GeoSecure: Towards Secure Outsourcing of GPS Data over Cloud

Vikram Patil, Shivam Parikh, Priyanka Singh and Pradeep K. Atrey
Department of Computer Science, College of Engineering and Applied Sciences
University at Albany, State University of New York
Albany, NY, USA
Email: {vpatil, sparikh, psingh9, patrey}@albany.edu

***Abstract*—Today geolocation data is used extensively in multiple applications and devices. GPS trajectory data can reveal political, religious affiliations, personal habits, shopping preferences etc. It threatens large number of users who use location-based services on their devices, because they are afraid of revealing their locations and concerned about being tracked. Multiple approaches have been proposed to solve problems related with trajectory data such as encrypting geolocation data, decoupling from users' unique identifiable data using privacy algorithms, and storing the data using compression algorithms. Generally, Location Based Service (LBS) providers must perform these operations sequentially on data which turns out to be inefficient. In this paper, we propose a novel approach to resolve the issues of high data storage cost, security, and privacy threats all at once. Operations on encoded GPS trajectory data are performed using modified delta compression and the Haversine distance in a lossless and privacy ensured way. It can be used to calculate velocity, acceleration, distance, etc. without actually revealing location of the user. the cloud storage cost of the GPS data is reduced using modified delta compression.**

I. Introduction

Use of GPS-enabled devices such as smartphones, fitness trackers, and IoT devices is rapidly increasing. These devices leverage location based services like Google Maps, which searches for nearby or far off locations. The aggregate amount of GPS data generated daily by these devices is increasing exponentially. With the advent of cloud computing, huge amounts of computing power and storage space have become very affordable. It is now possible to create large, scalable applications in the cloud for a fraction of the cost of provisioning and deploying on premise infrastructure [1]. Furthermore, hardware maintenance costs and energy costs are included in cloud fees charged by cloud service providers (CSPs).

CSPs focus on providing convenience and faster roll out of services to their customers, which sometimes comes at the cost of not ensuring privacy and security of user-generated data. In addition, privacy concerns regarding user data is an issue that is growing in importance, and questions are being raised about potential misuse of user data and tracking user activity without user notification. Today, there are an estimated 1 billion users of smartphones, and it is predicted that in the near future, another billion smartphone users will be added. Hence, the growth in user generated data and possibility of private data being used without users' consent presents us with following three main issues that are associated with the outsourcing of GPS data over CSPs:

1) *How to compress GPS data*: Although CSPs offer storage services at a very cheap rate, the amount of data that is generated everyday is growing exponentially amounting to more storage cost. GPS devices trajectories, transactions, back-ups, etc. are consuming storage space and are demanding high network bandwidth. Hence, there is a need for lossless way to compress this data with optimized storage and bandwidth cost.
2) *How to preserve users' privacy*: Privacy of users' personal data is the prime reason behind users' not trusting and not willing to give their data to applications and companies that provide cloud based services (e.g. people turn off location services for various applications like Facebook, Whatsapp, Snapchat, etc). It demands the creation of privacy aware solutions, which can store users' data without revealing their actual location. Privacy and usability of the data are often inversely proportional. If we make data more anonymous, it will provide higher degree of privacy but at the same time reduce its utility for practical applications.
3) *How to secure GPS data*: GPS data is very sensitive data, since it can be misused for surveillance and for tracking of an individual user. Recently, Amazon Web Services (AWS) faced a DDoS attack and they were down for hours. Such incidents emphasize the need for securing the GPS data over CSPs. Although the encryption of content prior to outsourcing to a CSP increases security, but then the maintenance of encryption key with increasing data is an overhead for users. Moreover, encryption of GPS data makes it unsuitable for many location -based services. Hence, a new approach to secure GPS data from potential security threats is needed.

Based on the above discussion, the intriguing research question is that without much loss in the utility of GPS data, how to: i) *compress* the data in order to minimize the storage cost over CSP, ii) maintain the *confidentiality* of the data over cloud and during transfer between users' smartphone device and the cloud, and iii) ensure that *privacy* of users' data is not invaded. The focus of this paper is essentially to address this research question.

The main challenge is to come up with a single approach that solves all the above mentioned challenges since there exists a trade-off between utility of the data and the above three goals, i.e. compression, confidentiality and privacy. While existing solutions like encryption for security, compression for storage cost reduction, and anonymization for privacy are better suited for solving one specific problem. Applying one of these existing solutions creates a negative impact on other two challenges.

