A Clustering-based Approach for Discovering Interesting Places in Trajectories

Andrey Tietbohl Palma
Instituto de Informática, UFRGS, Brazil
andrey@inf.ufrgs.br

Vania Bogorny
Hasselt University & Transnational University of Limburg, Belgium
vania.bogorny@uhasselt.be

Bart Kuijpers
Hasselt University & Transnational University of Limburg, Belgium
bart.kuijpers@uhasselt.be

Luis Otavio Alvares
Instituto de Informática, UFRGS, Brazil & Hasselt University, Belgium
alvares@inf.ufrgs.br

ABSTRACT

Because of the large amount of trajectory data produced by mobile devices, there is an increasing need for mechanisms to extract knowledge from this data. Most existing works have focused on the geometric properties of trajectories, but recently emerged the concept of semantic trajectories, in which the background geographic information is integrated to trajectory sample points. In this new concept, trajectories are observed as a set of *stops* and *moves*, where stops are the most important parts of the trajectory. Stops and moves have been computed by testing the intersections of trajectories with a set of geographic objects given by the user. In this paper we present an alternative solution with the capability of finding interesting places that are not expected by the user. The proposed solution is a spatio-temporal clustering method, based on speed, to work with single trajectories. We compare the two different approaches with experiments on real data and show that the computation of stops using the concept of speed can be interesting for several applications.

Categories and Subject Descriptors

H.2.8 [**Database Applications**]: [Spatial Databases and GIS]

Keywords

Spatio-temporal clustering, moving objects, data mining, semantic trajectory annotation, trajectories

1. INTRODUCTION

In the last few years there has been an explosion of moving object data produced by mobile devices, emerging the necessity of efficient analysis of these data in different application domains. The trajectories left behind moving objects have been considered as the path followed by a moving object in space and time. Each point in a trajectory represents a position in space in a certain instant of time.

Existing approaches for trajectory data mining and knowledge discovery have focused on the geometrical properties of trajectories, without considering the background geographic information. For many application domains, however, useful information may only be extracted from trajectory data if their semantics and the background geographic information is considered [2]. Several works for trajectory data analysis have been developed specifically considering the road network as the background geographic information [6],[10], [12], [18].

Recently [17] has introduced a new model for reasoning over trajectories, which allows powerful semantic analysis, called *stops* and *moves*. A stop is a semantically important part of a trajectory that is relevant for an application, and where the object has stayed for a minimal amount of time. For instance, in a tourism application, a stop could be a touristic place, a hotel, an airport, etc. In a traffic management application, important places can be traffic lights, roundabouts, parking places, etc. According to the application, the minimal stop duration can vary significantly.

In [2], an algorithm named SMoT (Stops and Moves of Trajectories) has been provided to extract stops and moves from trajectory sample points, and an evaluation is presented to show how simple trajectory data analysis becomes by using this semantic model. In [1] moving patterns are extracted from stops and moves, and such patterns are modeled in the geographic conceptual schema in order to visualize trajectory patterns in the geographic space.

In all approaches that follow the model of stops and moves the user has to specify the places of interest, since stops and moves are defined from an application point of view. The main drawback of this assumption is that important places that may lead to the discovery of interesting patterns can be missed if they are not known by the user.

Figure 1(1) shows an example of a trajectory where the user has specified the places of interested A, B, and C. Observing Figure 1(1) we may see two dense parts in the trajectory that seem to be a stop: before A, and between B and C; but which were not specified as important places by the user.

In this paper we present an alternative to the algorithm SMoT, presented in [2] to compute stops and moves. While SMoT searches for intersections among trajectories and the relevant geographic objects, we propose a speed-based spatio-temporal clustering approach to find important places of

SAC'08 March 16-20, 2008, Fortaleza, Ceará, Brazil

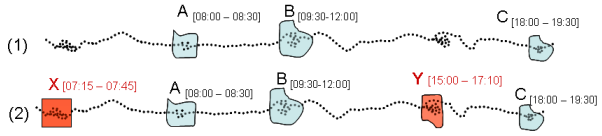

Figure 1: single raw trajectory and single semantic trajectory

trajectories. An example of our approach is shown in Figure 1(2). Considering that a minimal stop duration is 30 minutes, besides A, B, and C, our method would also find X and Y, that are not captured by SMoT. Our approach is generic enough to be applied in many different scenarios, not being exclusive to one single application.

