Trajectory Patterns Based on Segment-Cutting Clustering

Luis Cabrera-Crot[1], Mónica Caniupán[2], Andrea Rodríguez[1], and Diego Seco[1]

[1] Universidad de Concepción, Chile
[2] Universidad del Bío-Bío, Chile
luiscabrera@udec.cl, mcaniupan@ubiobio.cl, andrea@udec.cl, dseco@udec.cl

Abstract. Trajectory patterns characterize similar behaviors among trajectories, which play an important role in applications such as urban planning, traffic congestion control, and studies of animal migration and natural phenomena. In this paper we model trajectories as a sequence of line segments that represent the steady movement of an object along time. We use a segment-clustering process to group trajectories' segments and partial segments based on their temporal and spatial closeness. Then, it defines a *trajectory pattern* that results from the aggregation of segment clusters, aggregation that is not only based on spatial and temporal sequentiality, but also on the compatibility of trajectories in each segment cluster. The experimental assessment shows the effectiveness of the method.

Keywords: Pattern recognition · data mining · trajectories · spatio-temporal databases.

1 Introduction

Due to the current advances in sensor networks, wireless technologies, and RAID-enabled ubiquitous computing, data about moving objects (also called trajectories) is an example of massive data relevant in many real applications [4]. According to [4], a trajectory pattern is a set of individual trajectories that visit the same sequence of places with similar travel times. A classical representation of trajectories is given by a sequence of time-stamped locations in a, typically, 2D space trajectory clustering algorithms aim at grouping similar trajectories (or part of trajectories) according to similarity measures. The main problem of trajectory aggregation is the imprecision of spatio-temporal data [8], which makes more challenging the definition of similarity measures that compare different trajectories. There are several notions of trajectory similarity measures [13] being the Euclidean Distance Measure, in its simplest version, robust for trajectory shifts.

We propose a method to derive *trajectory patterns*, which correspond to patterns of trajectories with similar behavior during a time interval. To obtain the trajectory patterns, we first group complete or partial segments of trajectories that have a similar behavior in space and time. Then we aggregate segment clusters considering a compatibility measure. A similarity measure in terms of a distance function is sensible to the visual comparison, is computationally affordable, and is complemented with strategies to avoid sensitivity to sampling rate and noise. We focus here on a similarity measure

defined in terms of Euclidean distance for free movements, but it is also possible to consider different distance functions when trajectories are constrained to other types of spaces such as networks. Using the Euclidean distance, two segments of trajectories can be considered similar if they coincide (approximately) in their starting and ending points. In addition, two segments can be also considered similar if they coincide in some parts of the segments [6, 12]. But it is not enough to have spatial similarity; temporal similarity must ensure that trajectories also occur close in time to capture the time-sensitivity of the similar behavior.

There exist several proposals for the extraction of trajectory patterns [6, 5, 1, 8, 3, 13, 12, 14]. They mostly vary in terms of their similarity functions and the capacity of making incremental extraction of patterns. They consider similarity between sampling points of trajectories (i.e., segments of trajectories), without considering that, between sampling points, trajectories can be partially similar. According to [14] our work falls in the category of trajectory clustering. We argue in this work that clustering not only whole segments, but also portions of segments, captures similarity between trajectories that could otherwise be underestimated.

We compare our proposal with the work presented in [6] which proposes the *partition and group* framework that allows the discovery of common sub-trajectories from a trajectory database. First, it partitions trajectories into a set of line segments and then it groups similar complete line segments. Trajectories are partitioned according to characteristics points. Then, they generate a representative trajectory for each cluster. The algorithm they propose to obtain clusters is based on a suitable *distance function*, which is composed of the *perpendicular distance*, the *parallel distance*, and the *angle distance*. The algorithm works over a set of line segments and classifies them as part of a cluster or as a noise. Only clusters with a cardinality over a threshold are considered valid. Finally, the algorithm computes a representative trajectory for every cluster. The main difference of this previous work with respect to our proposal is that they cluster complete line segments, while we are able to cluster portions of line segments. In addition, we also include time in the comparison of segments and, when obtaining clusters, we do not exclude a segment as a noise, but we keep portions of segments that do not match a cluster, and use them in future computations of clusters. Only at the end of the process of clustering, we eliminate noise segments. In this way, our implementation is not strict with possible noise segments as the one proposed in [6].

