Human Mobility Simulator for Smart Applications

Alessandra De Paola, Andrea Giammanco, Giuseppe Lo Re and Marco Morana
Department of Engineering, University of Palermo
Palermo, Italy
Email: alessandra.depaola@unipa.it, andrea.giammanco@unipa.it, giuseppe.lore@unipa.it, marco.morana@unipa.it

***Abstract*—Several issues related to Smart City development require the knowledge of accurate human mobility models, such as in the case of urban development planning or evacuation strategy definition. Nevertheless, the exploitation of real data about users' mobility results in severe threats to their privacy, since it allows to infer highly sensitive information. On the contrary, the adoption of simulation tools to handle mobility models allows to neglect privacy during the design of location-based services. In this work, we propose a simulation tool capable of generating synthetic datasets of human mobility traces; then, we exploit them to evaluate the effectiveness of algorithms which aim to detect Points of Interest visited by users of a Smart Campus. Our simulator exploits an activity-based mobility model, thus it is based on the assumption that mobility of campus users is motivated by the activities they plan to perform. It is capable of simulating the weekly repetitiveness of human behavior and to model different mobility profiles for each day of the week through a fifth-order Markov model.**

***Index Terms*—Human Mobility Simulation, Smart Cities, Smart Campus, Markov model**

I. Introduction

Many Smart City applications require the knowledge of human mobility models. In particular, location-aware services aim to users' location from raw trajectory data, in order to provide users with personalized services. Despite the increasing diffusion of personal devices it makes very easy to keep track of users' trajectories, the wide use of such data during the design phase poses severe threats for users' privacy. It has been shown that even coarse spatio-temporal datasets, e.g., with 1 hour as temporal resolution, provide a poor level of anonymity [1], therefore, the adoption of a simulation tool to generate synthetic mobility data represents a good trade-off between privacy protection and accurate performance evaluation during the design phase.

Simulating human mobility is useful for addressing several issues. Simulating crowd mobility is critical task for evaluating the suitability of evacuation strategies during natural emergencies or artificial disasters [2]. Moreover, in the field of transportation and civil engineering, models of site-specific pedestrian mobility is useful in order to design walking facilities, and large structures such as stadiums or shopping malls [3]. Finally, mobility is one of the key elements influencing the performance of ad hoc networks [4], as well as the spatial diffusion of information [5].

In this paper, we propose a simulation tool for human mobility in a Smart Campus. Despite the specific scenario which drives its design, it can be easily applied to other Smart City scenarios. A *Smart Campus* [6], [7] is a digitally augmented campus where pervasive instrumented objects and spaces are made responsive to the state of the environment and its inhabitants. In a Smart Campus, in addition to the ubiquity of users' smartphones, several other sensory devices, such as cameras, RFID readers, and bluetooth beacons, collect raw measurements, that can be exploited by an intelligent system in order to reason upon current context and supply advanced services to users.

We propose an activity-based mobility simulator for Smart Campus users, which generates synthetic GPS tracks of their movements. Such tool allows to assess the performance of users profiling techniques, as well as personalized location-based services offered to users, before their real deployment. The proposed tool exploits the intentional and recurrent nature of human mobility [8], and models its weekly periodicity [9]. Since the main applications for which our simulator has been developed concern the supply of context-aware services, the simulator is focused on macro-mobility rather than micro-mobility features, that is it considers where users go (their destinations), and not the specific followed route nor the movement speed.

As proof of concept, we used the proposed simulator to assess the performance of some Point-of-Interest (PoI) detection algorithms. An accurate evaluation of such algorithms is of the utmost importance since they often represent the first building block to design context-aware services, such as location-based recommendations.

The remainder of the paper is organized as follows. Section II reviews the literature about human mobility simulation. Section III outlines the considered Smart Campus ICT architecture, by providing a high level description of its components. Section IV describes the proposed activity-based human mobility simulator. Section V describes some PoI detection algorithms used as proof of concept. Section VI describes the experimental evaluation enabled by our simulator. Finally, Section VII states some conclusions and discuss the future work.

II. Related Work

This Section outlines previous studies on human mobility models and simulation. Despite human mobility is extremely regular and repetitive [8], building a model which takes into account all possible details is impractical. Consequently, models and simulators presented in the literature focus only on domain-dependant aspects of interest. Available tools adopt different granularity to analyze movements [3]: at a high level of abstraction, *macro-mobility* concerns users' activities affecting their movements; on the contrary, micro-mobility concerns which specific paths people follow to reach their destinations, as well as the walking speed and direction taken in order to avoid collisions with obstacles and other pedestrians. Micro-mobility models are particularly useful to simulate emergency evacuations [2].

