SmartSPEC: customizable smart space datasets via event-driven simulations

Andrew Chio, Daokun Jiang, Peeyush Gupta, Georgios Bouloukakis, Roberto Yus, Sharad Mehrotra, Nalini Venkatasubramanian

SmartSPEC: Customizable Smart Space Datasets via Event-driven Simulations

Andrew Chio*, Daokun Jiang*, Peeyush Gupta*, Georgios Bouloukakis**,
Roberto Yus†, Sharad Mehrotra*, Nalini Venkatasubramanian*
*Dept. of Computer Science, University of California, Irvine, {achio,daokunj,peeyushg,sharad,nalini}@uci.edu
**Dept. of Computer Science, Télécom SudParis, IP Paris, georgios.bouloukakis@telecom-sudparis.eu
†Dept. of Computer Science and Electrical Engineering, University of Maryland, Baltimore County, ryus@umbc.edu

***Abstract*—This paper presents SmartSPEC, an approach to generate customizable smart space datasets using sensorized spaces in which people and events are embedded. Smart space datasets are critical to design, deploy and evaluate robust systems and applications to ensure cost-effective operation and safety/-comfort/convenience of the space occupants. Often, real-world data is difficult to obtain due to the lack of fine-grained sensing; privacy/security concerns prevent the release and sharing of individual and spatial data. SmartSPEC is a smart space simulator and data generator that can create a digital representation (twin) of a smart space and its activities. SmartSPEC uses a semantic model and ML-based approaches to characterize and learn attributes in a sensorized space, and applies an event-driven simulation strategy to generate realistic simulated data about the space (events, trajectories, sensor datasets, etc). To evaluate the realism of the data generated by SmartSPEC, we develop a structured methodology and metrics to assess various aspects of smart space datasets, including trajectories of people and occupancy of spaces. Our experimental study looks at two real-world settings/datasets: an instrumented smart campus building and a city-wide GPS dataset. Our results show that the trajectories produced by SmartSPEC are $1.4x$ to $4.4x$ more realistic than the best synthetic data baseline when compared to real-world data, depending on the scenario and configuration.**

***Index Terms*—smart space, sensor, simulation, trajectory**

I. Introduction

With the emergence of IoT and support technologies (e.g., Edge networks, storage), community infrastructure - including buildings and homes - are becoming more interconnected, fueling the creation of smart spaces. We characterize a *smart space ecosystem* using four basic entities: (i) *spaces*, which constitute the geographical layout of the smart space; (ii) *people*, who represent the inhabitants of spaces; (iii) semantic *events*, which describe the actions of people in spaces; and (iv) *sensors*, which observe various phenomena in the space. These entities are highly interrelated: *sensors* observe *people* attending *events* in *spaces*. In an example university scenario, professors and students move to offices and lecture halls to attend semantic events such as meetings, lectures and seminars. Their movements are captured by various sensors, such as WiFi Access Points (APs), Bluetooth beacons, etc.

Smart space ecosystems enable multiple benefits such as optimization of space/energy usage in organizations, e.g., through personalized heating, ventilation and air conditioning (HVAC) systems. Ongoing work in the research community also aims to address system-level challenges such as facilitating the development of sensor-based smart applications [1], sensor planning and operator placement to reduce network latency [2], [3], middleware platforms to manage data flows for timely, reliable data exchange [4], support for privacy-aware collection and storage of user data [5], [6], Edge-to-Cloud computation for predictive analytics [7], and formal methods for dependable execution of pervasive applications [8], [9].

Such smart space approaches, among others, must be tested and validated using "smart space datasets" - obtaining such datasets is nontrivial. One challenge is the cost and complexity of data collection, including purchasing and deploying sensors, recruiting participants and reliably gathering and storing data. Secondly, such data must be preprocessed to preserve participant privacy and adhere to data privacy regulations. Furthermore, creating and exploring hypothetical situations such as fires, floods, earthquakes, etc., and their impact on the smart space ecosystem at scale is challenging. This may require adapting an existing dataset to model dynamic change (e.g., building damage in a fire can result in inaccessible exits). This can also be useful in performing scalability studies (e.g., to evaluate the impact of activities with many participants.) Recent events have also emphasized the importance of studying interactions among users and organizational policies in spaces (e.g., exposure tracing and social distancing during COVID).

Researchers often rely upon offline modeling and simulation techniques to explore complex scenarios (e.g., human activity models, fire spread simulators, etc.). When working with general-purpose simulation toolkits, it is often challenging to ensure that results are realistic for the desired space and setting. Variability in the movements of people over time and the dynamic nature of events add to the complexity of capturing realism. Proper evaluation of smart space datasets must determine the degree of realism i.e., how closely do synthetic datasets model their real-world counterparts. However, such evaluation is not straightforward. Recent literature has illustrated the utilization of digital twins to assist in realistically modeling specific spaces and activities [10], [11].

In this paper, we present SmartSPEC, an event-driven approach centered around people to generate customizable smart space datasets using a flexible model of a smart space ecosystem incorporating spaces, people, events, and sensors. To facilitate the definition of models, we use input data from sensors (e.g., WiFi/Bluetooth), and apply ML-based ap-

proaches to extract higher-level metamodels that characterize the components of a smart space. The metamodels in turn are used to generate synthetic datasets. We further develop a structured methodology to assess the realism of the produced synthetic datasets. Our experimental results show the realism of datasets generated by SmartSPEC in real-world settings at the building level/city scale. Our key contributions are:

- A framework to generate multiple realistic, customizable smart space datasets using semantics of a provided set of spaces, people, events, and sensors.
- A learning mechanism that extracts models of people, events, and semantic trajectories from input seed data.
- A methodology to assess realism of generated smart space datasets using well-established similarity measures and clustering techniques (e.g., for trajectories, occupancy).
- Validation of the SmartSPEC system and concept by leveraging multiple real-world instrumented smart spaces and hypothetical settings.

We organize the rest of the paper as follows: §II discusses related works on smart space modeling while §III presents our SmartSPEC architecture. §IV and §V covers the process to learn SmartSPEC models and generate synthetic data. §VI and §VII discuss our methodology to assess realism and our experimental validation. §VIII discusses future directions.

II. Related Work

We describe related literature in datasets for indoor/outdoor spaces and generative models for synthetic data generation.

Generating Indoor Space Datasets. Several tools have been developed to enable applications for indoor smart spaces. Diasim [12] provides an event-driven framework to run sensor-based applications in actual/simulated environments with support for monitoring, visualizing and debugging deployed applications. Here, the logic for interactions between simulated entities must be manually encoded, requiring significant effort. IE Sim [13] focuses on a GUI-based interface to model interactions with simulated environments, including *Activities of Daily Living* (ADL) for normal and abnormal activities (e.g. hazard scenarios). Persim-3D [14] generates datasets for complex activity scenarios where end-users can define space structures, interior elements (e.g., appliances), and sensors. OpenSHS [15] is an open-source 3D smart home simulator for data generation that provides methods for generating large representative smart home ADL datasets. These efforts focus on low-level human activities (e.g., eating, standing up) in the scope of small spaces (e.g., a single room), and are generally inadequate to describe movements of people over large spaces, such as in multistory buildings, which is the focus of our work.

Generating Outdoor Environment Datasets. Techniques to model trajectories of people in outdoor spaces has traditionally used coarse mobility models such as Lévy flights [16]–[20] and Preferential Attachment processes [21]–[23]. Such models help characterize the movement of people statistically; they are not designed to capture realistic smart space trajectories. Early work by Brinkoff [24] focuses on network-based trajectory generation in the context of road/railway networks and relies upon A* search to generate subsequent trajectories. MWGen [25] expanded on this to generate trajectories utilizing a database system to simulate human movement over large distances via multiple modes (e.g., bus, walk). Trajectory generation in road networks [26] has utilized semantic labels and user input to manually define mobility models consisting of prescribed stops and mode of transportation. However, this greatly increases the amount of effort required to configure a simulation. In comparison, SmartSPEC considers more complex semantic relationships such as people-event interactions. Markov models have also been used to simulate trajectories using mobile devices [27]; this approach requires call detail records and GPS data to recreate trajectories following human travel patterns such as the return probability (i.e., probability of returning to a previously visited area (i.e., return probability) and the distance traveled from a starting "home" point (i.e., radius of gyration). While such heuristics can help characterize general human mobility, they do not provide mechanisms to specify the mobility for individuals, which is our focus.

