2011 23rd IEEE International Conference on Tools with Artificial Intelligence

Trajectory Outlier Detection Using An Analytical Approach

Elio Masciari *

Abstract

Trajectory data streams are huge amounts of data pertaining to time and position of moving objects. They are continuously generated by different sources exploiting a wide variety of technologies (e.g., RFID tags, GPS, GSM networks). Mining such amounts of data is challenging, since the possibility to extract useful information from this peculiar kind of data is crucial in many application scenarios such as vehicle traffic management, hand-off in cellular networks, supply chain management. Moreover, spatial data poses interesting challenges both for their proper definition and acquisition, thus making the mining process harder than for classical point data. In this paper, we address the problem of trajectory data outlier detection, that revealed really challenging as we deal with data (trajectories) for which the order of elements is relevant. We propose a complete framework starting from data preparation task that allows us to make the mining step quite effective. Since the validation of data mining approaches has to be experimental we performed several tests on real world datasets that confirmed the efficiency and effectiveness of the proposed technique.

1 Introduction

Trajectory data are intrinsically quite difficult to be analyzed due to their *ordered* characteristic that makes the mining task more complex since the object order is an information to take into account. This difficulty propagates through the knowledge discovery process affecting the quality of the obtained results. In particular we can have imprecise information about: *a) the data*, i.e. we are unsure of what exactly we observe or measure (almost everything in real life exhibits a spatial variability). As an example consider the symptom of natural disasters like hurricanes, such as sudden temperature and pressure variations: a hurricane monitoring system collecting data from different devices should detect the symptoms to plan a proper reaction to the event. Moreover, trajectory data are collected *incrementally* as the source devices generate new data points; *b) the extracted rules*, e.g. we are unsure about the conclusions we can draw from even perfect data, i.e. an unusual data is an outlier or is it the symptom that something is changing in the structure of data (concept drift)?

An outlier is an observation that differs so much from other observations as to arouse suspicion that it was generated by a different mechanism [8]. There exist several approaches to the identification of outliers, namely, statistical-based [4], deviation-based [3], distance-based [9, 2], density-based [5], projection-based [1], MDEF-based [14], and others. In this paper we deal with trajectory data that are data pertaining to time and the position of moving objects (or groups of objects). Trajectory data can be generated in a wide variety of applications, such as GPS systems [7], supply chain management [13] , vessel classification by satellite images [12]. As an example of such data consider moving vehicles, where both cars and trucks leave a digital trace by the vehicular mobile devices that can be collected by wireless network infrastructures. Furthermore, mobile phones continuously signal their locations (cells) when connected to their GSM network. Consider the case of a phone setting up a call and moving trough the network, there may occur the so called hand-off problem, i.e. the cell where the user is moving through does not have enough bandwidth to place the call. In the same way, GPS-equipped portable devices can record their latitude-longitude position at each moment that they are exposed to a GPS satellite, and transmit their trajectories to a collecting server. Moreover, many major retailers ask their providers to exploit RFID technology to tag pallets entering their warehouses and then moving across the supply chain. RFID readers periodically scan the tags appearing in their working area and generate data to report the object being read, the timestamp and the actual location of that object. Such a wide spectrum of pervasive and ubiquitous technologies guarantee an increasing availability of data pertaining to individual trajectories. Due to this large amount of moving objects data generated every day, there is a great need for analyzing it efficiently in order to extract useful information.

Related Work. Despite its importance, trajectory outlier

*Elio Masciari is with the Institute of High Performance Computing and Networks (ICAR-CNR). Address: Via Bucci 41c, I87036 Rende (CS), Italy. Phone: +39 0984 831735. Fax: +39 0984 839054. E-mail: masciari@icar.cnr.it.

1082-3409/11 $26.00 © 2011 IEEE
DOI 10.1109/ICTAI.2011.62

detection has not been paid the attention it deserves, consider as an example of this statement, meteorologists trying to figure out the cause of sudden changes in hurricanes path. In this scenario predicting sudden changes is of prime importance since it is crucial for issuing an evacuation order early. In [10] the authors have presented one of very few attempts. In their technique, a trajectory is represented as a set of key features instead of a sequence of points. That is, a trajectory is summarized by the coordinates of the starting and ending points; the average, minimum, and maximum values of the directional vector; and the average, minimum, and maximum velocities. The distance function is simply defined as the weighted sum of the difference of these values. Then, a distance-based algorithm is applied for detecting trajectory outliers. This technique compares trajectories as a whole; i.e., the basic unit of outlier detection is the whole trajectory. Since our approach also consider the trajectory as a whole we will compare our work with this approach (we will refer to it as *KNORR* in the experimental section). Another approach for outlier detection has been proposed in [11], but the authors search for outlying portions of the trajectories under analysis. Since this approach has a different goal w.r.t. our approach we only mentioned it for the sake of completeness.

