Author manuscript, published in "IEEE Transactions on Knowledge and Data Engineering (2011) 1-17"
DOI : 10.1109/TKDE.2011.39

Segmentation and Sampling of Moving Object Trajectories based on Representativeness

Costas Panagiotakis, Nikos Pelekis, Ioannis Kopanakis, Emmanuel Ramasso, and Yannis Theodoridis

Abstract—Moving Object Databases (MOD), although ubiquitous, still call for methods that will be able to understand, search, analyze, and browse their spatiotemporal content. In this paper, we propose a method for trajectory segmentation and sampling based on the representativeness of the (sub-)trajectories in the MOD. In order to find the most representative sub-trajectories, the following methodology is proposed. First, a novel global voting algorithm is performed, based on local density and trajectory similarity information. This method is applied for each segment of the trajectory, forming a local trajectory descriptor that represents line segment representativeness. The sequence of this descriptor over a trajectory gives the voting signal of the trajectory, where high values correspond to the most representative parts. Then, a novel segmentation algorithm is applied on this signal that automatically estimates the number of partitions and the partition borders, identifying homogenous partitions concerning their representativeness. Finally, a sampling method over the resulting segments yields the most representative sub-trajectories in the MOD. Our experimental results in synthetic and real MOD verify the effectiveness of the proposed scheme, also in comparison with other sampling techniques.

Index Terms—Trajectory Segmentation; Subtrajectory sampling; Data mining; Moving Object Databases

hal-00585610, version 1 - 13 Apr 2011

1 Introduction

Nowadays, there is a tremendous increase of Moving Objects Databases (MOD) [1] due to, on the one hand, location-acquisition technologies like GPS and GSM networks and, on the other hand, computer vision based tracking techniques. This explosion of information combines an increasing interest in the area of trajectory data mining and, more generally, knowledge discovery from movement-aware data [2]. All these technological achievements require new services, software methods, and tools for understanding, searching, retrieving, and browsing spatiotemporal trajectories content.

In this paper, we tackle a problem combining three different aspects. First of all, we study the problem of alternative representations of trajectories of moving objects (other than the usual sequences of 3-D line segments), according to contextual information that can be automatically derived by the total trajectory population. More specifically, we investigate for an effective way to represent each trajectory by a continuous function that implicitly describes the "representativeness" of each constituent part of it (i.e. a segment) w.r.t. the whole MOD. Given such an intuitive representation, a second interesting arising problem is that of its segmentation in a way that an analyst could gain insight into "representative" (i.e. interesting, dense, frequent) portions (i.e. sub-trajectories), but also into "non-representative" parts, which are also of interest in various application scenarios (for example, in detecting movement outliers). On top of the previous issues, and due to the complex nature of the trajectory data and the vast volumes of MOD, a third interesting problem arises; that of "trajectory sampling". This is a very challenging problem where very limited work has been carried out so far. An insightful solution to the problem would be an analyst to be able to supervise the sampling procedure, not only regarding the volume of the sampled dataset, but also the properties of the dataset that reveal the underlying movement patterns of the MOD. In this paper, we argue that this problem can be effectively tackled if interconnected to the previous two discussed problems. In other words, we propose an automatic method for sub-trajectory sampling based on the "representativeness" of the sub-trajectories. In this approach, an analyst may request the top-k representative sub-trajectories that best describe the MOD in an optimised way, where optimization is with respect to the "representativeness". Fig. 1 illustrates an example of a MOD comprised by four trajectories $(\{T_1, ..., T_4\})$ and the top-2 representative sub-trajectories $(\{S_1, S_2\})$ that best describe the MOD.

Recently, in the literature there have been proposed several works that try to either efficiently analyze trajectory data or mine movement-aware patterns. In the domain of trajectory segmentation related works deal with the problem locally, by partitioning trajectories in

- *C. Panagiotakis is with the Dept. of Commerce and Marketing, Tech. Educational Inst. of Crete and with the Dept. Of Computer Science, University of Crete, Greece. E-mail: cpanag@csd.uoc.gr*
- *N. Pelekis is with the Dept. of Statistics and Insurance Science, University of Piraeus, Greece. E-mail: npelekis@unipi.gr*
- *I. Kopanakis is with the Dept. of Commerce and Marketing, Tech. Educational Inst. of Crete, Greece, Greece. E-mail: i.kopanakis@emark.teicrete.gr*
- *E. Ramasso is with the Dept. of Automatic Control and Micro-Mechatronic Systems, FEMTO-ST Research Inst., France. E-mail: e_ramasso@yahoo.fr*
- *Y. Theodoridis is with the Dept. of Informatics, University of Piraeus, Greece. E-mail: ytheod@unipi.gr*

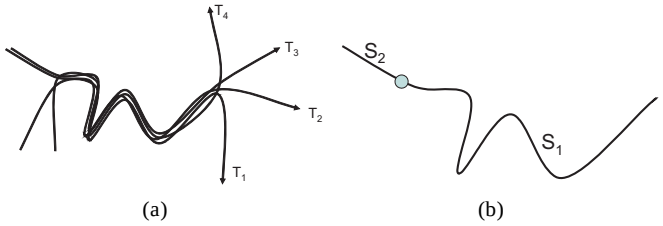


Fig. 1: **(a)** A MOD of four trajectories. **(b)** The two most representative sub-trajectories.

a way as to achieve better database organization [3] or extract more intuitive local patterns for clustering [4] and classification purposes [5]. In [6], a global distance-based approach was proposed for the segmentation of object trajectories that attempts to simplify the trajectories by using minimum bounding rectangles (MBRs) in a way that the original pairwise distances between trajectories are minimized, as such facilitating clustering and classification tasks. Furthermore, there are approaches that attempt to extract global patterns by clustering trajectories [7]–[9] based on different types of distance functions, while others try to take advantage of the partial (local) grouping of the trajectories [4], [10], [11]. There are also hybrid approaches that succeed global clustering by using local criteria and focusing in symbolic representations of the trajectories [12]. In the domain of sampling, although there are nice solutions for point data [13], [14], to the best of our knowledge, trajectory-oriented solutions include explorative, user-supervised, clustering-based sampling techniques [15], [16] as well as approaches that operate on approximate trajectories [17], with all of those solutions trying to select trajectories as wholes without being able to identify representative sub-trajectories, a problem completely different, which is exactly the challenge accepted by the current work.

Most of the above mentioned approaches propose different similarity metrics which they are used either for introducing indexing structures for efficient trajectory retrieval [7], or for clustering purposes [8], [9]. Some techniques simplify the given trajectories, focus on spatial criteria and ignore the temporal dimension [4], [5]. Other proposals either cannot capture the complex nature of the trajectories that follow arbitrary motion patterns [4], or focus on the discovery of very specific definitions of movement patterns [10], [11], [18]. Explorative, user-supervised approaches [15], [16] do not provide deterministic solutions as they base on clustering techniques, which in their turn rely on the selection of the clustering algorithm, its parameters and the intrinsic distance function used. On the contrary, in this paper, we do not simplify the MOD by any time-consuming preprocessing steps, we take into account the temporal information, while we do not make any assumption regarding the type of pattern, that is the actual final outcome. Given this $no-prerequisites$ setting, we argue that all of the above approaches, as well as those which are dealing with vast volumes of trajectory datasets would benefit if they would be applied in a representative subset (consisting of the representative sub-trajectories) that best describes the whole dataset. In other words, we seek for a methodology to sample those sub-trajectories from a MOD that preserve as much as possible the properties and the mobility patterns hidden in the original MOD. Consider, for example, the domain of visual analytics on movement data [19] in which it is meaningless to visualize datasets larger than a certain small size, due to the distinguishing properties of the human eye. How would one select a subset so that this would cover the whole space-time and, as such, get the gist of the whole dataset is the challenge set in this paper.

The contributions and merits of our work are summarized below:

1) We propose an index-based global voting method that allows us to represent the representativeness of a trajectory in a MOD as a smooth continuous descriptor.
2) We introduce an algorithm for the automatic segmentation of trajectories into “homogenous” sub-trajectories according to their “representativeness” in the MOD.
3) We define the problem of sub-trajectory sampling in a MOD as an optimization problem and we propose a novel solution to tackle the problem.
4) Finally, we conduct a comprehensive set of experiments over synthetic and real trajectory datasets, in order to thoroughly evaluate our approach.

The rest of the paper is organized as follows: Section 2 presents previous work in related domains, while Section 3 sets the scene by presenting the various aspects of the problem and also describing the base of our further developments. Section 4 presents the proposed three-step methodology, i.e. our trajectory voting scheme, the trajectory segmentation technique, and the sub-trajectory sampling method. The experimental results are given in Section 5. Finally, conclusions and discussion are provided in Section 6.

2 Related Work

In this section, we review existing works in the domains related with the current work. In our setting, representative (sub)-trajectories are a new type of mobility pattern, as such, our discussion includes trajectory pattern mining, segmentation and sampling in MOD.

A MOD consists of spatiotemporal trajectories of moving objects (e.g. humans, vehicles, animals, etc.). In the general case, trajectories are represented as 3-D sequences where each recording encodes the 2-D (two dimensional) geographic location and the 1-D temporal information of mobile objects. During the last decade several approaches have been proposed in the literature so as to enable well-known mining algorithms to operate on trajectories. One such approach is the use of different

hal-00585610, version 1 - 13 Apr 2011

hal-00585610, version 1 - 13 Apr 2011

types of distance functions as the mean to group trajectories into clusters. Some approaches are inspired by the time series analysis domain [20], [7], while other exploit on a set of distance operators based on primitive (space and time) as well as derived parameters of trajectories (speed and direction) [9]. An interesting approach also used in our approach is proposed in [21] for the efficient processing of most-similar trajectory (MST) queries. A similar distance function is used in [8], where Nanni and Pedreschi adapt the well-known density-based OPTICS [22] clustering algorithm, tailored to work with point data, to a new algorithm for trajectory data, named T-OPTICS (and its varriant TF-OPTICS, which focuses on the discovery of the temporal intervals that lead to best clustering results). The previously mentioned temporal intervals are given by the user, so TF-OPTICS essentially re-executes T-OPTICS on segments of trajectories, obtained by properly clipping the original ones. In comparison with our approach, we automatically segment trajectories to portions based on global criteria (i.e. the representativeness of the trajectory in the MOD). Furthermore, TF-OPTICS mainly clusters whole trajectories and is not tailored to identify patterns of sub-trajectories in an un-supervised way.

Recently, Pelekis et al. [12] proposed another approach, called CenTR-I-FCM, taking into advantage of local patterns in time dimension as the base to identify global clusters of whole approximate / symbolic trajectories. In comparison with the current work, this approach also utilizes a global but static and predefined temporal segmentation of trajectories. In addition, in this work trajectories are symbolically represented as intuitionistic fuzzy vectors and not as sequences of 3-D line segments. This approach also aims at clustering trajectories as a whole with special care for handling uncertainty.

Authors in [4] proposed TRACLUS, a partition-and-group framework for clustering 2-D trajectories which enables the discovery of common sub-trajectories, based on a trajectory partitioning algorithm that uses the minimum description length principle. The core of the framework uses a variant of the DBSCAN algorithm that operates on the partitioned directed line segments. Finally, the notion of the *representative trajectory* of a cluster is defined. The fundamental differences of TRACLUS with our approach is that, a) the temporal information is not considered in [4], b) the segmentation is performed per trajectory and it does not use global criteria, c) the identified clusters of segments conform to straight movement patterns and cannot identify complex (e.g. snake-like) patterns, which are usual in real world applications, and d) the proposed algorithm for identifying the representative trajectory is defined per cluster and it is a synthetic trajectory computed by an averaging technique. Instead, we automatically select the top-k most representative sub-trajectories from the given dataset.

