Hup-Me: Inferring and Reconciling a Timeline of User Activity from Rich Smartphone Data

David Montoya
Cofely Ineo, Engie group
& ENS Cachan & INRIA, France
david.montoya@inria.fr

Serge Abiteboul
INRIA & ENS Cachan, France
serge.abiteboul@inria.fr

Pierre Senellart
Institut Mines–Télécom;
Télécom ParisTech; CNRS LTCI
& NUS; CNRS IPAL
pierre.senellart@telecom-paristech.fr

ABSTRACT

We designed a system to infer multimodal itineraries traveled by a user from a combination of smartphone sensor data (e.g., GPS, Wi-Fi, accelerometer) and knowledge of the transport network infrastructure (e.g., road and rail maps, public transportation timetables). The system uses a *Transportation network* that captures the set of possible paths of this network for the modes, e.g., `foot`, `bicycle`, `road_vehicle`, and `rail`. This Transportation network is constructed from OpenStreetMap data and public transportation routes published online by transportation agencies in GTFS format. The system infers itineraries from a sequence of smartphone observations in two phases. The first phase uses a dynamic Bayesian network that models the probabilistic relationship between paths in Transportation network and sensor data. The second phase attempts to match portions recognized as `road_vehicle` or `rail` with possible public transportation routes of type `bus`, `train`, `metro`, or `tram` extracted from the GTFS source. We evaluated the performance of our system with data from users traveling over the Paris area who were asked to record data for different trips via an Android application. Itineraries were annotated with modes and public transportation routes taken and we report on the results of the recognition.

Categories and Subject Descriptors

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

Keywords

Activity Recognition, Dynamic Bayesian Networks, Multimodal Transport Networks, Itinerary Recognition, Smartphone sensors

1. INTRODUCTION

The democratization of connected and sensor-rich personal mobile devices has increased the demand for context-aware applications taking advantage of the information they are able to generate. Technology is still far from taking full advantage of smartphone sensors to understand the users' daily movements in urban environments. In this paper, we present *Hup-me*, a system we designed that uses a wide range of sensor data to infer a timeline of user's multimodal activities. Motivated by the increasing interest in activity tracking devices (e.g. calories spent and quality of sleep monitors), we believe that automatic inference of rich transportation routines may allow the construction of *intelligent travel assistants*. Such assistants may exploit their understanding of our travel patterns to suggest pertinent alternative itineraries in particular when unexpected events happen to affect our usual one (e.g. stopped train, empty bike station).

To infer the itinerary of a user, our system uses a Transportation network to generate candidate itineraries. We introduce an algorithm, called *Multimodal Itinerary Matching*, that looks for the itinerary of maximum probability given some observations of a user.

In such a setting, a critical aspect is that of the data that is used in the algorithm. Sensor data can be classified based on the kind of knowledge they deliver (e.g., location, dynamics) and their characteristics (e.g., frequency, accuracy). We use user observations obtained from multiple smartphone sensors, namely, GPS, Wi-Fi, Cellular, Accelerometer and Bluetooth. The system has also access to geographic data (roads, railways), public transportation data (routes and schedules). The system constructs a *Transportation network* from OpenStreetMap [11] data and public transportation routes published online by transportation agencies in GTFS [9] format. Difficulties arise from lack of data (e.g., lack of positioning inside the subway), from too much data (e.g., combination of possibly conflicting localization data, overlapping public transportation lines), and inaccuracies or imprecisions in the data (e.g., map errors, imprecise location).

The algorithm infers, from these observations and the Transportation network, the multimodal itinerary of the user. The algorithm has two phases. The first phase uses a dynamic Bayesian network [19] that models the probabilistic relationship between paths in the Transportation network and sensor data. The result is a path in the Transportation network employing the following modes: `foot`, `bicycle`, `road_vehicle`, and `rail`. Inference is performed using a particle filter [7, 8, 28]. The second phase refines this path by matching unimodal segments recognized as `road_vehicle` or `rail` with public transportation routes of type `bus`, `train`, `metro`, or `tram` directly extracted from the GTFS source. For each candidate unimodal segment, a score is computed with respect to a candidate public transportation route. Finally, the candidate itinerary with the highest score is output. We evaluated the performance of the system using data recorded from users traveling in the Paris area. For this, we provided them with an Android application, *Hup-me:User*, we developed that records the required smartphone sensor data. We manually annotated the journeys that had been recorded. Our annotations included the transportation modes as well as the public transportation routes that had been taken. The Paris region is rich in public transportation modes: sub-urban train, metro, bus, tram. We compared the findings of the algorithm to the annotations that are assumed to be correct. We also report on the execution time of the algorithm for varying lengths of observation sequences.