2 Trajectory Pattern

A trajectory of an object is represented based on points with temporal dimension [5]. A point SPoint is a tuple (x, y) on the Euclidean space and a temporal point TPoint is a tuple (SPoint, t) where t is a time instant. For simplicity, we also refer as a time interval a tuple $\mathsf{TT} = [t_s, t_e]$, where $t_s, t_e \in \mathbb{R}$ and $t_e \geq t_s$. A segment S of an object's trajectory is a pair of $\mathsf{TPoints}$ $[\mathsf{P}_s, \mathsf{P}_e]$, where $\mathsf{P}_s = ((x_s, y_s), t_s)$, $\mathsf{P}_e = ((x_e, y_e), t_e)$, and $t_e > t_s$. The definition of trajectory is based in the definition given in [9]. A trajectory T is a tuple composed of an identification and a sequence of consecutive segments $\langle id, \mathsf{S}_1, \mathsf{S}_2, \ldots, \mathsf{S}_m \rangle$ that describes the path of an object and where, for every pair of consecutive segments $\mathsf{S}_i = [\mathsf{P}_s, \mathsf{P}_e]$ and $\mathsf{S}_j = [\mathsf{P}'_s, \mathsf{P}'_e]$ in T, $\mathsf{P}_e.x = \mathsf{P}'_s.x, \mathsf{P}_e.y = \mathsf{P}'_s.y$

and $\mathsf{P}'_s.t - \mathsf{P}_e.t \leq \epsilon$, where$\epsilon$ is the length of the interval in which an object may stop at a particular TPoint. We call *sub-trajectory* Ts to a subset of consecutive segments of an object's trajectory T. A sub-trajectory is also a trajectory. With some abuse of notation, we say that $\mathsf{Ts} \subseteq \mathsf{T}$ if Ts is a sub-trajectory of T.

Notice that a trajectory is a sequence of segments, which differs from the classical definition in terms of a sequence of TPoints. We explicitly define trajectories as a sequence of segments to emphasize the treatment of the segments in the clustering process. Also, a sub-segment is also a segment. At a physical level, however, we can avoid the duplication of endpoints in the definition of segments.

Let T be a trajectory, S, S_1, and S_2 be segments, and P and P′ be SPoints, the following are useful operators.

- $\mathsf{SD}(\mathsf{P}, \mathsf{P}')$: it denotes the Euclidian spatial distance between two SPoints P and P′. If points are on a network, then this should be the distance on the network. Overloading this operator, $\mathsf{SD}(\mathsf{S}, \mathsf{P})$ is the shortest Euclidian distance from SPoint P to segment S (i.e., the length of the perpendicular line from P to S).
- $\mathsf{TD}(\mathsf{P}, \mathsf{P}') = |\mathsf{P}.t - \mathsf{P}'.t|$: it is the temporal distance between two TPoints P and P′.
- $\mathsf{ID}(\mathsf{T})$: it returns the identification of a trajectory T.
- $\mathsf{ID}(\mathsf{S})$: it returns the identification of the trajectory to which the segment S belongs to.
- $\mathsf{START}_s(\mathsf{T})$: it returns the starting segment of a trajectory T.
- $\mathsf{END}_s(\mathsf{T})$: it returns the ending segment of a trajectory T.
- $\mathsf{START}_p(\mathsf{S})$: it returns the starting point of a segment S.
- $\mathsf{START}_p(\mathsf{T})$: it returns the starting point of a trajectory T, this is, $\mathsf{START}_p(\mathsf{T}) = \mathsf{START}_p(\mathsf{START}_s(\mathsf{T}))$.
- $\mathsf{END}_p(\mathsf{S})$: it returns the ending point of a segment S.
- $\mathsf{END}_p(\mathsf{T})$: it returns the ending point of a trajectory T.
- $\mathsf{LENGTH}(\mathsf{T})$: it returns the number of segments of T.
- $\mathsf{ANGLE}(\mathsf{S}_1, \mathsf{S}_2)$: it returns the formed angle between segments S_1 and S_2.

