

Functions 7 and 8 are used to build the model:

$$X_k = Ax_{k-1} + Bu_k + W_{k-1} \tag{7}$$

$$Z_k = Hx_k + V_k \tag{8}$$

The process values are then initialized based on the values of the first latitude and longitude point of the GPS trajectory. The values are initialized using prediction (9 and 10) and correction (11, 12 and 13) functions [25].

Functions for time update (Prediction):

$$X_k = Ax_{k-1} + Bu_k \tag{9}$$

$$P_k = AP_{k-1}A^t + \mathrm{Q} \tag{10}$$

Measurement update functions (Correction):

$$K_k = P_k H^t \left(HP_k H^T + R\right)^{-1} \tag{11}$$

$$X_k = X_k + K_k(z_k - Hx_k) \tag{12}$$

$$P_k = (1 - K_k H)P_k \tag{13}$$

Once all the necessary information has been collected and the values have been initialized, the estimations can be repeated. Each point estimation is based on the previous point entry. The iterative process of the Kalman filter is decomposed in two stages: (1) the prediction of the state from the previous state is observed in the formula 9 and 10 and (2) the correction of the prediction using the observation of the current state is observed in the formulas 11, 12 and 13.

3.2 Trajectory Point Simplification

The simplification stage starts with the drawing of the initial line segment between the first and last points. Then, using the synchronous Euclidean distance, the distances from all points to the line segment are calculated, the point furthest from the line segment (or the maximum distance) is identified and marked. If the distance from the selected point to the line segment is less than the defined tolerance, all unmarked points are discarded, otherwise select the marked point for evaluation with the semantic layer and continue to divide the linear segment with this point. This procedure is executed recursively.

If the point is marked, then it is evaluated with the semantic layer to decide whether or not it can be added to the final simplified trajectory. To evaluate a marked point, the distance from the maximum circle of the point to all semantic points is calculated using the function 14

$$\cos(d) = (\sin a \ \sin b) + (\cos a \ \cos b \cos |c|) \tag{14}$$

Where a and b represent latitudes in degrees and c represents the absolute value of the longitude difference between the respective coordinates. A point is accepted if the distance to the nearest semantic point is less than the semantic tolerance.

3.3 Lossless Compression

For the compression of the resulting points, the lossless compression algorithm Botli [26–28] was selected and applied as stated in the literature. The application of this algorithm makes it possible the reduction of the space needed to store the GPS trajectories. This stage is a second line of data compression by which no data is lost.

Brotli is a compression algorithm announced by Google in September 2015. Brotli's decompression is as fast as gzip while significantly improving the compression ratio. The disadvantage is that compression is slower than gzip. This algorithm compresses the data using a sequence of bytes, starting with the first byte on the right side and proceeding to the left, with the most significant bit of each byte on the left. In this way, the result can be analyzed from right to left, with elements of fixed width in the correct order of msb-to-lsb and prefix codes in bit-reversed.

A compressed dataset consists on a header and a series of meta blocks. Each meta block decompresses to a sequence of 0 to 16,777,216 (16 MB) uncompressed bytes. The uncompressed final data is the concatenation of the uncompressed sequences of each meta block. The header contains the size of the slider window that was used during compression. The decompressor must retain at least the same amount of uncompressed data before the current position in the stream in order to decompress what follows. The size of the slider window is a power of two, minus 16, where the power is in the range of 10 to 24.

Each meta block is compressed using a combination of the LZ77 algorithm and Huffman coding. The result of Huffman's coding is called a "prefix code". The prefix codes for each meta block are independent of the previous or subsequent meta blocks. The LZ77 algorithm can use a reference to a duplicate string that occurs in a previous meta block, up to the size of a sliding window of uncompressed bytes before.

The meta block consists of two parts: a header describing the representation of the compressed data and the compressed data. Compressed data consists on a series of commands. Each command consists of two parts: a sequence of literal bytes (of strings that have not been detected as duplicates within the slider window) and a pointer to a duplicate string, which is represented as a <length, backward distance> pair. There can be zero literal bytes in the command. The minimum length of the string to duplicate is two, but the last command in the meta block is allowed to have only literal and no pointer to a string to duplicate.

