Mining Trajectory Data for Discovering Communities of Moving Objects

Corrado Loglisci
Department of Computer Science
University of Bari "Aldo Moro"
Bari, Italy
corrado.loglisci@uniba.it

Donato Malerba
Department of Computer Science
University of Bari "Aldo Moro"
Bari, Italy
donato.malerba@uniba.it

Apostolos N. Papadopoulos
Department of Informatics
Aristotle University
Thessaloniki, Greece
papadopo@csd.auth.gr

ABSTRACT

Recent advances on tracking technologies enable the collection of spatio-temporal data in the form of trajectories. The analysis of such data can convey knowledge in prominent applications, and mining groups of moving objects turns out to be a valuable mean to model their movement. Existing approaches pay particular attention in groups where objects are close and move together or follow similar trajectories by assuming that movement cannot change over time. Instead, we observe that groups can be of interest also when objects are spatially distant and have different but inter-related movements: objects can start from different places and join together to move towards a common location. To take into account inter-related movements, we have to analyze the objects jointly, follow their respective movements and consider changes of movements over time. Motivated by this, we introduce the notion of *communities* and propose a computational solution to discover them. The method is structured in three steps. The first step performs a feature extraction technique to elicit the inter-related movements between the objects. The second one leverages a tree-structure in order to group objects with similar inter-related movements. In the third step, these groupings are used to mine communities as groups of objects which exhibit inter-related movements over time. We evaluate our approach on real data-sets and compare it with existing algorithms.

Categories and Subject Descriptors

H.2.8 [**Database Applications**]: Data mining

Keywords

Trajectories, Mining, Groups of Moving Objects.

1. INTRODUCTION

The tremendous advances in positioning technologies, such as telemetry, GPS equipment and smart mobile phones, have enabled tracking of any type of moving objects and collecting spatio-temporal data into growing repositories. Some example applications follow:

- In location-based social networks, people travel in the real world and leave their location history in the form of a *trajectory*. These trajectories do not only connect locations in the physical world but also bridge the gap between people and locations [12].

- Portable GPS devices allow to record the corresponding vehicle locations [11]. Such information often includes data for human mobility.

- Zoologists are investigating the impact of the levels of urbanization on the migration, distribution and habitat use of animals [9].

In the aforementioned applications, one can be interested in the discovery of groups of objects which move together or in a similar manner. For instance, in car pooling it could be useful to determine people with the same route to share the car. Such problems are not novel in the literature [5] and most of the efforts result in mining groups of moving objects, such as *flocks* [1], *convoys* [4] and *swarms* [8].

The spatio-temporal properties of these groups is the main distinguishing aspect. In particular, a flock contains at least m objects moving in the same direction within an circular region with a user-defined radius. Variants of the flock include also a notion of time-interval (with minimum duration defined by the user) according to which in each time-stamp of the interval a disc containing m objects can be identified. The rigid characteristic of fixed circular shape could miss some groups of arbitrary form.

The introduction of the notion of *density* avoids this drawback and allows to discover groups, named as convoys, which have no limitations on the shape and size. A convoy is defined as a cluster of objects and it is identified by means of a density-based clustering technique which checks for the condition of density-connectedness on the objects and for all the time-stamps of a time-interval [4].

A more general group type is represented by the swarm concept, which, in contrast to flocks and convoys, it is not required to hold for all time-stamps of a time-interval, but it can occur more sporadically. In the classical notion of swarm this temporal constraint corresponds to a minimum number of time-stamps which are not necessarily consecutive.

Motivation. The algorithms to detect flocks, convoys and swarms are designed to capture similarities among (sub)trajectories but leave unexplored two interrelated aspects which instead appear to be new sources of information to exploit: *i)* movements may depent on each other and may hide interactions among the objects, *ii)* movements can reflect changes

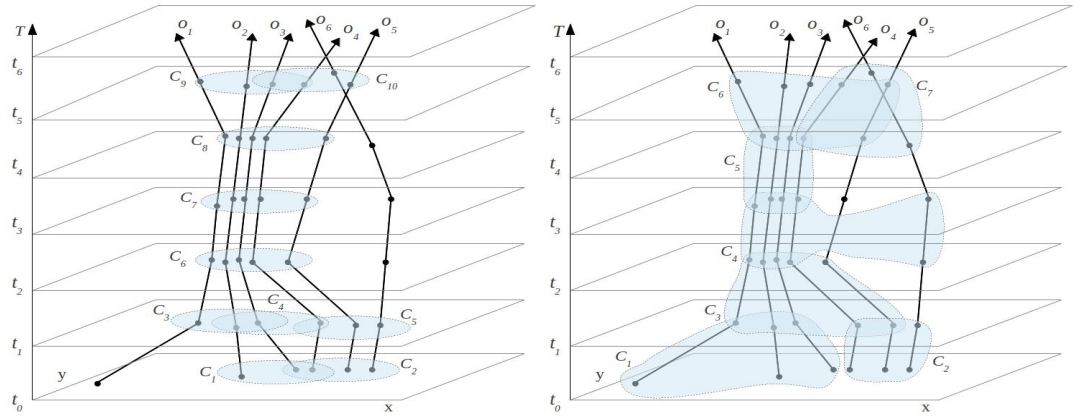


Figure 1: Moving objects grouped using: (a) flocks, convoys and swarms and (b) communities.

of the motion of the objects and implicitly denote their dynamic behaviour. Interactions reflect the possible relationships in which the objects can be involved in space and time, they can provide a more complete description of the groups by explaining even the cause of their formation. Interactions can evolve because the objects can move near each other and then move away. Indeed, moving objects intrinsically are dynamic, their motion is not necessarily linear and it can be influenced by the properties or needs of each object and by the interactions with other objects as well. For instance, in social studies, we can observe individuals which begin to move from different locations, they could come near until to join together in proximity of a point of interest, they could remain there for a time and then go away from each other. So, those individuals can be members of a group even without having followed similar trajectories.

In that kind of problems, a group can turn out to be interesting not only when its members are spatially close and move similarly, but also when they are far apart and have different but inter-related movements, or also when they have different movements but are involved in the same type of interactions. Existing approaches are not prepared to handle this concept, mainly due to the following reasons:

- Most of the existing techniques rely on a static group concept where objects have to always meet the same spatio-temporal properties: for instance, the members of a group are required to be close each other in each time-stamp.
- The trajectory corresponds to a geometric abstraction of the movement and is defined as a series of punctual time-stamped observations that cannot indicate neither how the object moves over time nor whether there exists any form of relationship with other trajectories.

