Discovering Trajectory Outliers between Regions of Interest

Vitor Cunha Fontes[1], **Lucas Andre de Alencar**[1], **Chiara Renso**[2], **Vania Bogorny**[1]

[1]Dep. de Informática e Estatística – Universidade Federal de Santa Catarina (UFSC)
Caixa Postal 476 – 88.040-900 – Florianópolis – SC – Brazil

[2]KDD-LAB – University of Pisa
Pisa, It.

{vitor.fontes,lucas.alencar}@posgrad.ufsc.br, vania.bogorny@ufsc.br

chiara.renso@isti.cnr.it

Abstract. *Different algorithms have been proposed in the last few years for discovering different types of behaviors in trajectory data. Existing approaches, in general, deal only with the outliers, and do not consider the standards routes and regions of interest. In this paper we propose a new algorithm for trajectory outlier detection between regions of interest. We show with two experiments on real data that the method correctly finds outlier patterns.*

1. Introduction and Motivation

Current advances in mobile technology have increased the interest in mobility data analysis in several application domains. Very simple actions as carrying a mobile phone may register the trace of an object. Some devices specially developed for tracking like GPS or sensor networks may capture the movement of people, animals, cars, boats, buses and natural phenomena. These tracks are called *trajectories of moving objects*. Several data mining methods have been proposed for extracting different types of patterns from trajectories. Some examples of trajectory patterns are chasing [de Lucca Siqueira and Bogorny 2011], objects moving together in flocks [Laube et al. 2005], sequences of visited places [Giannotti et al. 2007], periodic movements [Cao et al. 2007], outliers [Lee et al. 2008], and avoidance [Alvares et al. 2011]. In this paper we focus on trajectory outlier detection.

Trajectory outliers can be very useful in traffic analysis. This type of movement analysis between regions of interest is useful to help to understand the flow of people that move between the regions, how this flow is distributed and what are the characteristics of the movements. In high traffic areas outliers can show alternative paths that can reduce the volume of cars, or reveal the best or worst path that connects two regions. Moreover, the outliers can be interesting to discover suspicious behaviors, like company cars that scape from their normal route.

Figure 1 shows some examples of trajectory outliers moving between two regions. There are six trajectories that move from region R_1 to region R_2. Trajectories T_2, T_3 and T_4 move close to each other, using a similar path (probably the same) to move from R_1 to R_2, and form the standard path. If we consider that R_1 is a Shopping center area in a city and R_2 is the downtown region, there is a high probability that T_2, T_3 and T_4 followed a common route to move between the regions, while T_1 used an alternative way in its movement. Trajectory T_1 is far from the group (T_2, T_3 and T_4), so it may characterize

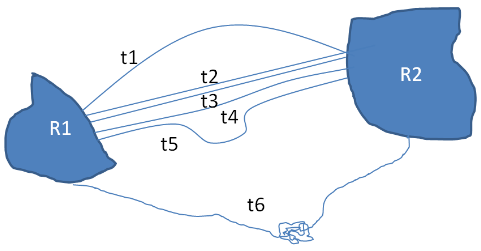

Figure 1. Examples of trajectory outliers.

an outlier in relation of the group. Trajectory T_5 took an alternative route in part of its movement (made a detour) in the middle of the way from R_1 to R_2. Trajectory T_6 made a very long detour, on the way from R_1 to R_2. By observing the movement of T_1, T_5 and T_6 we notice that these trajectories made a movement different from the rest of the trajectories (the standard path), what characterizes an outlier.

In this paper we present an algorithm to find spatial and spatio-temporal outliers between trajectories, and in summary, we make the following contributions in relation to existing approaches: (i) Define a different type of outlier pattern in trajectory data analysis, (ii) find both the standard path and the outlier patterns between regions of interest and (iii) define a new algorithm for discovering spatial and spatio-temporal outliers. It is important to mention that in this paper (at this first step) we are not interested in discovering why an object avoided a group, but to discover the main route and alternative ways to move between regions of interest.

The rest of the paper is organized as follows: section 2 presents the related works. Section 3 presents the main definitions and the algorithm. Section 4 presents experiments on real trajectory data. Finally, section 5 concludes the paper and suggests directions of future research.