SIGSPATIAL'15 November 03 - 06, 2015, Bellevue, WA, USA

ACM 978-1-4503-3967-4/15/11 ...$15.00.
DOI: http://dx.doi.org/10.1145/2820783.2820852.

2. INPUTS OF THE ALGORITHM

Our algorithm takes two kinds of input data: Transportation network data, and sensor data.

Transportation Network. We use the notion of a *Transportation network* to define the itineraries that a user may take. We define it as a series of *traversability* rules over a connected road and rail infrastructure, termed Spatial network. The Spatial network models places on Earth (termed nodes) such as road intersections, subway stations, and their linear connections (termed edges) such as roads and metro line tunnels. The Transportation network captures movement constraints with respect to a set of transportation modes. For instance, by foot, one may decide to do a U-turn between two spatial nodes, and reverse one's direction, whereas one cannot do that in a train or on a freeway. A *location* in a Transportation network $\mathcal{T}$ is a tuple (e, δ, σ, d), where e is an edge of the Spatial network, δ is a direction (either *forward* or *backward*), σ is a transportation mode, and d is an offset in $[0, \text{length}(e)]$. A *path* in $\mathcal{T}$ is defined as a sequence of locations in $\mathcal{T}$, where each pair of consecutive locations verifies $\mathcal{T}$ transition rules. The Spatial network and Transportation network are built from online public sources. For both, geographic information from OpenStreetMap (OSM) is used, and the latter uses additional rail transportation data from public transportation schedules published by transportation agencies in the General Transit Feed Specification format. Rail transportation schedules and OSM railway data are automatically combined using an alignment algorithm we designed.

Sensor data. We designed *Hup-me:User*, a smartphone app to collect mobile sensor data from traveling users. The app collects raw data over a given time interval from different sensors: Satellite Navigation (GPS/GLONASS), Wi-Fi, Cellular, Accelerometer and Bluetooth. To recognize the user's itinerary, raw sensor data is transformed into a feature vector of a smaller dimension that is composed of three main dimensions: location, dynamics (e.g. acceleration), and contextual (e.g. Bluetooth).

Location observation. Smartphone's location is determined with varying degrees of accuracy using Satellite Navigation (10-meter accuracy), Wi-Fi Networks (100-meter) and/or Cellular Networks (1-kilometer). Satellite location is sampled at 1 Hz and is also capable of measuring the movement's speed vector. Satellite location might be unavailable in areas with a partial line of sight of the sky (e.g. building, underground). A location is given by a center point, an accuracy value and, if available, a speed value. The accuracy is given by the estimated standard deviation of the location error. The location sequence forms a non-uniform time-series over a finite time interval. The sequence if resampled at a rate of 1 Hz by picking the sample with the highest accuracy (i.e. with the smallest radius) over a sliding 1-second window. If no sample is available within the window, no observation is retained. For performance reasons, locations above a certain accuracy threshold are discarded.

Dynamics observation. Accelerometer data sampled at 20 Hz is processed using existing state of the art techniques [24, 25] from which we output confidence levels over possible 5 possible activities: `stationary`, `foot`, `bicycle`, `motorized`, `unknown`. The activity `unknown` is used when the recognition of any other failed.

Contextual. Information about the user's environment can be inferred from the signal levels of Bluetooth, Wi-Fi, Cellular and Satellite. Signal reception depends on whether the user is in a building, underground, or outside in an open-field. An interesting property of Bluetooth is that more people carry discoverable network emitters with them (smart-watches, smartphones). From the number of discoverable Bluetooth network around the user it might be possible to estimate to number of people around. Also, if other people happen to move with the user, such as when taking the bus or metro, but also when sharing a car with multiple people, then some of these networks will stay in range over the course of the journey. The number of networks that stay in range over two consecutive scans can be computed from the unique identifiers used by network emitters. We call this figure *recurrent address count* (RAC). For each signal type, we measure three dimensions: *Number of emitters* (satellites, cellular towers, Wi-Fi access-points, discoverable Bluetooth devices), *Signal-to-Noise Ratio/Received Signal Strength Indication* (mean and variance over emitters) and *RAC*.

