UrbanGen

Generating Combined In- and Outdoor Trajectories

Sun, Yunkai; Nielsen, Nichlas; Xie, Xike; Pedersen, Torben Bach; Simonsen, Ulf; Lu, Hua; Ainciburu, Maite

Published in:
Proceedings - 2022 23rd IEEE International Conference on Mobile Data Management, MDM 2022

DOI (link to publication from Publisher):
10.1109/MDM55031.2022.00057

Publication date:
2022

Document Version
Accepted author manuscript, peer reviewed version

UrbanGen: Generating Combined In- and Outdoor Trajectories

Yunkai Sun*, Xike Xie*, Nichlas Nielsen†, Torben Bach Pedersen†, Ulf Simonsen†, Hua Lu‡, Maite Ainciburu§
* University of Science and Technology of China, China Email: yunkaisun@mail.ustc.edu.cn, xkxie@ustc.edu.cn
† Aalborg University, Denmark Email: nichlasdk@gmail.com, tbp@cs.aau.dk, ulfsim@gmail.com
‡ Roskilde University, Denmark Email: luhua@ruc.dk
§ Polytech Montpellier, France Email: ainciburu.maite@gmail.com

***Abstract*—The prevalence of mobile devices and positioning techniques has enabled so-called traffic-aware urban computing. In urban daily life, people's activities consist of indoor and outdoor parts whose transitions have important impacts on traffic behavior. For example, outdoor traffic events (e.g., jams) can be triggered by indoor events (e.g., the ending of exhibitions, tournaments, or working hours). In turn, indoor events (e.g., subway jams) can also be affected by outdoor events (e.g., snowy weather or other bad outdoor traffic conditions). For a wide range of applications like traffic monitoring and emergency response, it is thus interesting to develop techniques for analyzing data in an integrated indoor and outdoor space. Since real datasets of this kind are still scarce and small, a suitable data generator is needed for both functional and scalability testing. In this work, we present UrbanGen, which follows the constraints of road networks for the outdoor space and the constraints of topologies for the indoor space. The system provides the functionalities including: 1) integrating a model of indoor topologies with the state-of-art road networks; 2) parameterizing the movement of objects in the integrated model; 3) serializing and visualizing the generated trajectories.**

***Index Terms*—trajectories, in- and outdoor spaces, moving objects, visualization**

I. Introduction

It is possible to provide location information for integrated urban topologies including both indoor and outdoor spaces, as positioning becomes increasingly available with techniques such as GPS, Wi-Fi, and cellular networks. It enables the study of moving behavior in the combination of the two spaces, and their mutual influence. For example, when monitoring and predicting traffic trends of road networks, it is relevant to consider the potential effects of indoor moving objects. In turn, indoor movements can be affected by outdoor traffic conditions. Synthesizing both sides offers insights into object moving patterns and enables a broader range of applications such as allocating public or commercial resources, e.g., police forces and mobile base stations. However, there is a lack of such real data sets, which are to facilitate data management research. Thus, we develop a trajectory data generator, UrbanGen, based on an integrated topology. Doing this, a number of challenges arise.

First, in- and outdoor spaces follow different topology constraints. For outdoor space, we assume a road network which is general in addressing moving object behaviors or traffic conditions. The topology is often represented by a graph. Indoor spaces are characterized by entities such as walls, doors, rooms, etc., which render the Euclidean distance and spatial network distance unsuitable. Such entities imply topological constraints that enable or disable movements. Existing solutions either focus on one of two cases [9], [10], [7], or combine them with a simplified model [8]. It is not clear how the two heterogenous models can be merged.

Second, the movement patterns in outdoor and indoor spaces follow different characteristics. For road networks, the movement mainly depends on the current traffic conditions. For example, the movement speed decreases if the capacity of a road segment is exceeded by having too many vehicles. Besides inheriting such characteristics, indoor spaces have the unique feature of dwelling time. For example, an individual can dwell in an indoor unit, i.e., a room, for some events, e.g., a football match or an opera. Also, it remains unclear how to simulate a movement with transitions from outdoor to indoor spaces and vice versa.