Generative Models. A notable emerging direction in modeling human trajectories leverages Generative Adversarial Networks (GANs) to produce synthetic mobility traces [28]–[31]. These methods typically train two competing neural network models: the *generator* model creates new synthetic examples which the *discriminator* must classify as real or fake. Gupta [31] integrates multiple Long Short-Term Memory (LSTM) neural networks with a novel "pooling" layer in a GAN architecture augmented with social contexts to produce small-scale trajectories. This is augmented by Rossi [28] who proposed new evaluation metrics to optimize a GAN architecture utilizing LSTM to predict an entire distribution of potential human trajectories. Similar techniques are utilized for outdoor environments: Ouyang [30] leverages a location-major trajectory representation with a non-parametric GAN consisting of multiple convolution and filter layers. Through this process, local mobility patterns are extracted, and then reapplied to generate appropriate trajectories. Kulkarni [29] takes a different approach and instead advocates for the usage of non-parametric copulas generative models, which relies on learning an implicit dependence structure using marginal densities and a copula density. While these methods are new and promising in generating realistic human trajectories, it is difficult to ensure that both space semantics alongside people-event interaction semantics are learned and enforced. This is exacerbated by the notorious difficulty to train GANs.

In contrast to the above methods and approaches, SmartSPEC focuses on generating realistic smart space datasets by creating realistic event-based semantic trajectories. Our approach also learns models governing the generation directly from seed data, thus reducing the human effort required.

III. The SmartSPEC Approach

The high-level SmartSPEC architecture consists of two main components (see Fig. 1): (i) *Scenario Learning* uses input seed data and a priori knowledge of the underlying space

and sensors to learn higher-order SmartSPEC concepts; and (ii) *Scenario Generation* takes SmartSPEC data to generate a synthetic dataset from which a smart space dataset (e.g., trajectory dataset, sensor observation dataset, etc.) can be derived. We use the data models below to define a scenario and its variations (i.e., by defining a new set of people and events), which drives the generation of new observable phenomena in the smart space. The SmartSPEC toolkit and a detailed manual with instructions on operating the system is on GitHub [32].

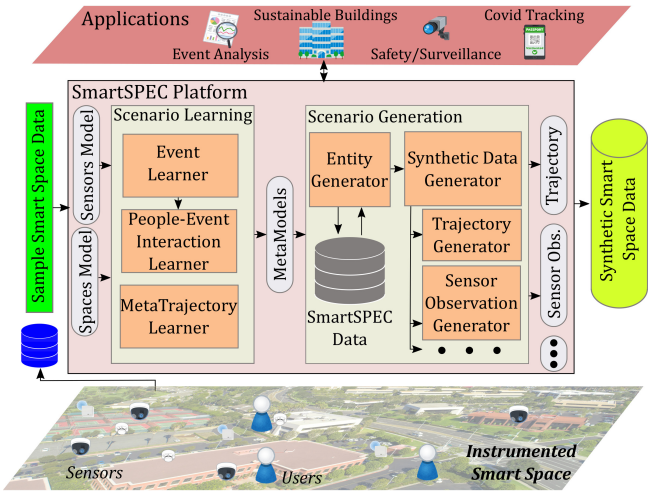


Fig. 1: SmartSPEC approach.

The first component, *Scenario Learning* (see §IV) takes input sensor observation data alongside a priori knowledge of the underlying smart space and its deployed sensors to accomplish three goals: (i) extract semantic events/people that explain the observed phenomena in the smart space; (ii) learn *metamodels* of events/people to enable the generation of new scenarios; and (iii) learn higher-order *metatrajectories*, which is a data-driven approach to generate the movement of people between spaces. These outputs are given to the *Scenario Generation* component (see §V), which will proceed in three steps. First, the *Entity Generator* will use the events/people metamodels to create a new set of events/people. This is passed to the *Synthetic Data Generator* alongside a configuration file to generate log files for data generators such as the *Trajectory Generator* and *Sensor Observation Generator*, among others, to produce smart space datasets. This synthetic data can be given to applications utilizing smart space datasets.

Below, we describe the attributes of each model utilized by SmartSPEC. In general, we define a smart space ecosystem as a 4-tuple of sets $(\mathcal{C}, \mathcal{P}, \mathcal{E}, \mathcal{S})$, which represent a set of spaces, people, semantic events, and sensors, respectively. SmartSPEC enables the generation of new scenarios through the definition of metadata models for events and people, which we describe as a set of *MetaEvents* $\mathcal{ME}$ and *MetaPeople* $\mathcal{MP}$. SmartSPEC requires that models for spaces and deployed sensors are defined a priori, as they are not learned nor generated for new scenarios. We leave such generation as future work; one example space generation strategy is presented in [33].

Space Model. Let $C = (crd, adj, cap)$ denote a logical space in SmartSPEC, where coordinates crd and neighboring spaces adj are used to encode a reachability graph, and cap represents the maximum capacity of the space.

Person Model. Let person P be defined by the 2-tuple (TP, aff), where TP denotes a time profile (defined later), which details the expected time that a person enters/exits the simulated space and aff is a probability model for the person's affinity to different types of events. For example, a specific grad student may attend evening classes from 4-7pm (represented by time profile TP) and has a high affinity towards study events (represented by event affinity aff).

Event Model. Let a semantic event E be defined by the 3-tuple (TP, C_e, att), where TP denotes a *time profile* (defined later), which details when the event is held, and C_e defines the event's hosting space. att expresses attendance requirements as a map of the types of people to the event. For the university campus scenario, one example of a semantic event is a specific lecture held in space 100 (represented by C_e), occurring daily between 10-11am (represented by TP and requires one faculty member and 30 students (represented by att).

Sensor Model. We model a sensor S focusing on the characteristics that SmartSPEC requires to generate a smart space dataset. Thus, S is defined by the 3-tuple (mob, cov, int), where mob denotes the mobility pattern of the sensor (i.e., static or mobile) and cov is either a list of covered spaces if the sensor is static (e.g., a thermometer), or a person if the sensor is mobile (e.g., accelerometer in a phone). Sensors produce observations in periodic intervals int.

Time Profile. Time profiles describe a pattern of when events occur/when people enter or exit the simulated smart space. We represent a time profile TP as a list of 6-tuples $(d_s, d_e, per, t_s, t_e, t_{exp})$ which represents an "active" time period for an "active" day; its semantics depend on the object it describes. In a time profile, (d_s, d_e) denotes a date range where an active day occurs with periodicity per. Within an active day, the active time is specified by (t_s, t_e), which expresses a "mean" time with "std" minutes for the starting/ending time. We use t_{exp} to express an expected duration of time whose semantics also depend on the modeled object: for people, it denotes the amount of time they must spend in the space while for events, its denotes the expected time commitment necessary to attend the event. An example time profile for a Monday seminar event during a semester can be expressed as follows: (2022-01-03,2022-06-06,M,11:00,13:00,(110,15)). Note that while the event spans 2 hours, an individual is only expected to attend $\mathcal{N}(110, 15)$ mins.

Using SmartSPEC. The three modes of operation of SmartSPEC to generate a smart space dataset are represented in Fig. 2. These modes vary in the level of user involvement/automation and control with respect to the generation of models and metamodels for both people and events. First, the user defines the relevant Space and Sensor entities according to the models above ❶. Then, relevant metamodels for people and events must be defined: this can be done manually ❷a or automatically (with the Scenario Learning component of SmartSPEC) ❷b. Users may modify the automatically gen-

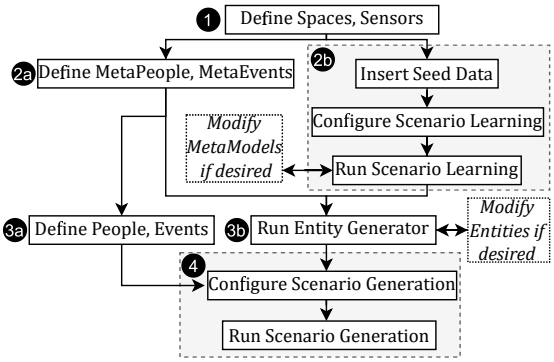


Fig. 2: SmartSPEC workflow.

erated metamodels if desired. After metamodels are defined, the user can either manually define a corresponding set of people and events 3a, or use the entity generator in 3b to do that automatically. Once again, the produced entities can be modified if desired. Finally, the the user configures the Scenario Generation component and executes it 4.

