Encoding network-constrained travel trajectories using routing algorithms

Pablo Martinez Lerin*

Graduate School of Engineering,
Nagoya Institute of Technology,
Gokiso, Showa, Nagoya, Aichi 466-8555, Japan
Email: pablo@moss.elcom.nitech.ac.jp
*Corresponding author

Daisuke Yamamoto

Graduate School of Engineering,
Nagoya Institute of Technology,
Gokiso, Showa, Nagoya, Aichi 466-8555, Japan
and
Core Research for Evolutional Science and Technology,
Japan Science and Technology Agency,
K's Gobancho, 7, Gobancho,
Chiyoda-ku, Tokyo 102-0076, Japan
Email: yamamoto.daisuke@nitech.ac.jp

Naohisa Takahashi

Graduate School of Engineering,
Nagoya Institute of Technology,
Gokiso, Showa, Nagoya, Aichi 466-8555, Japan
Email: naohisa@nitech.ac.jp

Abstract: This study proposes a generic encoder for network-constrained travel trajectories, and it implements two encoders by combining the proposed generic encoder with two routing algorithms, which reduce the size of a travel trajectory's path along a road network without modifying it. Although most previous trajectory compression methods introduce an error in the spatial component of trajectories to achieve compression, we argue that the path of a travel trajectory, i.e. the sequence of roads travelled, is crucial information for many applications. Experimental results obtained using a large-data set of real travel trajectories show that the two proposed encoders achieve high compression, outperforming a general purpose data compression method.

Keywords: network-constrained trajectories; travel trajectories; trajectory encoding; lossless trajectory compression; shortest path; following path; routing algorithms; knowledge and web intelligence; map matching; GPS; GIS.

Reference to this paper should be made as follows: Martinez Lerin, P., Yamamoto, D. and Takahashi, N. (2013) 'Encoding network-constrained travel trajectories using routing algorithms', *Int. J. Knowledge and Web Intelligence*, Vol. 4, No. 1, pp.34–49.

Biographical notes: Pablo Martinez Lerin is a PhD student at the Computer Science Department in Nagoya Institute of Technology, Japan. He received BE and ME degrees in Computer Science in 2009 from the Polytechnic University of Valencia, Spain. His research interests include travel information analysis, web mapping, collaborative GIS systems, and user interface modelling and design.

Daisuke Yamamoto is an Associate Professor in the Information Technology Center at Nagoya Institute of Technology, Japan. He received a PhD in Information Science from Nagoya University. His research interests include web services, web interaction, content technologies, e-learning, GIS and multimedia systems.

Naohisa Takahashi is a Professor at the Department of Computer Science at Nagoya Institute of Technology since 2001. He received BE and ME degrees in Electrical Engineering from the University of Electro-Communications, Tokyo, Japan, in 1974 and 1976, respectively. He received a doctorate in Computer Science in 1987 from Tokyo Institute of Technology. His recent research interests are network computing, ubiquitous computing, geographical information systems and e-learning systems. He is a member of the IEEE, the Association for Computing Machinery, the Information Processing Society of Japan, the Japan Society for Software Science and Technology, and the Database Society in Japan.

This paper is a revised and expanded version of a paper entitled 'Encoding travel traces by using road networks and routing algorithms' presented at the '5th International Conference on Intelligent Interactive Multimedia Systems and Services', Gifu, Japan, 23–25 May 2012.

1 Introduction

Large amounts of network-constrained travel trajectories, i.e. the trajectories of travels along a road network, are required for applications such as transportation planning (Yuan et al., 2010a; Yuan et al., 2010b) and route prediction based in traffic information (Xue et al., 2009). Travel trajectories are easily generated in the form of a sequence of Global Positioning System (GPS) points by using a tracking system (e.g. a GPS logger) while travelling. However, collecting and storing large amounts of travel trajectories for future reference impose significant transmission and storage challenges.

Many works (Meratnia and de By, 2004; Hönle et al., 2010; Koegel et al., 2011) have proposed approaches to reducing the size of trajectories that smooth a trajectory's trace introducing a bounded error (e.g. using line simplification techniques). However, since these works do not consider the distance to nearby roads in the underlying road network, the error introduced may modify the original path of the travel trajectories, i.e. the sequence of sections of roads (called links) that are travelled. The path of a travel trajectory, which can be obtained from a sequence of GPS points by a well-studied process called map matching (Brakatsoulas et al., 2005; Yuan et al., 2010a; Yuan et al., 2010b), is crucial information for many applications such as transportation planning. In fact, several works (Li and Lin, 2006; Zhang et al., 2006; Roh et al., 2011) have proposed approaches that query the path of travel trajectories on a road network. Therefore, methods to reduce the size of a travel trajectory without modifying its original path are needed.

We address this problem by separating a trajectory into its spatial components, i.e. its path, and its temporal component, i.e. its time-stamped positions. This paper focuses on path encoding and proposes a generic encoder that encodes and decodes the path of a travel trajectory using a road network and a routing algorithm. The encoder is able to achieve high compression and retrieve the original path without error.