Related Work and Contributions. In this paper, we introduce the concept of *community* based on the concepts of interaction among the objects and change in the movements of the objects. The interaction between two objects o_i and o_j is defined on the basis of the movement that an object o_i performs with respect to another object o_j, while the change concerns the variations of spatio-temporal characteristics that can be observed in the movement of each object. Therefore, changes of an object's motion may influence or determine its interaction with other objects.

A community consists of a set of objects in common to a set of groups arranged in sequence. In its turn, a group contains $n-1$ pairs formed with objects taken from a set of n elements: a pair is formed with one object in common to all pairs (*reference object*) and the other object taken from the remaining $n-1$ (*participants*). The pairs of a group exhibit very similar spatio-temporal features. Differently from flock, convoy and swarm, the timing of a community is based on time-intervals created from the time-stamps of the positions. We clarify the difference between a community and other group types in the following example.

Another notion of *community*, proposed in [10], models the similarities of moving objects in four information sources, namely semantic properties of the locations, temporal duration of the trajectory, spatial proximity and movement velocity. This notion anyway requires that the objects move similarly in all time-stamps whereas the result cannot include communities with discontinuities over time.

Example 1. In Figure 1(a), six objects are tracked and have the positions in six time-stamps included in the time-intervals $[t_0,t_1]$, $[t_1,t_2]$, $[t_2,t_3]$, $[t_3,t_4]$, $[t_4,t_5]$, $[t_5,t_6]$. Let k=3 the minimum number of objects required for the final groups. Clusters $\{C_3,C_6,C_7,C_8,C_9\}$ share the objects $\{o_1,o_2,o_3\}$ which form a flock in $[t_1,t_2]$, $[t_2,t_3]$, $[t_3,t_4]$, $[t_4,t_5]$, $[t_5,t_6]$. The group $\{o_1,o_2,o_3,o_4\}$ corresponds to a convoy if we consider the notion of density-connectedness on the clusters $\{C_4,\ C_6,\ C_7,\ C_8,\ C_9\}$. Finally, with the objects in common to the clusters $\{C_2,C_5,C_{10}\}$ we have the swarm $\{o_4,o_5,o_6\}$. In Figure 1(b), we have two communities, namely $\{o_1,o_2,o_3\}$ and $\{o_4,o_5,o_6\}$ respectively. The first one is composed of the objects in common to the sequence of the groups $\{C_1,C_3,C_4,C_5,C_6\}$, where the group C_1 is collocated into the time-interval $[t_0,t_2]$, C_3 is collocated into the time-interval $[t_1,t_3]$, C_4 is collocated into $[t_2,t_4]$, C_5 is associated with $[t_4,t_6]$. The group C_1 is composed of the pairs (o_2,o_1) (where o_2 is the reference, o_1 is the participant) and (o_2,o_3) (where o_2 in the reference, o_3 is the participant). The other groups can be interpreted in the same manner. The motions of the pairs $(o_2,o_1$)and (o_2,o_3) tells us that they start far apart and tend to move near while observing a variation of the mutual distance (in $[t_0,t_3]$), then, they move together without any variation of the distance

(in $[t_2, t_5]$), finally they move apart (in $[t_4, t_6]$). The community $\{o_4, o_5, o_6\}$ is obtained from the non-consecutive groups $\{C_2, C_7\}$: the first group is collocated into $[t_0, t_2]$, the second group in $[t_4, t_6]$. In this community, the pairs (o_5, o_4) and (o_5, o_6) proceed by keeping the same distance in $[t_0, t_2]$ while they exhibit a reduction of the mutual distance in $[t_4, t_6]$. ■

The previous example shows the difficulty of existing algorithms to discover communities. Indeed, the algorithm for finding flocks is inadequate since it works with clusters in the strict form of a fixed disc. The method for detecting convoys cannot be used since it operates on the density-connectedness corresponding to the simultaneous application of conditions on the size and closeness for each cluster, which are criteria hard to be satisfied when considering distant objects. The difficulty of the algorithm for the discovery of swarms [8] lies in the accommodation of the temporal component and, specifically, in the fact that the members of the swarms are required to stick together for a number of possibly non-consecutive time-stamps. But this could mean having insignificant swarms characterized by completely disjointed time-stamps and fragmented movements. In summary, the contributions of this paper include:

- A new definition of group of moving objects which extends the classical notion of cluster based on the spatial closeness and density-connectedness.
- The exploitation of two new sources of information corresponding to the interactions among the objects and changes of their motions.
- The definition of spatio-temporal features able to model the interactions and changes of the movements of pairs of moving objects.
- The synthesis of a grouping technique which does not rely on a distance/dissimilarity measure.
- A performance evaluation and experimental comparison with existing techniques.

Roadmap. The remainder of this work is organized as follows. The next section presents some fundamental concepts related to our approach. Section 3 studies our proposal in detail. Performance evaluation results are offered in Section 4 whereas Section 5 concludes our work and discuss briefly future research directions.

2. FUNDAMENTAL CONCEPTS

In this section we present some fundamental concepts related to our proposal. Some frequently used symbols are given in Table 1. Let $\mathcal{O}=\{o_1, o_2, \ldots, o_n\}$ be the set of all moving objects and $\mathcal{T} = \{\tau_1, \tau_2, \ldots, \tau_m\}$ be the set of all time-stamps. The trajectory of an object o is a finite sequence of time-stamped locations denoted as $t(o) : \langle (p_1, \tau_1), (p_2, \tau_2), \ldots, (p_m, \tau_m)\rangle$ during the time-interval $[\tau_1, \tau_m]$, where $p_i \in \mathbb{R}^2$ is the geo-spatial position sampled at $\tau_i \in \mathcal{T}$. A trajectory may have time-stamps not necessarily equally distanced, they can be different from those of another trajectory as well as different trajectories may have different lengths (number of geo-spatial positions).

In this work, we do not analyze the original trajectories but we adopt a transformation technique which projects the trajectory data into a descriptive space which includes a finite set of features $\mathcal{F}=\{F_1, \ldots, F_l, \ldots, F_f\}$ which are the real subject of our analysis. The features can take value in categoric or numeric domains. In particular, for each pair (o_u, o_v), the transformation technique returns a set of valued features for the (sub)trajectories observed in two consecutive time-intervals, which we denote as $[\tau_i, \tau_j] \cup [\tau_{j+1}, \tau_k]$ and name as *feature time-intervals*.

