Multiplexing Trajectories of Moving Objects

Kostas Patroumpas[1], Kyriakos Toumbas[1], and Timos Sellis[1,2]

[1] School of Electrical and Computer Engineering
National Technical University of Athens, Hellas
[2] Institute for the Management of Information Systems, R.C. "Athena", Hellas
{kpatro, timos}@dbnet.ece.ntua.gr, toumbask@gmail.com

Abstract. Continuously tracking mobility of humans, vehicles or merchandise not only provides streaming, real-time information about their current whereabouts, but can also progressively assemble historical traces, i.e., their evolving trajectories. In this paper, we outline a framework for online detection of groups of moving objects with approximately similar routes over the recent past. Further, we propose an encoding scheme for synthesizing an indicative trajectory that collectively represents movement features pertaining to objects in the same group. Preliminary experimentation with this multiplexing scheme shows encouraging results in terms of both maintenance cost and compression accuracy.

1 Motivation

As smartphones and GPS-enabled devices proliferate and location-based services penetrate into the market, managing the bulk of rapidly accumulating traces of objects' movement becomes all the more crucial for monitoring applications. Apart from effective storage and timely response to user requests, data exploration and trend discovery against collections of evolving trajectories seems very challenging. From detection of flocks [9] or convoys [4] in fleet management, to similarity joins [1] for car-pooling services, or even to identification of frequently followed routes [2,8] for traffic control, the prospects are enormous.

We have begun developing a stream-based framework for *multiplexing trajectories* of objects that approximately travel together over a recent time interval. Our perception is that a symbolic encoding for sequences of trajectory segments can offer a rough, yet succinct abstraction of their concurrent evolution. Taking advantage of inherent properties, such as heading, speed and current position, we can continuously report groups of objects with similar motion traces. Then, we may regularly construct an indicative path per detected group, which actually epitomizes spatiotemporal features shared by its participating objects.

Overall, such a scheme could be beneficial for:

- *Data compression*: collectively represent traces of multiple objects with a single "delegate" that suitably approximates their common recent movement.
- *Data discovery*: find trends or motion patterns from real-time location feeds.
- *Data visualization*: estimate significance of each multiplexed group of trajectories and illustrate its mutability across time (e.g., on maps).

– *Query processing*: utilize multiplexed traces for filtering when it comes to evaluation of diverse queries (range, k-NN, aggregates etc.) over trajectories.

We believe that our ongoing work fuses ideas from trajectory clustering [5] and path simplification [7], but proceeds even further beyond. Operating in a geostreaming context, not only can we identify important motion patterns in online fashion, but we may also provide concise summaries without resorting to sophisticated spatiotemporal indexing. Symbolic representation of routes was first proposed in [1] for filtering against trajectory databases. Yet, our encoding differs substantially, as it attempts to capture evolving spatiotemporal vectors using a versatile alphabet of tunable object headings instead of simply compiling timestamped positions in a discretized space. Finally, this scheme may be utilized in applications that handle motion data (navigation, biodiversity, radar etc.).

The remainder of this paper is organized as follows. In Section 2, we introduce a framework for multiplexing evolving trajectories in real time and explain the basic principles behind our encoding scheme. In Section 3, we report indicative performance results from a preliminary experimental validation of the algorithm. Section 4 concludes the paper with a brief discussion of perspectives and open issues for further investigation.

