Cent. Eur. J. Comp. Sci. • 2(3) • 2012 • 183-193
DOI: 10.2478/s13537-012-0018-4

VERSITA

Central European Journal of **Computer Science**

Mining moving object data

Review Article

Jaroslav Zendulka[1,2*], Martin Pešek[1†]

1 *Faculty of Information Technology, Brno University of Technology*
Božetěchova 1/2, 612 66 Brno, Czech Republic

2 *IT4Innovations Centre of Excellence*
Božetěchova 1/2, 612 66 Brno, Czech Republic

Received 31 January 2012; accepted 17 August 2012

Abstract: Currently many devices provide information about moving objects and location-based services that accumulate a huge volume of moving object data, including trajectories. This paper deals with two useful analysis tasks – mining moving object patterns and trajectory outlier detection. We also present our experience with the TOP-EYE trajectory outlier detection algorithm, which we applied to two real-world data sets.

Keywords: data mining • moving object data • trajectory • moving object patterns mining • trajectory outlier detection

1. Introduction

With recent advances in positioning, telemetry and telecommunication technologies, and with wide availability of devices that produce information about the position of an object in some time, enormous amounts of data about moving objects are being collected and employed by many applications. Examples of such devices include mobile phones and devices with embedded GPS or sensor networks. In general, the moving object data are spatio-temporal data, a complex data type that gained attention of researchers after data mining techniques for relational data had proved their usefulness. Because trajectories are very important characteristics of the behavior of moving objects and large amounts of trajectories are currently collected and stored in databases, trajectory mining has become a major challenges in data mining.

Several useful trajectory data analysis tasks have been introduced and algorithms to solve them have been developed. This paper deals with two of them, namely moving object patterns mining and trajectory outlier detection. The objective of the former is to find moving clusters with specific properties; the objective of the latter is to detect suspicious or anomalous moving objects. We also present our experience with the TOP-EYE algorithm, which is a trajectory outlier detection algorithms. A more complete overview on mining moving object data was presented by Han at DASFAA 2010 [4, 5].

* *E-mail: zendulka@fit.vutbr.cz (Corresponding author)*
† *E-mail: ipesek@fit.vutbr.cz*

2. Mining moving object patterns

Moving object pattern analysis is a data mining task whose objective is to discover potentially useful patterns in moving object data. Such patterns can be beneficial for economic and social studies related to social and economic behaviors of people, or to climate and ecological studies related to movements of animals and changes of natural phenomena. The moving object patterns can be categorized as follows [9]:

- *Repetitive pattern*. This pattern concerns periodic behaviors. Sme animals have repetitive movement patterns. But it might be difficult for biologists to discover the patterns even though a lot of animal movement data is currently available.
- *Relationship pattern*. This type of pattern is focused on relationships among moving individuals. The fundamental task of this type is to find groups of objects that move together. But in some cases there are also some other relationships among objects in the group.
- *Frequent trajectory pattern*. The objective of this task is to find general moving trends of all objects in a data set in terms of space and time (where, when, with what speed etc.).

Here, we will discuss in more detail on relationship patterns. If we want to find groups of objects that move together, we want to find clusters of objects in space and time with specific behavior, for example that the clusters contain the same objects. The fundamental concept here is a *moving object cluster*. It can be defined in both spatial and temporal dimensions [8]:

1. a group of moving objects should be geometrically close to each other,
2. they should be together for at least some minimum time duration.

These two properties can be specified by thresholds: max_r – the radius of the cluster and/or min_o – the minimum number of moving objects in the cluster; and min_t – the minimum number of consecutive timestamp snapshots in which the group of moving objects is in the same cluster. Then a moving cluster can be defined as *Moving_cluster*(min_o, min_t). There are several specific moving object patterns referred to as relative motion patterns (some of these patterns are illustrated in Figure 1) [5]:

- *Flock*(min_o, max_r) – at least min_o objects are within a circular region of radius max_r and they move in the same direction. It can be extended to include also a temporal constraint to *Flock*(min_o, max_r, min_t).
- *Leadership*(min_o, max_r, min_t) – at least min_o objects are within a circular region of radius max_r, they move in the same direction and at least one of the objects was already heading in this direction for at least min_t time steps.
- *Convergence*(min_o, max_r) – at least min_o objects will pass through the same circular region of radius max_r (assuming they keep their direction).
- *Encounter*(min_o, max_r) – at least min_o objects will be simultaneously inside the same circular region of radius max_r (assuming they do not change their speed and direction).