The basic idea of the proposed encoder and a preliminary evaluation of the encoder have been presented by Martinez Lerin et al. (2012a) at KES-IIMSS-12 (Gifu, Japan, 23–25 May 2012). This paper improves upon both aspects of the paper by Martinez Lerin et al. (2012a) by presenting two implementations of the proposed encoder, as well as a comparative study of those implementations and a general purpose data compression method as follows.

1 This paper presents a generic road-network-based encoder that uses an encapsulated routing algorithm that has two primitive functions, makeUniquePath and isUniquePath, and which makes a path and verifies it based on the routing algorithm and a road network.

2 It implements two encoders, FPRA-Encoder and SPRA-Encoder, by combining the proposed generic encoder with two routing algorithms: Following Path Routing Algorithm (FPRA) and Shortest Path Routing Algorithm (SPRA), respectively. The routing algorithms are based on the principle, derived from observation, that when people travel, they tend to follow the straightest, shortest path possible. The FPRA is a new and optimised version of the initial Following Path Algorithm (FPA) presented by Martinez Lerin et al. (2012a).

3 It evaluates the above two implementations as well as a general purpose data compression method, zip, using 75 real travel trajectories. It also shows the relationships between the length of each trajectory and the compression ratio, and the number of links in each trajectory and the computation time.

The rest of this paper is organised as follows. Section 2 presents related works on network-constrained trajectory compression. Section 3 describes the proposed generic encoder. Sections 4 and 5 describe the routing algorithms FPRA and SPRA that are used by the encoders FPRA-Encoder and SPRA-Encoder, respectively. Section 6 presents the results of the evaluation and relevant considerations. Finally, conclusions are given in Section 7.

2 Related work

In this section, we describe related works on network-constrained trajectory compression, i.e. related works that have achieved trajectory compression by considering the underlying road network. Considering the underlying road network is important to achieve the purpose of this research: reducing the size of a travel trajectory without modifying its original path, where the path of a trajectory is the sequence of road sections (called links) travelled in a road network.

Kellaris et al. (2009) propose an approach that achieves trajectory compression by replacing sections of its path with the shortest paths in a road network. This approach, however, is not suitable for our purpose because it modifies the trajectory's original path.

One approach suitable for our purpose modifies the road network by partitioning it into network's paths. Let us considering a road network as a graph on which the vertices are road intersections, and the edges, referred to as links, are sections of the road between

neighbouring intersections. The key idea of this approach is to merge several links into one link so that a path's trajectory can be represented using fewer links. This idea is proposed in two works as follows: Civilis et al. (2005) propose one related work that modifies a road network using its topology and geometry, and then uses the modified road network to track efficiently vehicles, i.e. the process is carried out before collecting trajectories. Conversely, Sandu Popa et al. (2011) propose the other related work that modifies a road network based upon the density and distributions of a given data set of trajectories, i.e. the process is performed after collecting trajectories. Our proposed approach is different from these works in two respects: it does not modify the underlying road network and the compression is done based on each trajectory.

Cao and Wolfson (2005) propose a compressed representation of a trajectory, called a non-materialised trajectory, which is composed of a sequence of tuples (on street p_i, at location l_i along p_i, at time t_i). Compression is achieved by merging tuples that are on the same street. Our proposed approach is different from this in that it implements routing algorithms that are (a) based upon the road network geometry and (b) able to encode a long path that goes through several streets using one item.

Schmid et al. (2009) propose a semantic approach, which proceeds as follows. First, a given trajectory composed of a sequence of GPS points is matched to a semantically annotated transportation network. Next, the following three kinds of semantics are searched within the trajectory: (a) change of transportation channel (e.g. enter a new street, enter a new train line), (b) descriptions of how movement continues at road intersections (e.g. straight, left and right), and (c) places where time is spent (e.g. stops). Finally, compression is achieved by representing the trajectory as a sequence of semantic events (e.g. change from street S1 to train line L1). This semantic approach is similar to ours in that a path between two encoded items is reconstructed by using a network. However, our approach reconstructs a path using a routing algorithm that is based exclusively on the geometry of the road network; therefore, it only requires a simple road network (the same as the network required for map matching). Although the semantic approach is very useful, accurate transport networks with semantic annotations may be difficult and costly to produce, and semantic annotations are more likely to change over time than the geometry of roads. The semantic approach abstracts geometry to a qualitative level by using semantic concepts such as *straight*, *right* and *left*. Although it is similar to the presented FPRA, they are different in that the FPRA finds a unique link that continues from a given link considering misalignments in the road network. The found link may have a big deviation, i.e. it may not continue *straight*. As opposite the semantic approach finds the set of links that continue *straight* from a given link.

3 Generic road-network-based encoder

This section describes the proposed generic encoder. First, we define the important concepts.