2 A Multiplexing Framework against Trajectory Streams

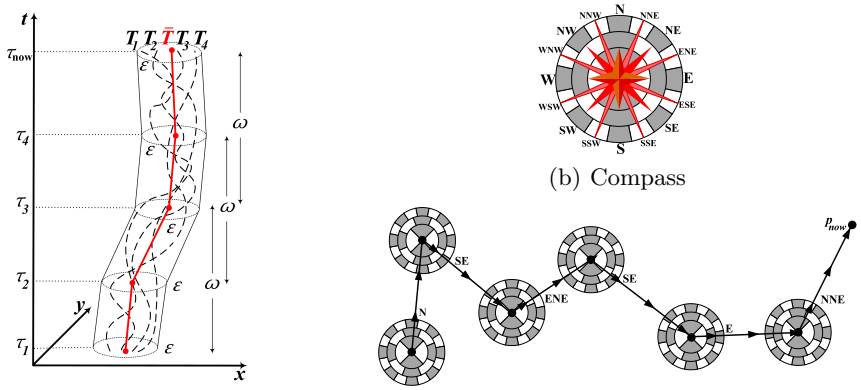
In this section, we first present the specifications of the problem and then outline a methodology for multiplexing similar trajectories, which effectively provides almost instant, yet approximate results.

2.1 Problem Formulation

Without loss of generality, trajectory T_o is abstracted as a sequence of pairs $\{\langle p_1, \tau_1\rangle, \langle p_2, \tau_2\rangle, \ldots, \langle p_{\mathtt{now}}, \tau_{\mathtt{now}}\rangle\}$ for a given moving object o. Positions $p_k \in \mathbb{R}^d$ in Euclidean space have d-dimensional coordinates measured at discrete, totally ordered timestamps $\tau_k \in \mathbb{T}$, hence $o(\tau_k) \equiv p_k$. Note that $\mathbb{T}$ is regarded as an infinite set of discrete time instants with a total order $\leq$. Then:

Definition 1. *Trajectories of two objects o_i and o_j are considered* similar *along interval ω up to current time $\tau_{now} \in \mathbb{T}$, iff $L_2(o_i(t), o_j(t)) \leq \epsilon, \forall\, t \in (\tau_{now}-\omega, \tau_{now}]$, where ϵ is a given tolerance parameter and L_2 the Euclidean distance norm.*

Hence, pairs of concurrently recorded locations from each object should not deviate more than ϵ during interval ω. This notion of similarity is confined within the recent past and does not extend over the entire history of movement. However, it can be easily generalized for multiple objects with pairwise similar trajectory segments (Fig. 1a). Given specifications for proximity in space (within distance ϵ) and simultaneity in time (over range ω), our objective is not just to identify such groups of trajectories, but also to incrementally refresh them periodically (every β time units) adhering to the *sliding window* paradigm [6]. More concretely, a framework for online trajectory multiplexing must:



(a) Similar trajectories (b) Compass (c) Motion signature: N-SE-ENE-SE-E-NNE

Fig. 1. Orientation-based encoding of streaming trajectories.

(i) distinguish objects into groups $\{g_1, g_2, \ldots\}$, each containing synchronized, pairwise similar trajectories during interval ω given a tolerance ϵ.

(ii) create an indicative *"delegate"* trajectory $\bar{T}_k$ for each group g_k with more than n members. For any sample point $\bar{o}(t) \in \bar{T}_k, \forall\ t \in (\tau_{\texttt{now}} - \omega, \tau_{\texttt{now}}]$, it holds that $L_2(o_i(t), \bar{o}(t)) \leq \epsilon, \forall\ o_i \in g_k$.

(iii) insert, remove or adjust groups regularly (at execution cycles with period β) in order to reflect changes in objects' movement.

2.2 Trajectory Encoding

Checking similarity of trajectory segments according to their timestamped positions soon becomes a bottleneck for escalating numbers of moving objects or wider window ranges. To avoid this, we opt for an approximative representation of traces based on consecutive velocity vectors that end up at the current location of each respective object (Fig. 1c). Every vector is characterized by a symbol that signifies the orientation of movement using the familiar notion of *compass* (Fig. 1b, for movement in $d = 2$ dimensions), which roughly exemplifies an object's course between successive position messages.

Effectively, compass resolution α determines the degree of motion smoothing; when $\alpha = 4$, orientation symbols {N, S, E, W} offer just a coarse indication, but finer representations are possible with $\alpha = 16$ symbols (Fig. 1c) or more. Instead of original positions, only the last $\lceil \frac{\omega}{\beta} \rceil$ symbols and speed measures need be maintained per trajectory thanks to the sliding window model, thus offering substantial memory savings. Typically, once the window slides at the next execution cycle, an additional symbol (marking motion during the latest β timestamps) will be appended at the tail of this FIFO sequence, while the oldest one (i.e., at the head) gets discarded.