Other related works include techniques for *flock patterns* [10], *moving clusters* [11] and *convoys* [18]. Although these approaches provide lucid definitions of the mined patterns, they are rather rigorous and sensitive to parameters, while they are not in the context of automatic trajectory segmentation according to the representativeness of the trajectories, as well as sub-trajectory sampling.

In [23], *hot motion paths* (frequently traveled trails of numerous moving objects) are detected in a distributed system under the assumption that the moving objects can communicate with a central unit (coordinator). The goal of this work is in the same context with only one of the aims of our research (i.e. the one concerning trajectory representativeness). However, this approach focuses on the online processing and maintenance of hot motion paths by imposing a sliding time window of size W, which excludes from consideration any locations received more than W time units ago.

In [6], distance-based criteria have been proposed for the segmentation of trajectories. The distance between two trajectories is defined using their MBR representation. The segmentation problem is designed as a maximization problem, that attempts to create MBRs in such a way that the original pairwise distances between trajectories are minimized. We also provide a method for trajectory segmentation having global criteria but this splitting is entailed by the "representativeness" of the trajectory, which is not simply defined in terms of distance but via a novel voting method.

In [24], we have proposed an approach for expressing the "representativeness" in MOD via a voting process that is applied for each segment of a given trajectory. Moreover, a simplistic trajectory ranking method selects the trajectories of highest voting and as such ignores trajectories in low density regions of the MOD. In the current work, we improve the voting scheme originally proposed in [24] with the use of a metric distance function that further allows us to propose an index-based algorithm for the efficient implementation of the voting scheme. More importantly, we improve [24] by handling trajectory segmentation and sub-trajectory sampling aspects.

Trajectory sampling is a new topic in the spatiotemporal database literature with a variety of applications including MOD summarization, visualization, searching and retrieval. Although very interesting as a problem and with great potentials in the MOD domain, to the best of our knowledge there is no related work that tackles the problem in a tailored way to the complex nature of trajectory datasets. In [13], [14] the authors propose techniques for density biased sampling (DBS) in point sets that obviously can not be applied to trajectory data. In our approach, we extend the idea of DBS in a way that density properties as well as the similarity of trajectories segments is taken into account in a way that allows us to effectively apply sampling in the MOD domain. In [17] we have proposed an approach for unservised trajectory sampling, however this work operates on approximate, symbolic trajectories (i.e. represented as sequences of cells of a spatio-temporal grid wherefrom the moving

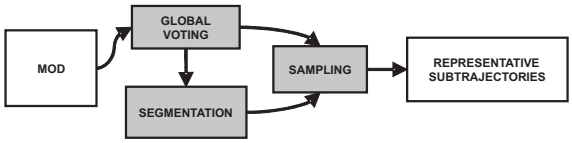


Fig. 2: Scheme of the proposed system architecture.

object pass), while it focuses on the selection of whole trajectories and not portions of them (i.e. sub-trajectories produced by a segmentation method). These two observations make the problem addressed in the current work completely different.

An interesting approach focusing on the visualization of large trajectories datasets and which uses standard sampling techniques has been proposed in [15], [16]. The authors use uniform sampling as a starting step to minimize the volume of the trajectories that can be clustered by the T-OPTICS algorithm [8]. At a subsequent step, they either select a certain percent of trajectories from a cluster using distance-based thresholds or they apply a partitioning on each cluster so as to identify spherical sub-clusters, each of which is represented by its medoid. Obviously, the outcome of this approach could be reached by directly applying K-Means or K-Medoid algorithm and then choosing the mean or medoid of each cluster as the representatives that will be sampled. Of course, this choice has the disadvantage that the number of clusters should be known apriori. This approach, which is basically a stratified sampling technique, where strata are produced by the clustering algorithm, has the limitation that it is user-supervised and it depends on the results of the clustering. Notwithstanding our methodology is automated, which is a self-evident advantage, in our experimental study we show the superiority of our approach in comparison to uniform random and stratified sampling techniques used in [15], [16], by using standard sampling evaluation metrics.

3 Setting the Scene

In this section, we set the scene of the various aspects of the problem that this paper addresses, and concurrently we present the stepping-stones where our subsequent developments base on. Let us assume a MOD $D = \{T_1, T_2, \cdots, T_N\}$ of N trajectories, where T_k denotes the k-th trajectory of the dataset, $k \in \{1, 2, ..., N\}$. We assume that the objects are moving in the xy plane. Let $p_k(i) = (x_k(i), y_k(i), t_k(i))$ be the i-th point, $i \in \{1, 2, ..., L_k\}$, of k-th trajectory, where L_k denotes the number of points of k-th trajectory. $x_k(i), y_k(i)$ and $t_k(i)$ denote the 2-D location and the time coordinate of point $p_k(i)$, respectively.

Similar to the work of [25], we consider linear interpolation between successive sampled points $p_k(i)$, $p_k(i+1)$, so that each trajectory consists of a sequence of 3-D line segments $e_k(i) = p_k(i)p_k(i+1)$, where each line segment represents the continuous moving of the object during sampled points.

Symbols	Definitions
D	Given MOD, $D = \{T_1, T_2, \cdots, T_N\}$
T_k	k^{-th} trajectory of MOD
L_k	Number of points of T_k
$p_k(i)$	i-point of T_k, $p_k(i) = (x_k(i), y_k(i), t_k(i))$
$x_k(i)$	x-spatial coordinate of $p_k(i)$
$y_k(i)$	y-spatial coordinate of $p_k(i)$
$t_k(i)$	time coordinate of $p_k(i)$
$e_k(i)$	3-D line segment, $e_k(i) = p_k(i)p_k(i+1)$
$l_k(i)$	Lifespan of $e_k(i)$, $l_k(i) = t_k(i+1) - t_k(i)$
V_k	Voting trajectory descriptor of T_k
LP_k	Number of T_k sub-trajectories
$P_k(i)$	$i-th$ sub-trajectory of T_k
$VP_k(i)$	Voting descriptor of $i-th$ sub-trajectory of T_k in D
$Nl_k(i)$	Normalized lifespan vector of $i-th$ sub-traj. of T_k in D
S	Sub-trajectory sampling set of D
$\widetilde{VP}_k(i)$	Voting descriptor of $i-th$ sub-traj. of T_k in S
$SR(S)$	Representativeness function of S

TABLE 1: Symbol table.

The goal of this work include:

- the automatic segmentation of the given trajectories T_k, into "homogenous" sub-trajectories according to their "representativeness" in MOD and
- sampling of the most representative sub-trajectories of the MOD.

Fig. 2 illustrates a scheme of the proposed system architecture. The following three sections formalize the issues of trajectory representativeness, trajectory segmentation, and sub-trajectory sampling, respectively. Table 1 summarizes the symbols' definitions used in this work.

3.1 Trajectory line segment representativeness

In this research, the "representativeness" in MOD is defined by extending the definition of density biased sampling (DBS) in point sets [13] for trajectory segments. According to DBS, the local density for each point of the set is approximated by the number of points in a region, divided by the volume of the region. In our case, the "representativeness" of a trajectory segment is defined by the number of the objects that follow this segment along with time, space and direction.

Technically, "representativeness" is calculated by a voting process that is applied for each segment $e_k(i)$ of the given trajectory T_k, improving a preliminary version of the proposed method, presented in [24]. Thus, $e_k(i)$ will be voted by the trajectories of MOD, according to the distance of $e_k(i)$ to each trajectory. The sum of these votes is related to the number of trajectories that are close and similar to $e_k(i)$, having the number of trajectories in MOD as upper limit. We avoid to give each segment the ability of voting, because, in such a case, long trajectories moving around the same area could vote many times. Moreover, under this definition, voting has the physical meaning of how many objects co-move (i.e. co-location and co-existence) for a period of time. Thus, the voting results will be used to detect the representative paths and sub-trajectories.

The previous discussion implies that we have to compute the distance $(d(e_k(i), e_m(j)))$ between two 3-D

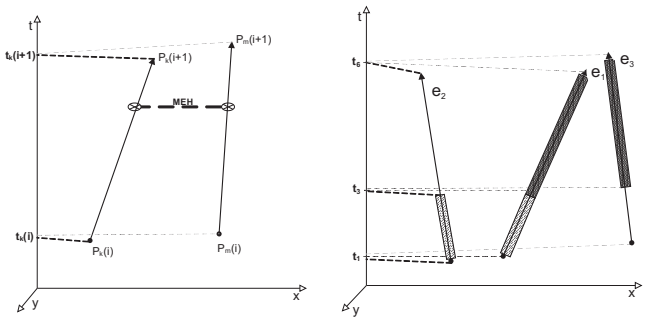
hal-00585610, version 1 - 13 Apr 2011

hal-00585610, version 1 - 13 Apr 2011

line segments. More specifically, we need to be able to identify the trajectories that are the closest (i.e. Nearest Neighbors - NNs) to a 3-D line segment, and this process should be done in an efficient and preferably in an incremental way, in the sense that the k-th NN can be obtained with very little additional work once the (k-1)-th NN has been found. These requirements imply that data should be indexed and, as such, the distance function should be obey the metric properties. To meet the above prerequisites, in this framework we use and extend the approach for incremental continuous NN search algorithms proposed by Frentzos et al. in [26]. In this approach, which assumes that data are indexed by one of the two following members of the R-tree family for trajectory data (namely, the TB-tree [25] or the 3D-R-tree [27]), the distance between 3-D line segments is defined as their *Minimum Euclidean Horizontal* distance (MEH, for short) during their common lifespan (see Fig. 3(a))[1]. As proved in [26], the Euclidean horizontal distance function between two 3-D line segments follows the quadratic form $D(t) = \sqrt{at^2 + bt + c}$, where a, b, c are the factors of this trinomial (real numbers, $a \geq 0$), and whose minimum is a closed formula that can be computed in $O(1)$. Note that, by simply defining the distance between two 3-D line segments using MEH distance, will make the distance function dependable to a single value and not to the whole lifespan of the segments and, as such, very sensitive to small variations. As a counter-example consider the case where two objects move on the same road segment following different directions. At some timepoint they will meet, so their MEH will be (very close to) zero, but obviously this is not representative of their co-motion. As in our approach we aim at transforming trajectories into 1-D continuous descriptors that will smoothly describe the $representativeness$ of the trajectories in the database, a more consolidated approach for measuring the distance between 3-D segments is needed.

Intuitively, an ultimate distance function would make use of the fact that for a single 3-D segment, different portions of it may have different NNs of other segments, and the sizes of these portions are taken into account. To this end, from the palette of algorithms proposed in [26] we base on the best-first historical continuous trajectory NN $IncTrajectoryNNSearch$ algorithm ($HCNN\_Q_T$, for short). The $HCNN\_Q_T(D, Q_T, Q_{per})$ query over the query moving object with trajectory Q_T during a time period Q_{per} returns a list of triplets for each of the k-NN, i.e. $LoT_{kNN} =< \cdots, (R_j, T_j, [t_{j-start}, t_{j-end})), \cdots >$ consisting of the time-varying real value R_j along with a moving object T_j (belonging in database D) and the corresponding time period $[t_{j-start}, t_{j-end})$ for which the nearest distance between Q_T and T_j stands. These time-varying real values R_j are, in any time instance of their lifetime, smaller or equal to the distance between any moving object in D and the query trajectory Q_T. The time periods $[t_{j-start}, t_{j-end})$ are mutually disjoint and their union forms Q_{per}. With e_j we denote the portion of the segment that corresponds to each triplet in LoT_{kNN}.

1. If there does not exist common lifespan, the distance is given by the diameter of the dataset.



(a) (b)

Fig. 3: **(a)** The distance between 3-D line segments is defined as their *Minimum Euclidean Horizontal* distance during their common lifespan. **(b)** The 2-NN of the query segment e_1.