In this paper, we present an approach called "GeoSecure" that uses a delta compression based strategy to not only compress the GPS data, but also maintain its confidentiality and privacy. Moreover, the proposed approach preserves the utility of the data by making it suitable for providing location-based services without revealing users' location or compromising confidentiality.

The core idea behind our approach is to keep the first GPS coordinate (latitude and longitude) at the user's smartphone device and outsource the differentials (i.e. difference between the first and the second coordinates, and so on) to the CSP. The CSP computes the Haversine distance using these differentials and uses it to determine different parameters like distance traveled, velocity and acceleration of users without actually revealing their location. These parameters are used to provide various location-based services such as distance traveled with modes of transportation (i.e. how long a user walks, runs, cycles or drives everyday – generally used in apps like Fitbit that tracks activity and records workouts), traffic congestion (i.e. how long a user was stuck in traffic on given days), and co-relating trajectories of multiple people (i.e. how many people have similar activities and workouts).

The rest of the paper is organized as follows: Section II discusses the related work and section III presents the proposed approach. Section IV gives detailed implementation, application and results. Conclusions along with future work are discussed in section V.

II. Related Work

In the last decade, many techniques have been proposed to solve some of the problems encountered with processing of GPS trajectories and using them for various applications. The paper by Zheng is a detailed survey paper which covers all the major areas and summarizes related work about processing GPS trajectories such as trajectory mining, pattern matching, compression, prediction, privacy etc [2]. There have been a few solutions proposed to address the problems of storage cost reduction (compression), security (confidentiality), and privacy (identity protection), but they only achieve success by solving one challenge as shown in Table I.

The paper by Seidl et al. compared different approaches to preserve privacy in GPS trajectory data [3]. There are many privacy-related algorithms, like *k*-anonymity, that deal with creating privacy aware data sets by decoupling user data from a specific user [4]. This approach is widely used for anonymizing datasets, but it is vulnerable to multiple privacy attacks. The authors in [5] introduced improvements to *k*-anonymity called (k, δ)-anonymity. Subsequently, Gao et al.

TABLE I: A comparison of existing approaches with proposed work

Existing Approaches For	Achieves Compression?	Confidentiality?	Privacy?
Confidentiality	No	Yes	Yes/No
Privacy	No	No	Yes
Compression	Yes	No	No
Proposed Work	**Yes**	**Yes**	**Yes**

[6] used *k*-anonymity for GPS trajectories, making set of trajectories unidentifiable. Further, the paper by Xiong et al. [7] elaborates some of the limitations of *k*-anonymity.

There are multiple techniques like obfuscation and masking, such as donought masking and affine transformation, which have been discussed at length in [3], [7], [8], [9], [10],[11]. In [12], the concept of privacy-by-design has been explored, which gives a framework for securely outsourcing the data to the CSP. In [13], the authors propose a scheme to securely provide LBS without tracking users, by using obfuscation functions and merkel trees.

Still many of these techniques are based on data minimizing, introduction of random noise, misguiding attackers etc. These techniques are useful in anonymizing data sets but we can only use them for limited applications. Most of them are also batch processing algorithms (i.e. offline algorithms) which require all the data set for performing the operations. So we need a new approach which can be used in online as well as offline fashion and which is lossless (we can retrieve back original information). In addition to it, we can also use it for compression, encrypted domain applications, and providing different data sharing options for users so that based on their trust in the service provider, they can choose how much data to share and in what format.

Douglous Peucker algorithm is one of the most popular algorithms for GPS trajectory compression [14]. This algorithm tries to fit line segments to the trajectory and thus reduces number of points required to represent a trajectory. Subsequently, algorithms like STTrace are prediction based algorithms which utilize temporal information and try to reduce Synchronized Euclidean Distance (SED) error [15]. Furthermore, the work [16] and [17] by Muckell et al. summarized approaches for GPS trajectory compression and introduced new algorithms SQUISH and SQUISH-E.

LempelZivWelch (LZW) is a dictionary-based approach which takes advantage of repeated strings in the data, representing them with a string pattern so achieving compression [18], [19]. On the other hand, delta compression relies on difference between two successive points, though both LZW and delta compression algorithms have run time complexity of $O(n)$.