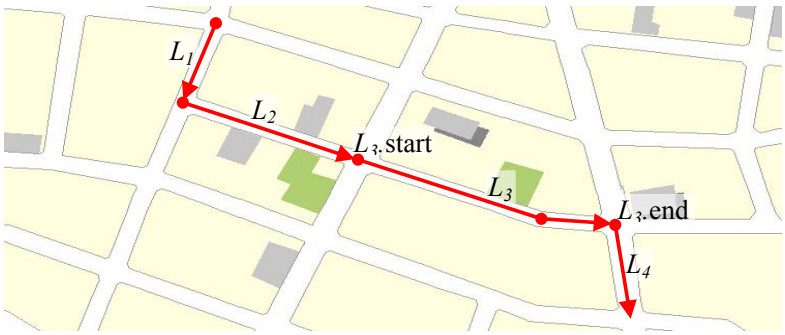
Link: A link L represents a section of road between two neighbouring intersections, and is associated with an identifier L.id, a length L.len, and a sequence of shape points L.points. The shape points define a polyline, on which the first and last points are referred to as L.start and L.end, respectively, as shown in Figure 1.

Road network: A road network is a directed graph $R(V,E)$, where V is a set of vertices that represents the intersections and terminal points of the roads, and E is a set of directed edges that represents the links.

Path: A path P is a sequence of connected links in a road network, i.e. $P = (L_1, L_2, \ldots, L_N)$, where L_i end $= L_{i+1}$ start and $1 \leq i < N$. Figure 1 shows an example of a path composed of four links.*Routing algorithm*: A routing algorithm finds a unique path between the given locations in a given road network. For example, some automotive navigation aids implement a routing algorithm that finds the shortest path. In this paper, we consider that a routing algorithm *RALG* serves two functions. A function *RALG.makeUniquePath*(R,L_A,L_B) returns the unique path found by *RALG* in the road network R between the links L_A and L_B. A function *RALG.isUniquePath*(R,P) returns *true* if and only if path P is a unique path that can be found by *RALG* in R.

The encoder can be defined as follows: Given a road network R, a path P in R, and a routing algorithm *RALG*, an encoding function transforms P into a sequence S of links that belong to P by using R and *RALG*, and a decoding function transforms S into P by using R and *RALG*. The main objective is to minimise the number of links in S and thereby reduces the storage requirements. Hereinafter, we refer to P as the original path and S as the encoded path.

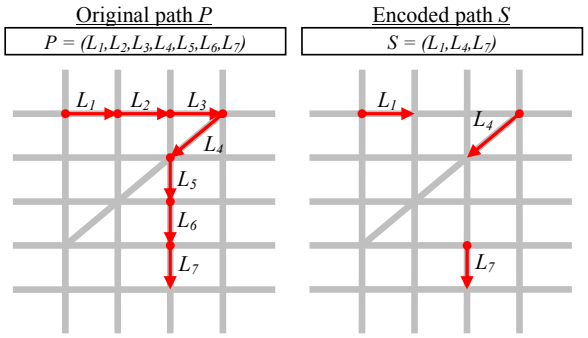
Figure 1 Representation of a path in a road network. (see online version for colours)



The key idea of the encoder is to find consecutive sub-paths that are as long as possible within P, where a sub-paths is a path that the *RALG* would find. Figure 2 shows the results of the encoding function that were obtained by using the routing algorithm SPRA that finds the shortest path. The arrows indicate the links and the thick lines represent R. The encoding function finds two sub-paths, $subP1 = (L_1,L_2,L_3,L_4)$ and $subP2 = (L_4,L_5,L_6,L_7)$, i.e. *SPRA.isUniquePath*$(R,subP1)$ and *SPRA.isUniquePath*$(R,subP2)$ returns *true*. The decoding function recovers the original path by executing *SPRA.makeUniquePath* (R,L_1,L_4) and *SPRA.makeUniquePath*(R,L_4,L_7).

Below we describe the algorithms for the encoding and decoding functions. A sequence S of n elements is defined as $S = (e_1,e_2, \ldots, e_i, \ldots, e_n)$ and the i-th element of S is denoted by $S[i]$. When $S = (e_1, e_2, \ldots, e_{n-1}, e_n)$ and $Q = (e_1, e_2, \ldots, e_{n-1})$, we obtain $S = (Q, e_n)$. When $S = (e_1, e_2, \ldots, e_m, e_{m+1}, e_{m+2}, \ldots, e_{m+n})$, $R = (e_1,e, \ldots, e_m)$, and $Q = (e_{m+1},e_{m+2}, \ldots, e_{m+n})$, we obtain $S = (R, Q)$.

Figure 2 Example of a path encoded using the routing algorithm *SPRA* that finds the shortest path (see online version for colours)



Function Encoding (*R*, *P*, *RALG*)

Input: A road network *R*, a path *P* of *N* links in *R*, and a routing algorithm *RALG*.

Output: A sequence of links *S*.

S = (*P*[1])

subPath = (*P*[1], *P*[2])

For *i* = 2 to *N* – 1 **do**

subPath = (subPath, *P*[*i* + 1])

If ! *RALG.isUniquePath*(*subPath*, *R*) **then**

S = (*S*, *P*[*i*])

subPath = (*P*[*i*], *P*[*i* + 1])

EndIf

EndFor

S = (*S*, *P*[*N*])

Return S

Function Decoding (*R*, *S*, *RALG*)

Input: A road network *R*, a sequence *S* of *N* links in *R*, and a routing algorithm *RALG*.