Table 1: List of symbols.

Symbol	Explanation
$\mathcal{O}$	all moving objects
$\mathcal{T}$	all time-stamps
$t(o)$ $(t(o_u))$	trajectory of the object o (o_u)
$\mathcal{F}$	set of descriptive features
F_l	l-th features describing a pair of trajectories
$F^l_{o_u,o_v}$	value of the l-th features for trajectories of o_u, o_v
$[\tau_1, \tau_m]$	time-interval containing time-stamps of $\mathcal{T}$
$\mathcal{G}$	pair group
o_r	reference object of a pair group
o_s	participant object of a pair group
$\mathcal{G}_f$	feature group
$\epsilon_{min_l}(\epsilon_{max_l})$	min (max) value of feature F_l
ϵ_l	fixed value of the categoric feature F_l
$\mathcal{C}$	a community

A simple illustration is reported in Figure 2. Consider the trajectories of three objects o_1, o_2, o_3. Let $F^1_{o_u,o_v}$,$F^2_{o_u,o_v}$ be two features which describe the reciprocal movement between the objects o_u, o_v and their average mutual distance respectively. The domain of the feature $F^1_{o_u,o_v}$ has categoric values {"const","far","close"} where "const" corresponds to two objects that travel together by keeping constant their distance, "away" corresponds to two objects that are moving away, and "close" corresponds to two objects that are moving closer. The domain of the feature $F^2_{o_u,o_v}$ has numeric values in the set of natural numbers $\mathbb{N}$. The values of F^1 and F^2 are computed on the feature time-intervals $[\tau_1, \tau_2] \cup [\tau_2, \tau_3]$, $[\tau_3, \tau_4] \cup [\tau_4, \tau_5]$, $[\tau_5, \tau_6] \cup [\tau_6, \tau_7]$. So, for instance, the value of the feature $F^1_{o_1,o_2}$ in the feature time-intervals $[\tau_1, \tau_2] \cup [\tau_2, \tau_3]$ is "const", while the value the feature $F^2_{o_1,o_2}$ is 20. Figure 2 reports the remaining values of the features.

DEFINITION 1 (PAIR GROUP). *Given a subset of $\mathcal{O}$ with*

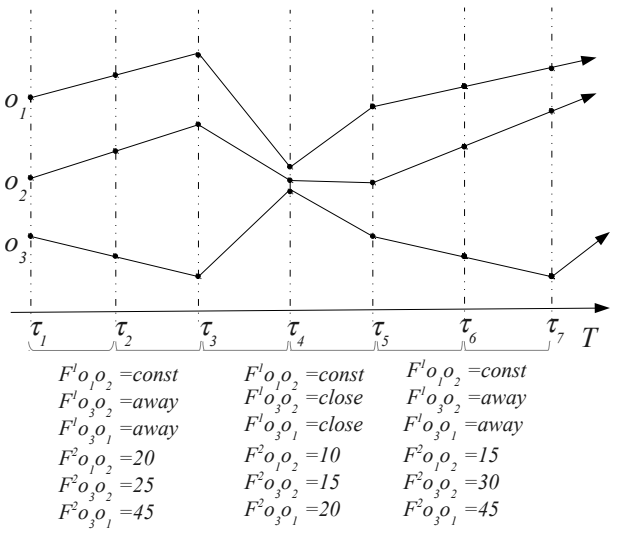


Figure 2: Feature generation from trajectories.

m objects, a pair group $\mathcal{G}$ consists of the $(m-1)$ pairs of objects (o_r,o_s), where $r \in \{1,\ldots,m\}, s = 1,\ldots,m, r \neq s$. The object o_r appears in all pairs and it is named as reference object, while the objects o_s are named participant objects.

For readability, o_r is the first object of each pair in a pair group and each participant corresponds to the second object o_s.

DEFINITION 2 (FEATURE GROUP). *Given $\mathcal{G}$ a pair group, $\mathcal{F}=\{F_1,\ldots,F_l,\ldots,F_f\}$ the set of features, a feature group $\mathcal{G}_f$ consists of the pairs of $\mathcal{G}$ which, in the feature time-intervals $[\tau_i,\tau_{i+k}] \subseteq \mathcal{T}$, $[\tau_{i+k+1},\tau_{i+2k}] \subseteq \mathcal{T}$, ..., $[\tau_p,\tau_{p+k}] \subseteq \mathcal{T}$, $[\tau_{p+k+1},\tau_{p+2k}] \subseteq \mathcal{T}$, satisfy the following conditions*

- $\forall$ *$(o_r,o_s) \in \mathcal{G}$: $\epsilon_{min_l} \leq F^l_{o_{jr},o_{js}} < \epsilon_{max_l}$, iff F_l has numeric values,*

  where $\epsilon_{min_l} \in \mathbb{R}$, $\epsilon_{max_l} \in \mathbb{R}$ are minimum and maximum values respectively for the feature F_l.

- $\forall$ *$(o_r,o_s) \in \mathcal{G}$: $F^l_{o_r,o_s} = \epsilon_l$, iff F_l has categoric values, where ϵ_l is a fixed value in the domain of F_l.*

The values of ϵ_{min_l}, ϵ_{max_l}, ϵ_l are specific for each feature group. The feature time-intervals have identical width and are arranged in chronological order.

Intuitively, a feature group is characterized by two components, one of nature geo-spatial, the other one of nature temporal. Definition 2 says that, in the time-intervals $[\tau_i,\tau_{i+k}] \cup [\tau_{i+k+1},\tau_{i+2k}]$, ..., $[\tau_p,\tau_{p+k}] \cup [\tau_{p+k+1},\tau_{p+2k}]$, the pairs of objects of $\mathcal{G}$ have the same value for each categorical feature and the same range for each numeric feature. For instance in Figure 2, we have a feature group formed by the pairs (o_3,o_2), (o_3,o_1) in the time-intervals $[\tau_1,\tau_2] \cup [\tau_2,\tau_3]$ and $[\tau_5,\tau_6] \cup [\tau_6,\tau_7]$. Indeed, considered ϵ_1="away" (ϵ_l for F^1), ϵ_{min_2}=25, ϵ_{max_2}=50 (respectively, ϵ_{min_l} and ϵ_{max_l} for F^2), the values of the feature F^1 are the same ("away") and the values of the feature F^2 have the same numeric range. These conditions hold in the feature time-intervals $[\tau_1,\tau_2] \cup [\tau_2,\tau_3]$ and $[\tau_5,\tau_6] \cup [\tau_6,\tau_7]$, but they do not hold in the time-intervals $[\tau_3,\tau_4] \cup [\tau_4,\tau_5]$ because the value of the feature F^1 is "close" which is different from "away".