IV. Scenario Learning in SmartSPEC

To learn semantic events and people, SmartSPEC uses sensor observation data and a priori knowledge of a smart space and its deployed sensors. We accomplished this by utilizing change point detection algorithms [34] to extract events, and then characterizing people based on the log of events they attend. We will learn metamodels for events/people (which we refer to as *MetaEvents/MetaPeople*) using agglomerative clustering methods [35]. Lastly, we use time-series methods to derive *MetaTrajectories* to help generate the synthetic data.

We define a *connectivity log* dataset D to consist of a series of triples (dt, P, C), denoting that person P was in space C at datetime dt. D can readily be derived from any wireless connectivity technology such as WiFi, Bluetooth beacons, and cellular [36]. Such technologies are pervasive and available in many smart spaces: buildings usually have WiFi Access Points (APs) deployed to provide Internet connectivity. Note that, to obtain a more realistic output from SmartSPEC, D should be extracted from a scenario for which we want to generate the synthetic dataset. While one could potentially utilize a connectivity dataset obtained from one scenario to a new scenario, bias and spurious correlations must be removed from SmartSPEC data (e.g., using a technique like in [37]), although that is out of the scope of this paper.

Learning Events/MetaEvents from Smart Space Data. Given a connectivity log D, we identify individual events in each space and group them into higher-order *MetaEvents*. For example, in a university campus scenario, the set of seminars can be extract first, then grouped into a seminar metaevent from which additional instances of seminars can be generated.

MetaEvent Model. Formally, a MetaEvent ME is a 4-tuple $(TP, \mathcal{C}_{me}, att, \mathcal{E}_{me})$, where TP and $\mathcal{C}_{me}$ are lists of time profiles and spaces from which events can be generated, respectively. Each metaevent uses att to describe a mapping of *metapeople* (defined later) to normal distribution parameters (i.e., mean, stdev) to determine a possible attendance set. The set $\mathcal{E}_{me}$ stores events that are characterized by ME.

Algorithm 1: Extracting Events, Learning MetaEvents.

Input: Dataset D, Spaces $\mathcal{C}$, Date $start$, Date end, int b
Output: Events $\mathcal{E}$, MetaEvents $\mathcal{ME}$

$\mathcal{E} \leftarrow \emptyset$
for $d \leftarrow start \ldots end$ **do**
  for $c \leftarrow \mathcal{C}$ **do**
    $data \leftarrow D.query(space = c, day = d)$
    $ts \leftarrow computeOccupancy(data, minutes = b)$
    $bkpts \leftarrow changePointDetection(ts)$
    $\mathcal{E} \leftarrow \mathcal{E} \cup createEvents(c, bkpts)$
$distMat \leftarrow computeDistanceMatrix(\mathcal{E})$
$clusters \leftarrow doAgglomerativeClustering(distMat)$
$\mathcal{ME} \leftarrow makeMetaEvents(clusters)$
return $\mathcal{E}$, $\mathcal{ME}$

The process of learning events is detailed in Alg. 1 and relies heavily on change point detection [38] to determine when one event ends and the next starts. For each day d and space C in D, we compute a time-series of occupancy counts in C in blocks of b minutes (e.g., for $b = 10$, we count the occupancy from 8:00-8:10, 8:10-8:20, etc.). Then, we apply a change point detection algorithm on the occupancy time-series to produce breakpoints denoting when events start and end. In the implementation of our evaluation metric, we use a change point detection Python package called *ruptures* [39].

We use agglomerative clustering with single linkage to cluster events into metaevent models; this unsupervised ML model initializes all events in its own group, then merges groups in order of the closest distance. We define the distance between two evens as the weighted sum of: the difference between respective start/end times and the *Jaccard Index* metric between attendees. We note that distance between semantics of spaces is excluded because it is too context-specific. With a measure of event distance defined, we group events until a user-defined threshold is met.

Learning People-Event Interactions, MetaPeople. The key characteristic defining the behavior of a person is the set of events they attend. Thus, with the set of events previously learned, SmartSPEC characterizes each person with their attended events and then groups them into *MetaPerson* models.

MetaPerson Model. Let a metaperson MP be a 3-tuple $(TP, aff, \mathcal{P}_{mp})$, where TP is a list of time profiles and aff describes the event affinity, which details a probability model for attendance. We store the set of associated people in $\mathcal{P}_{mp}$.

We use the set of attended events to infer both the event affinity aff and time profile TP for a person P. Namely, P will be more likely to attend an event they have previously attended, and TP defines its active times, assuming that P will try not to miss their events. MetaPeople are generated analogously to MetaEvents: agglomerative clustering with single linkage is utilized; the distance between people is defined to be the weighted sum of the difference between time profiles TP and the *Jaccard Index* between attended metaevents.

Learning MetaTrajectories. SmartSPEC uses *metatrajectories* to describe the paths that a person can take to move between two (nonadjacent) spaces. First, we create estimates

for the start/end times that a person is in a space given timestamps from D: when a sensor consecutively observes a person, SmartSPEC assumes that there is a high likelihood that they are in the same space. Let $valid$ denote the amount of time between consecutive entries to be considered as the staying in the same space; this allows some consecutive sensor observations to be merged to yield start/end times. Then, we label spaces in which the duration of time spent inside was greater than some user-defined threshold. This partitions a person's trajectory into sub-trajectories, each of which start and end at a labeled space. A path between two spaces is selected from one of the previously learned trajectories.

V. Scenario and Data Generation in SmartSPEC

The *Scenario Generation* component consists of three steps: (i) *Entity Generation* uses previously learned metaevent/metaperson models to generate a new set of events/people, which is given to the (ii) *Synthetic Data Generator* which combines them with space/sensor entities, producing a log dataset from which (iii) *data generators* such as the *Trajectory Generator* and the *Sensor Observation Generator* can produce trajectory and sensor observation smart space datasets, respectively.

Entity Generation. As described in the SmartSPEC models, and event is characterized by the 3-tuple (TP, C_e, att); we derive each of these attributes from a metaevent, which is characterized by $(TP, \mathcal{C}_{me}, att, \mathcal{E}_{me})$. Alg. 2 describes the event generation process. First, a metaevent ME is selected from which an event is generated with probability proportional to the size of the $\mathcal{E}_{me}$. The event time profile TP and space C_e are chosen at random from the list of time profiles TP and spaces $\mathcal{C}_{me}$, respectively. However, if C_e cannot host an event at TP, then another space is selected; if none exist, then the current event is discarded and the process to generating a new event restarts. The event attendees att is obtained by sampling the normal distribution parameters for each metaperson in att. The process to generate a new event is overloaded to generate a new person $P = (TP, aff)$: a metaperson is chosen from which both TP and aff are determined.

Synthetic Data Generation. SmartSPEC generates synthetic datasets by simulating people who attend events during their active times in the simulated space, as seen in Alg. 3.

For each person $P \in \mathcal{P}$ and for each active date and time $d, t_s, t_e \in d_s \dots d_e$, obtained from P's associated time profile (Lines 2,3), P will (i) choose an event to attend using the input semantic models (Lines 6-19); and (ii) move to attend the event, recording their movements (Lines 20-26). In the event selection process, previous commitments to (periodic) events attended periodic events will be considered first (Line 6) before new events. However, if P must select a new event, they consider both physical constraints such as the space capacity of the event's hosting space and the expected travel/wait time, as well as semantic constraints imposed by attendee capacities and their prior commitments (i.e., from previously attended events) (Lines 12). The specific paths and timestamps yielding a person's simulated movement are detailed by a metatrajectory and are recorded (Lines 24,25).

Algorithm 2: CreateEvent

Input: MetaEvents $\mathcal{ME}$, Events $\mathcal{E}$
Output: Event E

$ME \leftarrow selectMetaEvent(\mathcal{ME})$
$TP \leftarrow selectTimeProfile(TP)$
$C_e \leftarrow selectNonConflictingSpace(\mathcal{C}_{me}, \mathcal{E})$
if $C_e =$ **null then**
  return $GenerateEvent(\mathcal{ME} - \{ME\})$
$att \leftarrow map()$
for $a, (\mu, \sigma) \leftarrow att$ **do**
  $att[a] \leftarrow \mathcal{N}(\mu, \sigma).sample(1)$
return $E(TP, C_e, att, ME)$

Algorithm 3: Synthetic data generation.