Output: A path *P*.

P = (*S*[1])

For i = 1 to *N* **do**

subPath = *RALG.makeUniquePath*(*S*[*i*], *S*[*i* + 1], *R*)

P = (*P*, *subPath*)

P = (*P*, *S*[*i* + 1])

EndFor

Return *P*

4 Following path routing algorithm

The FPRA finds the path that starts from a link and continues forward, and at each intersection follows the link with the smallest deviation, which is referred to as Following Path. FPRA is based on the FPA that was presented for developing the Focus+Glue+Context map system EMMA (Takahashi, 2008; Yamamoto et al., 2009).

The functions *makeUniquePath* and *isUniquePath* of the FPRA are defined below. Let us consider the function *FLA(L,R)*, which returns the link connected to the link L in the road network R with the smallest deviation (angle), considering misalignments in the road network. By considering the misalignments, the compression ratio slightly improves in 4 of the 75 evaluated trajectories.

Function FLA(R, L): Given a road network R and a link L in R, the function first selects the subset SR that contains the links in R that lie within or partially within a square of area $s_b \times s_b$ centred on the point L.end. The length s_b (in meters) is a system parameter, discussed in Section 6. Then, the function returns a link computed by obeying the rules prescribed below. Let us consider $L_1 - L_N$ the N links that are connected to L and belong to SR. Let us consider α_i the deviation (angle) between the connected links L_i and L, as shown in Figure 3.

Rule 1: When $N = 0$, the function returns *NULL*.

Rule 2: When $N = 1$, the following link is L_1.

Rule 3: When $N > 1$ and the deviation α_i between L and L_i is the smallest among the deviations α_k ($1 \leq k \leq N$) between L and the connected links, the following link is L_i, as shown in Figure 3.

Rule 4: When L_i.len < l_0, we assume that the links connected to L_i are directly connected to L, as shown in Figure 4. The very small length l_0, for example 2 m, is a system parameter that determines when a link is too small. Very small links indicate misalignments, as shown in Figure 5, for which the algorithm uses Rule 4 to select the link L_s as the following link of L.

Figure 3 Deviation (angle) between the link L and the links connected to L (see online version for colours)

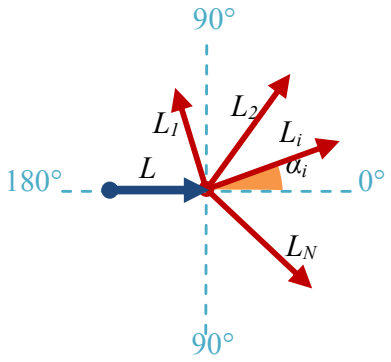


Figure 4 Rule 4 of function FLA (see online version for colours)

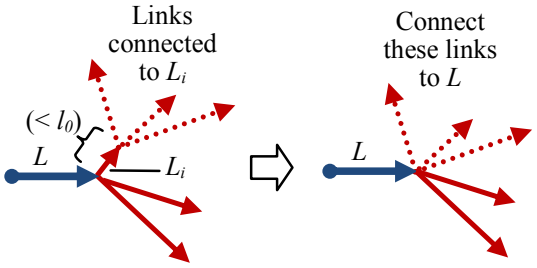
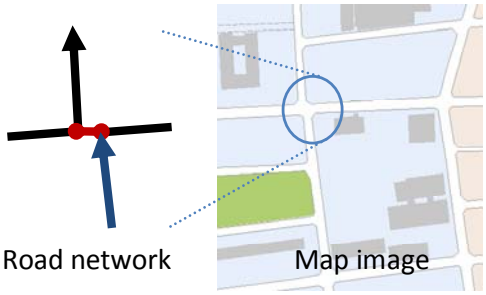


Figure 5 Example of a slightly misaligned intersection (see online version for colours)



Function FPRA.makeUniquePath (R, L_A, L_B)

Input: A road network R, and two links L_A and L_B in R.

Output: A path P, which is a Following Path.

$L = FLA(L_A, R)$

$P = (P, L)$

While L.end <> L_B.start **do**

$L = FLA(L, R)$

$P = (P, L)$

EndWhile

Return P

Function FPRA.isUniquePath (R, P): Given a road network R and a path P composed of N links in R, i.e. $P = (L_1, L_2, \ldots, L_N)$, the function returns true if and only if $FLA(L_{i-1},R)$ returns L_i for $1 < i < N$, and P only reaches the intersection L_N start once. This last condition ensures that P can be retrieved using $FPRA.makeUniquePath(R, L_1, L_N)$.

When the FPRA is used in the generic encoder, i.e. in the encoding function, it is known from previous iterations that path P' that contains the first $N-1$ links of P is a Following Path, i.e. $FPRA.isUniquePath(R, P')$ returns *true*. Therefore, the function only needs to evaluate the result of *FLA* for $i = N-1$, instead of evaluating the results of *FLA*

for each value of i. Therefore, when used in the generic encoder, *FPRA.isUniquePath (R, P)* returns *true* if and only if *FLA*(L_{N-2}, *R)* returns L_{N-1} and P only reaches L_N.start once. This last condition can be evaluated in constant time if the encoding function maintains a hash table of the intersections reached.

