# Trajectory Compression under Network Constraints

Georgios Kellaris, Nikos Pelekis, and Yannis Theodoridis

Department of Informatics, University of Piraeus, Greece
{gkellar,npelekis,ytheod}@unipi.gr
http://infolab.cs.unipi.gr

**Abstract.** The wide usage of location aware devices, such as GPS-enabled cellphones or PDAs, generates vast volumes of spatiotemporal streams modeling objects movements, raising management challenges, such as efficient storage and querying. Therefore, compression techniques are inevitable also in the field of moving object databases. Moreover, due to erroneous measurements from GPS devices, the problem of matching the location recordings with the underlying traffic network has recently gained the attention of the research community. So far, the proposed compression techniques are not designed for network constrained moving objects, while map matching algorithms do not consider compression issues. In this paper, we propose solutions tackling the combined, map matched trajectory compression problem, the efficiency of which is demonstrated through an experimental evaluation using a real trajectory dataset.

**Keywords:** Trajectory Compression, Road Network, Map-Matching.

## 1 Introduction

A Moving Object Database (MOD) is a collection of objects whose location changes over time. The preservation of vast volumes of moving objects' trajectories for future reference raises compression aspects, i.e., the *trajectory compression* problem (hereafter, called *TC*). Locations are often recorded by GPS receivers which embed an error of some meters. Thus, there arises the problem of matching these data points onto a network, also known as the *map-matching problem* (hereafter, called *MM*).

In this work, we study the combined problem of the compression of a moving object's trajectory keeping it at the same time matched on the underlying road network (hereafter, called map matched trajectory compression problem - MMTC).

To the best of our knowledge, there is no related work directly addressing the MMTC problem. Regarding its two components, TC and MM, the state-of-the-art is [6] and [1], respectively. In particular, Meratnia and de By [6] propose a compression technique that uses the Douglas-Peucker method and, moreover, takes the parameter of time into account. In particular, it replaces the Euclidean distance used in Douglas-Peucker by a time-aware one, called *Synchronous Euclidean Distance* (SED) [6]. The time complexity of the algorithm proposed in [6] is O($n^2$), where $n$ is the number of points composing the trajectory. Regarding MM, Brakatsoulas et al. [1] propose the following methodology: for every point $P_i$, given that point $P_{i-1}$ has already been matched, the adjacent edges to this edge are the candidate edges to be matched to $P_i$ and they are evaluated. In order to choose among the candidate edges two measures are used that take into consideration the distance and the orientation of the edges. The higher the sum of these measures is, the better the match to this edge is. The quality of the result is improved by using a "look ahead" policy. That is, the total score of each candidate edge is calculated by adding the scores of a fixed number of edges, which are ahead of the current position, to the initial one. The time complexity of the algorithm proposed in [1] is O($n$).

Also related to ours is the work by Tiakas et al. [7], where a method for trajectory similarity search under network constraints is proposed. In particular, the cost to travel from one node to another (which could be travel distance, average travel time, etc.) is used to calculate the *network distance* between two trajectories as the average of the equivalent node distances, and, on top of that, the total similarity $D_{total}$ between two trajectories is expressed as a weighted average of their network and *time* distances.

The most related to our work is the one by Cao and Wolfson [2], which explored the combination of the map-matching with the storage-space problem by proposing a solution that uses the a priori knowledge of the road network. However, our approach is different as our goal is to reduce the size of the MOD keeping the trajectory data without altering its infrastructure, but only by removing and/or altering certain records. Of course, the proposed solution is not a lossless compression technique, but it preserves the structure of the original data providing at the same time data ready to be used without any preprocessing.

The contribution of the paper is three-fold:

1. Taking into consideration off-the-shelf TC and MM algorithms, we propose two working solutions for the combined MMTC problem.
2. Formulating MMTC as a cost-optimization problem, we present a theoretical analysis for finding the optimal solution accompanied, due to its high computational cost, by an approximate algorithm, called MMTC-App.
3. Performing an experimental study conducted over a real trajectory dataset, we demonstrate the effectiveness and the efficiency of the proposed solution.