2. Related Works

Several types of patterns can be extracted from trajectories. Laube in 2005 [Laube et al. 2005] proposed five types of trajectory patterns based on movement, direction, and location, which are very well known: convergence, encounter, flock, leadership, and recurrence.

Lee [Lee et al. 2008] proposed an algorithm to find outliers, which are the trajectories that move differently from the rest of the trajectories in the dataset. No regions of interest, standard path or time is considered. In [Li et al. 2007] an approach is proposed to find hot routes. These routes are discovered based on the density of the roads, and not among trajectories that move together in space between regions of interest. A similar work for discovering popular routes is proposed by [Chen et al. 2011]. This approach considers as regions of interest the origin and destination of the trajectories and hot routes are discovered based on trajectory turns. The routes where several trajectories make turns are considered popular.

A closer approach to our method could be the T-pattern [Giannotti et al. 2007]. It is a sequential trajectory pattern mining algorithm that first generates regions of interest considering dense areas in space, and than computes sequences of visited regions, taking into account transition time from one region to another and minimum support. Although it finds the trajectories that move between regions, it does not look at the path followed by the objects, if they move together, or if there is a standard route. The basic idea of our approach is to detect if there is a standard path to move between places and to find the trajectories that avoid this path. In the following section we present the basic concepts for outlier patterns and the proposed algorithm.

3. Mining outlier patterns from Trajectories

Before defining the outlier we present some definitions like point and trajectory.

Definition 1 *Point. A point p is a tuple (x, y, t), where x and y are spatial coordinates and t is the time instant in which the coordinates were collected.*

Definition 2 *Trajectory. A trajectory T is a list of points $\langle p_1, p_2, p_3, ..., p_n \rangle$, where $p_i = (x_i, y_i, t_i)$ and $t_1 < t_2 < t_3 < ... < t_n$.*

Usually the patterns do not hold for the whole trajectory or during the complete trajectory life. Trajectory patterns occur in part of the trajectories, and this is specially true for outlier. Therefore, we make use of subtrajectories, that is a concept commonly used in trajectory research.

Definition 3 *Subtrajectories. Let $T = \langle p_1, p_2, ..., p_n \rangle$ be a trajectory. A subtrajectory S of T is a list of consecutive points $\langle p_k, p_{k+1}, ..., p_m \rangle$, where $p \in T, k \geq 1$, and $m \leq n$.*

Most existing works for trajectory pattern mining look for patterns in the whole dataset, without having a specific interest. For instance, for chasing patterns, flocks or outliers, the whole dataset is searched. When looking for *outlier patterns* in trajectory data we first look for trajectories that move around the same places. It would not make much sense to compare a trajectory that moves in Paris around Eiffel Tower with a trajectory moving around Hotel des Invalides. Trajectories should be in close areas to deviate from others. Therefore, we look for outlier patterns between *regions of interest*.

Regions of interest can have different size and format, depending on the application. Regions of interest can be districts, dense areas, hot spots, important places, etc. A region can be a pre-defined important place or computed by an algorithm that finds dense areas. How to find these regions is not the focus of this work, but we consider a region as a polygon, as in [Giannotti et al. 2007], that is a well known concept in GIS comunity.

The use of regions allows filtering from the whole dataset only the subtrajectories that move between the same regions, and outliers will be searched among these sets, what significantly reduces the search space for outlier. It is important to mention that at this point, among the trajectories that cross specific areas, we are only interested in the part of the trajectories (subtrajectories) that move between the regions, and not in the trajectory inside the area. We call these subtrajectories that move between regions as *candidates*.

We define candidate as the smallest subtrajectory that moves between two regions, i.e., we take the last point of the subtrajectory that intersects the first region and the first point that intersects the final region, as shown in Figure 2(left). In this example the candidate has the points from p_i to p_m.

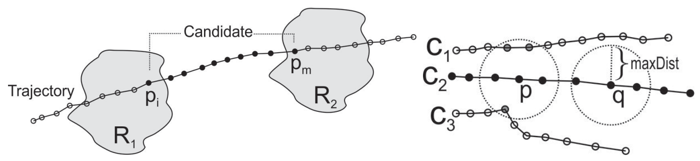

Figure 2. (left)Example of candidate (right) Example of neighborhood.