The remainder of this paper is structured in the following way: in the next section we introduce the concept of stops and moves. In Section 3 we present the new clustering method. In Section 4 we show experiments with real data to compare SMoT and our approach. In section 5 we present the related works and in Section 6 we conclude the paper and suggest some directions of future research.

2. STOPS AND MOVES

In this section we present the definitions of trajectory, stops, and moves, based on the definitions provided in [2], that will be used in the remaining of this paper.

DEFINITION 1. *Trajectory Sample: A trajectory sample is a list of space-time points* $\{p_0 = (x_0, y_0, t_0), p_1 = (x_1, y_1, t_1), \ldots, p_N = (x_N, y_N, t_N)\}$, *where* $x_i, y_i \in \Re$, $t_i \in \Re^+$ *for* $i = 0, 1, \ldots, N$, *and* $t_0 < t_1 < t_2 < \cdots < t_N$.

Stops represent the important places of a trajectory where the moving object has stayed for a minimal amount of time. The important places are defined according to an application, and correspond to different spatial feature types [14] defined in a geographic database. For each relevant spatial feature type a minimal amount of time is defined, such that a trajectory should continuously intersect this feature in order to be considered a stop. We call this pair as a candidate stop.

DEFINITION 2. *Candidate Stop: A candidate stop* C *is a tuple* (R_c, Δ_c), *where* R_c *is a topologically closed polygon in* $\Re^2$ *and* Δ_c *is a strictly positive real number. The set* R_c *is called geometry of the candidate stop and* Δ_c *is called its minimum duration. The candidate stops are dependent of specific applications. An application is a finite set* $\{C_1 = (R_{c1}, \Delta_{c1}), C_2 = (R_{c2}, \Delta_{c2}), \ldots, C_N = (R_{cN}, \Delta_{cN})\}$ *of candidate stops with non-overlapping geometries* R_{c1}, R_{c2}, $\ldots, R_{cN}$.

DEFINITION 3. *Stop: A stop of a trajectory* T *with respect to an application* $\mathcal{A}$ *is a tuple* (R_k, t_j, t_{j+n}) *such that a maximal subtrajectory of* T $\{(x_i, y_i, t_i) \mid (x_i, y_i)$ *intersects* $R_k\} = \{(x_j, y_j, t_j), (x_{j+1}, y_{j+1}, t_{j+1}), \ldots, (x_{j+n}, y_{j+n}, t_{j+n})\}$, *where* R_k *is the geometry of* C_k *and* $|t_{j+n} - t_j| \geq \Delta_k$.

DEFINITION 4. *Move: A move of a trajectory* T *with respect to an application* $\mathcal{A}$ *is: (i) a maximal contiguous subtrajectory of* T *in between two temporally consecutive stops of* T*; OR (ii) a maximal contiguous subtrajectory of* T *in between the starting point of* T *and the first stop of* T*; OR (iii) a maximal contiguous subtrajectory of* T *in between the last stop of* T *and the last point of* T*; OR (iv) the trajectory* T *itself, if* T *has no stops.*

In other words, every point of a trajectory that is not in a stop is in a move. A move has no minimal time duration, and may either intersect or not a candidate stop. If it does, the intersection time interval must be less than the candidate stop minimal duration.

It is important to note that the minimal stop duration should be specified taking into account the average periodicity of the trajectory time points, in order to warrant that there will be sufficient points to characterize a stop.

The algorithm SMoT, presented in [2] to extract stops and moves, searches for intersections between the trajectory sample points and the candidate stops. In the following section we present a clustering-based algorithm to identify stops and moves of trajectories, called CB-SMoT (Clustering-Based SMoT).