The paper outline is as follows. In Section 2 we formalize the MMTC problem. Section 3 presents two solutions based on existing TC and MM algorithms as well as our novel approximate solution (MMTC-App) followed by their experimental evaluation in Section 4. Section 5 concludes the paper.

## 2 MMTC Problem Formalization

Before we present our solutions for the MMTC problem, we formalize the MMTC problem. Due to space limitations, the definitions of the key concepts of our work (*Network*, *Trajectory*, *Map-matched trajectory*, *Map-matched counterpart of a trajectory*, and *Compressed version of a trajectory*) are presented in detail in [5]. On top of those definitions, the MMTC problem is formalized as follows:

**Definition 1 (MMTC Problem).** Given a road network *G*(*V*, *E*) consisting of graph vertices *V* and edges *E* and a trajectory *T* consisting of time-stamped point locations <*x*, *y*, *t*>, the *map-matched trajectory compression* (MMTC) problem asks to find a network-constrained trajectory, $T_{MMTC}$, which is (a) a compressed version of the map-matched counterpart of *T*, called *T'*, and (b) as similar to *T'* as possible.

The degree of compression *Comp*($T_{MMTC}$, *T'*) between $T_{MMTC}$ and *T'* is measured as follows:

$$Comp(T_{MMTC}, T') = \begin{cases} 1 - \frac{|T_{MMTC}|}{|T'|}, if\ |T_{MMTC}| < |T'| \\ 0, otherwise \end{cases} \quad (1)$$

while the degree of similarity *Sim*($T_{MMTC}$, *T'*) between $T_{MMTC}$ and *T'* is measured as follows:

$$Sim(T_{MMTC}, T') = 1 - D_{total}(T_{MMTC}, T') \quad (2)$$

where $D_{total}(T_{MMTC}, T')$ is calculated as in [0] using the network distance defined in [5].

Since it is expected that optimizing both *Comp*() and *Sim*() is contradicting, an overall quality measure $0 \le Q(T_{MMTC}, T') \le 1$ is defined as an aggregate of *Comp*() and *Sim*(). The trajectory $T_{MMTC}$ that maximizes *Q* among all possible network-constrained trajectories is the solution to the MMTC problem. ■

In the rest of the paper, we adopt

$$Q(T_{MMTC}, T') = Comp(T_{MMTC}, T') \cdot Sim(T_{MMTC}, T') \quad (3)$$

which requires the values of both components to be as high as possible in order for its value to be high. Of course, other quality functions could be equally taken into consideration.

In the following section, we investigate the MMTC problem and propose solutions either exploiting off-the-shelf TC and MM techniques or designing from scratch.

## 3 Solutions to the MMTC Problem

By combining the algorithms of data compression and map-matching we devise two naïve solutions:

- 1st approach (called, TC+MM): the original trajectory *T* is compressed to $T_{TC}$ using TC, which afterwards is map-matched to $T_{MMTC}$ using MM.
- 2nd approach (called, MM+TC+MM): the original trajectory *T* is map-matched to $T_{MM}$ using MM, which afterwards is compressed to $T'_{MMTC}$ using TC, which afterwards is map-matched to $T_{MMTC}$ using MM (since applying a general TC algorithm on a map-matched trajectory destroys its map-matched properties).

If we adopt the algorithms [6] and [1], with O($n^2$) and O($n$) time complexity, respectively, it turns out that the complexity of both TC+MM and MM+TC+MM is O($n^2$).

Alternatively, in this paper we propose a novel compression method. The motivating idea is that the compressed trajectory is built by replacing some paths of a given map-matched trajectory with shorter ones. This could be done by executing a shortest path algorithm (hereafter, called *SP*) on appropriately selected points of the trajectory without considering the weights of the edges. We ignore the weights in order to get the result with the minimum number of nodes and, hence, achieve a high compression. Moreover, we need to choose among all the possible shorter paths, the one that maximize the value of the quality measure *Q* defined in Eq. (3).

In particular, we adopt the *Minimum Description Length* (MDL) principle. There are two components that comprise MDL, namely $L(H)$ and $L(D|H)$, where $H$ denotes the hypothesis and $D$ denotes the data, as presented in [3]. The best hypothesis $H$ to explain $D$ is the one that minimizes the sum of $L(H)$ and $L(D|H)$. Mapping the above discussion to our problem, $L(H)$ represents the compression achieved by a compressed trajectory and $L(D|H)$ represents the difference between the compressed trajectory and the original one as explained in [5].