3. THE ALGORITHM

We developed an algorithm, called *Multimodal Itinerary Matching*, that given a Transportation network $\mathcal{T}$, and a sequence of observations $o_{1:T}$ finds a sequence $(l_1, \ldots, l_T)$ of locations in $\mathcal{T}$ and a path $p = l_1\pi_1 l_2 \cdots l_{T-1}\pi_{T-1}l_T$ between these locations (called the *itinerary*) *best matching* the observation sequence. The algorithm works in two phases: **1. Filtering**, which derives the most likely itinerary in $\mathcal{T}$ for the modes `foot`, `road_vehicle`, `bicycle`, and `rail`, and **2. Smoothing**, which derives the most likely public transportation line for `rail` and `road vehicle` modes.

Filtering. To implement Phase 1, we use a probabilistic algorithm based on a dynamic Bayesian network [14] (DBN). The DBN has time span T. The DBN models the state of a traveler at each time-step t via a set of random variables, as well as conditional probability distributions relating observations at time t with respect to the traveler state at time t and relating the traveler's state at time $t+1$ with respect to her state at time t. The set of random variables modeling the traveler state is split into two groups $\boldsymbol{x}_t$ and $\boldsymbol{y}_t$, where $\boldsymbol{x}_t$ is the single continuous-domain variable representing the traveler's offset on a edge of the Spatial network, while $\boldsymbol{y}_t$ represents finite domain variables such as the traveler's mode, the path she followed, or mode transition times. The algorithm, termed *Multimodal Itinerary Filtering*, processes observations in sequence and generates candidate intermediate itineraries by projecting each location observation into a set of candidate Transportation network $\mathcal{T}$ locations, and generating candidate paths between each consecutive pair of candidate $\mathcal{T}$ locations. It performs approximate inference by maintaining a posterior density estimate of $\Pr(\boldsymbol{y}_{1:t}, \boldsymbol{x}_t \mid \boldsymbol{o}_{1:t})$ for each time t. This estimate is maintained based on the Rao-Blackwellized particle filter [7, 10] that generates samples of $\boldsymbol{y}_{1:t}$ while keeping $\boldsymbol{x}_t$ in closed form using a Kalman Filter. In particular, our filter implements *sequential importance resampling*, which uses a *proposal distribution* $q(\boldsymbol{y}_{1:t} \mid \boldsymbol{o}_{1:t})$ to generate samples and attaches weights to particles to approximate the estimated density. After all observations have been processed, Phase 1 outputs an approximate Maximum A Posteriori estimate $y_{1:T}$, that is, a valuation $y_{1:T}$ for which $\Pr(\boldsymbol{y}_{1:T} = y_{1:T} \mid \boldsymbol{o}_{1:t}) = \int_{\boldsymbol{x}_T} \Pr(\boldsymbol{y}_{1:T}, \boldsymbol{x}_T \mid \boldsymbol{o}_{1:T})\, \mathrm{d}\boldsymbol{x}_T$ is approximately maximal.

Smoothing. Phase 1 of the algorithm ignores the distinction between certain transportation modes. Phase 2 refines this analysis, by inferring the exact transportation mode of `road_vehicle` amongst `car` and `bus`, and the transportation mode of `rail` amongst `tram`, `metro`, `train`. For public transportation modes, we infer the actual routes as defined by the GTFS data. Phase 2 also smooths out

transportation mode transitions to clean up the final itinerary.

