Semantic Trajectories: Mobility Data Computation and Annotation

ZHIXIAN YAN, Swiss Federal Institute of Technology (EPFL), Switzerland
DIPANJAN CHAKRABORTY, IBM Research, India
CHRISTINE PARENT, University of Lausanne, Switzerland
STEFANO SPACCAPIETRA, Swiss Federal Institute of Technology (EPFL), Switzerland
KARL ABERER, Swiss Federal Institute of Technology (EPFL), Switzerland

With the large-scale adoption of GPS equipped mobile sensing devices, positional data generated by moving objects (e.g., vehicles, people, animals) are being easily collected. Such data are typically modeled as streams of spatio-temporal (x,y,t) points, called *trajectories*. In recent years trajectory management research has progressed significantly towards efficient storage and indexing techniques, as well as suitable knowledge discovery. These works focused on the geometric aspect of the raw mobility data. We are now witnessing a growing demand in several application sectors (e.g., from shipment tracking to geo-social networks) on understanding the *semantic* behavior of moving objects. Semantic behavior refers to the use of semantic abstractions of the raw mobility data, including not only geometric patterns but also knowledge extracted jointly from the mobility data and the underlying geographic and application domains information. The core contribution of this paper lies in a *Semantic Model* and a *Computation and Annotation Platform* for developing a semantic approach that progressively transforms the raw mobility data into semantic trajectories enriched with segmentations and annotations. We also analyze a number of experiments we did with semantic trajectories in different domains.

Categories and Subject Descriptors: H.2.8 [**Database Management**]: Database Applications—*spatial and temporal databases, annotation, data mining*; H.4.0 [**Information Systems Applications**]: General—*Mobility data analysis, Geographic Information System*

General Terms: Algorithms, Design, Experimentation

Additional Key Words and Phrases: spatio-temporal / structured / semantic trajectory, trajectory computing, trajectory annotation, trajectory segmentation, spatial join, map matching, hidden Markov model

1. INTRODUCTION

It has become increasingly common for moving objects (e.g., cars, people) to carry embedded GPS chipsets, which allow collecting movement data. Berg Insight[1], for example, forecasts an increase in GPS handsets to 960 million units in 2014. As a consequence of this steady growth, the number of applications using mobility data for a

[1] `http://www.berginsight.com/`

This paper extends our prior works of "*A Hybrid Model and Computing Platform for Spatio-Semantic Trajectories*" (ESWC 2010) [Yan et al. 2010] and "*SeMiTri: A Framework for Semantic Annotation of Heterogeneous Trajectories*" (EDBT 2011) [Yan et al. 2011], and presents a complete platform for building semantic trajectories from raw mobility data, extending our model and our platform and including recently conducted, exhaustive experimental results.

Author's addresses: Z. Yan, S. Spaccapietra, and K. Aberer are with EPFL-IC, 1015 Lausanne, Switzerland; email: {firstname.secondname}@epfl.ch. C. Parent is with UNIL-HEC, 1015 Lausanne, Switzerland; email: christine.parent@unil.ch. D. Chakraborty is with IBM Research, India; email: cdipanjan@in.ibm.com.

DOI 10.1145/0000000.0000000 http://doi.acm.org/10.1145/0000000.0000000

variety of purposes is similarly increasing. Examples of well-recognized application of mobility data range from tracking, urban planning, and traffic management, to wildlife behavior analysis, mobility-aware social computing, and geo-social network.

Traditionally, research on mobility data management has centered around moving object databases and statistical analysis. These works primarily focus on: (1) *Data Models*: Definitions and extensions of trajectory related datatypes such as *moving point/region* [Güting and Schneider 2005; Wolfson et al. 1998]. (2) *Data Management*: Efficient storage of mobility data with ad-hoc indexing and querying techniques [Saltenis et al. 2000; Chen et al. 2008]. A number of trajectory database management systems like Secondo [Güting 2005], HERMES [Pelekis et al. 2006] and DOMINO [Wolfson et al. 1999] have been built within these works. (3) *Data Mining*: Design of trajectory mining & learning algorithms (e.g., clustering, classification, outlier detection, finding convoys, sequential pattern mining) and of prototypes for pattern discovery over real-life GPS data [Han et al. 2008]. Existing prototypes include *MoveMine* [Li et al. 2011], *GeoLife* [Zheng et al. 2010] and *GeoPKDD* [Nanni et al. 2010].

While providing efficient data management and mining techniques, these studies mainly focus on raw trajectories (spatio-temporal records $\langle x, y, t\rangle$ using geodetic coordinates), ignoring the background contextual information (e.g., land-use grids and geographical objects) that can contribute significant semantic knowledge about movements. As a result, it is hard to have a *holistic* interpretation (encompassing all relevant semantic information) of movement behaviors that includes contextual data. Thus, many new applications are interested in understanding and using a semantic interpretation or behavioral aspect of the moving object. For example, geo-fencing based applications essentially focus on generating high-level events (e.g., inter-region movement) when mobile endpoints cross domain boundaries or deviate from pre-defined trajectories. There is a strong emphasis on developing techniques for higher level and semantic events (e.g. Harry just *reached office*, Sally is *shopping in CoopCity*, Dave is *stuck in traffic*) inferred from raw GPS-alike data. *Semantics* simply speaking refers to additional information available about the moving object, apart from its mere position data. Semantics is contained both in the geometric properties of the spatio-temporal stream (e.g. when the user stops/moves) as well as in the geography on which the trajectory passes (e.g. shops, roads). An example of semantically enriched trajectory could be:

(*Begin, home, -9am, -*) → (*move, road, 9am-10am, on-bus*) → (*stop, office, 10am-5pm, work*)
→ (*move, road, 5pm-5:30pm, on-metro*) → (*stop, market, 5:30pm-6pm, shopping*)
→ (*move, road, 6pm-6:20pm, walking*) → (*End, home, 6:20pm-, -*)

Note that the above example includes generic movement characteristics (e.g., stop/moves), application-specific geographical objects (e.g., office) and also additional behavioral context (e.g., shopping, work).

This paper reports our research to build a framework that is capable of developing suitable spatio-temporal and semantic abstractions of complete trajectories (from begin to end), exploiting both the geometric properties of the stream and the semantics of the underlying geographic context. Semantic enrichment materializes as annotations embedded into the trajectory data, i.e. additional data attached to the spatio-temporal positions in the trajectory and encoding extra knowledge about the trajectory. Examples of annotations include recording the observed activity of a moving animal (with activity values "feeding", "resting", "moving", etc.), computing and recording the instant speed of the moving object, inferring and recording the means of transportation used by a moving person (e.g. by foot, bus, metro, bicycle). A careful design of our framework ensures that our semantic trajectory representation model and our algorithms are generic enough to be applicable on trajectories of various moving objects, showing various patterns and qualities of movement data.

1.1. Challenges

Designing a generic model and the corresponding framework for generating semantic trajectories is not a trivial task. Several issues need to be addressed.

(1) The model and framework should *be application-independent*, i.e., able to support the requirements of different scenarios (e.g. traffic monitoring, fauna behavioral analysis). No application-specific data should be hard-coded inside. Instead, the framework should have the capability to acquire from 3rd party sources whatever geographic or application-specific data is needed for building semantic trajectories.
(2) Building semantic trajectories directly from each individual GPS record is computationally inefficient. The trajectory model must offer generic means of semantically *aggregating* correlated records and provide their condensed representation at the semantic level. Applications support different levels of granularity.
(3) The annotation algorithms should be generic to exhibit a *good performance* over a wide range of trajectories with different characteristics and data qualities. For example, GPS sampling rates can be different. As a result, correctly mapping trajectories to location artifacts in complex environments such as dense urban areas is a challenge. The algorithms should be able to handle such variations in data quality while annotating trajectory parts.
(4) In order to provide a *holistic* annotation framework, several independent sources need to be integrated. This makes the amount of candidate annotation data rich and spatially dense. The framework needs to select the most relevant semantic annotation for each trajectory segment. For example, it does not make sense to annotate a moving car with the list of restaurants it passes by, unless it stops around one.

1.2. Core Contributions

This paper overviews our research on developing a semantic approach whose functionalities enable progressively turning raw mobility data into semantic trajectories readily suitable for use by applications. The approach aims at promoting trajectory semantic annotation while minimizing the computational cost of data annotation. While parts of this work have already been presented elsewhere, a novelty of this paper is to offer a consolidated and complete document collecting and unifying material scattered over previous papers, to serve as a basic reference to our work.

The main innovation emphasized by this contribution is the global framework that we provide to develop a suite of concepts (supported by a suite of implemented processes) that allows an application designer to get exactly the representation of trajectories at the level needed by the application, from the low level raw data to the upper level characterizing semantically rich trajectories. Specifically, we design a semantic model that extends prior models (e.g., [Spaccapietra et al. 2008; Yan et al. 2008]) to be generic enough to capture semantics from both geometric properties of the stream and from background geographic data. With this semantic model, we provide a complete system that first exploits the spatio-temporal data to extract structured trajectories (as stop and move episodes) and then utilizes the geographic context to annotate stops & moves with the geographic objects relevant to the application. In short, the core contributions of the paper are:

(1) *Spatio-temporal & semantic trajectory model*: The model captures trajectories at different levels, from low-level location feeds to high-level semantic behaviors. It covers spatio-temporal trajectories, structured trajectories, and semantic trajectories.

(2) *Trajectory computing platform*: We built a computing platform that encapsulates our data abstractions by using several data processing layers (i.e., data cleaning, trajectory identification, and trajectory segmentation).
(3) *Trajectory semantic annotation*: The platform supports various annotation strategies and mechanisms to enrich trajectories, using knowledge from various background geographic data sources (e.g., region information, road networks, points of interest) as well as application-specific sources.
(4) *Experiments and Evaluations*: We report on several experiments we did using large-scale real GPS location feeds (vehicle movement, people trajectories). We validate our results with both statistical analysis and limited ground truth.

The paper is organized as follows: Section 2 compares our approach and techniques to related work from the existing literature. Section 3 presents the data model that is peculiar to our approach The computation framework is presented in two steps. Section 4 presents the creation of structured trajectories from raw data. Section 5 presents the annotation of the structured trajectories that generates semantic trajectories. Section 6 reports on experiments and their analysis. Section 7 presents concluding remarks.

2. RELATED WORK

Trajectory data analysis recently has become an active research area because of a large availability of mobile tracking sensors, e.g., GPS embedded smartphones. Many works are related to this paper; we divide them into three categories: trajectory data *modeling*, *processing*, and *semantic enrichment*.

2.1. Trajectory Data Modeling

Traditional trajectory studies largely focus on data analysis from a spatial or spatio-temporal perspective. Thus, data modeling mainly concerns designing moving object or trajectory data types for data management, in order to support efficient data indexing and query processing [Güting and Schneider 2005][Kuijpers and Othman 2007].

In order to build a rich mobility data model that can capture high-level semantics, our prior work has explored approaches for developing new conceptual models where the semantics of movement can be explicitly expressed, e.g., the trajectory conceptual view in terms of a stop-move model [Spaccapietra et al. 2008] and trajectory ontologies for conjunctive query processing and reasoning [Yan et al. 2008]. Such trajectory modeling concepts have been largely used in several projects on mobility, e.g., GeoPKDD[2] (*Geographic Privacy-aware Knowledge Discovery and Delivery*) [Giannotti and Pedreschi 2008], MODAP[3] (*Mobility, Data Mining, and Privacy*) and SEEK[4] (*SEmantic Enrichment of trajectory Knowledge discovery*). These modeling concepts are well fitted for the semantic analysis of movements, like tourist movements [Alvares et al. 2007], the semantic interpretation of stops [Gómez and Vaisman 2009] and moves [Mouza and Rigaux 2005].

However, an important challenge not yet addressed is to have a *generic* model with a supporting platform to develop these abstracted conceptual trajectories from the low-level mobility GPS feeds. In this paper, we provide a comprehensive model (*a hybrid spatio-temporal & semantic trajectory model*) that supports multi-level trajectory abstractions, ranging from the raw mobility data to high-level semantic trajectories.

[2]`http://www.geopkdd.eu/`
[3]`http://www.modap.org/`
[4]`http://www.seek-project.eu/`

2.2. Trajectory Data Processing

Similarly to conventional data modeling and management, trajectory data processing focuses on the geometric perspective when analyzing mobility. The study is mainly about building processing algorithms for trajectory reconstruction.

For the initial data preprocessing, researchers have designed algorithms for cleaning (i.e., dealing with data errors and outliers) and compression. For example, Marketos et al. propose a parametric online approach that filters noisy positions (outliers) by taking advantage of the maximum allowed speed of the moving object [Marketos et al. 2008]. On the other hand, random errors are small distortions from the true values and their influence is decreased by *smoothing* methods (e.g., [Jun et al. 2006][Schüssler and Axhausen 2009]). Additionally, many works study trajectory data compression. For instance, Meratnia and de By design the *opening window* techniques for online compression, among which there are two choices in threshold violation, i.e., using the point that causes the violation (NOPW - *NOrmal Opening Window*) or using the point just before the violation (BOPW - *Before OPening Window*) [Meratnia and de By 2004]. Different from these works, our semantic trajectory computation and annotation platforms can support more efficient one-loop data cleaning and semantic data compression. Recent progress has been made for semantic trajectory reconstruction for real-time movement streaming data [Yan et al. 2011].

Segmentation is yet another important step in understanding mobility data. Zheng et al. provide a change point based segmentation for GPS trajectories according to the transportation means [Li et al. 2008][Zheng et al. 2011]. Their algorithm first identifies the *walk* segments, and then uses them to infer the other *non-walk* segments. However, they use a universal threshold for determining whether a segment is walk or non-walk. In our approach, the trajectory segmentation supports dynamic stop threshold that can avoid false-negatives like traffic congestion. Recently, Buchin et al. present a theoretic framework that computes an optimal segmentation by using several criteria (e.g., speed, direction, location disk) from the computational geometry perspective [Buchin et al. 2010]. However, their methods do not provide any experimental study for validating their segmentation framework.

In contrast to these largely piece-meal trajectory processing studies, our approach provides a holistic multi-layer trajectory computation platform for trajectory reconstruction. Our platform supports various applications with a complete and plug-able workflow including *data cleaning* (considering both random errors and outliers), *data compression*, and several kinds of *trajectory segmentation* algorithms (e.g., dynamic velocity threshold, density) for various kinds of trajectory applications.

2.3. Trajectory Data Enrichment

The goal of trajectory data enrichment is to add semantic annotations by using geographic and application domain knowledge. Like trajectory data processing, the literature is also piece-meal, full of enrichment algorithms that annotate a specific type and/or a specific part of trajectories. Dedicated algorithms are independently designed for trajectory annotations with each kind of geo-objects: regions, lines or points.

Enrichment with geographic regions: The works focus on computing topological correlations (called *spatial predicates*) between trajectories and regions. For example, Alvarez et al. perform a spatial join between the trajectories and a given set of regions of interests (ROIs) for computing frequent *moves* between *stops* [Alvares et al. 2007]. Other works (e.g., [Nergiz et al. 2009]) apply similar data abstraction concepts for cloaking people locations to preserve their privacy. Our method uses the two kinds of formats that are provided by geographic sources: vector regions (e.g., regions from Openstreetmap) and raster regions (e.g., regions implicitly defined in land use grids).