Definition 4 *Candidate. Let R_1 and R_2 be two regions such that $R_1 \cap R_2 \neq 0$ and T a trajectory. A candidate from R_1 to R_2 is the subtrajectory $S = \langle p_i, p_{i+1}, ..., p_m \rangle$ of T, where $(S \cap R_1) = \{p_i\}$ and $(S \cap R_2) = \{p_m\}$.*

After defining the set of candidates we start looking for outliers. A candidate will be an outlier when it follows a different path in relation to the majority of the candidates from its group. We can say that a path that is different from the route used by most candidates is of low density, and it has less trajectories around, while a crowded path has many trajectories in its neighborhood. In order to discover these two types of paths we introduce the concept of neighborhood, that is computed for each point of the candidate. A candidate is a neighbor of a point if it is close to the point. If a point has a few candidates in its neighborhood, then at that time the moving object was following a path different from the majority of candidates. The maximal distance for a candidate to be a neighbor of a point is called *maxDist*.

Definition 5 *Neighborhood. Let p be a point. The neighborhood of p $N(p, maxDist) = \{c_i | c_i$ is a candidate and $\exists q \in c_i, dist(p, q) \leq maxDist\}$.*

Figure 2(right) shows an example of neighborhood. The neighborhood of point p are the candidates C_1 and C_3, since these two candidates have at least one point inside the radious of size *maxDist* around p. Notice that point q has no candidates inside its radios of size *maxDist*, so its neighborhood is empty. We can conclude that at point p, C_2 was moving with C_1 and C_3 (same path), but at point q, C_2 was moving far from C_1 and C_3 (different path).

In general, there exist one or more frequent paths (main routes) to move from one region to another, and which are more frequently used than alternative ways. To find these standard paths we use the minimum support concept (*minSup*), which is the minimal amount of candidates that a point should have in its neighborhood to be part of a crowded or dense path. In the example in Figure 2(right), considering *minSup* $= 2$, the point p in candidate C_2 is in a dense path, while the point q in C_2 moves alone. A candidate that has all its points in a crowded path is considered a *standard*.

Definition 6 *Standard. Let $c = \langle p_1, p_2, p_3, ..., p_n \rangle$ be a candidate, c is a standard candidate if and only if $\forall p_i \in c, | N(p_i, maxDist) | \geq minSup$.*

The candidates that have at least one point where the cardinality of its neighborhood is less than *minSup* are called *potential outlier*. Therefore, the candidates are split in standards and potential outliers, such that a candidate will always be either a standard or a potential outlier. When all candidates between two regions are potential outlier, there is no standard. As a consequence, there is no standard path that an object could avoid or deviate. On the other hand, if there is at least one standard path, then the potential outlier did really perform a detour, and becomes a *spatial outlier*.

An important remark here is that no outlier will exist if there is no standard path. This is one of the main difference of our approach in relation to existing works on trajectory pattern mining. So the first assumption to define an outlier is that it should move between two regions of interest. The second is that there must be a standard path that connects the regions such that the outlier should deviate from it. Therefore, any subtrajectory that uses a path different from the standard is an outlier.

Definition 7 *Outlier. Let C be the set of candidates between two regions. A potential outlier is an outlier O if $\exists c \in C | c$ is a standard.*

When two candidates leave the start region at the same time interval we can say that they are *synchronized*. For instance, when two students leave the university together to go to the cinema, we can say their trajectories are synchronized. Two candidates leave the same region at the same time interval if the difference between the timestamps of the first point of the candidates is less than a given *time tolerance*. When the trajectories in the standard path are synchronized with the outlier, then the outlier becomes *spatio-temporal*.

Definition 8 *Spatio-temporal outlier. Let C be the set of candidates between two regions. An outlier O is a spatio-temporal outlier if $\exists c \in C | c$ is standard and c is synchronized with O.*

In this work we analyze the time that the objects leave the starting region, since the objective is to know if they have a synchronized departure, and it is not relevant here if they keep the synchronization during the entire movement until reaching the destination. After defining the main concepts related to outlier patterns, we show in listing 1 the pseudo-code of the algorithm. The input of the algorithm is a set of trajectories T, a set of regions of interest R, the maximal distance (*maxDist*), the minimum support (*minSup*) and the *TimeTolerance*.