Recall that, in our case, we want to apply such a NN query for each segment $e_k(i)$ of a trajectory, so following the notation of the $HCNN\_Q_T$ query, the Q_T is the $e_k(i)$ and Q_{per} becomes the respective temporal period of $l_k(i)$. To exemplify the proposed k-NN, in Fig. 3(b) we depict the 2-NN of the query segment e_1. The 1-NN list includes e_2 for the interval $[t_1, t_3)$ and e_3 for the interval $[t_3, t_6]$; the 2-NN list includes e_3 for the interval $[t_1, t_3)$ and e_1 for the interval $[t_3, t_6]$.

Given a segment $e_k(i)$ of a trajectory we calculate its distance from the NN segments e_j provided by a LoT_{kNN}, by computing the definite integral of the time-varying distance R_j between the two segments during the same period, following the approach proposed by Frentzos et al. in [21]:

$$d(e_k(i), e_j) = \int_{t_{j-start}}^{t_{j-end}} R_j(t)dt \quad (1)$$

As R_j follows the trinomial previously discussed, and as proved in [21] its integral can be computed in $O(1)$ by distinguishing between two cases for the value of the non-negative factor a ($a = 0$ and $a > 0$). As also proved in [21], $d(e_k(i), e_j)$ can be efficiently computed by approximating the integral with the Trapezoid Rule ($\frac{(R_j(t_{j-start}) + R_j(t_{j-end})) \cdot (t_{j-end} - t_{j-start})}{2}$), providing also bounds for the approximation error.

Given the previous setting, let V_k of $L_k - 1$ components be considered the voting trajectory descriptor ("representativeness" value) along the T_k line segments, computed by using the previous distance function. Each component of this signal $V_k(i)$ corresponds to the number of votes of $e_k(i), i \in \{1, 2, ..., L_k - 1\}$ of T_k. Section 4.1 describes the proposed global voting method that estimates $V_k(i)$.

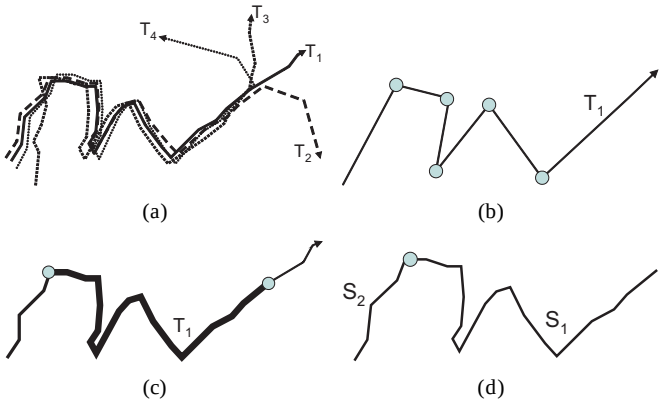


Fig. 4: **(a)** A MOD of four trajectories. **(b)** Segmentation of T_1 using the TRACLUS method (six sub-trajectories). **(c)** Segmentation of T_1 using the proposed method (three sub-trajectories). **(d)** The two most representative sub-trajectories according to the proposed method.

3.2 Trajectory Segmentation

In this section, the trajectory segmentation is presented. The goal of trajectory segmentation is a trajectory partitioning into sub-trajectories of homogenous representativeness irrespectively of their shape complexity.

Let VP_k of LP_k partitions be considered the voting sub-trajectory representativeness of T_k trajectory. Let $P_k(i)$, $i \in \{1, ..., LP_k\}$ be the i-th sub-trajectory of T_k, where LP_k denotes the number of partitions of T_k. Then according to previous definition, $VP_k(i)$ is the vector (descriptor) of votes along the line segments of $P_k(i)$ partition, showing how many objects follow each line segment of sub-trajectory. This value will turn out to be useful for sampling purposes, as it will be presented in the section that follows.

Fig. 4 illustrates an example of a MOD comprised by four trajectories ($\{T_1, ..., T_4\}$). In this figure, the time dimension has been ignored for visualization reasons. The segmentation of T_1 according to the TRACLUS method [4] and the proposed method are illustrated in Figs. 4(b) and 4(c), respectively. Indeed, most of the proposed techniques in literature like TRACLUS first simplify the trajectories into large line segments and then apply a grouping of similar of line segments, see six sub-trajectories on Fig. 4(b). On the contrary, according to our approach, the segmentation is applied on the original trajectory by taking into account its neighborhood in the rest of the MOD. In addition, there is not any constraint on sub-trajectory shape complexity, yielding a segmentation that is related only on line segment representativeness (voting results), see three sub-trajectories on Fig. 4(c). In Fig. 4(c), the line thickness is related with the line segments representativeness (voting results). It holds that the sub-trajectory representativeness values will be almost the same along the line segments of each sub-trajectory (see Fig. 4(c)). According to the proposed methodology, the top-2 representative sub-trajectories ($\{S_1, S_2\}$) that best describe the MOD are illustrated in Fig. 4(d).

3.3 Subtrajectory Sampling

The trajectory segmentation provides homogenous sub-trajectories concerning their representativeness. The goal of sub-trajectory sampling is to select the most representative subset of these sub-trajectories. By selecting the higher voting segments as in [24], which sounds to be an obvious decision, the high density regions of the MOD will be oversampled, resulting in a non-representative sample when the aim is to cover the whole space-time of the MOD as much as possible.

On the contrary, in this paper, we propose the sampling to be done by minimizing a formula (see Equation 3), taking into account the votes (i.e. representativeness) $VP_k(i)$ in the original MOD as well as the votes $\widehat{VP}_k(i)$ in the sampling MOD and the vector of the lifespans of the line segments divided by the lifespan of the trajectory ($Nl_k(i)$). $\widehat{VP}_k(i)$ is computed in sampling MOD using the voting method similar to $VP_k(i)$ computation in the original MOD. Thus, it takes into account the fact that the sampling set should contain dissimilar representative sub-trajectories. So, our goal is that the sampling set should contain high voting trajectories of the MOD and, at the same time, should cover the whole 3-D space as much as possible. This means that the resulting set should avoid to contain similar sub-trajectories.

Let S denote the sampling set, so that $S_k(i)$ is one, if $P_k(i)$ sub-trajectory belongs to the sampling set, and zero otherwise. According to the previous properties, the number of moving objects of the original MOD that are represented in sampling set $SR(S)$, should be maximized over the time. This is formalized in Equations 2 and 3.

$$SR_{gain}(k,i) = \sum_{j=1}^{|P_k(i)|} VP_k(i)(j) \cdot Nl_k(i)(j) \cdot (1 - \widehat{VP}_k(i)(j)) \quad (2)$$

$$SR(S) = \sum_{k=1}^{N} \sum_{i=1}^{LP_k} S_k(i) \cdot SR_{gain}(k,i) \quad (3)$$

$SR_{gain}(k,i)$ expresses the gain of $SR(S)$ if we add in sampling set the i^{th} sub-trajectory of the k^{th} trajectory of the MOD. $|P_k(i)|$ denotes the number of line segments of the $i-th$ sub-trajectory of T_k. The values $VP_k(i)(j)$, $Nl_k(i)(j)$ and $\widehat{VP}_k(i)(j)$ denote for the votes in D, the normalized lifespan and the votes in S of the j^{th} line segment of i^{th} sub-trajectory of k^{th} trajectory of the MOD, respectively. In other words, the number of objects of the MOD that follows the sub-trajectories of S is maximized over the time. Fig. 4(d) illustrates the top-2 representative sub-trajectories S_1 and S_2 according to the proposed method (S_1 is the top-1st since it represents all four objects, while S_2 is the top-2nd since it represents two objects. Let us suppose that the normalized lifespan of S_1 is $\frac{1}{2}$ and the normalized lifespan of S_2 is $\frac{1}{3}$. It

hal-00585610, version 1 - 13 Apr 2011

hal-00585610, version 1 - 13 Apr 2011

holds that S_1 will be selected first, since the SR_{gain} of S_1 is about $\frac{3}{2}$ (see Equation 2). S_2 will be selected next, since the SR_{gain} of S_2 is about $\frac{2}{3}$ (see Equation 2). In both of the cases, the corresponding $\widehat{VP}_k(i)(j)$ is zero, since S_1 and S_2 are defined into different time periods. Next, the SR_{gain} of other sub-trajectories will be zero or a negative value. Therefore, the proposed sampling method will select these two sub-trajectories, covering the time and space of the MOD.

4 METHODOLOGY

In this section, the proposed methodology is presented, consisting of the trajectory voting, the trajectory segmentation and the sub-trajectory sampling methods.

4.1 Global Voting Method

This section describes the Global Voting Algorithm (GVA). The input of the algorithm is a MOD $D = \{T_1, T_2, \cdots, T_N\}$ indexed by a R-tree-like structure such as the TB-tree [25] or the 3D-R-tree [27], as described in [26], a trajectory $T_k \in D$ and an intrinsic parameter $\sigma > 0$ of the method. The output of the method is the vector V_k of $L_k - 1$ components that can be considered as a trajectory descriptor along the line segments $e_k(i), i \in \{1, 2, ..., L_k - 1\}$ of trajectory T_k (recall from Section 3, that L_k denotes the number of points of T_k trajectory). As such, each component of the vector $V_k(i)$ corresponds to the number of votes (representativeness) for each $e_k(i)$ of T_k.

According to the problem formulation presented in Section 3, for each line segment $e_k(i)$ of T_k, the proposed GVA algorithm incrementally identifies the NN segments of other trajectories $T_j \in D, j \neq k$. For each set of NN segments represented by the list of segments/triplets in LoT_{kNN}, the distances $d(e_k(i), e_j)$ from the corresponding segments are computed. These distances are used to define the voting function $V(e_k(i), LoT_{kNN})$, which quantifies the $representativeness$ of the line segment to a LoT_{kNN}. In the literature, a lot of voting functions have been proposed, like step functions or continuous functions [28]. In this work, we have selected to use the continuous function of a gaussian kernel, which is widely used in a variety of applications of pattern recognition [29]. Formally,

$$V(e_k(i), LoT_{kNN}) = \sum_{\forall e_j \in LoT_{kNN}, T_k \neq T_j} e^{-\frac{d^2(e_k(i), e_j)}{2 \cdot \sigma^2}} \quad (4)$$

Note that for data collected from GPS devices where the segments are very small (due to the high sampling rate), in practice the previous function degenerates to the computation of a single gaussian kernel, as the HCNN of a segment results in a LoT_{kNN} containing only one NN. This is actually verified in our experiments where we used real GPS datasets. However, in cases where the datasets are highly compressed (in applications where storage cost is important), e.g. by an approach like the one proposed in [30], this may have an influence in the smoothness of the $V_k(i)$ descriptor. We leave such a study as future work. The control parameter $\sigma > 0$ shows how fast the function ("voting influence") decreases with distance. Given the previous assumption, and according to Equation 4, it holds that $0 \leq V(e_k(i), LoT_{kNN}) \leq 1$. If $d(e_k(i), e_j)$ is close to zero, the voting function gets its maximum value, i.e. 1. This means, that there exists a line segment of T_j that is being (in time, space and direction) very close to $e_k(i)$. Otherwise, if $d(e_k(i), e_j)$ is high, e.g. greater than $5 \cdot \sigma$, the voting function results in almost 0, meaning that T_j is very far away from $e_k(i)$.