Input: Date d_s, Date d_e, People $\mathcal{P}$, Events $\mathcal{E}$, Spaces $\mathcal{C}$
Output: LogFile log

1 $log \leftarrow \emptyset$
2 **for** $P \leftarrow \mathcal{P}$ **do**
3 **for** $d, t_s, t_e \leftarrow P.queryActiveDateTime(d_s \dots d_e)$ **do**
4 $t \leftarrow t_s$
5 **while** $t \leq t_e$ **do**
6 $E \leftarrow P.findPreviousEvent(d, t)$
7 **if** ! E *is* **null then**
8 $path \leftarrow getPath(P.space, E.space)$
9 **else**
10 $attd \leftarrow \emptyset$
11 **for** $E \leftarrow \mathcal{E}$ **do**
12 **if** $!E.hasSpaceCapacity(t)$ **or** $!E.hasPeopleCapacity(P)$ **or** $E.conflictsWith(P.prevEvents)$ **then**
13 **continue**
14 $P_e \leftarrow getPath(P.space, E.space)$
15 $arrival \leftarrow t + P_e.estTravelTime()$
16 **if** $|arrival - E.startTime| \geq \epsilon$ **then**
17 **continue**
18 $attd \leftarrow attd \cup \{(\mathcal{E}, P_e)\}$
19 $E, path \leftarrow select(attd, P.eventAffinity)$
20 **for** $c \leftarrow path$ **do**
21 Block until $C_e.cap(d, t) \leq C_e.maxCap$
22 Move P to c, updating t
23 $log.record(P, c, t)$
24 $log.record(P, E.space, E.t_e)$
25 $P.recordAttendance(E)$
26 $t \leftarrow E.t_e$
27 **return** log

Trajectory and Sensor Observation Generation. We define a semantic trajectory as a series of 5-tuples (P, E, C, t_s, t_e), which denotes the space C and time t from which the whereabouts of a person P can be mapped as they move to attend some semantic event E. We extract a semantic trajectory from a log by enumerating over the timestamped sequence of visited spaces and events; note that the log produced from Alg. 3 records such information. Similarly, to generate a sensor observation dataset, we use the synthetic data logs and the sensor model to determine which movements are captured. In Alg. 4 we define, for a log entry (P, E, C, t), $cover$ as the subset of sensors covering the space C at

Algorithm 4: Sensor observation generation.

Input: Sensors S, Trajectories $traj$
Output: Observations obs

$obs \leftarrow \emptyset$
for $t \leftarrow traj$ **do**
  $cover \leftarrow S.coverage(t.space, t.timestamp)$
  $sensors \leftarrow chooseSensors(cover, t)$
  for $s \leftarrow sensors$ **do**
    $ts \leftarrow chooseObsTime(s, t)$
    $obs.record(t.person, ts, s)$
return obs

time t. A user-defined filter is applied to $cover$ to address application-specific intermittent behavior. For each sensor s, an observation time is recorded into an observation dataset. Note that generating sensor observations is dependent upon the user applications and sensor, and thus needs to be customized.

VI. Assessing Realism of Smart Space Datasets

In this section, we will describe the methodology used to assess the degree of realism between two smart space datasets, as such a metric is not yet well-defined to the best of our knowledge. Consider two smart space datasets D and D', such that D is a log of real-world phenomena, while D' is a synthetically generated log. Each dataset consists of a series of 3-tuples (P, C, dt) denoting the space $C \in \mathcal{C}$ that a person $P \in \mathcal{P}$ is in at datetime dt. We assume that both datasets take place in the same set of spaces $\mathcal{C}$.

A. Space Occupancy Similarity

We use occupancy of spaces as one of the mechanisms to assess the similarity of D and D' wrt. the behavior of the recorded people. Formally, we define the occupancy λ_D^{C,t_s,t_e} as the number of unique people in dataset D who are in the space C during the time period (t_s, t_e). Then, we partition the time period (t_s, t_e) into b blocks for which occupancy counts are measured. We directly compare the difference in occupancy using the mean squared error, as seen in Eqn. 1.

$$\text{OccDist}(D, D') = \frac{1}{|\mathcal{C}|\,|b|} \sum_{C,t_s,t_e} \left| \lambda_D^{C,t_s,t_e} - \lambda_{D'}^{C,t_s,t_e} \right|^2 \quad (1)$$

B. People Trajectory Similarity

The second mechanism used to compare smart space datasets is semantic trajectories as defined in §V. Given datasets D and D', let $\delta_D^{(i)}$ and $\delta_D^{(j)}$ denote the ith and jth semantic trajectory extracted from trajectory sets Δ_D and $\Delta_{D'}$ (corresponding to D, D', respectively. To isolate the effect of confounding variables in our similarity metric, we introduce the notion of *control variables* to partition Δ_D and $\Delta_{D'}$ into "bins". Let V be a set of control variables that we wish to keep constant between compared trajectories; we propose setting V to be the rounded time period (t_s, t_e) for which a trajectory starts and ends (i.e., discretizing the starting/ending times into intervals). That is, a bin corresponding to $V = (t_s, t_e)$ will contain trajectories starting around t_s and ending around t_e. Then, let Δ_D^V denote the subset of trajectories in the bin corresponding to V, as illustrated in Fig. 3. This process yields partitioned trajectories Δ_D^V and $\Delta_{D'}^V$ from D and D', respectively.

To compare Δ_D^V against $\Delta_{D'}^V$, a distance metric between individual trajectories must be defined: let $\phi\left(\delta_D^{(i)}, \delta_{D'}^{(j)}\right)$ be a function that takes two trajectories and returns the "distance" between them. In our work, we propose using the Fréchet distance metric [40]–[42] for its sensitivity properties. Given this distance metric, we can then compute a *cost matrix* of distances between trajectories in Δ_D^V and $\Delta_{D'}^V$ to make a minimum-cost matching MAT; this is an application of the unbalanced assignment problem [43]. Since the size of each of the trajectory sets can be different (and thus one trajectory from the smaller set can be matched with multiple trajectories from the larger set), we define the distance between a given matching $\left(\delta^{(i)}, \{\delta^{(j_1)}, \ldots \delta^{(j_n)}\}\right)$ as the total distance between $\delta^{(i)}$ and each $\delta^{(j_k)}, k \in 1 \ldots n$. We additionally penalize the difference in the sizes of the trajectory sets, weighted by the constant α. Thus, our proposed distance similarity metric for semantic trajectory sets Δ_D and $\Delta_{D'}$ is the sum over control variables V and matching MAT of the individual trajectory distance $\phi\left(\delta_D^{(i)}, \delta_{D'}^{(j)}\right)$, as shown in Eqn. 2.

$$\text{TrajDist}(D, D') = \frac{1}{|V|} \cdot \sum_{v \in V} \sum_{(\delta^{(i)}, \{\delta^{(j_k)}\}) \in MAT} \left(\sum_k \phi\left(\delta_D^{(i)}, \delta_{D'}^{(j)}\right) + \alpha |\Delta_D^v - \Delta_{D'}^v| \right) \quad (2)$$

C. Interpreting Dataset Similarity

While the metrics above help to quantify the similarity of smart space datasets D, D', we must further define a methodology to interpret the similarity of the values obtained in order to determine the degree of realism between D, D'.

To this end, suppose that we split a real-world dataset D into multiple smaller datasets $\{D_i\}$ (e.g., by the week in which the data was measured). Then, using D_i as training data for the generator $\mathcal{G}$, we can produce k simulated datasets $D'_{i,k}$. Let RS_i denote the set of values obtained by comparing the *Real* dataset D_i against each *Simulated* dataset $D'_{i,k}$, wrt. the metric described in §VI-A or §VI-B. We also define RR_i to be the set of values obtained by comparing D_i against each of the other datasets D_j, where $j \neq i, i+1$. Intuitively, RS_i represents the distribution of similarity between synthetic

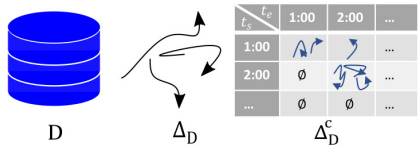


Fig. 3: Extraction of trajectories and placement into bins.

