iPark: Identifying Parking Spaces from Trajectories

Bin Yang Nicolas Fantini Christian S. Jensen
Department of Computer Science, Aarhus University, Denmark
{byang, nicolasf, csj}@cs.au.dk

ABSTRACT

A wide variety of desktop and mobile Web applications involve geo-tagged content, e.g., photos and (micro-) blog postings. Such content, often called User Generated Geo-Content (UGGC), plays an increasingly important role in many applications. However, a great demand also exists for "core" UGGC where the geo-spatial aspect is not just a tag on other content, but *is* the primary content, e.g., a city street map with up-to-date road construction data. Along these lines, the iPark system aims to turn volumes of GPS data obtained from vehicles into information about the locations of parking spaces, thus enabling effective parking search applications. In particular, we demonstrate how iPark helps ordinary users annotate an existing digital map with two types of parking, on-street parking and parking zones, based on vehicular tracking data.

Categories and Subject Descriptors

H.2.8 [**Database Applications**]: Data mining, Spatial databases and GIS

General Terms

Design, Experimentation

Keywords

Parking, Trajectories, User Generated Geo-Content

1. INTRODUCTION

Web applications increasingly involve content with a geo-spatial aspect, such as geo-tagged micro-blog postings[1], photos[2], and social network users and locations[3]. The geo-spatial aspect of this kind of User-Generated Geo Content (UGGC) [1] occurs mainly as meta data, namely as spatial attributes (e.g., locations) that describe other content (e.g., photos).

In addition to such UGGC, we are also faced with a great demand for core UGGC that is inherently geo-spatial, such as (poly-)lines representing new streets, turn restrictions, speed limits, and addresses of point-of-interests. For example, OpenStreetMap (OSM)[4] aims to create a free, editable map of the world, where users can create and update streets by modifying OSM map files. TomTom, a leading manufacturer of navigation systems, encourages users to make changes (e.g., altered turn restrictions) to its existing maps using TomTom Map Share[5] in order to achieve up-to-date navigation plans. Navteq, a major provider of electronic navigable maps, also allows users to update its maps using Navteq Map Reporter[6].

Parking is an important aspect of vehicular transportation, as search for parking by drivers is a significant contributor to congestion in cities and thus also generates considerable amounts of greenhouse gas emissions [2]. Further, drivers waste considerable time on searching for parking and on leaving early due to anticipated parking problems.

Several contributions towards enabling effective parking search services have been made recently [3, 4]. An important prerequisite for such parking search services is that the locations of parking spaces are recorded in digital maps. However, parking spaces (especially on-street parking) are often missing in full or in part for cities in existing maps (e.g., Google Maps, Bing Maps, and OSM). Thus, it is relevant to provide means of obtaining more complete parking information, e.g., using GPS records from vehicles that are available in increasingly large volumes.

We present iPark, a system that enables users to enhance the coverage of parking spaces in OSM. After uploading a collection of vehicular tracking data (e.g., GPS records), iPark identifies two types of parking, on-street parking and parking zone parking. Users of the system have the opportunity to make the final decision as to whether or not to upload the identified parking into OSM. We are not aware of other systems that facilitate users in creating parking space UGGC in an efficient and effective manner from GPS records.

2. SYSTEM DESIGN

Figure 1 gives an overview of the iPark system, which consists of four major modules: pre-processing, graph construction, label propagation, and post-processing.

The pre-processing module takes as input a collection of GPS records obtained from vehicles, and it outputs a collection of *parking trajectories* that exhibit parking search behavior and stop at possible parking spaces. By considering the similarity of the parking search behaviors exhibited by different parking trajectories, the graph construction module builds a *trajectory similarity graph* based

[1] `http://tinyurl.com/ca9n4ha`
[2] `http://www.flickr.com/groups/geotagging/`
[3] `https://foursquare.com/about/`
[4] `http://www.openstreetmap.org/`
[5] `http://www.tomtom.com/en_gb/maps/map-share/`
[6] `http://mapreporter.navteq.com/`

EDBT/ICDT '13, March 18 - 22 2013, Genoa, Italy.

