UNIVERSIDADE FEDERAL DE VIÇOSA
CENTRO DE CIÊNCIAS EXATAS E TECNOLÓGICAS
DEPARTAMENTO DE INFORMÁTICA

RESEARCH PROJECT

A TRAJECTORY DATA PREPROCESSING FRAMEWORK FOR TRAJECTORIES STORAGE AND ANALYSIS

Douglas Alves Peixoto
MSc Student in Computer Science

Jugurta Lisboa Filho
(Advisor)

Fabio Ribeiro Cerqueira
(Co-Advisor)

VIÇOSA - MINAS GERAIS
NOVEMBER - 2014

Sumário

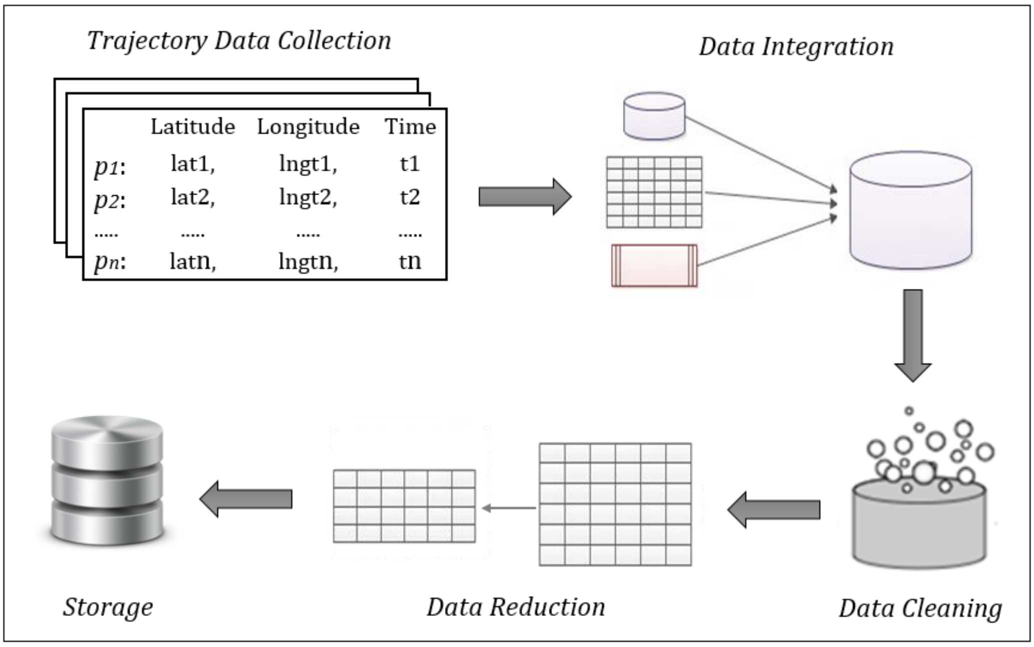
1 INTRODUCTION

The study of GPS trajectory data has become an important field of study in computer science due to the increasing volume of spatio-temporal data obtained from mobiles and GPS devices, which provide options for users to store their mobility log. This the massive amount of data to be analyzed has lead researchers to develop computational tools and data mining techniques combined with machine learning algorithms to enable a better management and understanding of mobile objects [19].

The importance of the data mining studies for the modern society is to help individuals and companies to extract useful information from big data sets [21][32], which is manually impracticable; hence, these techniques can be likewise applied to analyze spatial data in a large scale. One example of an application of trajectory data analysis is to understand human behavior and movement patterns in urban areas.

Although many efforts have been done to analyze trajectories, not much have been done to pre-process these huge amount of data [2][22]. Data preprocessing aims to prepare the data beforehand, mainly by means of statistical and data mining techniques, for its further analysis; such preprocessing techniques may include detection and removal of noise, data reduction and data integration. Raw spatial data are subjected to inconsistencies and noise due to GPS imprecision or device failure, poor data modeling, human mistake, and database problems among others. Even though there are many sources of trajectory data available, e.g. [25][44][46], these data are generally sparse and usually do not combine due to schema mismatch. Therefore, data preprocessing plays an important role into data analysis and storage, once it aims to provide a trustful, integrated, clean and quality data to be stored and accurately analyzed.

In the scope of this project we are going to carry out an study of data preprocessing techniques, in special data integration, data cleaning and data reduction, and apply it on real trajectory datasets. The goal is to propose a trustful framework for trajectory data preprocessing aiming its further storage without loss of information, that is, after the preprocessing we must make sure that the semantic of the data is kept. Figure 1 illustrates the steps of the proposed framework, which will be further described in this work.



	Latitude	Longitude	Time
p1:	lat1,	lngt1,	t1
p2:	lat2,	lngt2,	t2
.....	.....	.....	.....
pn:	latn,	lngtn,	tn

Figura 1: Framework steps overview.

The remainder of this paper is organized as follows. In Section 2 we discuss the related work. Section 3 describes the methodology to build up our framework, addressing every step for trajectory data preprocessing proposed in this project, while Section 4 presents the steps to be followed for the experimental validation of the proposed framework. Finally, we conclude presenting the timeline and the planned budget for this research project in Sections 5 and 6 respectively.

2 RELATED WORK

Despite the considerable amount of work proposed to analyze GPS trajectories, e.g. [12][19][44][45][47], and to store and share huge amount of spatial data, such as Spatial Data Warehouses (SDW) [4], Spatial Data

Infrastructures (SDI) [18][31] and Geographic DBMS (e.g. Oracle Spatial and PostGIS), to our best knowledge not many efforts have been done specifically to address the problem of preprocessing trajectory data, in particular focused on trajectories storage.

Alvares et al. [2], for instance, present a framework for trajectory data preprocessing, their goal is to provide a framework for applications that aim the semantic analysis and mining of trajectory data in a high level of abstraction. In [1] the authors proposed a model for enriching trajectories with geographic semantic information. According with with Alvares et al. [1], devices for collection of trajectories of moving object do not aim to collect the geographic semantic information. Based on this statement, the authors introduce a framework for preprocessing trajectories aiming to add meaningful characteristics on them, based on the trajectory speed and places that the trajectory intersect. Although it is not in the scope of our preprocessing framework to neither add nor remove semantic information from trajectories, no doubt whatsoever the Alvares et al. [1][2] approaches can be used as an addition to our framework to enrich trajectories with semantic, before a trajectory data mining or analysis in a high level of abstraction.