Models focused on micro-mobility aspects, pay little attention to specific destinations and to the waypoint visit order, which are in general randomly chosen. LEGION simulator [10] is a commercial solution for civil engineers, which aims to assess the impact of different levels of pedestrian demand on infrastructures. It is often used in urban planning to capture speed-distance relations that emerge when pedestrians walk while avoiding obstacles and other pedestrians. PEDSIM [11] is an open source library for micro-mobility pedestrian crowd simulation, useful for indoor evacuation simulations, or large scale outdoor simulations. Its simulation model is based on the *social force model*, a differential equation which takes into account the desired velocity, the interaction between pedestrians, and the interaction with the environment. Authors of [12] model human walks by means of statistical features which are similar to Levy Walks model, with power-law distributions of trip lengths between waypoints and pause times in the waypoints. The authors propose to use Least Action Trip Planning to generate trip sequences between randomly selected waypoints, thus neglecting any aspect related to people habits.

On the contrary, macro-mobility models take into account user activities and generate user movements consequently. Authors of [13] propose a Weighted Waypoint model, where transition probabilities between waypoint pairs are represented by means of a Markov chain. Similarly, authors of [14] propose a user behavioral model composed of a time table for specifying typical stay times for each waypoint, and a Markov chain that represents transition probabilities between places. They generate paths between waypoints by means of Google Maps APIs, as well as a set of usual and opportunistic waypoints using Google Places APIs for places of common interest, e.g., shops or restaurants, and a random generation algorithm for selecting private houses or workplaces.

Authors of [3] propose an activity-driven tool to simulate the mobility of theme park visitors. They model the environment spatial layout by means of walking areas, which are sequences of walkable segments obtained through Open Street Map, and activity areas, which are the theme park's attractions represented through polygons. At a micro-mobility level, authors use the Dijkstra algorithm to calculate shortest paths between activity areas. Moreover, guests are associated to a field of view, and a collision avoidance algorithm triggers a direction change whenever an obstruction of the field of view is predicted. Instead, at a macro-mobility level, authors identify three different activities a guest of the theme park can do: walking through the park, visiting an activity area, or waiting in a queue. User's preferences for next activity area are modeled by means of a Markov chain. Moreover, each area has its own visit duration distribution which is sampled at every visit.

In this work, we propose an activity-driven simulator for human mobility, drawing inspiration from [14] and [13], and which explicitly considers the repetitiveness of weekly user routines. To the best of our knowledge, such feature is a novelty with respect to other existing tools. Since, the main goal of such simulator is enabling the assessment of context-aware algorithms, which only rely on the sequence of Points of Interest visited by users, it neglects micro-mobility features, such as collision and congestion avoidance. It is worth noticing that our simulation tool does not allow to model exploratory users [15], i.e., random walkers which wander between a larger number of different locations. However, modeling such behavior is beyond the aim of our work.

III. Smart Campus architecture

This Section briefly describes the Smart Campus architecture for which the simulation tool proposed here has been designed, preliminary described in [16]. Such description allows to highlight which information are relevant for our system, and thus are considered during the simulation, and which aspects can be neglected.

For our purposes, a *Point of Interest* (PoI) is defined as a place where the user usually goes and stops for a while. It could be a place of interest for the whole community, such as a shop or a bus stop, or it could be a place of interest for a single user. Typical PoIs inside a campus include departments, libraries, parks, research laboratories, auditoriums and lecture halls.

The detection of PoIs visited by users during a day is a crucial task in order to understand the nature of the movements they carry out, and to recognize different user profiles. For example, by considering also the main activities performed in each PoIs, it is possible to detect that a specific user is a student who regularly visits the library in the afternoons.

The considered Smart Campus system relies on a multi-tier architecture that allows to extract relevant information from raw data in order to provide context-aware services for users. Such architecture is shown in Figure 1, with a detailed focus on the role played by the *PoI Detection* subsystem. The lowest layer of such architecture is responsible for detecting relevant events and monitoring physical phenomena through a pervasive sensory infrastructure [17]. Raw data gathered by such infrastructure, after a light preprocessing phase, are then exploited in order to perform the PoI detection.

The *PoI Detection* subsystem analyzes trajectory raw data to detect a set of *Stop Points*, which can be defined as regions where the user remains for a certain amount of time, and then merges the identified *Stop Points* with high-level context information in order to detect the actual visited PoIs. Such

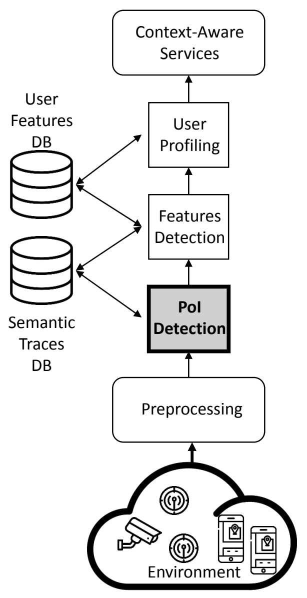

Fig. 1. Architecture of the considered Smart Campus system.

context information summarizes the relationship between each user and the activities he performs in different PoIs, in terms of frequency of visit, average stop time, and sequence of visited PoIs.

The detected PoIs are sent as input to the *Features Detection* module in order to extract a set of n-dimensional points which summarize the mobility behavior of each user, such as its predisposition to be sedentary or exploratory during the workday.