Enrichment with geographic lines: For trajectory enrichment with geographic lines, an important topic is developing efficient *map matching* algorithms. Map matching aims at identifying the correct road segment on which a vehicle is traveling and additionally approximating the vehicle's position on the segment [Quddus et al. 2007][Brakatsoulas et al. 2005]. Map matching methods can be classified into three categories: geometric [Bernstein and Kornhauser 1996], topological [White et al. 2000], and recent advanced methods [Newson and Krumm 2009][Lou et al. 2009]. Traditional map-matching techniques target high matching accuracy, which is usually suited for movement with a unique kind of vehicle (e.g., car or truck). On the other side, we study map matching for trajectories that use various transportation means (e.g., walking for boarding on a bus and then a train). Thus, we have an additional post map-matching task that infers the transportation mode for each movement episode. Zheng et al. study the transportation mode by using segment features such as *distance*, *average speed*, *stop rate* [Zheng et al. 2010]. Beside using such GPS-based segment features, our approach also uses extra semantic information from Openstreetmap (e.g., metro lines, bus stops) for improving the inference accuracy.

Enrichment with geographic points: Complementary research focuses on identifying meaningful points of interests (POI) related to trajectories, based on clustering [Zhou et al. 2007][Palma et al. 2008] or reinforcement inference techniques (e.g., HITS and PageRank) [Cao et al. 2010][Zheng et al. 2009]. In addition, [Xie et al. 2009] designs a semantic spatio-temporal join method to infer activities from trajectories, based on a small set of pre-defined geographic hotspots. [Li et al. 2010] design algorithm for mining periodic behaviors in trajectories, focusing on semantic points like *home*/*office*. However, most of these studies consider only environments with sparse POIs, where identifying the meaningful POI for each trajectory part is trivial. In our approach, we consider trajectories in a city center with very dense POIs. We design an HMM-based POI inference for identifying the latent stop behaviors hidden in the raw mobility data.

In summary, we observe that these semantic enrichment works focus on specific situations and provide algorithms that are applicable to compute and annotate only certain kinds (or parts) of trajectories [Alvares et al. 2007][Yin et al. 2004][Palma et al. 2008][Xie et al. 2009][Newson and Krumm 2009], e.g., map-matching for vehicle moves or extracting important POIs for hotspots. None of them considers the analysis of complete trajectories that contain heterogeneous semantics, like the example of semantic trajectory in Section 1 (with semantics on both stops and moves). It is difficult to adapt these works to different types of moving objects (e.g., vehicles and people trajectories), or to trajectories crossing geo-objects of different kinds (e.g. lines and regions and points). Moreover, inferring such heterogeneous semantics needs multiple geographic data sources to be combined meaningfully. Our objective is to create a holistic framework for end-to-end computation and annotation of heterogeneous trajectories.

3. HYBRID SPATIO-TEMPORAL & SEMANTIC TRAJECTORY MODEL

Current mobility models focus either on high-level data representation (e.g., ontologies) or low-level GPS processing (e.g., mobility data management and mining). Our proposal is a hybrid *Spatio-Temporal & Semantic (STS)* trajectory model that: (1) encapsulates raw GPS spatio-temporal trajectory data; (2) provides a progressive abstraction of the raw data up to higher-level semantic representations; (3) supports well-known concepts like stop-move in [Spaccapietra et al. 2008]. Our key design considerations for this hybrid model are:

— *Raw Data characteristics:* The model should consider characteristics of raw mobility tracking data (e.g., spatial and temporal gaps, uncertainties) to create simple *low-level* representations (e.g., hourly, daily, monthly and geo-fenced trajectories).

—*Progressive computation:* The model should be designed so that a layered computing platform can generate higher-level semantic abstractions from the underlying lower-level trajectory representation.
—*Encapsulate various semantics:* The model should be able to encapsulate various kinds of semantic annotations inferred from heterogeneous 3rd party geographic artifacts (e.g. landuse, road networks, points of interests) and rules about the real world (e.g. cars stop at red lights, buses stop at bus stops).

Therefore, our hybrid model consists of (1) the *Raw Data Model* that provides the trajectory definitions available from the raw data perspective; (2) the *Conceptual Model* which is a mid-level abstraction of a trajectory that provides a structured view of the raw mobility data; (3) the *Semantic Model* that provides a semantically enriched and more abstract view of the trajectory. Fig. 1 provides an illustration of these models.

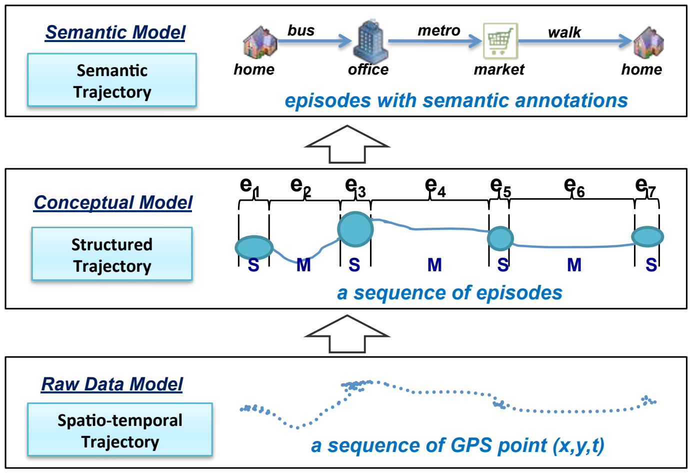


Fig. 1: The Hybrid Spatio-Semantic Trajectory Model

3.1. Raw Data Model

The raw data model is the first abstraction level over the raw mobility data. The raw data like GPS records are typically captured by positioning sensors that continuously record the location of the moving object. So, the raw mobility data for a moving object is in essence a long sequence of spatio-temporal tuples $(position, timestamp)$ collected over some time interval. Most real-life location traces today are essentially GPS-like tuples (*longitude*, *latitude*, *timestamp*) – (x, y, t) in short. From now on, we use the term *GPS feed* to represent the *raw* sequence of spatio-temporal points of a moving object.

In our *Raw Data Model*, we decompose each GPS feed into *subsequences* so that each subsequence represents one meaningful unit of movement. We call these meaningful units "*spatio-temporal trajectories*". Consequently, a spatio-temporal trajectory has a starting point (x, y, t), called *Begin*, and, further, an ending point, called *End*; These two spatio-temporal points delimit the subsequence of the trajectory, along with the corresponding time interval $[t_{begin}, t_{end}]$.

DEFINITION 1 (SPATIO-TEMPORAL TRAJECTORY – $\mathcal{T}_{spa}$). *Given a GPS feed $\mathcal{G}$ of a moving object, $\mathcal{G} = \{p_1, p_2, \ldots, p_m\}$ (where each $p_i = (x_i, y_i, t_i)$ represents a spatio-temporal point), a spatio-temporal trajectory $\mathcal{T}_{spa}$ is a cleaned subsequence of $\mathcal{G}$ for a given time interval $[t_{begin}, t_{end}]$, such that the subsequence does not contain any significant space or time gap.*

3.2. Conceptual Model

The term conceptual model refers to the logical partitioning of a spatio-temporal trajectory $\mathcal{T}_{spa}$ into a series of non-overlapping *episodes*. A $\mathcal{T}_{spa}$ partitioned into episodes is called a *Structured Trajectory* ($\mathcal{T}_{str}$). Conceptually, an *episode* abstracts a subsequence of spatio-temporal points in $\mathcal{T}_{spa}$ that show a high degree of correlation w.r.t. some spatio-temporal feature (e.g., velocity, angle of movement, density, time interval). An *episode* has the following salient features:

— *It is a generic trajectory structuring concept:* By generically denoting a subsequence of a trajectory, the episode concept generalizes several other concepts that have been defined in the literature. *Stop and move* episodes were defined in [Spaccapietra et al. 2008]. In [Andrienko et al. 2011], the authors visualize trajectories as sequences of time-bars that are episodes defined according to range intervals of a given attribute (e.g. distance to a given geo-object, speed, direction).
— *It can be computed automatically:* Episodes can be computed with trajectory structuring algorithms by using the correlations in the spatio-temporal characteristics of consecutive points of the GPS feed, like *velocity*, *acceleration*, *orientation*, *density*.
— *It enables data compression:* Instead of tagging with an annotation each GPS record (which is possible), we can tag the episode. This reduces the size of the data needed to represent structured trajectories. For instance, Fig. 1 shows the annotation of 7 episodes in the conceptual model ("S" and "M" annotations), which is more efficient than annotating each individual GPS record.

DEFINITION 2 (STRUCTURED TRAJECTORY – $\mathcal{T}_{str}$). *A structured trajectory $\mathcal{T}_{str}$ consists of a sequence of "episodes", i.e., $\mathcal{T}_{str} = \{e_1, e_2, \ldots, e_m\}$, where $e_i = (time_{from}, time_{to}, da, rep)$*

— *$time_{from}$ is the instant of the first point of the episode, $time_{to}$ is for the last point of the episode.*
— *da is the "defining annotation" of the episode. It represents the common spatio-temporal characteristic that is shared by all the spatio-temporal points of the episode.*
— *rep is the spatio-temporal or spatial representation of the episode. It is either the sequence of points of the episode or a spatial abstraction of this sequence: the couple of the two extremity points of the episode, the center point of the episode, or the bounding rectangle of the episode.*

3.3. Semantic Model

In the *Semantic Model*, a semantic trajectory $\mathcal{T}_{sem}$ is a structured trajectory enhanced with semantic annotations of its episodes. An example of *semantic trajectory* is shown in the upper layer of Fig. 1 (the semantic trajectory example in Section 1). It shows the semantic trajectory of a given employee on a given day: he goes to work from *home* (morning); after *work* (later afternoon), he leaves for shopping in *market*, and finally reaches *home* (evening).

Semantic trajectories can be computed by integrating data from third party geographic sources (e.g., geographic databases describing landuse, road network, or points of interest), social networks containing data related to locations, and common sense knowledge about the real world (e.g. usually midnight GPS points of persons are located at home). Our system describes a set of semantic enrichment methodologies that can be applied by using such third party data for computing the semantic trajectories (through the trajectory annotation platform in Section 5).

DEFINITION 3 (SEMANTIC TRAJECTORY $\mathcal{T}_{sem}$). *A semantic trajectory $\mathcal{T}_{sem}$ is a structured trajectory where the spatial data (the coordinates) are replaced by geo-annotations and further semantic annotations may be added. Episodes are enriched to generate semantic episodes (se) with geographic or application knowledge: the spatio-temporal or spatial representation of the episode is replaced by a reference to the geo-object where the episode takes place, i.e., $\mathcal{T}_{sem} = \{se_1, se_2, \ldots, se_m\}$, where each semantic episode is defined by: $se_i = (da, sp_i, t_{in}^{(sp_i)}, t_{out}^{(sp_i)}, tagList)$*

— *da is the defining annotation of the episode (e.g. "stop" or "move").*
— *sp_i (semantic position) is a geo-object or one of its characteristic. The geo-object represents the location of the episode at the semantic level. It is a real-world object taken from the available geographic knowledge (e.g., a building, a roadSegment, an administrativeRegion, a landuse region) or from application domain knowledge (e.g., the home or the office of a specific person of the application). A frequent characteristic of geo-objects used for semantically locating episodes is the type of the geo-object, e.g. Hotel, Restaurant, LocalStreet, CollectorStreet.*
— *$t_{in}^{(sp_i)}$ is the incoming timestamp for the trajectory entering this semantic position (sp_i), and $t_{out}^{(sp_i)}$ is the outgoing timestamp for the trajectory leaving sp_i. They can be approximated by the $time_{from}$ and $time_{to}$ of the episode.*
— *$tagList$ is a list of additional semantic annotations about the episode, e.g., the activity performed during stop episodes by the moving agent (shopping, working or eating), the transportation mode used by the moving agent for the move episodes (bike, bus, car, or walk).*

Our hybrid STS model is generic and can be used to represent various ontological frameworks for trajectory modeling [Yan et al. 2008] [Wessel et al. 2009]. In the following we focus on the computation and annotation platforms that enable the creation of semantic trajectories from GPS feeds and 3rd party geographic data sources.

4. TRAJECTORY COMPUTATION

The *Trajectory Computing Platform* exploits the Spatio-Semantic Trajectory model and builds trajectory instances at different levels (*spatio-temporal*, *structural*), from large-scale real-life GPS feeds. Fig. 2 shows the three layers in our platform, each containing several techniques for progressive computation of the trajectory instances.

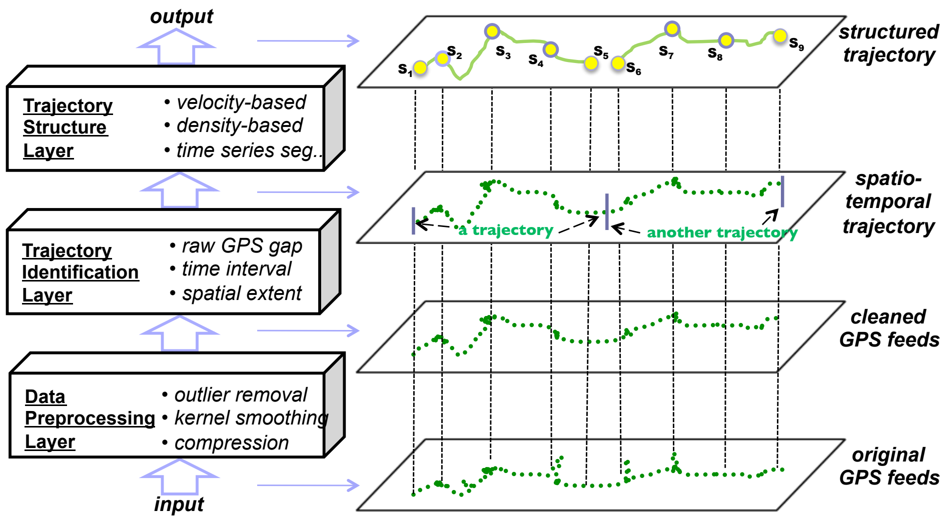


Fig. 2: Trajectory Computing Platform

1) *Data Preprocessing Layer:* This layer cleans the raw GPS feed, in terms of preliminary tasks such as outliers removal and regression-based smoothing. The outcome of this step is a cleaned sequence of (x, y, t). We also have a data compression functionality, but this is not the focus of this paper.

2) *Trajectory Identification Layer:* This layer divides the sequence of cleaned (x, y, t) points into several meaningful trajectories (spatio-temporal trajectories $\mathcal{T}_{spa}$). This step exploits gaps present in the sequence and applies well-defined policies for temporal and spatial demarcations (e.g., daily time intervals, city areas etc).
3) *Trajectory Structure Layer:* This layer is for computing *episodes* present in each spatio-temporal trajectory and generates structured trajectory $\mathcal{T}_{str}$. It contains several algorithms for computing correlations between consecutive GPS points.

4.1. Data Preprocessing Layer

Due to GPS measurements and sampling errors from mobile devices, the recorded position of a moving object is not always correct [Zhang and Goodchild 2002]. Usually the recorded data is unreliable, imprecise, incorrect and contains noise. There exist work on determining possible causes for such uncertainty [Frentzos 2008].

We provide a *Data Preprocessing Layer* for cleaning the data. For this layer, we redesigned GPS preprocessing techniques [Schüssler and Axhausen 2009] to perform our preprocessing steps. In particular, we have built techniques to detect (1) *systematic errors* (outliers): observations that deviate significantly from the desired correct position; (2) *random noise*: GPS signals can have noise from several sources. E.g., ionospheric effects and clocks of satellites can contribute towards white noise of ± 15 meters.