In another work Idrissov and Nascimento [22] present a framework for cleaning trajectories focused on trajectory data clustering. The framework is divided into three steps: stop detection, missing segment interpolation and inaccuracy removal. Although the framework demonstrates an improvement on trajectory data quality for clustering, it uses some approaches that may affect the semantic of the trajectory, such as the stop detection step, which aims to detect stops on the trajectory and remove them in order to improve the accuracy of clustering algorithms. However, some methods for trajectory data mining, such as Zheng et al. [47] uses stops to detect points of interest in cities. Bearing it in mind, we propose a framework for preprocessing trajectories without semantic loss.

Additional work in the literature have been found to address on-line trajectory preprocessing and analysis [16][24][30], which aim to deal with data on-the-fly, that is, while the collector devices, such as Radio-Frequency

IDentification (RFID) [16] and Stream Sensors [30], are still receiving the data. Our proposed framework is to deal with off-line preprocessing, once we are going to carry out our methodology on datasets from which the data have already been collected. Methodologies to deal with on-line data preprocessing and analysis differ from ours, for they aim to make predictions on data on-the-fly, so that they can detect anomalies and outliers on the data while it is being collected by means of statistical inference.

A few other works have focused on trajectory outlier detection [13][17] [20][27]. However, these works aim on identify trajectories that deviates from a standard path or trajectory dataset pattern, not on preprocessing neither on identifying outliers in the trajectory itself.

To sum up, in this work we intent to study the WEKA toolkit, which is a free Java library for machine learning and data mining. This toolkit have been extended to deal with geographic data mining, including some preprocessing techniques [6].

3 METHODOLOGY

As previously stated, the objective of this research is to provide a framework for preprocessing GPS trajectory data without loss of semantic. Provide clean, integrated, reduced and quality data for trajectories storage and analysis.

In this section we describe the methodology that shall be adopted to achieve the goals of this work. This section is divided into five parts: at first, we briefly introduce the datasets which will be used in this project; secondly, we propose a conceptual model and rules to integrate trajectories into a single schema; next, we describe the statistical and other techniques used upon the data to address the problem of trajectory data cleaning; subsequently, we introduce some data reduction techniques, explaining how they can be handled to simplify a trajectory, reducing its size and still keeping its semantic; finally, we describe the methodology that will be used to validate our approach.

3.1 Data Cleaning: Datasets

In this project we are going to work with real trajectory datasets instead synthetic data to carry out our experiments, for they better represent the reality (e.g. real-world traffic flows and traffic jams) and are more prone to noise. Trajectories of moving objects can be in most of cases represented as an ordered set of vertexes $p_i = (x_i, y_i, t_i)$ inputs. In other words, a trajectory τ with n coordinate points is given by $\tau = \{p_1, p_2, \ldots, p_n\} = \{(x_1, y_1, t_1), (x_2, y_2, t_2), \ldots, (x_n, y_n, t_n)\}$, where (x_i, y_i) are the coordinates of the i_{th} trajectory vertex p_i (e.g. latitude and longitude), and t_i is the coordinate time-stamp. The datasets which will be used in this work are presented as follows:

3.1.1 GeoLife Dataset

In this section, we briefly introduce the Microsoft's GeoLife networking service, Zheng et al. [46]. The GeoLife dataset recorded a broad range of users' outdoor movements, including not only life routines like go home and go to work but also some entertainments and sports activities. It is a very popular trajectory data source due to its variety and flexibility of data.

The GeoLife dataset contains GPS trajectories collected by Microsoft Research Asia among 182 individuals distributed in over 30 cities of China and in some cities located in the USA and Europe in a collaborative manner [29]. Information about this dataset is shown in Table 1. The data was collected in a period over five years (April 2007 to August 2012) and were recorded by different devices such as GPS loggers and GPS-phones. The trajectories in this dataset are composed by a sequence of raw GPS coordinates with latitude, longitude and time-stamp. The majority of these trajectories were recorded in a dense time and distance interval, i.e. every $1 \sim 5$ seconds or every $5 \sim 10$ meters per coordinate.

GeoLife Dataset	
Number of Users	182
Number of Trajectories	18.670
Number of Coordinates	24.876.978
Total Distance	1.292.951 km
Total Duration	50.176 hours
Effective Days	11.129

Tabela 1: GeoLife Dataset Info.

3.1.2 T-Drive Dataset

The T-Drive dataset contains GPS trajectories of of over 33,000 taxis during a 3 months period of 2008 within the city of Beijing. The total number of points in this dataset is about 15 million and the total distance of the trajectories reaches to 9 million kilometers. The average sampling interval is about 177 seconds with a distance of about 623 meters. This dataset have been efficiently used to mine fast routes to a given destination based on the experience of taxi drivers [43][44].

3.2 Data Integration

From collecting trajectory data from different sources, problems may occur when merging these data into a single dataset, due to modeling divergences on each source schema. The analysis of the data should be carried out in a single coherent dataset for the sake of efficiency and simplicity, but each dataset may have been modeled into a different schema, and their variables and tuples shall probably not match perfectly into a single schema, even though the data sources are related to a same domain. The same problem aforementioned occurs with spatial data.

In the second part of this project we aim to provide a methodology and a schema to integrate GPS trajectories from different sources, by selecting and keeping into the schema only those fields that are worth to be stored and important for the analysis of the data, such as latitude, longitude and timestamp. The conceptual data schema for trajectory integration proposed in our framework is shown in Figure 2. Spaccapietra et al. [36] present two models for representing trajectories from a conceptual point of view. Although the models also take the semantic representation of trajectories into account, both models focus on their geometric representation, and were mainly designed for Geographic DBMS. In our framework we sole intent to provide a model for integrate trajectories, by keeping those fields which are essential for a trajectory representation and analysis, and are mostly available from all trajectory datasets. The conceptual model of Figure 2 was build using the UML GeoProfile [33], which was designed for modeling of geographic databases using UML diagrams.

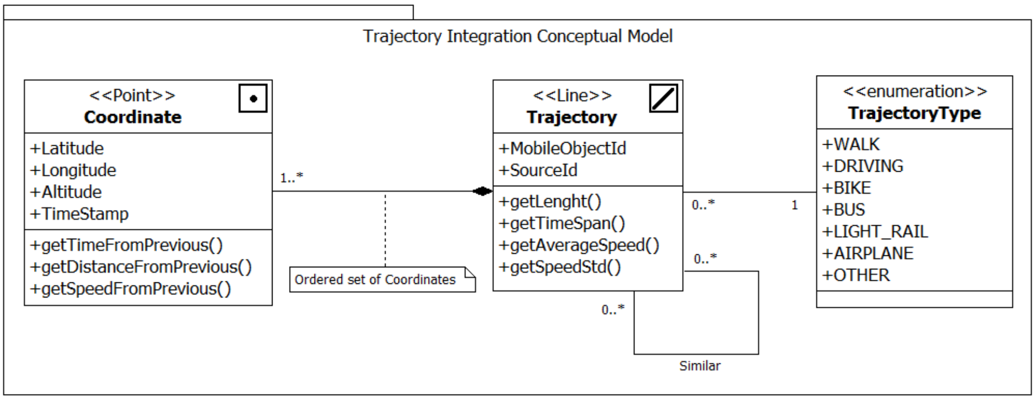


Figura 2: Trajectory Integration Conceptual Model.