The function *FLA(L,R)* has a constant cost because it only considers the few links that are connected to the input link L. Therefore, using FPRA, the encoding and decoding functions of the generic encoder have a cost O(n), where n is the number of links in the original path.

5 Shortest path routing algorithm

The SPRA finds the shortest path between two given links. The functions *makeUniquePath* and *isUniquePath* of the SPRA are defined below.

Function SPRA.makeUniquePath (R, L_A, L_B): Given a road network R and two links L_A and L_B in R, the function returns the shortest path from L_A.end to L_B.start in R by implementing the well-known algorithm presented by Dijkstra (1959). Since the road network that is available is often very extensive, the function computes the shortest path in a delimited area *SR* of the road network R. The area *SR* contains the links in R that lie within or partially within the minimum bounding box that includes two squares of area $s_a \times s_a$ centred on the points L_A.end and L_B.start. The length s_a (in meters) is a system parameter, discussed in Section 6. The function returns *NULL* when there is no path in *SR* that reaches L_B.start from L_A.end.

Function SPRA.isUniquePath (R, P): Given a road network R and a path P in R, let us consider L_A and L_B to be the first and last links of P, respectively, and consider P' to be the path P without the links L_A and L_B. The function returns *true* if and only if *SPRA.makeUniquePath(R, L_A, L_B)* returns P'.

Dijkstra's algorithm (optimised) has a cost of O(v*log$v$), where v is the number of intersections in the regions of road network used. Therefore, when using the SPRA in the generic encoder, the decoding function has a cost of O(m*v*log$v$), where m is the number of links in the encoded path, and the encoding function has a cost of O(n*v*log$v$), where n is the number of links in the original path.

6 Evaluation

6.1 Experiment

For the evaluation, we used a data set of 75 network-constrained trajectories of real trips within Japan. The 75 trajectories, each composed of a sequence of GPS points, included a total of 39,224 GPS points. Forty one trajectories were logged by members of our laboratory at a sampling rate of 5 s per point. The remaining 34 trajectories had sampling rates of 2–5 s per point, and are available in The OpenStreetMap (see http://www.openstreetmap.org/) Project under the Creative Commons BY-SA 2.0 license (see http://creativecommons.org/licenses/by-sa/2.0/). The experiments were conducted on an Intel Core i7, 2.93 GHz desktop PC with 4 GB RAM and running Windows 7. We used the entire road network of Japan stored in a MySQL database in our laboratory. The methods were implemented using Java and ActionScript.

Original path: In a pre-processing step, the path of each trajectory in the data set, referred to as the original path, was obtained by a point-to-curve map-matching algorithm presented by Martinez Lerin et al. (2012b). After the map matching, we visually examined the results to ensure that the links were properly matched. Finally, the original paths were stored separately. A path of links is stored by using the id of its links, i.e. it is stored as a sequence of ids, where each id is a sequence of 2-B characters. The 75 original paths have a total length of 1307 km, a total number of 12,295 links and a size of 298.81 KB.

Encoded path: For each evaluated method, the experiment was conducted as follows. First, the encoding function transformed each original path into an encoded path. Then, using the decoding function, we ensured that the original path could be retrieved from the encoded path without error. Finally, the encoded paths were stored separately, as described for each method in the next subsection.

We evaluated the compression ratio achieved for each path using following equation (1). The compression ratio is a value from 0 to 1, where smaller values indicate higher compression. The computation times of the encoding and the decoding functions were also individually measured.

$$\text{compression ratio} = \frac{\text{encoded path size}(\text{bytes})}{\text{matched path size}(\text{bytes})} \tag{1}$$

6.2 Methods

We experimentally evaluated the five methods described below.

M1 (FPRA-Encoder): We term as M1 the proposed encoder FPRA-Encoder that combines the generic road-network-based encoder and the FPRA. The encoded paths are stored in the same way as the original paths, as described in the previous subsection. The FPRA uses $s_b = 10$ m and $l_0 = 5$ m. We evaluated different values for the parameters s_b and l_0 and concluded that small changes had no effect on the result as long as s_b is greater than l_0.

M2 (SPRA-Encoder): We term as M2 the proposed encoder SPRA-Encoder that combines the generic road-network-based encoder and the SPRA. The encoded paths are stored in the same way as the original paths. The SPRA uses $s_a = 1500$ m. We evaluated different values for the parameter s_a and concluded that (a) the greater the parameter s_a, the higher the compression, (b) the greater the parameter s_a, the higher the computation time, and (c) when the parameter s_a exceeds 1000 m, in most cases the compression does not vary. The reason for these conclusions is that when the parameter s_a is small, the delimited area is also small, and so a shortest path may not be detected because part of it falls outside the delimited area.