The *User Profiling* module then aims at outlining different classes of users according to the features previously extracted. Besides behavioral characteristics inferred from the such knowledge, user profiles also include information explicitly provided by the users themselves, such as their role (e.g., “researcher” or “student”).

Finally, knowledge about user profiles and their current and next position and activity can be exploited in order to provide users a set of *Context-Aware Services*.

For instance, a location-based recommender system can provide information relevant to the position of users, e.g., suggesting the nearest free library seat. Moreover, the capability of predicting next users location can be exploited to provide recommendations related to the next place a user will visit, e.g., enabling the suggestion of free parking space near the next destination.

IV. Mobility Traces Simulator

In this work, we propose a simulation tool capable of generating synthetic datasets of GPS traces of users in a Smart Campus. The proposed tool, whose architecture is shown in Figure 2, is based on the human mobility studies proposed in [8], [18], [19] and it has been designed by following the guidelines proposed in [13], [14].

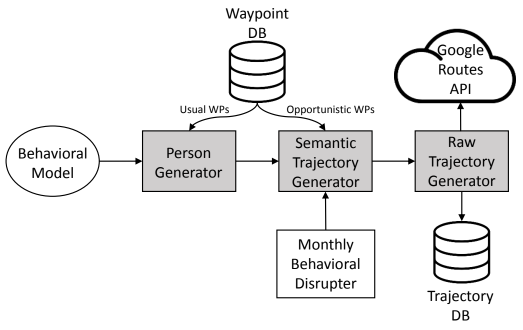

Fig. 2. Architecture of the simulation tool proposed to generate synthetic GPS trajectories.

The simulation requires the explicit knowledge of Campus PoIs. Since paths between pairs of PoIs inside the campus are automatically retrieved through the Google Routes APIs, there is no need to explicitly specify walking areas, thus making the spatial representation of the campus lighter. Data obtained from Google Routes are stored as JSON files in a dedicated DB to avoid further unnecessary APIs calls.

PoIs are represented through areas whose vertexes are stored in a specific DB, named *Waypoint DB in Figure 2*. Each PoI is associated with a label that represents a class of PoIs, and can be related to activities performed by users in that type of places. The set of available labels and the association between PoIs and label is modeled through an opportune ontology [20], thus making our tool adaptable for different campus settings. In our case, 8 classes are considered: faculty building, coffee shop, recreational place (e.g., parks), administration building, parking area, library, ceremonial hall, entrance.

The main features considered for simulations are the behavioral habits of users. Human movements are repetitive and intentional, because users move from one place to another by following a to-do list, and a place change generally corresponds to an activity change. Moreover, in a Campus, user’s habits are also related to the day of the week [8], since users show a tendency to repeat some specific actions at specific days and day times. Therefore, each class of users is modeled through a set of parameters whose values depend on the day of the week. These parameters include: label of the places where users stay most of the time; typical arrival time and stay time at the campus; transport mode used to reach the campus (a user traveling by car looks for a parking area after entering the campus); probability of leaving the campus before the usual time; typical lunch time and preferred place’s label for lunch break. By tuning parameters of such models, different classes of users can be simulated. For instance, it is possible to model a student which attends morning classes, arrives at the campus around 8 AM and has lunch inside his faculty building on odd-numbered days, whereas he attends afternoon classes and arrives around 14 PM on even-numbered days. It is possible to model students with a high campus drop rate, students which respect scheduled plans, as well as students which go to library on Friday and stay at home during exam sessions.

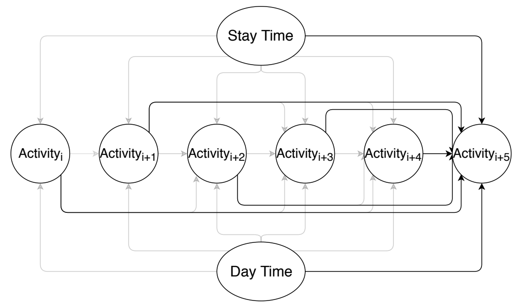

Fig. 3. The fifth-order Markov chain which models the transition probability from an activity to another.

Moreover, each user is associated to a probabilistic model for his specific mobility patterns. Such *behavior* is composed of two components: (i) a *transition scheme*, which is a fifth-order Markov model representing the transition probability from an activity to another, shown in Figure 3, and (ii) a *pause scheme* which represents the probability distribution of stop time for each waypoint. The reason behind the choice of a fifth-order Markov model for the transition scheme is that, on average, people inside a campus visit 4.5 places per day [18].

On a monthly basis, parameters are subject to small perturbations in order to simulate possible changes in user routine. Moreover, in some specific months, the added noise is higher, thus simulating a behavioral shift during some particular campus-life periods such as exam sessions.

After the selection of the next activity to be performed, the simulator selects the corresponding class of PoI to visit. Then, for each user, in each day, the simulator generates a *semantic trajectory*, i.e., a sequence of labels. An example of semantic trajectory is the sequence <*Faculty building* → *Coffee Shop* → *Library*>.