Let $p_1 \cdots p_T$ be the path output by the first phase. We split it into a sequence of r unimodal path segments $q_1, \cdots, q_r$. The second phase, begins by reconsidering the mode for any short-length or short-duration segment q_i in-between two segments q_{i-1} and q_{i+1} whose modes are equal by comparing the mean estimated speed and variance of q_i to some threshold tuned for the estimated mode. Then, given the sequence of mode transition times output by the first phase, we can compute the duration, the starting time and ending time of each segment q_i. For each q_i such that its mode is either `road_vehicle` or `rail`, the second phase aims at determining possible matching GTFS routes for q_i and the matching sequence of stops (a substring of the route stop sequence). Candidates routes are computed by looking at stops within a distance R of the start and end locations of q_i. To rank a GTFS route ρ and a sequence of stops $(s_1, \ldots, s_m)$ we use the following metrics: (a) the average distance between each s_i and the set of locations in q_i; (b) the difference between q_i's time duration, and for a vehicle ρ at that particular time of the day; (c) the difference in departure time between either the immediately previous trip or immediately next trip for route ρ departing before or after q_i's start time. The result is a mapping from a candidate route and a stop substring to $\mathbb{R}^3$. The best candidate route is picked using a score value computed from I. Finally, for `road_vehicle` segments, we differentiate between `car` and `bus` by using the average Bluetooth component of contextual observations (a $\mathbb{R}^3$ vector) over the period of q_i.

4. EVALUATION

To evaluate our algorithm, we used Transportation network data for the Paris region. It encompasses all roads, railways, tram tracks. It also knows of train, metro, bus, tram routes. The following gives some intuition on the size of the Transportation network: 808,093 nodes, 1,045,052 road edges, 34,414 rail edges, 3,412 trip patterns.

We implemented the Multimodal Itinerary Matching algorithm as well as all the necessary data integration, network construction, path generation algorithms and Android application Hup-me:User in Scala. We use manual annotations on observation traces to evaluate different features of the algorithm and fine tune it. In order to facilitate annotations, we also developed a Web interface that displays location data on a map and the other observations on a series of charts. The annotator then chooses a mode and possibly a route and or trip pattern for different time intervals in a journey.

We evaluated our implementation in parallel on a 16-core virtual machine with 104 GB memory implemented by a 2.3 GHz Intel Xeon E5 v3. We set the number of particles to 2000. We measured the execution time of our algorithm with respect to the duration of a journey. The execution time is roughly linear in the journey duration. On average, the algorithm takes 0.1638 CPU seconds per second of journey, with a median of 0.108. We believe the performance can still be improved, especially with respect to memory consumption, which has taken up to 10GB for some journeys.

We collected and annotated 87 journeys (42.5 hours) in the Paris region. Manual annotations were made by one person, prior to running the algorithm. The annotator was familiar with Paris transportation system and was given access to all data available to *Hup-Me*. Additionally, part of the annotations were given by the travelers themselves through the mobile application, however most travelers found it tedious to enter annotations, and most of the annotations were provided by the annotator. In most cases the annotator was confident in his annotations, in very few cases he was not completely sure. After running the algorithm the annotator was confronted with the results of the algorithm. The algorithm validated his annotations, and in the few cases there was disagreement, the annotator believes

		Predicted mode						
		`foot`	`bike`	`car`	`bus`	`train`	`tram`	Time
Actual mode	`foot`	87 %	8 %	1 %	1 %	2 %	1 %	$1068'$
	`bike`	2 %	98 %	0 %	0 %	0 %	0 %	$69'$
	`car`	5 %	2 %	82 %	10 %	0 %	0 %	$718'$
	`bus`	4 %	5 %	0 %	90 %	1 %	0 %	$419'$
	`train`	12 %	0 %	2 %	3 %	83 %	0 %	$149'$
	`tram`	15 %	3 %	6 %	1 %	0 %	75 %	$129'$
Precision		91 %	36 %	96 %	80 %	81 %	92 %	$2552'$

Table 1: Confusion matrix by transportation mode.

	`bus`	`train`	`tram`
	95 %	78 %	99 %
Total Time	$381'$	$127'$	$98'$

Table 2: Percentage of correct transportation line recognition.

the algorithm was wrong. We measured the fraction of time the algorithm predicted j when the true (according to the annotator) mode was i. The results are presented in what we call the confusion matrix (Table 1). The `train` and `metro` modes are merged (noted `train`) because the dataset was not large enough to analyze this distinction. The *Time* column gives the cumulated journey duration for each mode. When the algorithm guessed the correct mode amongst `bus`, `train`, `tram`, we evaluated percentage of time for which the correct transportation line was detected (Table 2). Overall, the results are comparable to the state of the art [12, 25], considering we infer a full itinerary and transportation lines. More work has to be done to improve distinguishing (i) between `bus` and `car` modes and (ii) between different underground lines. We need to further analyze the relevance of each sensor in different recognition tasks, to improve the performance of the algorithm. Finally, we want to design a more sophisticated parameter learning strategy from annotated journeys.