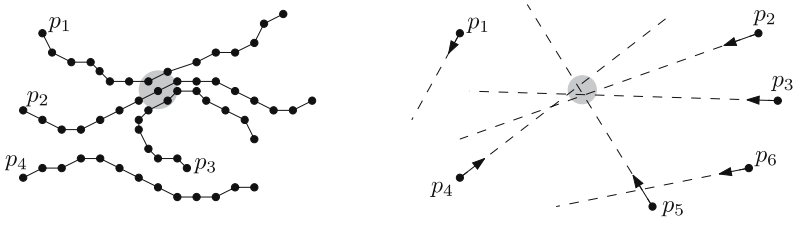

Figure 1. An example of a flock pattern (left) and a convergence pattern (right). Adapted from [3].

One disadvantage of the flock pattern is its rigid constraint in a form of the circle radius. Such a constraint can result in the loss of some objects that move together with the cluster and are close to it but are outside the circle defined by the radius. Avoiding this problem was the motivation for the *convoy* pattern, which uses density-based clustering at each timestamp.

Another rigid constraint for both flock and convoy patterns, in the temporal dimension, can result in the loss of interesting objects. It is illustrated in Figure 2. For example, if $min_t = 3$, no moving cluster will be found. But we would say that these four objects move together even though some objects temporarily move out of the cluster for several snapshots. To avoid such loss, the concept of *swarm* is introduced in [8]. A swarm is a group of moving objects containing at least min_o objects which are in the same cluster for at least min_t timestamp snapshots, not necessarily consecutive. For example, if $min_o = 2$ and $min_t = 3$, we can find swarm $(\{o_1, o_3, o_4\}, \{t_1, t_3, t_4\})$ in Figure 2.

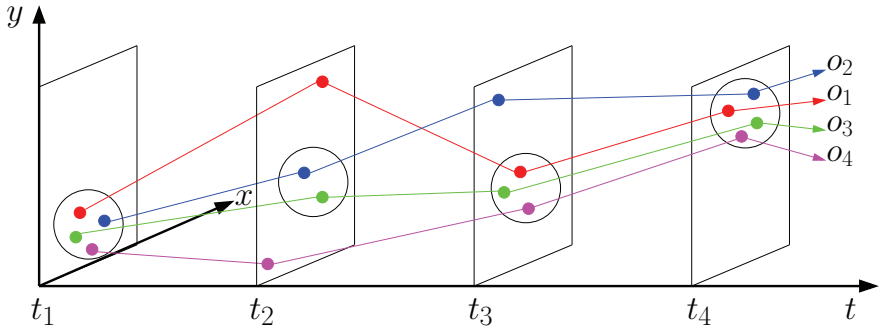

Figure 2. The loss of interesting moving objects clusters. Adapted from [8].

An example of a tool that integrates several moving object data mining functions including periodic and swarm pattern mining is MoveMine [9].

3. Moving object trajectory outlier detection

Automatic detection of suspicious movement of objects is usually focused on detection of outliers in moving objects trajectories. Here, the outlier is a trajectory which differs substantially from or is inconsistent with the remaining set of trajectories.

The objective of trajectory outlier detection as a descriptive task is to find outliers in a trajectory database. The goal of the trajectory outlier detection as a predictive task is to decide if a trajectory of a moving object is outlier or not. In the process of detecting trajectory outliers, either whole trajectories or only their parts (partial trajectory outliers) can be considered. Both unsupervised and supervised learning can be employed to solve these tasks.

The key task of outlier detection is to measure the abnormality of a trajectory. In case of supervised learning, two further questions arise – how to encode a trajectory and what classifier to use. Moreover, if the outliers should be detected in real-time, it may be useful or necessary to combine various aspects of abnormality of moving objects into an unified evolving abnormality score which has the ability to simultaneously capture the evolving nature of many abnormal moving trajectories and/or detect the outlier (maybe potential) as soon as possible.

In this section, we will briefly present two approaches to measure abnormality – distance-based and motif-based. After that, we will present our results with the TOP-EYE algorithm, which is able to detect top-k evolving trajectory outliers.