M3 (ZIP): Zip is a widely used method (DEFLATE algorithm). The encoding constitutes separately zipping the stored original paths to obtain the encoded paths, which are stored in binary files. The decoding recovers the original paths by unzipping.

M4 (FPRA-Encoder–ZIP): This method combines M1 and M3. The encoding function of M4 follows two steps: (a) each original path is transformed into an encoded path and is stored using the method M1; (b) the stored encoded paths are zipped. The decoding function of M4 follows two steps: (a) the encoded paths are unzipped; (b) the original paths are recovered by using the decoding function of the method M1.

M5 (SPRA-Encoder–ZIP): This method is the same as method M4, but uses M2 instead of M1.

6.3 Results

In order to evaluate in greater detail the results achieved by the five methods, we classified the trajectories in five groups according to their lengths, as follows:

- Group 1 (0–5 km): Composed of 16 trajectories.
- Group 2 (5–10 km): Composed of 15 trajectories.
- Group 3 (10–20 km): Composed of 16 trajectories.
- Group 4 (20–30 km): Composed of 15 trajectories.
- Group 5 (30–80 km): Composed of 13 trajectories.

The average of each path's compression ratio for each method is listed in Table 1 for all trajectories, and for each group of trajectories is shown in Figure 6. The average computation time of methods M1 and M2 for each group of trajectories is shown in Table 2. The computation time of the rest of the methods is not shown because the zip algorithm only slightly increases the computation time.

Table 1 Average compression ratio for each method

Methods	*Compression ratio*
M1 (FPRA-Encoder)	0.138
M2 (SPRA-Encoder)	0.094
M3 (ZIP)	0.179
M4 (FPRA-Encoder–ZIP)	0.046
M5 (SPRA-Encoder–ZIP)	0.040

Table 2 Average computation time (s) of the encoding and decoding functions of M1 and M2 for each group of trajectories

	Group 1 (0–5 km)	*Group 2 (5–10 km)*	*Group 3 (10–20 km)*	*Group 4 (20–30 km)*	*Group 5 (30–80 km)*
M1 encoding	0.10	0.21	0.53	0.82	1.21
M1 decoding	0.07	0.19	0.54	0.85	1.29
M2 encoding	8.46	14.27	165.13	541.17	472.74
M2 decoding	1.67	1.51	8.57	31.45	27.89

In order to understand the behaviour of the computation time in greater detail, Figures 7 and 8 show the computation time results of encoding and decoding each trajectory according to the number of links in its original path.

Figure 7 Computation time (s) of the encoding and decoding functions of M1 (FPRA-Encoder) (see online version for colours)

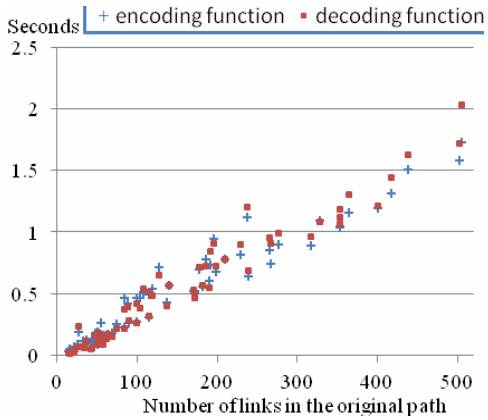
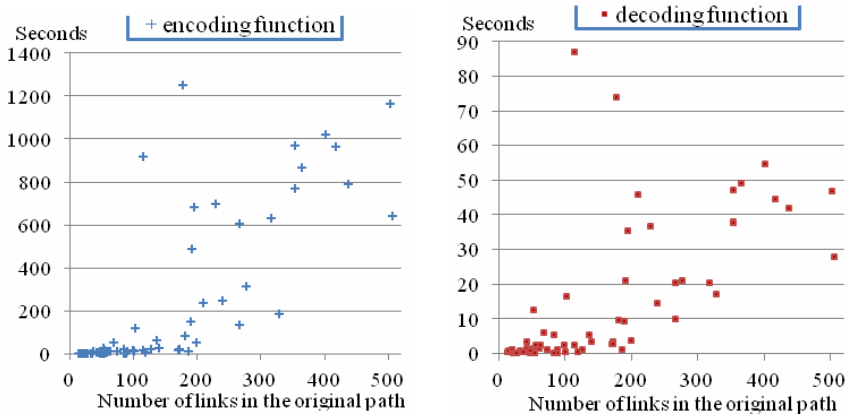


Figure 8 Computation time (s) of the encoding and decoding functions of M2 (SPRA-Encoder) (see online version for colours)



7 Considerations

The results for the compression ratio are summarised as follows:

1. All five methods achieve high compression for all groups of trajectories.
2. Longer trajectories have smaller values of compression ratio, i.e. produce higher compression, though for trajectories of more than 20 km, the compression ratio appears to stabilise.
3. M2 achieves higher compression than M1, especially for trajectories of less than 5 km.
4. When the two proposed encoders are combined with the ZIP method, they achieve a very similar compression.