Given the selected PoI class, the specific PoI to visit is chosen through a gap-reducing approach according to which a user visits the nearby destinations before visiting farther destinations [19]. Moreover, in order to model a slight exploratory behavior, for each selection there is a small probability to select a random PoI. This random selection is performed by associating each PoI to a probability which exponentially decays with respect to its distance from the current Stop Point.

Finally, raw trajectories, i.e. sequences of $(latitude, longitude, timestamp)$ triples, are generated from PoI sequences by exploiting *Google Routes APIs*. In particular, for each pair of consecutive waypoints, a set of position samples is generated, and then such data are corrupted by adding random noise in order to simulate the GPS errors. Noise features can be specified as input parameters. The dataset described here has been generated by adding Gaussian noise with mean value equal to 0 and standard deviation equal to 0.00005, thus to obtain some noisy readings far 10 mt from the true user's position.

The experimental evaluation presented in this work has been performed by generating a dataset of trajectories for different users, covering an academic year with a sampling rate of 5 minutes, by considering several user behavioral models.

V. Case Study

This Section describes the PoI detection algorithms used as case study in order to validate the simulator suitability. Such algorithms address the *personalized automatic check-in problem* [21], which can be divided into two sub-problems: the *Stop Points* discovery from data, and the *Checked-in PoI* detection for each stop point.

For the sake of clarity, it is opportune to formally define the concepts of *Measurement Point*, *Stop Point* and *Checked-in PoI*.

- A *Measurement Point* is a triple $(latitude,\ longitude,\ timestamp)$ collected through a sensor (e.g., GPS sensors, RFID readers, Wi-Fi access points) which indicates the presence of the user in a given point at a certain time instant.
- A *Stop Point* is a tempo-spatial cluster of *Measurement Points* which represents a geographic region where the user stopped for a while.
- A *Checked-in PoI* is a PoI, among those contained in the system knowledge base, which has been actually visited by the user.

The user's *Stop Points* can be automatically extracted by means of clustering [22], [23]. Nevertheless, mapping a user's position to a set of known PoIs is not a trivial task [21], [24], [25], due to the intrinsic error in measurements (e.g., smartphones' GPS have an average error of 10 meters [26]) and the presence of areas dense of meaningful places. The user's past behavior can be used to refine the estimation of their location.

In the following, we present the strategies adopted by the considered algorithms to address the *Stop Points* discovery problem, and the *Checked-in PoI* detection problem.

A. Stop Points Discovery

Several approaches have been presented in the literature in order to discover Stop Points from raw data. In particular, approaches described in [22], [23] are characterized by a good trade-off between accuracy and computational burden.

According to authors of [22], which propose the first algorithm considered in our case study, a *Stop Point* is discovered whenever there exist two *Measurement Points*, p_a and p_b, for which the following constraints are satisfied:

- $SpaceDistance(p_a, p_b) < \delta_d$,
- $TimeDifference(p_a, p_b) > \delta_t$,
- $TimeDifference(p_k, p_{k+1}) < \Delta_t,\ \forall k :\ a \leq k \leq b$,

where δ_d is the maximum distance a user can cover to be considered staying in the same *Stop Point*, δ_t is the minimum visiting time necessary to discover a *Stop Point*, and Δ_t is the maximum temporal distance between two consecutive samples to be considered as part of the same cluster of measurements. The main weakness of this approach is that it is not capable of dealing with possible signal losses, which can be frequent when only GPS measurements are used for detecting users location.

Authors of [23] which propose the second algorithm considered in our case study, suggest to solve such issue by relaxing the constraint about the maximum time threshold. *Measurement Points* that violate such constraint are stored in a temporary buffer to be compared with the next *Stop Point*. Whenever a new *Stop Point* is discovered, its distance from the previous one is evaluated; if such distance is under a given threshold, these two points are merged, together with all the *Measurement Points* stored in the buffer.

Even with this improvement, the method proposed in [23] generates too many *Stop Points*, since it does not take into account any information about the *nature* of some waypoints, which often can be known a priori.

To overcome such limitation, we propose to compare the discovered *Stop Points* with information stored in a database of known waypoints, in order to merge consecutive *Stop Points* which match the same waypoint. Such enhanced *Stop Point* algorithm, represents the first step of our Context-aware PoI Detection (CAPD) algorithm, which represents the third algorithm considered as case study.

B. Checked-in PoI Detection

Our Context-aware PoI Detection (CAPD) algorithm exploits results of the *Stop Points* discovery phase in order to identify the PoIs visited by the user, defined as *checked-in PoIs*.

The easiest approach to detect a *checked-in PoI* is to select the known PoI which is nearest to the centroid of the current *Stop Point*. Such strategy, named *nearest neighbor method*, is equivalent to apply a reverse geocoder to *Stop Points* [21], [27]. Despite its simplicity, such an approach is negatively affected by unavoidable errors of GPS sensors and by the presence of areas with a high density of PoIs. The last problem, in particular, can be particularly relevant in a Smart Campus.