3.1. Distance-based approach

Distance-based measures are well-known in clustering where they are used to quantify the similarity of objects. Therefore, they can also be used to measure the abnormality of outliers. In fact, outliers can be discovered as a side-effect of clustering. They are objects lying outside clusters. But the main objective of clustering is to find clusters in a data set, not to detect outliers. Knorr et al. [6] introduce the concept of a distance-based outlier (*DB outlier*) and present several

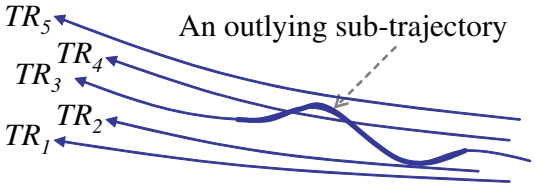

Figure 3. An example of an outlying sub-trajectory. Adapted from [7].

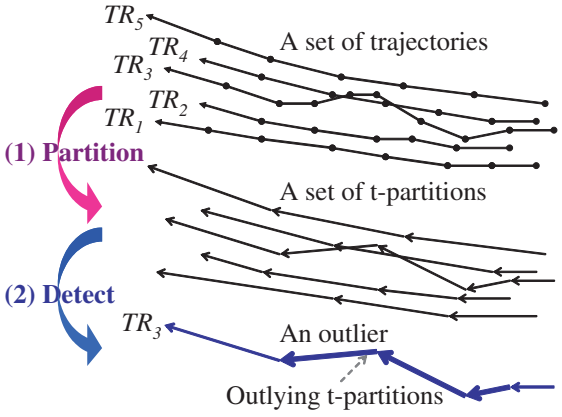

Figure 4. The concept of a partition-and-detect framework from [7].

algorithms for mining them in a k-dimensional data set. They define a $DB(p, D)$ outlier in a data set T as an object O of T such that at least the fraction p of the objects in T lies greater than distance D from O. One of the real-life case studies they present in their paper is a trajectory outlier detection in a dataset extracted from surveillance videos. They consider whole trajectories. The trajectory is usually recorded and considered as a sequence of timestamped 2D points in such applications. This representation, however, may be too detailed for distance computation. Therefore, they use only several summary characteristics of the trajectory, namely start and end points, heading (the average, minimum and maximum values of the directional vector of the tangent of the trajectory at each point), velocity (average, minimum and maximum velocity of the person during the trajectory). The distance of two trajectories represented as a point in this 4-dimensional space was computed as a weighted sum of differences along the dimensions. The weights were determined by domain experts.

The disadvantage of techniques based on comparing trajectories as a whole is that they are usually not able to detect outlying portions of the trajectories as it is illustrated on Figure 3. To solve this problem, in [7], the authors introduce a partition-and-detect framework and present an outlier detection algorithm TRAOD based on it. The idea is to partition a trajectory into a set of trajectory partitions referred to as *t-partitions*, and then, to detect outlying t-partitions (see Figure 4).

The t-partition of a trajectory A is a line segment $L_{ij} = p_i p_j$ $(i > j)$, where p_i and p_j are two points of A. It allows a coarse-grained partitioning of a trajectory because points p_i and p_j are not necessarily two consecutive registered points of the trajectory. The authors present a two-level trajectory partitioning strategy to speed up outlier detection. First, a coarse granularity partitioning is applied. It is based on a principle referred to as the minimum description length (MDL) principle, which is a concept coming from information theory. A set of coarse t-partitions corresponds to hypothesis, and a trajectory corresponds to data here. Finding the best hypothesis using the MDL principle leads to a partitioning which is a good trade-off between preciseness and conciseness. The course granularity partitioning allows early pruning of many portions of trajectories. Only t-partitions that are likely to be outlying are partitioned into fine t-partitions and inspected.