For *outliers*, we applied velocity threshold to remove points that do not give us a reasonable correlation with expected velocity. Each GPS feed has domain knowledge of the moving object (e.g., car, bike, people walk etc). This allows us to remove outliers by using the velocity of this kind of object. For *random noises*, we design a Gaussian kernel based local regression model to smooth out the GPS feed. The smoothed position $(\widehat{x_{t_i}}, \widehat{y_{t_i}})$ is the weighted local regression based on the past points and future points within a sliding time window, where the weight is a Gaussian kernel function $k(t_i)$ with the kernel bandwidth σ (Formula 1). To control the smoothing related information loss, we adopt a reasonably small value for σ (e.g., $5 \times$ GPS sampling frequency) so that only nearby points can affect the smoothed position. This is necessary as we wanted to calibrate the technique to handle only the noise while avoiding under-fitting.

$$(\widehat{x_{t_i}}, \widehat{y_{t_i}}) = \frac{\sum_i k(t_i)(x_{t_i}, y_{t_i})}{\sum_i k(t_i)}, where\ k(t_i) = e^{-\frac{(t_i - t)^2}{2\sigma^2}} \tag{1}$$

Fig. 3, 4, 5 show an example of our smoothing algorithm on a real data set taken from wildlife tracking data on a given day. It contains 52 GPS (x,y,t) records. Fig. 3 shows the smoothed longitude (actually transformed X in cartesian coordinate). Fig. 4 shows the smoothed latitude (transformed Y in cartesian coordinate) and Fig. 5 plots the original GPS feed before smoothing and the smoothed one.

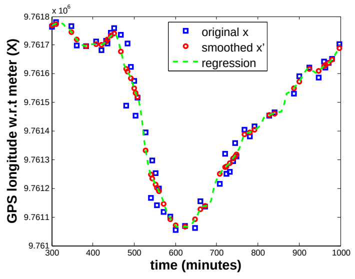


Fig. 3: Smooth GPS (x)

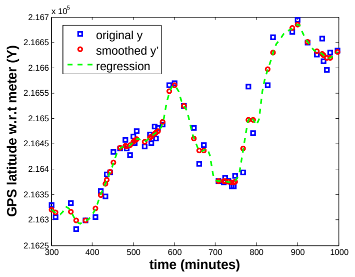


Fig. 4: Smooth GPS (y)

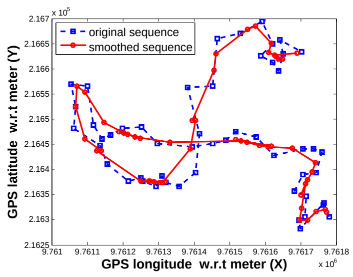


Fig. 5: Original/smoothed

These smoothing techniques are designed for cleaning GPS data of the freely moving objects. However, in many cases, objects (e.g. vehicles) move along network constrained paths (e.g., transportation network) [Güting et al. 2006]. Regarding network-constrained trajectory data, map matching can be applied for determining the correct positioning and removing noise - by integrating positioning data with spatial road network to identify the correct road segment on which a vehicle is traveling and to determine the location of a vehicle on this segment [Quddus et al. 2007] [Brakatsoulas et al. 2005]. We also apply map matching for annotating trajectories, in particular for the *move* episodes. The details of map matching can be found in Section 5.2.

Some other trajectory data preprocessing methods can also be applied at this stage. For instance, a couple of data compression and uncertainty models deal with the raw GPS feeds [Frentzos 2008]. On the contrary, this paper focuses on using semantic abstraction to further compress the raw mobility data.

4.2. Trajectory Identification Layer

This layer uses the cleaned data and extracts relevant non-overlapping spatio-temporal trajectories $\mathcal{T}_{spa}$ (*data model*). The central issue here is to determine reasonable identification policies, to identify the *division points* (x_i, y_i, t_i) that divide the continuous GPS feed into consecutive trajectories at appropriate positions. We present several identification policies we have implemented for various trajectory scenarios.

POLICY 1 (RAW GPS GAP). *Divide the sequence of (x,y,t) GPS records into several spatio-temporal trajectories according to the GPS gaps that satisfy one of the following conditions:*

(*1*) *Given a large time interval $\Delta_{duration-large}$, if two consecutive GPS records, $p_i(x_i, y_i, t_i)$ and $p_{i+1}(x_{i+1}, y_{i+1}, t_{i+1})$, are such that the temporal gap $t_{i+1} - t_i > \Delta_{duration-large}$, then p_i is the ending point of the current trajectory whilst p_{i+1} is the starting point of the next trajectory.*

(*2*) *Given both a time interval $\Delta_{duration}$ and a spatial distance $\Delta_{distance}$, if two consecutive GPS records, p_i and p_{i+1}, are such that the temporal gap $t_{i+1} - t_i > \Delta_{duration}$ and the spatial gap $\sqrt{(x_{i+1} - x_i)^2 + (y_{i+1} - y_i)^2} > \Delta_{distance}$, then p_i is the ending point of the current trajectory whilst p_{i+1} is the starting point of the next trajectory.*

This policy utilizes the significant temporal (and spatial) gaps in the GPS feed for separating two consecutive spatio-temporal trajectories $\mathcal{T}_{spa}$. GPS trajectories often exhibit such gaps due to several reasons. For example, tracking devices usually turn off the GPS if the object does not move for a long while (to save power) or if there is no satellite coverage (indoor locations). The first sub-policy exploits large temporal gaps $\Delta_{duration-large}$ to extract $\mathcal{T}_{spa}$. This is typically relevant for vehicle movement scenarios. E.g., our dataset of 17,241 car GPS traces (2,075,213 GPS records) resulted in 83,134 spatio-temporal trajectories. The second sub-policy uses both temporal and spatial gaps, where the two parameters are determined by statistical analysis of GPS feeds (e.g., gap distribution, type of movement: vehicular, pedestrian etc).

POLICY 2 (PREDEFINED TIME INTERVAL). *Divide the stream of GPS feed into several subsequences contained in given time intervals, e.g., hourly trajectory, daily trajectory, weekly trajectory, monthly trajectory.*

This policy allows us to meaningfully divide a GPS feed into periods for analyzing mobility behaviors. Short-term period is particularly relevant for human movements (e.g., daily movement of weekday behavior analysis). Wildlife monitoring on the other hand needs to capture longer-term trajectory behaviors such as monthly or seasonal patterns (e.g., yearly movement analysis for the bird migration scenario).

POLICY 3 (PREDEFINED SPACE EXTENT). *Divide the stream of GPS feed into several subsequences according to a spatial criteria, e.g., fixed distance, geo-fenced regions, movement between predefined points for network constrained trajectories.*

This policy allows us to divide a GPS feed according to the covered distance (e.g., every 20 miles); according to a specific area (e.g., trajectories in EPFL campus, Lausanne downtown, or even Switzerland), where trajectories are defined when the object enters or exits the area; or between two given positions.

POLICY 4 (TIME SERIES SEGMENTATION). *Divide the stream of GPS feed into several subsequences according to a (semi-) automatic algorithm for segmenting time series, based on spatial or/and temporal correlations.*

Trajectory data in essence is a special kind of time series, where the values are the locations $\langle x, y \rangle$ as time flows. Therefore, conventional time series segmentation algorithms can be applied for trajectory identification. Keogh et al. [Keogh et al. 2004] categorizes time series segmentation methods into three types: sliding window, top-down, and bottom-up. We use these methods for time series based segmentation of the mobility data. Policy 2 and Policy 3 can be considered as sliding window-based methods, where the window is dynamically determined by the given temporal intervals or spatial areas. The top-down and bottom-up methods can generate much over-fragment of trajectories (i.e., a lot of small segments), which is not good for the *trajectory identification* step. Nevertheless they can be applied for the *trajectory structuring* step, e.g., the multi-dimensional mobile data segmentation [Guo et al. 2012].

The choice of the trajectory identification policy (from Policy 1 to Policy 4) depends on the application and data characteristics (e.g., with/without big gaps). For example, our people with smartphone trajectory data uses Policy 2 (daily trajectories); the taxi trajectories can be divided according to the Lausanne zone by using Policy 3, analyzing the inside-city and outside-city trajectories.

4.3. Trajectory Structure Layer

After identifying separate spatio-temporal trajectories, the next task is to compute their internal structures, constructing structured trajectories $\mathcal{T}_{str}$ that consist of meaningful episodes. The core issue in *trajectory structure* is to group consecutive GPS points into an episode. We have implemented *velocity*, *density*, *orientation* and *time series* based algorithms for identifying episodes. Hence, the focus is on the whole trajectory data computing platform. In this paper, we present the two representative methods, i.e., *velocity-based* and *density-based* trajectory structure.

In trajectory structure, we mainly focus on two kinds of episodes (i.e. *stops* and *moves*) due to their commonality in many trajectory applications. The idea is to determine whether a GPS point $p(x, y, t)$ belongs to a stop episode or a move episode by using a speed threshold (Δ_{speed}). Hence, *if the instant speed of p is lower than Δ_{speed}, it is a part of a stop, otherwise it belongs to a move*. Fig. 6 traces the speed evolution of a vehicle, showing how stops can be determined by a given Δ_{speed}. Besides Δ_{speed}, we also use a second parameter - *minimal stop time* τ in order to avoid false positives (e.g., short-term *congestions* with low velocity should not be stop episodes).

Determining a suitable value for Δ_{speed} is a challenging problem: *if Δ_{speed} is too high, many stops appear; on the contrary, if Δ_{speed} is too low, probably no stops are computed*. Fig. 6 simply shows a constant Δ_{speed} applied all across the trajectory. This is not practical in real-world scenarios, where the value of Δ_{speed} should rather be flexible according to the context of the moving object. For example, vehicles with different levels of performance (bicycles or motor cars), different road networks (on a highway or a secondary road path), different weather conditions (sunny or snowy days) call for diverse speed thresholds. Although it is possible to get this contextual information, it would substantially increase the number of information sources that need to be integrated. We take a different approach. We design a generic method for determining

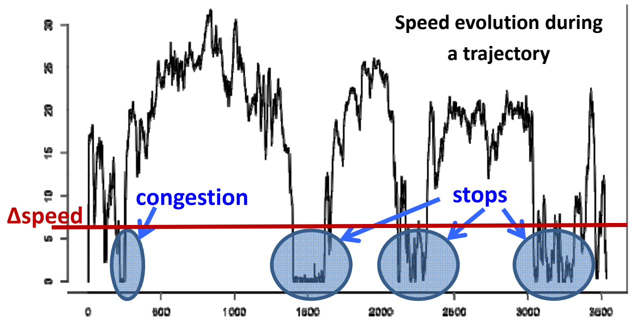


Fig. 6: Velocity-based Stop Identification

Δ_{speed}, based on the class of moving objects being monitored (which is available) and then aggregate statistics of other moving objects in the area of consideration.

DEFINITION 4 (DYNAMIC VELOCITY THRESHOLD - Δ_{speed}). *For each GPS point $Q(x,y,t)$ of a given moving object (obj_{id}), the Δ_{speed} is dynamically determined by the moving object (by using $\overline{objectAvgSpeed}$ – the average speed of this moving object) and the underlying context (by $\overline{positionAvgSpeed}$ – the average speed of most moving objects in this position $\langle x,y\rangle$); i.e., $\Delta_{speed} = min\{\delta_1 \times \overline{objectAvgSpeed}, \delta_2 \times \overline{positionAvgSpeed}\}$, where δ_1 and δ_2 are coefficients.*

In this definition, $\overline{objectAvgSpeed}$ is easy to calculate as the average speed of the moving object. Regarding $\overline{positionAvgSpeed}$, we need to approximate it by using space division. We divide the space into regular cells (or directly using the available landuse grid) and calculate the average speed in each cell $\overline{cellAvgSpeed}$ as the contextual information. For network-constrained trajectory data, we can apply the speed condition on the underlying network (e.g., the average passing speed of the nearest road crossing $\overline{crossingAvgSpeed}$ and the average passing speed of the map matched road segment $\overline{segmentAvgSpeed}$), instead of the $\overline{cellAvgSpeed}$. Algorithm 1 provides the pseudocode to determine Δ_{speed}. We analyze sensitivity of the coefficients δ_1 and δ_2 (e.g., $\delta_1 = \delta_2 = \delta = 30\%$) through experiments.

ALGORITHM 1: getDynamicΔ_{speed} (gpsPoint, obj_{id}, δ)

```
input  : gpsPoint $p = (x, y, t)$, moving object $obj_{id}$
output: dynamic speed threshold $\Delta_{speed}$
1  get the average speed of this moving object $obj_{id}$: $\overline{objectAvgSpeed}$;
2  if network-constrained trajectory then
3      get the average speed of the nearest road crossing to $p$: $\overline{crossingAvgSpeed}$;
4      get the average speed of the map matched road segment of $p$: $\overline{segmentAvgSpeed}$;
5      $positionAvgSpeed \leftarrow min\{\overline{crossingAvgSpeed}, \overline{segmentAvgSpeed}\}$
6  else
7      get the average speed of the cell that (x,y) belongs to: $\overline{cellAvgSpeed}$;
8      $positionAvgSpeed \leftarrow \overline{cellAvgSpeed}$
9  compute the dynamic speed threshold by Definition 4;
10 return $\Delta_{speed}$
```

In some scenarios, GPS tracking data have instant speed values (s) captured by the devices. We use them for calculating Δ_{speed} and identifying the stops; otherwise, s is approximated by the average speed between the previous spatio-temporal point

$(x_{i-1}, y_{i-1}, t_{i-1})$ and the next one $(x_{i+1}, y_{i+1}, t_{i+1})$, i.e. $s_i = \frac{\|\langle x_{i+1}, y_{i+1}\rangle - \langle x_{i-1}, y_{i-1}\rangle\|_2^2}{t_{i+1}-t_{i-1}}$. This is possible as GPS data is usually sampled frequently (e.g., few samples per min).

Alg. 2 summarizes *velocity-based trajectory structure*: firstly, we compute the instant speed if it is not available from GPS devices; secondly, we compute the dynamic Δ_{speed} (using Algorithm 1) and annotate the GPS point with 'M' or 'S' tag; finally, stops and moves are computed by aggregating all consecutive points with the same tag, with a precondition on the minimal stop duration τ. This algorithm has linear complexity on the size of GPS feed, together with linear complexity on the size of road segments in the underlying network. It currently performs two data scans while tagging points and grouping consecutive points for the episodes. However, it is possible to combine the two scans together for better performance and shorten the computing time.