To overcome such limits, some methods have been proposed in the literature [21], [24], [25].

The CAPD algorithm adopts a Dynamic Bayesian Network (DBN) [28] in order to include context awareness and the knowledge about past history into the PoI detection algorithm, and in order to deal with the unavoidable noise in sensory readings and in the discovered Stop Points. In the CAPD algorithm, *Stop Points* discovered through the enhanced method described in Section V-A represent the observable manifestation of the hidden user state, i.e., the real checked-in PoI. The DBN allows to model the evolution of the hidden state over time, and the probability dependency of the current state, from past state and from context features.

CAPD uses context information which heavily depends on the recurrent nature of human mobility [9], that can be often modeled through a weekly routine. In particular, it takes into account three different aspects related to the time relation between the discovered *Stop Point* and the potential PoI: the arrival time at the *Stop Point*, the duration of the visit and the day of the week. Furthermore, it exploits the intentional

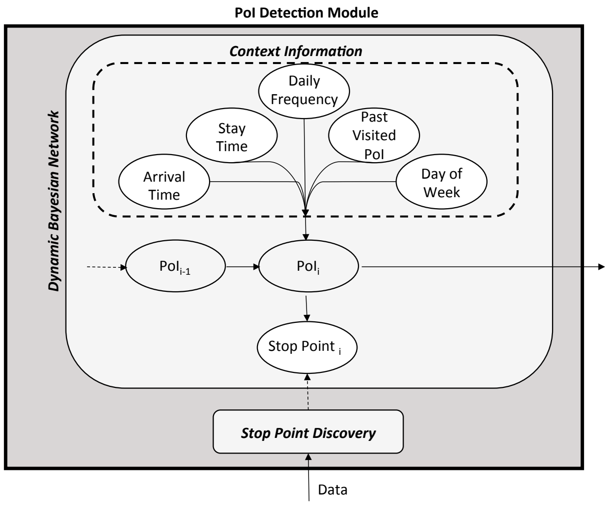

Fig. 4. Block Diagram of the *PoI Detection* module.

nature of human mobility by considering the sequence of past visited places during the day as further contextual feature.

Figure 4 sketches the overall structure of the PoI Detection module, and highlights the adopted DBN and its interaction with the Stop Points Discovery subsystem, which can be considered as a virtual sensor which perceives noisy manifestations of real (unknown) PoI. The main goal of this module is to infer the i-th PoI visited during a day, which is represented by the hidden variable x_i. The belief about such variable depends from the past history, the current sensory reading e_i, i.e. the discovered *Stop Point*, and a set of context information $c_i = (C_i^1 \dots C_i^k)$.

The characterization of the DBN requires the definition of the *sensor model* and the *state transition model*. The probability distribution $P(e_t|x_t)$ represents how *Stop Points* are affected by the current visited PoI, it is named *sensor model*, and it is inversely proportional to the distance between the centroids of the discovered *Stop Point* and the hypothesized PoI x_i. The *state transition model*, i.e., $p(x_i|x_{i-1}, c_i)$, represents the probability that the user visited a given PoI, given the previously visited PoI x_{i-1} and the current context information c_i.

The context information are obtained through a set of feature extractors, that for each user u, and for each PoIs' class β, quantify their relationship in temporal and behavioral terms. The considered features are the following: the frequency of visits during a day, the typical arrival time, the average stop time and the history of waypoints of the same class visited in the past, expressed as frequency of n-grams of previously seen PoIs.

Since such DBN is a first-order Markov model, the *belief* about the PoI visited in the i-th slot, i.e. x_i, can be defined as:

$$belief(x_i) = p(x_i|e_{1:t}, c_{1:t}). \tag{1}$$

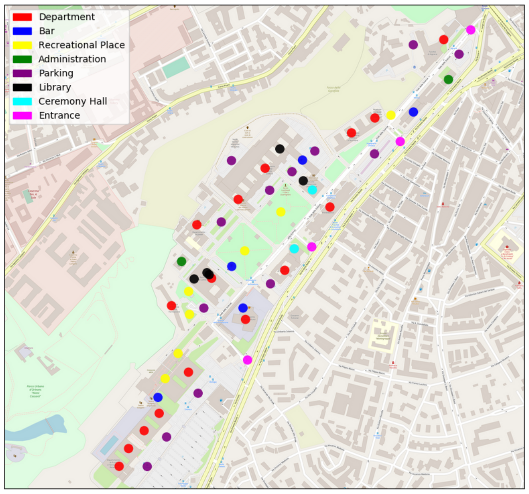

Fig. 5. Distribution of PoIs in the University Campus used for simulations.

As described in [17], [29], the belief can be computed with the following practical formulation:

$$belief(x_i) = \eta \cdot p(e_i|x_i) \cdot \sum_{x_{i-1}} p(x_i|x_{i-1}, c_i) \cdot belief(x_{i-1}), \tag{2}$$

where η is a normalization constant.