A trajectory A is said to be close to a t-partition L_j if the total length of t-partitions of A which are in a distance less than a threshold D from L_j is greater or equal than the length of l_j. The definition of an outlying t-partition is similar to the definition of $DB(p, D)$ outlier mentioned above. A t-partition L_i is outlying if there is a fraction p of trajectories in the database which are not close to L_i. The parameter p is given by a user. A trajectory outlier is then defined as a trajectory whose fraction of outlying t-partitions is greater or equal that a user specified threshold.
The authors also extend their definition of closeness by incorporating the density of trajectories in order not to favor t-partitions in dense regions over those in sparse ones. Therefore, this technique can be classified as hybrid, since it is not purely distance-based. In addition, they introduce a distance function composed of three components: the perpendicular distance, the parallel distance and the angel distance. This makes it possible to consider two types of outliers: positional outlier and angular outlier. They differ in the weights of the components of the distance function applied in detection.
The experimental evaluation of TRAOD on two datasets containing hurricane track data and animal movement data proved promising results and good performance characteristics. We will mention some of these results in Subsection 3.3 where we present our results obtained applying an algorithm TOP-EYE.
Another interesting approach where distance measure is derived from the idea of Minimum Hausdoff Distance is published in [11]. This distance function considers the direction and velocity of objects. Moreover, R-Tree is used to reduce the costs of its computation.

3.2. Motif-based approach

The motif-based approach is represented by a ROAM (*R*ule- and M*o*tif-based *A*nomaly Detection in *M*oving Objects) [10]. Trajectories are expressed using discrete pattern fragments called *motifs* here. A rule-based classifier which accepts features derived from sequences of *motifs* with additional attributes is then used to classify trajectory outliers.
The concept of a motif is illustrated in Figure 5. The two trajectories in the figure have similar shapes except that the right one has an extra loop. The trajectories could be partitioned into such fragments that the similarity could be detected. Provided that there is a pre-defined set of representative fragments – motifs, the trajectories can be represented using the motifs. In our example both trajectories consist of the same motifs m_2 and m_4, the right one contains another motif m_1.

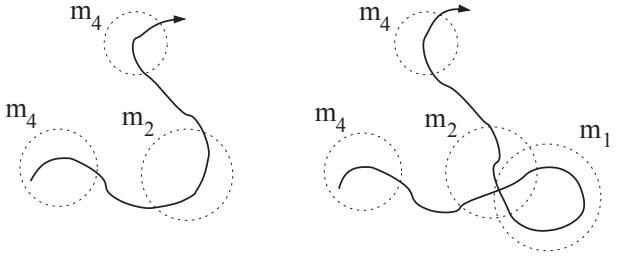

Figure 5. An example of two trajectories and motifs expressing them. Adapted from [10].

The framework ROAM includes three modules providing functionality for three basic steps of the trajectory anomaly detection process: motif extraction, mapping to features and classification.
First, motifs of the input trajectory data set must be extracted and trajectories mapped into a corresponding motif space. ROAM uses a sliding window to partition the trajectories in the data set. After that, clustering is used to find representative sets – motifs. Once the motifs of a given input data set are identified, the trajectories are compared with the set of motifs using a sliding window of the same size as for motifs identification. The Euclidean distance is used to measure similarity. Let w be a fragment of a trajectory A in the sliding window and m a motif. We say that the motif m is expressed in the trajectory A if $||w - m|| \leq \epsilon$, where ϵ is a parameter specified by a user.
In ROAM a trajectory is represented by a sequence of so-called motif expressions. Each motif expression has the form of $(m_i, t_{start}, t_{end}, l_{start}, l_{end})$, where m_i is the motif identification, t_{start} and t_{end} are starting and ending times, and l_{start} and l_{end} are starting and ending locations. The complete representation of the trajectory is referred here to as the *motif trajectory*. Moreover, for each motif expression, a set of other attributes that may be useful for classification can be specified.

Once the motif expressions have been extracted, motif trajectories must be mapped to a feature space that will be the input of a classifier. Because the set of different pairs of attribute – attribute_value, which is the base of the mapping, can be large, the authors propose feature generalization to reduce dimensionality.
Finally, the classifier is used to the anomaly detection. The authors proposed a rule-based classifier CHIP (*C*lassification using *HI*erarchical *P*rediction Rules).

3.3. Evolving trajectory outlier detection

The objective of the evolving trajectory outlier detection approach is to identify evolving outliers at very early stage with relatively low false positive rate. The concept of evolving trajectory outlier was introduced in [2] where the authors present a top-k evolving trajectory outlier detection method named TOP-EYE. The method continuously computes the outlying score of a tested trajectory with respect to other trajectories in the database. The outlying score can be defined based on moving direction or density of trajectories.