4 Analysis of the Results

Three experiments are conducted to evaluate the results of the GR-B algorithm. The first experiment is designed to measure the amount of disk space occupied by the data compressed with the proposed algorithm. The second experiment is designed to validate the compression ratio and the error rate of the TD-TR algorithm and the algorithm developed until the simplification stage. The third experiment is designed to evaluate the compression ratio of the developed algorithm compared to other algorithms proposed in the literature. In the first experiment a random trajectory called the X trajectory is selected and the GR-B algorithm is applied to compress the GPS trajectory points. The result of the experiment is the number of points, the disk space occupied by the compressed trajectory and the compression rate. The original X trajectory is formed by a total of 7591 vehicle GPS trajectory points and takes up 776 kb of disk space initially. After applying the GR-B algorithm the number of end points of the trajectory is 11, occupying 1.18 kb of disk space with a compression rate of 99.86%. After the experiment it was possible to verify that it is possible to reconstruct the trajectory using only the points resulting from the compression process.

The second experiment is designed to evaluate the compression ratio and the error rate. For this purpose, the TD-TR algorithm and the developed algorithm until the simplification stage are executed. This experiment is intended to check whether the parameters are significantly improved with respect to the TD-TR algorithm. It is defined as a null hypothesis (Ho) for experiment 1 that 'sample groups conform to a normal distribution'. Seven observations from the California database were selected for the execution of the experiment with the following characteristics:

- Observation 1: Forty-five trajectories, each one containing between 550 and 1200 points.
- Observation 2: Forty-two trajectories, each one containing between 650 and 1200 points.
- Observation 3: Forty-two trajectories, each one containing between 750 and 1200 points.
- Observation 4: Forty-two trajectories, each one containing between 1000 and 3000 points.
- Observation 5: Two hundred and forty-four trajectories, each one containing between 2000 and 4000 points.
- Observation 6: Forty-two trajectories, each one containing between 2000 and 4000 points.
- Observation 7: Thirty-nine trajectories, each one containing between 4000 and 9000 points.

Table 3 shows the results obtained from the experiments, which are statistically processed using Shapiro-Wilk's test to check the assumption of normality of the data. Figure 1 shows that the values are not adjusted to a normal distribution with a p-value equals to 0.0001402 and 0.3645. Therefore, the null hypothesis (Ho) for the metric, error margin, is rejected and in the metric compression ratio, Fig. 2 shows that the

Table 3. Test results

Tests	Number of points	Number of trajectories	Assumption of normality (compression ratio)	Assumption of normality (margin of error)	Assumption of normality (compression ratio)	Assumption of normality (margin of error)
1	550–1200	45	Rejected Ho	Not rejected Ho	Rejected Ho	Not rejected Ho
2	650–1200	42	Rejected Ho	Not rejected Ho	Rejected Ho	Not rejected Ho
3	750–1200	42	Rejected Ho	Not rejected Ho	Rejected Ho	Not rejected Ho
4	1000–3000	42	Rejected Ho	Not rejected Ho	Rejected Ho	Not rejected Ho
5	2000	244	Rejected Ho	Not rejected Ho	Rejected Ho	Not rejected Ho
6	2000–4000	42	Rejected Ho	Not rejected Ho	Rejected Ho	Not rejected Ho
7	4000–9000	39	Rejected Ho	Not rejected Ho	Rejected Ho	Not rejected Ho