Delta encoding is a technique in which we express series of data points as differences between consecutive points. Cudre-Mauroux et al. [20] used delta compression for storage of GPS trajectories and also explores it for comparison and queries. They compared trajectories and removed the redundant parts in trajectories. In this work, the authors also used fixed-point arithmetic scheme, in which a fixed

TABLE II: A comparative analysis of proposed work with the existing approaches

Work	Privacy?	Confide -ntiality?	Comp -ression?	Utility*?
Douglous Peucker [14]	No	No	Yes	Yes
SQUISH [16]	No	No	Yes	Yes
LZW based [19]	No	No	Yes	No
k-anonymity [4]	Yes	No	No	No
(k, δ)-anonymity [5]	Yes	No	No	No
Obfuscation and masking [3]	Yes	No	No	Yes (limited)
Trajstore [20], Trajic [21]	No	No	Yes	Yes
Homomorphic encryption[23]	Yes	Yes	No	Yes
Proposed method	**Yes**	**Yes**	**Yes**	**Yes**

*Here, utility refers to the availability of location-based service for a particular user.

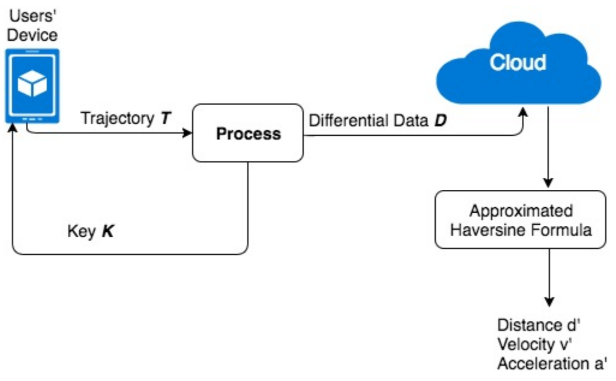

Fig. 1: GeoSecure Workflow

number multiplies every point so that the differences are in real numbers instead of decimals. They also claim that, their approach can achieve 25% of compress ratio, which is way better than rest of the approaches. However, their focus was on querying the data, so they do not address privacy or security concerns. Furthermore, in a recent work, Nibali and He [21] proposed a system called "Trajic" that achieved better compression by combining delta compression with predictor function.

Security of the data refers to protecting it from unauthorized access. From the GPS data security perspective, a new approach for secure indexing targeted toward social crowdsourcing was proposed by Liu et al. [22]. In this work, the authors used homomorphic encryption and SKD trees. Another paper by Liu et al [23] explored similarity computation of encrypted trajectory data using Paillier cryptosystem. Although homomorphic encryption can be the one way of performing computations on GPS trajectory data, it is often slower and it offers only limited operations.

Table II summarizes the key differences of the proposed work with the existing approaches.

III. Proposed Method

In this section, first we define the terms compression, confidentiality, privacy and utility in the context trajectory data. Next, we present the proposed algorithm, and finally, we discuss the threat model.

A. Definitions

Definition 1: (Trajectory Compression) A trajectory T can be said to be compressed if it can be represented by a transformed trajectory T' such that $T' = f(T)$ and $|T'| < |T|$, where f is a transformation function and $||$ represents the size.

Definition 2: (Trajectory Confidentiality) The confidentiality of the trajectory T can be said to be maintained if the identity of the associated user in the transformed trajectory T' is not revealed to the adversary.

Definition 3: (Trajectory Privacy) The privacy of a trajectory T is said to be preserved if an attacker/ adversary is unable to find the exact location of the user in the transformed trajectory T'.

Definition 4: (Trajectory Utility) The trajectory utility $U_{T,S}$ for a given location-based service S is calculated as:

$$U_{T,S} = \frac{|\phi_{T',S}|}{|\phi_{T,S}|} \tag{1}$$

where $|\phi_{T,S}|$ and $|\phi_{T',S}|$ are accuracies by which a given service S is accomplished using trajectories T and T', respectively.

B. Proposed Algorithm

Figure 1 illustrates the workflow of the proposed GeoSecure method. The proposed algorithm takes a GPS trajectory T consisting of points P_1 to P_n as input. Each point consists of three properties like latitude, longitude and time. Here, we illustrate the proposed algorithm with an example on a GPS trace file from the Microsofts GeoLife dataset.