3. THE METHOD CB-SMOT

The intuition of our method is that the parts of a trajectory in which the speed is lower than in other parts of the same trajectory, correspond to interesting places. In a tourism application, for instance, let us suppose that a tourist is visiting a new city. His trajectory would be something like: visit an important monument, visit a museum, go to his hotel, go to a night-club, and return to the hotel. Probably his trajectory has a lower speed around these places than it has in other parts of the trajectory where he was moving from one place to another. In a traffic management application, for instance, the speed of car trajectories will be lower in traffic jams, traffic lights, roundabouts, and electronic velocity controllers.

Following this reasoning, we propose a clustering-based algorithm to find low speed regions.

3.1 Definitions

DBSCAN [7] is a well known density-based clustering algorithm. In DBSCAN, the neighborhood is defined as an area around points (in 2-dimensional space). Because we are specially interested in finding clusters in a single trajectory and also consider time, we have changed some concepts of DBSCAN. The first one is the notion of neighborhood, that should contain only points in the considered trajectory. Besides, it should consider the distance over the trajectory and not the direct distance between two points. Therefore, we define the Eps-linear-neighborhood of a point p_k as the set of points before and after p_k in the trajectory whose distance from p_k is less or equal to *Eps*.

DEFINITION 5. *Eps-linear-neighborhood of a point: Let* $\{p_0, p_1, ..., p_k, p_{k+1}, \ldots, p_N\}$ *be a trajectory, where* $p = (x, y, t)$. *The Eps linear neighborhood of a point* p_k, *denoted by* $LNEps(p_k)$, *is the maximal set of points* p_i, *such that:*

$$(\sum_{i=m}^{k-1} dist(p_i, p_{i+1})) \leq Eps \cup (\sum_{i=k+1}^{n} dist(p_{i-1}, p_i)) \leq Eps$$

where $t_0 \leq t_m < t_k < t_n \leq t_N$

Eps is a positive number that represents the maximum distance between a point p and its neighbors on the tra-

jectory. Instead of considering a minimal number of points for a region to be dense, we will use the notion of minimal time. In the following definitions we keep the same name of the definitions presented in [7], but considering the notion of time.

DEFINITION 6. *Core point: A point $p = (x_p, y_p, t_p)$ of a trajectory is called core point with respect to Eps and $MinTime$ if $|t_n - t_m| \geq MinTime$, where n is the last point of LNEps(p), and m is the first one (the neighborhood is ordered by time).*

Definition 6 corresponds to the maximum speed condition. The ratio $Eps/MinTime$ gives the maximum average speed (speed limit) of the respective neighborhood. By increasing the $MinTime$ parameter the related speed decreases. Besides, using time instead of number of points will also avoid problems like the absence of points because of some equipment failure and the time differences in the sample rate.

DEFINITION 7. *Directly density-reachable: A point q is directly density-reachable to a point p if $q \in LNEps(p)$ and p is a core point with respect to Eps and $MinTime$.*

DEFINITION 8. *Density-reachable: A point q_0 is density-reachable from a point p with respect to Eps and $MinTime$ if there exists a chain $q_0, q_1, q_2, \ldots, q_N$ where $q_N = p$ and q_k is directly density-reachable to q_{k+1}.*

DEFINITION 9. *Density-connected: Two points p and q are density-connected with respect to Eps and $MinTime$ if there exists a point o and both p and q are density-reachable from o.*

A non-core point can be density-connected to another non-core point if both have a common core point. Having these definitions, a trajectory cluster can be defined as follows.

DEFINITION 10. *Trajectory cluster: A cluster G of a trajectory T with respect to Eps and $MinTime$ is a non-empty subtrajectory of T formed by a set of contiguous time-space points such that:*

1. *$\forall p, q \in T$: if $p \in G$ and q is density-reachable from p with respect to Eps and $MinTime$, then $q \in G$.*
2. *$\forall p, q \in G$: p is density-connected to q with respect to Eps and $MinTime$.*

There are three main modifications in the DBSCAN algorithm for clustering single trajectories: (i) instead of searching for a minimal amount of points inside the neighborhood, we search for a minimal duration; (ii) we replace the original function that returns the neighborhood of a point by an algorithm that follows Definition 5; (iii) we use the quantile function to calculate the *Eps*, as will be explained in the following section.