During the process of integration some redundancy may be detected (i.e. similar trajectories); correlation analysis should be carried out to detect those redundancies [21]. The problem of detecting similar trajectories play an important role in trajectory storage and analysis; for this reason, a data correlation analysis will be taken into account during the data integration process, which is described later.

The steps to be followed in our integration methodology are briefly described as follows:

1. From all datasets, extract the attributes pursuant to the conceptual model of Figure 2.

2. *Latitude* and *Longitude* attributes will be kept in decimal degrees, or converted if they are not yet.

3. The *Altitude* attribute, when applied, will be kept, or converted, into meters, or will receive a NULL value if it is missing or not valid.

4. *TimeStamp* attribute of each coordinate will be kept, or converted, into the number of days (with fractional part) that have passed since 01/01/1900.

5. We will use GMT (Greenwich Mean Time) in the TimeStamp attribute of all coordinates, as in the GeoLife dataset, to avoid potential confusion of time zone.

6. All trajectories must be associated with a type of trajectory, which are represented in the enumeration of Figure 2. This will be useful for further preprocessing steps and future data analysis. If the type is unknown, or different from those represented in the model, one must set it as "OTHER". More trajectory types can be added to the *TrajectoryType* enumeration according with the data sources one is working with.

7. All trajectory objects in the model contain three attributes named *MobileObjectId* and *SourceId*, which refer to the origin of the trajectory. For instance, in this project all the trajectories from the GeoLife dataset will have the *SourceId* attribute tagged as "GEOLIFE", the attribute *MobileObjectId* is the mobile object identifier whose the trajectory was generated from (can be a person, a vehicle, a device, or something else). The same will be done for all other datasets that will compose the integrated schema.

8. The methods *getLength()*, *getTimeSpan()*, *getAverageSpeed()* and *getSpeedStd()* belonging to the class *Trajectory*, return respectively the length of the trajectory in meters; its time span, that is, the time-stamp of its last coordinate minus the time-stamp of its first coordinate; its average speed and its standard deviation speed.

9. The methods belonging to the class *Coordinate*, namely, *getTimeFromPrevious(), getDistanceFromPrevious()* and *getSpeedFromPrevious()* from Figure 2, return, respectively, for a coordinate p_i, the time taken, the distance, and the speed from the coordinate p_{i-1}, and are for use in the cleaning process, which will be described later.

3.2.1 Check Trajectories Similarity

During the process of integration, some redundancy among trajectories may be detected; similarity analysis should then be carried out to detect those redundancies during the data integration process in order to group similar trajectories. The relationship *Similar* in Figure 2 was created to address this issue.

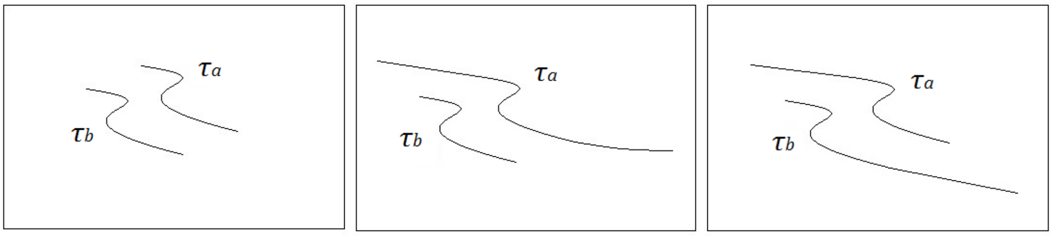
Redundancy sometimes can be important for trajectory analysis; for instance, one may be interested in planning a road network capacity, planning municipal transportation or detect usual road paths in a city to avoid traffic jam [13]. In this sort of analysis, redundant trajectories are important, and should not be removed from the dataset, rather they should be grouped according with some similarity criteria chosen by the user of the framework. On the other hand, depending on the problem one is modeling, and the purpose of the pre-processing, this approach can be used to provide a set of disjoint trajectories instead, by removing redundant trajectories.

The definition of similarity depends on the sort of problem one is interested in; for instance, one may be interested in grouping, or removing, trajectories that are geographically identical, as shown in Figure 3 (a); or group them by location, that is, all trajectories within a given spatial range are similar; one may also be interested in grouping trajectories which move along the same road or direction, for instance, all trajectories that cross the Wall Street in NY are similar, or trajectories that move towards Sydney in Australia are likewise similar. The end user is free to define the similarity criteria that better suits the problem.

One common way to compare time-series objects (i.e. trajectories) is by means of Dynamic Time Warping (DTW) [5], which can be used to evaluate the distance between two objects that have the same pattern but different speeds. Several other efforts have been done to evaluate similarity between trajectories based on curve similarity, e.g. [7][28][34][35] [37][38][39][40].

Although the aim of this framework at this point is to sole provide a methodology to integrate trajectory datasets, it is intended to make a comparative study among methods to detect similar trajectories based on curve similarity. When comparing similarity among curves, three situations may happen, and we must approach them differently, they are shown in Figure 3. As an example, lets take two trajectories, τ_a and τ_b, defined as an ordered set of time-stamped coordinate points as follows:

$$\tau_a = \{p_{a1}, p_{a2}, p_{a3}, ..., p_{an}\}, \quad \tau_b = \{p_{b1}, p_{b2}, p_{b3}, ..., p_{bn}\}$$



(a) (b) (c)

Figura 3: Curve Similarity.

The first scenario, shown in (a), occurs when both trajectories, τ_a and τ_b, are very similar (according with a similarity threshold ε), or identical, that is, $\tau_a = \tau_b$. The second scenario, shown in (b), occurs when one of the trajectories is longer, in one side or both sides, than the other trajectory, that is, $\tau_a \subset \tau_b$ or $\tau_b \subset \tau_a$. The last scenario, shown in (c), occurs when both trajectories are longer than the other one but in different directions. For each pair of similar trajectories, we label its type and set the degree of similarity pursuant to the scenario.

3.3 Data Cleaning

Data in real world tend to be noisy, inconsistent and/or incomplete due to a variety of issues that may happen during the process of collection, modeling and storage of the data, and also due to human mistakes and collection devices error. Data cleaning techniques aim to work on this problems, by filling in missing values, identifying and correcting inconsistencies, and smoothing noise [21]. The main objective of data cleaning is to increase the accuracy of a further analyses of the data; cleaned datasets can also improve data matching. There are a considerable amount of data mining techniques do deal with noisy in a broad range of fields; Chapman [10], for instance, present a survey on techniques for cleaning databases of insect species based on their geographic occurrence; in another work, the author present techniques for data quality in spatial data [9], arguing that quality is merely a factor of fitness for use or potential use, and it is relative to the domain and purpose one is interested in.