Exact vs. Approximate Trajectory Representation. Trajectory data carry information about actual positions and timestamps of moving objects having a detail level often unnecessary. Indeed, many proposals split the search space in regions having the suitable granularity and represent them as areas tagged by an annotated symbol. The sequence of regions define the trajectory traveled by a given object. Based on the above representation of trajectory data (i.e. region based instead of a sequence of multidimensional points) mining steps are performed based on proper techniques. Thus, regioning is a common assumption in trajectory data mining [12, 7]. However, in many application scenarios we need to work on the original data points in order to signal a trajectory as outlying. Consider as an example, trajectories related to RFID readings tracking object movements through the supply chain, if a reader is not fixed due to production needs but it can be moved (e.g. monitoring optional production task) looking at single portions (as in the mentioned related work) of the trajectories could lead to signal as anomalies even trajectories that are not outliers. Due to this observation, we propose in this paper an approach that works directly on the original bi-dimensional trajectory representation. Thus we allow to extract crucial information about common trajectories "shape", i.e. not only the length, but the *actual* direction of movement and the eventual complex turn made by the trajectories. We propose a framework that suitably pre-elaborate original trajectories for the subsequent mining step performed using non separable transforms. Working directly on the multidimensional representation of the trajectories makes the mining process quite effective since we do not require any regioning step (that introduces an approximation in the data being mined). We point out that, in this paper we tackle the outlier detection problem from a different point of view w.r.t. existing approaches since in the outlier definition we exploit a notion of distance between trajectories that will consider all the data points and angular features of the trajectories being analyzed. The presented approach guarantees more flexibility and better performances as will be shown in the experimental section.

Our pre-processing strategy. As trajectories flow into the system we perform a data pre-elaboration based on a proper filtering of the multidimensional points based on *Lifting Schemes*[17]. The aim of lifting is to represent a spatial signal (i.e. the whole trajectory) using a shorter sequence by a proper filtering step that allow prediction and update of proper coefficients. Lifting schemes are successfully exploited in image processing since they offer really good performances without any loss of information. We exploit this approach since in many application scenarios (e.g. RFID trajectory monitoring where we do not want to group set of readers) it is mandatory to work on the overall trajectory considering all the involved dimensions, in such context existing approaches based on regioning are not completely satisfactory.

How to deal with outliers? In many applications, data must be processed efficiently because of real time requirements (e.g. climate disaster prevention). We exploit in this work non separable Fourier transforms as they can be computed incrementally as new data values arise. To this end we define an outlier detection strategy based on multidimensional Fourier Analysis in order to catch "structural" dissimilarities between trajectories. The basic intuition exploited is that a trajectory has a "natural" interpretation as a time series (namely, a discrete-time signal), in which numeric values summarize some relevant features of the elements belonging to the trajectory. In a sense, the analysis of the way the signal shapes differ can be interpreted as the detection of different locations crossed by the trajectories. Moreover, the analysis of the frequencies of common signal shapes can be seen as repeated crossing of the same location. In this context, the proposed approach is an efficient technique, which can satisfactorily evaluate how much two trajectories are similar w.r.t. the structural features previously discussed. We point out that in this work we disregard speed and time information since they are out of the scope of this paper so we assume that points in the trajectories appear at fixed time interval. This is not at all a limitation since we are interested in discovering unusual structures in trajectory datasets so the time information is useless. The

choice of comparing the frequency spectra of trajectories is due to both effectiveness and efficiency issues. Indeed, the exploitation of Non Separable Fourier Transforms (in particular we use Discrete Fourier Transform - *DFT* [15] since it enables us to analyze and manipulate signals in a very powerful way) allows to abstract from minor details which, in most application contexts, should not affect the similarity estimation (e.g., multiple occurrences of the same location due to simple traffic problems). Thus, the comparison is less sensitive to minor mismatches. Moreover, a frequency based approach allows to estimate the similarity through simple measures (e.g., vector distances) which are computationally less expensive than techniques based on the direct comparison of the original trajectory structures. To summarize, we propose to represent a trajectory as a time series of fixed frequency, in which each trajectory point corresponds to an impulse. By analyzing the frequency spectra of the signals so far obtained, we can hence state the degree of similarity between trajectories. It is worth noticing that the overall cost of the proposed approach is only $O(N \log N)$, where N is the maximum number of points in the trajectories to be compared. This low complexity is well suited for dealing with application scenarios where a quick outlier identification is mandatory. Moreover, as will be shown in the experimental section, it exhibits excellent performances when compared with other existing approaches. Indeed, as for every data mining task the evaluation has to be experimental in nature, thus we performed several tests in order to assess the validity of our proposal, based on well defined mathematical tools exploited in a novel way, and the results obtained are quite convincing.