Step 1: Select the primary fields from the GPS trajectory that contains multiple fields such as latitude, longitude, time, elevation, etc. Here, we have considered latitude, longitude, date and time from the file of GeoLife dataset (Columns 1,2,6,7) and represented the GPS trajectory as matrix T as follows:

$$\begin{pmatrix} 26.898747 & 112.591398 & 2009-10-02 & 06:02:28 \\ 26.898631 & 112.591448 & 2009-10-02 & 06:02:30 \\ 26.89874 & 112.591413 & 2009-10-02 & 06:02:31 \\ 26.898493 & 112.591428 & 2009-10-02 & 06:02:34 \\ 26.897625 & 112.591361 & 2009-10-02 & 06:02:37 \\ \vdots & \vdots & \vdots & \vdots \end{pmatrix}$$

Step 2: Since there are six digits after decimal point, for efficient delta compression, we need to multiply each number by 10^6. Then the calculated differences between consecutive points will be in integers represented as follows:

$$\begin{pmatrix} 26898747 & 112591398 & 2009-10-02 & 06:02:28 \\ 26898631 & 112591448 & 2009-10-02 & 06:02:30 \\ 26898740 & 112591413 & 2009-10-02 & 06:02:31 \\ 26898493 & 112591428 & 2009-10-02 & 06:02:34 \\ 26897625 & 112591361 & 2009-10-02 & 06:02:37 \\ \vdots & \vdots & \vdots & \vdots \end{pmatrix}$$

Step 3: Differences are computed from second point onward with respect to its predecessor point as follows:

$$\begin{pmatrix} 26898747 & 112591398 & 2009-10-02 & 06:02:28 \\ -116 & 50 & 0 & 2 \\ 109 & -35 & 0 & 1 \\ -247 & 15 & 0 & 3 \\ -868 & -67 & 0 & 3 \\ \vdots & \vdots & \vdots & \vdots \end{pmatrix}$$

Step 4: First row of the matrix is taken as the key K and rest of the matrix as differences matrix D.

$$K = \begin{pmatrix} 26898747 & 112591398 & 2009-10-02 & 06:02:28 \end{pmatrix}$$

$$D = \begin{pmatrix} -116 & 50 & 0 & 2 \\ 109 & -35 & 0 & 1 \\ -247 & 15 & 0 & 3 \\ -868 & -67 & 0 & 3 \\ \vdots & \vdots & \vdots & \vdots \end{pmatrix}$$

Step 4 : Assign a random number to the trajectory called *trajectoryID* and save the tuple $T($*trajectoryID*$,\ K)$ to the users device. Save the matrix D with *trajectoryID* and send it to the cloud.

Now if the user wants the original trajectory data back, it can use following steps to retrieve the data from the differences and key of the file.

1) Locate the key and the corresponding data file.
2) If the file is encrypted then decrypt it.
3) In the data file, multiply every entry of latitude and longitude by 10^6.
4) Add first entry in latitude to the latitude in the key and repeat this for longitude.
5) Repeat this process for the entire file.

C. Threat Model and Analysis

We analyze the scenario of transferring trajectory T to the CSP, by considering the following threat model.

There are three entities in this model: user, CSP and adversary. While a user is considered honest, CSP is semi-malicious i.e. the CSP will honestly provide the services, but it is curious about the users' data outsourced to the cloud. The adversary is usually an external entity which tries to attack user, CSP and the communication channel between them and attempts to know users' data. The communication between users' device and the CSP is assumed to be secure. Generally the protocols like HTTPS are used for cloud communication which encrypt data end to end.

We have analyzed the model with respect to following threats:

- *Man in the middle attack*: In this attack, hackers try to eavesdrop on the public channel to gain access to user's data before it is received at the cloud. In our proposed solution the trajectory T is divided into $[K, D]$ where K is the key and D is the difference matrix. Only the difference D is sent to the cloud. Now, to be successful at determining exact location of the user, the adversary will also need key K. Since the key never leaves users device, it is very unlikely to calculate the exact location. Thus we can say that trajectory privacy is maintained in our proposed solution and thus, it prevents the man in the middle attack.
- *Linkage attack*: In this attack, adversary will analyze data on social media platforms such as Facebook, twitter, etc. about the user and combine that information with user's data. In case of GPS data, user's geotagged images and check on social media platform provides time and location of the user for one particular instance. If the adversary gets access to such data, then it might be possible to find those locations and timestamp in the data and link it with the user to get more information about the user. The proposed method prevents this attack since the starting location of the user is never disclosed to the CSP, so it becomes unlikely to match the difference data with the social media data. Hence, the proposed method prevents the linkage attack.
  Thus, the proposed solution maintains trajectory privacy and confidentiality.

IV. Implementation, Application and Results

To validate the proposed method, we have tested using Microsoft Geolife dataset [24],[25]. This dataset is publically available and consists of thousands of GPS trajectories collected by 182 Microsoft users over period of 3 years in Beijing, China. We ran our algorithm on 100 GPS trajectories. When a user starts using our algorithm, the device will record the current latitude, longitude and time, and will save it as first GPS point, we refer it as Key K. As described in Figure 1, the user will then use the proposed algorithm for next points and create differences D with respect to key K. Key K will be stored on users' device and the differences D are directly sent to the CSP such as AWS. User can choose to send the data in real time or in batch mode.