VI. Experimental evaluation

This Section illustrates how the simulation tool can be used to assess the performance of PoI detection algorithms in a Smart Campus.

A. Simulation Parameters

Through the proposed tool, we simulated the traces of 100 users, in a University Campus which measures approximatively 350'000 m^2, during an academic year. The topography of such Campus is shown in Fig. 5. The spatial distribution of PoIs, whose centroids are represented by means of different colors, shows the presence of some hubs with multiple near PoIs closed in the same building: these are the zones with the highest error rate. The campus area contains 54 Points of Interest, belonging to 8 different labels. Traces have been generated by using a sampling rate of 5 minutes, and by considering several user behavioral models.

During the simulation, the GPS signal is considered absent when a user moves inside buildings.

The generation of such dataset took 3 minutes on average, by running the simulator on a PC with a Intel i5-3470 with clock rate equal to 3.2GHz, equipped with a 8GB-sized RAM. Since our simulator neglects micro-mobility features, i.e., collision and congestion avoidance, there is no correlation between the trajectories of different users. Therefore, the performances of our tool could highly benefit from distributed or parallel simulations.

Such dataset has been exploited to compare the performance the PoI detection algorithms considered as case study and described in Section V. For these three algorithms, the *Stop Points* extraction has been performed by setting the time threshold δ_t to 9 minutes, and the spatial threshold δ_d to 45 meters.

TABLE I
Comparison of accuracy of Stop Point extraction algorithm

Methods	Accuracy
[22]	0.57
[23]	0.78
CAPD - blind	0.82

B. Performance Metrics

The accuracy of PoI Detection algorithms can be evaluated by comparing the ground truth with the sequence of inferred PoIs, through the Damerau-Levenshtein distance [30]. Let $\Phi(\cdot,\cdot)$ be the Damerau-Levenshtein distance function between two strings, $\mathbf{PoI_G}$ be the ground-truth sequence of PoIs for a specific user, $\mathbf{PoI_D}$ be the sequence of inferred PoIs, and $\#\{\cdot\}$ be the function which gives the numbers of elements in a sequence, then the average accuracy for each user is computed as:

$$accuracy = \sum_{trajectories} \frac{1-\Phi(\mathbf{PoI_G}, \mathbf{PoI_D})}{max(\#\{\mathbf{PoI_G}\}, \#\{\mathbf{PoI_D}\})}. \tag{3}$$

The division by the maximum value between the length of the real itinerary and the detected one aims to penalize strings with different lenghts, e.g., in the case of an excess of extracted *Stop Points*.

C. Experimental Results

The first experiment aims to compare the performance of the three *Stop Points* extraction methods described in Section V-A.

In order to evaluate such methods we adopt the same simple approach to associate a PoI to the inferred *Stop Point*. In particular, we adopt the nearest neighbor method, which selects the nearest PoI to the *Stop Point*'s centroid. The best performance is obtained by the CAPD algorithm, which exhibits an accuracy of 0.82%, thus increasing the accuracy of method described in [22] by 25% and of method described in [23] by 4%. Such results are summarized in table I.

The second experiment aims to compare the performance of the blind nearest neighbor assignment method with the probabilistic approach adopted by the CAPD algorithm. The main goal of such experiment is the evaluation of relevance of different contextual information. The training and validation of the DBN adopted by the CAPD algorithm have been performed by a 6-fold cross validation on the whole data set. Table II shows results of such comparison, and proves that the probabilistic inference, by means of different set of context

TABLE II
COMPARISON OF ACCURACY, UNCERTAINTY AND EXECUTION TIME FOR CONTEXT-AWARE POI DETECTION (CAPD) WITH DIFFERENT SUBSETS OF FEATURES

AT	ST	DF	PVP	DW	Accuracy	Uncertainty	Time
-	-	-	-	-	0.818	0	0.537×
✓	✗	✗	✗	✗	0.871	0.113	0.819×
✗	✓	✗	✗	✗	0.891	0.060	0.818×
✗	✗	✓	✗	✗	0.864	0.146	0.880×
✗	✗	✗	✓	✗	0.865	0.123	0.846×
✗	✗	✗	✗	✓	0.891	0.123	0.815×
✓	✓	✗	✗	✗	0.893	0.061	0.837×
✓	✗	✓	✗	✗	0.873	0.158	0.893×
✓	✗	✗	✓	✗	0.879	0.123	0.872×
✓	✗	✗	✗	✓	0.872	0.129	0.847×
✗	✓	✓	✗	✗	0.889	0.069	0.892×
✗	✓	✗	✓	✗	0.895	0.060	0.864×
✗	✓	✗	✗	✓	0.900	0.047	0.832×
✗	✗	✓	✓	✗	0.856	0.162	0.916×
✗	✗	✓	✗	✓	0.854	0.168	0.889×
✗	✗	✗	✓	✓	0.852	0.133	0.872×
✓	✓	✓	✗	✗	0.895	0.072	0.911×
✓	✓	✗	✓	✗	0.900	0.062	0.897×
✓	✓	✗	✗	✓	0.901	0.068	0.868×
✓	✗	✓	✓	✗	0.801	0.173	0.940×
✓	✗	✓	✗	✓	0.773	0.197	0.920×
✓	✗	✗	✓	✓	0.803	0.151	0.906×
✗	✓	✓	✓	✗	0.893	0.068	0.931×
✗	✓	✓	✗	✓	0.899	0.065	0.905×
✗	✓	✗	✓	✓	0.902	0.058	0.896×
✗	✗	✓	✓	✓	0.713	0.187	0.940×
✓	✓	✓	✓	✗	0.905	0.077	0.963×
✓	✓	✓	✗	✓	0.901	0.080	0.942×
✓	✓	✗	✓	✓	0.903	0.068	0.933×
✓	✗	✓	✓	✓	0.696	0.160	0.981×
✗	✓	✓	✓	✓	0.898	0.082	0.965×
✓	✓	✓	✓	✓	0.883	0.088	1.000×