Our finding that longer trajectories produce higher compression can be attributed to the fact that FPRA and SPRA can find longer paths when the trajectory has few turns and when the underlying road network is not dense, which is the case in inter-city travel, i.e. long trajectories. To the contrary, short trajectories usually correspond to intra-city travels that include a high number of turns in places where the road topology is dense and complex.

M2 achieves higher compression than M1 because a Following Path is usually a Shortest Path, but a Shortest Path is not necessarily a Following Path. As a result, M2 finds paths that are longer than or equal to those found by M2. This difference is smaller in longer trajectories because long trajectories tend to use more widely travelled roads, such as highways, where long Shortest Paths and Following Paths can be found.

The results for the computation time are summarised as follows:

1 M1 is very fast and its computation time grows linearly with the number of links in the original path.

2 M2 may take a very long time and its computation time is difficult to predict.

The computation time of M2 exhibits irregularities because the amount of road network that is used, each time a Shortest Path is computed varies with the distance between the start and end points of the Shortest Path and the density of the underlying road network. Several works (Huang et al., 1997; Sanders and Schultes, 2007) have proposed solutions that can very quickly find the shortest path between two distant points by pre-processing the road network.

To sum up, M5 achieves the best results when the ZIP method is used. Otherwise, either M1 or M2 can be considered the most suitable depending on the needs of the application, namely the length of the input trajectories, and the importance of the compression ratio and computation time. If we consider the compression ratio and computation time, M1 seems to produce better results for long trajectories and M2, for short trajectories.

As a final consideration, it is important to note that in some cases the ZIP method cannot be applied or is not convenient. For example, it needs to have received the entire trajectory before it begins compression, and the zipped data cannot be used as is.

8 Conclusions

Collection and storage for future querying, along with the processing of large amounts of network-constrained travel trajectories, i.e. travel trajectories along a road network, imposes significant transmission and storage challenges. In this paper, we argue that the path of a trajectory, i.e. the sequence of roads travelled, is crucial information for many applications. We have therefore presented a generic encoder and two implementations of that encoder, FPRA-Encoder and SPRA-Encoder, which reduce the size of a travel trajectory's path without modifying it. The evaluation presented in this paper, which uses a large-data set of real travel trajectories, shows that the two presented encoders drastically reduce the size of a trajectory's path. If we consider the compression ratio and computation time, the FPRA-Encoder produces the best results when the encoders are combined with the ZIP method. Otherwise, the FPRA-Encoder produces better results for long trajectories, and the SPRA-Encoder produces better results for short trajectories.

Stream-based encoding, in which a trajectory is encoded as it is being received without waiting for it to be completely received, is necessary for certain applications, such as reducing the communication requirements of a moving object that is regularly sending its current position to a server. Adapting the proposed generic encoder to support stream-based input is straightforward because it accesses the input path and builds the encoded version successively, item by item.

We concentrated on encoding the spatial components of network-constrained travel trajectories and developed the proposed methods under the following assumptions:

1 The decoding function can use the same road network as that used by the encoding function;

2 The spatial components of travel trajectory are often used without temporal components. Even if the temporal components of travel trajectories are necessary, they can be kept without encoding and added to the decoded spatial components, or they can be encoded and decoded by other tools.

These assumptions will impose limitations on the proposed methods. Further researches are required to remove the limitations and enhance the proposed method.

When the road network used in the encoding function is not the same as the road network used in the decoding function; the proposed generic encoder may not be able to recover the original path. This situation arises, for example, when the road network is updated and the encoded paths are shared and decoded using different road network data. One solution may be to store the road network along with the trajectories to ensure that it never changes, though this is not always possible or efficient. Future work that remains to be done in developing the proposed encoders involves (a) improving the encoders so that they can detect whether changes in the road network have caused errors in the decoded trajectory, e.g. by using CRC values, and (b) improving the encoders so they can deal with errors caused by changes in the road network and retrieve a trajectory that is as similar as possible to the original trajectory.

The spatial and temporal components can be encoded and decoded independently, and combined after decoding them. General purpose data compression tools like the zip function can be applied to the temporal components of travel trajectories. The size of encoded a travel trajectory might be extremely reduced by introducing approximation of the temporal components. We are currently working on adding some of the trajectory's temporal information of a trajectory to the encoded path. One simple idea in this regard, for example, is to add a time-stamp to each link in the encoded path and assume a constant speed between the time-stamped links.

Acknowledgements

We would like to thank Yahoo! Japan Corporation for supporting us in the development of the prototype system. This work was also supported by JSPS KAKENHI 20509003 and 23500084.

References

Brakatsoulas, S., Pfoser, D., Salas, R. and Wenk, C. (2005) 'On map-matching vehicle tracking data', *VLDB 2005: Proceedings of the 31st international conference on Very Large Data Bases*, pp.853–864.

Cao, H. and Wolfson, O. (2005) 'Nonmaterialized motion information in transport networks', *ICDT 2005: Proceedings of the 10th international conference on Database Theory*, Springer-Verlag, Heidelberg, Berlin, pp.173–188.

Civilis, A., Jensen, C.S. and Pakalnis, S. (2005) 'Techniques for efficient road-network-based tracking of moving objects', *IEEE Transactions on Knowledge and Data Engineering*, Vol. 17, No. 5, pp.698–712.