The use of a continuous voting function, like the gaussian kernel, gives smooth results for small changes on parameters (σ in our case), and the possibility to get decimal values as results of voting process increasing the robustness of the method. However, σ depends on space units the object movements of MOD and it is difficult to tune it. We have solved this problem by estimating σ as the percentage (e.g. 0.1%) of dataset diameter (maximum space distance). This percentage can be kept almost constant for every dataset. Finally, $V_k(i)$ is computed by getting the sum of votes for all of the nearest neighbor segments of trajectories $T_j \in D, j \neq k$, according to GVA (see Algorithm 1).

input : An (indexed) database $D = \{T_1, T_2, \cdots, T_N\}$, a trajectory T_k in D and the parameters σ, ε.
output: Voting vector V_k of T_k

```
1  for i = 1 to L_k − 1 do
2      V_k(i) = 0
3      repeat
4          LoT_kNN = Get_Next_HCNN_Q_T(D, e_k(i), l_k(i))
5          V(e_k(i), LoT_kNN) = Σ_{∀e_j∈LoT_kNN, T_k≠T_j} e^{−d²(e_k(i),e_j)/(2·σ²)}
6          if V(e_k(i), LoT_kNN) > ε then
7              V_k(i) = V_k(i) + V(e_k(i), LoT_kNN)
8          else
9              break
10         end
11     until LoT_kNN = ∅
12 end
```

Algorithm 1: Global Voting Algorithm (GVA).

Note that for each $e_k(i)$ (line 1) the algorithm starts a loop (loop 3) wherein each time it incrementally retrieves the next LoT_{kNN} (line 5, function Get_Next_HCNN), for which its $representativeness$ is computed. This measure is added to the $V_k(i)$ if the current voting is significant (i.e. larger than a small value (ε is a very small number comparing to one, e.g. $\varepsilon = 10^{-3}$), lines 6,7), otherwise the loop is terminated (line 9), which means that we stop retrieving subsequent NN. The loop

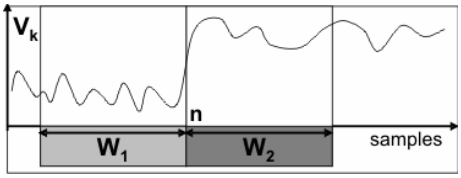


Fig. 5: The two sequential sliding windows W_1 (light gray horizontal lines) and W_2 (heavy gray vertical lines) locating at sample n on the given voting signal V_k.

is also terminated if there are no other NN given the spatiotemporal criteria of $e_k(i)$ (line 11). Given the above discussion, a nice property that holds for the proposed local trajectory descriptor V_k is that it smoothly changes over the trajectory segments.

4.2 Trajectory Segmentation Method

Having presented the voting procedure in the previous section, the next step is to provide a solution to the trajectory segmentation problem defined in section 3.2. For this purpose we propose the Trajectory Segmentation Algorithm (TSA) (see Algorithm 2). The input of the algorithm is the normalized trajectory voting signal V_k, and two intrinsic parameters w, τ of the method. The normalization is done by dividing V_k by the maximum over all V_k, thus bounding $V_k \leq 1$. The output of the method is the segmentation P_k of T_k into LP_k partitions, where LP_k is automatically estimated by the proposed scheme.

The method uses two sequential sliding signal windows W_1 and W_2 of w samples estimating the sample when the "difference" between the two windows is maximized.

This methodology has been successfully applied on sound signal segmentation [31] and P Phase Picking of seismic signals [32]. To facilitate the discussion, Fig. 5 illustrates the two sequential sliding windows W_1 (light gray horizontal lines) and W_2 (heavy gray vertical lines) locating at sample n on the given voting signal V_k. First, as the two windows slide, the two means m_1, m_2 and two variances σ_1^2, σ_2^2 of two sequential signal windows and W_1, W_2 locating at sample n are estimated, respectively (see Fig. 5) (see lines 1-7 of Algorithm 2). The next equations define m_1 and σ_1^2; m_2 and σ_2^2 are similarly defined in window W_2.

$$m_1 = \frac{1}{w}\sum_{i\in W_1} V_k(i) \quad (5)$$

$$\sigma_1^2 = \frac{1}{w}\sum_{i\in W_1} (V_k(i) - m_1)^2 \quad (6)$$

The symmetric Mahalanobis Distance [32], [33], presented in Equation 7, is used to measure the distance between the two windows locating at sample n.

$$d(n) = (m_1 - m_2)^2 \cdot (\frac{1}{\sigma_1^2} + \frac{1}{\sigma_2^2}) \quad (7)$$

The symmetric Mahalanobis Distance has been selected since it outperforms other frequently used distances like symmetric Kullback-Leibler (KL2) or Bhattacharyya [34] in segmentation problem, which has been shown in [32].

A sample n is characterized as border point (start of a new partition),

1) if $d(n) > \tau$ and
2) $d(n) \geq d(i),\ i \in \{n-w, ..., n+w\}$ (see lines 12-17 of Algorithm 2).

The first statement ensures that the difference of two windows W_1, W_2 will be high enough while the second selects the sample n, where $d(n)$ is locally maximized, meaning that the difference of two windows W_1, W_2 is locally the highest. Both statements are related with the number of partitions LP_k, while the second ensures that the minimum partition size is w samples (w 3-D line segments) [32].

Parameter w sets the minimum size of a partition, so w depends on the given dataset and the user preferences. In other words, w expresses the minimum number of line segments that can define a sub-trajectory. In addition, w is analogous to sampling rate of the trajectories (e.g. if a MOD has double sampling rate, then w should be multiplied by two). TSA needs to estimate mean and variance measures, thus we need at least two samples ($w \geq 2$). Low values on w can affect the robustness of mean and variance estimation yielding false alarms (over-segmentation). High values on w gives more robust results, and it will affect the results of the method, only if there is a sub-trajectory with length less than w that will not be detected. According to our experiments, when $w \in [5, 15]$ most of sub-trajectories were robustly detected without important false alarms.

Regarding τ, it should be a positive number close to zero, in order to be sure that TSA will detect all the sub-trajectories. According to our experiments, when $\tau \in [0.001, 0.1]$ most of sub-trajectories were detected without important false alarms, since this parameter is related with the segmentation sensitivity of our method. As τ increases, the number of sub-trajectories reduces. It holds that τ can be set as a positive number close to zero (e.g. 0.01), due to the fact that in the first step, we perform normalization by dividing V_k by the maximum over all V_k, thus bounding $V_k \leq 1$.

4.3 Subtrajectory Sampling

In the previous sections, we have presented our methodology for segmenting the trajectories of a MOD into sub-trajectories using the votes gathered for the MOD. In this section, we exploit on this knowledge in order to select the top representative sub-trajectories to be the result of a sampling process. In particular, we propose the Subtrajectory Sampling Algorithm (SSA). SSA solves the sampling problem defined in section 3.3 (see Algorithm 3). The input of the algorithm is the set of sub-trajectories of the MOD P_k as estimated by TSA, the voting $VP_k(i)$

hal-00585610, version 1 - 13 Apr 2011

hal-00585610, version 1 - 13 Apr 2011

input : A normalized trajectory voting vector V_k and two parameters w and τ.
output: The segmentation P_k of LP_k partitions.

1 **for** $n = w + 1$ **to** $L_k - w - 1$ **do**
2 $\quad m_1 = \frac{1}{w} \sum_{i=n-w}^{n-1} V_k(i)$
3 $\quad m_2 = \frac{1}{w} \sum_{i=n}^{n-w-1} V_k(i)$
4 $\quad \sigma_1^2 = \frac{1}{w} \sum_{i=n-w}^{n-1} (V_k(i) - m_1)^2$
5 $\quad \sigma_2^2 = \frac{1}{w} \sum_{i=n}^{n-w-1} (V_k(i) - m_2)^2$
6 $\quad d(n) = (m_1 - m_2)^2 \cdot (\frac{1}{\sigma_1^2} + \frac{1}{\sigma_2^2})$
7 **end**
8 $LP_k = 1$, $a = 1$
9 $P_k(1) = [a : L_k - 1]$
10 **for** $n = w + 1$ **to** $L_k - w - 1$ **do**
11 $\quad m = max_{i \in \{n-w,\ldots,n+w\}} d(i)$
12 $\quad$ **if** $d(n) > \tau \wedge d(n) \geq m$ **then**
13 $\quad\quad P_k(LP_k) = [a : n - 1]$
14 $\quad\quad P_k(LP_k + 1) = [n : L_k - 1]$
15 $\quad\quad a = n$
16 $\quad\quad LP_k = LP_k + 1$
17 $\quad$ **end**
18 **end**

Algorithm 2: Trajectory Segmentation Algorithm (TSA).

and the normalized lifespan $Nl_k(i)$ vectors of the trajectory segments. The output of the method is the sub-trajectory sampling set S consisting of M samples. M can be given as input to the method or (more interestingly) it can be automatically estimated by the proposed scheme.

The goal of SSA is the maximization of the number of sub-trajectories $SR(S)$ of the original MOD that are represented in the sampling set (see Equation 3). The complexity of an exhaustive algorithm that would search for all the possible solutions in order to maximize Equation 3 is $O(\binom{N}{M})$. On the other hand, our proposed algorithm suboptimally solves the problem in $O(N \cdot M)$ iterations by applying iterative optimization.

SSA starts with an empty sampling set ($S_k(i) = 0$), where $S_k(i)$ is defined in section 3.3. In each iteration step, SSA adds in sampling set an unselected sub-trajectory of MOD that maximizes Equation 3. This is equivalent with the maximization of $SR(S)$ gain $SR_{gain}(k,i)$ (see Equation 2). Recall that $SR_{gain}(k,i)$ expresses the gain of $SR(S)$ if we add in sampling set the i^{th} sub-trajectory of k^{th} trajectory of the MOD. According to the proposed algorithm, it holds that $SR(S)$ gain is a monotonically decreasing function as sampling size increases. Since $\widehat{VP_k}(i)(j) \geq 0$, and recalling Equation 2, it holds that:

$$SR_{gain}(k,i) \leq \sum_{j=1}^{|P_k(i)|} VP_k(i)(j) \cdot Nl_k(i)(j) \qquad (8)$$

Let $SR_{gain\ zero}(k,i)$ denotes the sum of Equation 8, that corresponds to the $SR_{gain}(k,i)$ when $S = \emptyset$.

$$SR_{gain\ zero}(k,i) = \sum_{j=1}^{|P_k(i)|} VP_k(i)(j) \cdot Nl_k(i)(j) \qquad (9)$$

Therefore, an efficient way to reduce the computation cost of the maximization of SR_{gain}, is to keep the sub-trajectories' indexes in a vector (vec) sorted in descending order by $SR_{gain\ zero}$ (line 6 of Algorithm 3). Instead of computing SR_{gain} for each sub-trajectory of the MOD in order to find the maximum, we get the $j^{th}, j \in \{1, 2, ..., \sum_{k=1}^{N} LP_k\}$ (line 9 of Algorithm 3) sub-trajectory from the vector and we compare its voting with current highest SR_{gain} (SR_{gain}^{max}) (lines 11-15 of Algorithm 3). First, SR_{gain}^{max} is set to minus infinity (line 8 of Algorithm 3). Let the j^{th} sub-trajectory of vec be the i^{th} sub-trajectory of k^{th} trajectory of the MOD (line 10 of Algorithm 3). This loop terminates, if SR_{gain}^{max} will be higher than the $VP_k(i)$, because it holds that the SR_{gain} of each sub-trajectory from the rest sub-trajectories from the list would be lower than its $SR_{gain\ zero}$ (see Equation 8) and lower than SR_{gain}^{max}, since we sort them in descending order by $SR_{gain\ zero}$. Then, the v^{th} sub-trajectory of u^{th} trajectory of the MOD that corresponds to SR_{gain}^{max} is added on sampling set (line 18).

SSA terminates when the size of the sampling set reaches M (which is an input parameter). Alternatively, the algorithm terminates if $SR_{gain}(k,i)$ is lower than a given threshold (ϵ, a positive number close to zero, see line 19 of Algorithm 3). If this threshold is set to zero, it means that $SR(S)$ has been maximized, as the latter is increased in each step by SR_{gain}^{max}. An advantage of the proposed method is that it provides a deterministic solution in contrary with other probabilistic techniques [13], [14] that provide a randomly constructed sampling set trying to fit it to a desired distribution or user-supervised, explorative approaches [15], [16] that base on clustering, which in turn depends on the algorithm itself, its parameters and the distance function adopted.