In this demonstration, we present UrbanGen which 1) is based on integrated in-/outdoor topologies; 2) generates trajectory data based on parameters specified for different aspects of movements. The input of the generator consists of three parts: the road networks, the templates of buildings, and user-specified parameters. The output of the generator takes two forms: a trajectory dataset conforming to the given constraints, and/or a multi-scaled visualization result.

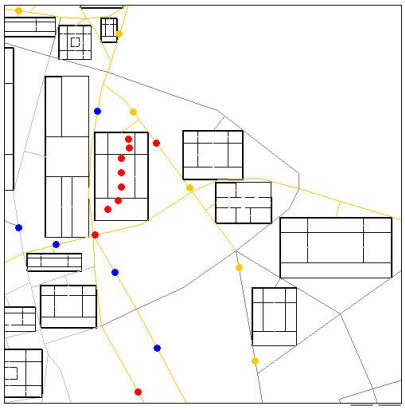

Fig. 1: Example of trajectories in integrated topologies: a red object leaves a building for road networks.

Figure 1 demonstrates an example of trajectories of three objects marked with different colors. We can have two ob-

servations for the red objects. First, the two topologies are integrated such that the object can switch smoothly between them. Second, the object moves slowly in the indoor topology (e.g., when walking) and fast in the outdoor counterpart (e.g., when driving). UrbanGen also considers more advanced topology constraints, e.g., capacities and velocities, for a topology unit, e.g., a road-segment or a room.

Existing works do not cover all the features mentioned above. For outdoor spaces, the two most popular generators are Brinkhoff's generator [3] and BerlinMOD [4]. Mokbel et al. developed a web-based traffic generator [5] that supports both Brinkhoff and BerlinMOD generators and retains flexibility for extension to other generators. However, these generators only consider outdoor moving objects. Huang et al. implemented a data generator for simulating object movement in a RFID-deployed indoor topology [6]. The generator focuses on semantic locations, i.e., ID of RFID sensors, instead of general indoor topologies and it is only applied for indoor spaces. In a parallel effort [7], Vita ignores outdoor spaces and specializes at generating indoor positioning data for simulated indoor movements. Hussein et al. considered an airport bag-tracking system that combines both indoor and outdoor topologies [8]. However, 1) that work simplifies the topologies into semantic locations, i.e., IDs of locations which limited the extension for generally accepted trajectory data formats; 2) it is not a data generator but a system for predicting possible movements with existing settings; 3) it is focused on applications where symbolic locations are important, i.e., where most of the time is spent at the symbolic locations, rather than the transitions between them.

The rest of the paper is organized as follows. Section II presents the UrbanGen architecture. Section III covers the demonstration overview and core components. Section IV concludes the paper.

II. System Overview

In Figure 2, we show the UrbanGen architecture, consisting of three parts: the Input module, the Logical module, and the Output module.

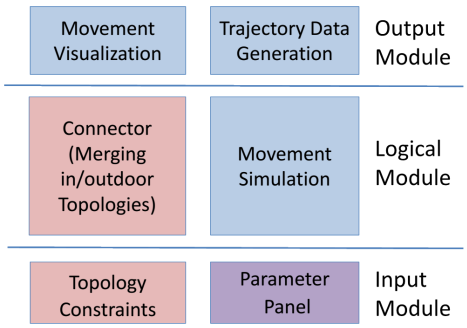

Fig. 2: Architecture: red blocks are for topologies; blue blocks are for trajectories; purple block is for both.