3.2 The Eps Parameter

The *Eps* parameter indicates the absolute distance used to calculate the neighborhood of a point. However, it is difficult to the user to specify a good value for this parameter without knowing well the characteristics of each trajectory, since it is an absolute value. Considering this, we elaborate an alternative to the user in order to adjust this parameter.

A trajectory T can be viewed as a list of distances d_i between two consecutive points p_i and p_{i+1}. These distances have an arithmetic mean μ and a standard deviation σ. With these two parameters it is possible to plot the appropriate Gaussian curve. This curve represents some properties about the trajectory and it is useful as a kind of scaling tool. Therefore, we can exploit this feature in order to avoid the knowledge about the trajectory domain, using the quantile function. It is the inverse of the cumulative distribution function, where **quantile function** : $[0, 1] \rightarrow \Re$. The quantile function is defined as:

$$F^{-1}(p, \mu, \sigma) = \mu + \sigma\sqrt{2}erf^{-1}(2p - 1)$$

and

$$erf^{-1}(x) = \sum_{k=0}^{\infty} \frac{c_k}{2k+1}(\frac{\sqrt{\pi}}{2}x)^{2k+1}$$

where, μ is the mean, σ is the standard deviation, $c_0 = 1$ and c_k is:

$$c_k = \sum_{m=0}^{k-1} \frac{c_m c_{k-1-m}}{(m+1)(2m+1)}$$

The objective is to have a relative parameter, related to the mean and the standard deviation, instead of the user defined absolute *Eps* value. The user needs to known approximately the proportion of points that generate potential stops in relation to the total amount of points in the trajectory. So, as an input, we use a parameter called *area*, with value between 0 and 1, and from this value, the *Eps* parameter is computed.

3.3 The CB-SMoT algorithm

We propose a two-step algorithm to extract stops and moves, called CB-SMoT, which is shown in pseudo-code in Listing 1. In the first step the slower parts of a trajectory, that we call *potential stops*, are identified using the variation of the DBSCAN algorithm that considers one-dimensional line (trajectories) and speed, explained in Section 3.1. In the second step, the algorithm identifies where these potential stops (clusters) found in the first step are located, considering the geography behind the trajectories.

CB-SMoT takes each potential stop (clusters) and tests both intersection and minimal stop duration with the candidate stops. In case that a potential stop does not intersect any of the candidate stops, it still can be an interesting place. Then, in order to provide this information to the user, the algorithm labels such places as *unknown stops*.

DEFINITION 11. *Unknown stop: An unknown stop of a trajectory T with respect to an application $\mathcal{A}$, Eps, and $MinTime$, is a cluster G_k of T which does not intersect any R_j of $\mathcal{A}$ for at least Δ_j, where $C_j = (R_j, \Delta_j)$ is a candidate stop.*

Figure 2 illustrates this concept. The trajectory $T = \{p_0, p_1, \ldots, p_n\}$ represented in Figure 2 has 4 potential stops, the clusters G_1, G_2, G_3 and G_4. In this example the user has specified 4 candidate stops, identified by the ellipsis R_{C1}, R_{C2}, R_{C3} and R_{C4}. The cluster G_1 intersects the candidate stop R_{C1} for a time greater than Δ_{c1}, then the first stop of the trajectory is R_{C1}. The same occurs with the cluster G_2, considering R_{C3}, which is the second stop of

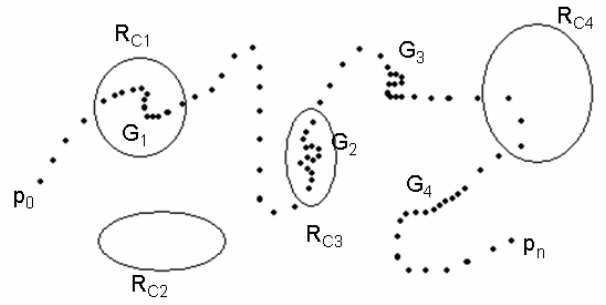

Figure 2: Example of a trajectory with 2 stops and 2 unknown stops

the trajectory. The clusters, G_3 and G_4 do not intersect any candidate stop. Therefore, G_3 and G_4 are unknown stops.