DEFINITION 3 (COMMUNITY). *A set of feature groups $\{\mathcal{G}_{f_1}, \mathcal{G}_{f_2}, \ldots, \mathcal{G}_{f_n}\}$ defines a community $\mathcal{C}$ iff:*

- *the feature groups $\mathcal{G}_{f_1}$, $\mathcal{G}_{f_2}$, ..., $\mathcal{G}_{f_n}$ consists of the same pair group $\mathcal{G}=\mathcal{G}_1=\mathcal{G}_2 = \ldots = \mathcal{G}_n$ composed by (m-1) pairs of objects with the same reference object and the same set of m-1 participants.*

- *the feature time-intervals of two different feature groups are disjointed $([\tau_i,\tau_{i+k}] \cup [\tau_{i+k+1},\tau_{i+2k}] \bigcap [\tau_p,\tau_{p+k}] \cup [\tau_{p+k+1},\tau_{p+2k}]) = \oslash$ and chronologically ordered $(i+2k<p)$.*

The sequence of the feature time-intervals associated with the feature groups is called time-line.

For instance, in Figure 2, we have a community formed by the pairs (o_3,o_2) and (o_3,o_1) in the time-line $[\tau_1,\tau_3]$, $[\tau_3,\tau_5]$, $[\tau_5,\tau_7]$, where o_3 is the reference object, o_2 and o_1 are participant objects. In particular, in the feature time-intervals $[\tau_1,\tau_2] \cup [\tau_2,\tau_3]$ and $[\tau_5,\tau_6] \cup [\tau_6,\tau_7]$, the feature F^1 has value "away", while the feature F^2 has values in the range [25,50) (ϵ_1="away", ϵ_{min_2}=25, ϵ_{max_2}=50). In the feature time-interval $[\tau_3,\tau_4] \cup [\tau_4,\tau_5]$, the feature F^1 has value "close", while the feature F^2 has values in the range [15,25) (ϵ_1="close", ϵ_{min_2}=15, ϵ_{max_2}=25).

To capture possible discontinuities, we should handle the case in which $i + 2k < p - 1$, namely when the feature time-intervals are separated over time. At this aim, we introduce an input parameter γ which defines the maximum temporal gap that can be admitted between two feature time-intervals.

Now, we can give a formal statement of the problem of discovering communities from trajectories:

> **Given** a set of moving objects $\mathcal{O}$ and the corresponding trajectories, a set of time-stamps $\mathcal{T}$, the features $\mathcal{F}$ and the width of the associated time-intervals Δ, **Discover** the communities as formalized in Definition 3: for each community $\mathcal{C}$, the temporal gap in the time-line does not exceed γ and the number of involved objects is greater than or equal to the minimum input threshold *minO*.

3. PROPOSED METHOD

The proposed solution comprises three steps: *i)* transformation of the original trajectories in descriptive spatio-temporal features, *ii)* arrangement of the feature vectors produced in the previous step in a tree-like structure in order to generate feature groups and *iii)* discovery of communities from feature groups.

3.1 Transformation of Trajectory Data

Tracking devices often record the positions of moving objects with irregularity and discontinuity, mainly due to physical and instrumental factors which can affect the data quality. To remove possible inconsistencies we have to handle this kind of error sources. Moreover, the analysis of interactions among objects, we intend to conduct, suggests that we should apply a pre-processing step able to return positions (of the objects) equally distanced over time, so that the trajectories can be handled with regular timing. We adopt a data transformation technique which first performs a temporal segmentation operation and then projects the segmented trajectories into the descriptive space. Preliminarily, an outlier removal operation is applied on the trajectories.

The temporal segmentation performs a discretization step on the set $\mathcal{T}$ and generates time-intervals $[\tau_i, \tau_{i+k}]$, $[\tau_{i+k+1}, \tau_{i+2k}]$, ..., $[\tau_p, \tau_{p+k}]$, $[\tau_{p+k+1}, \tau_{p+2k}]$ with width equal to Δ. This allows to have a sort of re-sampling of the trajectories at regular time-stamps. In particular, for each object a single geo-spatial location is associated with the set of positions observed in each time-interval (segment). This location is determined by an aggregation operation applied to the original positions in a time-interval. As aggregation operator we prefer to use the geometric mean due to its simplicity and because other pre-processing operations (such as, smoothing and interpolation) could introduce data loss and potential creation of artifact in the trajectory data.

The descriptive space includes spatio-temporal features defined to model the interactions and changes of the movements of pairs of objects. The use of new descriptors to represent the original trajectories is not novel. In the literature we can find several types of features (also called movement

parameters) which have been defined basically for eliciting information which the trajectories are not able to do directly [3]. Typically, features are produced by simple feature extraction algorithms applied to original trajectories and their purpose is to model physical and spatial characteristics of the movements, such as speed, acceleration, duration, direction, etc. In this work, the features are extracted from the aggregate values computed in two consecutive time-intervals (segments). More precisely, the value of a feature is computed for each pair of objects and it is determined from the two aggregate values computed in the respective time-intervals for each object of the pair. We investigate six features defined as follows (please refer to Figure 3):

Categoric Reciprocal Movement (CRM) is the feature which represents the movement of an object with respect to the movement of another one. It takes five possible categoric values in function of the two aggregate locations. The set of possible values was defined manually and comprises {"one_away", "both_away", "const", "one_close", "both_close"}. More specifically, "const" corresponds to two objects that travel together by keeping their distance constant. We have "one_away" when one of the two objects is moving away from the other one while the latter does not change. The value "one_close" occurs when one of the two objects is moving close while the trajectory of the other one remain unchanged. The value "both_away" corresponds to two objects that are moving away from each other. On the other hand, when the trajectories tend to move close we have "both_close".

Numeric Reciprocal Movement (NRM) is the feature which, like CRM, represents the movement of an object relatively to another one but with numeric values. The value of NRM is derived from the distances computed, in each time-interval, between the two aggregate locations of the pair of objects. It is equal to the difference between these two distances. Thus, when two objects are moving close to each other, NRM has a negative value, while, otherwise the value is positive.