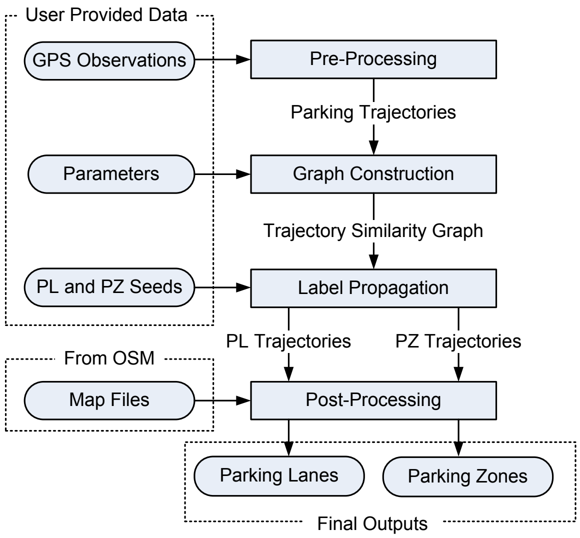

Figure 1: iPark Overview

on user-specified parameters. A small number of *seed trajectories* that exhibit parking lane (PL) and parking zone (PZ) parking behavior are identified either in an automatic manner with the help of existing parking locations recorded in OSM or manually by the users. The label propagation module then applies semi-supervised learning [5] to the trajectory similarity graph to classify parking trajectories into two categories, PL trajectories and PZ trajectories. The post-processing module takes as input existing OSM map files and the identified PL and PZ trajectories, and it identifies possible on-street and parking zone parking. In the following, we describe each module in detail.

2.1 Pre-Processing

The input to the pre-processing module is a collection of GPS vehicle tracking records. The pre-processing conducts three tasks: reorganizing the GPS records, map matching, and parking trajectory filtering.

Data Reorganization: A GPS record is of the form

$$(VID, t, l, h, s, mm),$$

where VID is a vehicle identifier; t indicates the time of the observation; l, h, s indicate the location (a latitude-longitude pair) of the vehicle at t, the heading (degrees w.r.t. North) of the vehicle's movement at t, and the instantaneous speed of the vehicle at t, respectively; and mm is an attribute reserved for map matching.

After grouping the GPS records by vehicle identifier, and ordering them based on time, GPS tracking observations are reorganized into collections of trajectories. A trajectory is a sequence of GPS observations that typically indicates a trip.

Map Matching: Trajectories are map matched onto an OSM map using a map matching tool [6]. In particular, a GPS observation in a trajectory is mapped to a specific point on a road segment in an OSM map, and the mm field of the observation is updated with the specific point and the road segment identifier. However, a small portion of GPS observations cannot be mapped to any road segments. When this occurs, the corresponding mm fields are associated with empty values.

Parking Trajectory Filtering: Not every trajectory ends at a parking space, so we eliminate the trajectories that do not finish at parking spaces, yielding a collection of *parking trajectories*.

Intuitively, all such trajectories finish at a parking space, but we need to filter some special cases. For example, vehicles that travel through the Limfjord tunnel, Aalborg, Denmark, generate non-parking trajectories due to missing GPS signal reception in the tunnel. We eliminate non-parking trajectories by considering the speeds of the last few observations of the trajectories. Specifically, we treat a trajectory as a parking trajectory if the speed of its last GPS observation is zero; if its last few GPS observations have very low instantaneous speeds; and if it exhibits a clear slowing down process during the ending part of the trajectory. In addition, a parking duration threshold enables the separation of vehicles that are parked from vehicles that merely are stopped temporarily, e.g., due to red traffic lights.

2.2 Graph Construction

The graph construction module generates a *trajectory similarity graph* that is employed as the data foundation in the label propagation module. A trajectory similarity graph is a weighted, undirected graph $G = (\mathbb{V}, \mathbb{E}, F)$, where $\mathbb{V}$ and $\mathbb{E}$ are vertex and edge sets and function $F : \mathbb{E} \rightarrow \mathbb{R}$ records the weights of edges in $\mathbb{E}$.