Every unknown stop receives an identifier. In case two or more unknown stops intersect each other, they will receive the same identification. Figure 3 shows an example in which trajectory T_1 has a cluster G_1 that does not intersect any candidate stop. Similarly, trajectory T_2 has a cluster G_2 that does not intersect any candidate stop. As G_1 intersects G_2, both are located in the same region, and therefore they will receive the same identification.

Listing 1: CB-SMoT algorithm

```
INPUT: T //set of trajectories
       A //application
       a //area for the quantile function
       minTime //minimun time  for clustering

OUTPUT: S //set of stops
        M //set of moves

METHOD:
FOR each trajectory t in T DO
// compute the clusters
  set clusters as empty
  Eps = quantile(μ(t),σ(t),a)
  FOR each unprocessed point p in t DO
    neighbors = linear_neighborhood(p,Eps)
    IF p is a core point wrt minTime,Eps
      FOR each neighbor n in neighbors DO
        add to neighbors every unprocessed point
              in linear_neighborhood(n,Eps)
        set neighbors as a cluster in clusters
        set all points in neighbors as processed
      ENDFOR
  ENDFOR
// find stops and moves
  FOR each cluster in clusters DO
    FOR each intersection with a different R_c
            with duration time t ≥ Δ_C DO
       generate a stop in S
    ENDFOR
    FOR each subtrajectory that is not stop DO
       IF duration ≥ minTime
          generate an unknown stop in S
    ENDFOR
  ENDFOR
  FOR each subtrajectory which is not a stop DO
    generate a move in M
  ENDFOR
ENDFOR
```

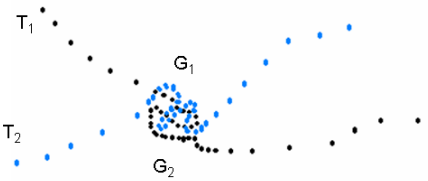

Figure 3: Two trajectories with the same unknown stop

Table 1: Trajectories X Buildings (*minTime*=120s)

Algorithm	Stops	Unknown Stops	Time (s)
SMoT	6	-	189
CB-SMoT (area = 0.3)	1	69	160
CB-SMoT (area = 0.35)	1	105	196
CB-SMoT (area = 0.4)	1	182	274

4. EXPERIMENTS AND EVALUATION

In order to accomplish experiments with both trajectory and geographic data in real applications, we have extended the tool Weka-GDPM [5] to support both geographic data and trajectories. Weka-GDPM is an extension of Weka [8], which is a data mining toolkit that implements several algorithms for association rule mining, clustering, and classification. We have implemented both SMoT and CB-SMoT algorithms in this tool, and their output (stops and moves) is stored in the geographic database. Therefore, any of the classical data mining methods available in the tool can be directly applied over stops and moves.

We have used trajectory data collected in the city of Amsterdam to perform some preliminary experiments. These trajectories correspond to an educational game [16], and contain around 125.000 points of 487 different trajectories of students.

The first experiment was performed considering a set of buildings as the candidate stops, with parameter $MinTime$ as 120 seconds. The clustering algorithm was applied three times, with area as 0.3, 0.35, and 0.4. The algorithm SMoT found 6 stops, as shown in Table 1. For all different values of the area parameter, almost all clusters found by CB-SMoT were unknown stops. This large amount of unknown stops is understandable by visualizing the data shown in Figure 4, where the candidate stops are only a few buildings (small polygons in black). The buildings dataset has around 42 thousand buildings, but in the region where the trajectories were collected there are only a few. This is one of the reasons why SMoT found only a few stops and CB-SMoT found several unknown stops.