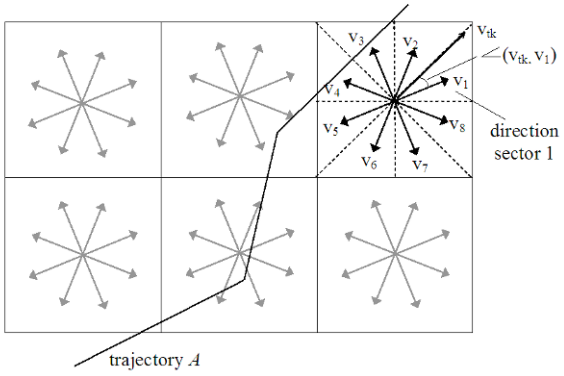

Figure 6. Direction distance measure in TOP-EYE method.

At first, we focus on direction-based outliers. To compute the score effectively, the continuous space of the monitored area is discretized into a regular grid. Moreover, a probabilistic model is used to represent directions of trajectories in the database in each cell of the discretized space grid. Each cell is partitioned into eight direction sectors, each with an angle of $\pi/4$ (see Figure 6). The goal is to represent the direction information in a cell c by a vector:

$$\mathbf{g} = (p_{1c},\ p_{2c},\ p_{3c},\ p_{4c},\ p_{5c},\ p_{6c},\ p_{7c},\ p_{8c}), \tag{1}$$

where p_{ic} $(i = 1, \ldots, 8)$ is the frequency of moving objects whose trajectory has direction along sector i in cell c. The frequencies are computed for trajectories in the database (a special training set or data in which we want to detect outliers) after the monitored area is partitioned. As a result, each cell is represented by such a vector.
Now, let A be a new trajectory that is to be tested. For each grid cell c the trajectory passes, the instant outlying score is computed. Assume that A has K directions in cell c. Then passing of trajectory A in cell c can be represented by a direction vector $\mathbf{v}_c = (v_{A1}, v_{A2}, \ldots, v_{AK})$ where v_{A1} $(k = 1, \ldots, K)$ are the directions. The direction-based instant outlying score $OScoreDir$ of A in c is computed as:

$$OScoreDir_{Ac} = 1 - \sum_{k=1}^{K} q_k \sum_{i=1}^{8} p_i \cdot cos\angle(v_{Ak}, v_i), \tag{2}$$

where $q_k = 1/K$ $(k = 1, \ldots, K)$ are normalizing constants, $cos\angle(v_{Ak}, v_i)$ is the cosine value of the angle between direction v_{Ak}, which is the k-th direction of A in c, and v_i, which is the center direction of the i-th direction sector of c as it is shown in Figure 6.
The outlying score can also be based on the density of trajectories. Assume that each cell c is represented by the number d_c of trajectories from the database passing it. Then the density-based outlying score of A in c is defined as:

$$OScoreDen_{A_c} = \begin{cases} s & \text{if } d_c < \tau \\ 0 & \text{otherwise} \end{cases} \quad (3)$$

where a density score s is a penalty for a low density and τ is a density threshold.
The main idea of TOP-EYE is that abnormal behavior of a moving object is gradually reflected in its moving characteristics represented by its trajectory. Therefore, it is possible to detect the trajectory outliers in an evolving way combining the instant outlierness of the object with the influence of its prior movement. The authors introduce an exponential decay function $\exp(-\lambda\Delta t)$ to control the influence. Here, λ is a user-specified parameter that determines the decay rate and Δt is a time interval between the time for which the current instant outlying score is calculated and some time t in history of the object movement for which the instant outlying score was calculated before. Let t_0 be the initial time instant for which the instant outlying score obtained by Equation 2 or Equation 3 is S_{t_0}, next time instant is t_1 and score S_{t_1} and so on. Then the evolving outlying score at time instant t_i is calculated as:

$$EScore_{t_i} = S_{t_i} + S_{t_{i-1}} \cdot \exp(-\lambda\Delta t_{i-1}) + S_{t_{i-2}} \cdot \exp(-\lambda\Delta t_{i-2}) + \cdots + S_{t_0} \cdot \exp(-\lambda\Delta t_0) \quad (4)$$