2.3 Group Detection

Symbolic sequences are more amenable to similarity checks since they act as *motion signatures*. Presently, we identify objects with common signatures through a hash table. Objects with identical symbolic sequences might have almost "parallel" courses, but can actually be very distant from each other. So, the crux of our approach is this:

Proposition 1. *Objects with identical signatures that are currently within ϵ distance from each other, most probably have followed similar paths recently.*

Therefore, identifying groups of at least n objects with a common signature can be performed against their current locations $p_{\texttt{now}}$ through a point clustering technique. We provisionally make use of DBSCAN [3] to detect groups of similar trajectories with proximal current positions. Afterwards, a delegate path $\bar{T}_k$ per discovered cluster with sufficient membership ($\geq n$ objects) can be easily created. Reconstruction of $\bar{T}_k$ starts by calculating centroid $\bar{o}(\tau_{\texttt{now}})$ of its constituent $p_{\texttt{now}}$ locations; apparently, this will be the point at the tail of the sequence. In turn, preceding points are derived after successively averaging respective speeds retained in participating motion signatures. This can be easily accomplished by simply rewinding the symbolic sequence backwards up to its head, i.e., reversely visiting all samples within the sliding window frame.

3 Preliminary Evaluation

To assess the potential of our framework for data reduction and timely detection of trends, we have conducted some preliminary simulations against synthetic trajectories. Next, we present the experimental settings and we discuss some indicative results concerning performance and approximation quality.

Experimental Setup. We generated traces of 10 000 vehicles circulating at diverse speeds along the road network of greater Athens (area $\sim$250 km^2). After calculating shortest paths between randomly chosen network nodes (i.e., origin and destination of objects), we took point samples at 200 concurrent timestamps along each such route. Typically, most trajectories originate from the outskirts of the city, pass through the center and finish up in another suburb.

Table 1. Experiment parameters.

Parameter	Values		
Number N of objects	**10 000**		
Window range ω (in timestamps)	10	**20**	50
Window slide β (in timestamps)	2	**4**	10
Tolerance ϵ (in meters)	**100**, 200		
Cluster threshold n	10, **20**, 50, 100		
Compass resolution α	8, **16**, 32		

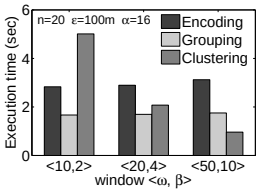


Fig. 2. Per stage cost.

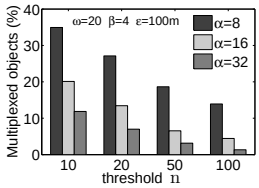


Fig. 3. Multiplex effect.

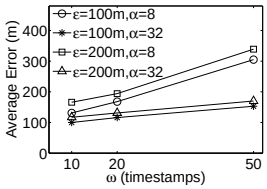


Fig. 4. Result quality.

Algorithms have been implemented in C++ and experiments were performed on a conventional laptop machine with an Intel Core 2 Duo CPU at 2.4 GHz and 4GB RAM. All figures show calculated averages of the measured quantities over 200 time units. Table 1 summarizes experimentation parameters and their respective ranges; the default value is shown in bold.

Experimental Results. Admittedly, the proposed algorithm performs a lossy approximation susceptible to errors. The primary causes are crude compass resolutions or speed variations amongst objects placed in the same group. By and large, our empirical validation confirms this intuition, and also expectations for prompt detection of groups.