Another observation in this experiment is that CB-SMoT has found only 1 known stop, while SMoT found 6. This occurs because CB-SMoT is based on speed, and the velocity of the trajectories in the stops found by SMoT was not lower enough to be considered as a stop (cluster) by CB-SMoT.

A second experiment, shown in Table 2, was performed with geographic data corresponding to areas that cover most places where the trajectories are located. Considering minimal time as 300s, the SMoT algorithm found 357 stops, while CB-SMoT found much less stops for the different values of the area parameter. In this experiment we have the

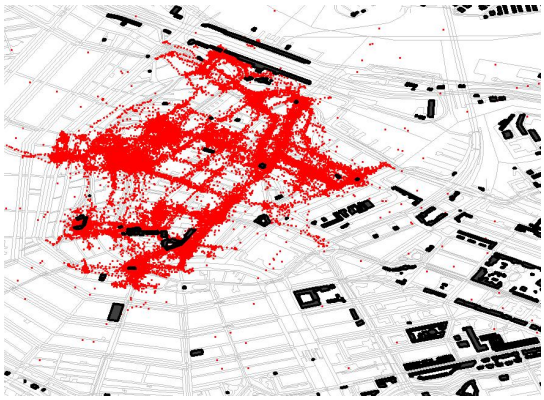

Figure 4: Trajectory samples and buildings

Table 2: Trajectories X Area and *minTime*=300s

Algorithm	Stops	Unknown Stops	Time (s)
SMoT	357	-	485
CB-SMoT (area = 0.3)	37	0	259
CB-SMoT (area = 0.35)	51	0	388
CB-SMoT (area = 0.4)	69	0	476
CB-SMoT (area = 0.45)	143	0	537

opposite scenario in relation to the previous one. The candidate stops are dense and cover the complete region, as can be visualized in Figure 5 (polygons). Because all trajectory points are located in one of the polygons that cover the complete area, no unknown stops were identified by CB-SMoT.

In a scenario like experiment 2, where the complete area is covered by candidate stops (dense), SMoT finds much more stops because each polygon of the background area that the trajectory has crossed for the minimal amount of time will be considered as a stop. On the contrary, CB-SMoT will find stops only in the polygons in which the speed is low enough for the minimal stop duration.

In many applications as, for instance, traffic management and tourism, the algorithm CB-SMoT can find several interesting places that are previously unknown by the user. In a traffic management application, CB-SMoT would find roundabouts, traffic lights, and velocity controllers even if they are not given as candidate stops by the user. In a tourism application, CB-SMoT could find several unexpected locations (touristic or not) that attract tourists.

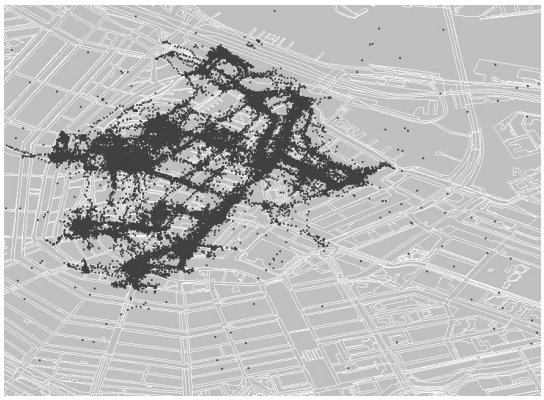

Figure 5: Trajectory samples and areas

Although SMoT may only discover stops related to interesting places defined by the user, it may be better than CB-SMoT in applications where the speed is not relevant.

An important remark about CB-SMoT is that it is able to generate clusters in parts of the trajectory where some points are missing. This is very common inside buildings, where the GPS signal may be lost. Usually, in this case, the trajectory sample points are discontinued at the moment the moving object enters in a building, and restart when the object leaves the building. Pure density-based clustering algorithms are not able to directly find clusters in such regions. Our approach, on the contrary, will discover such places because the average speed between the two points (entrance and exit of a building) is very low.