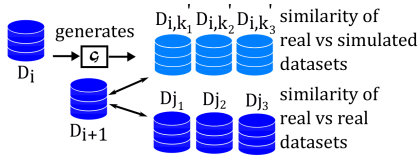


Fig. 4: Comparison of real and simulated datasets.

datasets and real-world datasets, while RR_i represents the distribution of similarity between only real-world datasets. This process is depicted in Fig. 4. After computing RR_i and RS_i, we run an *ANOVA* test with significance level α, which produces a p-value; if p-value $\leq \alpha$, then the simulated dataset is distinguishable from the real dataset. Otherwise, there is not enough evidence to support the hypothesis that the simulated dataset is distinguishable from the real-world dataset.

D. Sensitivity Analysis of the Trajectory Distance Metric

We analyze the sensitivity of the trajectory distance metric proposed above wrt. changes in their trajectories. We will show that the proposed trajectory distance metric using the Fréchet distance is resistant against small deviations in spaces, up to a threshold set for each pair of matched trajectories. Intuitively, this enables the metric to express that multiple realistic scenarios can exist for a given real-world trajectory dataset, without penalizing for any and all deviations.

Let $\Delta_P = \{\delta_P\}$ and $\Delta_Q = \{\delta_Q\}$ be sets of trajectories to compare using the trajectory distance metric proposed above. We assume that the metric ϕ computes the Fréchet distance. Let $\delta_P = [C_P^1, \dots C_P^n]$ and $\delta_Q = [C_Q^1, \dots C_Q^m]$ denote arbitrary trajectories in Δ_P and Δ_Q, respectively, which consists of a sequence of spaces. WLOG, assume that $n \leq m$. For the sensitivity analysis, suppose that $\delta_Q^{'}$ denotes a modified trajectory, where one element has been changed: (i) a new space C_Q^* is added to δ_Q; or (ii) a space C_Q^* is removed from δ_Q. Let $\Delta_Q^{'}$ denote a new set where $\delta_Q^{'}$ replaces δ_Q.

We adapt the definition of the (discrete) Fréchet distance from [42] as shown in Eqn. 3. Intuitively, let L denote a list of pairs of spaces from δ_P and δ_Q such that it starts with (C_P^1, C_Q^1), ends with (C_P^n, C_Q^m) and the trajectories inferred from the resulting list match δ_P and δ_Q (after removing consecutive duplicates). Then, $\|L\|$ can be defined to be the maximum displacement between two paired spaces.

$$\begin{aligned}
&\phi(\delta_P, \delta_Q) = \min\left\{\|L\| : \text{L is a coupling of } \delta_P, \delta_Q\right\}, \\
&\quad \text{where } L = [(C_P^{f_1}, C_Q^{g_1}), \dots (C_P^{f_m}, C_Q^{g_m})], \\
&\qquad f : \{1..m\} \to \{1..n\}; g : \{1..m\} \to \{1..m\}, \\
&\qquad\quad f_1 = g_1 = 1, f_m = n, g_m = m, \\
&\qquad\quad \forall_{i \in 1..m} : f_{i+1} \in \{f_i, f_{i+1}\}, g_{i+1} \in \{g_i, g_{i+1}\} \\
&\quad \|L\| = \max\{d(C_P^{f_i}, C_Q^{g_i})\}
\end{aligned} \tag{3}$$

There are three main factors that determine the total trajectory distance as defined in Eqn. 2: (i) the choice in metric (in our case, we propose using the Fréchet distance, which enables us to inherit some of its properties); (ii) the matching produced by the unbalanced assignment problem; and (iii) the control variables used to bin trajectories. In our analysis, we will focus upon the effects produced by (ii), as different metrics will provide different properties and the effects of binning will depend entirely on the control variables chosen by the user.

Case 1. $|\Delta_P| = |\Delta_Q| = 1$. Since there is only one trajectory in each set Δ_P, Δ_Q, there will be only one possible matching. The properties of $\phi(\Delta_P, \Delta_Q)$ will be the same as $\phi(\delta_P, \delta_Q)$. Therefore, $\phi(\Delta_P, \Delta_Q^{'}) = \phi(\delta_P, \delta_Q^{'})$: the total distance will change only if the added/removed space C_Q^* modified the previous maximum pair in L. Thus, a key property of using the Fréchet distance is that the total distances is resistant against small changes in trajectories. This allows multiple trajectories to potentially yield the same total distance, which allows variation in trajectory sets to exist without penalty.

Case 2. $|\Delta_P| = 1, |\Delta_Q| = q$. In this case, the matching will be between $\delta_P \in \Delta_P$ to each $\delta_Q \in \Delta_Q$ as no other matching exists. The change between $\phi(\Delta_P, \Delta_Q^{'})$ and $\phi(\Delta_P, \Delta_Q)$ is bounded by the change in Case 1: $\phi(\Delta_P, \Delta_Q^{'}) = \phi(\Delta_P, \Delta_Q) - \phi(\delta_P, \delta_Q) + \phi(\delta_P, \delta_Q^{'})$. One difference from Case 1 is the term $\alpha|q-1|$, where α is a user-defined penalty for differences in trajectory set sizes.

Case 3. $|\Delta_P| = p, |\Delta_Q| = 1$. On the other hand, if we suppose that δ_Q is modified into $\delta_Q^{'}$, then there will be a larger effect on the total distance as compared to Case 2: every δ_P must recompute their distance from the newly modified $\delta_Q^{'}$.

Case 4. $|\Delta_P| = p, |\Delta_Q| = q$. In this last case, it is possible for the matching between trajectories in Δ_P and Δ_Q to change, thus changing the total trajectory distance. If we suppose that δ_Q is modified into $\delta_Q^{'}$ and the matching between the two trajectory sets does not change, then, the change in the total trajectory distance will be equivalent to case 1,2 or 3. Otherwise, if the matching does change, then only newly reassigned pairs can contribute to the final total distance.

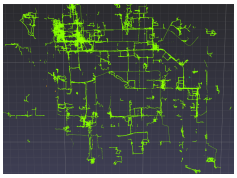
VII. Experiments

We evaluate the accuracy of SmartSPEC using two real-world datasets and baselines based on human activity models. We examine the learned models from the *Scenario Learning* component and the synthetic data produced from the *Scenario Generation* component. We also demonstrate the utility of SmartSPEC with additional hypothetical scenarios.

Datasets. We evaluated the realism of datasets produced by SmartSPEC with respect to two real-world datasets. The first dataset contains WiFi connectivity events captured in a 6 floor campus building at the University of California, Irvine, instrumented with 64 WiFi APs. The building has 4 lecture halls, 10 classrooms, 125+ faculty offices, 90+ research labs, along with other facilities, as seen in 5a. Our dataset spanned 5 weeks and contained an average of 300K device connections each week. We cleaned this dataset by removing entries that were outside of the range [8:00,21:00] to eliminate extreme outliers. To lighten the effect of people who only pass by the building without attending any semantic events inside, we removed entries that connected fewer than 10 times in a week. Due to the intermittent nature in which devices connect to WiFi APs, we associated each connection with a validity period of 10 minutes, which assumes a person's spatial presence for at most 10 minutes. Then, we use the resulting connectivity dataset with the methodology of §VI to extract necessary aspects of smart space datasets for evaluation.

To show the generality of SmartSPEC, the second dataset we used contains GPS trajectories collected in Beijing, China by Microsoft Research Asia as part of the GeoLife project [44]–[46]. Out of the 182 users spanning over 5 years

(a) Bren Hall, UC Irvine.



(b) Geolife GPS.

Fig. 5: Real-world datasets.

featured in the dataset, we examine 63 users over ∼2 years who indicated walking in their trajectory logs, for an average of of 36K GPS logs per month. Fig. 5b shows the GPS traces used. To define a notion of "spaces" from the GPS coordinates, we used OpenStreetMap [47] to extract 1,150 points of interest (POI) in Beijing, and clustered each GPS coordinate to its nearest POI, yielding a connectivity dataset.