2 Problem Statement

In this paper we tackle the problem of outlier detection from large corpus of trajectory data. While for transactional data a tuple is a collection of features, a trajectory is an ordered set (i.e., a sequence) of timestamped points. Trajectory data are usually recorded in a variety of different formats, and they can be drawn from a continuous domain. We assume a standard format for input trajectories, as defined next.

Definition 1 (Trajectory) *Let P and T denote the set of all possible (spatial) positions and all timestamps, respectively. A trajectory is defined as a finite sequence $s_1, \cdots, s_N$, where $N \geq 1$ is the trajectory length and each s_i is a pair (p_i, t_i) where $p_i \in P$ and $t_i \in T$.*

We assume that P and T are discrete domains. In the followingn sections, we will take into account only the spatial information for designing our anomaly detection strategy. This is not a limitation since we are interested in this work in detecting relevant spatial information about trajectory shapes.

We will exploit in this paper a distance based outlier function to catch anomalies in trajectory data. More formally, given a set of trajectories S, a positive integer k, and a positive real number R, a trajectory $t \in S$ is a $DB(k, R)$-outlier, or a distance-based outlier with respect to parameters k and R, if less than k objects in S lie within distance R from t.

Trajectories lying at distance not greater than R from t are called neighbors of t . Each trajectory t is considered a neighbor of itself. This definition assumes that a distance function relating each pair of dataset objects is available. In [9] has been shown that the distance-based outlier definition generalizes the notion of outlier provided by several discordance tests developed in statistics. Furthermore, this definition is suitable for multivariate data and can also be applied even if the distribution of the data is unknown (as in our trajectory data sets). A suitable distance function based on Fourier analysis will be exploited in our setting in order to search for outlying trajectories based on their features.

2.1 Data Pre-processing

Trajectory data are usually two dimensional, as mentioned above when a high accuracy is required we cannot disregard any point. Therefore, there is a need for a multi-resolution analysis to take into account both the spatial dimensions in the pre-processing step. We first give some preliminary definition on multi-resolution analysis.

Definition 2 *Let $\{S_m\}$ be a set of subspace of $L^2(\mathcal{R})$ and $m \in \mathcal{Z}$ such that the following hold: 1) $S_m \in S_{m+1}$; 2) $\bigcup_{m \in \mathcal{Z}} S_m = L^2(\mathcal{R})$; 3) $\bigcap_{m \in \mathcal{Z}} S_m = \emptyset$; 4) $x(t) \in S_m \Leftrightarrow x(\frac{t}{2}) \in S_{m-1}$ then $\{S_m\}$ is a multi-resolution system.*

R and Z have the usual meaning of real and integer number domains, while t is the timestamp of a given point. The choice of $\{S_m\}$ defines the analysis being performed in particular if we choose orthogonal subspace we have *orthogonal multi-resolution* analysis. We exploit here L space since it has a proper norm.

Trajectory data multi-resolution analysis aims at representing each trajectory being analyzed (and then elaborated using a math transform) using a reliable set of coefficient. In order to perform this step allowing a *perfect* (lossless) reconstruction of trajectories we adopt the so called *Lifting Scheme* approach.

2.2 Lifting Schemes

An effective approach for multi-resolution analysis is the *lifting scheme*. It was originally introduced for filtering signals due to its intuitive features, moreover it allows to ex-

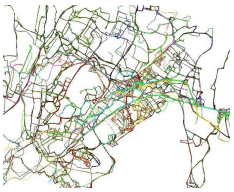

Figure 1. A set of trajectories

actly reconstruct the original input sequences. In this respect, it has been widely used as a support for image compression in wavelet based systems. Moreover it is well suited as a preprocessing step for non separable transforms. In order to perform the proper lifting we need to define a filtering function. A filtering function $\mathcal{F}$, is a function that transforms an input sequence I into an output sequence O according to an optimization function. Most widely used filters are Least Mean Square, Regression or Kalman filtering. In our implementation we exploited the Least Mean Square filter since it works by minimizing the least mean squares of the error signal, i.e. the difference between the computed and the actual signal. This choice offered best performances in our experimental setting.