A vertex $v_i \in \mathbb{V}$ represents a parking trajectory, and an edge $e_k \in \mathbb{E}$ is defined by a set of two distinct vertices. For example, $e_k = \{v_i, v_j\}$ is an edge connecting vertices v_i and v_j, $(v_i \neq v_j)$. The parking search similarity between two trajectories represented by vertices v_i and v_j, denoted as $sim(v_i, v_j)$, is determined by Equation 1. If $sim(v_i, v_j)$ exceeds a threshold α, an edge $e_k = \{v_i, v_j\}$ is created with weight $sim(v_i, v_j)$, which is recorded in function F by keeping an entry $F(\{v_i, v_j\}) = sim(v_i, v_j)$. Figure 2 illusttrates a trajectory similarity graph where the similarity threshold α is set to 0.5.

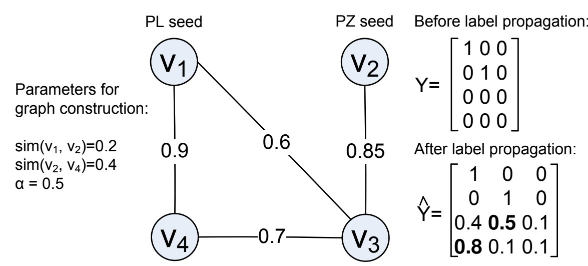

Figure 2: Trajectory Similarity Graph and Label Propagation

After analyzing more than 100 million GPS records collected from Aalborg, Denmark, six distinct features that can be derived from GPS records were identified that indicate parking behavior. The similarity between trajectories v_i and v_j is defined as a weighted average of the similarities on the six individual features, as defined in Equation 1.

$$sim(v_i, v_j) = \frac{\sum_{k=1}^{6} \lambda_k \cdot norm_k(|f_k(v_i) - f_k(v_j)|)}{\sum_{k=1}^{6} \lambda_k}, \qquad (1)$$

where λ_k is a relative importance weight of the k-th feature; $|f_k(v_i) - f_k(v_j)|$ is the difference between trajectories of v_i and v_j on the k-th feature; and $norm_k(\cdot)$ is a normalization function that returns a value in $[0, 1]$. Note that different (e.g., linear and non-linear) normalization functions may be used for evaluating different features. We proceed to cover the six features and how to compute similarity for each.

Heading Difference: The heading differences of the last β_1 GPS observations of a parking trajectory is a good indicator of whether a parking trajectory ended at a PL or a PZ. On-street parking normally involves a relatively complicated movement compared to parking zone parking, and thus PL parking trajectories typically contains more heading changes. Thus, $f_1(v_i)$ returns the absolute value of the average heading changes of the last β_1 observations of parking trajectory v_i.

Opposite Headings: This feature considers how many opposite headings exist in the last β_1 GPS observations of a parking trajectory. If the heading difference between two consecutive observations exceeds β_2 degrees, we treat it as an opposite heading occurrence. Feature $f_2(v_i)$ returns the number of opposite headings in the last β_1 observations of parking trajectory v_i.

Average Speed: We also consider the average speed of the last β_1 GPS observations of a parking trajectory. A PL trajectory normally has a lower average speed at the end since the vehicle has to do back and forth movements with low speed. A PZ trajectory is expected to have a higher average speed since the vehicle can enter a parking space more directly. Feature $f_3(v_i)$ returns the average speed of the last β_1 observations of parking trajectory v_i.

Near Points: The near points in a parking trajectory are the GPS observations that locate within a circle with the location of the last GPS observation as center and β_3 meters as radius. The presence of many such near points suggests that the vehicle did parking maneuvers, which increases the possibility that the vehicle parked at a PL. Thus, $f_4(v_i)$ returns the number of near points of parking trajectory v_i.