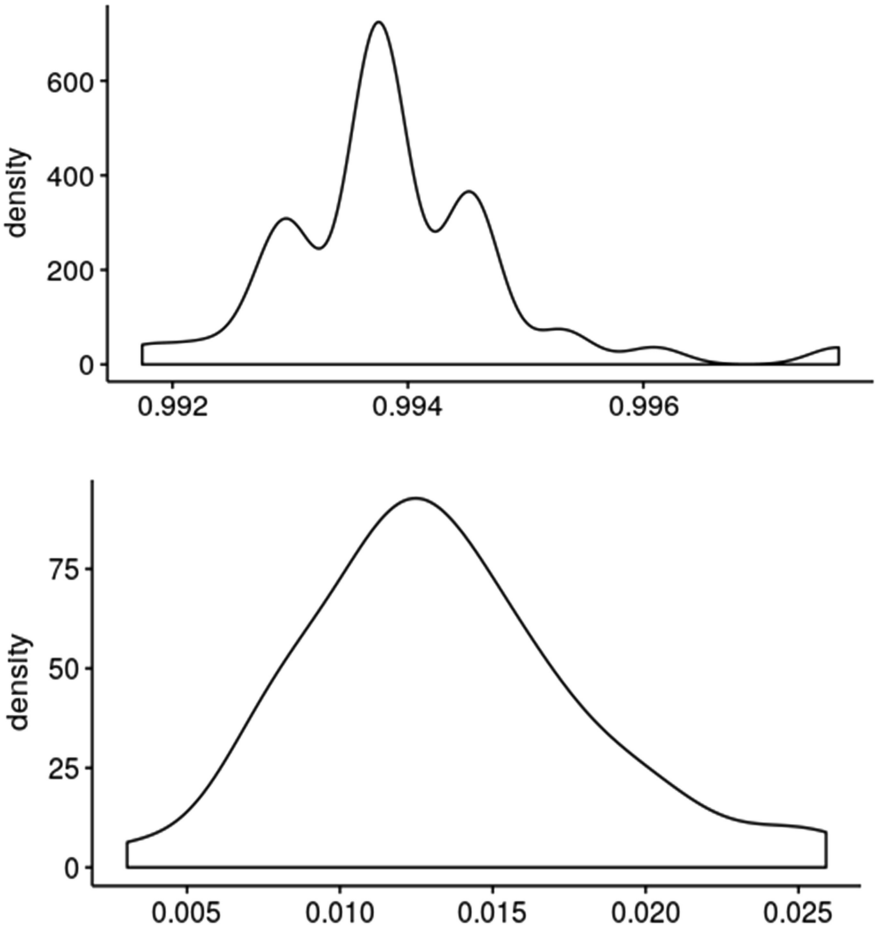

Fig. 1. Density and p-value graph for the metric error rate.

values of the sample are adjusted to a normal distribution with p-values equals to 001621, 0.0003759 and 0.0003857, therefore, the null hypothesis (Ho) is not rejected.

Subsequently, the Mann-Whitney test is applied, obtaining p-values lower than 0.05, which shows significant differences according to the test applied with 95% of confidence. Finally, the Fischer test is applied to check the assumption of the homogeneity of the variances and the Student test to compare the means of the results obtained for the metric error rate. In the application of the Fisher test, it is observed that the p-values obtained are greater than 0.05, so the homogeneity mentioned above is assumed. Once the assumption of homogeneity has been verified, the Student test is applied to compare the vector means of the results obtained for the metric error rate. From the analysis of the p-values obtained, greater than 0.05, it can be concluded that the means of the compared groups are significantly similar. All tests were performed with 95% confidence. The verification performed can ensure that the compression ratio of the proposed algorithm using only the first two steps is better than TD-TR simplification algorithm. It is evident that the error rate remains the same.