Experimental Setup. The campus dataset was partitioned into 5 periods of 1 week each, whereas the Geolife dataset was partitioned into 28 periods of 1 month each. In both experiments, we learn a set of metaevents/metapeople for each week-long or month-long period. Here, we considered time in blocks of 5 minutes and applied a exponential moving average function to smoothen the resulting occupancy time-series. Each simulation spans for the same amount of time and uses the same number of events/people as their associated seed data (i.e., ∼1650/∼4800 people per week for the campus scenario and ∼200/∼7 people per month for the city scenario). For the scenario generation phase, we created a default "leisure" event that people would attend if there were no other suitable events; this "leisure" event occurs at designated outdoor space for 10±2 minutes. For each set of learned metaevents/metapeople, we generated 3 simulations and average their results. Configuration files and metamodels used in the experiments are available on GitHub [32].

Baselines. Smart space dataset generation is based on realistic semantic trajectories in SmartSPEC. We compared our approach against four heavily studied human mobility models commonly used in application domains requiring simulations (e.g., transportation, urban planning, disaster management, etc.) [48]. Each model realistically replicates human mobility while differing in the mechanism used to select the next space.

In the *Random Waypoint* (RAND) mobility model [16], [17], [49], [50] and its variants, people select the next space to visit at (mostly) random without any restrictions. The *Brownian Motion* (BROW) mobility model [51], [52] is similar to the RAND model, but constrains the next space to be adjacent to a person's current space. However, to address the shortcomings of the RAND and BROW models, the Lévy flight model (LÉVY) was introduced [16]–[20], which imposed a power-law probability distribution on the potential spaces to visit based on their distance from a person's current space with power law frequency. Our last baseline mobility model relies on exponential preferential return [21]–[23], which augments the LÉVY mobility model so that a person is more likely to select a previously visited space next. This proceeds in one of two cases: (i) the LÉVY model is used to select a new space to visit; or (ii) a previously visited space is chosen with probability proportional to the number of times visited.

In each mobility model, all people enter the simulated space in the morning (sampled from 10:00 ± 2 hrs), exit the simulated space in the evening (sampled from 18:00 ± 2 hrs) and take the shortest path to move to their next location. In each visited space, a person spends 60 ± 20 minutes before selecting the next space. SmartSPEC learns such values for various profiles of people with the training smart space dataset.

A. Evaluating Learned Models

SmartSPEC performs an event-driven generation of semantic trajectories where people move to spaces to attend events. We validate the process of learning metatrajectories by comparing generated trajectories against the expected path that individuals take (shortest path in graph). We evaluate learned events by their comparing event occupancy counts against ground truth. We perform this validation based on the campus dataset only, for which we know ground truth.

MetaTrajectories. To evaluate the learned people directly, we must establish ground truth by identifying a number of device owners and tracking the types of events they attend as well as recording times at which they enter/exit the sensorized space. However, obtaining such data at scale is challenging and poses many privacy concerns. Hence, we instead evaluated the quality of metatrajectories which represent movement patterns of individuals between nonadjacent spaces. We compare such paths produced by the *MetaTrajectory Learner* against real paths extracted from data. We consider the shortest path algorithm as a baseline. Our evaluation followed the methodology shown in Fig. 4 - namely, we split the campus dataset into periods of 1 week and compared the current week against the consecutive week. Tab. I shows the distance between metatrajectories computed using Eqn. 2.

TABLE I: Avg. MetaTrajectory Distance (m).

	Week 1	Week 2	Week 3	Week 4
Real	21.93	20.85	20.68	21.37
SmartSPEC	21.63	21.03	20.69	19.44
Shortest Path	12.00	13.35	10.63	12.27

Our results show that the learning of metatrajectories by SmartSPEC results in realistic paths between non-adjacent spaces. In general, the shortest path resulted in realistic situations only when the source and destination spaces were physically located near each other. We find that the average difference between real-world trajectories and shortest path trajectories is much greater than when using SmartSPEC.

Event Occupancy. We compared events learned by SmartSPEC against ground truth events that were formally advertised in the campus building. In particular, we consulted the campus-wide schedule of classes for logistical data on courses (e.g., time, periodicity, enrollment), and other advertised events (e.g., seminars, club meetings), and identified a total of 510 events out of an average of 1,424 events learned by SmartSPEC each week. We made a best-effort attempt to map each event (and their associated enrollments) to WiFi APs regions. Note that, since we are mapping events to regions,

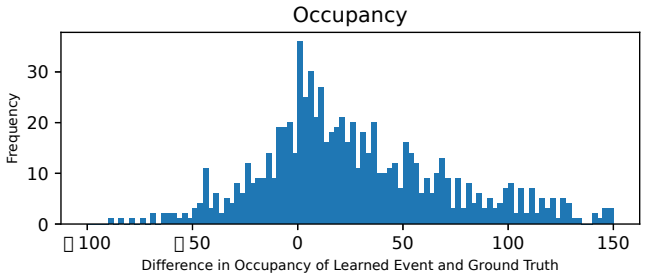


Fig. 6: Histogram of differences in event occupancy.

more than one event can happen in the same region at the same time and in those cases we will combine them into a single event. For instance, the region that covers two classrooms will be associated with two classes at the same time. This means that the attendance to the event that combines them both will be the sum of the attendance to both classes. Additionally, we assumed that attendance in ground truth events (e.g., courses) was typically 30-70% of official enrollment counts/maximum occupancy of hosting spaces to account for absentees.

In our analysis, we made a best-effort attempt to match each ground truth event with its learned counterpart based on their start/end times. The average paired difference in start/end times was 15 ± 18 and 21 ± 27 minutes, respectively. In Fig. 6, we plotted the difference in occupancy between learned and ground truth events and found that a majority of events had minimal differences in occupancy. We note that it is infeasible to capture all ground truth events for comparison: people can spend time in the building while not attending any formal event (e.g., wait in lobby), which helps explain the heavy tails.

B. Evaluating Generated Semantic Trajectories

We evaluate realism of the produced semantic trajectories for two real-world datasets and compare them against the baselines, using the methodology and metrics in §IV and §VI.

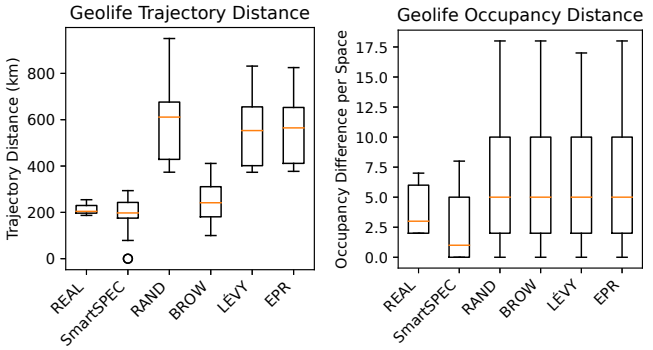
We split the campus dataset into periods of 1 week and binned the extracted semantic trajectories using rounded half-hour intervals of start/end times as control variables (i.e., each trajectory was characterized by the closest half-hour for which their start/end times fall). In Tab. II, we report the average similarity in semantic trajectories, measured as the distance (m) as explained in §VI, over all bins. We add a row "Real" to Tab. II which represents the comparison of real semantic trajectories to each other. The results show that, in general, SmartSPEC was able to produce semantic trajectories whose distance from extracted real-world semantic trajectories was close to that of other real-world semantic trajectories: on average, there was only a difference of 35% between the "real" dataset and the SmartSPEC dataset, while the other had an average percent change of 280%, 153%, 264%, and 208%, as listed. An interesting outcome was that the BROW baseline seemed to do better than other baselines; we observe this phenomena because in the BROW mobility model, people select to move to adjacent locations, where they stay for some time. Overall, this produces semantic trajectories largely located in only one area, which allows the distance

TABLE II: Avg. trajectory similarity (m) in campus dataset.

	Week 1	Week 2	Week 3	Week 4
Real	185.65	188.67	191.31	194.60
SmartSPEC	263.92	252.09	272.43	240.99
RAND	789.8	754.07	740.23	606.74
BROW	533.27	479.68	501.39	407.32
LÉVY	760.3	713.53	713.18	583.97
EPR	693.38	554.26	635.81	459.4

to be approximated with a simple max. We observe this phenomenon for the Geolife dataset as well. It is clear that SmartSPEC was able to significantly outperform all baselines in the campus scenario.