Parking Track Points: Given a parking trajectory, its parking search region is a circle with the location of its last GPS observation as center and β_4 meters as radius ($\beta_4 \gg \beta_3$). The GPS records whose locations are within the parking search region and have speeds that are lower than β_5 km/h are the parking track points. The intuition is that a substantial number of PL trajectories involve search for parking, meaning that PL trajectories typically have more parking track points than do PZ trajectories. Feature $f_5(v_i)$ returns the number of parking track points of parking trajectory v_i.

Parking Place Matched: Whether the last β_6 observations of a parking trajectory are matched to a road segment is also a very important feature that distinguishes PL trajectories from PZ trajectories. Thus, $f_6(v_i)$ returns the number of map-matched observations in the last β_6 observations of trajectory v_i.

2.3 Label Propagation

Label propagation [5], a semi-supervised learning algorithm, is conducted on top of the trajectory similarity graph in order to assign each trajectory a label indicating that the trajectory ended at a PL, a PZ, or an unknown type of parking.

Before running the label propagation algorithm, a small number of trajectories must be labeled as PL and PZ trajectories—we call them *seed trajectories*. If a trajectory stopped at a parking space that is already recorded in OSM, the trajectory can be labeled with the corresponding seed. For example, if the last GPS record of a trajectory is in a parking zone that is already recorded in OSM, the trajectory becomes a PZ seed. Further, users can also manually label trajectories, especially with the PL label. Only a small number of seed trajectories (for our data set, 8 PL and 8 PZ seeds) need to be created before the label propagation algorithm can automatically propagate labels to unlabeled trajectories and assign each trajectory a label.

We use a matrix $\mathbf{Y} \in \mathbb{R}^{N\times M}$ to denote the initial label assignment, where $N = |\mathbb{V}|$ is the total number of parking trajectories, and $M = 3$ indicates the three possible labels, i.e., PL, PZ, and unknown. If the i-th trajectory is a PL (PZ) seed trajectory, its corresponding entry in matrix $\mathbf{Y}$ is set to 1. Figure 2 shows an example where v_1 and v_2 is a PL and PZ seed trajectory, respectively. Thus, $\mathbf{Y}_{1,1} = 1$ and $\mathbf{Y}_{2,2} = 1$, and all the remaining entries in $\mathbf{Y}$ are 0.

After label propagation, we get a new matrix $\hat{\mathbf{Y}} \in \mathbb{R}^{N\times M}$ that records the estimated labels for all trajectories. Specifically, the value of entry $\hat{\mathbf{Y}}_{i,j}$ indicates the probability of the i-th trajectory being labeled with the j-th label. For the i-th trajectory, the j-th label with the biggest probability ($j = \arg\max_{x\in\{1,2,3\}} \hat{\mathbf{Y}}_{i,x}$) is used as the final label of the trajectory. For example, trajectory v_3 and v_4 is finally labeled as a PZ and a PL trajectory, respectively, because $\hat{\mathbf{Y}}_{3,2}$ and $\hat{\mathbf{Y}}_{4,1}$ has the biggest value on the third and fourth rows of $\hat{\mathbf{Y}}$, respectively, as shown in Figure 2.

The process of label propagation amount to minimizing the objective function

$$O(\hat{\mathbf{Y}}) = \sum_{x=1}^{M} \Big[\underbrace{(\mathbf{Y}_{\cdot x} - \hat{\mathbf{Y}}_{\cdot x})^{\mathrm{T}}\mathbf{S}(\mathbf{Y}_{\cdot x} - \hat{\mathbf{Y}}_{\cdot x})}_{\text{Keeping seed labels}}$$
$$+ \underbrace{\mu_1 \hat{\mathbf{Y}}_{\cdot x}^{\mathrm{T}}\mathbf{L}\hat{\mathbf{Y}}_{\cdot x}}_{\text{Spreading labels}} + \underbrace{\mu_2 ||\hat{\mathbf{Y}}_{\cdot x} - \mathbf{R}_{\cdot x}||_2^2}_{\text{Regularization}} \Big],$$

where the intuition is to keep the labels for seed trajectories; to spread labels over the graph while ensuring that similar trajectories (evaluated by Equation 1) obtain similar labels; and to do regularization to avoid over-fitting. Specifically, $\mathbf{Y}_{\cdot x}$, $\hat{\mathbf{Y}}_{\cdot x}$, and $\mathbf{R}_{\cdot x}$ indicate the x-th column of the corresponding matrices. Next, $\mathbf{L}$ is a graph Laplacian matrix derived from the trajectory similarity graph G. How to construct the auxiliary matrices $\mathbf{S}$ and $\mathbf{R}$, and how to choose appropriate values for hyper-parameters μ_1 and μ_2 are beyond the scope of the paper and is covered elsewhere [5].