Displacement (DIS) denotes the displacement done by the pair of objects over the two time-intervals. The value of DIS is derived from the middle locations between the two aggregate locations (in the two time-intervals) and it is equal to the distance between the two middle locations.

Cardinal Direction (CD). The features listed above provide a spatial description of the movement without specifying any geographic connotation. We introduce the feature CD in order to elicit the information about the spatial direction and capture that information as the classical cardinal direction of the movement of the pair of objects. The value of CD is derived from the middle locations between the two aggregate locations (in the two time-intervals) and it takes the direction which goes from the middle location of the first time-interval towards the middle locations of the second time-interval.

Position (POS). The purpose of this feature is to provide information on the localization of the movement. Indeed, the features listed above cannot distinguish whether two identical movements are localized in the neighbourhood or in completely distant locations. The value of POS is derived from the middle locations between the two aggregate locations (in the two time-intervals) and it corresponds to the middle point of the two middle locations.

Finally, for each pair of objects o_u and o_v, the transformation technique returns a vector of valued features $\langle$ CRM, NRM, DIS, CD, POS $\rangle$ computed on the two consecutive time-intervals $[\tau_i, \tau_{i+k}]$, $[\tau_{i+k+1}, \tau_{i+2k}]$. It is worthwhile that the extraction of features for each pair of objects on consecutive time-intervals has a two-fold result: *i)* modelling the interaction of the smallest admissible group of objects (namely, two objects), and *ii)* capturing relevant changes of their movements which turn out to be evident only on time-intervals rather than instantaneous time-stamps.

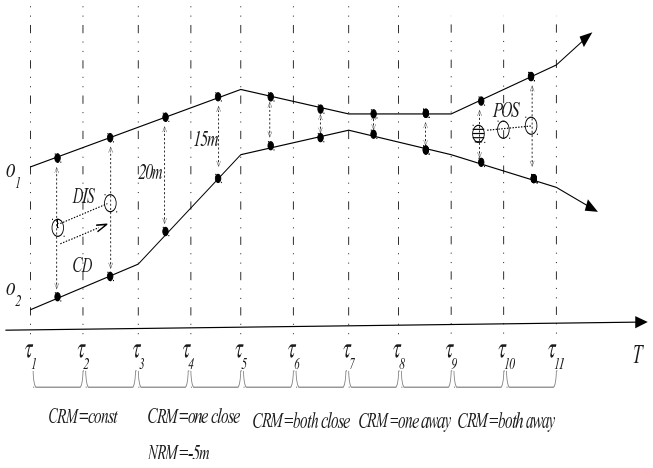


Figure 3: Trajectory transformation.

3.2 The Feature Tree

Once the feature vectors have been generated, they populate a B-tree [2] which is used to discover first the feature groups and then the communities. The tree structure does not change when the vectors are inserted and it is defined on the basis of the set of features introduced in Section 3.1. The arrangement of tree levels is such to represent the features in the order {CRM, NRM, CD,DIS, POS-x, POS-y} (Figure 4(a)). The feature ordering is decided by domain experts on the basis of their criteria about the discriminative power of the features. Thus, the features CRM and NRM are ranked first because they depict, better than the others, the interaction in a pair of objects. Then, we place features CD and DIS because they are able to denote characteristics on the changes of the movement, and, finally the features POS-x and POS-y which provide a spatial indication not directly related to the interactions and changes in moving objects.

Nodes of a specific level refer to one feature and access to nodes (children) of the lower level which, in their turn, refer to another feature. More precisely, a node has as many child nodes as the number of the possible values of the feature associated with its level, therefore, the number of nodes of a specific level is equal to the number of the possible values of the feature associated with the parent level. In the case of categorical features, the child nodes are denoted with distinct values ϵ_l defined in Section 3.1. For example, at the level associated with feature CD, the nodes have eight child nodes, one for each value of the set {"north", "north-east", "east", "south-east", "south", "west", "south-west", "north-west"}. In the case of numeric features, the child nodes are denoted with distinct ranges $[\epsilon_{min_l}, \epsilon_{max_l})$ produced by a discretization technique. In this work, we adopt equi-frequency discretization since it guarantees the balancing of

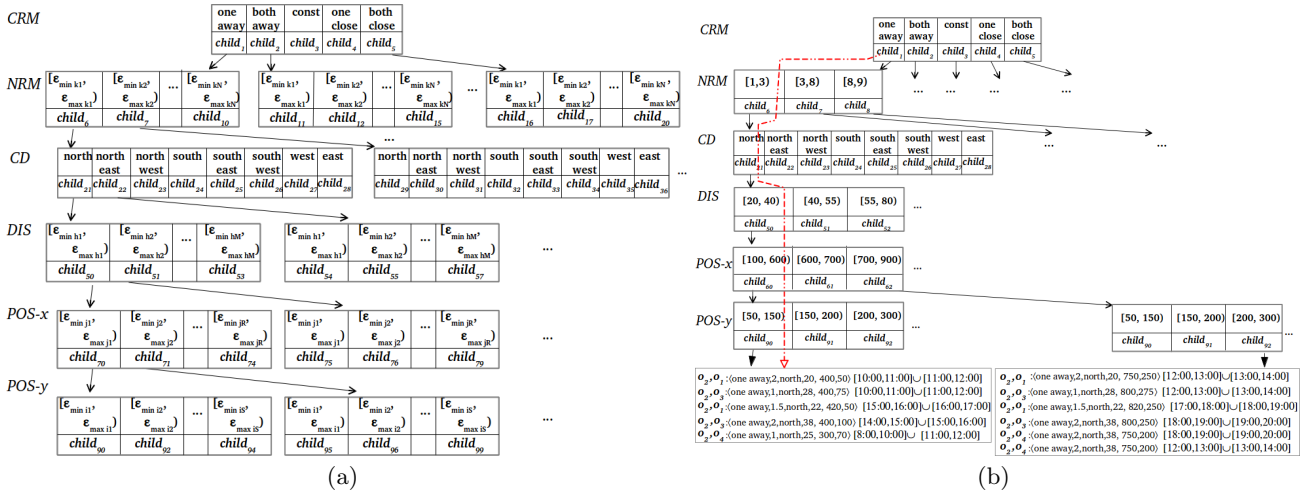


Figure 4: Levels are associated with features and nodes have as many children as the values of the associated features. The red dotted line illustrates a path example.

the tree due to the uniform distribution of vectors to ranges.