The authors show in [2] that the evolving outlying score as defined in Equation 4 both makes it possible to detect an trajectory outlier in its early stage and can be robust with respect to the accidental increase of the instant outlying score if the score threshold is properly set.
Because of promising properties of the TOP-EYE method we prepared experiments on real-world data sets. We were mainly interested in its applicability for mining outlying trajectories in surveillance systems.
The algorithm and a user-friendly application for testing was implemented in Java [12]. We carried out experiments with two data sets. The first one contained trajectories extracted from videos we use in our research and development of a multi-camera surveillance system SUNAR [1]. This dataset will be referred to as SUNAR dataset. The second one was a Hurricane dataset[1] which is one of datasets that were used in experimental evaluation of the TRAOD algorithm mentioned in Subsection 3.1. The comparison of results of TOP-EYE with results of another trajectory outlier detection algorithm was also one of our goals.
The SUNAR dataset contained trajectories extracted from a set of videos from five cameras monitoring some space at the airport that comes from i-LIDS dataset[2]. We found that the quality of extracted trajectories, in many cases, was low. The task of object detection and tracking in a monitored space where there are several people moving in various directions is difficult. When we compared the discovered trajectory outliers with the videos from which the trajectories were extracted, we have found that many of them are not real trajectories of one particular person. Instead, they were composed of segments of trajectories of several people. One of correct density-based trajectory outliers that represent the movement of only one person is shown in Figure 7.

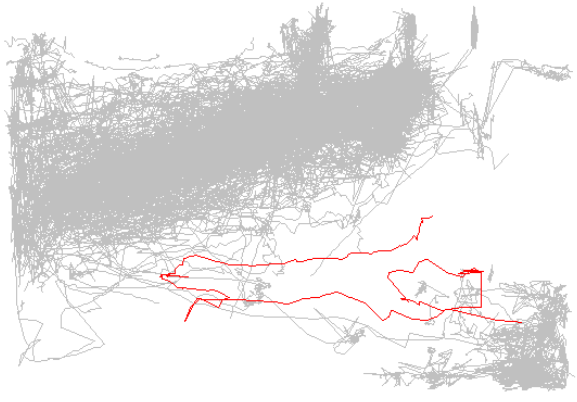

Figure 7. An example of trajectory outlier detected in SUNAR data.

[1] *http://weather.unisys.com/hurricane/atlantic/*
[2] *http://www.homeoffice.gov.uk/science-research/hosdb/i-lids/*

The parameters from Equations 3 and 4 were set to $\lambda = 0.4$, $s = 2$ and $\tau = 10$. The threshold of the evolving outlying score was set to 3.0 and the size of the grid was 50 units. The snapshot from corresponding video is in Figure 8. The person detected as an outlier is marked with blue rectangle.

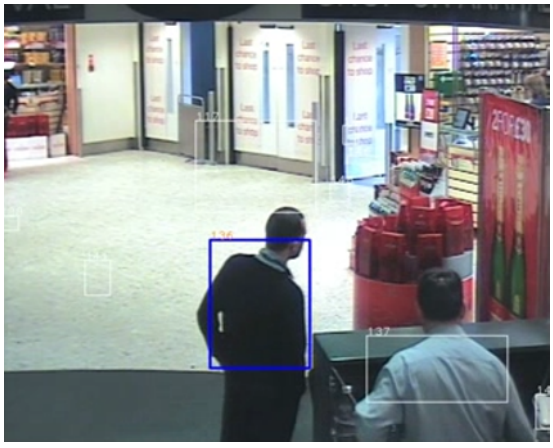

Figure 8. Snapshot from video with object detected as outlier. The video was adapted from i-LIDS dataset.

The Hurricane dataset that we used in experiments contained trajectories of Atlantic hurricanes from 1950 to 2008. Each trajectory is described as a sequence of time, longitude and latitude values together with some meteorological data registered in six-hour intervals. The result of the direction-based trajectory outliers detection with $\lambda = 0.7$ is in Figure 9.

Figure 9. Hurricane direction-based trajectory outliers.

The detected density-based trajectory outliers for parameters $\lambda = 0.7$, $s = 1$, $\tau = 10$ and the grid cell size of $5°$ is shown in Figure 10. The thresholds for direction-based score and density-based score were set to 2.5 and 1.8 respectively.
Our Hurricane dataset that was very similar to the one used in [7] to evaluate the algorithm TRAOD. The result adapted from that paper is in Figure 11. It can be seen that TRAOD detects outlying partitions of the trajectories, but not necessarily the whole trajectories.
We have also studied the influence of the algorithm parameter settings. The value of the decay rate λ makes it possible to control the influence of the historical trajectory outlierness on the current value of the evolving score. The number of discovered outliers decreases with increasing value of λ.
The density score s is a penalty for passing a grid cell whose density is below the threshold τ. For a given threshold of the evolving score, the increasing value of s increases the number of detected outliers. The increase of τ also results in the increase of the number of detected outliers.