4.4 Computational Complexity Issues

Concerning the complexity of GVA, and given the use of the R-tree-like structures, the computational cost for each line segment $e_k(i)$, of T_k is $O(log(\bar{L} \cdot N))$, where $\bar{L}$ denotes the mean number of trajectory points. Executing GVA for each trajectory of the database, the total computation cost is $O(\bar{L} \cdot N \cdot log(\bar{L} \cdot N))$.

Concerning the complexity of TSA, the computational cost for the segmentation of a trajectory T_k is $O(L_k)$, since the mean and the variance of a sliding window can be estimated recursively in $O(1)$ by the mean and the variance of the sliding window of the previous step. For example, let m_1 be the mean of the window $W_1 = V_k(n-W : n-1)$ and $\acute{m}_1$ be the mean of the next window $\acute{W}_1 = V_k(n - W + 1 : n)$. Then, it holds that

$$\acute{m}_1 = \frac{m_1 \cdot w - V_k(n - W) + V_k(n)}{w} \qquad (10)$$

hal-00585610, version 1 - 13 Apr 2011

input : The voting vector $VP_k(i)$, the normalized lifespan vector $Nl_k(i)$ and the segmentation $P_k(i)$, $\forall k \in \{1, ..., N\}, i \in \{1, ..., LP_k\}$, ϵ. The size of sampling set M.
output: The sampling set $S_k(i)$.

1 **for** $k = 1$ **to** n **do**
2 **for** $i = 1$ **to** LP_k **do**
3 $S_k(i) = 0$ //Empty sampling set
4 **end**
5 **end**
6 vec = sort(VP,dt) //sorted in descending order
7 **for** $m = 1$ **to** M **do**
8 $SR_{gain}^{max} = -\infty$
9 **for** $j = 1$ **to** $\sum_{k=1}^{N} LP_k$ **do**
10 [k,i] = vec(j)
11 **if** $S_k(i) = 0 \wedge SR_{gain}(k, i) > SR_{gain}^{max}$ **then**
12 $SR_{gain}^{max} = SR_{gain}(k, i)$
13 $u = k$
14 $v = i$
15 **end**
16 **if** $SR_{gain}^{max} > VP_k(i)$ **then** break
17 **end**
18 $S_u(v) = 1$
19 **if** $SR_{gain}^{max} \leq \epsilon$ **then** break
20 **end**

Algorithm 3: Subtrajectory Sampling Algorithm (SSA).

If we perform TSA for each trajectory of the MOD, then the total computation cost is $O(\bar{L} \cdot N)$.

Concerning the complexity of SSA, the sorting of VP costs $O(N \cdot \frac{\bar{L}}{\overline{LP}} \cdot log(N \cdot \frac{\bar{L}}{\overline{LP}}))$, where $\overline{LP}$ denotes the mean number of trajectory segments and $\frac{\bar{L}}{\overline{LP}}$ denotes the mean number of sub-trajectory points. Next, in the worst case, the method computes $SR_{gain}(k, i)$ ($M \cdot \sum_{k=1}^{N} LP_k = M \cdot \overline{LP} \cdot N$) times. $SR_{gain}(k, i)$ computation requires the computation of $V_k(S)$. The cost of $V_k(S)$ is $O(\frac{\bar{L}}{\overline{LP}} \cdot log(\bar{L} \cdot M))$, since the maximum size of sampling set is M and the mean number of a sub-trajectory points is $\frac{\bar{L}}{\overline{LP}}$. Therefore, in the worst case, the overall cost of the proposed segmentation-and-sampling method is $O(N \cdot \frac{\bar{L}}{\overline{LP}} \cdot log(N \cdot \frac{\bar{L}}{\overline{LP}}) + \bar{L} \cdot M \cdot N \cdot log(\bar{L} \cdot M))$. In the best case, the break of line 16 of the algorithm can stop the intrinsic loop in two (O(1)) instead of $N \cdot \bar{L}$ steps. In this case, the cost turns out to be $\Omega(N \cdot \frac{\bar{L}}{\overline{LP}} \cdot log(N \cdot \frac{\bar{L}}{\overline{LP}}) + M \cdot log(\bar{L} \cdot M))$.

Conclusively, the most computationally intensive part of the proposed method is the GVA with $O(\bar{L} \cdot N \cdot log(\bar{L} \cdot N))$ complexity. In turn, the most time consuming step in GVA is the search of the nearest neighbors of a trajectory in a given time period. In order to make the application of our approach feasible to large datasets, we have adopted efficient Continuous Nearest Neighbor query processing techniques [26], where trajectories are indexed by R-tree-like structures. Scalability experiments under various cases for such queries have been presented in [26], where it has been shown their applicability in large datasets, with an almost linear behavior with the size of dataset. Actually, this conclusion is in accordance with the above theoretical analysis of the computational complexity of the proposed method.

4.5 On the effect of the MOD extension

As already mentioned, the proposed method is deterministic, which implies that different invocations for a given MOD will have the same result. This is a crucial and distinct characteristic of our approach w.r.t. other sampling approaches. In this section, we discuss the behavior of the proposed methods when the MOD is extended either in space or time dimensions. More specifically, we study the following scenario: In a given dataset S, we add trajectories which come from a different spatial or temporal space (extension of the MOD). The question is whether the GVA, TSA and SSA are affected by such an extension?

According to this scenario, the results of the voting procedure (GVA) will not change concerning the given dataset S. The new trajectories do not affect the trajectory descriptors of the trajectories of S, since they exist in different spatial or temporal space (see Equation 4). Therefore, the results of TSA will be exactly the same concerning the given dataset S. Similarly, the new sampling set will contain the same sub-trajectories as the sampling set of S (when the algorithm terminates if SR_{gain} is lower than a given threshold), with some additional samples selected from the new trajectories, since the input of the SSA algorithm concerning S remains the same. In other words, the sampling set of the union of two distinct (in space and/or in time dimension) MOD, is the same with the union of the sampling sets, when the sampling process is performed to each MOD independently. Therefore, the effectiveness of the proposed method is not affected by the extension of the MOD.

5 Experimental Results

In this section, the experimental results of our performance study are presented. In order to evaluate the effectiveness and robustness of the proposed scheme, we have performed two different experiments, with synthetic and real datasets. The implementation of the proposed algorithms was done in Matlab. A linear scan instead of an R-tree-based index scan was performed for HCNN search. However, this does not affect the effectiveness of our approach but only scalability. In other words, the results of GVA produce the same results either by using the R-tree-based index-based implementation, or by adopting a linear scan for the HCNN search.

In all of presented experiments, we have used $\tau = 0.01$ and $w = 8$. Similar results were obtained when we set w equal to a percentage (e.g. $20\% - 25\%$) of the mean number of line segments per trajectory. In addition, we

hal-00585610, version 1 - 13 Apr 2011

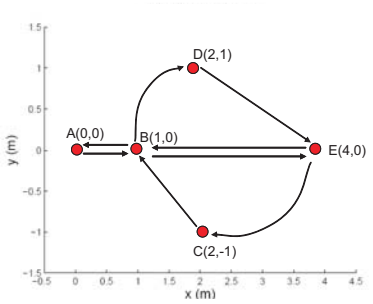


Fig. 6: The 2-D map of SMOD with the four one-directional and two bidirectional roads.

have run several experiments that show that the quality of sampling results not sensitive to those parameters. These parameters affect only the results of segmentation (e.g. when τ and w decrease, TSA will gives an oversegmentation of the MOD).

5.1 Experimenting with Synthetic Data

In order to measure the performance of segmentation and sampling for movements that can be created by vehicles, we have created a synthetic MOD (SMOD). Note that in order to evaluate our approach, well known synthetic trajectory generators [35] cannot be utilized as we would not be aware of the ground truth of the generated trajectories. As such, assume the following points $A(0,0)$, $B(1,0)$, $C(2,-1)$, $D(2,1)$ and $E(4,0)$ be the destination nodes of a simple graph upon which objects move. We assume that the half of the objects are moving with normal speed (5 units per second) and the rest of them are moving with high speed (10 units per second). The objects are moving under the following scenario (rules), for a lifetime of one second.

- There are four one-directional roads ($B \rightarrow D$, $D \rightarrow E$, $E \rightarrow C$, $C \rightarrow B$) and two bidirectional roads ($A \leftrightarrows B$, $B \leftrightarrows E$).
- At $t = 0$ sec, all objects of MOD start from (in a real application, very close to) point A. Thus, the first destination of all objects is point B.
- When an object arrives at a destination point, it ends its trajectory with a probability of 10%. Otherwise, it continues with the same speed to the next point according to the road network rules. If there exist more than one possible next points, it decides randomly the next destination.

As an example, if an object moves from A to B, then the next destination could be A, D or E with the same probability 30% (the rest 10% is the case where the object ends its trajectory at B). Fig. 6 illustrates the 2-D map of SMOD with the four one-directional and the two bidirectional roads.

According to the rules that the generated trajectories should obey, there exist 78 distinct sub-trajectories in the MOD (ground truth). Table 2 illustrates the spatial (first column) and temporal (second column) projection

Path	Time	Support
$A \rightarrow B$	$[0, 0.1]$	50%
$A \rightarrow B$	$[0, 0.2]$	50%
$B \rightarrow A$	$[0.2, 0.4]$	15%
$B \rightarrow A$	$[0.1, 0.2]$	15%
$B \rightarrow D$	$[0.1, 0.26]$	15%
$B \rightarrow D$	$[0.2, 0.52]$	15%
$B \rightarrow E$	$[0.1, 0.4]$	15%
$B \rightarrow E$	$[0.2, 0.8]$	15%

TABLE 2: Some representative sub-trajectories and their support in the synthetic MOD (ground truth).

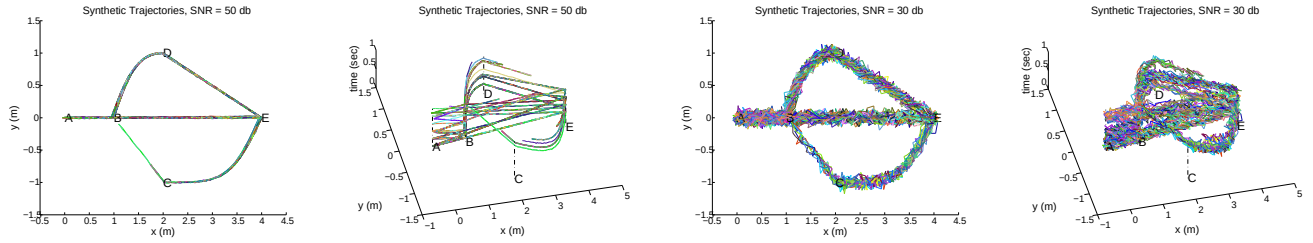
of the top-8 representative sub-trajectories, along with their support (third column), i.e. the average percentage of objects in the MOD that "vote" for the corresponding sub-trajectory. These percentage values have been computed under the aforementioned assumptions of the previous paragraph. The percentage of objects of MOD that follows a path could change between experiments due to random decisions of the prementioned scenario.

According to this synthetic MOD, the possible endpoints of each trajectory partition are points A, B, C, D, and E (ground truth). The above dataset is ideal for the purposes of experimentation: It consists of a high number of predefined sub-trajectories with each one being associated with a known representativeness (as ground truth).

In order to measure the stability of our method to noise effects, we have added Gaussian white noise of different Signal to Noise Ratio (SNR) levels, measured in db, to spatial coordinates of synthetic MOD. The synthetic MOD with additive noise of SNR = 50 db and SNR = 30 db projected in 2-D spatial and 3-D spatiotemporal space is illustrated in Fig. 7.