The goal of this cleaning process is to provide more trustful, accurate and quality trajectory data to be analyzed and stored without loss on its semantic. Missing and incorrect attribute values contribute for inaccurate trajectory data analysis and a poor data storage, as well as do noisy coordinates. To address the problem of trajectory data cleaning, in our framework we propose an approach for cleaning trajectories in a four step process, namely: (1) remove duplicate coordinates, (2) remove inconsistent values, (3) predict missing values, and (4) identify and ignore noisy coordinates.

3.3.1 Preliminaries

Before we can start, let us introduce some basic terminology. Let I be our integrated set of trajectories as described in Section 3.2, and lets suppose our set I contains k trajectories, that is $|I| = k$. For the sake of brevity, we are going to present our cleaning process for a single trajectory τ with $n \in \mathbb{N}^*$ coordinate points. However, all the steps described here will be performed for all $\tau_i \in I$, where $i = \{1, 2, \ldots, k\}$.

Therefore, given a trajectory $\tau \in I$ with $n \in \mathbb{N}^*$ coordinate points, let $p_i = (x, y, t)$ be the i_{th} coordinate of τ, where x and y denotes the geographic object's position at time t. We define $t(i)$, $d(i)$ and $s(i)$ respectively as the time taken in seconds (s), the distance in meters (m) and the speed in meters per second (m/s) from a coordinate point p_{i-1} to p_i of τ as follows:

$$t(i) = \begin{cases} 0, & \text{if i = 1} \\ p_i.t - p_{i-1}.t, & \text{if i > 1} \end{cases}$$

$$d(i) = \begin{cases} 0, & \text{if i = 1} \\ dist(p_{i-1}, p_i), & \text{if i > 1} \end{cases}$$

$$s(i) = \begin{cases} 0, & \text{if i = 1} \\ d(i)/t(i), & \text{if i > 1} \end{cases}$$

Because we are dealing with spherical geometry, the $dist(p_{i-1}, p_i)$ function we use here, which returns the distance from p_{i-1} to p_i on a sphere (Earth), is the $HaversineDistance$, which calculates a great-circle distance between two points on a sphere from their latitudes and longitudes. For any two points p_i and p_j on a sphere the $HaversineDistance$ is given by:

$$dist(p_i, p_j) = 2rsin^{-1}(\sqrt{sin^2(\frac{x_i - x_j}{2}) + cos(x_i)cos(x_j)sin^2(\frac{y_i - y_j}{2})})$$

where r is the $Earth$ radius, x and y are respectively the latitudes and longitudes of the points p_i and p_j.

Finally, for each trajectory $\tau \in I$ with n coordinate points, we define three sets, namely $T, D, S \subset \mathbb{R}^+$, one for each of the trajectory functions previously described, where $T = \{t_1, t_2, \ldots, t_n\}$, $D = \{d_1, d_2, \ldots, d_n\}$ and $S = \{s_1, s_2, \ldots, s_n\}$, such that for each $i \in \{1, 2, \ldots, n\}$ we have $t_i = t(i)$, $d_i = d(i)$ and $s_i = s(i)$.

3.3.2 Data Transformation

In the last step of our cleaning process, we are going to use a speed-based statistical model; for this purpose, we perform a transformation on the speed set S to an appropriate form for cleaning. We will conduct a Min-Max linear transformation on the original set S, that is, $f_{mm} : S \rightarrow S'$, which aim to map each value of speed $s_i \in S$ to a new value $s'_i \in S'$ in a range $[newMin_S, newMax_S] \subset \mathbb{R}^+$, by computing $\forall s_i \in S$:

$$f_{mm}(s_i) = s'_i = \frac{s_i - min_S}{max_S - min_S}(newMax_S - newMin_S) + newMin_S$$

We chose the Min-Max normalization for it preserves the relationship among the original values [21]. This relationship is important for our cleaning process, in order to keep the semantic of the trajectory. This transformation will neither affect nor persist to the dataset I, once the new feature set S' will be used only during the cleaning process, moreover $S \cup S' \nsubseteq I$.

3.3.3 Data Cleaning Steps

Following we briefly present the four steps proposed for our cleaning framework.

STEP 1: Removing Duplicate Coordinates. Here we must remove duplicate occurrences of coordinates in a same trajectory τ. For instance, we must check if there are more than one coordinate with the same values for the attributes *Latitude*, *Longitude* and *TimeStamp* in a same trajectory; if so, the duplicates must be removed. Coordinates of a trajectory with only *Latitude* and *Longitude* duplicated, but different *TimeStamp* must not be removed, since a trajectory can cross the same point more than once at different times, and a moving object can stay still in a same position for a long time.

STEP 2: Removing Inconsistent Values. Remove inconsistent values that make no sense for the attributes value range. For instance, negative

TimeStamp; *Latitude* out of range from -90 to 90 degrees; *Longitude* out of range from -180 to 180 degrees; *TimeStamp* of a coordinate $p_i \in \tau$ bigger than the *TimeStamp* of a coordinate $p_{i+1} \in \tau$, that is, $p_{i+1}.t < p_i.t$; etc. All these inconsistent values must be removed and the missing values filled with an acceptable one. Step 3 will deal with missing values.

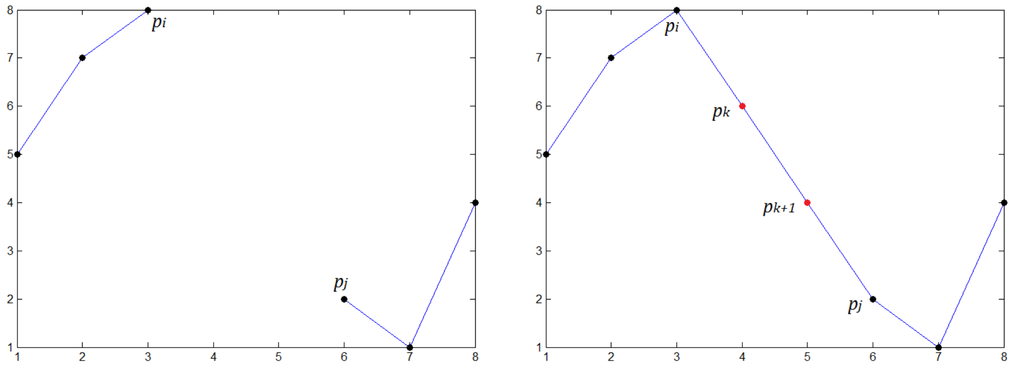
STEP 3: Predicting Missing Values. Missing and NULL values does not always imply that there is error in the data. But for the case of trajectories, we assume that all coordinate $p_i \in \tau$ must have at least the attributes *Latitude*, *Longitude* and *TimeStamp*. The lack of any of these attributes would make no sense for a trajectory schema. *Altitude* can be omitted if it has not been collected. If between two coordinates p_i and p_j of τ there are n coordinates $\{p_{i+1}, \ldots, p_{j-1}\}$ lacking their values for *Latitude*, *Longitude* and/or *TimeStamp*, we predict the missing values by making a local smoothing, by means of interpolation methods. The simplest example of such a problem is to find a linear polynomial whose graph passes through two known distinct points p_i and p_j, where we know beforehand that the missing values are between p_i and p_j, and fit the missing values according with the linear function. One example to predict missing values by means of a linear function using local mean is shown below.

```
FOR k = i+1 to j-1 DO {
        $v(p_k) = v(p_{k-1}) + increment$
}
```