Figure 10. Hurricane density-based trajectory outliers.

Figure 11. Hurricane trajectory outliers detected by TRAOD. Adapted from [7].

The influence of the grid cell size on the number of detected density-based trajectory outliers is illustrated in Figure 12. The bigger the cell size, the less the number of addends in Equation 4. This experiment was done with SUNAR dataset and values of parameters $\lambda = 0.75$, $s = 1$ and $\tau = 5$. The evolving score threshold was set to 2.5.

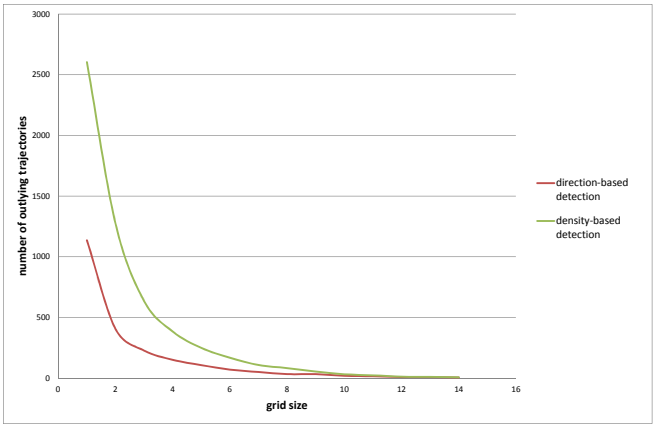

Figure 12. The dependency of the number of detected outliers on the grid cell size.

Several other experiments were focused on time complexity, namely on the dependency of processing time on grid cell size, number of trajectories and total number of points of all trajectories. The values of parameters were set to $\lambda = 0.75$, $s = 1$ and $\tau = 5$ in all experiments. We measured both time of model building and time for both direction-based and density-based trajectory outlier detection.
Figure 13 illustrates the dependency on the grid cell size.

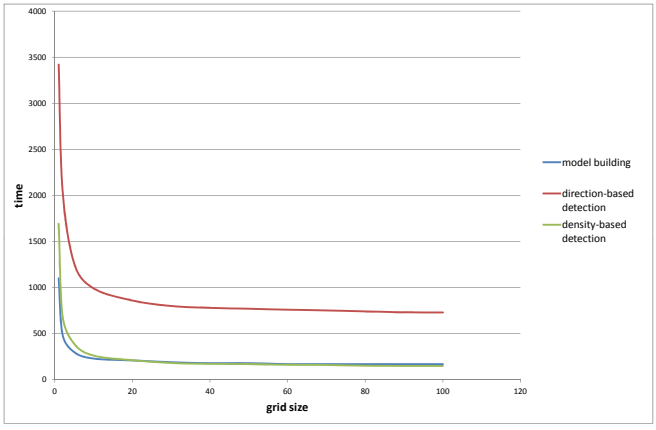

Figure 13. The dependency of the processing time on the grid cell size.

The dependency of processing time on the number of trajectories is shown in Figure 14. The dependency on the total number of points was very similar. It is evident that there are no differences in time complexity between model building and density-based outlier detection in both cases, which could be expected due to the principle of the algorithm. Although direction-based outlier detection is a bit more time consuming than density-based one, it falls into the same complexity class.

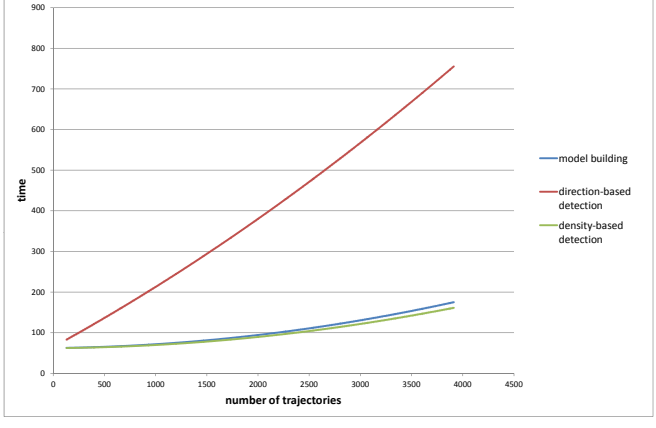

Figure 14. The dependency of the processing time on the number of trajectories.