For the Geolife dataset, we made splits of 1 month, but kept the same binning strategy as described above. Since this dataset spans over 2 years, we report the range of semantic trajectory distances (km) computed using the boxplots in Fig. 7a. Our result similarly show that SmartSPEC was able produce semantic trajectories in a large city setting that were close to that of real-world semantic trajectories: the average percentage difference was -13% compared to the 177%, 18%, 159% and 159% of the RAND, BROW, LÉVY and EPR baselines. It is clear that the RAND, LÉVY, and EPR baselines were unable to produce realistic semantic trajectories: the resulting values were significantly larger than what was expected. For the BROW baseline, we observed the same phenomena as with the campus dataset: the produced semantic trajectories are mostly in one area. However, since the Geolife dataset is set on the scale of a city, the effect of producing semantic trajectories in one area is more noticeable - people will typically not walk across the entire city to get from one place to the next. Overall, SmartSPEC outperforms the baselines for the Geolife dataset.



(a) Trajectory distance (m). (b) Occupancy distance per space.
Fig. 7: Geolife dataset distances.

C. Evaluating Generated Occupancy

We evaluate the realism of the synthetic datasets produced by SmartSPEC wrt. occupancy. We use the same datasets that were generated for the semantic trajectory analysis and computed the occupancy of each space in 10 minute intervals.

The results for the campus dataset, as listed in Tab. III, show that the occupancy of spaces inferred by the semantic trajectories of people generated by SmartSPEC was close to occupancy found from our real-world data. On average, there was a 36% change in the occupancy counts per space in the synthetic dataset produced by SmartSPEC, compared to the

120%, 96%, 114%, and 107% change in the datasets produced by the RAND, BROW, LÉVY, and EPR mobility models, respectively. Thus, SmartSPEC does much better than each of the baselines wrt. occupancy in the campus dataset.

We present the results for Geolife in Fig. 7b as a boxplot which shows the average occupancy difference per space, over all 1-month dataset splits. The results show a -37% difference in occupancy when comparing real space occupancy and SmartSPEC occupancy. In contrast, the RAND, BROW, LÉVY, and EPR baselines all have around a 68.5% difference. In fact, each of the baselines also has similar behavior in terms of the distribution of the occupancy difference found: they have the same average and interquartile range, with a heavy, one-sided tail. This implies that most of the 1-month datasets that were synthetically produced are similar to each other, but that there was a number of outlying datasets skewed results to high occupancy levels. We observe that SmartSPEC produces datasets that vary less from real-world data; there are also fewer outlying occupancy differences.

TABLE III: Occupancy difference per space in campus dataset.

	Week 1	Week 2	Week 3	Week 4
Real	6.67	5.45	7.29	5.96
SmartSPEC	8.63	10.0	7.16	8.61
RAND	14.20	13.92	14.01	13.65
BROW	12.29	12.37	12.75	12.34
LÉVY	13.83	13.49	13.64	13.23
EPR	14.75	12.86	14.83	10.05

D. Applicability and Utility of SmartSPEC

To demonstrate the applicability of SmartSPEC and utility of its generated datasets, we constructed a "daily life" scenario in two different settings: a mall and an airport. In the mall setting, we simulated store personnel, customers (both regular and infrequent) and mall management and present a visualization of their occupancy at a certain point in time in Fig. 8. Similarly, we simulated airport security personnel, passengers, restaurant staff and others for the airport scenario and visualize a snapshot of part of the airport in Fig. 9. Such datasets can be used to study hypothetical situations and optimize space usage. For example, in the mall setting, we can use the occupancy counts to identify spaces with high traffic - these spaces can guide the placement for a new shop or be used as a heuristic to know when physical advertisements are best distributed. Alternatively, occupancy levels in an airport setting can be used to help optimize the time taken for passengers to board/deboard a plane (e.g., by hypothetically changing the gates that airplanes use), or to identify densely populated areas that require more security/monitoring. In both scenarios, the smart space dataset generated by SmartSPEC can also be leveraged to generate other sensor data such as potential WiFi connectivity events, changes in temperature registered by the HVAC sensors, or number of people in pictures taken by security cameras. SmartSPEC may also aid in enabling resilience of critical facilities under extreme scenarios; this can be done thought the modeling of activities such as evacuations and sheltering in natural and man-made disasters such as

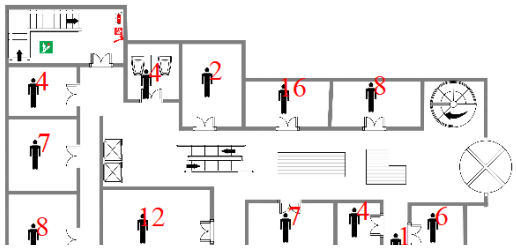


Fig. 8: Occupancy in a simulated mall.

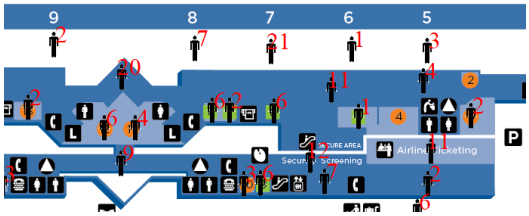


Fig. 9: Occupancy in a simulated airport.

structural fires and active shooter events, which can help in designing spaces that improve public safety.

VIII. Conclusions and Future Work

We presented SmartSPEC, an event-driven smart space simulator and data generator that produces customizable smart space datasets using models of spaces, people, events and sensors. In SmartSPEC, we ease the process of defining such models by applying ML techniques to extract higher-level metamodels of people and events from input connectivity data. Through this process, we describe new scenarios and generate new synthetic datasets. We also introduced a new structured methodology to evaluate the realism of synthetic data. The experimental results show that SmartSPEC generates realistic datasets in different situations using input seed data. However, the quality of the generated dataset is obviously limited by the quality of the metamodels derived from the seed data. Reusing such metamodels in domains other than the one for which they were created remains an open challenge for SmartSPEC. We are planning to explore the generalization of people and event metamodels extracted from input datasets by removing domain-specific bias. To strengthen the usability of SmartSPEC, we are developing graphical tools to simplify the process of modeling the geographical component of the space itself, thus alleviating the effort required to dynamically change the layout. Finally, we envision that SmartSPEC can be used as a starting point towards the design of next-generation, reconfigurable spaces and flexible buildings.

Acknowledgements

This material is based on research sponsored by DARPA under Agreement No. FA8750-16-2-0021. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright notation thereon. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of DARPA or the U.S. Government. This work is also supported by NSF Grants No. 2032525, 1952247, 2008993 and 2133391.

REFERENCES

[1] R. Yus, G. Bouloukakis, S. Mehrotra, and N. Venkatasubramanian, "Abstracting interactions with iot devices towards a semantic vision of smart spaces," in *6th ACM Int. Conf. on Systems for Energy-Efficient Buildings, Cities, and Transportation*.

[2] L. Ying, Z. Liu, D. Towsley, and C. H. Xia, "Distributed operator placement and data caching in large-scale sensor networks," in *27th Conf. on Computer Communications*, 2008.

[3] A. da Silva Veith, M. D. de Assuncao, and L. Lefevre, "Latency-aware placement of data stream analytics on edge computing," in *Int. Conf. on Service-Oriented Computing*, 2018.

[4] K. E. Benson, G. Bouloukakis, C. Grant, V. Issarny, S. Mehrotra, I. Moscholios, and N. Venkatasubramanian, "Firedex: a prioritized iot data exchange middleware for emergency response," in *19th Int. Middleware Conf.*, 2018.

[5] A. Martínez-Ballesté, P. A. Pérez-Martínez, and A. Solanas, "The pursuit of citizens' privacy: a privacy-aware smart city is possible," *IEEE Communications Magazine*, vol. 51, no. 6, 2013.

[6] P. Pappachan, M. Degeling, R. Yus, A. Das, S. Bhagavatula, W. Melicher, P. E. Naeini, S. Zhang, L. Bauer, A. Kobsa, S. Mehrotra, N. M. Sadeh, and N. Venkatasubramanian, "Towards privacy-aware smart buildings: Capturing, communicating, and enforcing privacy policies and preferences," in *37th IEEE Int. Conf. on Distributed Computing Systems Workshops*, 2017.

[7] S. Nastic, T. Rausch, O. Scekic, S. Dustdar, M. Gusev, B. Koteska, M. Kostoska, B. Jakimovski, S. Ristov, and R. Prodan, "A serverless real-time data analytics platform for edge computing," *IEEE Internet Computing*, vol. 21, no. 4, 2017.