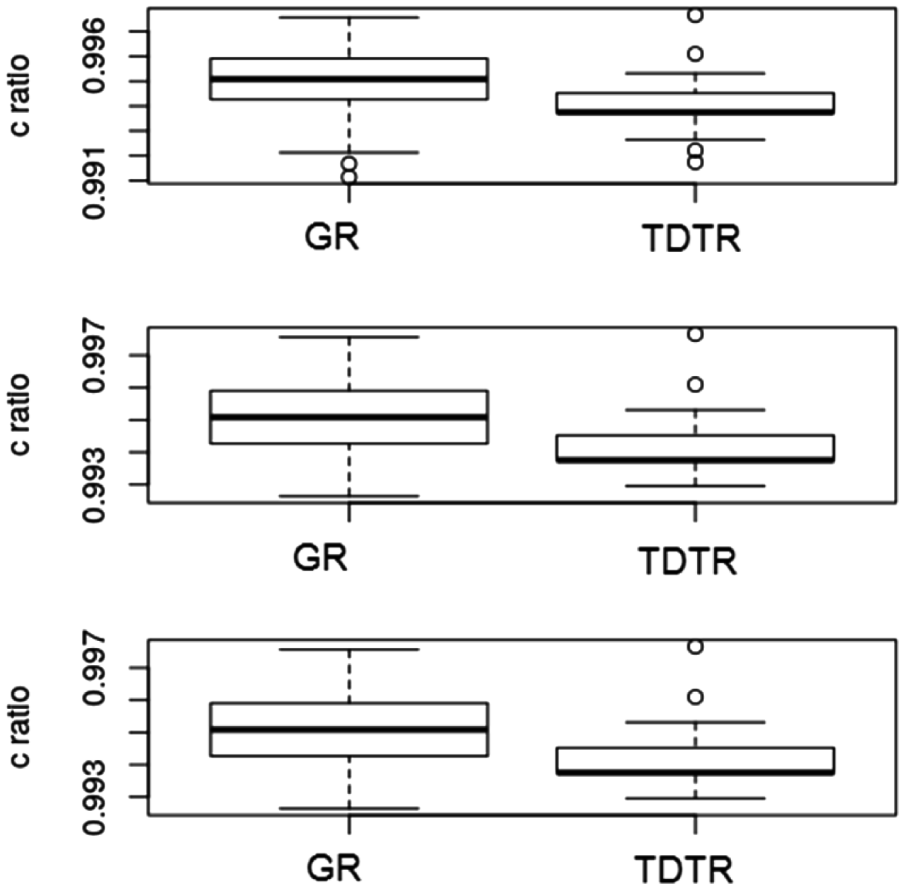

Fig. 2. P-value chart and box diagrams for the compression ratio metric

As third experiment, the evaluation of the compression ratio of the algorithm developed with statistical tests is proposed. The performance of the lossless compression algorithms Brotli, bzip2, gzip, xz is compared with the algorithm developed up to the simplification stage. This experiment is designed to determine which lossless compression algorithm is best suited to the solution. Four observations with the following characteristics are selected:

- Observation 1: three hundred trajectories, each one containing between 300 and 2500 points.
- Observation 2: three hundred trajectories, each containing between 310 and 4000 points.
- Observation 3: three hundred trajectories, each one containing between 320 and 2000 points.
- Observation 4: three hundred trajectories, each one containing between 360 and 2000 points.

The p-values obtained by the compression ratio metric to check the assumption of normality is 2.2e−16. This evidence that they do not fit a normal distribution therefore Ho is rejected. The Kruscal-Wallis test is then applied to compare several independent groups that do not fit to a normal distribution. The p-values obtained are less than 0.05, which means that there are significant differences according to the test applied with 95% of confidence. The median values obtained are higher for the GR-B algorithm so it can be concluded that has the best compression ratio.

5 Conclusions and Future Work

The study of noise reduction, line simplification and semantics in GPS trajectory compression allowed the foundation of a GPS trajectory compression algorithm that improves the compression ratio compared to those present in the literature. The comparison of the main algorithms for GPS trajectory compression demonstrated that TD-TR has the highest compression ratio in the experimental data set used and is therefore used as the basis for the simplification of points in the developed algorithm. The aim of the research is to increase the compression ratio while maintaining the margin of error, which is demonstrated by the results obtained from the processed data. These results are validated by means of statistical tests. The performed experiments show that the proposed GPS trajectory compression algorithm shows a greater compression ratio compared to similar ones analyzed in the literature, which reduces the amount of data to be processed. As you can see, the proposed algorithm compresses the GPS trajectory data in a significant way. As future work, it is planned to perform these experiments on other types of trajectories with a lower level of data redundancy and make the necessary adjustments to the algorithm to maintain the results achieved.