ALGORITHM 2: ***Velocity-based trajectory structure***

```
Input: a raw trajectory $\mathcal{T}_{raw} = \{p_1, p_2, \cdots, p_n\}$
Output: a structured trajectory $\mathcal{T}_{str} = \{e_1, e_2, \ldots, e_m\}$ where $e_i$ is a tagged trajectory episode (stop $\mathcal{S}$ or
        move $\mathcal{M}$)
begin
    /* initialize: calculate GPS instant speed if needed */
    ArrayList$\langle x, y, t, tag\rangle$ $gpsList \leftarrow$ getGPSList($\mathcal{T}_{spa}$);
    if no instant speed from GPS device then
        compute GPS instant speed $s_i$ for all $p_i = (x, y, t) \in gpsList$;
    /* episode annotation: tag each GPS point with 'S' or 'M' */
    forall the $p_i = (x, y, t) \in gpsList$ do
        // get dynamic $\Delta^{(i)}_{speed}$ by Algorithm 1
        $\Delta^{(i)}_{speed} \leftarrow$ getDynamic$\Delta_{speed}$ ($p$, $obj_{id}$, $\delta$);
        // tag GPS point as a stop point 'S' or a move point 'M'
        if instant speed $s_i < \Delta^{(i)}_{speed}$ then
            tag current point $p_i(x, y, t)$ as a stop point 'S';
        else
            tag current point $p_i(x, y, t)$ as a move point 'M';
    /* compute episodes: grouping consecutive same tags*/
    forall the consecutive points with the same tag 'S' do
        // compute stop episode
        get the total time duration $t_{interval}$ of these points;
        if $t_{interval} > \tau$ the minimal possible stop time then
            $stop \leftarrow (time_{from}, time_{to}, center, boundingRectangle)$;
            $\mathcal{T}_{str}$.($stop$, 'S'); // add the stop episode
        else
            change the 'S' tag to 'M' for all these points; // as "congestion"
    forall the consecutive points with the same tag 'M' do
        // compute move episode
        $move \leftarrow (stop_{from}, stop_{to}, duration)$ // create a move episode
        $\mathcal{T}_{str}$.($move$, 'M'); // add the move episode
    return the structured trajectory $\mathcal{T}_{str}$;
```

Using only velocity for identifying stops is not enough for some applications. For example when analyzing bird migrations, we need to find the foraging stops. Some birds, like water-birds, when they are looking for food, can fly at high speed, but inside a small area. Another example is in traffic applications, when someone is driving quickly around a block looking for a parking place. The velocity-based algorithm cannot detect these kinds of stops. Therefore, we designed *density-based* stop identification, which considers not only the speed but also the maximum distance that the moving object has traveled during a given time duration. For this algorithm, we need to define density areas for extracting stop or move episodes.

DEFINITION 5 ($\mathcal{A}_{density}$ - DENSITY AREA). *Given a cleaned sequence of GPS points $\{\langle x_i, y_i, t_i\rangle\}$, a maximum distance σ, and a time duration τ, a density area $\mathcal{A}$ is a sub-sequence of the GPS points $\{\langle x_{i1}, y_{i1}, t_{i1}\rangle, \ldots, \langle x_{im}, y_{im}, t_{im}\rangle\}$ that satisfies two conditions:*

1) *For any two different points of the density area, if they are temporally distant by less than τ then they are spatially distant by less than σ, i.e. $\forall\ \langle x_{ia}, y_{ia}, t_{ia}\rangle, \langle x_{ib}, y_{ib}, t_{ib}\rangle \in \mathcal{A}$, $\|t_{ib} - t_{ia}\| \leq \tau \Rightarrow \|\langle x_{ia}, y_{ia}\rangle - \langle x_{ib}, y_{ib}\rangle\| \leq \sigma$*
2) *For the last (first) point of the GPS sequence that is just before (after) the density area, say $\langle x_b, y_b, t_b\rangle$ ($\langle x_a, y_a, t_a\rangle$), there exists a point inside the density area, which is temporally distant by less than τ and spatially distant by more than σ, i.e. $\exists \langle x', y', t'\rangle \in \mathcal{A}$ $\|t' - t_b\| \leq \tau$ and $\|\langle x', y'\rangle - \langle x_b, y_b\rangle\| > \sigma$ ($\|t_a - t'\| \leq \tau$ and $\|\langle x_a, y_a\rangle - \langle x', y'\rangle\| > \sigma$)*

Both velocity-based and density-based trajectory structure methods annotate each GPS point $\langle x, y, t\rangle$ with 'M' or 'S'. Stops and moves are then computed based on contiguous 'M'/'S' tags, together with the begin/end tags ('B'/'E') resulting from the trajectory segmentation layer. Thus, a continuous sequence of $\langle x, y, t\rangle$ points having all 'M' tags is integrated into a single *move*, whilst, a continuous sequence of $\langle x, y, t\rangle$ points, all with 'S' tags, is integrated into a single *stop*. The first and last $\langle x, y, t\rangle$ point of each trajectory are respectively computed as its Begin and End.

Further details of all our approaches, including time series for network-constrained trajectory modeling Traj-ARIMA (Autoregressive Integrated Moving Average) are presented in [Yan 2010]. We use Traj-ARIMA for velocity fitting and prediction. Furthermore, we apply it for stop identification in situations where the forecasted speed is very different from the real speed, as there might be a stop happening.

5. TRAJECTORY ANNOTATION

The trajectory computation layers developed different levels of data abstraction, reconstructed trajectories as a sequence of highly-correlated episodes, resulted in structured trajectories $\mathcal{T}_{str}$. To better understand trajectory semantics, the meanings of the trajectory episodes need to be further discovered. For e.g., one episode is at *home*, another episode is on a public transportation (say bus) from *home* to *office*, as the semantic trajectory shown on the top of Fig. 7. Therefore, 3rd party geographic information sources like landuse distribution, road network from Openstreetmap are needed for obtaining such semantic enrichment. These semantic annotations are captured using the *Semantic Trajectory* model introduced earlier in Section 3. This section describes the design and details of the annotation platform.

Our objective here is to provide a uniform and generic annotation platform for enriching trajectories with multiple geographic artifacts. To accommodate heterogeneity of 3rd party geographic information sources, we categorize them into three categories, i.e. Regions of Interest (ROI), Lines of Interest (LOI), and Points of Interest (POI), according to their geometric shapes. We entitle them *semantic places*.

DEFINITION 6. *Semantic Places ($\mathcal{P}$) - A set of meaningful places for annotating and understanding mobility data. Each place sp has additional attributes containing useful metadata information $(a_1, a_2, \cdots, a_n)$ for describing such place. There are basically three subsets according to the geometric shape, i.e. $\mathcal{P} = \mathcal{P}_{region} \bigcup \mathcal{P}_{line} \bigcup \mathcal{P}_{point}$,*

— *a set of semantic regions, $\mathcal{P}_{region} = \{r_1, r_2, \cdots, r_{n_1}\}$*
— *a set of semantic lines, $\mathcal{P}_{line} = \{l_1, l_2, \cdots, l_{n_2}\}$*
— *a set of semantic points, $\mathcal{P}_{point} = \{p_1, p_2, \cdots, p_{n_3}\}$*

We have identified and redesigned (particularly for line and point annotation) widely applicable algorithms considering our objective - algorithms should exhibit good per-

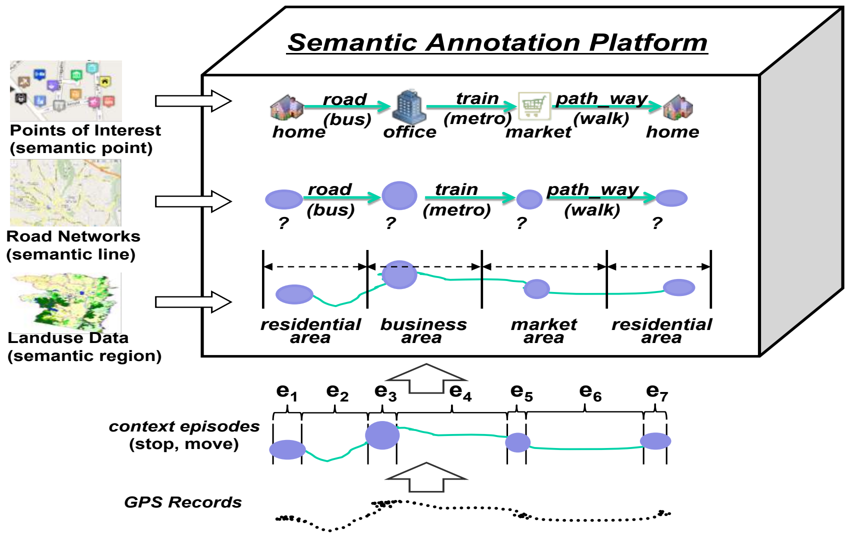

Semantic Annotation Platform
Points of Interest (semantic point)
Road Networks (semantic line)
Landuse Data (semantic region)
home
road (bus)
office
train (metro)
market
path_way (walk)
home
?
residential area
business area
market area
residential area
e1 e2 e3 e4 e5 e6 e7
context episodes (stop, move)
GPS Records

Fig. 7: Trajectory Annotation Platform

formance over a wide range of trajectories with varying data quality. We follow a layered approach, carefully designed to support efficient semantic annotation. We first apply *spatial join* for computing $\mathcal{T}_{sem}^{(region)}$ (a sequence of regions) with ROIs (e.g., landuse data), to pick up regions that the trajectory has passed through, primarily to form a coarse-grained view of the semantic movement. We design a *semantic line annotation* algorithm that annotates *move* episodes, computing $\mathcal{T}_{sem}^{(line)}$ (a sequence of semantic moves) using LOIs (e.g., road network). For $\mathcal{P}_{point}$, we design a *hidden Markov model* (*HMM*) based algorithm for annotating *stop* episodes, computing $\mathcal{T}_{sem}^{(point)}$ with POIs (i.e. home, office, shopping mall, restaurant etc).

5.1. Annotation with Semantic Regions

This layer enables annotation of trajectories with meaningful geographic regions. It does so by computing topological correlations of trajectories with 3rd party data sources containing semantic places of spatial kind regions ($\mathcal{P}_{region}$).

The topological correlation is measured using *spatial join* between a trajectory $\mathcal{Q}$ and semantic regions $\mathcal{P}_{region}$ (i.e. $\mathcal{Q} \bowtie_\theta \mathcal{P}_{region}$). Several forms of spatial predicates are used to compute θ, depending on the type of data. These can be a combination of *directional*, *distance*, and *topological* spatial relations (e.g., *intersection*) [Brinkhoff et al. 1993]. E.g. for *stop* episodes, we found spatial subsumption (ObjectA is *inside* ObjectB) as the most used predicate. For the spatial extent, we use either the spatial *bounding rectangle* of the episode (for move or stop) or its *center* (for stop) to perform spatial join. After finding the appropriate regions (r_i), the layer annotates input trajectories with these regions and associated metadata.

The semantic regions can be free form regions like the EPFL campus, a recreation facility with a swimming pool, both taken from Openstreetmap[5], and regions formed from grids of regular cells of repositories such as the Swisstopo[6] landuse and city zones. Fig. 8 shows one person's trajectory on Sunday, annotated with semantic places of vari-

[5]http://www.openstreetmap.org
[6]http://www.swisstopo.admin.ch/

ous kinds taken from Swisstopo (building area, recreational area) and Openstreetmap (EPFL campus). By using an application database (e.g., EPFL's employee database) annotations for this personal trajectory can be expressed as: *home* $\rightarrow$ *EPFL campus (staying 4 hours)* $\rightarrow$ *a swimming pool (staying 1 hour)* $\rightarrow$ *home*.

Fig. 9 illustrates landuse classification categories and subcategories that Swisstopo uses to annotate 1,936,439 cells ($100\text{m}\times 100\text{m}$) covering Switzerland. Fig. 10 is an example of annotating trajectories with such landuse cells.

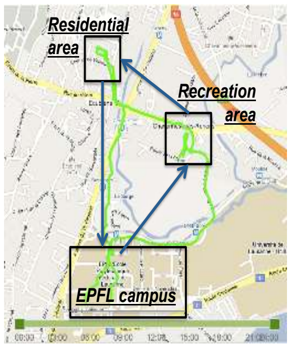


Fig. 8: Region annotation

L1 Settlement and urban areas
1.1 industrial and commercial area
1.2 building areas
1.3 transportation areas
1.4 special urban areas
1.5 recreational areas and cemeteries
L2 Agricultural areas
2.6 orchard, vineyard and horticulture areas
2.7 arable land
2.8 meadows, farm pastures
2.9 alpine agricultural areas
L3 Wooded areas
3.10 forest (except brush forest)
3.11 brush forest
3.12 woods
L4 Unproductive areas
4.13 lakes
4.14 rivers
4.15 unproductive vegetation
4.16 bare land
4.17 glaciers, perpetual snow

Fig. 9: Landuse Ontology

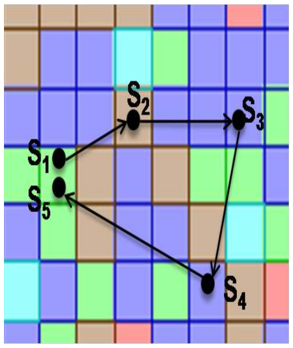


Fig. 10: Landuse

ALGORITHM 3: *Trajectory annotation with ROIs*

```
Input: (1) a raw trajectory $\mathcal{Q}$ with its sequence of GPS points $\{Q_1, \cdots, Q_n\}$, (2) a set of
       semantic regions $\mathcal{P}_{region} = \{region_1, \cdots, region_{n_1}\}$
Output: structured semantic trajectory $\mathcal{T}_{region}$
begin
    $\mathcal{T}_{region} \leftarrow \emptyset$; //initialize the trajectory
    /* compute intersections between $Q$ and $\mathcal{P}_{region}$; */
    do spatial joins $\mathcal{Q} \bowtie_{intersect} \mathcal{P}_{region}$;
    /* process each intersection and compute trajectory tuple */
    forall the intersected regions do
        group continuous GSP point $Q_i \in \mathcal{Q}$ in the intersection;
        approximate entering time $t_{in}$ and leaving time $t_{out}$;
        create a trajectory $tuple \leftarrow (region_j, t_{in}, t_{out}, reg_{type})$;
        if current $reg_{type}$ = previous $reg_{type}$ then
            merge the two tuples into a single tuple ;
        else
            $\mathcal{T}_{region}$.add($tuple$); //add the previous tuple to $\mathcal{T}_{region}$;
    $\mathcal{T}_{region}$.add($tuple$); //add the last tuple to $\mathcal{T}_{region}$;
    return trajectory $\mathcal{T}_{region}$
```

Alg. 3 shows the pseudocode of the annotation algorithm with regions, which directly annotates GPS records with regions. Note that, depending on requirements, the spatial join can be computed only for selected episodes. We apply R*-tree index on semantic regions $\mathcal{P}_{region}$ [Beckmann et al. 1990] to improve efficiency of the algorithm. The complexity of the annotation algorithm with region is $O(n * log(m))$, where n is the number of GPS records (or stop episodes) whilst m is the size of $\mathcal{P}_{region}$. For well-divided landuse data, the complexity can be even less, i.e. $O(n)$.

5.2. Annotation with Semantic Lines

This layer annotates trajectories with LOIs and considers variations present in heterogeneous trajectories (e.g., vehicles run on road networks, human trajectories use a combination of transport networks and walk-ways etc). Given data sources of different form of road networks, the purpose is to identify *correct* road segments as well as infer *transportation modes* such as *walking*, *cycling*, *public transportation like metro*. Thus, the algorithms in this layer include two major parts: the first part is designing a global map matching algorithm to identify the correct road segments for the move episodes, and the second one is inferring the transportation mode that the moving object used.

Map-matching algorithms usually design a distance metric (e.g., *perpendicular distance*) to map the GPS points to the nearest road segment [Quddus et al. 2007]. Though suitable for well-defined high-way networks, perpendicular distance is not suitable for dense networks, parallel road-ways and arbitrary crossings. This is because vertical projections of (x,y,t) points on corresponding road segments often do not fall on the segment. Thus, we apply the *point-segment distance*, defined as:

$$d(Q, A_iA_j) = \begin{cases} d(QQ') & \text{if } Q' \in A_iA_j \\ min\{d(QA_i), d(QA_j)\} & \text{otherwise} \end{cases} \tag{2}$$

where Q' is the projection of the GPS point Q on the line determined by the two crossings A_i and A_j; $d(QQ')$ is the perpendicular distance between Q and that line; $d(QA)$ is the Euclidean distance between Q and the crossing A.