Dijkstra, E.W. (1959) 'A note on two problems in connection with graph theory', *Numerische Mathematik*, Vol. 1, pp.269–271.

Hönle, N., Grossmann, M., Reimann, S. and Mitschang, B. (2010) 'Usability analysis of compression algorithms for position data streams', *GIS 2010: Proceedings of the 18th SIGSPATIAL International Conference on Advances in Geographic Information Systems, ACM*, NY, USA, pp.240–249.

Huang, Y-W., Jing, N. and Rundensteiner, E.A. (1997) 'A hierarchical path view model for path finding in intelligent transportation systems', *GeoInformatica*, Vol. 1, No. 2, pp.125–159.

Kellaris, G., Pelekis, N. and Theodoridis, Y. (2009) 'Trajectory compression under network constraints' *SSTD 2009: Proceedings of the 11th International Symposium on Advances in Spatial and Temporal Databases*, Springer-Verlag, Heidelberg, Berlin, pp.392–398.

Koegel, M., Baselt, D., Mauve, M. and Scheuermann, B. (2011) 'A comparison of vehicular trajectory encoding techniques', *Proceedings of Med-Hoc-Net*, pp.87–94.

Li, X. and Lin, H. (2006) 'Indexing network-constrained trajectories for connectivity-based queries', *Proceedings of International Journal of Geographical Information Science*, Vol. 20, No. 3, pp.303–328.

Martinez Lerin, P.M., Yamamoto, D. and Takahashi N. (2012a) 'Encoding travel traces by using road networks and routing algorithms', *KES-IIMSS 2012: Proceedings of the 5th International Conference on Intelligent Interactive Multimedia Systems and Services*, Smart Innovation, Systems and Technologies, Springer, Vol. 14, pp.233–243.

Martinez Lerin, P.M., Yamamoto, D. and Takahashi N. (2012b) 'Making a pictorial and verbal travel trace from a GPS trace', *W2GIS 2012: Proceedings of the 11th International Symposium on Web and Wireless Geographical Information Systems*, Springer-Verlag, Heidelberg, Berlin, Vol. 7236, pp.98–115.

Meratnia, N. and de By, R.A. (2004) 'Spatiotemporal compression techniques for moving point objects', *EDBT 2004: Proceedings of the 9th International Conference on Extending Database Technology*, pp.765–782.

Roh, G-P., Roh, J-W., Hwang, S-W. and Yi, B-K. (2011) 'Supporting pattern-matching queries over trajectories on road networks', *IEEE Transactions on Knowledge and Data Engineering*, Vol. 23, No. 11, pp.1753–1758.

Sanders, P. and Schultes, D. (2007) 'Engineering fast route planning algorithms', *WEA 2007: Proceedings of the 6th International Conference on Experimental Algorithms*, Springer-Verlag, Berlin, Heidelberg, Vol. 4525, pp.23–36.

Sandu Popa, I., Zeitouni, K., Vincent, O., Barth, D. and Vial, S. (2011) 'Indexing in-network trajectory flows', *The International Journal on Very Large Data Bases*, Vol. 20, No. 5, pp.643–669.

Schmid, F., Richter, K-F. and Laube, P. (2009) 'Semantic trajectory compression', *SSTD 2009: Proceedings of the 11th International Symposium on Advances in Spatial and Temporal Databases*, Springer-Verlag, Berlin, Heidelberg, pp.411–416.

Takahashi, N. (2008) 'An elastic map system with cognitive map-based operations', *International Perspectives on Maps and Internet*, Springer-Verlag, pp.73–87.

Xue, G., Li, Z., Zhu, H. and Liu, Y. (2009) 'Traffic-known urban vehicular route prediction based on partial mobility patterns', *ICPADS 2009: Proceedings of the International Conference on Parallel and Distributed Systems*, IEEE Computer Society, Washington, DC, USA, pp.369–375.

Yamamoto, D., Ozeki, S. and Takahashi, N. (2009) 'Focus+Glue+Context: an improved fisheye approach for web map services', *Proceedings of the 17th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, Seattle, Washington, pp.101–110.

Yuan, J., Zheng, Y., Zhang, C., Xie, W., Xie, X., Sun, G. and Huang, Y. (2010a) 'T-drive: driving directions based on taxi trajectories', *GIS 2010: Proceedings of the 18th SIGSPATIAL International Conference on Advances in Geographic Information Systems*, ACM, NY, USA, pp.99–108.

Yuan, J., Zheng, Y., Zhang, C., Xie, X. and Sun, G. (2010b) 'An interactive-voting based map matching algorithm', *MDM 2010: Proceedings of the 11th International Conference on Mobile Data Management*, IEEE Computer Society, pp.43–52.

Zhang, Y., Marca, J.E., Rindt, C.R., Jayakrishnan, R. and McNally, M.G. (2006) 'Efficient path and subpath storage data structures for network travel analysis with route behavioural elements', *The 86th Annual Meeting of the Transportation Research Board*, Washington, DC, USA.