5. RELATED WORKS

There are only a few works in the literature that consider semantic properties of trajectories. In [2], an intersection-based approach is presented to integrate trajectory sample points with geographic information, with the algorithm SMoT.

Ashbrook and Starner [3] use the well-know clustering algorithm K-means to find important places in trajectories. The problem of K-means is that the number of clusters must be given a priori, what in our problem is unknown. Zhou [19] uses information obtained from GPS to classify personal gazetteers. The gazetteers correspond to the most important places of a person, such as home, work, supermarket, etc. A set of trajectories is processed by the DJ-Cluster algorithm in order to find the baseline places. DJ-Cluster is a density-based algorithm similar to DBSCAN that works on the notion of connectivity between neighborhoods. However, DJ-Cluster does not consider the temporal dimension.

GDBSCAN [15] is another extension of DBSCAN, developed for clustering non-spatial attributes. ST-DBSCAN [4] was developed to consider both spatial and temporal aspects, but it treats them separately in the evaluation function. In our case we use the spatio-temporal data together, in order to consider the speed.

Several works for trajectory clustering have been developed to find similar subtrajectories or dense regions as [9], [11], [13]. While these works look for clusters in a set of trajectories, we look for clusters in a single trajectory, in order to discover potential stops.

6. CONCLUSION AND FUTURE WORKS

The increasing amount of location based data, specifically trajectory data, emerged the necessity of intelligent analysis and knowledge discovery from these data. The analysis of trajectory sample points is very limited for several applications. Recently, a new approach based on the notion of interesting places of trajectories (stops) has emerged as a promising way for more meaningful analysis on trajectories.

In this paper we have introduced a new approach to discover interesting places in trajectories. It is a speed-based method to find clusters in single trajectories. In general, the main contributions of this paper include:

- a new spatio-temporal clustering algorithm, which is a variation of DBSCAN, where the distance between points is calculated along over the trajectory, instead of the traditional euclidean distance. We consider the

notion of minimal time instead of minimal number of points for a region to be considered dense. Additionally, we introduced the quantile function as a way to automatically estimate the value of the Eps parameter;

- the discovery of interesting places which are not known a priory by the user (unknown stops);
- the generation of stops only in regions where the speed of the trajectory is low;
- the discovery of clusters in regions where sample points are missing, as for instance, when the moving object enters in a building.

Additionally to the new method presented in this paper for extracting interesting places from trajectories, we implemented both methods SMoT and CB-SMoT in Weka, with a friendly GUI for trajectory data analysis and knowledge discovery.

In future works, we will evaluate the discovered unknown stops with the user of the application. We will perform experiments with other kind of background geographic information of the city of Amsterdam. Indeed, we will apply the proposed method over car trajectories, in the context of a traffic management application.

7. ACKNOWLEDGMENTS

This research has been funded by the Brazilian agencies CAPES and CNPq, the European Union (FP6-IST-FET programme, Project n. FP6-14915, GeoPKDD: Geographic Privacy-Aware Knowledge Discovery and Delivery), and the Research Foundation Flanders (FWO-Vlaanderen), Research Project G.0344.05. We would like to thank Bart Moelans for his comments, and to the Waag-Society for the trajectory data.

8. REFERENCES

[1] L. O. Alvares, V. Bogorny, J. F. de Macedo, B. Moelans, and S. Spaccapietra. Dynamic modeling of trajectory patterns using data mining and reverse engineering. In *Twenty-Sixth International Conference on Conceptual Modeling - ER2007 - Tutorials, Posters, Panels and Industrial Contributions*, volume 83, pages 149–154. CRPIT, November 2007.

[2] L. O. Alvares, V. Bogorny, B. Kuijpers, J. A. F. de Macedo, B. Moelans, and A. Vaisman. A model for enriching trajectories with semantic geographical information. In *GIS'07: Proceedings of the 15th Annual ACM International Symposium on Advances in Geographic Information Systems*, New York, NY, USA, 2007. ACM Press.