According to this synthetic MOD, there exist eight different paths that are defined by four one-directional roads and two bidirectional roads. We have used 600 moving objects with 1934 sub-trajectories. The mean value and the standard deviation for number of line segments per trajectory (L_k) are 82.03 and 30.95, respectively. The proposed GVA/TSA gives between 1800 and 2200 sub-trajectories under any case of additive noise. The average accuracy of trajectory segmentation method is measured using the average distance error (in spatial coordinates) between estimated segment borders and the ground truth borders. In order to get a normalized error, we divide the average distance error by the half of each corresponding sub-trajectory length getting the normalized segmentation error in the range $[0, 1]$. Fig. 8 plots this segmentation error for SMOD under additive noise of various SNR.

Fig. 9 illustrates the segmentation results projected on $d(n)$ and on normalized voting signal for the trajectories $A \rightarrow B \rightarrow A \rightarrow B \rightarrow E \rightarrow C$ with additive noise of 50 db (Fig. 9(a)) and $A \rightarrow B \rightarrow D \rightarrow E \rightarrow B \rightarrow A \rightarrow B$ with additive noise of 30 db (Fig. 9(b)). Concerning the first trajectory, in Fig. 9(c), the samples 1, 10, 20, 30, 60 and 100 (horizontal axis) correspond to ground truth ends



(a) (b) (c) (d)

Fig. 7: The trajectories of our synthetic MOD (SMOD) with additive noise of SNR = 50 db projected in **(a)** 2-D spatial space ignoring time dimension and **(b)** spatiotemporal 3-D space. The trajectories of our synthetic MOD with additive noise of SNR = 30 db projected in **(c)** 2-D spatial space ignoring time dimension and **(d)** spatiotemporal 3-D space.

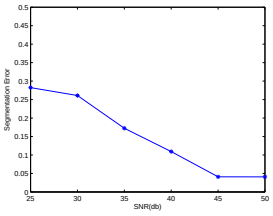


Fig. 8: The segmentation error for SMOD under additive noise of various SNR.

of borders. In this case, we get almost zero error results under any case (see dotted lines of Fig. 9(c)) apart from the sample 20, that was not detected by TSA, since 90% the object of the third sub-trajectory $A \rightarrow B$ continues to $B \rightarrow A$, yielding an almost zero $d(n)$.

Concerning the second trajectory, in Fig. 9(d), the samples 1, 10, 30, 45, 80, 90 and 100 (horizontal axis) correspond to ground truth ends of borders. In this case, we get errors on borders' approximation less than 3 samples due to additive white noise.

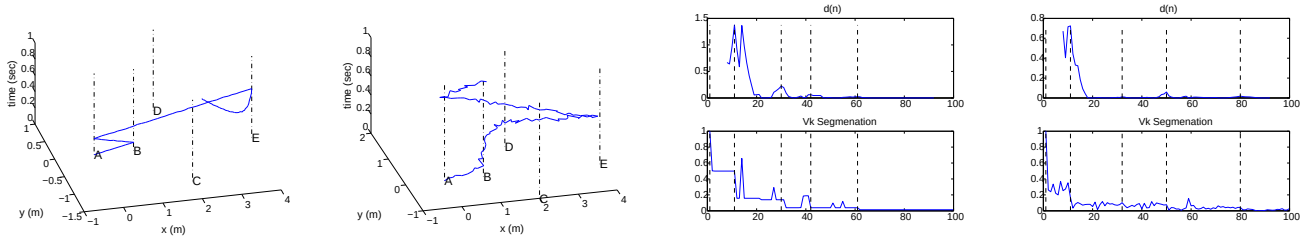
Next, we present the results of the proposed sampling method, in other words the results of the proposed GVA / TSA / SSA three-step methodology. Fig. 10 illustrates results of sampling method for synthetic MOD with additive noise of $SNR = 50db$. It holds that $SR(S)$ gain (see Fig. 10(b)) is almost zero when the sampling set size is greater than 60 meaning that the sampling set of the top-60 representative sub-trajectories is an appropriate minimum number of sampling set size. We get 60 sub-trajectories instead of 78 (ground truth), since some sub-trajectories are not perfectly segmented, e.g. see the sub-trajectory $A \rightarrow B \rightarrow A$ of Fig. 9(a). Figs. 10(c), 10(d) illustrate sampling / visualization of the dataset using the top-60 representative sub-trajectories under the proposed method using a colormap according to $SR(S)$ gain and line segments representativeness, respectively. The most representative sub-trajectories belong to two bidirectional roads ($A \leftrightarrows B$ (629 movements), $B \leftrightarrows E$ (523 movements)) which always appear to have high traffic, while the less representative sub-trajectories belong to a one-directional road ($C \rightarrow B$) which has the lowest traffic (66 movements) mainly appeared during two time periods (see Fig. 7(b)). The sub-trajectories of sampling set are projected in spatiotemporal 3-D space using a colormap according to $SR(S)$ gain (Fig. 11(c)) or line segments representativeness (10(d)). We have used dark blue-to-dark red colormap according to the corresponding to segments representativeness (dark red color for high values, dark blue color for low values). Moreover, the proposed method is not affected by the trajectories' shape, yielding high performance results for both the non-straight and straight movements. Fig. 11 illustrates results of sampling method for synthetic MOD with additive noise of $SNR = 30db$. The selected sub-trajectories represent well the dataset, getting similar results with the case of $SNR = 50db$.

5.2 Experimenting with Real Data

Moreover, we have evaluated the proposed scheme using two real MODs:

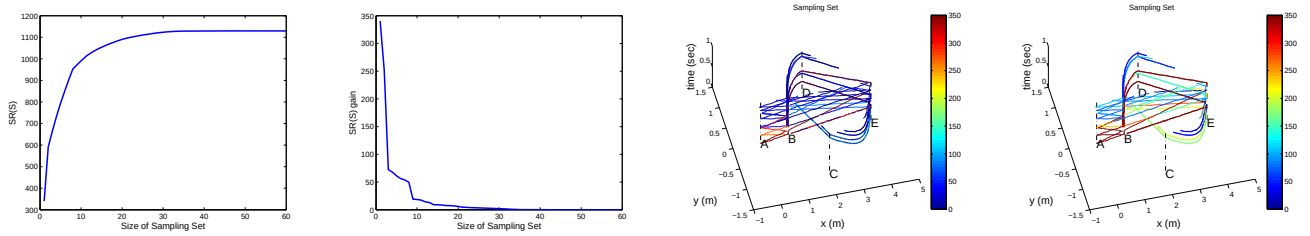
- the "Athens trucks" MOD containing 1100 trajectories. The original dataset consisted of 276 trajectories. From this original dataset, the number of 1100 trajectories was identified by splitting the recordings of a truck in subsets when a temporal gap larger than 15 minutes appears between two consecutive recordings. The dataset is available online at [36]. The GVA / TSA produces 1453 sub-trajectories, since the dataset is sparse in time. The mean value and the standard deviation for number of line segments per trajectory (L_k) are 84.54 and 52.77, respectively.
- The "Milano" dataset consists of GPS traces describing the movement of a set of 17K vehicles during one week at the beginning of April 2007. The original dataset contains 45k trajectories, for a total of 4.7 million points. The dataset is property of Octo Telematics S.p.A., therefore it cannot be made available. The dataset was provided to us for research purposes in the context of the GeoPKDD project [37]. From this dataset, we extracted two subsets; a dense one and a sparse one in the

hal-00585610, version 1 - 13 Apr 2011



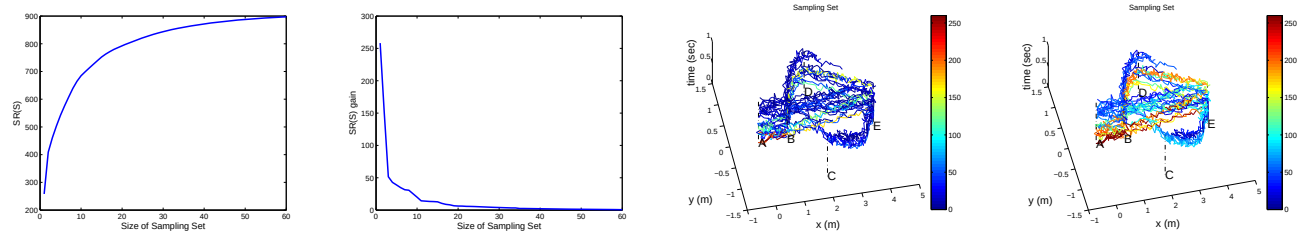
(a) (b) (c) (d)

Fig. 9: Two trajectories of the synthetic MOD used for the segmentation method evaluation projected in spatiotemporal 3-D space **(a)** with additive noise of $SNR = 50db$, **(b)** with additive noise of $SNR = 30db$. **(c), (d)** The dotted lines shows the segmentation results projected on $d(n)$ and on normalized voting signal V_k for the trajectory **(a)** and **(b)**, respectively.



(a) (b) (c) (d)

Fig. 10: Results of sampling method for synthetic MOD with additive noise of $SNR = 50db$. **(a)** The function $SR(S)$ under different sizes of sampling set. **(b)** The $SR(S)$ gain as sampling set increases for synthetic MOD. **(c), (d)** The 60 most representative sub-trajectories are projected in spatiotemporal 3-D space using a colormap according to **(c)** $SR(S)$ gain **(d)** line segments representativeness.



(a) (b) (c) (d)

Fig. 11: Results of sampling method for synthetic MOD with additive noise of $SNR = 30db$. **(a)** The function $SR(S)$ under different sizes of sampling set. **(b)** The $SR(S)$ gain as sampling set increases for synthetic MOD. **(c), (d)** The 60 most representative sub-trajectories are projected in spatiotemporal 3-D space using a colormap according to **(c)** $SR(S)$ gain **(d)** line segments representativeness.

temporal domain, called “Milano-I” and “Milano-II”, respectively. The “Milano-I” MOD contains 5910 trajectories all existing inside a temporal period of 100 minutes duration. For this MOD the GVA / TSA produces 9835 sub-trajectories. The mean value and the standard deviation for number of line segments per trajectory (L_k) are 31.04 and 29.41, respectively. The “Milano-II” consists of 11500 randomly selected trajectories from the original dataset, spanning inside a temporal period of 5 days duration. The mean value and the standard deviation of the number of line segments per trajectory (L_k) are 30.77 and 31.34, respectively. For this MOD the GVA / TSA produces 12895 sub-trajectories. As such, the “Milano-II” is a very sparse in time dataset and we have used it in order to show the limitations of our method.

Fig. 15 illustrates the trajectories of “Athens trucks” MOD projected in 2-D spatial space ignoring time dimension (Fig. 15(a)) and in 3-D spatiotemporal space (Fig. 15(b)). Fig. 16 illustrates the trajectories of “Milano-II” MOD projected in 2-D spatial space ignoring time dimension (Fig. 16(a)) and in 3-D spatiotemporal space (Fig. 16(b)) (the corresponding figures for “Milano-I” MOD are quite similar). It is clear that the full visual-

hal-00585610, version 1 - 13 Apr 2011

hal-00585610, version 1 - 13 Apr 2011

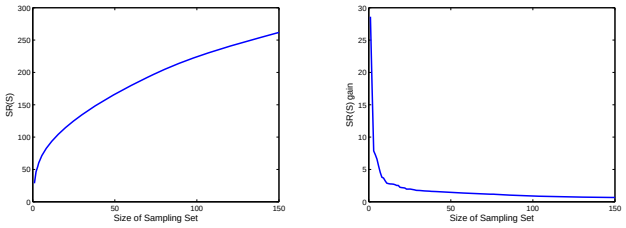
ization cannot be effective, due to the large number of projected trajectories in almost the same time and space. In order to solve this problem, we have used the results of SSA sampling the most representative sub-trajectories of the MOD.