This tree structure allows us to collocate in the same node the vectors whose values of the feature are identical (ϵ_l) or are included in the same range ($[\epsilon_{min_l}, \epsilon_{max_l})$). The root node contains vectors which have only one feature with identical value (categoric), while the leaf nodes contain vectors which have all categoric features with identical values and all numeric features with values included in the same ranges. Therefore, the vectors collocated in the same leaf node will be those that have traversed the same path in the tree and that we consider similar since share the same categoric values and same numeric ranges. For instance, in the leftmost leaf node in Figure 4(b), the pairs (o_2,o_1), (o_2,o_3) share the same categoric values, namely "one away" for CRM and "north" for CD, and the same numeric ranges, namely [1,3) for NRM, [20,40) for DIS, [100,600) for POS-x, and [50,150) for POS-y.

The insertion process starts at the root and descends the tree. For each level, it chooses the node whose value of the associated feature is identical to (categorical) or includes (numeric) the value of the same feature of the current vector. From the chosen node we access to its child nodes where we replicate the insertion considering the appropriate feature until to reach the leaf nodes.

3.3 Feature Groups and Communities

We exploit the structure of the feature tree to determine the geo-spatial and temporal components of feature groups which are, in their turn, necessary for the communities. From each leaf node we can extract at least one feature group. Indeed, the pair group $\mathcal{G}$ of a feature group can be searched among the pairs of the inserted vectors, while the geo-spatial component is determined directly from the tree path which characterizes each leaf node. The temporal component is computed by the method given in the sequel.

The method analyses the content of the leaf nodes separately and, for each of these, it searches the feature time-intervals which are in common. In particular, the method identifies all possible pair groups (Definition 1) present in each leaf node and, for each pair group, it processes all sequences of feature time-intervals in order to find the sequences in common. The analysis is thus focused on each pair group and is conducted in two phases: first, generation of candidate sequences, then, selection of the more interesting candidates with respect to preference criteria.

In the first phase, we adopt the efficient algorithm proposed in [6] in order to find sequences (candidates) in common between a reference sequence of feature time-intervals and the set of all sequences of the current pair group. The algorithm solves the problem by searching the intersections between the feature time-intervals of the reference sequence and the feature time-intervals of the remaining sequences. In particular, for each time-interval of the reference sequence (query interval), the algorithm uses two binary search operations, one into the sorted list of the time-stamps which terminate the time-intervals and the other into the sorted list of the time-stamps which open the time-intervals. Each search excludes the time-intervals that cannot intersect the query interval, leaving solely those that must intersect the query interval. A detailed description can be found in [6]. Eventually, the intersecting time-intervals are sorted and combined to form the candidate sequences.

In the second phase, two selection operations are performed, one subsequent to the other one. The first one filters out the candidates which have time-intervals shorter than the width Δ, while the second one selects the candidates which meet the preference criteria. We have two preference criteria, one alternative to the other one. The first criteria (*maxInterval, MI*) aims to prefer feature groups with feature time-intervals as long as possible, while the second criteria (*maxObjects, MO*) aims to pick feature time-intervals associated with the largest set of pairs as possible. Note that the considered preference criterion is strictly connected to the choice of the reference sequence seen in the first phase. So, when we choice *maxInterval* the reference sequence is chosen as the longest sequence in the set of all sequences of the pair group, while when we choice the criteria *maxObjects* the reference sequence is chosen as the shortest sequence which has the maximum number of pairs, since feature groups with higher number of pairs are more probable in shorter sequences.

The result of the two phases consists in only one sequence

Algorithm 1 COM /* community discovery */

Input: $\{\mathcal{G}_{f_1}, \mathcal{G}_{f_2}, \ldots, \mathcal{G}_{f_n}\}$, γ, o_r, $minO$
Output: $\mathcal{T}_{lines}$

```
1: T_L ← ∅; T_lines ← ∅; D ← ∅;
2: S ← sort_by_time({G_f1, G_f2, ..., G_fn});
3: F_prev ← nextTimeInterval(S, τ1);
4: insert(T_L, F_prev); remove(S,F_prev);
5: while S ≠ ∅
6:     F_next ← nextTimeInterval(S, getEndTimeStamp(F_prev));
7:     if gap(F_prev, F_next) ≤ γ
8:         if testParticipants(getParticipants(F_prev),
                   getParticipants(F_next))
8:             update(T_L, F_next); remove(S, F_next)
9:             F_prev ← F_next;
10:        else
11:            insert(D, F_next); remove(S, F_next);
12:    else
13:        S ← S ∪ D; T_L ← ∅;
14:        D ← ∅; T_lines ← T_lines ∪ {T_L};
14:        F_prev ← nextTimeInterval(S,τ1);
15:        insert(T_L, F_prev); remove(S, F_prev);
16: prune_by_minO(T_lines);
```

which contains the feature time-intervals shared in the current pair group. It provides a temporal characterization which completes the description of the feature group.

According to Definition 2 and the structure of the feature tree, a reference object is associated with only one feature group in each leaf node. Thus, a reference object is associated with a set of feature groups $\{\mathcal{G}_{f_1}, \mathcal{G}_{f_2}, \ldots, \mathcal{G}_{f_n}\}$ computed from all leaf nodes. These feature groups anyway have different sets of participant objects. The method for discovering communities follows this same idea and builds groups of moving objects relatively to reference objects. It works with the feature groups of the same reference object and operates on the selected sequences of the feature time-intervals by generating a sequence of ordered feature time-intervals (time-line) with the same set of participant objects.

Two alternatives are adopted depending on the chosen preference criterion (*maxObjects* or *maxInterval*). They operate in the same way (Algorithm 1) but they differ in the following aspect: the first technique aims at generating time-lines with the larger number of participant objects, while the second one aims at generating time-lines with the longer feature time-intervals. Both techniques start off with sorting (by chronological order) the sequence of the time-intervals of the feature groups associated with the current reference object o_r. This can return an ordering in which the time-intervals of the same feature group are separated and time-intervals of different feature groups are adjacent.