Listing 1. Algorithm

```
1   INPUT:
2   T;  // Set of trajectories
3   R;  // Set of regions
4   maxDist; // maximum distance
5   minSup;  // minimal number of neighbour
6   TimeTolerance;
7
8   OUTPUT:
9   Set of semantic spatial and spatio−temporal outliers.
10
11  METHOD:
12  FOR EACH PAR OF REGIONS (startRegion, endRegion) in R{
13      C = findCandidates(T, startRegion, endRegion); // find candidates.
14      StandardSet = findStandard(C, maxDist, minSup); // find standards.
15      IF ( StandardSet != EmpytSet ) {
16          SpatialOutSet = C − StandardSet; // Set of spatial outliers
17          FOR EACH outlier out in SpatialOutSet {
18              out.Time_granularity_refinement;
19              out.Comput_synchronized_standards(TimeTolerance);
20              IF (out.duration > avg_duration_standards)
21                  Out.is("slower outlier");
22              IF (out.duration < avg_duration_standards)
23                  out.is("faster outlier ");
24
25
26  } } }
27  return out
```

For each pair of regions (line 12), the algorithm starts by computing the candidates that move from *startRegion* to *endRegion* (line 13), with the function *findCandidates*. This function checks for every trajectory if it intersects the pair of regions. Once the candidates are computed, the algorithm searches for the standards with the function *findStandard* (line 14), considering the parameters *maxDist* and *minSup*. The function *findStandard* checks for all points of a candidate in the set if the number of neighbors is greater than *minSup*. If this is the case, then the candidate is considered a standard. If the set of standards is not empty (line 15), then it goes for finding the spatial outliers, since there is a standard path that connects both regions.

Spatial outlier are all candidates which are not standards (line 16). Once we have the standard path and the spatial outlier, the algorithm starts the time analysis. For each spatial outlier (line 17) the algorithm discretizes the time dimension (line 18). Instead of simply showing the timestamp of the spatial and spatio-temporal outlier, as has been done in most data mining algorithms, we automatically discretize the time for the user to rapidly identify the periods of the outlier. Such discretization simplifies postprocessing steps. For this purpose, the algorithm extracts from the timestamp several information, including: the day of the week that the outlier occurred, the period of the day , and the month of the year. Such granularity refinement is useful to interpret the patterns.

It is important to notice that we first discover the patterns, and afterwards interpret them (discretize the time). If a time interval was defined a priori and the data filtered by this time interval in preprocessing steps, the method would be very limited and several patterns of previously unknown periods would never emerge. So the idea is to discover the standard path and outlier trajectories for then checking when these patterns occur.

The next step of the algorithm is to check if the outlier is synchronized with any standard, i.e., if there are standards that leave the start region at a similar time as the spatial outlier (line 19). In case there is a synchronized standard, then the spatial outlier becomes a spatio-temporal outlier. In the last step the algorithm verifies if the duration of the outlier is greater than the average duration of the standards (line 20). When the outlier is spatio-temporal, the average duration is compared only with the synchronized standards. If the duration of the outlier is greater, it means that the outlier took more time to move between the regions, and is classified as slower outlier. If its path was faster, the outlier is classified as faster outlier.

In this section we presented the main concepts related to trajectory outlier detection and presented an algorithm to find both the standard and outlier subtrajectories. The following section presents two experiments with real trajectories.

4. Experimental Results

In this section we evaluate the proposed method with two datasets with different characteristics. The first are trajectories of cars of people that leave and work in the city of Porto Alegre. It is a dense dataset, where trajectory points are collected every second. The second dataset are trajectories of taxi drivers in the city of San Francisco, and the trajectory points are collected in an average of one minute. More detailed experiments with other datasets and a better comparison with the method TRAOD can be found in [Fontes and Bogorny 2013].

4.1. Porto Alegre Dataset

This experiment considers a dataset with 241 trajectories, with a set of 197959 points. As mentioned before, the sampling rate is one second. Figure 3 shows this dataset over

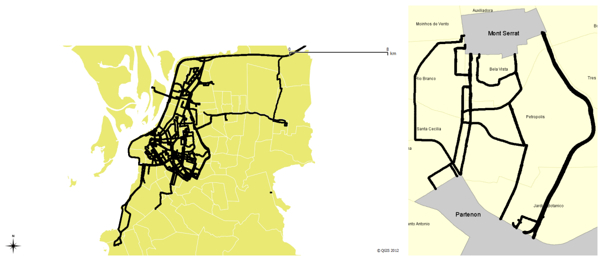

Figure 3. (left) Car Trajectories in Porto Alegre (right) Candidate trajectories between districts MontSerrat and Partenon.

a map of districts of the city. In this experiment we considered as interesting regions two districts which are crossed by the highest number of trajectories: Montserrat and Partenon. Among the 241 trajectories, 59 cross these districts, so there are 59 candidates, shown in Figure 3 (right).