Third party service providers like fitness trackers, maps etc. will have access to this data. They can use it for determining velocity, acceleration and other services for the user, without actually knowing users' current location. Whenever a user wants to get the original data back, he can request the data from CSP and then decompress using the key K stored with him on his device. In this way, the user shared the data with the CSP in a secured way without revealing the actual location of the user and can receive the data back and retrieve original trajectory using the decompression algorithm.

The original trajectory T is expressed in terms of differences between consecutive points. So the same trajectory is saved in less space. We were able to compress file 20091002060228.plt from Geolife dataset from 548kb into 155kb using our algorithm, thus achieving compression and space saving of 72%. Thus, the proposed solution provides trajectory compression as per *Definition* 1.

Using the proposed approach even after compression, just by looking at differences between consecutive latitude, longitude co-ordinates and time, the CSP can calculate velocity and acceleration. Hence, the proposed method allows the CSP to provide location-based services to the user without revealing the actual user location. Thus, the proposed solution provides trajectory confidentiality and privacy as per *Definition* 2 and *Definition* 3.

The details of how the traditional method works and the proposed method enhances it without knowing the actual coordinates is detailed as follows:

A. Calculating Distance Traveled, Velocity and Acceleration

1) Using traditional method: To calculate distance based on latitude and longitude, the famous Haversine formula is widely used [26], e.g. in navigation and astronomy applications. For any two given points $P_1(lat_1, long_1)$, $P_2(lat_2, long_2)$ and R being the radius of the earth (mean radius = 6,371 km), the distance d between points P_1 and P_2 can be calculated as follows:

$$dlon = long_2 - long_1$$

$$dlat = lat_2 - lat_1$$

$$a = (\sin(\frac{dlat}{2}))^2 + \cos(lat1) \times \cos(lat2) \times (\sin(\frac{dlon}{2}))^2$$

$$c = 2 \times atan2(\sqrt{a}, \sqrt{1-a})$$

$$d = R \times c \tag{2}$$

Once we know the distance, we can calculate velocity and acceleration.

2) Using proposed method: In equation (2), we have two terms $\cos(lat1)$ and $\cos(lat2)$. Since in few seconds, relative change in the latitudes of consecutive points is very small, we can say that their multiplication will yield very small result. We did large number of approximations and we can safely say that this term can be approximated to 1. This might introduce small error of order of 0.1 m for distance calculation using the Haversine formula, which is smaller than average GPS error. For example, let's try to approximate above formula for two points P_1 and P_2 which are very close to each other, $P_1(26.898747, 112.591398)$ and $P_2(26.898631, 112.591448)$. The distance calculated by actual formula is 0.0138km and the distance by approximating it for two very close points is 0.0140 km. So the error between actual distance and calculated distance is 0.0002 km = 0.1 meter. So we can use this approximation to calculate distance.

3) Calculating error range: Let the original trajectory $T = \{P_1, P_2, \ldots, P_n\}$ be set of points from P_1 to P_n. Then actual distance d between two successive points can be calculated using equation (2) as $d = R \times c$.

Now, let's assume the transformed trajectory be denoted as $T' = \{P_1, D_1, D_2, \ldots, D_{n-1}\}$; which contains first point P_1 and the differences $D_1, D_2, \ldots, D_{n-1}$. Then the approximated distance d' between two successive points in T' can be calculated as:

$$a' = (\sin(\frac{dlat}{2}))^2 + (\sin(\frac{dlon}{2}))^2$$

$$c' = 2 \times atan2(\sqrt{a'}, \sqrt{1-a'})$$

$$d' = R \times c' \tag{3}$$

Now, let ϵ ($0 \leq \epsilon \leq \epsilon_{max}$) be the error between d and d', where ϵ_{max} is maximum possible error between actual and calculated distance (d and d', respectively) between two successive points. Precisely, ϵ can be calculated as follows:

$$\begin{aligned}
\epsilon_{max} &= |d - d'| \\
&= R \times c - R \times c' \\
&= R \times (c - c') \\
&= R(2 \times atan2(\sqrt{a}, \sqrt{1-a}) - 2 \times atan2(\sqrt{a'}, \\
&\qquad \sqrt{1-a'})) \\
&= 2R(atan2(\sqrt{a}, \sqrt{1-a}) - atan2(\sqrt{a'}, \sqrt{1-a'})) \\
&= 2R \times atan2(\sqrt{a} \times \sqrt{1-a'} - \sqrt{a'} \times \sqrt{1-a'}, \\
&\qquad \sqrt{1-a} \times \sqrt{1-a'} + \sqrt{a} \times \sqrt{a'})
\end{aligned}$$

B. Experimental Results for Distance, Velocity and Acceleration

The proposed method has been tested on file 20091002060228.plt which is part of Microsoft Geolife dataset. As mentioned in the steps of our algorithm, we considered latitude, longitude and time components only from this file. Then, difference is calculated for latitude and longitude using the actual Heaversine (equation (2)) and approximated Heaversine formula (equation (3)).