Recalling Definition 1, the solution we envisage for the MMTC problem is the path that minimizes the sum $L(H) + L(D|H)$ adopting the MDL principle. Unfortunately, the cost of finding this path is prohibitive since we need to consider every combination of shortest paths between the points of the original trajectory. Actually, it is similar to the problem of finding all possible acyclic paths between two graph nodes. Since we cannot expect to find the optimal trajectory in reasonable time, we propose an algorithm that would approximate it.

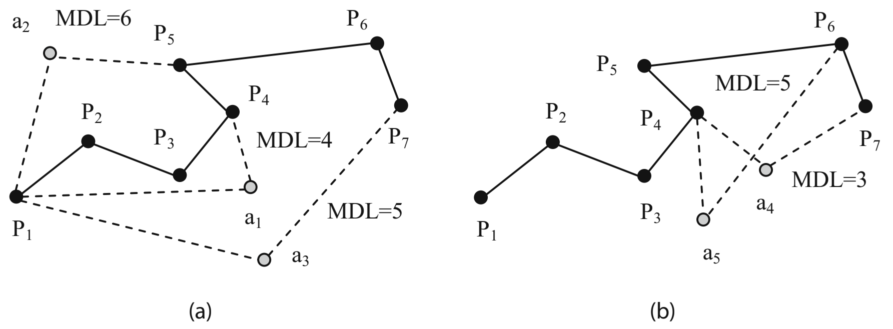

**Fig. 1.** Example of the MMTC-App algorithm: shortest paths and their MDLs (a) from the first node and (b) from the fourth node

The main idea of our approach is illustrated in Fig. 1 whereas the pseudocode of the proposed approximate algorithm, called MMTC-App, is listed in [5] (due to space limitations). First, we calculate a map-matched counterpart $T_{MM}$ of the original trajectory $T$ simply by matching every point of the trajectory to a network edge.

In particular, we choose the edge that is closest to the examined point and it is adjacent to the previous selected. Following the example illustrated in Fig. 1, given $T_{MM}$ $\{P_1, P_2, P_3, P_4, P_5, P_6, P_7\}$, the algorithm calculates the MDL of every SP as illustrated in Fig. 1(a), and chooses the one with the minimum MDL value. Let us suppose that this value is given by the SP from $P_1$ to $P_4$ via node $a_1$. Then we can replace the sub-path $\{P_1, P_2, P_3, P_4\}$ of the temporary result by $\{P_1, a_1, P_4\}$. We also need to calculate the time the object is located at node $a_1$. This can be estimated by using the temporal information on nodes $P_1$ and $P_4$ and considering the object is moving at constant speed. The algorithm stores this result and continues by considering as first point the last node of the already found SP and by checking the SPs from this node to its next ones. In our running example, $P_4$ is checked against its next points. Finally, the SP with the best score is the one from $P_4$ to $P_7$ via $a_4$, as illustrated in Fig. 1(b). Since $P_7$ is the end point of the trajectory, the algorithm terminates by returning the compressed trajectory $\{P_1, a_1, P_4, a_4, P_7\}$. In case no SP is found in a sub-path which is under evaluation, the algorithm adds the remaining points to the output and, then, terminates.

The MMTC-App algorithm requires the SPs from the original nodes to all others to be pre-calculated. This is an offline procedure of general purpose. Theoretically, the time complexity for running an all-pair shortest path algorithm on a network $G(V, E)$ is $O(|V|^2\log|V|)$ [4]. The complexity of MMTC-App algorithm is $O(n^2\log n)$ on average, where $n$ is the number of points composing the trajectory, excluding the cost of shortest path calculations, with the proof found in [5].

## 4 Experimental Study

We evaluated the proposed techniques over a real dataset consisting of trajectories of vehicle movements in the city of Milano (described in [5]). For each trajectory, a compressed and map-matched version of it was constructed, following one of the above techniques. The resulting trajectories were compared against their map-matched counterparts with respect to (a) a quality criterion (Eq.(3)) (b) the compression achieved (Eq.(1)) and (c) the execution time.