This experiment was performed considering 50 and 80 meters as the *maxDist*, but we show the results for 50 meters only since the results were quite similar. Minimum support *minSup* was set to 10, indicating that at least 10 candidates should move in a distance of around 50 meters for generating a standard path. The *TimeTolerance* was set to 10 minutes, but no spatio-temporal pattern was found, because the dataset has not many synchronized trajectories.

Among the 59 candidates, 29 subtrajectories move from Partenon in direction to Montserrat, and from the 29 subtrajectories, 26 move in the standard path, which is shown in Figure 4(left), and only 3 are outlier. This shows that the standard path is used by the majority of the trajectories that move between these districts. The standard path corresponds to the Carlos Gomes Avenue, which is a popular street that crosses Porto Alegre. The average duration of the trajectories in the standard path is 6 minutes. Only two standards took more time than the average (11 and 13 minutes), and both happened at the end of the day. The majority of the standards (17) happened at morning.

We show two outlier patterns moving between these regions. Figure 4(center) shows an outlier moving from Partenon to Montserrat which was faster than the standard path, taking 4 minutes to make his trip (35% faster than the standards). As can be seen in the Figure, this path is shorter than the standard, and can be a good alternative for avoiding travel on the standard path. This pattern follows the Lucas de Oliveira Avenue. Another outlier pattern took 10 minutes in its movement, making a longer trip, as can be seen in Figure 4(right).

We compare the output found in this experiment with the TRAOD algorithm [Lee et al. 2008]. This comparison is performed to show that both methods discover different patterns, which is mainly obvious since the proposals are different. The algorithm TRAOD does not consider regions, the standard path and it does not perform any further analysis over outliers, but in order to compare the results of both algorithms we considered the same trajectory candidates as input for both methods. Different input would generate different output. TRAOD has as input the maximal distance between trajectory partitions (D), the maximal percentage of trajectories (p) for not being outliers and the fraction (F)

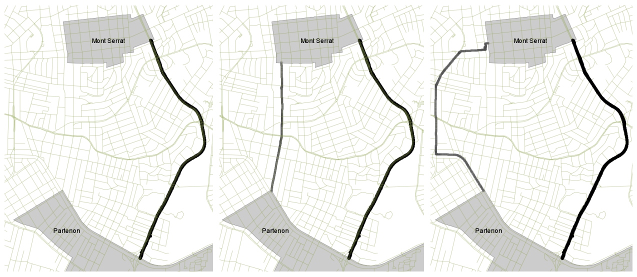

Figure 4. (left) standard path from Partenon to Montserrat; (center) faster outlier moving from Partenon to Montserrat; (right) slower outlier moving from Partenon to Montserrat.

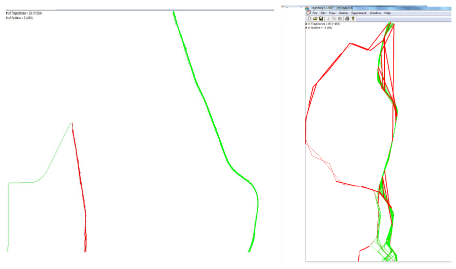

Figure 5. (left) Results for TRAOD (D=50, p=0.7, F0.2) for the POA dataset and (right) Results with parameters D=100, p=0.9, F=0.2 for the taxi dataset.

of partitions that a trajectory should have to be an outlier. We ran the TRAOD, with the parameters D = 50, p = 0.7 and F = 0.2 (these parameters are close to the parameters used in the experiment with our algorithm). We keep the original algorithm output, therefore outliers are shown in red while trajectories are shown in green, so it is not possible to overlap the output with the geographic map and show the regions. TRAOD transforms subtrajectories in lines, what makes the result a bit different. The algorithm found only 2 outliers, as shown in Figure 5(left). It is important to notice that the output of TRAOD is the total number of outliers and the outliers presented over the set of trajectories.