5. CONCLUSION & STATE OF THE ART

Activity recognition, that is concerned with determining the actions and goals of one or several agents given a series of observations on their actions and the environment, has gained increased attention over the years in the fields of articial intelligence, robotics, and ubiquitous computing. Our work follows pioneering works published over the last ten years on sensor-based activity recognition, and notably accelerometer-based approaches [4, 15, 18, 21, 23]. More recently, several researchers have focused on transportation mode identification [12, 24, 29, 30]. Using activity recognition techniques, we can determine at a point in time whether the user is currently *stationary, walking, running, cycling* or in a *motor vehicle*. In fact, this technology is already available for developers on the two most widespread smartphone platforms, namely Android [1] and iOS [2].

Our goal is to determine the transportation modes over the course of a journey as well as the traveled route. To serve this purpose, and provide context awareness to travel assistant technologies, location-based activity tracking were developed, that use positional information from an embedded GPS chip to track the user's most frequent locations, travel routines and routes taken [3, 6, 16]. Location-based vehicle tracking has also been successfully used to generate accurate schedules and provide information about traffic delays to the rest of the community [5, 26, 27]. Our interest in the long run is to provide the user with real-time itinerary-aware applications and with an accurate summary of one's routines. To this respect, our work is closely related to [6, 16]. They use a combination of online activity recognition techniques and location-based map-matching. However, these works have limitations. They do not distinguish between overlapping public transportation routes with different schedules and do

not handle long periods of missing GPS observations. We tackled these two issues in our system.

Map-matching is the problem of matching a set of positional observations to a path in a given road network. An exhaustive survey of map-matching techniques can be found in [22] and more recent results in [13, 17, 20]. Kalman and particle filtering on a dynamic Bayesian network are known techniques in the map-matching literature. However, as stated by [13], dynamic Bayesian networks encounter the *selection bias problem* at low frequencies. Instead, they use, as well as [30], a Conditional Random Field. For this reason, we exploit observations of many more sensors. Finally, we also take into consideration timetable and the possibility for geographic routes to belong to several transport modes and lines. This increases the difficulty of the problem. In the past, both static [26] and real-time [25] timetables have been used for tracking user in a transportation network.

In itinerary detection, the use of a dynamic Bayesian network is not new. Our work, is to our knowledge, the first one considering such a complex transportation network (multimodal distinguishing lines and timetables), with such a rich combination of user data (both positional sensors and accelerometer). Another novelty of our approach is in the processing of long periods of time without GPS observations, which happen frequently in high density urban areas, e.g., with an underground transit system. The current evaluation of the algorithm is very promising with respect to mode and trip pattern recognition. In the future, we plan to perform a more comprehensive evaluation. A main motivation for Hup-me is to enrich available personal data (calendar, mail, search history, etc.) to provide new functionalities such as *personal travel assistance*.

References

[1] Android Developers Guide. *Recognizing the User's Current Activity*. 2013. URL: `http://developer.android.com/training/location/activity-recognition.html` (visited on 05/16/2014).

[2] Apple Inc. *CMMotionActivity*. 2013. URL: `https://developer.apple.com/library/ios/documentation/CoreMotion/Reference/CMMotionActivity_class/` (visited on 05/16/2014).

[3] D. Ashbrook and T. Starner. "Using GPS to learn significant locations and predict movement across multiple users". *Personal and Ubiquitous Computing* (2003).

[4] L. Bao and S. Intille. "Activity Recognition from User-Annotated Acceleration Data". In: *Pervasive Computing*. Lecture Notes in Computer Science. Springer, 2004.

[5] J. Biagioni, T. Gerlich, T. Merrifield, and J. Eriksson. "EasyTracker: Automatic Transit Tracking, Mapping, and Arrival Time Prediction Using Smartphones". In: *ACM SenSys*. 2011.

[6] J. Chen and M. Bierlaire. "Probabilistic multimodal map-matching with rich smartphone data". *Journal of Intelligent Transportation Systems: Technology, Planning and Operations* (2013).

[7] A. Doucet, N. de Freitas, K. Murphy, and S. Russell. "Rao-Blackwellised Particle Filtering for Dynamic Bayesian Networks". In: *UAI*. 2000.