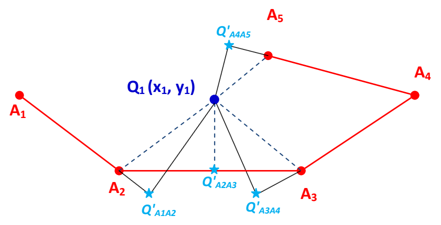


Fig. 11: Point-segment distance

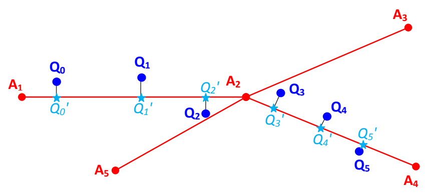


Fig. 12: Global Map-Matching

As a subsequence of raw trajectory Q, a *move* episode also includes a list of spatio-temporal points. Choosing the candidate road segment for each single point independently sometimes results in incorrect mapping, specially for non-perpendicular path ways. Global map matching algorithms have shown better matching quality [Brakatsoulas et al. 2005][Quddus et al. 2007] as they consider the context of neighboring points. We adopt this with the *point-segment distance*, in terms of designing two metrics (*localScore* and *globalScore*) to map *move* episodes to appropriate road segments for heterogeneous road structures.

We consider a *global view radius* R around candidate points, with a context window of size $2R$. Therefore, mapping results of point Q depend also on the effects of its neighboring points (N_1 points before and N_2 points after in radius R). For computational efficiency, only the *neighboring* segments are considered as candidate road segments $candidateSegs(Q)$. They can be efficiently accessed with R*-tree index [Beckmann et al. 1990]. We normalize the point-segment distance $d(Q, A_iA_j)$ as the $localScore$ between point Q and road segment A_iA_j.

$$localScore(Q, A_iA_j) = \begin{cases} \frac{d_{min}(Q)}{d(Q,A_iA_j)} & A_iA_j \in candidateSegs(Q) \\ 0 & \text{otherwise} \end{cases} \quad (3)$$

where $d_{min}(Q)$ is the shortest distance from Q to all possible candidate road segments A_iA_j. Based on *localScore*, we compute a global measurement - *globalScore* - between Q and A_iA_j considering the context window $2R$ containing N_1 points prior to Q and the forthcoming N_2 points.

$$globalScore(Q, A_iA_j) = \frac{\sum_{k=-N_1}^{N_2} w_k \cdot localScore(Q_k, A_iA_j)}{\sum_{k=-N_1}^{N_2} w_k} \quad (4)$$

$$w_k = \begin{cases} exp(-\frac{d(Q_0Q_k)^2}{2\sigma^2}) & d(Q_0Q_k) < R \\ 0 & \text{otherwise} \end{cases} \quad (5)$$

where Q_k is the k^{th} neighboring point of Q (e.g., Q_0 is Q itself, Q_{-1} is the previous point whilst Q_{+1} is the next point); w_k is the corresponding weight determined by a Kernel smoothing function with the Kernel bandwidth σ.

After the first step of the global map matching, each episode is annotated in terms of a list of road segments, i.e. $ep = \{r_1, r_2, \ldots, r_l\}$. We further infer the annotation of transportation mode on each segment (or route), getting the pairs of $\langle r_i, mode_i\rangle$. In our experiment, we consider four types of transportation modes, i.e. *walking*. *bicycle*, *bus* and *metro*. Such annotation is determined by the characteristics of the move episode and the matched road segments, including *average velocity*, *average acceleration*, *road type* etc.

Alg. 4 shows the detailed procedure of semantic line annotation: (1) select candidate road segments, (2) calculate the point-segment distance, (3) normalize the distance as $localScore$, (4) compute the weight and calculate $globalScore$, (5) determine the map matching segment for each point based on $globalScore$, (6) further infer the transport mode based on the features of the segment and the road type information.

Since each GPS point considers only the neighboring road segments as a set of candidate segments (by R*-tree), the candidate set size is significantly smaller than the total size of road networks in real-life datasets. This makes the algorithm, besides having better matching quality, also efficient, with linear complexity on the size of the GPS points $O(n)$. The global map matching parameters (e.g., radius R and kernel width σ) are tuned in the experiment.

5.3. Annotation with Semantic Points

This layer annotates the *stop* episodes of a trajectory with information about plausible *points of interest* (POIs). Examples of POI are *Gino restaurant*, *Armani shop Via Manzoni* etc. For scarcely populated areas, it is trivial to identify the POI that is the goal of a stop (e.g., the goal of a stop at a highway petrol pump is the petrol pump itself). However, densely populated urban areas may have many candidate POIs for each stop. Further, low GPS sampling rate due to battery outage and signal losses makes the problem more intricate. For instance, the Milan dataset in our experiments has 39,772 POIs with largely varying density. This large number makes it probabilistically intractable to infer the exact POI of the stop from imprecise location records. So, instead of inferring the POI instance for each stop, we chose to infer some semantic characteristic of the POI that is important for the applications. For instance we can infer the POI type (e.g. restaurant, shop) or the activity usually performed in the POI (e.g. eating, shopping). We tested our method on the Milan dataset whose POIs are organized into

ALGORITHM 4: ***Trajectory annotation with LOIs***

```
Input: (1) a move episode of raw trajectory $\mathcal{Q}$ of GPS points $\{Q_i(x_i, y_i, t_i)\}$
       (2) a set of road segments $P_{line} = \{r_1, r_2, \cdots, r_m\}$
Output: semantic trajectory $\mathcal{T}_{line}$
begin
    $preSeg \leftarrow \emptyset$, $\mathcal{T}_{line} \leftarrow \emptyset$; //initialize the trajectory
    forall the $Q_i = (x, y, t) \in \mathcal{Q}$ do
        /* select candidate roads for $Q_i$ (R*-tree)*/
        $candidateSegs(Q_i) \leftarrow \{r_1^{(i)}, \cdots, r_n^{(i)}\}$; // select only neighboring road segments
        /* calculate dist., normalize it as localScore */
        compute the distance between point $Q_i$ and $\forall r_j^{(i)} \in candidateSegs(Q_i)$;
        choose the closest segment $min\{d(Q_i, r_j^{(i)})\}$ (Equ. 2);
        normalize distance as $localScore(Q_i, r_j^{(i)})$ $\forall r_j^{(i)} \in candidateSegs(Q_i)$ by Formula 3;
        /* calculate globalScore: (point, segment) */
        choose global points $(Q_{-N_1}, \cdots, Q_{+N_2})$ in radius $R$;
        compute their Kernel smoothing weights by Formula 5;
        compute the $globalScore(Q_i, r_j^{(i)})$ for $\forall r_j^{(i)} \in candidateSegs(Q_i)$ by Formula 4;
        /* compute $Q'$ with road position (if needed) */
        rank the computed $globalScore(Q_i, r)$
        choose the highest score to match $segmentId$ for $Q_i$;
        compute the corrected position $(x', y')$ if needed ;
        /* add road segment as a trajectory tuple */
        if $preSeg \neq null$ and $preSeg \neq segmentId$ then
            /* infer transportation mode */
            get $tranportMode$ by velocity distribution, road information etc.
            /* add the semantic episode */
            $(segmentId, time_{in}, time_{out}, mode) \rightarrow \mathcal{T}_{line}$;
            $preSeg \leftarrow segmentId$;
    return structured semantic trajectory $\mathcal{T}_{line}$
```

a hierarchy according to their category for the local administration. The top level of the hierarchy contains five generic categories: *services, food, home_item, personal_item,* and *other*. So inferring the category of the stop out of these five categories becomes a tractable problem.

Therefore we have designed a *Hidden Markov Model* (*HMM*) based technique for the semantic annotation of *stops* with POI category. Unlike most other algorithms that identify the POIs of the stops [Alvares et al. 2007][Xie et al. 2009], an unique novelty of our approach is that it works for densely populated areas with many possible POI candidates for annotation, thus catering to heterogeneous people and vehicle trajectories.

HMM is a classical statistical signal model in which the system being modeled is assumed to be a Markov process with unobserved state [Rabiner 1990]. We consider the *temporal sequence* of GPS stops: $\mathcal{S} = (S_1, S_2, \cdots, S_n)$ as the observed values.

Fig. 13 expresses the resultant HMM problem. The initial input is the raw trajectory $\mathcal{Q}$, i.e. the sequence of (x,y,t) points; A *sequence of stops* is computed and forms the real observation (O); The POI *instances* are the superficial hidden states, whilst the POI *categories* are the real hidden states that we are interested in. Our goal is to identify the real hidden states and use them to annotate the stops.

Modeling: Let there be *m* POI categories $C_1 \ldots C_m$. Typically, a HMM λ has three major components, i.e. $\lambda = (\pi, \mathcal{A}, \mathcal{B})$; where π is the probability of the initial states, i.e. $Pr(C_i)$, $\mathcal{A}$ is the state transition probability matrix ($[Pr(C_j|C_i)]_{m \times m}$), $\mathcal{B}$ is the observation probability for each state $Pr(o|C_i)$.

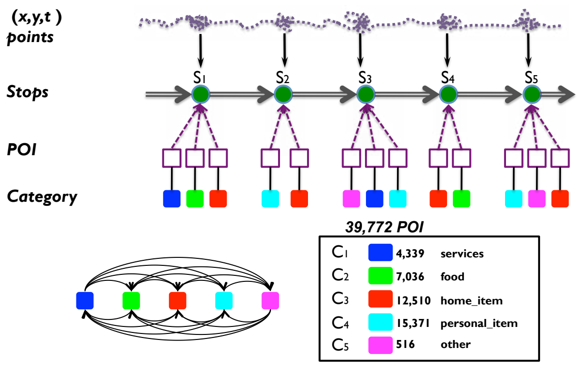


Fig. 13: HMM formalism for inferring POI category

$$\mathcal{A}=\begin{pmatrix} 0.8 & 0.05 & 0.05 & 0.05 & 0.05 \\ 0.05 & 0.8 & 0.05 & 0.05 & 0.05 \\ 0.05 & 0.05 & 0.8 & 0.05 & 0.05 \\ 0.05 & 0.05 & 0.05 & 0.8 & 0.05 \\ 0.15 & 0.15 & 0.15 & 0.15 & 0.4 \end{pmatrix}$$

Fig. 14: Example state transition matrix

— **Initial Probabilities** (π). We approximate the probability of initial states π as the percentage of POI samples belonging to each category from the information source. Therefore, for Milan POI dataset,

$$\pi = \{\frac{4339}{39772}, \frac{7036}{39772}, \frac{12510}{39772}, \frac{15371}{39772}, \frac{516}{39772}\}$$

— **State Transition** ($\mathcal{A}$). State transition probability $Pr(C_j|C_i)$ in our formulation represents the possible sequences of stop categories; i.e. the probability to stop in a POI of category C_j given that the previous stop was in a POI of category C_i. Wherever available, category sequences (e.g., *food* $\rightarrow$ *items for people* or *food* $\rightarrow$ *other*) are obtained through other information sources (e.g., from *region* transitions). For trajectories having insufficient history, we initialize the state transition matrix following nomenclatures of the POI categories (e.g., associate high probability for meaningful state transitions and low probabilities for non-meaningful state transitions in Fig. 14). *Learning* dynamic and personalized transition matrix $\mathcal{A}$ is interesting but not the focus of this paper.

— **Observation Probabilities** ($\mathcal{B}$). $Pr(o|C_i)$ intuitively represents the probability of seeing a *stop o* (as the observation) in $\mathcal{T}$ *caused by* user's interest in places belonging to category C_i. $Pr(o|C_i)$ can be approximated by using the center of the *stop* $Pr(center_{xy}|C_i)$ or the bounding rectangle $Pr(boundRectangle|C_i)$.

Computing $\mathcal{B}$ for areas having high POI density is not easy. Our solution is based on the intuition that the influence of a POI category on a *stop* is proportional to the number of POI instances of that category in the stop area. We model the influence of a POI as a two-dimensional Gaussian distribution - the mean is the POI's physical position (x, y) and the variance is $[\sigma_c^2, 0; 0, \sigma_c^2]$, where σ_c is category specific. Fig. 15 displays an example of 12 POIs' Gaussian distributions with the corresponding densities in Fig. 16. By Bayesian rule, we deduce the lemma to determine $Pr(o|C_i)$ in $\mathcal{B}$.

LEMMA 1. *$Pr(o|C_i)$ is proportional to the sum of the probability of each POI that belongs to this category C_i, namely $Pr(o|C_i) \propto \Sigma_j Pr(o|poi_j^{(C_i)})$.*

PROOF. of Lemma 1

$$Pr(o|C_i) = \frac{Pr(o, C_i)}{Pr(C_i)} = \frac{\Sigma_j Pr(o, poi_j^{(C_i)})}{\Sigma_j Pr(poi_j^{(C_i)})} = \frac{\Sigma_j Pr(o|poi_j^{(C_i)}) Pr(poi_j^{(C_i)})}{\Sigma_j Pr(poi_j^{(C_i)})}$$
$$\propto \Sigma_j Pr(o|poi_j^{(C_i)}) Pr(poi_j^{(C_i)}) \propto \Sigma_j Pr(o|poi_j^{(C_i)})$$

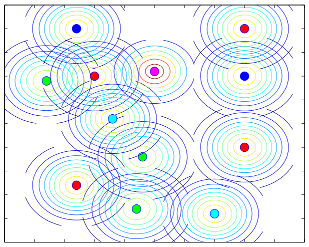


Fig. 15: POI distribution

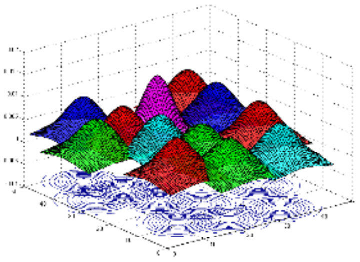


Fig. 16: POI densities

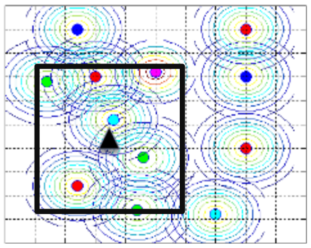


Fig. 17: Discretization

□

We employ *discretization* and *neighboring* techniques to improve the efficiency of computing $Pr(o|C_i)$. Using *discretization*, we divide the area of POIs into grids (jk) and pre-compute discretized probability values of $Pr(grid_{jk}|C_i)$, as the approximation of $Pr(center_{xy}|C_i)$. Further, for each $grid_{jk}$, we consider only *neighboring* POIs in that box (black rectangle in Fig. 17), instead of all the POIs in the area.

Inferring Hidden States: Using the above defined complete form HMM $\lambda = (\pi, \mathcal{A}, \mathcal{B})$, we infer their hidden states (the purpose behind the stops) $HS = \{pc_1, pc_2, \cdots, pc_n\}$ from the stop sequence $OV = \{stop_1, stop_2, \cdots, stop_n\}$ available through the stop/move computation; where pc_t is the POI category $pc_t \in \{C_1, \cdots, C_m\}$. This problem can be formalized as maximizing the likelihood $\mathcal{L}(HS|OV, \lambda)$.

We redefine this problem as a *dynamic programming* problem, defining $\delta_t(i)$ as the highest probability of the t^{th} *stop* caused due to POI category C_i (Formula 6). Formula 7 gives the corresponding induced form of highest probability at the $(t+1)^{th}$ stop for category C_j, considering the state transition probabilities. We record the previous state C_i that gives the highest probability to current state C_j by $\psi_{t+1}(j)$ (Formula 8).

$$\delta_t(i) = \max_i Pr(pc_1, \cdots, pc_t = C_i, o_1, \cdots, o_t|\lambda) \tag{6}$$

$$\delta_{t+1}(j) = \max_i \{\delta_t(i)A_{ij}\} \times B_j(o_{t+1}) \tag{7}$$

$$\psi_{t+1}(j) = \operatorname*{argmax}_i \quad \delta_t(i)A_{ij} \tag{8}$$

Finally, we employ the Viterbi algorithm [Forney 1973] to solve this dynamic programming problem for inferring the hidden state (stop category) sequence. We first recursively compute $\delta_t(i)$, and deduce the final stop state with the highest probability in the last stop, then backtrack to the previous stop state by $pc^*_{t-1} = \psi_t(pc^*_t)$. The details of the algorithm for inferring hidden stop category sequence is in Algorithm 5. The output of this layer is a sequence of semantic episodes describing the stops. The results from the three annotation algorithms are combined to produce the final semantic trajectory, which is exposed to applications.