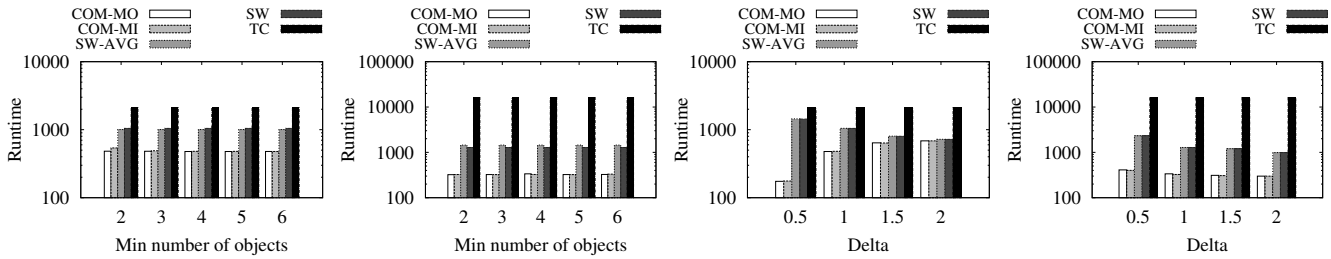
Algorithm 1 generates a time-line incrementally by evaluating whether joining the next time-interval (getNextTimeInterval()) to the current time-line ($\mathcal{T}_L$). In particular, the time-interval F_{prev} is considered for the join whether *i)* it follows temporally the last time-interval added to the time-line $\mathcal{T}_L$ and there is no time-interval with the same participants which precedes it, and *ii)* it is not temporally distant more than γ. Thus, the time-line is updated whether the test is positive. The implementation of this test distinguishes the two techniques: for *maxObjects* the test is implemented as getParticipants(F_{prev}) = getParticipants(F_{next}), while for *maxInterval* the test is getParticipants($\mathcal{T}_L$) ∩ getParticipants (F_{next}) ≠ ∅, where getParticipants($\mathcal{T}_L$) returns the participants which are in common to the time-intervals added to $\mathcal{T}_L$. The output ($\mathcal{T}_{lines}$) is a set of candidate time-lines which are further processed: the time-lines with number of participants less than $minO$ are pruned, then, from those remaining, we select only the time-line which better satisfies the preference criterion (either highest number of participants or longest sequence of time-intervals).

EXAMPLE 2. We extract some feature groups and communities based on Figure 4(b) with Δ=1 hour. On the leftmost leaf node, we have a feature group $\mathcal{G}_{f_1}$ whose pair group is composed by the pairs (o_2, o_1) and (o_2, o_3), the geo-spatial component is equal to "one away" (CRM), "north" (CD), [1,3) (NRM), [20,40) (DIS), [100,600) (POS-x), and [50,150) (POS-y), while the temporal component corresponds to the sequence of intersecting feature time-intervals ⟨ [10:00,12:00], [15:00,16:00] ⟩. On the rightmost leaf node, we see a feature group $\mathcal{G}_{f_2}$ whose group consists of the pairs (o_2, o_1), (o_2, o_3), and (o_2, o_4) the geo-spatial component is equal to "one away" (CRM), "north" (CD), [1,3) (NRM), [20,40) (DIS), [700,900) (POS-x), and [200,300) (POS-y), while the temporal component corresponds to the sequence of intersecting feature time-intervals ⟨ [12:00,14:00], [18:00,19:00] ⟩. Let o_2 be the reference object and γ=4 hours. The time-intervals are sorted as follows ⟨ [10:00, 12:00] ($\mathcal{G}_{f_1}$), [12:00, 14:00] ($\mathcal{G}_{f_2}$), [15:00, 16:00] ($\mathcal{G}_{f_1}$), [18:00, 19:00] ($\mathcal{G}_{f_2}$) ⟩. By choosing the criterion *maxObjects*, we obtain the community composed of the pairs (o_2,o_1),(o_2,o_3), and (o_2,o_4) which exhibit on the time-line ⟨ [12:00, 14:00] [18:00, 19:00] ⟩ the movement so described: "one away" (CRM), "north" (CD), [1, 3) (NRM), [20, 40) (DIS), [700, 900) (POS-x), and [200, 300) (POS-y). Instead, by choosing the criterion *maxInterval*, we obtain the community composed by the the pairs (o_2,o_1) and (o_2,o_3) which exhibit the movement "one away" (CRM), "north" (CD), [1, 3) (NRM), [20, 40) (DIS), [100, 600) (POS-x), [50, 150) (POS-y) in [10:00, 12:00], [15:00, 16:00], and the movement "one away" (CRM), "north" (CD), [1,3) (NRM), [20, 40) (DIS), [700, 900) (POS-x), [200, 300) (POS-y) in [12:00, 14:00], [18:00, 19:00]. ■

4. PERFORMANCE EVALUATION

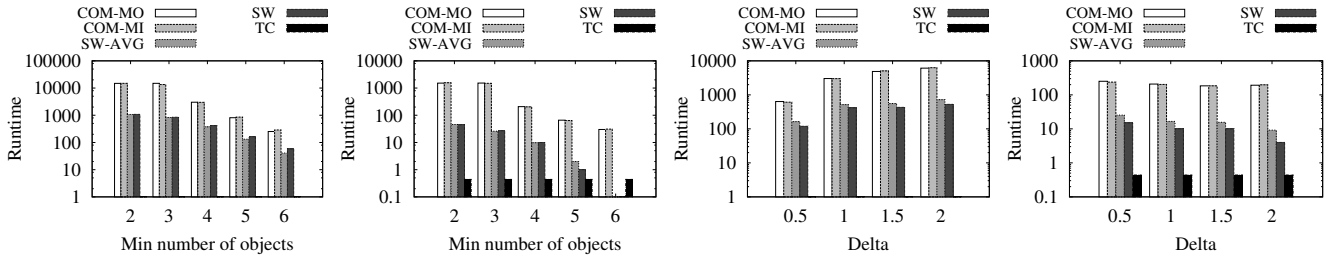
Experiments were conducted in order to test the efficiency of **COM** and the influence of the parameters on the discovered communities with both preference criteria (**COM-MO** and **COM-MI**). We performed also comparative experiments with two competitors. The first one (**TC**) is used as baseline and it aims at discovering the common sub-trajectories with a density-based line-segment clustering algorithm [7]. It takes user-defined parameters on the minimum number of line-segments and radius of the clusters. The second competitor (**SW**) discovers groups of objects moving for certain snapshots that could be not consecutive [8]. The algorithm **SW** works on pre-existing clusters and adopts a candidate generation strategy. It takes user-defined parameters on the minimum number of the objects and minimum duration the swarms (which correspond to $minO$ and Δ of **COM**). These algorithms have a different input parameterization, so we conducted a different comparative analysis. Moreover, **SW** cannot be directly applied since it does not handle either trajectories of different length or missing values. To perform a fair comparison, we tested it on the pre-processed trajectories returned by the temporal segmentation (Section 3.1). **TC** was used with the original data since it handle trajectories of different length.