2.1 Relations between Trajectory Segments

Intuitively, two segments S_1 and S_2 will be *totally related* if their initial and ending points are close, spatially and temporally, and they have the same direction, this is, the angle between the segments is less than $90°$. We formalize this in the following definition.

Definition 1. Let S_1 and S_2 be two trajectory segments from different trajectories. S_1 and S_2 are *totally related* if and only if:

- $\mathsf{SD}(\mathsf{START}_p(\mathsf{S}_1), \mathsf{START}_p(\mathsf{S}_2)) \leq \Delta_s$
- $\mathsf{TD}(\mathsf{START}_p(\mathsf{S}_1), \mathsf{START}_p(\mathsf{S}_2)) \leq \Delta_t$
- $\mathsf{SD}(\mathsf{END}_p(\mathsf{S}_1), \mathsf{END}_p(\mathsf{S}_2)) \leq \Delta_s$
- $\mathsf{TD}(\mathsf{END}_p(\mathsf{S}_1), \mathsf{END}_p(\mathsf{S}_2)) \leq \Delta_t$
- $\mathsf{ANGLE}(\mathsf{S}_1, \mathsf{S}_2) < 90^o$. □

(a) (b) (c) (d)

Fig. 1. Partial relations between two segments S_1 and S_2 from different trajectories

Two segments are *partially related* if they are not totally related, they have the same direction, and one extreme point of a segment is close, spatially and temporally, to some point of the other segment. It is possible to have different forms of partial relations between two segments. We have considered four forms of partial relations.

Definition 2. Let S_1 and S_2 be two trajectory segments that are not totally related. S_1 and S_2 are *partially related* if and only if they satisfy one of the following partial relations, which are illustrated in Figure 1:

- Case 1 (Figure 1(a)). In this case: (i) The starting points of segments S_1 and S_2 are spatially and temporally close to each other. This is, they satisfy the distances established by threshold Δ_s and Δ_t. (ii) The ending point of segment S_1 is not close to any point of segment S_2, but the ending point of segment S_2 is close to some point (different from an endpoint) of segment S_1.
- Case 2 (Figure 1(b)). In this case: (i) The ending points of segments S_1 and S_2 are spatially and temporally close to each other. (ii) The starting point of segment S_1 is not close to any point of segment S_2, but the starting point of segment S_2 is close to some point (different from an endpoint) of segment S_1.
- Case 3 (Figure 1(c)). In this case: Neither the starting and ending point of segment S_1 are close to any point of segment S_2, but the starting and ending point of segment S_2 are close to some point (different from an endpoint) of segment S_1.
- Case 4 (Figure 1(d)). In this case: (i) The starting point of segment S_1 is close to some point of segment S_2. (ii) The ending point of segment S_1 is not close to any point of S_2.(iii) The starting point of segment S_2 is not close to any point of segment S_1. (iv) The ending point of segment S_2 is close to some point (different from an endpoint) of segment S_1. □

2.2 Segment Clustering

We use the concept of *segment cluster* to refer to a set $\mathsf{SC} = \{S_1, S_2, \ldots, S_n\}$ of not necessarily consecutive segments or sub-segments from different trajectories that are totally related to each other. The process of generating segment clusters that are not totally related is as follows. Consider a trajectory T_i with segments $S_1, \ldots, S_n$: (i) If a segment S_i of T_i is not totally related to any of the segments in the already computed segment clusters, but it is partially related to some segment S_j in a cluster c_j, then: (i) We first identify the type of partial relation between segments S_i and S_j, according to Definition 2. (ii) A new segment S_s is generated and corresponds to the projection of

the shortest segment on the longest segment. Note that this implies to calculate a TPoint that splits the original segment. Let us assume that S_i is shorter than S_j then, segment S_s corresponds to the projection of S_i over S_j. (iii) Cluster c_j is updated and contains segments S_s and S_i, a new cluster is generated containing the residual segment $S_j - S_s$.

The dynamic computation of segment clusters refers to the capability of adding new trajectory segments when we had already computed a set of segment clusters, without starting the whole process of segment clustering. The segments of a new trajectory can be added into existing segment clusters or can form new clusters.