[8] A. Doucet, N. Freitas, and N. Gordon. "An Introduction to Sequential Monte Carlo Methods". In: *Sequential Monte Carlo Methods in Practice*. Statistics for Engineering and Information Science. Springer, 2001.

[9] Google. *General Transit Feed Specification Reference*. 2015. URL: `https://developers.google.com/transit/gtfs/reference` (visited on 03/23/2015).

[10] N. Gordon, D. Salmond, and A. Smith. "Novel approach to nonlinear/non-Gaussian Bayesian state estimation". *IEEE Radar and Signal Processing* (Apr. 1993).

[11] M. Haklay and P. Weber. "OpenStreetMap: User-Generated Street Maps". *IEEE Pervasive Computing* (2008).

[12] S. Hemminki, P. Nurmi, and S. Tarkoma. "Accelerometer-based Transportation Mode Detection on Smartphones". In: *ACM SenSys*. 2013.

[13] T. Hunter, P. Abbeel, and A. M. Bayen. "The Path Inference Filter: Model-Based Low-Latency Map Matching of Probe Vehicle Data". *IEEE TITS* (2014).

[14] D. Koller and N. Friedman. *Probabilistic Graphical Models - Principles and Techniques*. MIT Press, 2009.

[15] J. R. Kwapisz, G. M. Weiss, and S. A. Moore. "Activity recognition using cell phone accelerometers". *ACM SigKDD Explorations* (2011).

[16] L. Liao, D. J. Patterson, D. Fox, and H. Kautz. "Learning and inferring transportation routines". *Artificial Intelligence* (2007).

[17] Y. Lou, C. Zhang, Y. Zheng, X. Xie, W. Wang, and Y. Huang. "Map-matching for Low-sampling-rate GPS Trajectories". In: *ACM SIGSPATIAL*. 2009.

[18] U. Maurer, A. Smailagic, D. P. Siewiorek, and M. Deisher. "Activity recognition and monitoring using multiple sensors on different body positions". In: *IEEE BSN*. 2006.

[19] K. P. Murphy. "Dynamic bayesian networks: representation, inference and learning". PhD thesis. University of California, 2002.

[20] P. Newson and J. Krumm. "Hidden Markov Map Matching Through Noise and Sparseness". In: *ACM SIGSPATIAL*. 2009.

[21] J. Parkka, M. Ermes, P. Korpipaa, J. Mantyjarvi, J. Peltola, and I. Korhonen. "Activity classification using realistic data from wearable sensors". *IEEE TITB* (Jan. 2006).

[22] M. A. Quddus, W. Y. Ochieng, and R. B. Noland. "Current map-matching algorithms for transport applications: State-of-the art and future research directions". *Transportation Research Part C* (2007).

[23] N. Ravi, N. Dandekar, P. Mysore, and M. L. Littman. "Activity recognition from accelerometer data". In: *AAAI*. 2005.

[24] S. Reddy, M. Mun, J. Burke, D. Estrin, M. Hansen, and M. Srivastava. "Using mobile phones to determine transportation modes". *ACM TOSN* (2010).

[25] L. Stenneth, O. Wolfson, P. S. Yu, and B. Xu. "Transportation mode detection using mobile phones and GIS information". In: *ACM SIGSPATIAL*. 2011.

[26] A. Thiagarajan, J. Biagioni, T. Gerlich, and J. Eriksson. "Cooperative transit tracking using smart-phones". In: *ACM SenSys*. 2010.

[27] A. Thiagarajan, L. Ravindranath, K. LaCurts, S. Madden, H. Balakrishnan, S. Toledo, and J. Eriksson. "VTrack: accurate, energy-aware road traffic delay estimation using mobile phones". In: *ACM SenSys*. 2009.

[28] S. Thrun, W. Burgard, and D. Fox. *Probabilistic robotics*. MIT press, 2005.

[29] S. Wang, C. Chen, and J. Ma. "Accelerometer Based Transportation Mode Recognition on Mobile Phones". In: *IEEE APWCS*. Apr. 2010.

[30] Y. Zheng, Y. Chen, Q. Li, X. Xie, and W.-Y. Ma. "Understanding transportation modes based on GPS data for web applications". *ACM TWEB* (2010).