Figure 2 shows a graph in which, trajectory points P_1 to P_n in chronological order with P_1 as the starting point and P_n as the last point are considered. The actual distance between two consecutive points in a trajectory is compared with the calculated distance using the proposed approximation formula. The total distance traveled for the complete trajectory consisting of 13,432 GPS data points, calculated by traditional method is 1.9936×10^5 meters and by proposed methos is 2.1506×10^5 meters. By observing Figure 2, we can say that the actual and calculated distances using the traditional method and the proposed method is almost same. Also in Figure3, the absolute error between actual and calculated distances are computed. By observing Figure 3 , we can infer that the error is very small which again validates the proposed solution.

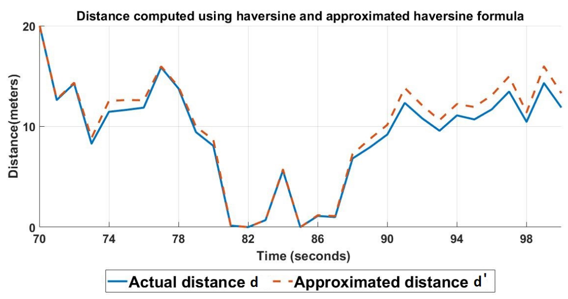

Fig. 2: Analysis for sample points of GeoLife Dataset distance comparison

Similarly the actual velocity and acceleration between two consecutive points in a trajectory is compared with

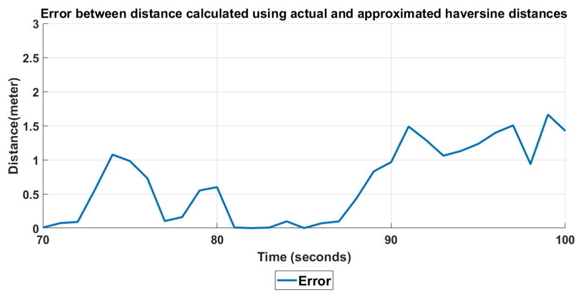

Fig. 3: Analysis for sample points of GeoLife Dataset distance Error

the calculated velocity and acceleration using the proposed approximation formula as depicted in Figure 4 and Figure 6. The average velocity computed for the same trajectory, calculated by traditional method is 12.5915 meters/sec and by proposed methos is 13.5830 meters/sec whereas the acceleration comes out to be 11.6046 meter/sec^2 and 12.5334 meter/sec^2 using the traditional and the proposed method.

Also, the absolute error between the velocity and acceleration are computed which are depicted in Figure 5 and Figure 7. From the figures we can observe that the error is very small for velocity as well as acceleration which validates the proposed method. Thus, the proposed solution provides trajectory utility as per *Definition* 4.