where $v(p_k)$ is the value of the attribute (i.e. *Latitude*, *Longitude*, *TimeStamp* or *Altitude* (if applied)) of the coordinate $p_k \in \tau$, and

$$increment = \frac{mean}{n} = \frac{v(i) + v(j)}{2 * n}$$

One example of linear interpolation for (x, y) attributes of a trajectory is shown in Figure 4, where the points p_k, p_{k+1} and p_{k+2} have been add between points p_i and p_j by estimation of their values for x and y. One drawback of this approach is that if the variables are very sparse, a linear interpolation function would not represent the missing data accurately, showing a high estimation error rate. In this case, it is customary to use a higher degree polynomial interpolation method, such as *Lagrange-Polynomial* which can perform a smoother representation of a curve, or *Ordinary Kriging* for attributes estimation, which take the spatial correlation among the points into account to minimize the estimation error rate. Other estimation methods that can be exploit here are *Triangulation* and *Inverse Distance* [23]. We aim to use those methods in this work, and compare one to another about their efficiency and error rate to approximate missing values for each of the coordinate attributes aforementioned.

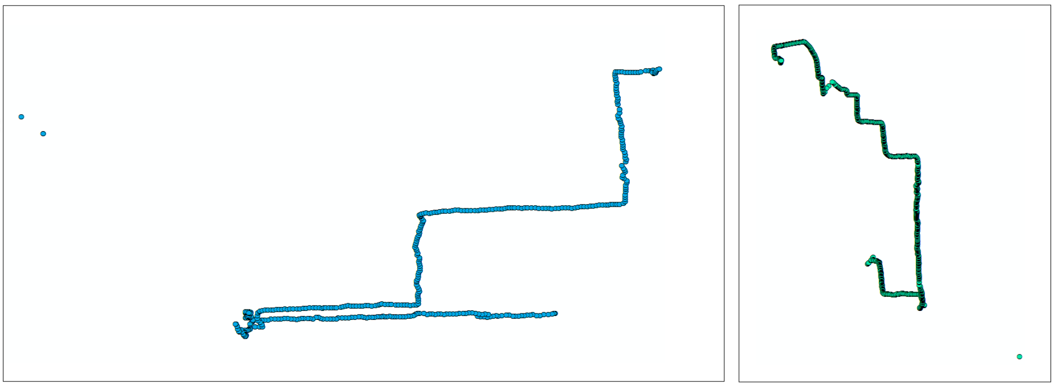


(a) Before Interpolation

(b) After Interpolation

Figura 4: Example of Linear Interpolation

STEP 4: Identifying and Ignoring Noisy Coordinates. In this final step of our cleaning process, we aim to remove noisy coordinates from trajectories. GPS trajectory data are quite susceptible to random noise for many reasons, such as human mistake, data transmission error and system or recording device error, among others. Figure 5 (a) and (b) show two trajectories from the GeoLife dataset with some coordinates that deviates from the trajectory pattern; at one point a moving object suddenly took a very high speed, more than 200km/s, in period of less than 5 seconds, what is quite improbable. To remove this sort of noise, we are going to carry out an speed-based approach along the trajectories.



(a) (b)

Figura 5: Noisy Coordinates

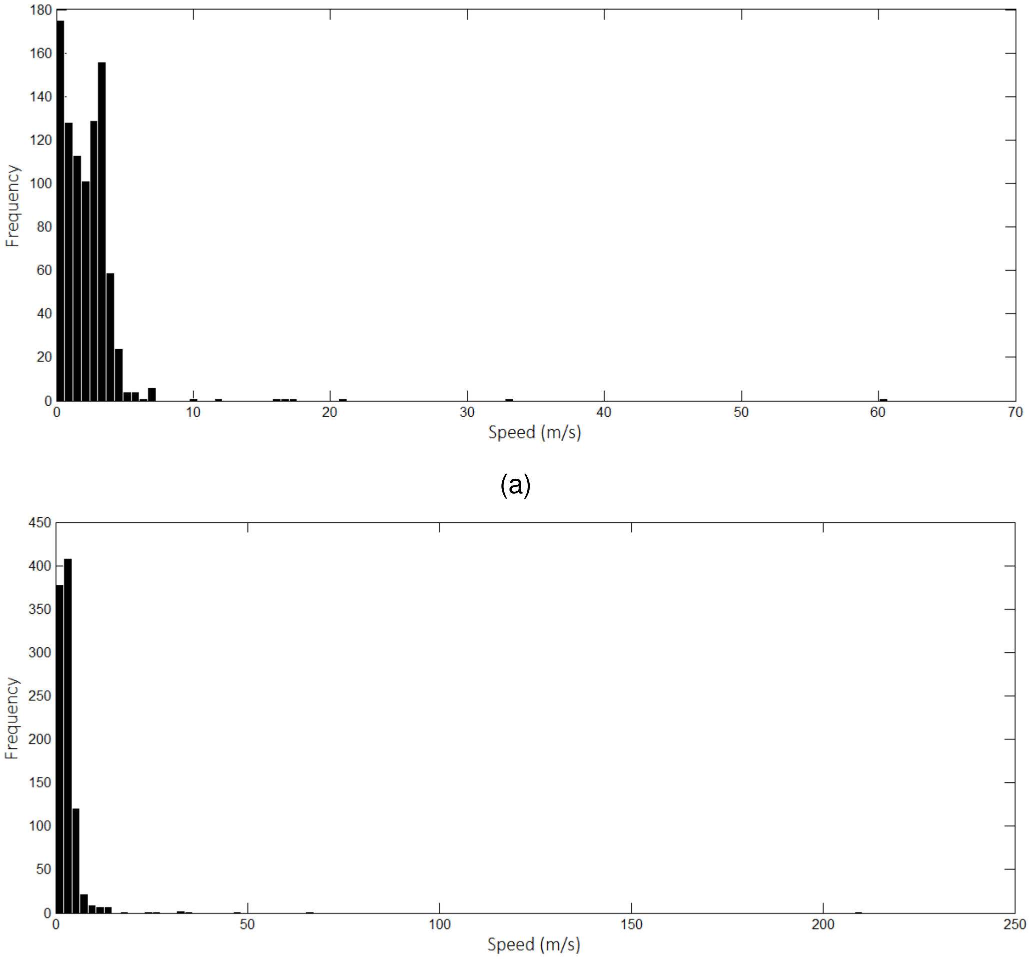
Firstly, we will calculate the speed set S for a trajectory $\tau \in I$, as described in Section 3.3.1. Secondly, for the trajectory speed set S we perform our transformation function $f_{mm} : S \rightarrow S'$ in order to get a normalized speed set S', as described in Section 3.3.2. Finally, from S' we check if in between any two coordinates the speed fluctuates to unacceptable values. To check for unacceptable values for speed, we are going to perform an statistical analysis of the distribution of the speed along the trajectory, that is, an statistical analysis of the elements of S'. Coordinates such that its speed deviates too much from the speed distribution shall be removed.