We introduce the following notation over segment clusters that is useful in the next definition of trajectory patterns. Let cl be a segment cluster composed of a set S of segments. Then, (i) $\mathsf{CONVEX}(cl)$ denotes the convex hull over the TPoints that form the segments of the cluster; (ii) $\mathsf{TInterval}(cl)$ denotes the time interval $[t_s, t_e]$ such that $t_s = \min_{\forall s_i \in S}\{START_p(s_i).t\}$ and $t_e = \max_{\forall s_i \in S}\{\mathsf{END}_p(s_i).t\}$; and (iii) $\mathsf{IDS}(cl) = \{\mathsf{ID}(s_i) | s_i \in cl\}$ is the set of trajectories's ids that are part of the cluster. Given a segment cluster cl, we call the *pattern* of cl, the tuple of the form (geo, tt, ids), with $geo = \mathsf{CONVEX}(cl)$, $tt = \mathsf{TInterval}(cl)$, and $ids = \mathsf{IDS}(cl)$. Notice that this pattern is essentially a geometric approximation of the segment cluster. For simplicity, we use $cp.a$ with $a \in [geo, tt, ids]$ to access each of the elements of the pattern cp.

2.3 Trajectory Pattern

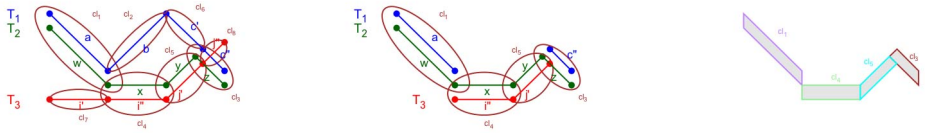
Segment clusters can be aggregated to form *trajectory patterns* if, they have at least two segments, their areas intersect, their share a temporal interval, and they satisfy a compatibility threshold of common trajectories. Trajectory patterns are patterns that combine two or more patterns extracted from single segment clusters. We first introduce the concept of compatible patterns, useful to define trajectory patterns.

Definition 3. Let cp_1, cp_2 be two patterns, with $cp_1 \neq cp_2$. Patterns are compatible if:

1. $cp_1.geo \cap_{spatial} cp_2.geo \neq \emptyset$, where $\cap_{spatial}$ denotes geometric intersection.
2. $cp_1.tt \cap_{time} cp_2.tt \neq \emptyset$, where $\cap_{time}$ denotes temporal intersection.
3. $\frac{|cp_1.ids \cap cp_2.ids|}{|cp_1.ids \cup cp_2.ids|} >= \Delta_j$, where Δ_j is the threshold of the number of common trajectories. □

Definition 4. Let cp_1 and cp_2 be two compatible patterns. A trajectory pattern tp for cp_1 and cp_2 is a tuple of the form (geo, tt, ids), where geo is the geometric union of $cp_1.geo$ and $cp_2.geo$, tt is the time union of $cp_1.tt$ and $cp_2.tt$, and ids the set union of $cp_1.ids$ and $cp_2.ids$. Because trajectory patterns are patterns, we can recursively apply this definition to aggregate more than two patterns. □

The extraction of trajectory patterns starts with an empty set of trajectory patterns TP, a set C of segment clusters and its corresponding patterns C^* and, iteratively, finds compatible clusters to aggregate to trajectory patterns. At the end of this iterative process, TP is empty if it does not find compatible clusters or contains patterns composed of at least two patterns in C^*. This concept is similar to the concept of *macro clusters* presented in [7], where, unlike our proposal, patterns of trajectories are computed considering only complete segments of trajectories and only spatial closeness.



(a) Segment clusters (b) Segment clusters to be aggregated (c) Convex hull

Fig. 2. Aggregation of segment clusters

Example 1. Consider the segment clusters in Figure 2(a). Figure 2(b) shows the segment clusters that can be aggregated from the set of segment clusters, this is, clusters cl_1, cl_3, cl_4 and cl_5. Clusters cl_2, cl_6, cl_7 and cl_8 are discarded because they have only one segment. Since the areas and time intervals of clusters cl_1, cl_3, cl_4 and cl_5 intersect (see Figure 2(b)), they may be aggregated if they satisfy a specific compatibility threshold (Δ_j). □

Intuitively, a set of *trajectory patterns* corresponds to the aggregation of trajectories that have similar behavior and satisfy a compatibility threshold, which indicates a percentage of common trajectories. Note that by applying Definition 4 iteratively, it is possible to get a set of trajectory patterns, each of them with possible different levels of compatibility.