6. EXPERIMENT ANALYSIS

Testing and evaluation is an extremely important phase in a research project. This section discusses in detail how we addressed these tasks. We used several sets of GPS trajectories produced by three kinds of moving objects: private cars, taxis and people with smartphones. Our choice was primarily driven by the availability of the data sets, secondarily by the fact that the chosen objects have different mobility patterns, which makes the testing more significant.

ALGORITHM 5: *Trajectory annotation with POIs*

```
Input: (1) an observation sequence of stops
    $O = \{Stop_1, Stop_2, \cdots, Stop_n\}$; (2) points of interest
    $POIs = \{\langle p_1, q_1\rangle, \cdots, \langle p_k, q_k\rangle\}$ where $q_i \in \{C_1, \cdots, C_5\}$
Output: a hidden state sequence about stop behaviors (in terms of POI categories), i.e.
        $S = \{q_1, q_2, \cdots, q_n\}, q_i \in \{C_1, \cdots, C_5\}$
1  begin
2      /* learn the model from POIs */
3      $\lambda = (\pi, \mathcal{A}, \mathcal{B})$
4      /* initialization */
5      forall the POI category $C_i$ do
6          $\delta_1(i) = \pi_i B_i(o_1), 1 \le i \le N; \psi_1(i) = 0$
7      /* recursion */
8      forall the $t$: 2 to $n$ do
9          forall the categories $C_j$ do
10             $\delta_t(j) = \max_i[\delta_{t-1}(i)A_{ij}] \times B_j(o_t)$
11             $\psi_t(j) = \operatorname{argmax}_i[\delta_{t-1}(i)A_{ij}]$
12     /* termination */
13     $P^* = \max_i[\delta_T(i)]; q_n^* = \operatorname{argmax}_i[\delta_T(i)]$
14     /* state sequence backtracking */
15     forall the $t$: $n$ to 2 do
16         $q_{t-1}^* = \psi_t(q_t^*)$
17     /* get the semantic trajectory with POI tags */
18     $S = \{\langle stop_1, q_1\rangle, \cdots, \langle stop_n, q_n\rangle\}$
19     summarize $\mathcal{T}_{point}$ from extracted POI sequence ($\langle stop, t_{in}, t_{out}, tagList\rangle$).
20     return structured semantic trajectory $\mathcal{T}_{point}$
```

Rigorous validation of automatic inferencing of semantic data against actual human behavior is inherently challenging. In particular, knowing where people have been doesn't readily tell us why they went to that place and what they did there. Validation, strictly speaking, relies on comparing computed results against the corresponding ground truth. Unfortunately only the moving person knows the truth, i.e. what (s)he was doing and why. We can ask a person to annotate his/her trajectories with ground truth (e.g. the performed activity), but this is only feasible for small data sets (cf. section 6.6). In other application domains, e.g. animal monitoring, it is simply not possible to acquire ground truth data (we cannot ask animals to tell us what they were doing).

Whenever ground truth is not available, the existence of statistical data may be used as a weaker yet interesting alternative [Bamis et al. 2010]. We followed this strategy for the taxi and Milan data sets (see section 6.5). Using statistical evidence means that we cannot guarantee correctness of inferences for each individual trajectories, but we can globally evaluate our results based on their statistical likelihood of correctness.

Finally, it is possible to have a by-definition validation strategy, i.e. ensuring that the inference algorithms cannot produce incorrect results. To this extent, we have to define a set of inference rules that we know will by definition lead to a correct interpretation of the semantics we are looking for. Imagine, for example, that we can extract the following facts about a trajectory: the person has spent two hours in a department store, and the person bought several items using her credit card. These facts let us conclude that the persons activity during this time interval is shopping. Much of the required inference rules rely on the availability of external knowledge complementing the trajectory data. Obviously there are things that cannot be inferred. For example, given the limitations of GPS data we cannot infer that the person visiting a commercial centre has bought this item from this shop and not that item from the nearby shop (unless video recordings are available). Instead, provided we have the necessary knowledge, it is possible to identify whether a person is in a commercial area for work, shopping, meeting, thanks to the fact that behaviors of workers, shoppers and meeting participants are quite different. The good news is that in most cases peoples behavior is predictable or inferable looking at their habits and considering common sense rules (e.g., a person stopping at a restaurant from 10am to 4pm is likely to be an employee rather than a customer).

6.1. Experiment Setup

Fig. 18 presents the architecture of our semantic trajectory platform, positioned between raw trajectory data and applications. It follows a layered structure that progressively abstracts higher-level semantic trajectory concepts from lower-level raw GPS feeds. We first compute trajectory episodes (stops/moves) from GPS feeds, by the *trajectory computation* layer; then the *trajectory annotation* layer with dedicated algorithms is designed for specific episodes (i.e., spatial join with landuse for both stops/moves, map matching based transportation inference with network for moves, and hidden Markov model using POIs for inferring stop behaviors). In addition to these two layers, extra layers are set up: (1) The *Trajectory Analytics Layer* computes statistical information (e.g., distribution of trajectory and episode characteristics, e.g., the mean, variance, max, min velocity). (2) The *Web Interface* presents users with a visual and integrative way to query and retrieve the mobility data at several abstracted levels, i.e., the enriched semantic trajectories as well as the raw mobility traces.

We implemented and deployed our platform on a Linux operating system - Ubuntu 9.10, with the Intel(R) 2$\times$3.00GHz CPU and 7.9GiB memory. The algorithms are implemented in Java 6. PostgreSQL 8.4, with the spatial extension PostGIS 1.5.1, is used for implementing the database stores. The raw GPS records and geographic information from 3rd party sources are loaded into databases and queried by the various layers during execution time. The trajectory Web interface is deployed on Apache Tomcat. Users access the system via a Web browser with the Google Earth Plugin.

6.2. Trajectory and Geographic Dataset

There are two types of datasets: one records fast-moving vehicle trajectories (e.g., taxi and private cars); another is people trajectories from smartphones with embedded GPS. The datasets related to vehicle trajectories are as follows (see Table I for details):

— **Trajectories**: We consider two large GPS datasets of vehicle trajectories, a small benchmark dataset for testing map matching, and two public datasets for sensitivity analysis of the trajectory computation layer: (1) 3 millions GPS records of two Lausanne taxis, collected over 5 months by Swisscom [7]; (2) 2 millions GPS records of 17,241 private cars tracked in Milan during one week from the GeoPKDD project; (3) A GPS trace of 2-hour drive of a private car in Seattle, provided by Krumm [8]; (4) Two public Athens datasets from R-tree portal [9].

— **Geo-Data Sources**: We use: (1) The landuse data of Lausanne on the taxi data to validate the *Semantic Region Annotation*; (2) A large POI dataset of Milan on the Milan private cars data for testing the *Semantic Point Annotation*; (3) The benchmark dataset containing the road network of Seattle and the ground truth paths to evaluate the *Semantic Line Annotation*.

[7]http://www.swisscom.ch/

[8]http://research.microsoft.com/en-us/um/people/jckrumm/MapMatchingData/data.htm

[9]http://www.rtreeportal.org

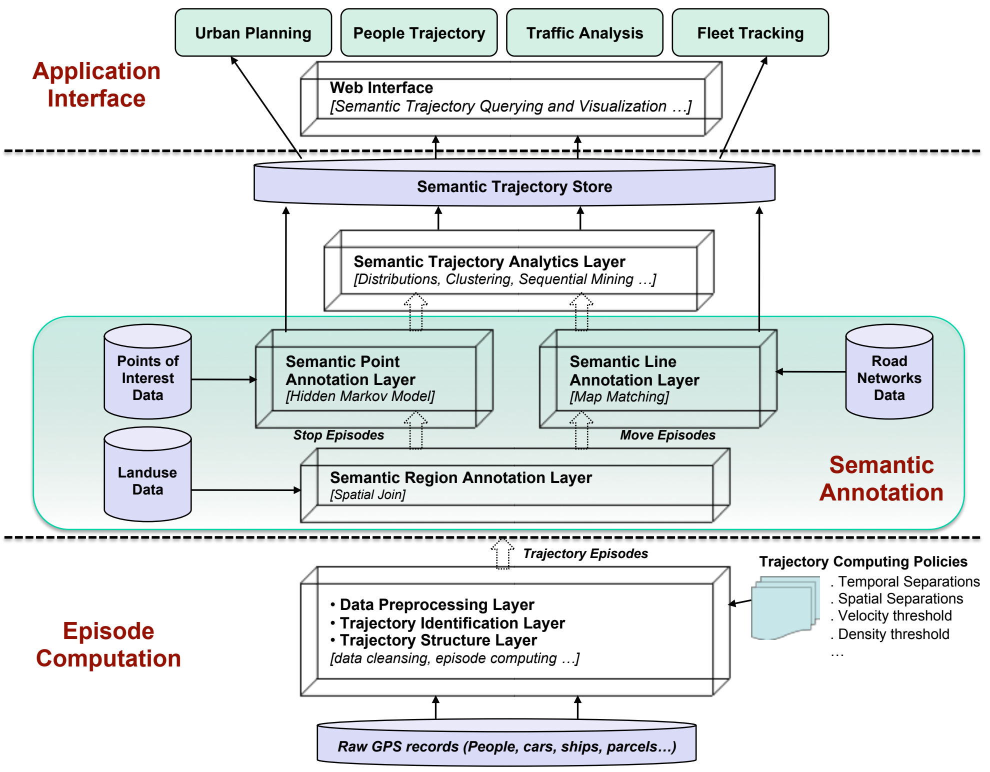

Urban Planning
People Trajectory
Traffic Analysis
Fleet Tracking
Application Interface
Web Interface
[Semantic Trajectory Querying and Visualization …]
Semantic Trajectory Store
Semantic Trajectory Analytics Layer
[Distributions, Clustering, Sequential Mining …]
Points of Interest Data
Semantic Point Annotation Layer
[Hidden Markov Model]
Semantic Line Annotation Layer
[Map Matching]
Road Networks Data
Stop Episodes
Move Episodes
Landuse Data
Semantic Region Annotation Layer
[Spatial Join]
Semantic Annotation
Trajectory Episodes
Trajectory Computing Policies
. Temporal Separations
. Spatial Separations
. Velocity threshold
. Density threshold
…
Episode Computation
• Data Preprocessing Layer
• Trajectory Identification Layer
• Trajectory Structure Layer
[data cleansing, episode computing …]
Raw GPS records (People, cars, ships, parcels…)

Fig. 18: System architecture

Table I: Datasets of Vehicle Trajectories

	Dataset	*# objects*	*# GPS records*	*Tracking time*	*Sampling frequency*
(1)	Lausanne taxis	2	3,064,248	5 months	1 second
(2)	Milan private cars	17,241	2,075,213	1 week	avg. 40 seconds
(3)	Seattle drive	1	7,531	2 hours	1 second
(4)	Athens bus	2	66,095	108 days	30 seconds
(5)	Athens truck	50	112,203	33 days	30 seconds

Geo-Data Sources (3rd party info. sources)	(1) Lausanne (Switzerland): landuse - 1,936,439 cells (2) Milan: points of interest - 39,772 POIs (3) Seattle network (Krumm's benchmark): 158,167 road lines

People trajectories are far less homogeneous than vehicle trajectories: (1) Many reasons can cause GPS data loss, such as the limited power of smartphones, battery outage, and indoor signal loss. (2) Non-stationary sampling rates due to on-chip power saving software modules that monitor the sensor; (3) Compared to vehicles, humans can take complicated on-road/off-road routes, and choose diverse transportation modes (e.g., *walk*, *bicycle*, *bus*, *metro*) during their daily movements. Therefore, the capabilities of our platform are carefully tested through systematic semantic enrichment of such trajectories. Table II shows the detailed dataset related to people trajectories:

— **Trajectories**: This dataset [Kiukkoneny et al. 2010] is provided by Nokia Research Center, Lausanne. They distributed nearly 200 smartphones (Nokia N95) to people in Lausanne, and collected multiple phone sensors readings including GPS feeds.

We analyzed 185 users who traveled 23,188 daily trajectories, generating 7.3M GPS records. Additionally, we studied a subset of users — 1,077 daily trajectories from six users, for whom we have information about their movement behavior as the ground truth data. This is subsequently useful for validation.

— **Geo-Data Sources**: We used the cells of the Swiss landuse map; we also extracted additional geographic data from Openstreetmap[10] - including regions, POIs, road networks of several types (through OSM files) and loaded them into our PostGIS data store (using Osm2pgsql [11]).

Table II: People trajectory data from mobile phones

Complete Phone Dataset	***Details of 6 users with ground truth(Ground Truth Users))***				
summary	*user-id*	*from-date*	*to-date*	*#days-with-GPS*	*#GPS points*
185 smartphone users	1	2009-02-17	2010-04-27	191	50,274
23,188 daily trajectories	2	2009-02-25	2010-05-16	330	200,418
7,306,044 GPS records	3	2009-09-14	2010-05-16	166	62,272
from date: 2009-02-01	4	2009-11-19	2010-05-16	161	66,304
to date: 2010-08-16	5	2009-12-18	2010-05-16	140	69,467
	6	2010-01-25	2010-05-16	89	45,137
Geo-Data Sources (3rd party info. sources)	(1) Lausanne (Switzerland): landuse - 1,936,439 cells (2) Swiss-map: 109,954 geo-objects of kind point, 344,975 of kind line, and 233,896 of kind region				

6.3. Trajectory Computation Results

To easily present the trajectory computation results, we implemented a hybrid trajectory visualization tool using Java 2D API. Fig. 19 provides a snapshot of the tool presenting three sub-figures corresponding to original *GPS feeds*, *spatio-temporal trajectories* and the *structured trajectories*, computed for the Athens truck dataset. The order of the sub-figures (from left to right) follows the progressive computation of higher-level mobility semantic abstraction from the raw-level GPS data feed.

— Sub-figure (a) visualizes the spatial locations of 112,203 raw GPS records, in terms of their 2D geometric coordinates (x, y), without any further meaning (output of *Data Preprocessing* Layer).

— Sub-figure (b) shows 310 spatio-temporal trajectories obtained from the (x, y, t) cleaned sequences (output of *Trajectory Identification* Layer). In order to improve the readability, neighboring trajectories are shown in different colors.

— Sub-figure (c) displays the trajectory episodes (i.e., stops and moves) and visualizes structured trajectories (output of *Trajectory Structure* Layer). There are 1826 stops (visualized as *points*) and 1849 moves (as *lines between points*).