features, can be useful to identify visited PoIs. The features considered for this experiments are: the Arrival Time (AT) of the user at the stop point, the Stop Time duration (ST), the Daily Frequency (DF) of visits of a particular semantic class, the sequence of Past Visited Places (PVP), and the Day of the Week (DW). In order to compute DF and NG features, we have to make an approximation of the real itinerary by considering the most probable waypoint as the actually visited one.

As results show, a good trade-off is achieved by considering the day of the week (DW) and the duration of the stop (ST), obtaining a 8% accuracy increase with respect to the blind approach, and achieving also the minimum value for uncertainty. Such results suggest that, even with a simplified model of user behavior (a first-order Markov model against the fifth-order Markov chain of the simulated model), few context information allow to obtain a great increase in system accuracy.

VII. Conclusions and Future Work

This paper described a simulation tool which aims to generate mobility traces of users in Smart Environments, and in particular in Smart Campuses. The models adopted by the proposed simulator reflect the regular and intentional nature of human mobility. Output datasets can be used as testbed for the evaluation of location-aware recommendation systems for a Smart Campus.

The suitability of the proposed simulation tool have been proved by using the obtained dataset to compare some algorithms presented in the literature with a novel context-aware PoI detection algorithm.

As future work, we will address the simulation of other pervasive sensors which can be used to refine the detection of activities performed by users [31], [32].

Moreover, we plan to engage users in labeling PoIs' classes, through a participatory sensing approach that relies on techniques of reputation management [33] to discard information provided by malicious users who intentionally send incorrect data to create a disservice.

Acknowledgement

This research is included in the Project VASARI funded by Italian MIUR (PNR 2015-2020, DD MIUR n. 2511), CUP F26C18000220005.

References

[1] Y.-A. De Montjoye, C. A. Hidalgo, M. Verleysen, and V. D. Blondel. Unique in the crowd: The privacy bounds of human mobility. *Scientific reports*, 3:1376, 2013.

[2] G. Solmaz and D. Turgut. Pedestrian mobility in theme park disasters. *IEEE Communications Magazine*, 53(7):172–177, 2015.

[3] V. Vukadinovic, F. Dreier, and S. Mangold. A simple framework to simulate the mobility and activity of theme park visitors. In *Proc. of the 2011 Winter Simulation Conference (WSC)*, pages 3248–3260, 2011.

[4] D. Bhattacharjee, A. Rao, Shah C, M. Shah, and A. Helmy. Empirical modeling of campus-wide pedestrian mobility observations on the usc campus. In *Proc. of the IEEE 60th Vehicular Technology Conference (VTC2004-Fall 2004*, volume 4, pages 2887–2891 Vol. 4, 2004.

[5] M. Collard, P. Collard, and E. Stattner. Simulating human mobility and information diffusion. In *Proc. of the 2013 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM 2013)*, pages 197–201, 2013.

[6] F. Concone, P. Ferraro, and G. Lo Re. Towards a smart campus through participatory sensing. In *Proc. of the 2018 IEEE Int. Conf. on Smart Computing (SMARTCOMP)*, pages 393–398, 2018.

[7] V. Agate, F. Concone, and P. Ferraro. Wip: Smart services for an augmented campus. In *Proc. of the 2018 IEEE Int. Conf. on Smart Computing (SMARTCOMP)*, pages 276–278, 2018.

[8] M. C. Gonzalez, C. Hidalgo, and A.-L. Barabasi. Understanding individual human mobility patterns. *Nature*, 453:779–82, 2008.

[9] C. Song, Z. Qu, N. Blumm, and A.-L. Barabási. Limits of predictability in human mobility. *Science*, 327(5968):1018–1021, 2010.

[10] J. L. Berrou, J. Beecham, P. Quaglia, M. A. Kagarlis, and A. Gerodimos. Calibration and validation of the legion simulation model using empirical data. In *Proc. of Pedestrian and Evacuation Dynamics 2005*, pages 167–181, 2005.

[11] C. Gloor. Pedsim: Pedestrian crowd simulation,. *http://pedsim.silmaril.org*, 2016.