2.4 Post-Processing

All the locations of the last GPS records of the identified PL trajectories indicate locations on road segments with possible on-street parking spaces. By considering the directions of the PL trajectories, the street sides that allow parking can be decided as well. The identified PLs can be recorded in existing OSM road segments by modifying the "parking:lane:{both|left|right}" attribute of such segments.

By clustering the locations of the last GPS observations of PZ trajectories, parking zones are identified, where each cluster corresponds to a parking zone. If a cluster only contains one GPS observation, a node with tag "amenity=parking" is created. If a cluster contains more than one GPS observations, a polygon that covers all the GPS observations is created to indicate the extent of the identified parking zone.

We provide two approaches to help ensure that the identified parking spaces are correct. First, we only consider the PL and PZ trajectories with high confidences. For example, we only consider v_4 for PL, but do not consider v_3 for PZ because v_3 has similar confidences for both PL and PZ labels as shown in Figure 2. Second, the user is asked to check the identified parking spaces and make the final decision.

3. DEMONSTRATION OUTLINE

The user interface of iPark is developed on top of JOSM[7] and OpenLayers[8]. Figure 3, to be explained shortly, shows a screenshot of iPark's user interface. The four modules of iPark are implemented in Java, where the label propagation module is built based on Junto[9]. Next, we describe how demonstration participants can interact with iPark to experience the working of the four modules.

[7]http://josm.openstreetmap.de/

[8]http://www.openlayers.org/

[9]http://code.google.com/p/junto/

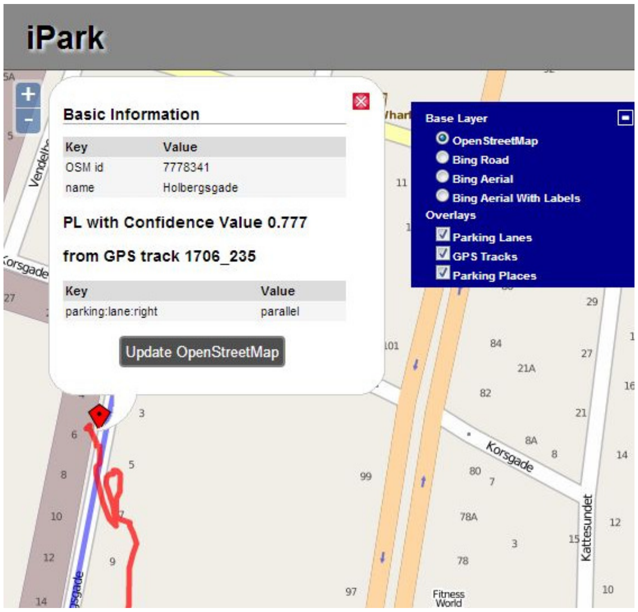

Figure 3: User Interface of iPark

Pre-Processing: We provide several collections of GPS tracking data collected from Aalborg, Denmark. Participants may choose any of these for conducting the pre-processing step. Upon pre-processing, parking trajectories are listed. The participants can then choose to visualize the parking trajectories, along with the corresponding map matched road segments, on OSM or Bing Maps. Figure 3 shows a parking trajectory as a red line along with its corresponding, map-matched road segment, shown as a blue line, where OSM is used.

Graph Construction: A data entry panel is provided that enables the participants to set parameters that control the working of the system. Default parameter settings are also provided. The participants are able to observe differences in the graphs obtained when choosing different parameter settings. This part of the demonstration offers an intuitive understanding of the effects of the different parameters.