3 Evaluation and Comparison with the State of the Art

To show that our method is effective to find trajectory patterns we use both real and synthetic datasets. To evaluate effectiveness we compare our method with the method, called Traclus, proposed in [6], which does not consider the temporal distance between segments, and also compare complete segments but not sub-segments of trajectories. In this case we use synthetic datasets. We also present a qualitative evaluation of our method that shows some features that are not supported by the baseline. To do so, we use a real dataset that corresponds to truck trajectories in Greece that has been used in [10, 11, 2] to determine trajectory patterns.

3.1 Quantitative Comparison with the State of the Art

We generated 1,000 trajectories with the Brinkhoff generator[3]. Then, we randomly selected 100 of them and created 50 copies of each one. Thus, these 100 trajectories correspond to the ground truth of trajectory patterns that a clustering algorithm should recognize as patterns. Figure 3 shows the comparison between the baseline method presented in [6] and our proposed method.

[3] `http://chorochronos.datastories.org/?q=node/51`

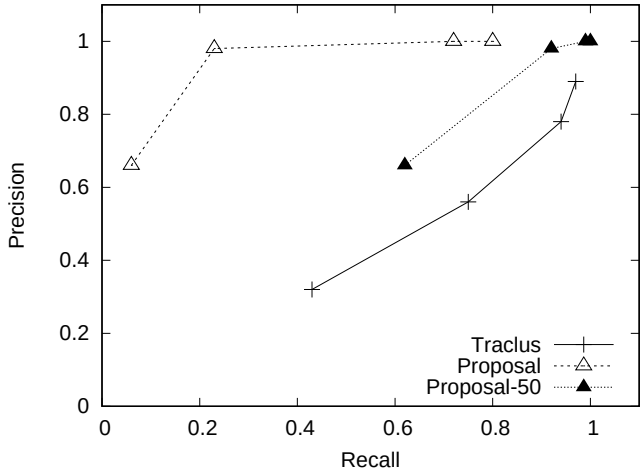


Fig. 3. Comparison with the state of the art

For our method we graphic two lines, one unfiltered and another filtering those segments that have less than 50 elements. For each method we show a line and not a single point because, to consider a segment of the ground truth equal to a segment obtained by an algorithm, a tolerance may be applied to ignore precision issues. For each method, the point located on the leftmost endpoint is the one with tolerance 0, and then it increases to 10, 100 and 250. Obviously, the greater the tolerance, we are considering more distant segments as equal. Even so, they are small tolerance values.

In the chart, we show precision-recall values, which are standard in the information retrieval community. In our domain, precision may be interpreted as the portion of the ground truth that an algorithm can retrieve, whereas recall would be the portion of the retrieved patterns that are indeed in the ground truth. Hence, the ideal method would be the one that gets earlier to the upper right corner (precision 1 and recall 1). As it can be seen, our method with filtering is the one that dominates the others by being the only one to get close to that point. Regarding our unfiltered method, although it obtains a precision of 1, it is at the expense of the recall (that is, it recovers all the reference, but also a lot of noise). The baseline is improving by increasing the tolerance in the comparison, but it does not reach our method with filtering.

3.2 Qualitative evaluation of our proposal

The trucks dataset contains GPS (with reference system GGRS87) positions of 50 trucks transporting concrete in the Athens area between August and September of 2002. This dataset contains temporal information, and the time interval to take the samples is 30 seconds. The set contains 111,927 segments, conforming 276 trajectories, with an average of 406 segments per trajectory. We use a spatial threshold Δ_s of 420 units, a temporal threshold Δ_t of 209,950 units of time, and different com-