4. Conclusion

Recently popular location-based services produce much moving-object trajectory data. Mining movement patterns of such objects and detecting trajectory outliers are two important analysis tasks with real-world application potential. This paper introduces fundamental concepts related to them and presents several representative approaches and techniques to solve them.

Our experimental results provided by the TOP-EYE trajectory outlier detection algorithm on real data are briefly discussed here. The TOP-EYE method is able to capture an evolving nature of outlying trajectories and to identify outlying partitions of the trajectories. The main disadvantage of this method can be a consideration of the monitored area as a regular grid and a relatively large number of user-specified parameters.

Based on our experimental results and advantages and disadvantages of the TOP-EYE algorithm, our future research will focus on finding a method that preserves the advantages of the TOP-EYE algorithm and eliminates its drawbacks.

Acknowledgment

This work was supported by the grant VG20102015006 of the Ministry of the Interior of the Czech Republic, the European Regional Development Fund in the IT4Innovations Centre of Excellence project (CZ.1.05/1.1.00/02.0070), the project CEZ MSM0021630528 Security-Oriented Research in Information Technology and the specific research grant FIT-S-11-2.

References

[1] Chmelar P., Lanik A., Mlich J., SUNAR: Surveillance Network Augmented by Retrieval, LECT NOTES COMPUT SC, 2010, 6475, 155-166

[2] Ge Y., Xiong H., Zhou Z.-H., Ozdemir H., Yu J., Lee K.C., TOP-EYE: Top-k Evolving Trajectory Outlier Detection, In: Huang J. et al. (Eds.), Proceedings of the 19th ACM International Conference on Information and Knowledge Management (CIKM) (Toronto, Canada, October 2010), ACM, 1733-1736, 2010

[3] Gudmundsson J., van Kreveld M., Speckmann B., Efficient Detection of Patterns in 2D Trajectories of Moving Points, Geoinformatica, 11, 195-215, 2007

[4] Han J., Li Z., Tang L.A., Mining Moving Object, Trajectory and Traffic Data, Lect. Notes Comput. Sc., 5982, 485-486, 2010

[5] Han J., Li Z., Tang L.A., Mining Moving Object, Trajectory and Traffic Data, DASFAA 2010 Tutorial, http://www.cs.uiuc.edu/homes/hanj/pdf/dasfaa10_han_tuto.pdf

[6] Knorr E.M., Ng R.T., Tucakov V., Distance-based outliers: algorithms and applications, VLDB J., 8, 237-253, 2000

[7] Lee J.-G., Han J., Li X., Trajectory Outlier Detection: A Partition-and-Detect Framework, In: Proceedings of the 2008 IEEE 24th International Conference on Data Engineering (ICDE) (April 2008, Cancun, Mexico), IEEE Comp. Soc., 140-149, 2008

[8] Li Z., Ding B., Han J., Kays R., Swarm: Mining Relaxed Temporal Moving Object Clusters, In: Proceedings of the VLDB Endowment, 3, 723-734, 2010

[9] Li Z. et al., MoveMine: Mining Moving Object Data for Discovery of Animal Movement Patterns, ACM Transactions on Intelligent Systems and Technology (ACM TIST), 2, 37:1-37:32, 2011

[10] Li X., Han J., Kim S., Gonzalez H., ROAM: Rule- and Motif-Based Anomaly Detection in Massive Moving Object Data Sets, In: Proceedings of the 7th SIAM International Conference on Data Mining (SDM) (Minneapolis, USA, April 2007), SIAM, 273-284, 2007

[11] Liu L., Fan J., Qiao S, Song J., Guo R., Efficiently Mining Outliers From Trajectories of Unrestraint Movement, In: Proceedings of the 3rd International Conference on Advanced Computer Theory and Engineering (ICACTE) (Chengdu, China, August 2010), V2-261-V2-265, 2010

[12] Pešek M., Získávání znalostí z časoprostorových dat, MSc thesis (FIT BUT, Brno, Czech Republic, 2011) (in Czech)