Label Propagation: To enable users to conveniently identify PL and PZ seed trajectories, iPark provides an interface that can visualize trajectories on top of an aerial image, e.g., by choosing the "Bing Aerial (with labels)" button in the upper-right box in Figure 3. If the visualization shows that a trajectory stops along a street (or in a parking lot), users can label the trajectory as a PL (or PZ) seed trajectory.

For example, since Figure 4 clearly shows that the trajectory finished in a parking zone, participants can label it as a PZ seed trajectory by clicking "PZ seed" in the popup window. To facilitate the demonstration, we also provide a group of pre-selected PL and PZ seed trajectories.

After running the label propagation process based on the chosen seeds, iPark provides an interface for the users to inspect the results of the label propagation, i.e., the label (along with its confidence value) assigned to each parking trajectory.

Post-Processing: After post-processing, a collection of PLs and a collection of PZs are identified. Next, the participants can set a confidence threshold, so that identified parking spaces with confidence exceeding the threshold can be uploaded to an OSM map file directly. The parking spaces identified with confidence below the threshold are listed so that users can determine by inspection whether they are correct.

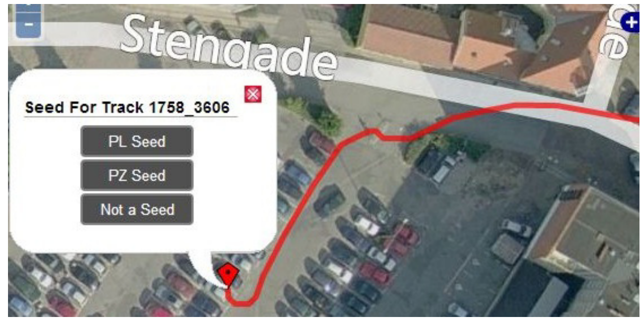

Figure 4: Choosing Seed Trajectories

For example, the red marker shown in Figure 3 indicates an identified parking space with low confidence. By clicking the marker, a pop-up window shows its type, confidence value, corresponding parking trajectory identifier, and other information about the corresponding road segment. By means of a visualization on top of an aerial image, the demonstration participants can check the correctness of the identified parking spaces, and they can decide whether to update the OSM map files with the identified parking spaces.

4. CONCLUSION AND OUTLOOK

We demonstrate how the iPark system allows users to turn volumes of GPS records from vehicles into an OSM parking layer that captures on-street parking and parking in parking zones.

Several improvements to iPark are possible, e.g., better filtering of trajectories that experience long non-parking stops, e.g., due to traffic jams; distinguishing private and public parking spaces; and integrating iPark with parking search services. It is also of interest to explore other opportunities for facilitating the use of GPS data for the creation of core UGGC that may enhance map-based applications. Examples include improved representations of rotaries in maps and the association of road width information (e.g., number of lanes) with polyline representations of roads in maps.

Acknowledgments

This work was partially supported by the REDUCTION project, funded by the European Commission as an FP7-ICT-2011-7 STREP project, contract number 288254.

5. REFERENCES

[1] D. Pfoser. On user-generated geocontent. In *SSTD*, pages 458–461, 2011.

[2] L. Montini, A. Horni, N. Rieser-Schüssler, and K. Axhausen. Searching for parking in gps data. In *IATBR*, pages 1–25, 2012.

[3] D. Ayala, O. Wolfson, B. Xu, B. Dasgupta, and J. Lin. Parking slot assignment games. In *GIS*, pages 299–308, 2011.

[4] V. Verroios, V. Efstathiou, and A. Delis. Reaching available public parking spaces in urban environments using ad hoc networking. In *MDM*, pages 141–151, 2011.

[5] Y. Wang, B. Yang, L. Qu, M. Spaniol, and G. Weikum. Harvesting facts from textual web sources by constrained label propagation. In *CIKM*, pages 837–846, 2011.

[6] F. Pereira, H. Costa, and N. Pereira. An off-line map-matching algorithm for incomplete map databases. *European Transport Research Review*, 1(3):107–124, 2009.