One inherent advantage of our abstraction process is the decrease in the data size as trajectories are abstracted to higher level models. To quantify this, we compute the *semantic abstraction rate* as $log_2(\frac{\#GPS}{\#dataComputed})$, where $\#GPS$ is the number of the initial GPS records and $\#dataComputed$ is the number of computed instances, i.e., the number of trajectories and episodes (stops and moves). For example we observe that for the taxi dataset, 3,347,036 GPS records are abstracted to 1,145 structured trajectories

[10]`http://www.openstreetmap.org/`
[11]`http://wiki.openstreetmap.org/wiki/Osm2pgsql`

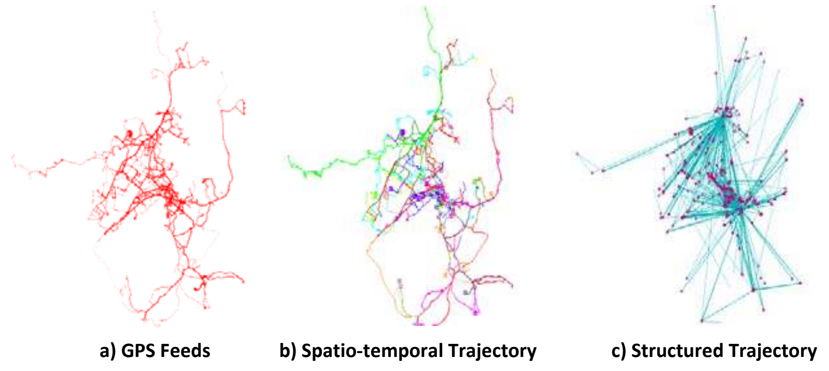


Fig. 19: Trajectory Computation from GPS feeds

with 1,874 stops and 2,925 moves. Fig. 20 shows the abstraction results for the four datasets.

We also observe that, as expected, the abstraction rate is proportional to the GPS sampling frequency. From left to right in Fig. 20, the GPS recording frequency is respectively one record per 40 seconds (on average), 30 seconds, 30 seconds, and one second. We also see that the higher the recording frequency is (like taxi data), the higher is the compression (i.e., the higher abstraction rate).

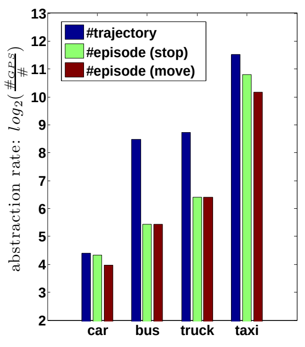


Fig. 20: Vehicle data

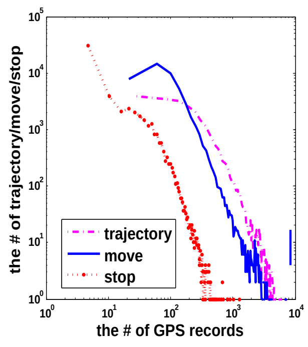


Fig. 21: Phone Dataset

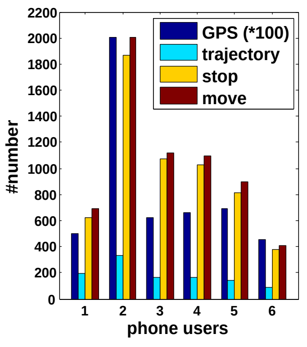


Fig. 22: Ground Truth Users

Similarly, through trajectory episode (stop/move) computation over smartphone data, the 7.3M GPS records are abstracted as 46,958 moves and 52,497 stops of 23,188 daily trajectories. Fig. 21 shows the loglog plot of the length (i.e., the number of GPS records) of extracted trajectories, stops and moves. It shows that most of *moves*/*trajectories* have similar patterns, consisting of a large number of GPS records (say more than 1000), whilst the number of GPS records in a *stop* is usually between 100 and 500, with some between 10 and 100, and a few unusual cases between 500 and 1000. In addition, Fig. 22 shows the details of *stops* and *moves* for the selected 1,077 daily trajectories of 6 users (called “Ground Truth Users ”). Note that the number of GPS records for each user in Fig. 22 is divided by 100 for better representation purposes. The figure brings out the storage compression achievement.

6.4. Sensitivity Analysis

As mentioned earlier, the coefficient for the speed threshold plays a role in determining the number of stop and move episodes and is dependent on several factors (vehicle type, road type etc). Results presented in Fig. 20 have used the same coefficient of speed threshold ($\delta_1 = \delta_2 = \delta = 0.3$) and the same minimal stop duration ($\tau = 15\ mins$) to provide a comparative picture of the abstraction. However, these parameters affect the number of trajectory episodes and needs to be calibrated accordingly.

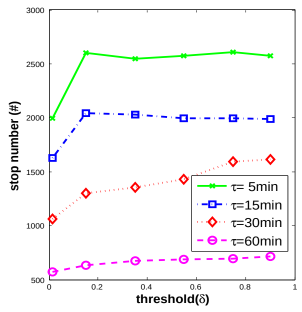


Fig. 23: Δ_{speed} w.r.t. total stop number

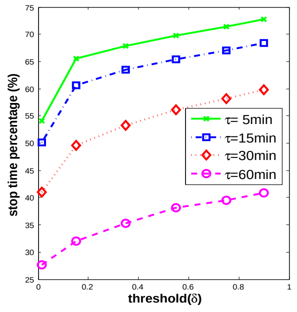


Fig. 24: Δ_{speed} w.r.t. total stop time

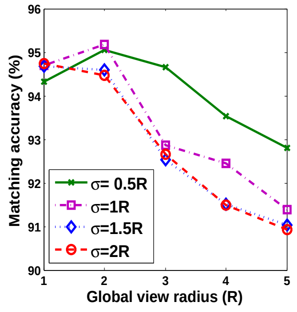


Fig. 25: Sensitivity of map matching accuracy w.r.t. R/σ

We analyzed the sensitivity of δ and τ in identifying stop episodes. Fig. 23 shows the number of stops we get with different δ and τ for the Athens truck data. With higher τ (from five minutes to one hour), the number of stops decreases from 2601 to about 633 when given $\delta = 0.15$; with higher δ (from 0.15 to 0.9), the number of stops goes up then saturates, because stops computed with higher coefficient δ (i.e., higher Δ_{speed}) usually have longer duration. Therefore the number of stops decrease as some stops join together. Nevertheless, we observe that the total percentage of time duration for stops always increases when the minimal stop time τ becomes smaller or the speed threshold δ increases (see Fig. 24). We are investigating means to dynamically calibrate these parameters in trajectory computation.

In the Semantic Line Annotation layer, a global map-matching is applied on the *move* episodes of trajectories in our experiments wherever road network data is available (for vehicle and people trajectories). To measure the efficiency of our approach, we perform a sensitivity analysis of the algorithm using Krumm's benchmark dataset. We first tune the global view radius (R) and the kernel width (σ) for the input data source. Fig. 25 shows the effect of various σ and R on matching accuracy. We observe that small values of R (=2) and σ(=0.5R) produce very high matching accuracy, similar to the recent results on this dataset [Newson and Krumm 2009], confirming the efficiency of the algorithm. Nevertheless, the focus of our *Semantic Line Annotation* is not only on the map matching accuracy, but also on the determination of transportation modes in heterogeneous trajectories.

6.5. Semantic Annotation Results with Statistical Validation

In order to validate the annotated vehicle trajectories without real ground-truth tags, we compute additional statistics. We check if these statistical estimates are relevant w.r.t our knowledge of the areas where the trajectories are collected.

We first check the algorithm that annotates trajectories with regions on the dataset of taxis of Lausanne. More precisely, we check the land use category that our algorithm (the *Semantic Region Annotation Layer*) found for each stop and move, and for each trajectory. Lausanne Landuse map has 4 generic categories and 17 sub-categories (ref. to Fig. 9). Fig. 26 shows the distribution we get for the trajectories, the stops and the moves. We observe that most of the taxi GPS records are in *building areas (1.2)* and *transportation areas (1.3)*, nearly 80% GPS points belong to these categories. In terms of statistical validation, this is generally consistent with typical land use categories covered by taxi trajectories, given the available categories in Lausanne (see the first column in Fig. 26).

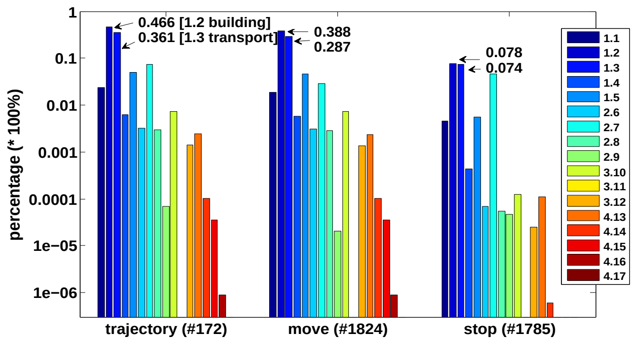


Fig. 26: Landuse category distribution for Taxi data (trajectory, moves, stops)

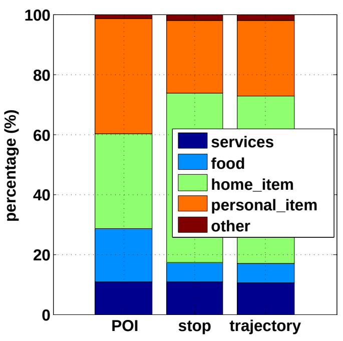


Fig. 27: Semantic stops/ trajectories by HMM-based POI annotation

Secondly, we analyze the results of the HMM-based *Semantic Point Annotation* algorithm for enriching the set of trajectories of private cars in Milan with information about the POIs. Milan POIs are classified in 5 generic categories: *services*, *food*, *personal-item*, *home-item* and *other*. The algorithm infers the most probable POI category for each stop. In Fig. 27 (second column), we observe that most of the stops (about 56.3%) belong to *home-item* (e.g. furniture) with the next one being *personal-item* (e.g., clothing) (about 24.2%). This corresponds to the facts that Milan is a well-known center of design and shopping in Europe, and people tend to go shopping for home-items with a car for two reasons: POIs of kind home-item are more likely to be further from somebody's home than POIs of kind personal-item that are everywhere in Milan; A car is useful for bringing back home-items that are usually heavier and bulkier than personal-items. Conversely, cars stop less frequently in POIs of kind services and food, usually attended on foot due to their proximity to home. Fig. 27 (last column) also shows the *trajectory category* defined as: *the category of $\mathcal{T}$ is the category which has the maximum stop time duration* (see Formula 9), which can be further applied as semantic trajectory classification. For example, if most of the stops of a trajectory are in *home-item*, then this trajectory belongs to the home-item category.

$$trajectory_{cat} = \underset{C_i}{\operatorname{argmax}} \sum_{stop.cat=C_i} (stop.time_{out} - stop.time_{in}) \tag{9}$$

Note that the distribution of $trajectory$ categories is statistically similar to the distribution of $stop$ categories (see Fig. 27). This is because the dataset has only 1.7 stops [12] per trajectory on an average, thereby resulting in a similar distribution. This is co-incidental and depends largely on the trajectory dataset.

[12]2M GPS records, 77,694 trajectories have 133,556 stops

6.6. Semantic Annotation Results with Ground Truth Validation

For a small subset of people trajectories from smartphones in Lausanne (6 persons) we collected some ground truth for validating semantic people trajectories. The ground truth we got consists of: 1) the semantic places where the persons live and where they work, 2) knowledge about their hobbies, and 3) the transportation networks (e.g., bus lines and metro lines) to validate our inference of transportation modes.

In order to validate the algorithm that annotates the *stop* episodes with regions, we computed with our *Semantic Region Annotation Layer* the land use category for each stop. Then we computed for each trajectory its land use coverage as the category where the person stops the most (with respect to the total duration of the stops). Fig 28 shows for each person the distribution of the land use category of his trajectories (with the identifiers of the top-5 categories). From the results of the *Semantic Region Annotation Layer* on these trajectories, we observe that most of the persons are staying in *building areas (1.2)* which corroborates with the ground truth. Moreover, we find that $user3$ has a relatively higher percentage of location records in *wooded area (3.12)*. This is because his accommodation is in the forested place close to the Geneva lake. $user4$'s home is close to a *commercial center area (1.1)* where he does a lot of shopping. $User2$ does hiking and skiing a lot in *forest (3.10)* in contrast with the other persons. This corroborates with our ground truth knowledge of their hobbies.

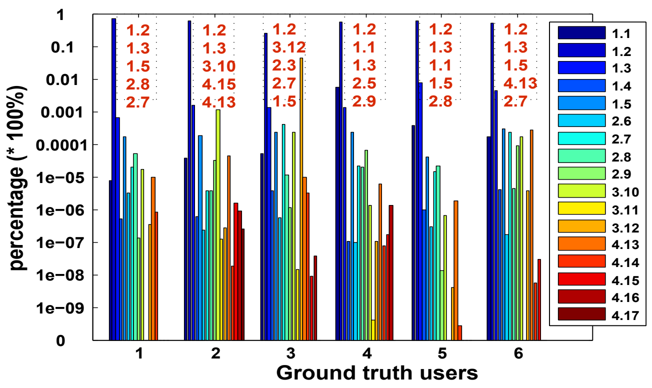
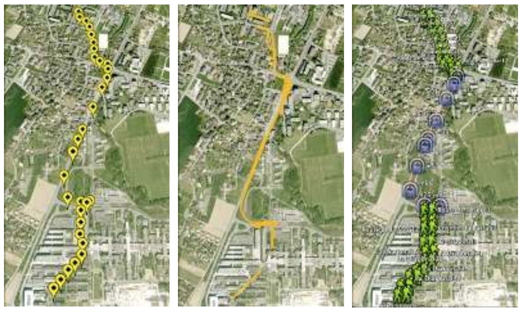


Fig. 28: Landuse category distribution and top-5 categories of people trajectories

In order to validate the algorithms that annotate the *move* episodes, we computed with our *Semantic Line Annotation Layer* the route and the transportation means for each move. This layer uses the underlying network information obtained through our map-matching algorithm on the moves, along with the velocity/acceleration distribution for each road segment in order to determine the transportation mode. As an example, Fig. 29 shows a typical *home-office* trip of $user4$, who walked a few blocks from home, then took the Metro line, and finally walked from the Metro stop to his office: sub-figure (a) shows the original GPS trace; (b) displays the initial map-matched road segments for these GPS points; (c) further infers the corresponding different transportation modes such as *metro* or *walk*; finally (d) summarizes the mobility trace in terms of sequences of roads that are stored in the semantic trajectory store.

$User4$ does not take always the same transportation mode (metro) for going to his office, some days he takes the bus, on sunny days he bikes or even walks. For instance in Fig. 31, the left subfigure (a) shows an example of using *bike* for moving between home and office; whilst in subfigure (b) the user took the *bus*, with *walk* during the



	Road name	Start time
Walk	Ch. veilloud	08:50:26
	Rt. du Boi	08:54:46
	Rt. de Villar	08:57:24
	Tir Fédéra	08:58:41
Metro	M1	08:59:24
Walk	Rt. de la Sorg	09:03:57
	Ch. du Barrag	09:04:42
	La Diagonal	09:05:24

(d) Move annotation

Fig. 29: Move annotation: a home-office move by Metro

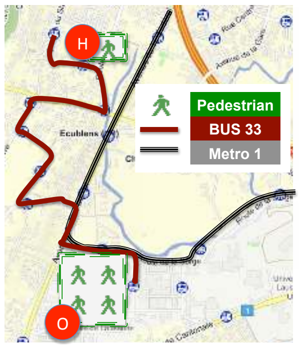


Fig. 30: Ground truth

beginning and ending parts of the trip. Note that the routes taken for bus and bike are different. Based on this inference, from his 161 daily trajectories (from home to office and back home), we computed 186 *home-office* (or *office-home*) moves, and inferred 66 *bike*, 39 *metro*, 49 *bus*, and 32 *walk* annotations respectively.