patibility thresholds. These thresholds were chosen because they are the values that minimize a ClusterCost function defined for a set of segment clusters C and a set R of isolated segments not included in any $c_i \in C$. Function ClusterCost is defined as: $\mathsf{ClusterCost} = \mathsf{ClusterWeight} + \mathsf{NoiseWeight}$, where:

$$\mathsf{ClusterWeight} = \begin{cases} \sum_{c_i \in C} \frac{\mathsf{AREA}(\mathsf{CONVEX}(c_i))}{|\mathsf{IDS}(c_i)|} & |C| > 0 \\ 0 & \text{otherwise} \end{cases}$$

$$\mathsf{NoiseWeight} = \begin{cases} \sum_{r_i \in R} \mathsf{LENGTH}(r_i)^2 * \pi & |R| > 0 \\ 0 & \text{otherwise} \end{cases}$$

Figure 4(a) shows the 17,855 segment clusters obtained by our algorithms with $\Delta_s = 420$ and $\Delta_t = 209,950$.

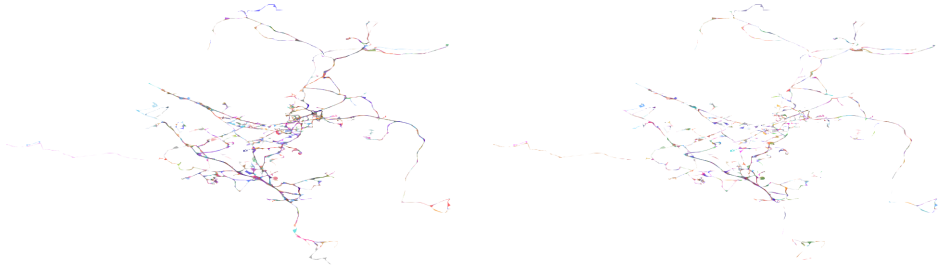
(a) Segment clusters (b) Trajectory patterns for $\Delta_j = 0\%$

Fig. 4. Segment clusters and trajectory patterns for the trucks dataset

Trajectory patterns for the trucks dataset We use different compatibility thresholds to compute trajectory patterns considering the 17,855 segment clusters in Figure 4(a). Consider $\Delta_j = 0\%$, this is, there is not a constraint over a percentage of common trajectories. In this case, all the segment clusters whose areas and time intervals intersect form a unique trajectory pattern. Figure 4(b) shows the result of the experiment, where there are six trajectory patterns and two non-aggregate clusters.

A value for Δ_j greater than 50% provides a lower proportion of trajectory patterns with respect to the total of trajectory patterns and non-aggregate clusters. Figure 5(a) shows the $3,064$ trajectory patterns with $\Delta_j = 60\%$. 5(b) illustrates the result for $\Delta_j = 100\%$, in which case there are $1,963$ trajectory patterns.

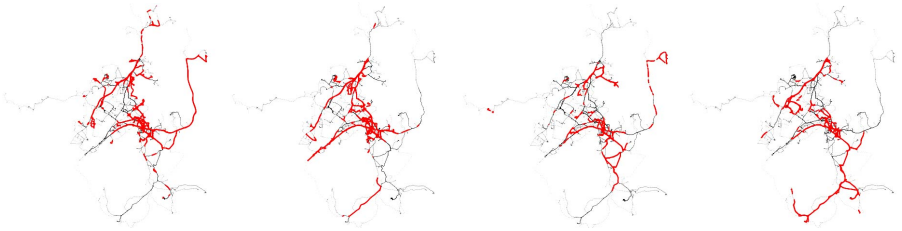
Trajectory patterns under different temporal intervals We also compute trajectory patterns for different periods of time. They were obtained considering the following parameters $\Delta_s = 420$ units, $\Delta_t = 209,950$ units of time, and $\Delta_j = 20\%$ (compatibility threshold). We consider different days of the week and weekends, and we can conclude that during the week, trucks remain mostly in the central zone in the afternoons. Figure



(a) $\Delta_j = 60\%$ (b) $\Delta_j = 100\%$

Fig. 5. Trajectory patterns for different Δ_j

6 shows the trajectory patterns for September 11 (Wednesday) to September 14 (Saturday) of 2002. This kind of finding would not be possible with the algorithm in [6] as it does not consider time.