Performing lifting steps. Given a filtering function $\mathcal{F}$ and a input trajectory Tr_x, the trajectory can be split in two subsequences Tr_{x_o} and Tr_{x_e} that are respectively the sequence of odd and even indices. The trajectory lifting is performed by iteratively updating the subsequences with their predicted version in order to obtain two shorter sequences that are representative of the *original* sequence (say it Tr'_x) and the *trend* sequence (say it Tr_h). Obviously, when performing this step some errors could arise (say it Tr_e). More formally, let $\mathcal{P}$ and $\mathcal{U}$ two filtering function exploited for prediction and update, a generic lifting step works as follows:

- $Tr_e(k) = Tr_{x_o}(k) - \mathcal{P}(Tr_{x'}(k))$;
- $Tr_h(k) = Tr_{x_e}(k) - \mathcal{U}(Tr_{x_e}(k))$.

By iterating the above steps, we obtain a succinct representation of the original trajectory with no loss of information. The proposed lifting scheme will be used for implementing the non separable transform based outlier detection algorithm described in next sections, since when the trajectory size becomes unpractical we will reduce it by a lifting step. To better clarify the proposed pre-elaboration steps we provide a toy example.

Example 1 *Consider the trajectories depicted in Figure 1 regarding buses movements in the Athens metropolitan area.*

Figure 2. A sample trajectory

We focus now on a sample trajectory reported in Figure 2 and zoomed for the sake of clarity.

Applying the lifting scheme defined above will produce the sequence in Figure 3 where the solid (blue) stars represent the trend *sequence, while the empty (red) stars represent the error sequence. As it is easy to see the number of points taken into account after lifting is smaller than the original trajectory and this feature is particularly useful when considering (real life) longer trajectories having more complex shapes. Moreover, lifted trajectories can be updated as new data points arrive thus making the overall process quite efficient. Furthermore, we can make the lifted trajectories equally sized in order to enhance the algorithm performances.*

Figure 3. Sample trajectory lifted version

3 Exploiting Fourier Transforms for Spatial Quincunx Lattices Outlier Detection

In this section we exploit non separable transforms in order to effectively manage two dimensional trajectories. Non separable transforms allow us to consider the whole trajectory taking into account both dimensions in the computation thus avoiding any approximation due to mono-dimensional transform composition. We will exploit them in a suitable way in order to catch similarity among trajectories. The first step to be performed for any mathematical transform is to define the basis function and the features of the search space where data reside. After a deep investigation of several trajectory datasets (they will be described in the experimental section) we found that the best representation for the trajectory search space is a *Quincunx Lattice*[1]. This peculiar structure, will allow us to represent with suitable detail even trajectories laying close to the edges of the search space. To better understanf this statement consider the quincunx lattice shown in Figure 4, it is clear that no region of the search space will be underevaluated in the analysis steps that will be performend on trajectories laying on this lattice.

It is well known from algebra that there is a connection between polynomial algebra and signal transforms, in par-

[1]It is always possible to find a basis that allow this representation for the search space

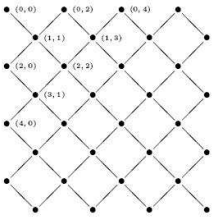

Figure 4. A Quincunx Lattice

ticular we briefly recall that the most important algebraic polynomials are *Chebyshev* polynomials that are defined using a recurrence equation:

$$C_n(x) = 2 \cdot x \cdot C_{n-1}(x) - C_{n-2}(x), n \geq 2$$

whose solution depends on the initial condition (i.e. C_0 and C_1).

Before defining the proper transform we briefly recall the notion of polynomial algebra that naturally induces the notion of transform. An algebra $\mathcal{A}$ is a vector space closed under multiplication and for which the distributivity law holds. As an example consider the set of polynomials $\mathcal{A} = \mathcal{C}[x]$ in one variable. Let $p(x)$ a given polynomial s.t. $p(x) \in \mathcal{C}[x]$, whose degree is $deg(p)$. The set of polynomials of degree less than $deg(p)$ allowing addition and multiplication modulo *p* induce the following algebra:

$$\mathcal{A} = \mathcal{C}[x]/p(x) = \{q(x)|deg(q) < deg(p)\}$$

$\mathcal{A}$ is called a polynomial algebra. $\mathcal{A}$ is a vector space, and its dimension $dim(\mathcal{A}) = deg(p)$. Further, $\mathcal{A}$ is obviously generated by *x*, since every element of $\mathcal{A}$ is a polynomial in *x*. Similarly to the one variable case, we can define polynomial algebras in more variables. For our case (two variables) we define a polynomial algebra as[2]:

$$\mathcal{A} = \mathcal{C}[x]/(p(x,y), q(x,y))$$

Once defined the polynomial algebra the polynomial transform is defined by an isomorphic decomposition of $\mathcal{A}$ w.r.t. a given basis. In particular in our case given the degree of p (say it n) and q (say it m) we can construct a matrix $\mathcal{P}_{b,a} = [p_l(a_k; b_k)]|0 < l, k < m \cdot n$. The matrix so far defined is the Fourier Transform for the given algebra (more details can be found in [6]). If we consider the basis B defined using the *Chebyshev* polynomials (i.e. $B = (T_i(x), T_j(y))|0 < i, j < n$) we obtain the quincunx lattice shown in Fig. 3. After properly constructed the algebra (details are not relevant for this paper, more details in [16]) the spatial signal spectrum is:

- $u_k = \cos(\frac{k+1/2}{n/2} \cdot \pi), 0 < k < n/2$
- $v_l = \cos(\frac{l+1/2}{n/2} \cdot \pi), 0 < l < n/2$
- $w_{k,l,\pm} = \pm\frac{1}{2}\sqrt{(1+u_k) \cdot (1+v_l)}$

Once obtained the spatial signal spectrum we can easily define the distance between two trajectories by considering their spectra. In particular, given two trajectories Tr_1 and Tr_2 and their spectra $Q(Tr_1)$,$Q(Tr_2)$, we compute their angular distances as $d_{\widehat{\alpha\beta}} = \arccos(u_{1k}) - \arccos(u_{2k})$ and $d_{\widehat{\gamma\delta}} = \arccos(v_{1l}) - \arccos(v_{2l})$. For each pair of $w_{k,l,\pm}$ we compute the modulo distance as $d_w = \sqrt{w_{1k,l,\pm}^2 - w_{2k,l,\pm}^2}$. Finally, we define the overall *Quincunx* based distance as:

$$dist_Q(Tr_1, Tr_2) = \sqrt{\sum_{k=1}^{n^2} d_w(k)^2} \cdot \cos(\mu(d_{\widehat{\alpha\beta}}) - \mu(d_{\widehat{\gamma\delta}}))$$

where $\mu(d_{\widehat{\alpha\beta}})$ is the average angle distance between each pair of u_k and $\mu(d_{\widehat{\gamma\delta}})$ is the mean between each pair of v_l. The distance so far defined is able to fully catch dissimilarity between trajectories since it consider the difference in *angular distances* between the two trajectories (the cos argument) while taking into account the *overall extension* of the spatial signal (the modulo part). We point out that the choice to exploit a $Quincunx$ lattice allows us to better cover the whole search space (since it is a directed lattice) and exploiting a non separable transform (i.e. defined on both dimensions not as a simple composition of two monodimensional transforms) we do not introduce any approximation error. Of course also in this case the distance effectiveness rely on the ability to catch the main features of the trajectories. As will be shown in the experimental section we performed several experiments to asses the validity of the approach and the results obtained are quite convincing.

3.1 Outlier Identification

Once defined a technique to state the similarity between two trajectories we need to define a strategy that, exploiting such a technique, identifies anomalies in the data flow.

Definition 3 (Fourier Based Trajectory Outlier) *Given a set of trajectories S, a positive integer k, and a positive real number R, a trajectory $o \in S$ is a $DB(k, R)$-outlier, or a distance-based outlier with respect to parameters k and R, if less than k objects in S lie within distance R from o w.r.t. $dist_Q$.*

[2]Note that here we need to compute modulo two polynomials to ensure that the dimension of A is finite

The threshold values R and k has to be chosen depending on the scenario being monitored. However, in order to assist decision makers to properly set these parameters, we precompute in our prototype the average value of the neighbor radius and the average value of the number of neighbors for the dataset being analyzed. In this way the user can have a useful insigth on data statistics in order to better set the parameters depending on her needs. Once defined our notion of outlier, we can design an effective method for outlier detection.

Algorithm 1 *Function* Compute Outlier:

***INPUT**: a subset of trajectories $\mathcal{S} = \{T_1, \cdots, T_n\}$, a pair of threshold values R and k, a trajectory T_{new};*
***OUTPUT**: Yes if T_{new} is an outlier no otherwise;*
begin
  $temp = 0$
  For each $T_i \in \mathcal{S}$ ***do***
    $d = computeDistQDistance(T_i, T_{new})$
    if $d > R$
      $temp = temp + 1$
  if $temp > k$ ***return*** Yes
    return No;
end

Function $computeDistQDistance$ evaluates the $distQ$ value between the trajectory being analyzed and the trajectories previously collected. If the computed distance is greater than the threshold distance set by the user we will increase an auxiliary variable $temp$. If $temp$ is greater than the threshold value k the trajectory is marked as outlying.