As Fig. 2 indicates, execution cost per cycle fulfils real-time objectives for varying window specifications, with almost stable overhead for encoding and grouping. Note that the encoding phase is only marginally affected when increasing the window range, although more positions per object must be handled at each execution cycle (every β timestamps). The cost of grouping remains practically stable, as it basically depends on the total count N of monitored objects. In contrast, the clustering overhead fluctuates, but drops sharply for wider windows as less objects tend to share motion signatures for too long. For shorter ω, more objects appear to move together lately and thus create more candidate groups; accordingly, the clustering cost escalates as it requires distance calculations among all members of each group.

Our next experiment attempts to appraise how effective this method is. Figure 3 plots the multiplexing degree, i.e., the fraction of objects assigned into identified groups of sufficient size. Clearly, distinguishing important trends is sensitive to threshold n. In case that membership into a group falls below limit n, its trajectories are not multiplexed at all. Besides, approximation gets more pronounced with a coarser resolution α. But for finer resolutions, less trajectory matchings are identified, as objects tend to retain particular features of their course and cannot easily fit into larger groups.

Still, average error between a delegate path and its contributing trajectories is tolerable (Fig. 4), especially for less smoothed signatures (larger α). When probing longer intervals, this deviation may well exceed the desired ϵ. This phenomenon must be attributed to the relaxed notion of "density-reachability" in DBSCAN [3], which does not dictate that all cluster members be within distance

ϵ from its centroid, but only pairwise. For less detailed trajectory representations (i.e., encodings based on small resolution α), this deviation propagates backwards when probing retained sequences to reconstruct a delegate path, so error may exacerbate ever more. However, the algorithm seems to achieve more reliable approximations in case of finer motion signatures, particularly for $\epsilon = 200$m.

4 Outlook

In this work, we set forth a novel approach for multiplexing trajectory features that get frequently updated from streaming positions of moving objects. We have been developing a methodology for detecting groups of objects that approximately travel together over the recent past. Thanks to an encoding scheme based on velocity vectors, this process can be carried out in almost real time with tolerable error, as our initial empirical study indicates.

We keep working on several aspects of this technique and we soon expect more gains in terms of scalability and robustness. In particular, we intend to take advantage of intermediate symbolic sequences in order to improve clustering with higher representativeness and less recalculations, even in presence of massive positional updates. We also plan to further investigate the grouping phase with more advanced schemes, like those used in string and sequence matching. Last but not least, it would be challenging to study this technique as an optimization problem, trying to strike a balance between similarity tolerance and resolution of the encoding scheme.

References

1. P. Bakalov, M. Hadjieleftheriou, E. Keogh, and V.J. Tsotras. Efficient Trajectory Joins using Symbolic Representations. In *MDM*, pp. 86-93, 2005.
2. Z. Chen, H.T. Shen, and X. Zhou. Discovering Popular Routes from Trajectories. In *ICDE*, pp. 900-911, 2011.
3. M. Ester, H.-P. Kriegel, J. Sander, and X. Xu. A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise. In *KDD*, pp. 226-231, 1996.
4. H. Jeungy, M.L. Yiu, X. Zhou, C.S. Jensen, and H.T. Shen. Discovery of Convoys in Trajectory Databases. *PVLDB*, 1(1):1068-1080, 2008.
5. J. Lee, J. Han, and K. Whang. Trajectory Clustering: a Partition-and-Group Framework. In *ACM SIGMOD*, pp. 593-604, 2007.
6. K. Patroumpas and T. Sellis. Maintaining Consistent Results of Continuous Queries under Diverse Window Specifications. *Information Systems*, 36(1):42-61, 2011.
7. M. Potamias, K. Patroumpas, and T. Sellis. Sampling Trajectory Streams with Spatiotemporal Criteria. In *SSDBM*, pp. 275-284, 2006.
8. D. Sacharidis, K. Patroumpas, M. Terrovitis, V. Kantere, M. Potamias, K. Mouratidis, and T. Sellis. Online Discovery of Hot Motion Paths. In *EDBT*, pp. 392-403, 2008.
9. M. Vieira, P. Bakalov, and V.J. Tsotras. On-line Discovery of Flock Patterns in Spatio-temporal Data. In *ACM GIS*, pp. 286-295, 2009.