[8] Z. Qin, G. Denker, C. Talcott, and N. Venkatasubramanian, "Achieving resilience of heterogeneous networks through predictive, formal analysis," in *Proceedings of the 2nd ACM international conference on High confidence networked systems*.

[9] N. Venkatasubramanian, C. Talcott, and G. A. Agha, "A formal model for reasoning about adaptive qos-enabled middleware," *ACM Transactions on Software Engineering and Methodology (TOSEM)*, vol. 13, no. 1.

[10] D. Jones, C. Snider, A. Nassehi, J. Yon, and B. Hicks, "Characterising the digital twin: A systematic literature review," *CIRP Journal of Manufacturing Science and Technology*, vol. 29.

[11] A. Rasheed, O. San, and T. Kvamsdal, "Digital twin: Values, challenges and enablers from a modeling perspective," *IEEE Access*, vol. 8.

[12] J. Bruneau and C. Consel, "Diasim: a simulator for pervasive computing applications," *Software: Practice and Experience*, vol. 43, no. 8, 2013.

[13] J. Synnott, L. Chen, C. D. Nugent, and G. Moore, "The creation of simulated activity datasets using a graphical intelligent environment simulation tool," in *36th Annual Int. Conf. of the IEEE Engineering in Medicine and Biology Society*, 2014.

[14] J. W. Lee, S. Cho, S. Liu, K. Cho, and S. Helal, "Persim 3d: Context-driven simulation and modeling of human activities in smart spaces," *IEEE Transactions on Automation Science and Engineering*, vol. 12, no. 4, 2015.

[15] N. Alshammari, T. Alshammari, M. Sedky, J. Champion, and C. Bauer, "Openshs: Open smart home simulator," *Sensors*, vol. 17, no. 5, 2017.

[16] M. C. Gonzalez, C. A. Hidalgo, and A.-L. Barabasi, "Understanding individual human mobility patterns," *Nature*, vol. 453, no. 7196.

[17] D. Brockmann, L. Hufnagel, and T. Geisel, "The scaling laws of human travel," *Nature*, vol. 439, no. 7075.

[18] A. Noulas, S. Scellato, R. Lambiotte, M. Pontil, and C. Mascolo, "A tale of many cities: universal patterns in human urban mobility," *PloS one*, vol. 7, no. 5.

[19] 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*, vol. 19, no. 3.

[20] M. U. Kraemer, A. Sadilek, Q. Zhang, N. A. Marchal, G. Tuli, E. L. Cohn, Y. Hswen, T. A. Perkins, D. L. Smith, R. C. Reiner *et al.*, "Mapping global variation in human mobility," *Nature Human Behaviour*, vol. 4, no. 8.

[21] C. Song, T. Koren, P. Wang, and A.-L. Barabási, "Modelling the scaling properties of human mobility," *Nature Physics*, vol. 6, no. 10.

[22] L. Alessandretti, U. Aslak, and S. Lehmann, "The scales of human mobility," *Nature*, vol. 587, no. 7834.

[23] A. D. Nguyen, P. Sénac, V. Ramiro, and M. Diaz, "Steps-an approach for human mobility modeling," in *Int. Conf. on Research in Networking*.

[24] T. Brinkhoff, "A framework for generating network-based moving objects," *GeoInformatica*, vol. 6, no. 2, 2002.

[25] J. Xu and R. H. Güting, "Mwgen: a mini world generator," in *IEEE 13th Int. Conf. on Mobile Data Management*, 2012.

[26] N. Pelekis, S. Sideridis, P. Tampakis, and Y. Theodoridis, "Hermoupolis: a semantic trajectory generator in the data science era," *SIGSPATIAL Special*, vol. 7, no. 1, 2015.

[27] L. Pappalardo and F. Simini, "Data-driven generation of spatio-temporal routines in human mobility," *Data Mining and Knowledge Discovery*, vol. 32, no. 3, 2018.

[28] L. Rossi, M. Paolanti, R. Pierdicca, and E. Frontoni, "Human trajectory prediction and generation using lstm models and gans," *Pattern Recognition*, vol. 120.

[29] V. Kulkarni, N. Tagasovska, T. Vatter, and B. Garbinato, "Generative models for simulating mobility trajectories," *arXiv preprint arXiv:1811.12801*, 2018.

[30] K. Ouyang, R. Shokri, D. S. Rosenblum, and W. Yang, "A non-parametric generative model for human trajectories." in *IJCAI*.

[31] A. Gupta, J. Johnson, L. Fei-Fei, S. Savarese, and A. Alahi, "Social gan: Socially acceptable trajectories with generative adversarial networks," in *IEEE Conf. on Computer Vision and Pattern Recognition*.

[32] "Smartspec," https://github.com/andrewgchio/SmartSPEC, 2022.

[33] P. Merrell, E. Schkufza, and V. Koltun, "Computer-generated residential building layouts," in *ACM SIGGRAPH Asia 2010 papers*.

[34] S. Aminikhanghahi and D. J. Cook, "A survey of methods for time series change point detection," *Knowledge and Information Systems*, vol. 51, no. 2, 2017.

[35] D. Eads, "hcluster: Hierarchical clustering for scipy," *URL http://scipy-cluster. googlecode. com*, 2008.

[36] Y. Lin, D. Jiang, R. Yus, G. Bouloukakis, A. Chio, S. Mehrotra, and N. Venkatasubramanian, "LOCATER: cleaning wifi connectivity datasets for semantic localization," *Proc. VLDB Endow.*, vol. 14, no. 3, 2020.

[37] P. Venkateswaran, V. Muthusamy, V. Isahagian, and N. Venkatasubramanian, "Environment agnostic invariant risk minimization for classification of sequential datasets," in *27th ACM SIGKDD Conf. on Knowledge Discovery & Data Mining*.

[38] V. Guralnik and J. Srivastava, "Event detection from time series data," in *5th ACM SIGKDD Int. Conf. on Knowledge Discovery and data Mining*, 1999.

[39] C. Truong, L. Oudre, and N. Vayatis, "Selective review of offline change point detection methods," *Signal Processing*, 2019.

[40] H. Alt and M. Godau, "Computing the fréchet distance between two polygonal curves," *Int. Journal of Computational Geometry & Applications*, vol. 5, no. 01n02, 1995.

[41] K. Toohey and M. Duckham, "Trajectory similarity measures," *Sigspatial Special*, vol. 7, no. 1, 2015.

[42] T. Eiter and H. Mannila, "Computing discrete fréchet distance," Citeseer, Tech. Rep., 1994.

[43] L. Ramshaw and R. E. Tarjan, "On minimum-cost assignments in unbalanced bipartite graphs," *HP Labs, Palo Alto, CA, USA, Tech. Rep. HPL-2012-40R1*, 2012.

[44] Y. Zheng, L. Zhang, X. Xie, and W.-Y. Ma, "Mining interesting locations and travel sequences from gps trajectories," in *18th Int. Conf. on World Wide Web*.

[45] Y. Zheng, Q. Li, Y. Chen, X. Xie, and W.-Y. Ma, "Understanding mobility based on gps data," in *10th Int. Conf. on Ubiquitous Computing*.

[46] Y. Zheng, X. Xie, W.-Y. Ma *et al.*, "Geolife: A collaborative social networking service among user, location and trajectory." *IEEE Data Eng. Bull.*, vol. 33, no. 2.

[47] OpenStreetMap contributors, "Planet dump retrieved from https://planet.osm.org ," https://www.openstreetmap.org, 2017.

[48] G. Solmaz and D. Turgut, "A survey of human mobility models," *IEEE Access*, vol. 7.

[49] R. Gallotti, A. Bazzani, S. Rambaldi, and M. Barthelemy, "A stochastic model of randomly accelerated walkers for human mobility," *Nature Communications*, vol. 7, no. 1.

[50] V. Zaburdaev, M. Schmiedeberg, and H. Stark, "Random walks with random velocities," *Physical Review E*, vol. 78, no. 1.

[51] P. G. Lind and A. Moreira, "Human mobility patterns at the smallest scales," *Communications in Computational Physics*, vol. 18, no. 2.

[52] R. Groenevelt, E. Altman, and P. Nain, "Relaying in mobile ad hoc networks: The brownian motion mobility model," *Wireless Networks*, vol. 12, no. 5.