References

1. Chen, M., Xu, M., Fränti, P.: A fast O(N) multi-resolution polygonal approximation algorithm for GPS trajectory simplification. IEEE Trans. Image Process. 1–14 (2012)
2. Wang, T.: An online data compression algorithm for trajectories. Int. J. Inf. Educ. Technol. **3** (4), 480–487 (2013)
3. Stacchini, J.C., Lessa, T., Pal, B.: Data compression in smart distribution systems via singular value decomposition. IEEE Trans. Smart Grid **8**(1), 275–284 (2017)
4. Corcoran, P., Mooney, P., Huang, G.: Unsupervised trajectory compression. In: IEEE International Conference on Robotics and Automation (ICRA), pp. 3126–3132 (2016)
5. Ji, Y., Liu, H., Liu, X., Ding, Y., Luo, W.: A comparison of road-network-constrained trajectory compression methods. In: IEEE 22nd International Conference on Parallel and Distributed Systems (2016)
6. Muckell, J., Olsen, P.W., Lawson, C., Ravi, S., Hwang, J.: Compression of trajectory data: a comprehensive evaluation and new approach. Geoinformatica **2014**, 435–460 (2014)
7. Salomon, D.: Data Compression, 4th edn. Springer, London (2007). https://doi.org/10.1007/978-1-84628-603-2
8. Gudmundsson, J., Katajainen, J., Merrick, D., Ong, C., Wolle, T.: Compressing spatio-temporal trajectories. Comput. Geom. Theory Appl. **42**(9), 825–841 (2009)
9. Lv, C., Chen, F., Xu, Y., Song, J., Lv, P.: A trajectory compression algorithm based on non-uniform quantization. In: 12th International Conference on Fuzzy Systems and Knowledge Discovery (FSKD), pp. 2469–2474 (2015)
10. Van Hunnik, R.: Extensive comparison of trajectory simplification algorithms. University Utrecht (2017)
11. Asif, M.T., Kannan, S., Dauwels, J., Jaillet, P.: Data compression techniques for urban traffic data. In: 2013 IEEE Symposium on Computational Intelligence in Vehicles and Transportation Systems (CIVTS), pp. 4–9 (2013)
12. Meratnia, N., de By, R.A.: Spatiotemporal compression techniques for moving point objects. In: Bertino, E., Christodoulakis, S., Plexousakis, D., Christophides, V., Koubarakis, M., Böhm, K., Ferrari, E. (eds.) EDBT 2004. LNCS, vol. 2992, pp. 765–782. Springer, Heidelberg (2004). https://doi.org/10.1007/978-3-540-24741-8_44
13. Lawson, C., Ravi, S., Hwang, J.-H.: Compression and mining of GPS trace data: new techniques and applications, New York (2011)
14. Zhilin, L.: An algorithm for compressing digital contour data. Cartogr. J. **25**, 143–146 (1998)
15. Rhind, D.W.: Generalisation and realism within automated cartographic systems. Can. Cartogr. **10**(1), 51–62 (1973)
16. McMaster, R., Shea, K.S.: Generalization in Digital Cartography. Association of American Geographers, Washington, D.C. (1992)
17. Sim, M., Kwak, J.-H., Lee, C.-H.: Fast shape matching algorithm based on the improved Douglas-Peucker algorithm. KIPS Trans. Softw. Data Eng. **5**(10), 497–502 (2016)
18. Wu, S., Silva, A.C.G., Márquez, M.R.G.: The Douglas-Peucker algorithm: sufficiency conditions for non-self-intersections. J. Brazilian Comput. Soc. **9**, 1–17 (2004)
19. Lin, X., Ma, S., Zhang, H., Wo, T., Huai, J.: One-pass error bounded trajectory simplification. In: 43rd International Conference on Very Large Data Bases (VLDB), pp. 841–852 (2017)
20. Wang, H.: SharkDB : an in-memory storage system for large scale trajectory data management. The University of Queensland (2016)
21. Visvalingam, M., Whyatt, J.D.: Line generalisation by repeated elimination of the smallest area. Cartographic Information Systems Research Group, July 1992

22. Koegel, M., Mauve, M., Baselt, D., Scheuermann, B.: A comparison of vehicular trajectory encoding techniques. In: The 10th IFIP Annual Mediterranean Ad Hoc Networking Workshop, pp. 87–94 (2011)
23. Zhang, S., Liu, Z., Cai, Y., Wu, Z., Shi, G.: AIS trajectories simplification and threshold determination. J. Navig. **2016**, 729–744 (2016)
24. Hershberger, J., Snoeyink, J.: Speeding up the Douglas-Peucker line-simplification algorithm (1992)
25. Bianco, J.: Estudio y aplicación de Filtros de Kalman sobre sistemas de posicionamiento global para el suavizado de trayectorias geoposicionadas. Universidad Nacional de Córdoba (2013)
26. Cegan, L.: Empirical study on effects of compression algorithms in web environment. J. Telecommun. Electron. Comput. Eng. **9**(2), 69–72 (2017)
27. Alakuijala, J., Kliuchnikov, E., Szabadka, Z., Vandevenne, L.: Comparison of Brotli, Deflate, Zopfli, LZMA, LZHAM and Bzip2 compression algorithms (2015)
28. Matejek, B., Haehn, D., Lekschas, F., Mitzenmacher, M., Pfister, H.: Compresso: efficient compression of segmentation data for connectomics. In: Descoteaux, M., Maier-Hein, L., Franz, A., Jannin, P., Collins, D.L., Duchesne, S. (eds.) MICCAI 2017. LNCS, vol. 10433, pp. 781–788. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-66182-7_89