Input module consists of 2 parts: 1) meta-data for describing the topology constraints; 2) parameters for specifying topologies and moving patterns. The outdoor topology is in Brinkhoff's format, i.e., two files representing nodes and edges, respectively. The indoor topology is described by a set of building templates, e.g., residence buildings and shopping malls, whose formats are general and are proposed in our previous work [1]. For industrial formats of indoor spaces, e.g., IFC and CAD, we can use converters, such as [9] (for IFC) and [10] (for CAD), to transform them into our formats. Also, the input module offers a parameter panel to specify the topology parameters (e.g., the number of floors) and movement parameters (e.g., the number of moving objects, and the travelling period).

Logical module interacts with the input module. It has two parts, Connector and Movement Simulator. Connector integrates the two heterogeneous topologies by connecting the exit of a building with the nearest node of the road network. Movement Simulator generates trajectory data conforming to the topology constraints offered by Connector.

Output module offers two outputs: movement visualization and a dataset of generated trajectories. We can choose to hide/show the indoor movements and switch the view point for different levels of topologies, i.e., floors of buildings.

III. Demonstration and Modules

A. Demonstration Overview

We will demonstrate how in-/outdoor trajectories are generated by guiding users throughout the in-/outdoor space integration, the parameter configuration, and the result generation and visualization. Our demonstration will be preloaded with a set of real road networks and indoor building templates (e.g., residence buildings, shopping malls, offices) with different sizes and topologies. An indoor template can have different floors that makes the integrated topology non-planar. The key objective of the demonstration is to show the unique functionalities provided by UrbanGen and to enable the audience to interactively experience its components.

Figure 3 illustrates the relevant components as follows. The parameters are specified by the control panel in (a). The scrolling bar on the right of the panel supports a multi-level view of the map, since the indoor topologies are on different levels. The road network input is shown in (b). The integration is done conforming to the parameters specified in the panel, e.g., max number of floors. An example of the integration result of both topologies is shown in (c). The generated trajectories are shown in (d), where different objects are rendered with different colors. As observed in Figure 3, movement transition patterns, such as dwelling time and number of transfers can also be parameterized by the control panel. In the sequel, we cover UrbanGen's major components to be demonstrated.

B. Graph-based Topology Model

Outdoor topologies are typically represented by a graph structure. However, for the indoor topologies, there are many variants. One variant is the *door graph* [1], where each door is a node and two doors are connected if they are within the same partition (e.g., room). An example of the door graph is shown as the green part in Figure. 4. Then, each edge is coupled with a weight which is the distance between two

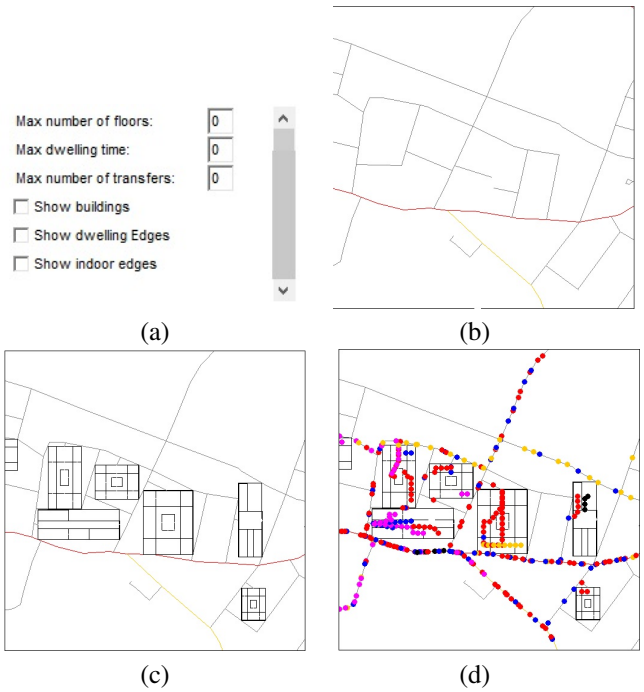

Fig. 3: Demonstration of Trajectory Generation

doors. Another variant is the *accessibility graph* [2], where each partition is a node and two partitions are connected if there exists a door between them. Basically, both models focus on addressing connectivity between different indoor units. Thus, they fall short in fully simulating object movements. If the door graph is applied, the object can never move inside a room because such an element is missing in the graph. Reversely, if the partition graph is applied, it is difficult to simulate the movements parameterized by speed and directions etc, because the distance information is not maintained.