We acknowledge that a user feedback-driven validation of all such *home-office* moves would have been ideal. However, given the lack of such data, we resort to indirect means of validation. To validate such *home-office* annotation like Fig. 29 and Fig. 31, we extract the real ground truth of bus lines and metro lines, as well as the pedestrian areas around home and office (see Fig. 30). We observe the correctness of such annotation, i.e., the consistency of “Bus33”/“Metro1” w.r.t the transportation modes of taking bus/metro, as well as the pedestrian walk for reaching and leaving bus/metro stops.

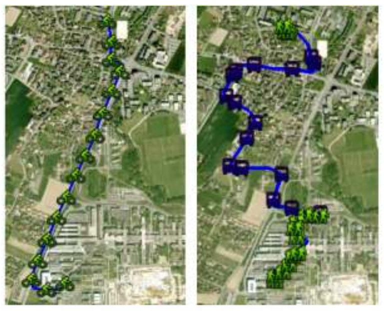


Fig. 31: Home-office (via Bike/Bus)

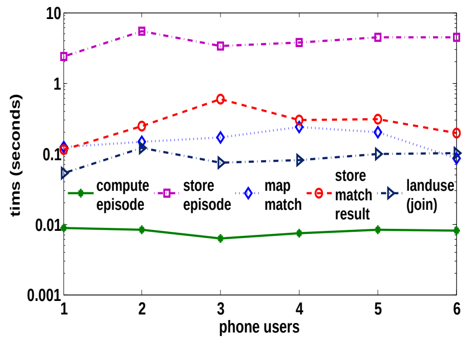


Fig. 32: Latency measures

6.7. Run-Time Performance

We achieve very high data abstraction by computing semantic trajectories from the raw mobility data. Taking the region-based annotation for example, the resultant trajectory representation achieves almost 99.7% storage compression (e.g., 3M GPS records require only 8,385 cells to trajectory data abstraction).

Additionally, we analyze the run-time performance of our platform. Fig. 32 summarizes the latency distribution of our platform for processing phone trajectories. We observe that computation and annotation latencies are much lower (both *map-matching* and *landuse*) than the storing time (writing the results into our semantic trajectory store). For all of the six users, the average time for *computing episodes*, *storing episodes*, *map matching annotation*, *storing matched results*, *landuse annotation* for a daily trajectory are respectively 0.008, 3.959, 0.162, 0.292, and 0.088 seconds. Latency distributions for vehicle trajectories are also similar. The run-time performance of our computation and annotation algorithms is linear w.r.t. the number of objects. Therefore, it is efficient and robust with high scalability.

6.8. Trajectory Web Interface

The trajectory interface that we developed, provides the query and visualization functionality through a Web browser, and showcases the following capabilities:

— *Spatio-Semantic Trajectories* - Shows the multiple levels of trajectory data abstraction: *raw GPS tracks*, *spatio-temporal trajectories* (exploiting space/time gaps), *structured trajectories* (e.g., stops/moves), and *semantic trajectories* (e.g., home-office-supermarket-home).
— *Semantic Places* - Annotates the trajectories with diverse geographic resources - *Landuse*, *Road network*, and *Point of interest* (POI) data.
— *User Interactions* - Provides a friendly Web interface for querying and visualizing trajectories (e.g., daily tracks) at various abstraction levels.
— *Analytics Results* - Highlights *statistical analytics* results of semantic trajectories, e.g., the average speed when the user is moving, Landuse distribution where the user has stopped, the most frequent transportation modes, etc.

7. CONCLUSIONS

In this paper, we designed and implemented a *semantic model and platform* for analyzing trajectories of various kinds of moving objects. Our hybrid spatio-temporal and semantic trajectory model encapsulates both the geometry and semantics of mobility data, supporting several levels of abstraction. Our platform (with computation and annotation functionalities) supports progressive construction of different levels of trajectories, and the enrichement of trajectory semantics from multiple 3rd party semantic sources. Through experimental analysis of real-life GPS feeds, we evaluated how our model and platform achieve the purpose of structural and semantic enrichment of trajectories. Our experiments with various vehicle and people trajectories confirmed the capability of our system to perform well over trajectories of varying data qualities and movement patterns.

An important contribution of our approach is to offer a consistent framework that aims at covering the requirements of a variety of applications, from those that are only interested in the raw data to those looking for very specific semantic enrichments. Providing a set of well-defined concepts that can handle different types of trajectories (including semantic trajectories), and the transitions in between (thanks to the tools provided by our layered framework), is one of the strongest points of our approach and a real innovation wrt works in the literature.

Our ongoing and future research focus is on two main aspects: (1) To build a real-time platform for constructing semantic trajectories for streaming movement data, where the computation and annotation algorithms should be more efficient and even applicable to a distributed context; (2) To augment GPS data with additional sensors (e.g. accelerometer) data to construct a richer (and more complete) inference of daily movement behaviors of people; e.g., using GPS for analyzing outdoor movement and accelerometer for studying indoor activities.

REFERENCES

ALVARES, L. O., BOGORNY, V., KUIJPERS, B., MACEDO, J., MOELANS, B., AND VAISMAN, A. 2007. A Model for Enriching Trajectories with Semantic Geographical Information. In *GIS*. 22.

ANDRIENKO, G., ANDRIENKO, N., AND HEURICH, M. 2011. An Event-Based Conceptual Model for Context-Aware Movement Analysis. *IJGIS 25,* 9, 1347–1370.

BAMIS, A., FANG, J., AND SAVVIDES, A. 2010. A Method For Discovering Components Of Human Rituals From Streams Of Sensor Data. In *CIKM*. 779–788.

BECKMANN, N., KRIEGEL, H.-P., SCHNEIDER, R., AND SEEGER, B. 1990. The R*-Tree: An Efficient and Robust Access Method for Points and Rectangles. *SIGMOD Record 19,* 2, 322–331.

BERNSTEIN, D. AND KORNHAUSER, A. 1996. *An Introduction To Map Matching For Personal Navigation Assistants*. Number 8. Princeton University.

BRAKATSOULAS, S., PFOSER, D., SALAS, R., AND WENK, C. 2005. On Map-Matching Vehicle Tracking Data. In *VLDB*. 853–864.

BRINKHOFF, T., KRIEGEL, H.-P., AND SEEGER, B. 1993. Efficient Processing of Spatial Joins using R-Trees. In *SIGMOD*. 237–246.

BUCHIN, M., DRIEMEL, A., KREVELD, M. V., AND SACRISTAN, V. 2010. An Algorithmic Framework for Segmenting Trajectories based on Spatio-Temporal Criteria. In *GIS*. 202–211.

CAO, X., CONG, G., AND JENSEN, C. S. 2010. Mining Significant Semantic Locations From GPS Trajectories. In *VLDB*. 1009–1020.

CHEN, S., JENSEN, C. S., AND LIN, D. 2008. A Benchmark for Evaluating Moving Object Indexes. *PVLDB 1,* 2, 1574–1585.

FORNEY, G. D. 1973. The Viterbi Algorithm. *Proceedings of the IEEE 61,* 3, 268–278.

FRENTZOS, E. 2008. Trajectory Data Management in Moving Object Databases. Ph.D. thesis, University of Piraeus.

GIANNOTTI, F. AND PEDRESCHI, D. 2008. *Mobility, Data Mining and Privacy, Geographic Knowledge Discovery*. Springer-Verlag.

GÓMEZ, L. I. AND VAISMAN, A. A. 2009. Efficient Constraint Evaluation in Categorical Sequential Pattern Mining for Trajectory Databases. In *EDBT*. 541–552.

GUO, T., YAN, Z., AND ABERER, K. 2012. An Adaptive Approach for Online Segmentation of Multi-Dimensional Mobile Data. In *MobiDE*.

GÜTING, R. H. 2005. SECONDO: A Database System for Moving Objects. *GeoInformatica 9,* 1, 33–60.

GÜTING, R. H., DE ALMEIDA, V. T., AND DING, Z. 2006. Modeling and querying moving objects in networks. *The VLDB Journal 15,* 2, 165–190.

GÜTING, R. H. AND SCHNEIDER, M. 2005. *Moving Objects Databases*. Morgan Kaufmann.

HAN, J., LEE, J.-G., GONZALEZ, H., AND LI, X. 2008. Mining Massive RFID, Trajectory, and Traffic Data Sets (Tutorial). In *KDD*.

JUN, J., GUENSLER, R., AND OGLE, J. 2006. Smoothing Methods to Minimize Impact of Global Positioning System Random Error on Travel Distance, Speed, and Acceleration Profile Estimates. *Transportation Research Record: Journal of the Transportation Research Board 1972,* 1, 141–150.

KEOGH, E., CHU, S., HART, D., AND PAZZANI, M. 2004. Segmenting Time Series: A Survey and Novel Approach. In *Data mining in Time Series Databases*. 1–22.

KIUKKONENY, N., BLOM, J., DOUSSE, O., GATICA-PEREZ, D., AND LAURILA, J. 2010. Towards Rich Mobile Phone Datasets: Lausanne Data Collection Campaign. In *ICPS*.

KUIJPERS, B. AND OTHMAN, W. 2007. Trajectory Databases: Data Models, Uncertainty and Complete Query Languages. In *ICDT*. 224–238.

LI, Q., ZHENG, Y., XIE, X., CHEN, Y., LIU, W., AND MA, W.-Y. 2008. Mining User Similarity Based on Location History. In *GIS*. 34.

LI, Z., DING, B., HAN, J., KAYS, R., AND NYE, P. 2010. Mining Periodic Behaviors for Moving Objects. In *KDD*. 1099–1108.

LI, Z., HAN, J., JI, M., TANG, L. A., YU, Y., DING, B., LEE, J.-G., AND KAYS, R. 2011. MoveMine: Mining moving object data for discovery of animal movement patterns. *ACM TIST 2,* 4, 37.

LOU, Y., ZHANG, C., ZHENG, Y., XIE, X., WANG, W., AND HUANG, Y. 2009. Map-matching for Low-Sampling-Rate Gps Trajectories. In *GIS*. 352–361.

MARKETOS, G., FRENTZOS, E., NTOUTSI, I., PELEKIS, N., RAFFAETÀ, A., AND THEODORIDIS, Y. 2008. Building real-world trajectory warehouses. In *MobiDE*. 8–15.

MERATNIA, N. AND DE BY, R. A. 2004. Spatiotemporal Compression Techniques for Moving Point Objects. In *EDBT*. 765–782.

MOUZA, C. AND RIGAUX, P. 2005. Mobility Patterns. *GeoInformatica 9,* 4, 297–319.

NANNI, M., TRASARTI, R., RENSO, C., GIANNOTTI, F., AND PEDRESCHI, D. 2010. Advanced Knowledge Discovery On Movement Data With The GeoPKDD System. In *EDBT*. 693–696.

NERGIZ, M., ATZORI, M., SAYGN, Y., AND GÜÇ, B. 2009. Towards Trajectory Anonymization: A Generalization-based Approach. *Transactions on Data Privacy 2,* 1, 47–75.

NEWSON, P. AND KRUMM, J. 2009. Hidden Markov Map Matching Through Noise and Sparseness. In *GIS*. 336–343.

PALMA, A. T., BOGORNY, V., KUIJPERS, B., AND ALVARES, L. O. 2008. A Clustering-based Approach for Discovering Interesting Places in Trajectories. In *ACM-SAC*. 863–868.

PELEKIS, N., THEODORIDIS, Y., VOSINAKIS, S., AND PANAYIOTOPOULOS, T. 2006. HERMES - A Framework for Location-Based Data Management. In *EDBT*. 1130–1134.

QUDDUS, M. A., OCHIENG, W. Y., AND NOLAND, R. B. 2007. Current Map-Matching Algorithms for Transport Applications: State-Of-The Art and Future Research Directions. *Transportation Research Part C: Emerging Technologies 15,* 5, 312–328.

RABINER, L. R. 1990. A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition. *Readings in speech recognition*, 267–296.

SALTENIS, S., JENSEN, C. S., LEUTENEGGER, S. T., AND LOPEZ, M. A. 2000. Indexing the Positions of Continuously Moving Objects. In *SIGMOD*. 331–342.

SCHÜSSLER, N. AND AXHAUSEN, K. W. 2009. Processing GPS Raw Data Without Additional Information. *Transportation Research Record: Journal of the Transportation Research Board 8*, 28–36.

SPACCAPIETRA, S., PARENT, C., DAMIANI, M. L., DE MACEDO, J. A., PORTO, F., AND VANGENOT, C. 2008. A Conceptual View on Trajectories. *Data and Knowledge Engineering 65*, 126–146.

WESSEL, M., LUTHER, M., AND MÖLLER, R. 2009. What Happened to Bob? Semantic Data Mining of Context Histories. *Description Logics*.

WHITE, C. E., BERNSTEIN, D., AND KORNHAUSER, A. L. 2000. Some map matching algorithms for personal navigation assistants. *Tramsportation Research Part C: Emerging Technologies 8,* 1-6, 91–108.

WOLFSON, O., SISTLA, P., XU, B., ZHOU, J., AND CHAMBERLAIN, S. 1999. DOMINO: databases fOr MovINg Objects tracking. In *SIGMOD*. 547–549.

WOLFSON, O., XU, B., CHAMBERLAIN, S., AND JIANG, L. 1998. Moving Objects Databases: Issues and Solutions. In *SSDBM*. 111–122.

XIE, K., DENG, K., AND ZHOU, X. 2009. From Trajectories to Activities: a Spatio-Temporal Join Approach. In *LBSN*. 25–32.

YAN, Z. 2010. Traj-ARIMA: A Spatial-time Series Model for Network-constrained Trajectory. In *IWCTS*. 11–16.

YAN, Z., CHAKRABORTY, D., PARENT, C., SPACCAPIETRA, S., AND KARL, A. 2011. SeMiTri: A Framework for Semantic Annotation of Heterogeneous Trajectories. In *EDBT*. 259–270.

YAN, Z., GIATRAKOS, N., KATSIKAROS, V., PELEKIS, N., AND THEODORIDIS, Y. 2011. SeTraStream: Semantic-Aware Trajectory Construction over Streaming Movement Data. In *SSTD*. 367–385.

YAN, Z., MACEDO, J., PARENT, C., AND SPACCAPIETRA, S. 2008. Trajectory Ontologies and Queries. *Transactions in GIS 12(s1)*, 75–91.

YAN, Z., PARENT, C., SPACCAPIETRA, S., AND CHAKRABORTY, D. 2010. A Hybrid Model and Computing Platform for Spatio-Semantic Trajectories. In *ESWC*. 60–75.

YIN, J., CHAI, X., AND YANG, Q. 2004. High-Level Goal Recognition in a Wireless LAN. In *AAAI*. 578–584.

ZHANG, J. AND GOODCHILD, M. F. 2002. *Uncertainty in Geographical Information* 1 Ed. CRC.

ZHENG, Y., CHEN, Y., LI, Q., XIE, X., AND MA, W.-Y. 2010. Understanding transportation modes based on GPS data for web applications. *TWEB 4,* 1.

ZHENG, Y., ZHANG, L., MA, Z., XIE, X., AND MA, W.-Y. 2011. Recommending friends and locations based on individual location history. *ACM Transactions on the Web 5,* 1, 1–44.

ZHENG, Y., ZHANG, L., XIE, X., AND MA, W.-Y. 2009. Mining correlation between locations using human location history. In *GIS*. 472–475.

ZHOU, C., FRANKOWSKI, D., LUDFORD, P. J., SHEKHAR, S., AND TERVEEN, L. G. 2007. Discovering Personally Meaningful Places: an Interactive Clustering Approach. *ACM Transactions on Information Systems 25,* 3, 12.