Figure 6 shows the speed histogram along the trajectories of Figures 5 (a) and (b) respectively. Figure 7 shows the speed distribution (without normalization) for the trajectory of Figures 5 (a) and (b) respectively. One may notice the anomalous fluctuation on the speed along the trajectories.

The goal at this step is to build a model $p(s)$ based on the normalized speed distribution $S' = \{s'_1, s'_2, \ldots, s'_n\}$ of τ and find a threshold ε such that $\forall p_i \in \tau$:

$$p_i = \begin{cases} noise, & \text{if } p(s'_i) < \varepsilon \\ normal, & \text{if } p(s'_i) \geq \varepsilon \end{cases}$$

For our anomaly detection approach, we can also apply another trans-



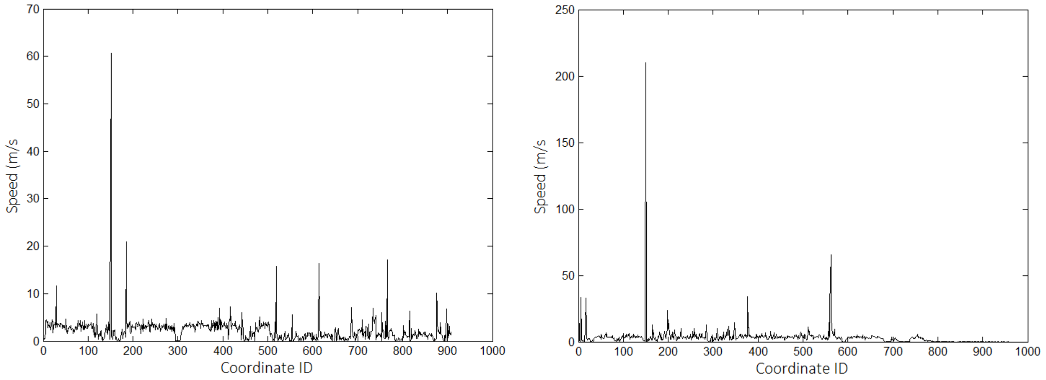
(a)

(b)

Figura 6: Trajectories Speed Histogram

formation on the speed distribution to make it close to a Gaussian distribution, and chose ε based on its standard deviation σ, in other words $\varepsilon = c\sigma$, where $c \in \mathbb{N}^*$ is a constant. An additional approach to address this problem is to use a density-based algorithm for discovering both clusters and outliers on the speed distribution, such as the DBSCAN algorithm Ester et al. [15], that is, cluster points by their speed using the point speed as distance function, so that points with similar speed will be put in a same cluster, whereas points with anomalous speed will be defined as outliers.

Finally, the step 3 of our cleaning process can be applied after the noise



(a)

(b)

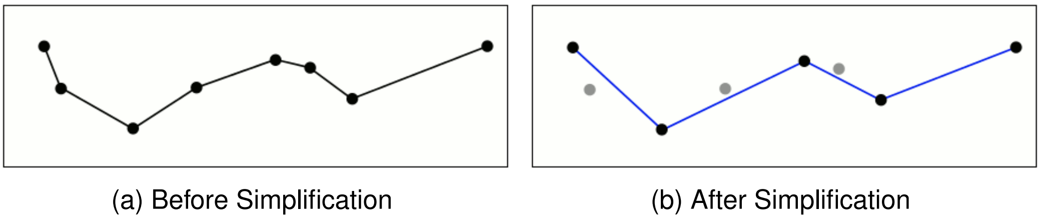
Figura 7: Trajectories Speed Distribution

removal as well, to predict the right position for the noisy coordinates.

3.4 DATA REDUCTION

Trajectories may impose huge storage requirements. To address this problem, the final step of our framework is to carry out a trajectory data reduction focus on line-simplification approaches, such as the Douglas-Peuker heuristic [14]. The goal of line-simplification approaches lies on approximate a trajectory by another one which is similar to the original, but has fewer points. This similarity is based on a predetermined accuracy bound φ, that is, the simplified trajectory deviates from the original one by less than a given accuracy bound φ. The value of φ is also known in the literature as *Haussdorff* distance, which measures how far two subsets in a two-dimensional space are from each other. Another well known algorithm for line-simplification that can be exploit in this project is the min-# algorithm of Chen and Daescu [11]. We aim with this approaches to reduce the trajectories storage consumption with a minimum or none information loss.

Figure 8 shows a curve simplification using the Douglas-Peuker algorithm. This method has been shown to outperform other reduction methods, such as Wavelet Transforms and Log-Linear models [8]. Besides, the integration process of our framework, presented in Section 3.2, is also a data reduction approach, once it perform an attribute (or feature) subset selection method.



(a) Before Simplification

(b) After Simplification

Figura 8: Curve Simplification with Douglas-Peuker Heuristic

4 EXPERIMENTAL VALIDATION

To validate our approach, we are going to carry out a comparative test between both our preprocessed trajectories and the original ones, by using them in a couple of recognized algorithms for trajectory analysis [19][45] [47], and compare their outcomes and performance for both set of trajectories. In addition, we shall compare our results with the methods for spatial preprocessing developed in the WEKA toolkit [6]. We expect that the outcomes of the algorithms show no expressive differences from one trajectory set to another, meaning that the semantic of the data is not affected by our preprocessing approach and there is no significant loss of information. On the other hand, we expect a significant drop in the storage requirements and a substantial increase in the performance of the algorithms aforementioned, once there will be less data to be stored and analyzed; hence, we expect a drop in the algorithms runtime.