We thus design a graph model which combines the two structures and enables a smooth connection to road networks. Essentially, the new model is to extend the door graph with the following: (a) adding a special class of nodes called *dwelling nodes* (e.g., red circles in Figure 4) for dangling partitions, i.e., partitions with one door, and *dwelling edges* which are the edges connecting to dwelling nodes (e.g., red dashed lines in Figure 4); (b) add the element partition as well as the bidirectional mapping between partitions and edges. Such a mapping is necessary when considering movement models.

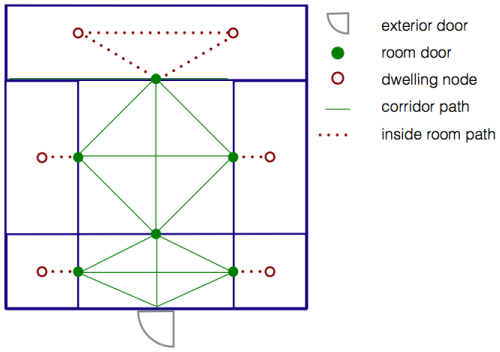

Fig. 4: Indoor Graph Model (the first floor of an office building)

The dwelling nodes and edges capture the movement features of objects inside a room. Details about the movement simulation are covered in later sections. After modeling indoor topologies, paths are built for connecting to road networks.

C. Adding an Indoor Topology Source

The outdoor topology, i.e., a road network, is specified by two binary files describing the nodes and edges of the road network. The binary files can also be transformed from other formats, such as TIGER/Line Files and MIF[1].

The indoor topology of a building is defined by specified *templates*. A template allows for identifying all the components of a building (e.g., nodes, edges, and rooms) by giving their attributes such as types and positions. The position is reported as a percentage of the floor plan's length and width. So, first, a building needs to be declared, initializing the length and width of the floor. Then, nodes and edges are listed and numbered following the order in which they are defined. Finally, partitions including their shapes and relationships with nodes and edges are added. We choose XML to describe building templates, since it is self-descriptive and developer-friendly. The XML format is specified by a DTD file. Figure 5 shows an example fragment of the DTD file. Figure 6 shows an example fragment for the building in Figure 4.

```
<!ELEMENT building
(id,size,coordinates,nodes,edge
s,rooms)>
<!ELEMENT nodes
(node+,downnode+,upnode+,roomno
de+,door)>
<!ELEMENT edges (edge)+>
<!ELEMENT rooms (room)+>
<!ELEMENT node (id,position)>

.....

<!ELEMENT roomnode
(id,position,roomid)>
<!ELEMENT edge
(id,node1,node2)>
<!ELEMENT room
(id,coordinates,edgeIDs)>
```

Fig. 5: Building Template DTD

```
<?xml version="1.0"?>
<!DOCTYPE building SYSTEM "building.dtd">
- <building>
    <id>0</id>
    <size/>
    <coordinates>100 100 0 0</coordinates>
  + <nodes>
  + <edges>
  - <rooms>
    - <room>
        <id>1</id>
        <coordinates>0 0 25 100</coordinates>
        <edgeIDs>2</edgeIDs>
      </room>
    - <room>
        <id>2</id>
        <coordinates>75 0 100 100</coordinates>
        <edgeIDs>3</edgeIDs>
      </room>
    - <room>
        <id>3</id>
        <coordinates>25 0 75 100</coordinates>
        <edgeIDs>1 4 5 6 7 8 9</edgeIDs>
      </room>
```

Fig. 6: Building Template XML

D. Output Trajectory Formats

In real life, people's activities can be represented by intermittent indoor dwelling and outdoor transportation. For example, a daily trajectory of a student is "home→road→school→road→home". So, a trajectory has a starting place, multiple transfers, and a destination which is picked from the domain. Each transfer is set as a node in a building, mimicking some indoor facility, e.g., a bank counter. Between the transfers there are a set of locations which can either be in indoor or outdoor.