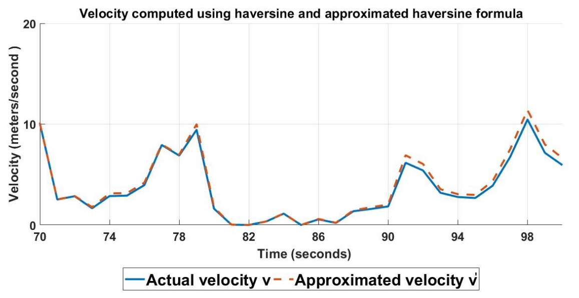

Fig. 4: Analysis for sample points of GeoLife Dataset velocity comparison

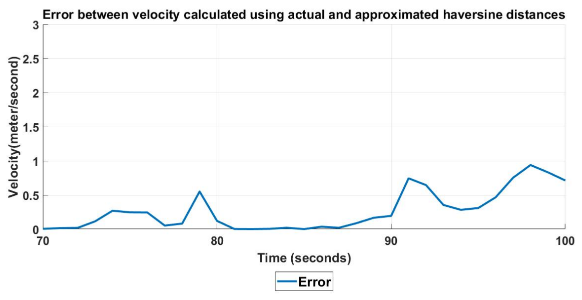

Fig. 5: Analysis for sample points of GeoLife Dataset Velocity error

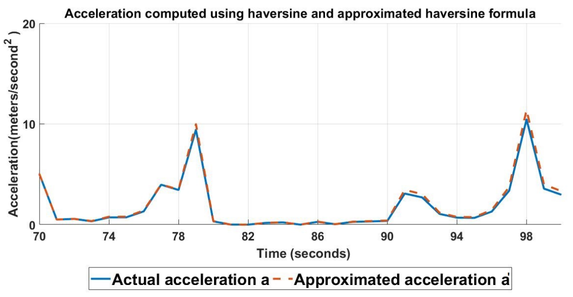

Fig. 6: Analysis for sample points of GeoLife Dataset acceleration comparison

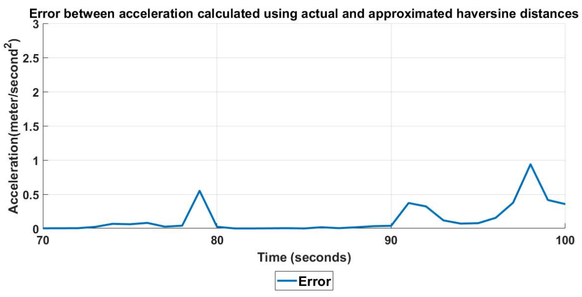

Fig. 7: Analysis for sample points of GeoLife Dataset acceleration error

C. Other Possible Location-based Services

Some other utilities which can be provided using the proposed method are discussed as follows:

1) *Traffic congestion*: By determining anomalies in the velocity, acceleration comparison with other trajectories, the CSP will be able to find out, how long the user was stuck in traffic.
2) *Mode detection*: By detecting acceleration and velocity, by proposed method, we can find patterns and determine mode of transportation like walking, cycling, driving etc. This is one of the very important area in Geographical Infomation Systems (GIS) and can be used in multiple applications such as fitness trackers.
3) *Finding stop points*: This use case is highly studied area since it gives us information about where user is stopping. This can be used to provide services such as highest visited locations, co-relating trajectories of multiple people etc.
4) *Creating fingerprint functions* (these yield same result for similar trajectories vs hash which gives same result for exactly same trajectories): This is useful for comparison of multiple trajectories which are similar. If a user is taking exactly same path everyday, still the trajectory can be slightly different each day since the GPS co-ordinates will be few meters away or because of errors in GPS devices. This type of fingerprint functions can help us in reducing redundant trajectories and allowing lateral compression.

V. Conclusion and Future Work

We discussed the primary challenges in moving GPS data into cloud that are raising issues related to storage cost, privacy of users data, and security of the data stored in the cloud. We also discussed shortcomings of the existing approaches that werent able to cater all the challenges at once. Therefore, we provided single novel solution that addresses all the challenges. Based on our results, we were able to demonstrate that proposed method can be used to provide location-based service without revealing users' location. Thus, making it privacy aware and at the same time makes the GPS data secure while reducing the storage cost. We also discussed applications of this method for processing data in encoded domain without revealing the actual location. This method will motivate users to trust service providers and avail more service options without compromising their privacy and confidentiality. In future, we plan to implement the applications discussed in the paper and release an open source API for the CSP as well as detecting traffic congestion in secure domain.

References

[1] "Amazon web services whitepaper," https://d0.awsstatic.com/whitepapers/aws-overview.pdf.

[2] Y. Zheng, "Trajectory data mining: An overview," *ACM Transactions on Intelligent Systems and Technology*, vol. 6, no. 3, pp. 1–29, May 2015.

[3] D. E. Seidl, P. Jankowski, and M.-H. Tsou, "Privacy and spatial pattern preservation in masked GPS trajectory data," *International Journal of Geographical Information Science*, vol. 30, no. 4, pp. 785–800, 2016.

[4] L. Sweeney, "k-anonymity: A model for protecting privacy," *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, vol. 10, no. 05, pp. 557–570, 2002.

[5] O. Abul, F. Bonchi, and M. Nanni, "Never walk alone: Uncertainty for anonymity in moving objects databases," in *International Conference on Data Engineering (ICDE '08)*. IEEE, 2008, pp. 376–385.

[6] S. Gao, J. Ma, C. Sun, and X. Li, "Balancing trajectory privacy and data utility using a personalized anonymization model," *Journal of Network and Computer Applications*, vol. 38, pp. 125–134, 2014.