[12] I. Rhee, M. Shin, S. Hong, K. Lee, S. J. Kim, and S. Chong. On the levy-walk nature of human mobility. *IEEE/ACM Transactions on Networking*, 19(3):630–643, 2011.

[13] W.-J. Hsu, K. Merchant, H.-W. Shu, C.-H. Hsu, and A. Helmy. Weighted waypoint mobility model and its impact on ad hoc networks. *ACM SIGMOBILE Mobile Computing and Communications Review*, 9:59–63, 01 2005.

[14] F. Giurlanda, P. Perazzo, and G. Dini. Humsim: A privacy-oriented human mobility simulator. In *Int. Conf. on Sensor Systems and Software*, pages 61–70, 2014.

[15] L. Pappalardo, F. Simini, S. Rinzivillo, D. Pedreschi, F. Giannotti, and A.-L. Barabási. Returners and explorers dichotomy in human mobility. *Nature communications*, 6:8166, 2015.

[16] A. De Paola, A. Giammanco, G. Lo Re, and G. Anastasi. Detection of points of interest in a smart campus. *Proc. of the 5th International Forum on Research and Technologies for Society and Industry (RTSI)*, pages 1–6, 2019.

[17] A. De Paola, P. Ferraro, S. Gaglio, G. Lo Re, and S. K. Das. An adaptive bayesian system for context-aware data fusion in smart environments. *IEEE Trans. on Mobile Computing*, 16(6):1502–1515, 2017.

[18] K. Lee, S. Hong, S. J. Kim, I. Rhee, and S. Chong. Slaw: self-similar least-action human walk. *IEEE/ACM Transactions on Networking (TON)*, 20(2):515–529, 2012.

[19] K. Lee, S. Hong, S. Kim, I. Rhee, and S. Chong. Slaw: A mobility model for human walks. In *IEEE INFOCOM 2009*, pages 855–863, 2009.

[20] S. Gaglio and G. Lo Re. *Advances onto the Internet of Things*, volume 260. Springer, 2014.

[21] K. Nishida, H. Toda, T. Kurashima, and Y. Suhara. Probabilistic identification of visited point-of-interest for personalized automatic check-in. In *Proc. of the 2014 ACM Int. Joint Conf. on Pervasive and Ubiquitous Computing (UbiComp 2014)*, pages 631–642, 2014.

[22] R. Montoliu, J. Blom, and D. Gatica-Perez. Discovering places of interest in everyday life from smartphone data. *Multimedia Tools and Applications*, 62(1):179–207, 2013.

[23] Z. Fu, Z. Tian, Y. Xu, and C. Qiao. A Two-Step Clustering Approach to Extract Locations from Individual GPS Trajectory Data. *ISPRS Int. Journal of Geo-Information*, 5:166, 2016.

[24] D. Lian and X. Xie. Learning location naming from user check-in histories. In *Proc. of the 19th ACM SIGSPATIAL Int. Conf. on Advances in Geographic Information Systems*, pages 112–121, 2011.

[25] B. Shaw, J. Shea, S. Sinha, and A. Hogue. Learning to rank for spatiotemporal search. In *Proc. of the 6th ACM Int. Conf. on Web Search and Data Mining (WSDM 2013)*, pages 717–726, 2013.

[26] P. A. Zandbergen. Accuracy of iphone locations: A comparison of assisted gps, wifi and cellular positioning. *Trans. in GIS*, 13(s1):5–25, 2009.

[27] X. Cao, G. Cong, and C. S. Jensen. Mining Significant Semantic Locations from GPS Data. *Proc. of the VLDB Endowment*, 3(1-2):1009–1020, 2010.

[28] D. Koller and N. Friedman. *Probabilistic graphical models: principles and techniques*. MIT press, 2009.

[29] A. De Paola, P. Ferraro, S. Gaglio, and G. Lo Re. Context-awareness for multi-sensor data fusion in smart environments. In *Proc. of the Conf. of the Italian Association for Artificial Intelligence*, pages 377–391, 2016.

[30] G. V. Bard. Spelling-error tolerant, order-independent pass-phrases via the damerau-levenshtein string-edit distance metric. In *Proc. of the fifth Australasian symposium on ACSW frontiers-Volume 68*, pages 117–124, 2007.

[31] F. Concone, G. Lo Re, and M. Morana. A fog-based application for human activity recognition using personal smart devices. *ACM Trans. on Internet Technology (TOIT)*, 19(2):20, 2019.

[32] F. Concone, S. Gaglio, G. Lo Re, and M. Morana. Smartphone data analysis for human activity recognition. In *Proc. of the Conf. of the Italian Association for Artificial Intelligence*, pages 58–71, 2017.

[33] V. Agate, A. De Paola, G. Lo Re, and M. Morana. A simulation framework for evaluating distributed reputation management systems. In *Proc. of the 13th International Conference on Distributed Computing and Artificial Intelligence*, pages 247–254. Springer, 2016.