Proposition 1 *Algorithm 1 works in time* $O(|S|\dot{N}\log(N))$.

The running time can be trivially computed by observing that for each trajectory in the dataset S being analyzed we have to compute the Fourier Transformation and this operation is performed in $O(N\log(N))$ time that is the dominant operation in the Algorithm.

4 Experimental Results

In this section, we present the experiments we performed to assess the effectiveness of the proposed approach in detecting outliers. To this purpose, a collection of tests is performed, and for each test some relevant groups of homogeneous trajectories (*trajectory classes*) are considered. The direct result of each test is a similarity matrix representing the degree of similarity for each pair of trajectories in the data set. The evaluation of the results relies on some *a priori* knowledge about the trajectory classes being used that was obtained by domain experts or available from the datasets providers.

We performed several experiments on a wide variety of real datasets. More in detail we analyzed the following data:1) *School Bus*: it is a dataset consisting of 145 trajectories of 2 school buses collecting (and delivering) students around Athens metropolitan area in Greece for 108 distinct days[3];2) *Trucks*: it is a dataset consisting of 276 trajectories of 50 trucks delivering concrete to several construction places around Athens metropolitan area in Greece for 33 distinct days[4];3) *Animals*: it is a dataset containing the major habitat variables derived for radio-telemetry studies of elk, mule deer, and cattle at the Starkey Experimental Forest and Range in northeastern Oregon[5]; 4) *Hurricanes*: It is a dataset containing data about Atlantic hurricanes since 1851, it includes position in latitude and longitude, maximum sustained winds in knots, and central pressure in millibars [6].

In order to perform a simple quantitative analysis we produce for each test a similarity matrix, aimed at evaluating the resulting *neighbors* similarities (i.e., the average of the values computed for trajectories belonging to the same class), and to compare them with the *outer* similarities (i.e., the similarity computed by considering only trajectories belonging to different classes). To this purpose, values inside the matrix can be aggregated according to the class of membership of the related elements: given a set of trajectories belonging to n prior classes, a similarity matrix S about these trajectories can be summarized by a $n \times n$ matrix CS, where the generic element $CS(i,j)$ represents the average similarity between class i and class j.

$$CS(i,j) = \begin{cases} \frac{\sum_{x,y \in C_i, x \neq y} DIST(x,y)}{|C_i| \times (|C_i|-1)} & \text{iff } i = j \\ \frac{\sum_{x \in C_i, y \in C_j} DIST(x,y)}{|C_i| \times |C_j|} & \text{otherwise} \end{cases}$$

where $DIST(x,y)$ is the chosen distance metric ($KNORR$ metric or the $dist_Q$ that we will refer as $Fourier_{2D}$).

The above definition is significant since we normalize the metric value so eventually we can use different approaches, this will allow us to compare performance in the ideal setting for any approach. In particular as mentioned in previous sections we will compare our approach with the one presented in [10].

The higher are the values on the diagonal of the corresponding CS matrix w.r.t. those outside the diagonal, the higher is the ability of the similarity measure to separate different classes. In the following we report a similarity

[3]available at http://www.rtreeportal.org
[4]available at http://www.rtreeportal.org
[5]http://www.fs.fed.us/pnw/starkey/data/tables/index.shtml
[6]available at http://weather.unisys.com/hurricane/atlantic/

$KNORR$	Bus 1	Bus 2
Bus 1	0.9790	0.8528
Bus 2	0.8528	0.9915

(a)

$Fourier_{2D}$	Bus 1	Bus 2
Bus 1	1	0.6250
Bus 2	0.6250	1

(b)

Figure 5. $KNORR$ and $Fourier_{2D}$ similarity matrices for Bus dataset

$Method$	Bus 1	Bus 2
$KNORR$	5	4
$Fourier_{2D}$	7	7

Figure 6. $KNORR$ and $Fourier_{2D}$ number of detected outliers for Bus dataset

matrix for each dataset being considered, as it will be clear the reported results show that our technique is quite effective for outlier detection. In particular, the similarity matrix will give an intuition about the ability of the approach to catch the neighboring trajectories for each class while the number of outliers detected for each dataset is reported in a separate table. We used for the experiments the following parameter values: as k value the maximum number of trajectories supposed to belong to each class and as R value the spatial extension of each class of trajectories, both values are obtained using the information given by the dataset providers.

Measuring Effectiveness for School Bus. For this dataset our prior knowledge is the set of trajectories related to the two school buses. As it is easy to see in Figure 5(a) and (b) $Fourier_{2D}$ outperforms $KNORR$ by allowing a perfect assignment of the proper neighboring class to each trajectory.