We evaluated the performance of the algorithms using



(a) Starkey (b) Geolife (c) Starkey (d) Geolife

Figure 5: Runtime (in seconds) vs. $minO$ and Δ (γ=1 hour).



(a) Starkey (b) Geolife (c) Starkey (d) Geolife

Figure 6: Number of results vs. $minO$ and Δ (γ=1 hour).

two real-world datasets: *i*) **Microsoft Geolife**[1] which comprises the trajectories of 182 users outdoor movements ($\mathcal{O}$) in a period of over three years sampled every 1-5 seconds or every 5-10 meters. This dataset contains almost 24 millions of observations in a set of 18 millions time-stamps ($\mathcal{T}$). *ii*) **Starkey**[2] which has been generated by the Starkey project and contains radio-telemetry locations of the movements of 128 elks ($\mathcal{O}$). The observation period is May 1993-August 1996 and comprises 168,000 distinct recordings in 166,000 time-stamps ($\mathcal{T}$). Each object has a portion of 0.09 observations per time-stamp. In both datasets, trajectories have different length and can contain positions recorded at different time-stamps.

Figures 5(a) and 5(b) illustrate the results of the efficiency when tuning $minO$. The results of **SW-AVG** include also the running times averaged on Δ={1/2, 1, 1.5, 2} hours, while those of **TC** are averaged on several settings of the input parameters. We observe that the running times of **COM** are significantly lower than those of **SW** and **TC** (y-axis is logarithmic). In addition, the performance of **COM** with respect to **SW** can be explained with the fact that **SW** spends time in a preliminary density-based clustering and exploration of the search space of the candidate swarms. **TC** requires more time because the clustering decision requires a distance measure on sub-trajectories whose execution is computationally intensive. **COM** exhibits the best runtimes also when tuning Δ (Figures 5(c) and (d)) but with a different behaviour due to the different density of the trajectories, as said before: in Geolife we have essentially a slight decreasing tendency, while it is increasing in Starkey. The decrease exhibited by **SW**, when increasing Δ, is due to the reduced number of clusters that are likely to be extracted from wider time-intervals.

[1] http://research.microsoft.com/apps/catalog/default.aspx?t=downloads

[2] http://www.fs.fed.us/pnw/starkey/data/tables

The different density and distribution of the two datasets is the key to analyze the number of the groups (communities, swarms and clusters) when varying Δ (Figure 6, $minO$=4). The results of **SW-AVG** are averaged on $minO$={2,3,4,5,6}. **COM** and **SW** have similar behaviour, namely slightly decreasing in Geolife and increasing in Starkey. This comfort us about the response of our approach with respect to trajectories which have very different characteristics. A deeper inspection reveals the different order of magnitude between the communities and swarms: this is quite expected since **SW** works on the spatial closeness of the objects, while **COM** can generate groups of objects even when they are not close. Instead, **TC** discovers an average number of clusters which is less than one. This difficulty could be due to the inherent complexity that an operation of grouping of line-segments can raise with respect to grouping simple geo-spatial locations, as in the case of **SW**.

5. CONCLUSIONS

We investigated the problem of mining groups of moving objects from trajectory data. Different from the existing proposed approaches relying on the spatial closeness (flock, convoy and swarm), the current work considers the interactions among the objects and changes of their motions which opens to the possibility of following the complete dynamics of a group. The proposed solution integrates an efficient grouping technique which avoids to re-scan all data. Experiments remark the efficiency with respect to other algorithms. We plan to extend the work in several directions including: *i)* the integration of pre-processing techniques (e.g., locality sensitive hashing) to guide the discovery process on sets of moving objects of particular interest, *ii)* the adaptation of the approach to a distributed architecture (e.g., MapReduce framework) to analyze massive trajectory data, and *iii)* the construction of the feature tree without considering any pre-defined order of the features.

6. REFERENCES

[1] M. Benkert, J. Gudmundsson, F. Hübner, and T. Wolle. Reporting flock patterns. *Comput. Geom.*, 41(3):111–125, 2008.

[2] D. Comer. The ubiquitous b-tree. *ACM Computing Surveys*, 11:121–137, 1979.

[3] S. Dodge, P. Laube, and R. Weibel. Movement similarity assessment using symbolic representation of trajectories. *International Journal of Geographical Information Science*, 26(9):1563–1588, 2012.

[4] H. Jeung, M. L. Yiu, X. Zhou, C. S. Jensen, and H. T. Shen. Discovery of convoys in trajectory databases. *PVLDB*, 1(1):1068–1080, 2008.

[5] P. Kalnis, N. Mamoulis, and S. Bakiras. On discovering moving clusters in spatio-temporal data. In C. B. Medeiros, M. J. Egenhofer, and E. Bertino, editors, *SSTD*, volume 3633 of *Lecture Notes in Computer Science*, pages 364–381. Springer, 2005.

[6] R. M. Layer, K. Skadron, G. Robins, I. M. Hall, and A. R. Quinlan. Binary interval search: a scalable algorithm for counting interval intersections. *Bioinformatics*, 29(1):1–7, 2013.

[7] J.-G. Lee, J. Han, and K.-Y. Whang. Trajectory clustering: a partition-and-group framework. In *ACM SIGMOD Conference*, pages 593–604, 2007.

[8] Z. Li, B. Ding, J. Han, and R. Kays. Swarm: Mining relaxed temporal moving object clusters. *PVLDB*, 3(1):723–734, 2010.

[9] Z. Li, J. Han, M. Ji, L. A. Tang, Y. Yu, B. Ding, J.-G. Lee, and R. Kays. Movemine: Mining moving object data for discovery of animal movement patterns. *ACM TIST*, 2(4):37, 2011.

[10] S. Liu, S. Wang, K. Jayarajah, A. Misra, and R. Krishnan. Todmis: mining communities from trajectories. In *CIKM*, pages 2109–2118, 2013.

[11] J. Yuan, Y. Zheng, C. Zhang, W. Xie, X. Xie, G. Sun, and Y. Huang. T-drive: driving directions based on taxi trajectories. In *GIS*, pages 99–108, 2010.

[12] Y. Zheng, X. Xie, and W.-Y. Ma. Geolife: A collaborative social networking service among user, location and trajectory. *IEEE Data Eng. Bull.*, 33(2):32–39, 2010.