The size of the regions may influence the standard path and outlier patterns. However, in this paper we are only interested in the part of the trajectories that move between the regions, and not in the trajectory inside the area. The analysis of the part of the trajectory inside the region can be interesting to understand why an object avoided the standard path, but this is out of the scope of this paper. To avoid much influence of the size of the region in the patterns, the size of the regions should not be so large. In the next experiment we considered very small regions, such that the size of the region should not much influence the selected route.

4.2. Taxi trajectories in San Francisco

This experiment was performed with trajectory data collected in the city of San Francisco, California. This dataset contains trajectories of taxi drivers. We considered trajectories of one month, with 1.8 million points. One trajectory corresponds to the movement of one taxi driver during the whole day, and the time collection interval is in average 1 minute.

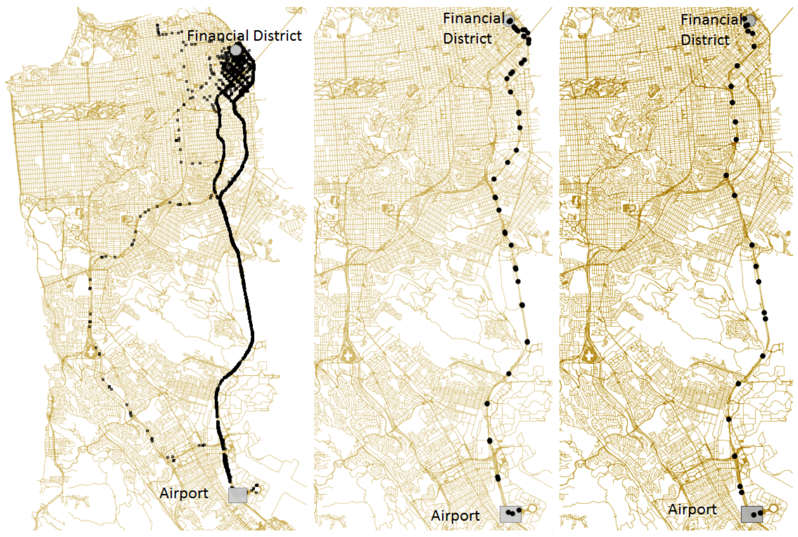

Figure 6. (left) candidates that move from airport (gray rectangle) to the financial district (gray circle), (center) standard path from Airport to Financial District, (right) standard path from Financial District to Airport.

Even with such a large time interval between every two points (which in general is every second) the method obtained very good results for the standard path and outlier patterns, what shows that the algorithm can deal with different types of trajectories.

Each taxi trajectory has an attribute *occupation*, which states if the taxi has passengers or is empty. Here we are interested in discovering the standard path and outlier patterns only when the taxi has passengers. This is interesting to discover those drivers that make detours from the main route. Therefore, we removed the trajectories with no passengers and split each trajectory of the same driver in a different one when the passenger changes. After splitting the trajectories the dataset resulted in 76.885 trajectories, having a total of 842.455 points.

In this experiment we want to analyze the movement between some specific places. We considered the trajectories of taxis moving from the airport to the Financial district in San Francisco. The parameters were distance of 100 meters for finding the neighbors, because of the large distance between the trajectory points, minimun support was set to 30 and time tolerance to 20 minutes. A total of 154 candidates was generated, what means that 154 objects traveled from the airport to the financial district. Between airport and financial district two standard paths were found, one from airport to financial area and another in the opposite direction. Figure 6 shows the candidates (left), the standard path from airport to the financial district (center) and the standard path in the opposite way (right). The average travel time on the standard path in this case is 18 minutes and has a length of 26 km. The standard path leaving the airport starts at Bayshore Freeway (US 101), changes to John F. Foran Freeway and follows to the King Street, later to Folsom Street and finally turns to Fremont Street.