5 TIMELINE

Table 2 shows the timeline for completion of major tasks for this research project, pursuant to the regulations from both the *Pró-Reitoria de Pesquisa e Pós Graduação (PPG)* and the *Departamento de Informática (DPI)* to obtain the title of Master of Computer Science at the *Universidade Federal de Viçosa (UFV)*.

2015	Jan	Fev	Mar	Apr	May	Jun	Jul
Finish Literature Review	x						
Submit Short Paper	x						
Data Integration	x	x					
Traj. Similarity Study			x	x			
Traj. Data Cleaning					x	x	x
Submit Short Paper							x
2015	Aug	Sep	Oct	Nov	Dec	-	-
Traj. Data Reduction	x	x					
Experimental Validation			x	x			
Dissertation Written			x	x	x		
Submit Paper Journal					x		
Dissertation Defense					x		

Tabela 2: Research Project Timeline.

6 BUDGET

Budget Specification	**Amount R$**	**Resources**
1 HUMAN RESOURCES		
1.1 Allowance for execution of the research (scholarship value x 12 months)	18.000	CAPES
1.2 Advisor Committee (10% of the salary of the professors committee x 12 months)	-	UFV
Subtotal	**18.000**	
2 BIBLIOGRAPHIC MATERIAL		
2.1 Books, technical journals, etc.	1.000	Own

Subtotal	**1.000**	
3 MATERIAL		
3.1 Reams of paper	100	Own
3.2 Ink cartridges for printer	-	-
3.3 CD and DVD	-	-
Subtotal	**100**	
4 THIRD PARTY SERVICES		
4.1 Typing and formatting	-	-
4.2 Linguistic revision	-	-
4.3 Print/Bookbinding	200	Own
4.4 Post services	100	Own
Subtotal	**300**	
5 ACCOMMODATION		
5.1 Field trip for data collection	-	-
Subtotal	**0**	
6 TECHNICAL RESERVE		
6.1 10% of the previous items		Own
Subtotal	**1.940**	
Total	**21.340**	

Tabela 3: Research Project Expected Budget.

Referências

[1] Alvares, L. O., Bogorny, V., Kuijpers, B., de Macedo, J. A. F., Moelans, B., and Vaisman, A. A model for enriching trajectories with semantic geographical information. In *Proceedings of the 15th Annual ACM International Symposium on Advances in Geographic Information Systems* (New York, NY, USA, 2007), GIS '07, ACM, pp. 22:1–22:8.

[2] Alvares, L. O., Oliveira, G., and Bogorny, V. A framework for trajectory data preprocessing for data mining. *Proceedings of the*

21st International Conference on Software Engineering & Knowledge Engineering (SEKE'2009). Boston, Massachusetts, USA (2009).

[3] ANTON, H. *Elementary Linear Algebra*, 10th ed. John Wiley & Sons, 2010.

[4] BARCLAY, T., GRAY, J., AND SLUTZ, D. Microsoft terraserver: A spatial data warehouse. *SIGMOD Rec. 29*, 2 (May 2000), 307–318.

[5] BERNDT, D. J., AND CLIFFORD, J. Using dynamic time warping to find patterns in time series. In *KDD Workshop* (1994), U. M. Fayyad and R. Uthurusamy, Eds., AAAI Press, pp. 359–370.

[6] BOGORNY, V., PALMA, A. T., ENGEL, P., AND ALVARES, L. O. Weka-gdpm: Integrating classical data mining toolkit to geographic information systems. In *SBBD Workshop on Data Mining Algorithms and Aplications (WAAMD 2006), Florianopolis, Brasil, October* (2006), pp. 16–20.

[7] BUCHIN, K., BUCHIN, M., AND WANG, Y. Exact algorithm for partial curve matching via the fréchet distance. In *Proceedings of 20th ACM-SIAM Symposium on Discrete Algorithms* (2009).

[8] CAO, H., WOLFSON, O., AND TRAJCEVSKI, G. Spatio-temporal data reduction with deterministic error bounds. *The VLDB Journal 15*, 3 (Sept. 2006), 211–228.

[9] CHAPMAN, A., AND FACILITY, G. B. I. *Principles of Data Quality*. Global Biodiversity Information Facility, 2005.

[10] CHAPMAN, A. D. *Principles and Methods of Data Cleaning: Primary Species and Species-occurrence Data*. Global Biodiversity Information Facility, 2005.

[11] CHEN, D. Z., AND DAESCU, O. Space-efficient algorithms for approximating polygonal curves in two-dimensional space. In *Proceeding of the 4th International Computing and Combinatorics Conference* (1998), pp. 55–64.

[12] CHEN, Z., SHEN, H. T., AND ZHOU, X. Discovering popular routes from trajectories. In *Proceedings of the 2011 IEEE 27th International Conference on Data Engineering* (Washington, DC, USA, 2011), ICDE '11, IEEE Computer Society, pp. 900–911.

[13] DE AQUINO, A. R., ALVARES, L. O., RENSO, C., AND BOGORNY, V. Towards semantic trajectory outlier detection. In *GeoInfo* (2013), MCT/INPE, pp. 115–126.

[14] DOUGLAS, D. H., AND PEUCKER, T. K. Algorithms for the reduction of the number of points required to represent a digitized line or its caricature. *Cartographica: The International Journal for Geographic Information and Geovisualization 10*, 2 (1973), 112–122.

[15] ESTER, M., PETER KRIEGEL, H., S, J., AND XU, X. A density-based algorithm for discovering clusters in large spatial databases with noise. AAAI Press, pp. 226–231.

[16] FAZZINGA, B., FLESCA, S., FURFARO, F., AND PARISI, F. Cleaning trajectory data of rfid-monitored objects through conditioning under integrity constraints. In *EDBT* (2014), pp. 379–390.

[17] FONTES, V. C., DE ALENCAR, L. A., RENSO, C., AND BOGORNY, V. Discovering trajectory outliers between regions of interest. In *GeoInfo* (2013), MCT/INPE, pp. 49–60.

[18] GROOT, R., AND MCLAUGHLIN, J. *Geospatial data infrastructure: Concepts, cases and good practice.*, vol. 93. Oxford: Oxford University Press., 2000.

[19] GUDMUNDSSON, J., THOM, A., AND VAHRENHOLD, J. Of motifs and goals: Mining trajectory data. In *Proceedings of the 20th International Conference on Advances in Geographic Information Systems* (New York, NY, USA, 2012), SIGSPATIAL '12, ACM, pp. 129–138.