The presence of several turns made by the buses makes the feature of $Fourier_{2D}$ well suited for this dataset since it takes into account all the angular values of the trajectory. In Figure 6 the number of detected outlier is reported. The actual number of outliers for each class is 7 so as it is easy to see $Fourier_{2D}$ exactly detected all the outliers in the dataset, such a result is quite understandable considering that the similarity matrix for $Fourier_{2D}$ exactly recognized the neighboring trajectories.

Measuring Effectiveness for Trucks. In this case we considered as class assignment the different trajectories reaching the area where concrete were delivered. In this case there were 6 main classes as it is shown in Figure 7. For this dataset $Fourier_{2D}$ method still outperforms $KNORR$.

This result is not so surprising, if the skewed nature of this set of trajectories is taken into account. Notwithstanding, the similarity matrices exhibit a neat separation from the other classes, and show homogeneous subgroups inside the class. This gives evidence of the effectiveness of the approach in detecting both inter-class and intra-class dis-

$KNORR$	Site 1	Site 2	Site 3	Site 4	Site 5	Site 6
Site 1	0.9700	0.7708	0.7680	0.8625	0.8170	0.7775
Site 2	0.7708	0.9850	0.7300	0.7925	0.7255	0.7767
Site 3	0.7680	0.7300	0.9800	0.6555	0.7915	0.7125
Site 4	0.8625	0.8925	0.8555	0.9740	0.8500	0.8515
Site 5	0.8170	0.7255	0.7915	0.7500	0.9750	0.8195
Site 6	0.7775	0.7767	0.7125	0.7515	0.8195	0.9805

$Fourier_{2D}$	Site 1	Site 2	Site 3	Site 4	Site 5	Site 6
Site 1	0.9991	0.6608	0.7210	0.6125	0.5245	0.7175
Site 2	0.6608	0.9923	0.6135	0.7594	0.5250	0.5868
Site 3	0.7210	0.6135	0.9908	0.6155	0.5896	0.6321
Site 4	0.6125	0.7594	0.6155	1	0.7444	0.7345
Site 5	0.5245	0.5250	0.5896	0.7444	0.9983	0.7913
Site 6	0.7175	0.5868	0.6321	0.7345	0.7913	0.9994

Figure 7. $KNORR$ and $Fourier_{2D}$ similarity matrices for $Trucks$ dataset

$Method$	Site 1	Site 2	Site 3	Site 4	Site 5	Site 6
$KNORR$	2	3	3	4	3	4
$Fourier_{2D}$	4	4	5	4	4	5

Figure 8. $KNORR$ and $Fourier_{2D}$ number of detected outliers for for $Trucks$ dataset

similarity. In Figure 8 we report the number of detected outliers, the actual number of outliers is 5 for each class and $Fourier_{2D}$ outperforms $KNORR$ also w.r.t. the number of detected outliers.

Measuring Effectiveness for Animals. In this case we considered as class assignment the different trajectories traversed by elks, mule deers, and cattles. We point out that it is worth studying animal data because the trajectories are in unrestricted space rather than on well known road network. In this case there were 3 main classes as it is shown in Figure 9. Also in this case $Fourier_{2D}$ outperforms $KNORR$. As we can see, differences among the various classes are marked with higher precision by $Fourier_{2D}$. This is mainly due to the fact that our approach is quite discriminative since it takes into account both angular and modulo distances in the spectrum of a trajectory.

For this dataset the number of actual outliers was 8 for each class, as it is easy to see in Figure 10 $Fourier_{2D}$ still outperforms $KNORR$ for this dataset.

Measuring Effectiveness for Hurricanes. In this case we considered as class assignment the different trajectories traversed by 4 hurricanes (named h_1, h_2, h_3, h_4). We point out that it is important studying hurricanes since early alert in this case will result in human life saving. In this case

$KNORR$	elk	mule deer	cattle
elk	0.9986	0.7759	0.7055
mule deer	0.7759	0.9889	0.7566
cattle	0.7055	0.7566	0.9920

$Fourier_{2D}$	elk	mule deer	cattle
elk	0.9885	0.7439	0.7108
mule deer	0.7439	0.9899	0.7223
cattle	0.7108	0.7223	0.9874

Figure 9. $KNORR$ and $Fourier_{2D}$ similarity matrices for $Animals$ dataset

$Method$	elk	mule deer	cattle
$KNORR$	6	4	5
$Fourier_{2D}$	7	8	8

Figure 10. $KNORR$ and $Fourier_{2D}$ number of detected outliers for $Animals$ dataset