[7] P. Xiong, T. Zhu, W. Niu, and G. Li, "A differentially private algorithm for location data release," *Knowledge and Information Systems*, vol. 47, no. 3, pp. 647–669, 2016.

[8] P. Singh, B. Raman, N. Agarwal, and P. K. Atrey, "Secure Cloud-Based Image Tampering Detection and Localization Using POB Number System," *ACM Transactions on Multimedia Computing, Communications, and Applications*, vol. 13, pp. 1–23, 2017.

[9] P. Singh, B. Raman, and M. Misra, "A secure image sharing scheme based on svd and fractional fourier transform," *Signal Processing: Image Communication*, vol. 57, pp. 46–59, 2017.

[10] P. Singh and B. Raman, "Reversible data hiding for rightful ownership assertion of images in encrypted domain over cloud," *AEU-International Journal of Electronics and Communications*, vol. 76, pp. 18–35, 2017.

[11] B. Hoh, M. Gruteser, H. Xiong, and A. Alrabady, "Achieving guaranteed anonymity in gps traces via uncertainty-aware path cloaking," *IEEE Transactions on Mobile Computing*, vol. 9, no. 8, pp. 1089–1107, 2010.

[12] A. Monreale, S. Rinzivillo, F. Pratesi, F. Giannotti, and D. Pedreschi, "Privacy by design in big data analytics and social mining," *EPJ Data Science*, vol. 3, no. 1, pp. 1–26, 2014.

[13] R. Di Pietro, R. Mandati, and N. V. Verde, "Track me if you can: Transparent obfuscation for location based services," in *World of Wireless, Mobile and Multimedia Networks (WoWMoM)*. IEEE, 2013, pp. 1–9.

[14] D. H. Douglas 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.

[15] M. Potamias, K. Patroumpas, and T. Sellis, "Sampling trajectory streams with spatiotemporal criteria," in *International Conference on Scientific and Statistical Database Management*. IEEE, 2006, pp. 275–284.

[16] J. Muckell, P. W. Olsen, J. H. Hwang, C. T. Lawson, and S. S. Ravi, "Compression of trajectory data: a comprehensive evaluation and new approach," *GeoInformatica*, vol. 18, no. 3, pp. 435–460, 2014.

[17] J. Muckell, J. H. Hwang, V. Patil, C. T. Lawson, F. Ping, and S. S. Ravi, "SQUISH: An Online Approach for GPS Trajectory Compression," in *International Conference on Computing for Geospatial Research and Applications*, vol. 13. ACM, 2011, pp. 1–8.

[18] R. N. Horspool, "Improving LZW (data compression algorithm)," in *Data Compression Conference*, Apr 1991, pp. 332–341.

[19] D. W. Xu, Y. D. Wang, L. M. Jia, G. J. Zhang, and H. F. Guo, "Compression Algorithm of Road Traffic Spatial Data Based on LZW Encoding," *Journal of Advanced Transportation*, 2017.

[20] P. Cudre-Mauroux, E. Wu, and S. Madden, "TrajStore: An adaptive storage system for very large trajectory data sets," in *International Conference on Data Engineering (ICDE 2010)*, 2010, pp. 109–120.

[21] A. Nibali and Z. He, "Trajic: An effective compression system for trajectory data," *IEEE Transactions on Knowledge and Data Engineering*, vol. 27, no. 11, pp. 3138–3151, 2015.

[22] B. Liu, L. Chen, X. Zhu, Y. Zhang, C. Zhang, and W. Qiu, "Protecting location privacy in spatial crowdsourcing using encrypted data," in *International Conference on Extending Database Technology (EDBT)*, 2017, pp. 478–481.

[23] A. Liu, K. Zhengy, L. Liz, G. Liu, L. Zhao, and X. Zhou, "Efficient secure similarity computation on encrypted trajectory data," in *International Conference on Data Engineering*, 2015, pp. 66–77.

[24] Y. Zheng, X. Xie, and W. Y. Ma, "Geolife: A collaborative social networking service among user, location and trajectory." *IEEE Data Engineering Bulltein*, vol. 33, no. 2, pp. 32–39, 2010.

[25] Y. Zheng, L. Zhang, X. Xie, and W. Y. Ma, "Mining interesting locations and travel sequences from GPS trajectories," in *International conference on World wide web*. ACM, 2009, pp. 791–800.

[26] B. Shumaker and R. Sinnott, "Astronomical computing: 1. Computing under the open sky. 2. Virtues of the haversine," *Sky and telescope*, vol. 68, pp. 158–159, 1984.