Fig. 12 illustrates the functions $SR(S)$ and $SR(S)$ gain, as sampling set size increases for the "Athens trucks" MOD. It holds that $SR(S)$ gain is low when the sampling set size is greater than 150, meaning that the most appropriate number of sampling set size is at least 150. $SR(S)$ gain always decreases as sampling size increases. Similarly, Fig. 13 illustrates the functions $SR(S)$ and $SR(S)$ gain, as sampling set size increases for the "Milano-I" MOD. It holds that $SR(S)$ gain is very low, when the sampling set size is greater than 120, meaning that the most appropriate number of sampling set size is about 120. Fig. 14 illustrates the functions $SR(S)$ and $SR(S)$ gain, as sampling set size increases for the "Milano-II" MOD. This is a very sparse in time dataset, since the maximum value of $SR(S)$ gain is about 10, while the dataset contains more than 11.000 trajectories. It holds that $SR(S)$ gain is less than one, when the sampling set size is greater than 150.

Fig. 15(c) illustrates a sampling/visualization of the dataset using the top-150 representative sub-trajectories under the proposed SSA method for "Athens trucks" MOD. We have used dark blue-to-dark red colormap according to the corresponding to $SR(S)$ gain (dark red color for high values, dark blue color for low values). Similarly, Fig. 15(d) uses a dark blue-to-dark red colormap according to the corresponding to sub-trajectories representativeness $VP_k(i)$ (dark red color for high values, dark blue color for low values) for "Athens trucks" MOD. The low values of sub-trajectories representativeness is due to the fact that "Athens trucks" MOD is sparse on its temporal dimension. Fig. 16(c) and 16(d) illustrate a sampling/visualization of the dataset using the top-150 representative sub-trajectories under the proposed SSA method for "Milano-II" MOD using colormaps according to $SR(S)$ gain and $VP_k(i)$, respectively. The sub-trajectory sampling set includes the perimeter of trajectories set as well as some intrinsic trajectories.

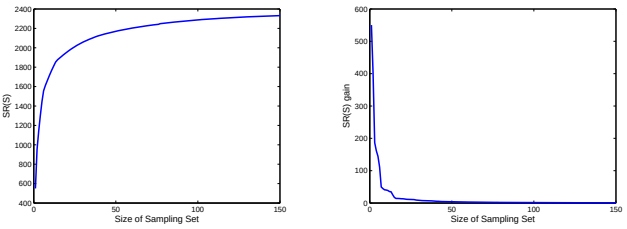
5.3 Comparison with other sampling techniques

The results of the proposed sub-trajectory sampling method have been compared with standard sampling techniques (random and stratified sampling) as these have been proposed in the literature [38]. The sampling techniques have been evaluated on the same dataset of sub-trajectories that is provided by the proposed TSA. According to random sampling, each sub-trajectory is chosen randomly and entirely by chance, such that each sub-trajectory has the same probability of being chosen at any stage during the sampling process. According to stratified sampling, the sub-trajectories need first to be divided into strata (groups). The only relevant approach



(a) (b)

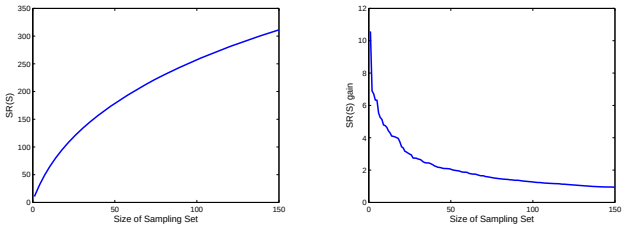
Fig. 12: **(a)** The function $SR(S)$ under different sizes of sampling set for "Athens trucks" MOD. **(b)** The $SR(S)$ gain as sampling set increases for "Athens trucks" MOD.



(a) (b)

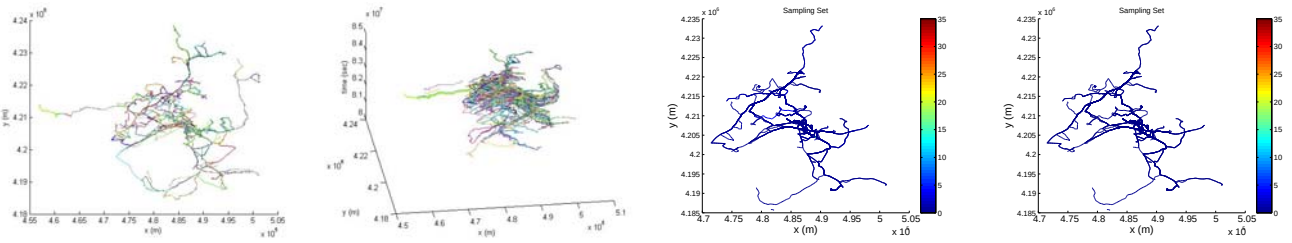
Fig. 13: **(a)** The function $SR(S)$ under different sizes of sampling set for "Milano-I" MOD. **(b)** The $SR(S)$ gain as sampling set increases for "Milano-I" MOD.

that has been proposed in the literature for MOD is [15], which uses the state-of-the-art T-OPTICS algorithm [8]. As such, in a way, we compare our method with the state-of-the-art stratified sampling technique proposed so far in the literature [15]. Following the stratification, the number of sub-trajectories that are selected from each of the strata is analogous to the size of the stratum (i.e. the number of sub-trajectories in this group). Finally, the sub-trajectories are randomly selected from each stratum. In order to measure the performance of sampling, we have used the root mean square error (RMSE) metric, that is defined by the square root of the mean of minimum squared distances between the line segments of dataset (D) and the line segments of the



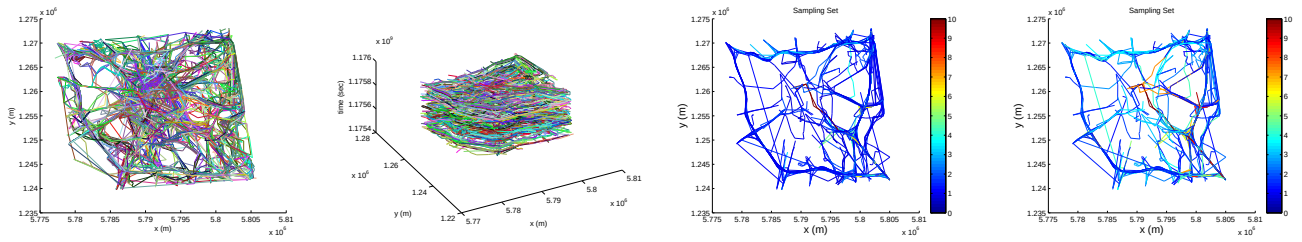
(a) (b)

Fig. 14: **(a)** The function $SR(S)$ under different sizes of sampling set for "Milano-II" MOD. **(b)** The $SR(S)$ gain as sampling set increases for "Milano-II" MOD.



(a) (b) (c) (d)

Fig. 15: The trajectories of "Athens trucks" dataset (1.100 traj.) projected in **(a)** 2-D spatial space ignoring time dimension and **(b)** spatiotemporal 3-D space. **(c), (d)** The top 150 representative sub-trajectories of the dataset projected in 2-D spatial space using a colormap according to **(c)** $SR(S)$ gain **(d)** sub-trajectories representativeness $VP_k(i)$.



(a) (b) (c) (d)

Fig. 16: The trajectories of "Milano-II" dataset (11500 traj.) projected in **(a)** 2-D spatial space ignoring time dimension and **(b)** spatiotemporal 3-D space. **(c), (d)** The top 150 representative sub-trajectories of the dataset projected in 2-D spatial space using a colormap according to **(c)** $SR(S)$ gain **(d)** sub-trajectories representativeness $VP_k(i)$.

sampling set (S) weighted by the normalized lifespan of line segment ($\frac{l_k(i)}{\sum_{i=1}^{L_k-1} l_k(i)}$) (see Equations 11 and 12).

$$SD(S,D) = \sum_{k=1}^{N} \sum_{i=1}^{L_k-1} (d(e_k(i), e_u(v)) \cdot \frac{l_k(i)}{\sum_{i=1}^{L_k-1} l_k(i)})^2 \quad (11)$$

$$RMSE(S,D) = \sqrt{\frac{SD(S,D)}{\sum_{k=1}^{N} \sum_{i=1}^{L_k-1} 1}} \quad (12)$$

$e_u(v)$ is the line segment from sampling set that minimizes $d(e_k(i), e_u(v))$. Intuitively, the smaller the RMSE metric the better the sampling, as this implies better coverage of the space-time of the MOD, or equivalently, it means that less "useless" sub-trajectories have been sampled, where by "useless" we mean trajectories that are already represented in the sampling set by some other sub-trajectory.

Fig. 17 illustrates the function $RMSE(S,D)$ under different sizes of sampling set for the synthetic MOD with additive noise of 50 db (see Fig. 17(a)), the synthetic MOD with additive noise of 30 db (see Fig. 17(b)), for "Athens trucks" MOD (see Fig. 17(c)), for "Milano-I" MOD (see Fig. 17(d)) and for "Milano-II" MOD (see Fig. 17(e)). The blue, red dashed and black dashdot lines have been used to plot SSA, random and stratified sampling RMSE curves.

According to the experimental results, the proposed method outperforms the other sampling techniques in all the cases of synthetics and "Milano-I" MODs. Concerning, the "Athens trucks" MOD, the proposed method outperforms the other sampling techniques in more than 85% of the cases. This is due to the fact that this MOD is temporally sparse. When a MOD is temporally sparse, most of the sub-trajectories have low representativeness, since the number of common time periods between different trajectories is low (see Equation 1). Therefore, in such cases, a sampling method that takes into account the time, as the proposed method does, could not always give the best results, since the selected samples do not represent other samples due to low representativeness values of sub-trajectories. In other words, the sparser in time a MOD is, the closer the behavior of our sampling method is with the random sampling. This is rational, as in highly sparse MOD (where an example of a highly sparse MOD could be the one in which each trajectory exists in different time periods) it is like omitting one of the two dimensions under which the problem definition is formulated, which obviously is a different kind of problem. As shown in Fig. 17(e), the above intuition is verified in this experiment for the "Milano-II" MOD, which is a very sparse in time. Nevertheless, the effectiveness of the proposed method is still high, since it outperforms or gives almost the same results with the other sampling techniques in more than 75% of the cases. In general, by using the standard evaluation metric of RMSE, which is independent to any distribution that

hal-00585610, version 1 - 13 Apr 2011

hal-00585610, version 1 - 13 Apr 2011

a dataset may follow, the results clearly show that our methodology covers better that space-time of a MOD.

6 Conclusions

In this paper, we have discussed the problem of finding representative sub-trajectories in a MOD. Especially, we addressed this issue by segmentation and sub-trajectory sampling based on global spatiotemporal similarity of trajectories. In particular, we have proposed three algorithms: GVA, TSA and SSA for trajectory voting, segmentation and sub-trajectory sampling, respectively. GVA extends the density biased sampling (DBS) from point sets [13] to trajectory segments providing a local trajectory descriptor per line segment that is related to line segment representativeness. Next, TSA automatically and effectively estimates the number of sub-trajectories and their borders, separating each trajectory of MOD into homogenous partitions concerning their representativeness. Finally, SSA is applied over the resulting partitions providing the most representative sub-trajectories of the MOD, also taking into account that high density regions of the MOD should not be oversampled. SSA can be automatically terminated by thresholding the number of moving objects of the original MOD that are represented in sampling set $SR(S)$. Moreover, the index-based voting algorithm, which is the computationally most expensive step in our framework, and the polynomial computational cost of the proposed algorithms makes the scheme applicable to large databases. In our approach, contrary to related work, the temporal dimension of the MOD is taken into consideration, while there is not any inherent constraint on sub-trajectory complexity and shape, yielding trajectory segmentation and sub-trajectory sampling that are related only to representativeness. We have evaluated the proposed method under real and synthetic databases, and the experimental results show the effectiveness and robustness of the proposed scheme.