Figure 7 shows different examples of outliers, where triangles are the outliers and

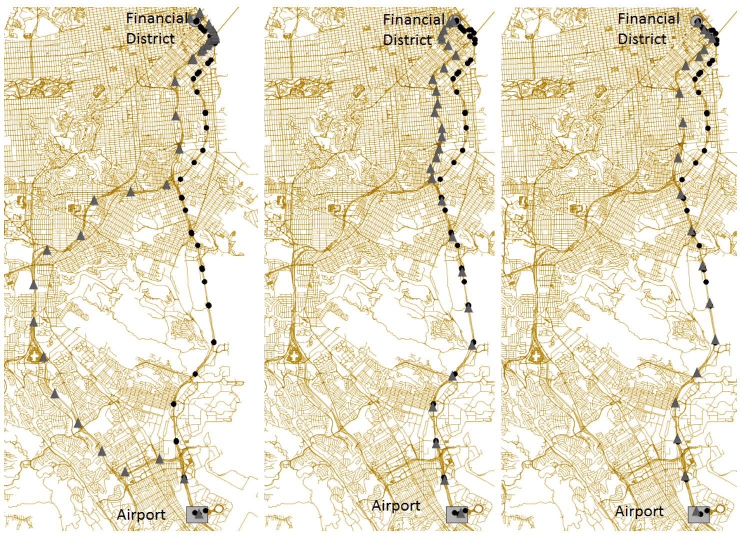

Figure 7. (left) very slow Outlier - 47 minutes (center) slow Outlier - 25 minutes (right) fastest Outlier - 17 minutes

circles the standard path. One outlier made a long detour taking 47 minutes and traveling 36 km (Figure 7 (left)). Another one also made a longer detour, taking 25 minutes and traveling 27 km (Figure 7(center)). Both trajectories that traveled a longer distance and took more time than the standard path were generated on Sunday afternoon and Saturday evening, respectively, characterizing a strange behavior for a weekend, where traffic flow should be normal. The last outlier shown in Figure 7 (right) was a little bit faster than the standard path (taking 17 minutes). This is a spatio-temporal outlier, i.e., this subtrajectory left the airport at the same time as the trajectories in the standard path.

A last analysis is on the standard path from the Financial Area to the Airport. The standard path which connects these regions is different from the previous one, as can be seen in Figure 6 (right), and is faster, taking in average 15 minutes, while the previous one takes 18. The traveled distance is the same, 26 km. Among the three examples of outlier shown in Figure 8, all examples are slower, taking respectively 30, 21, and 17 minutes, showing that for this direction the standard path is the best option. One driver made a very big detour. Two outlier trajectories partially followed the standard path, but when leaving the financial area each one took a different route, i.e., a slower one. The outlier on the right side in the figure is spatio-temporal. We compare the output found in this experiment with the TRAOD algorithm [Lee et al. 2008] that found different outliers even within the standard path (Figure 5(right)).

In general, most outlier patterns take more time to travel between the regions, and the standard path should be a better option if the user is more familiar with it.

4.3. Parameter Analysis

As in any data mining algorithm, the parameter definition is a concern, and it directly influences the results of the algorithm. The algorithm makes use of three parameters only: *maxDist*; *minSup* and *TimeTolerance*. *maxDist* is used to check if trajectories use

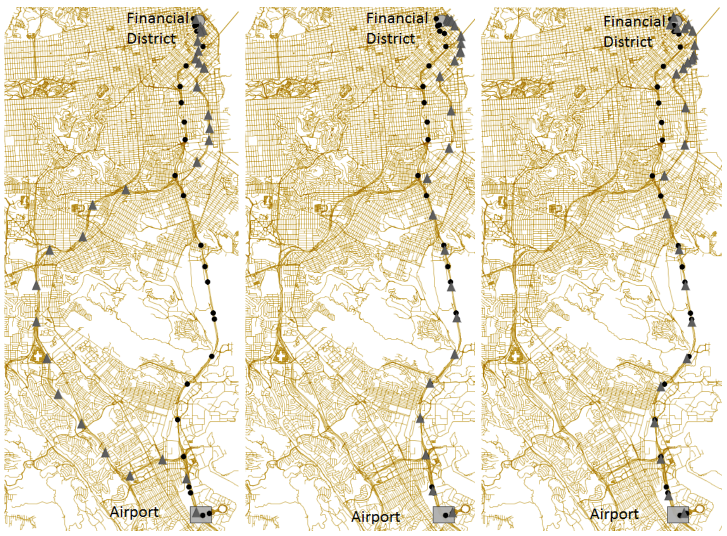

Figure 8. (left) very slow Outlier - 30 minutes (center) slow Outlier - 21 minutes (right) fastest Outlier - 17 minutes

the same path to move between regions. The best value for this parameter is the width of the streets, since outlier patterns are interesting for trajectories in cities. For instance, in cities where the average width of a street is 50 meters, $maxDist$ can be set as 80 meters, considering so 15 meters on each side of the street for GPS impreciseness. In cities with larger streets like 80 or 100 meters, $maxDist$ can be defined as 100 or even as 120 meters. It will depend on size of the streets where trajectories are collected. In narrow streets, $maxDist$ should be lower, while in larger streets it should be higher.