[3] D. Ashbrook and T. Starner. Using gps to learn significant locations and predict movement across multiple users. *Personal and Ubiquitous Computing*, 7(5):275–286, 2003.

[4] D. Birant and A. Kut. St-dbscan: An algorithm for clustering spatial-temporal data. *Data and Knowledge Engineering*, 60(1):208–221, 2007.

[5] V. Bogorny, A. T. Palma, P. Engel, and L. O. Alvares. Weka-gdpm: Integrating classical data mining toolkit to geographic information systems. In *WAAMD*, pages 9–16. SBC, 2006.

[6] S. Brakatsoulas, D. Pfoser, and N. Tryfona. Modeling, storing, and mining moving object databases. In *IDEAS '04: Proceedings of the International Database Engineering and Applications Symposium (IDEAS'04)*, pages 68–77, Washington, DC, USA, 2004. IEEE Computer Society.

[7] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. In E. Simoudis, J. Han, and U. M. Fayyad, editors, *Second International Conference on Knowledge Discovery and Data Mining*, pages 226–231. AAAI Press, 1996.

[8] E. Frank, M. A. Hall, G. Holmes, R. Kirkby, B. Pfahringer, I. H. Witten, and L. Trigg. Weka - a machine learning workbench for data mining. In O. Maimon and L. Rokach, editors, *The Data Mining and Knowledge Discovery Handbook*, pages 1305–1314. Springer, 2005.

[9] F. Giannotti, M. Nanni, F. Pinelli, and D. Pedreschi. Trajectory pattern mining. In P. Berkhin, R. Caruana, and X. Wu, editors, *KDD*, pages 330–339. ACM, 2007.

[10] R. H. Güting, V. T. de Almeida, and Z. Ding. Modeling and querying moving objects in networks. *VLDB J.*, 15(2):165–190, 2006.

[11] J.-G. Lee, J. Han, and K.-Y. Whang. Trajectory clustering: a partition-and-group framework. In *SIGMOD '07: Proceedings of the 2007 ACM SIGMOD International Conference on Management of data*, pages 593–604, New York, USA, 2007. ACM Press.

[12] X. Li, C. Claramunt, C. Ray, and H. Lin. A semantic-based approach to the representation of network-constrained trajectory data. In *Progress in Spatial Data Handling - 12th International Symposium on Spatial Data Handling*, pages 451–464, Berlin, 2006. Springer.

[13] M. Nanni and D. Pedreschi. Time-focused clustering of trajectories of moving objects. *Journal of Intelligent Information Systems*, 27(3):267–289, 2006.

[14] OGC. Topic 5, opengis abstract specification - features (version 4) (1999). Available at: http://www.OpenGIS.org/techno/specs.htm. Accessed in August (2005), 1999.

[15] J. Sander, M. Ester, H.-P. Kriegel, and X. Xu. Density-based clustering in spatial databases: The algorithm gdbscan and its applications. *Data Mining and Knowledge Discovery*, 2(2):169–194, 1998.

[16] W. Society. Frequency 1550. Available at: http://www.waag.org/project/frequentie. Accessed in September (2007), 2005.

[17] S. Spaccapietra, C. Parent, M.-L. Damiani, J. A. F. de Macedo, F. Porto, and C. Vangenot. A conceptual view on trajectories. Technical report, Ecole Polytechnique Federal de Lausanne, April 2007.

[18] M. Vazirgiannis and O. Wolfson. A spatiotemporal model and language for moving objects on road networks. In *SSTD*, pages 20–35, 2001.

[19] C. Zhou, N. Bhatnagar, S. Shekhar, and L. Terveen. Mining personally important places from gps tracks: a hybrid approach. In *Proceedings of the 23rd International Conference on Data Engineering Workshops, ICDE 2007, Istambul,Turkey*, pages 517–526. IEEE Computer Society, April 2007.