As future work, we plan to investigate the applicability of the proposed method for (sub-)trajectory clustering. The idea is that MOD clustering can be provided concurrently with MOD sampling. It holds that each sub-trajectory of the sampling set has been voted by different sub-trajectories of the MOD (cluster), under the minimization of the objective function proposed in the current work. Therefore, each sub-trajectory of the sampling set can be considered as a cluster representative (i.e. a seed around which a cluster is formatted). This is a different tactic as the one followed in [4]. In the same context, outliers [39] can be discriminated from low values in voting sub-trajectory descriptor (V_k).

Acknowledgements

We thank anonymous reviewers for their very useful comments and suggestions. This research was partially supported by the FP7 ICT/FET Project MODAP (Mobility, Data Mining, and Privacy) funded by the European Union. URL: www.modap.org.

References

[1] R. H. Guting and M. Schneider. *Moving Object Databases*. Morgan Kaufmann Publishers, 2005.

[2] F. Giannotti and D. Pedreschi. *Mobility, Data Mining and Privacy, Geographic Knowledge Discovery*. Springer-Verlag, 2008.

[3] M. Hadjieleftheriou, G. Kollios, V. Tsotras, and D. Gunopulos. Efficient indexing of spatiotemporal objects. In *Proc. EDBT*, 2002.

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

[5] J.-G. Lee, J. Han, X. Li, and H. Gonzalez. Traclass: trajectory classification using hierarchical region-based and trajectory-based clustering. *PVLDB*, pages 1081–1094, 2008.

[6] A. Anagnostopoulos, M. Vlachos, M. Hadjieleftheriou, E. Keogh, and P. S. Yu. Global distance-based segmentation of trajectories. In *Proc. KDD*, pages 34–43, 2006.

[7] L. Chen, M. T. Özsu, and V. Oria. Robust and fast similarity search for moving object trajectories. In *Proc. SIGMOD*, pages 491–502, 2005.

[8] M. Nanni and D. Pedreschi. Time-focused clustering of trajectories of moving objects. *Journal of Intelligent Information Systems*, 27(3):267 – 289, 2006.

[9] N. Pelekis, I. Kopanakis, G. Marketos, I. Ntoutsi, G. Andrienko, and Y. Theodoridis. Similarity search in trajectory databases. In *Proc. TIME*, pages 129–140, 2007.

[10] M. Benkert, J. Gudmundsson, F. Hubner, and T. Wolle. Reporting flock patterns. In *Proc. ESA*, pages 660–671, 2006.

[11] Y. Li, J. Han, and J. Yang. Clustering moving objects. In *Proc. KDD*, pages 617–622, 2004.

[12] N Pelekis, I. Kopanakis, E. E. Kotsifakos, E. Frentzos, and Y. Theodoridis. Clustering trajectories of moving objects in an uncertain world. In *Proc. ICDM*, 2009.

[13] G. Kollios, D. Gunopulos, N. Koudas, and S. Berchtold. Efficient biased sampling for approximate clustering and outlier detection in large datasets. *IEEE Transactions on Knowledge and Data Engineering*, 15:398–404, 2003.

[14] A. Nanopoulos, Y. Theodoridis, and Y. Manolopoulos. Indexed-based density biased sampling for clustering applications. *Data and Knowledge Engineering*, 57(1):37–63, 2006.

[15] G. Andrienko, N. Andrienko, S. Rinzivillo, M. Nanni, and D. Pedreschi. A visual analytics toolkit for cluster-based classification of mobility data. In *Proc. SSTD*, pages 432–435, 2009.

[16] G. Andrienko, N. Andrienko, S. Rinzivillo, M. Nanni, D. Pedreschi, and F. Giannotti. Interactive visual clustering of large collections of trajectories. In *Proc. VAST*, pages 3–10, 2009.

[17] N. Pelekis, C. Panagiotakis, I. Kopanakis, and Y. Theodoridis. Unsupervised trajectory sampling. In *Proc. ECML-PKDD*, 2010.

[18] H. Jeung, M. L. Yiu, X. Zhou, C. Jensen, and H. T. Shen. Discovery of convoys in trajectory databases. In *Proc. VLDB*, 2008.

[19] G. Andrienko, N. Andrienko, and S. Wrobel. Visual analytics tools for analysis of movement data. *SIGKDD Explor. Newsl.*, 9(2):38–46, 2007.

[20] M. Vlachos, D. Gunopulos, and Gautam Das. Rotation invariant distance measures for trajectories. In *Proc. KDD*, pages 707–712, 2004.

[21] E. Frentzos, K. Gratsias, and Y. Theodoridis. Index-based most similar trajectory search. In *Proc. ICDE*, 2007.

[22] M. Ankerst, M. M. Breunig, H.-P. Kriegel, and J. Sander. Optics: Ordering points to identify the clustering structure. In *Proc. SIGMOD*, 1999.

[23] D. Sacharidis, K. Patroumpas, M. Terrovitis, V. Kantere, M. Potamias, K. Mouratidis, and T. Sellis. On-line discovery of hot motion paths. In *Proc. EDBT*, pages 392–403, 2008.

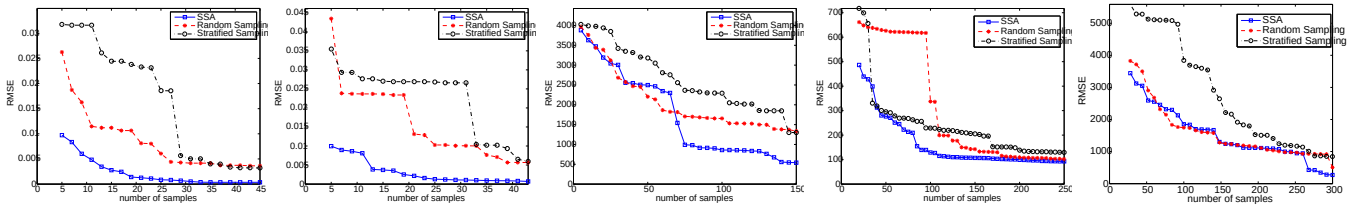
[24] C. Panagiotakis, N. Pelekis, and I. Kopanakis. Trajectory voting and classification based on spatiotemporal similarity in moving object databases. In *Proc. IDA*, pages 131–142, 2009.

[25] D. Pfoser, C.S. Jensen, and Y. Theodoridis. Novel approaches to the indexing of moving object trajectories. In *Proc. VLDB*, 2000.

[26] E. Frentzos, K. Gratsias, N. Pelekis, and Y. Theodoridis. Algorithms for nearest neighbor search on moving object trajectories. *GeoInformatica*, 11:159–193, 2007.

[27] Y. Theodoridis, M. Vazirgiannis, and T. Sellis. Spatio-temporal indexing for large multimedia applications. In *Proc. of ICMCS*, 1996.

[28] D. Patterson. *Artificial Neural Networks*. Prentice Hall, 1996.



(a) (b) (c) (d) (e)

Fig. 17: The function $RMSE(S, D)$ under different sizes of sampling set for **(a)** the synthetic MOD with additive noise of 50 db. **(b)** the synthetic MOD with additive noise of 30 db. **(c)** for "Athens trucks" MOD. **(d)** for "Milano-I" MOD. **(e)** for "Milano-II MOD.

[29] K. Wang J. Yuan, L. Bo and T. Yu. Adaptive spherical gaussian kernel in sparse bayesian learning framework for nonlinear regression. *Expert Syst. Appl.*, 36(2):3982–3989, 2009.

[30] N. Meratnia and R. By. Spatiotemporal compression techniques for moving point objects. In *Proc. EDBT*, 2004.

[31] C. Panagiotakis and G. Tziritas. A speech/music discriminator based on rms and zero-crossings. *IEEE Transactions on Multimedia*, 7(1):155–166, 2005.

[32] C. Panagiotakis, E. Kokinou, and F. Vallianatos. Automatic P-Phase Picking Based on Local-Maxima Distribution. *IEEE Transactions on Geoscience and Remote Sensing*, 46(8):2280–2287, 2008.

[33] P. C. Mahalanobis. On the generalized distance in statistics. In *Proc. NISI*, volume 2, pages 49–55, 1936.

[34] S. K. Zhou and R. Chellappa. From sample similarity to ensemble similarity: Probabilistic distance measures in reproducing kernel hilbert space. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(6):917–929, 2006.

[35] Brinkhoff T. A framework for generating network-based moving objects. *GeoInformatica*, 6(2):153180, 2002.

[36] URL: http://www.rtreeportal.org/index.php?option=com_content&task=view&id=30%&Itemid=43.

[37] URL:www.geopkdd.eu.

[38] S. K. Thompson. *Sampling*. Wiley-Interscience, 2002.

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

Costas Panagiotakis received the B.Sc., M.Sc. and Ph.D. degrees in Computer Science from University of Crete in 2001, 2003 and 2007, respectively. He is Assistant Professor in the Dept. of Commerce and Marketing, Technological Institute of Crete. Moreover, he is Visiting Professor in Computer Science Dept., University of Crete. He is the author of one book (monograph) and more than 30 articles in international journals and conferences. His research interests include signal processing, image and video analysis, multimedia and pattern recognition. [URL: www.csd.uoc.gr/~cpanag].

Nikos Pelekis is a Lecturer at the Dept. of Statistics and Insurance Science, University of Piraeus. He received his B.Sc. degree from the Computer Science Dept. of the University of Crete (1998). He has subsequently joined the Dept. of Computation in UMIST to pursue his M.Sc. (1999) and his Ph.D. (2002). He has co-authored more than 40 research papers and book chapters. His research interests include knowledge discovery, data mining, machine learning, spatiotemporal databases, management of location-based services, and geographical information systems. [URL: http://infolab.cs.unipi.gr/people/npelekis].

Ioannis Kopanakis is an Assistant Professor and Head of the Dept. of Commerce and Marketing at the Technological Educational Institute of Crete. He holds a Diploma in computer science from the University of Crete (1998), Greece, an MSc in information technology (1999), and a PhD in computation (2003), both from UMIST, UK. He is the scientific Director of the e-Business Intelligence Lab (www.e-bilab.com). His research interests include data mining, visual data mining, and business intelligence. He has published more than thirty papers in journals and refereed conferences. [URL: www.e-bilab.com/kopanakis].

Emmanuel Ramasso received a B.S.C degree in automatic control and computer science from the engineering school "POLYTECH'SAVOIE" (France) in 2004. He then earned a Ph.D. from the University JOSEPH FOURIER (France) in 2007. Emmanuel Ramasso was then a postdoctoral researcher at Commissariat a' l'Energie Atomique (France). In Sept. 2008, he became Assistant Professor at "Ecole Nationale Supe'rieure de Me'canique et des Microtechniques" (ENSMM, France). His research interests include graph theory, probability theory, Evidence theory/Transferable Belief Model, human motion recognition and image/video analysis. [URL: http://www.femto-st.fr/~emmanuel.ramasso/].

Yannis Theodoridis is Assoc. Professor with the Dept. of Informatics, University of Piraeus (UniPi). Born in 1967, he received his Diploma (1990) and Ph.D. (1996) in Electrical and Computer Engineering, both from the National Technical University of Athens. His research interests include database management, geographical information management, data / text mining. He has co-authored three monographs and over 70 publications in scientific journals and conferences with more than 500 citations in his work. [URL: http://www.unipi.gr/faculty/ytheod/]

hal-00585610, version 1 - 13 Apr 2011