A small *maxDist* may split objects that move in the same path, making it more difficult to find the standards. A very high *maxDist* may join objects that move in different paths (distant paths) in the same group. Therefore, this parameter depends on the application. In our experiments in San Francisco the best parameter was 100 meters, but good results were also discovered with 80 meters.

Minimun support will depend on the density of the dataset. The higher the number of trajectories to be in the standard path, the more difficult it will be to find the standard route. A low minimal support may find several standard paths and less outliers, while a high $minSup$ will generate large amounts of outliers. The minimal support is also application dependent, so it can be high for a dataset where dense regions have several trajectories passing by. The *TimeTolerance* influences the amount of spatio-temporal outlier. The higher the *TimeTolerance* the higher is the chance for several trajectories being traveling within the time window. However, a very high *TimeTolerance* may be meaning less in the sense that trajectories should be moving together.

5. Conclusion and Future Works

In this paper we presented a method for discovering the standard path which connects regions that are interesting for an application domain and the alternative routes to move between these regions, that are called outliers. We presented the definition and an algorithm to discover the outlier trajectories and the standard path. Both dimensions of space and time are considered, therefore allowing the interpretation of the outlier, like: when did it happen; which path is faster; and their duration.

The method presented in this paper is a first step towards trajectory outlier detection and interpretation, and several future works are ongoing. The first one is to distinguish the standard paths between the same regions moving in the same direction. So far we consider as standard path all standard candidates. A second one includes a deep analysis on the standard path and the use of context information around it aiming to discover the intent of the outlier. For instance, if there is a traffic jam in the standard path or an event like a police patrol, such information can help to interpret the outlier. In this method if a subtrajectory has a small portion of it which avoids the standard path it is considered an outlier. In next steps we will evaluate the use of minimal size of an outlier.

6. Acknowledgment

This work has been partially supported by EU project FP7-PEOPLE SEEK (N.295179 http://www.seek-project.eu) and the Brazilian agencies CAPES and CNPQ. Authors would like to thank also for the partial support from CNR-CNPQ Joint project 2012, DataSIM FP7-ICT-270833, and UFSC.

References

Alvares, L. O., Loy, A. M., Renso, C., and Bogorny, V. (2011). An algorithm to identify avoidance behavior in moving object trajectories. *J. Braz. Comp. Soc.*, 17(3):193–203.

Cao, H., Mamoulis, N., and Cheung, D. W. (2007). Discovery of periodic patterns in spatiotemporal sequences. *IEEE Trans. Knowl. Data Eng.*, 19(4):453–467.

Chen, Z., Shen, H. T., and Zhou, X. (2011). Discovering popular routes from trajectories. In Abiteboul, S., Böhm, K., Koch, C., and Tan, K.-L., editors, *ICDE*, pages 900–911. IEEE Computer Society.

de Lucca Siqueira, F. and Bogorny, V. (2011). Discovering chasing behavior in moving object trajectories. *T. GIS*, 15(5):667–688.

Fontes, V. C. and Bogorny, V. (2013). Discovering semantic spatial and spatio-temporal outliers from moving object trajectories. *CoRR*, abs/1303.5132.

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

Laube, P., Imfeld, S., and Weibel, R. (2005). Discovering relative motion patterns in groups of moving point objects. *International Journal of Geographical Information Science*, 19(6):639–668.

Lee, J.-G., Han, J., and Li, X. (2008). Trajectory outlier detection: A partition-and-detect framework. In *ICDE*, pages 140–149. IEEE.

Li, X., Han, J., Lee, J.-G., and Gonzalez, H. (2007). Traffic density-based discovery of hot routes in road networks. In Papadias, D., Zhang, D., and Kollios, G., editors, *SSTD*, volume 4605 of *Lecture Notes in Computer Science*, pages 441–459. Springer.