In our experiments, we included MMTC-App together with TC+MM(low), TC+MM(high), MM+TC+MM(low), MM+TC+MM(high), where low and high indicate different threshold values of TC, thus, the rate of compression. The details of the experimentation are discussed in [5].

The first set of experiments concentrates on the effectiveness of the proposed methods, i.e., as high compression (according to Eq.(1)) and overall quality (according to Eq.(3)) as possible. Fig. 2 illustrates the compression achieved (labeled 'Comp' in the chart) together with the resulting quality ('Q'). It is clear that MMTC-App significantly compresses the original trajectory keeping about 60% of its size while, at the same time, the overall quality is at least twice the quality of any of the naïve approaches. It is also worth to be mentioned that high speed and agility has offered slightly higher compression for methods MM+TC+MM and TC+MM with respect to their behavior in low speed and agility.

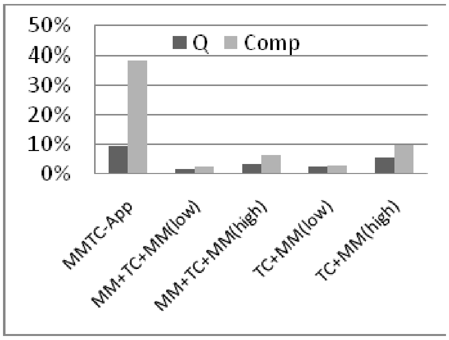

**Fig. 2.** Compression and overall quality achieved

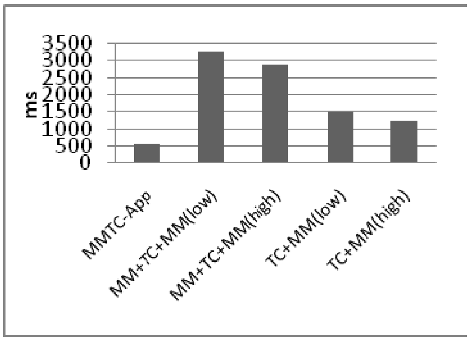

**Fig. 3.** Execution time

The second set of experiments concentrates on the efficiency of the proposed techniques. Fig. 3 illustrates the execution time required to run the proposed techniques over the Milano dataset. The proposed MMTC-App algorithm completes running in half second while the naïve approaches required a few seconds to perform their task (depending on the compression threshold set for the TC component).

## 5 Conclusion

MOD literature offers solutions to the problems of trajectory compression and map-matching succinctly, but none satisfies the combined problem of trajectory compression under network constrains.

In this paper, apart from straightforward solutions to the so-called MMTC problem, we have proposed a method for trajectory compression under network constraints. According to our results, our approximate solution turns out to be both efficient and effective, offering successful compression while at the same time retaining the quality of the output.

## References

1. Brakatsoulas, S., Pfoser, D., Salas, R., Wenk, C.: On Map-Matching Vehicle Tracking Data. In: Proc. 31st International Conference on Very Large Data Bases (VLDB) (2005)
2. Cao, H., Wolfson, O.: Nonmaterialized Motion Information in Transport Networks. In: Eiter, T., Libkin, L. (eds.) ICDT 2005. LNCS, vol. 3363, pp. 173–188. Springer, Heidelberg (2004)
3. Grünwald, P., Myung, I.J., Pitt, M.: Advances in Minimum Description Length: Theory and Applications. MIT Press, Cambridge (2005)
4. Johnson, D.B.: Efficient algorithms for shortest paths in sparse networks. Journal of the ACM 24(1), 1–13 (1977)

5. Kellaris, G., Pelekis, N., Theodoridis, Y.: Trajectory Compression under Network Constraints, UNIPI-INFOLAB-TR-2009-01, Technical Report Series, InfoLab, Univ. Piraeus (April 2009), http://infolab.cs.unipi.gr
6. 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)
7. Tiakas, E., Papadopoulos, A.N., Nanopoulos, A., Manolopoulos, Y.: Trajectory Similarity Search in Spatial Networks. In: Proc. 10th International Database Engineering and Applications Symposium (IDEAS) (2006)