(a) September 11 (b) September 12 (c) September 13 (d) September 14

Fig. 6. Trajectory patterns for days of September of 2002

4 Conclusions and Future Work

We presented a new concept of *trajectory pattern* that corresponds to the aggregation of compatible segment clusters from trajectories. To compute segment clusters, we consider the spatial and temporal distance between segments and sub-segments of trajectories, and also the angle of the segments. Aggregation of segment clusters is possible if the clusters satisfy a compatibility threshold, have more than one segment, and their areas and time intervals intersect. Our method allows the dynamic computation of segment clusters, and process the called *macro clustering* presented [7]. The experimentation we reported shows that the method is effective for computing trajectory patterns. For future work, we would like to optimize the algorithms for segment clustering and trajectory patterns detection. To do so, we can consider indexing structures in space and time.

Acknowledgements

Mónica Caniupán and Luis Cabrera-Crot are partially funded by DIUBB [181315 3/R], and the Algorithms and Databases Research Group [160119/EF]. Andrea Rodríguez is partially funded by Fondecyt [1170497], and the Complex Engineering Systems Institute (CONICYT: FBO16). Diego Seco is partially funded by Fondecyt [1170497].

References

1. Andrienko, N., Andrienko, G.: Spatial generalization and aggregation of massive movement data. IEEE Trans. Vis. Comput. Graph. **17**(2), 205–219 (2011)
2. Frentzos, E., Gratsias, K., Pelekis, N., Theodoridis, Y.: Algorithms for nearest neighbor search on moving object trajectories. Geoinformatica **11**(2), 159–193 (2007)
3. Giannotti, F., Nanni, M., Pedreschi, D.: Efficient mining of temporally annotated sequences. In: Proc. of the Sixth International Conference on Data Mining. pp. 348–359 (2006)
4. Giannotti, F., Nanni, M., Pinelli, F., Pedreschi, D.: Trajectory pattern mining. In: Proc. of the 13th International Conference on Knowledge Discovery and Data Mining. pp. 330–339 (2007)
5. Hung, C.C., Peng, W.C., Lee, W.C.: Clustering and aggregating clues of trajectories for mining trajectory patterns and routes. The VLDB Journal **24**(2), 169–192 (2015)
6. Lee, J.G., Han, J., Whangl, K.Y.: Trajectory clustering: a partition-and-group framework. In: Proc. of the SIGMOD Conference. pp. 593–604 (2007)
7. Li, Z., Lee, J.G., Li, X., Han, J.: Incremental clustering for trajectories. In: Proc. of the 15th International Conference on Database Systems for Advanced Applications. pp. 32–46 (2010)
8. Meratnia, N., de By, R.: Aggregation and comparison of trajectories. In: Proc. of the Tenth International Symposium on Advances in Geographic Information Systems. pp. 49–54 (2002)
9. Orlando, S., Orsini, R., Raffaetà, A., Roncato, A., Silvestri, C.: Trajectory data warehouses: Design and implementation issues. Journal of Computing Science and Engineering **1**(2), 211–232 (2007)
10. Panagiotakis, C., Pelekis, N., Kopanakis, I., Ramasso, E., Theodoridis, Y.: Segmentation and sampling of moving object trajectories based on representativeness. IEEE Trans. on Knowl. and Data Eng. **24**(7), 1328–1343 (2012)
11. Pelekis, N., Kopanakis, I., Kotsifakos, E., Frentzos, E., Theodoridis, Y.: Clustering uncertain trajectories. Knowledge and Information Systems **28**(1), 117–147 (2011)
12. Sankararaman, S., Agarwal, P.K., Mølhave, T., Pan, J., Boedihardjo, A.P.: Model-driven matching and segmentation of trajectories. In: Proc. of the 21st International Conference on Advances in Geographic Information Systems. pp. 234–243 (2013)
13. Wang, H., Su, H., Zheng, K., Sadiq, S., Zhou, X.: An effectiveness study on trajectory similarity measures. In: Proc. of the Twenty-Fourth Australasian Database Conference. pp. 13–22 (2013)
14. Zheng, Y.: Trajectory data mining: An overview. ACM TIST **6**(3), 29:1–29:41 (2015)