[20] GUPTA, M., GAO, J., AGGARWAL, C. C., AND HAN, J. Outlier detection for temporal data: A survey, 2014.

[21] HAN, J., KAMBER, M., AND PEI, J. *Data Mining: Concepts and Techniques*, 3rd ed. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2011.

[22] IDRISSOV, A., AND NASCIMENTO, M. A. A trajectory cleaning framework for trajectory clustering. In *Mobile Data Challenge (by Nokia) Workshop* (2012).

[23] ISAAKS, E. H., AND SRIVASTAVA, M. R. *An Introduction to Applied Geostatistics*. Oxford University Press, USA, Jan. 1990.

[24] LANGE, R., DÜRR, F., AND ROTHERMEL, K. Online trajectory data reduction using connection-preserving dead reckoning. In *Proceedings of the 5th Annual International Conference on Mobile and Ubiquitous Systems: Computing, Networking, and Services* (ICST, Brussels, Belgium, Belgium, 2008), Mobiquitous '08, ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering), pp. 52:1–52:10.

[25] LAURILA, J. K., BLOM, J., DOUSSE, O., GATICA-PEREZ, D., BORNET, O., EBERLE, J., AAD, I., AND MIETTINEN, M. The mobile data challenge: Big data for mobile computing research. In *Pervasive Computing* (Newcastle, UK, 2012).

[26] LAY, D. C. *Linear Algebra and Its Applications*, 3rd ed. Pearson Education, Addison-Wesley Publishing Company, 2003.

[27] LEE, J.-G., HAN, J., AND LI, X. Trajectory outlier detection: A partition-and-detect framework. In *Proceedings of the 2008 IEEE 24th International Conference on Data Engineering* (Washington, DC, USA, 2008), ICDE '08, IEEE Computer Society, pp. 140–149.

[28] LIU, H., AND SCHNEIDER, M. Similarity measurement of moving object trajectories. In *Proceedings of the 3rd ACM SIGSPATIAL International Workshop on GeoStreaming* (New York, NY, USA, 2012), IWGS '12, ACM, pp. 19–22.

[29] MICROSOFTCORPORATION. Geolife - user guide. Version 1.3 (2012/08/01).

[30] PUMPICHET, S., PISSINOU, N., JIN, X., AND PAN, D. Belief-based cleaning in trajectory sensor streams. In *ICC* (2012), IEEE, pp. 208–212.

[31] RAJABIFARD, A., AND WILLIAMSON, I. P. Spatial data infrastructures: Concept, sdi hierarchy and future directions. In *Suitability of Internet Technologies for Access, Transmission and Updating Digital Cadastral Databases on the Web. Proceedings of AURISA 97* (1999).

[32] RAJARAMAN, A., AND ULLMAN, J. D. *Mining of massive datasets*. Cambridge University Press., Cambridge, UK, 2012.

[33] SAMPAIO, G. B., NALON, F. R., AND FILHO, J. L. Geoprofile - uml profile for conceptual modeling of geographic databases. In *ICEIS International Conference On Enterprise Iformation Systems (3)* (2010), J. Filipe and J. Cordeiro, Eds., SciTePress, pp. 409–412.

[34] SANKARARAMAN, S., AGARWAL, P. K., MOLHAVE, T., AND BOEDIHARDJO, A. P. Computing similarity between a pair of trajectories. *CoRR 1303.1585* (2013).

[35] SIIRTOLA, P., LAURINEN, P., AND RÖNING, J. A weighted distance measure for calculating the similarity of sparsely distributed trajectories. In *ICMLA* (2008), M. A. Wani, X. wen Chen, D. Casasent, L. A. Kurgan, T. Hu, and K. Hafeez, Eds., IEEE Computer Society, pp. 802–807.

[36] SPACCAPIETRA, S., PARENT, C., DAMIANI, M. L., DE MACEDO, J. A., PORTO, F., AND VANGENOT, C. A conceptual view on trajectories. *Data Knowledge Engineering 65*, 1 (Apr. 2008), 126–146.

[37] TIAKAS, E., PAPADOPOULOS, A. N., NANOPOULOS, A., MANOLOPOULOS, Y., STOJANOVIC, D., AND DJORDJEVIC-KAJAN, S. Sear-

ching for similar trajectories in spatial networks. *The Journal of Systems and Software 82*, 5 (2009), 772–788.

[38] VAN KREVELD, M., AND LUO, J. The definition and computation of trajectory and subtrajectory similarity. In *Proceedings of the 15th Annual ACM International Symposium on Advances in Geographic Information Systems* (2007), GIS '07, pp. 44:1–44:4.

[39] VLACHOS, M., GUNOPULOS, D., AND KOLLIOS, G. Discovering similar multidimensional trajectories. In *ICDE* (2002), R. Agrawal and K. R. Dittrich, Eds., IEEE Computer Society, pp. 673–684.

[40] VLACHOS, M., GUNOPULOS, D., AND KOLLIOS, G. Robust similarity measures for mobile object trajectories. In *DEXA Workshops* (2002), IEEE Computer Society, pp. 721–728.

[41] WACKERLY, D., MENDENHALL, W., AND SCHEAFFER, R. *Mathematical Statistics with Applications*, 7th ed. Cengage Learning, 2007.

[42] WONG, D. W.-S., AND LEE, J. *Statistical Analysis of Geographic Information with ArcView GIS and ArcGIS*. John Wiley & Sons Hoboken, NJ, USA, 2005.

[43] YUAN, J., ZHENG, Y., XIE, X., AND SUN, G. Driving with knowledge from the physical world. In *Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (New York, NY, USA, 2011), KDD '11, ACM, pp. 316–324.

[44] YUAN, J., ZHENG, Y., ZHANG, C., XIE, W., XIE, X., SUN, G., AND HUANG, Y. T-drive: Driving directions based on taxi trajectories. In *Proceedings of the 18th SIGSPATIAL International Conference on Advances in Geographic Information Systems* (New York, NY, USA, 2010), GIS '10, ACM, pp. 99–108.

[45] ZHENG, Y., LI, Q., CHEN, Y., XIE, X., AND MA, W.-Y. Understanding mobility based on gps data. In *Proceedings of the 10th international conference on Ubiquitous computing* (2008), ACM, pp. 312–321.

[46] ZHENG, Y., XIE, X., AND MA, W.-Y. Geolife: A collaborative social networking service among user, location and trajectory. *IEEE Data Engineering Bulletin*, 2, 32–39.

[47] ZHENG, Y., ZHANG, L., XIE, X., AND MA, W.-Y. Mining interesting locations and travel sequences from gps trajectories. In *Proceedings of the 18th international conference on World wide web* (2009), ACM, pp. 791–800.