$KNORR$	h_1	h_2	h_3	h_4
h_1	0.9661	0.8559	0.8235	0.8441
h_2	0.8559	0.9672	0.7453	0.7221
h_3	0.8235	0.7453	0.9811	0.8120
h_4	0.8441	0.7221	0.8120	0.9834

$Fourier_{2D}$	h_1	h_2	h_3	h_4
h_1	1	0.6112	0.6235	0.6554
h_2	0.6112	1	0.6133	0.6441
h_3	0.6235	0.6133	1	0.5993
h_4	0.6554	0.6441	0.5993	1

Figure 11. $KNORR$ and $Fourier_{2D}$ similarity matrices for $Hurricanes$ dataset

there were 4 main classes as it is shown in Figure 11. Also in this case $Fourier_{2D}$ outperforms $KNORR$. As we can see, differences among the various classes are again marked with higher precision by $Fourier_{2D}$. In this case a crucial role is played by angular distances in the spectrum of a trajectory.

For this dataset since the similarity results allow a perfect identification of neighboring trajectories for each class, the number of detected outliers for $Fourier_{2D}$ is equal to the actual number of outliers (10). The overall results are reported in Figure 12.

5 Conclusion

In this paper we addressed the problem of detecting outliers in trajectory data. The technique we have proposed is mainly based on the idea of representing a trajectory with its lifted version. Thereby, the similarity between two trajectories can be computed by analyzing their Fourier transforms in the two-dimensional case thus defining a distance measure that can be exploited to define distance based outliers. Experimental results showed the effectiveness of the approach in detecting outlying trajectories.

References

[1] C.C. Aggarwal and P. Yu. Outlier detection for high dimensional data. In *SIGMOD01*, 2001.

[2] F. Angiulli and F. Fassetti. Dolphin: An efficient algorithm for mining distance-based outliers in very large datasets. *TKDD*, 3(1), 2009.

$Method$	h_1	h_2	h_3	h_4
$KNORR$	8	9	7	9
$Fourier_{2D}$	10	10	10	10

Figure 12. $KNORR$ and $Fourier_{2D}$ number of detected outliers for $Hurricanes$ dataset

[3] A. Arning, C. Aggarwal, and P. Raghavan. A linear method for deviation detection in large databases. In *KDD'96*, page 164169, 1996.

[4] V. Barnett and T. Lewis. *Outliers in Statistical Data*. John Wiley & Sons, 1994.

[5] M.M. Breunig, H. Kriegel, R. Ng, and J. Sander. Lof: Identifying density-based local outliers. In *SIGMOD00*, 2000.

[6] T. S. Chihara. *An Introduction to Orthogonal Polynomials*. Gordon and Breach, 1978.

[7] F. Giannotti, M. Nanni, F. Pinelli, and D. Pedreschi. Trajectory pattern mining. In *KDD'07*, pages 330–339, 2007.

[8] D. Hawkins. *Identification of Outliers. Monographs on Applied Probability and Statistics.* Chapman & Hall, 1980.

[9] E. Knorr and R.T. Ng. Algorithms for mining distance-based outliers in large datasets. In *VLDB'98*, page 392403, 1996.

[10] E. Knorr, R.T. Ng, and V. Tucakov. Distance-based outliers: Algorithms and applications. In *VLDB Journal, vol.8, no.3*, pages 237–253, 2000.

[11] J.G. Lee, J. Han, and X. Li. Trajectory outlier detection: A partition-and-detect framework. In *ICDE'08*, pages 140–149, 2008.

[12] J.G. Lee, J. Han, X. Li, and H. Gonzalez. *TraClass*: trajectory classification using hierarchical region-based and trajectory-based clustering. *PVLDB*, 1(1), 2008.

[13] Yunhao Liu, Lei Chen, Jian Pei, Qiuxia Chen, and Yiyang Zhao. Mining frequent trajectory patterns for activity monitoring using radio frequency tag arrays. In *PerCom*, pages 37–46, 2007.

[14] S. Papadimitriou, H. Kitagawa, P. Gibbons, and C. Faloutsos. Loci: Fast outlier detection using the local correlation integral. In *ICDE'03*, page 315326, 2003.

[15] W.H. Press et al. *Numerical Recipes in C++*. Cambridge University Press, 2001.

[16] M. Puschel and M. Rotteler. Fourier transform for the directed quincunx lattice. In *ICASSP*, 2005.

[17] Taubman D. Secker, A. Lifting-based invertible motion adaptive transform (limat) framework for highly scalable video compression. *IEEE Transactions on Image Processing*, 12(12):1530 – 1542, 2003.