We define the trajectory format by a DTD file as shown in Figure 7. A trajectory consists a series of positions. An outdoor position is represented by the entity "oloc" that includes the time and coordinates. An indoor location is represented by the entity "iloc" that includes the time, and identifiers of the room, floor, and building. The physical position, i.e., coordinates, of an indoor location can be recovered with the building identifier

[1]http://iapg.jade-hs.de/personen/brinkhoff/generator/

and road network. An example of an output trajectory with two positions is shown in Figure 8.

```
<!ELEMENT trajectories
(trajectory)>

<!ELEMENT trajectory
(objid,(oloc|iloc)+)>

<!ELEMENT oloc (time,pos)>

<!ELEMENT iloc
(time,buildingID,floorID,roomID
)>

.....
```

Fig. 7: Trajectories DTD

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE trajectories SYSTEM "traj.dtd">
- <trajectories>
  - <trajectory>
      <objid>1</objid>
    - <oloc>
        <time> 0 </time>
        <pos>123.12 8921.21 </pos>
      </oloc>
    - <iloc>
        <time> 1</time>
        <buildingID> 2</buildingID>
        <floorID> 1</floorID>
        <roomID> 0</roomID>
      </iloc>
  </trajectory>
</trajectories>
```

Fig. 8: Trajectories XML

E. Connector Component

Connector that connects the two heterogenous topologies is implemented as follows.

Find faces from road networks. The first step in the implementation of Connector is to find all Faces from the road network. A Face is a region of a graph consisting of edges where no two edges are crossing. In road networks, a face corresponds to a plot or block of a street, where a building is placed. To find the faces, we used the Planar Face Traversal function from the C++ library Boost[2]. This function moves through the graph and reports back all nodes that are part of one Face before moving to the next Face. When we have found the faces, we check if they are correct and remove those that are not.

Find largest rectangle inside a face. Since many of the faces found might have a shape that is unfit for a building, we find the largest rectangle inside the face. This is called the *maximum rectangle problem*[3]. We adopt an algorithm[4] for the problem that first approximates a face by a square mesh and retrieves the maximum rectangle through the stepwise detection. When the largest rectangle is found, we can place a building correctly within the rectangle.

Scale up the template to the rectangle. Our building templates are made to be easily scaled since all of the positions of room and nodes are based on percentage values. We also know the height and width of each building, since we know the position of the lower left corner and the upper right corner of them from the maximum rectangle found previously. So, for the scaling, we use the width of the building and the percentage values to calculate the positions of nodes in the building on the x-axis, and we do the same with the height to calculate the positions on the y-axis.

Connect the building to the roads. In our building templates, we define where the position of the door going outside to the road network is located. We can use this information to find the closest point on the nearest edge located on the road network. This point is then turned into a node in the network and the edge it was located on is then split in two and both parts are connected to the new node. Then we make a new edge from the door node to the new node, thus connecting the building to the road network.

[2]http://www.boost.org/doc/libs/1_39_0/libs/graph/doc/planar_face_traversal.html

[3]http://www.drdobbs.com/database/the-maximal-rectangle-problem/184410529#disqus_thread

[4]https://gist.github.com/mmadson/9637974

F. Movement Simulation

For simulating a trajectory, we consider the random waypoint model which is a commonly used synthetic model for mobility. The trajectory of an object contains multiple transfers, each of which is a node of some building. To determine its next transfer (or destination), we randomly select a building and then randomly select a node within the building. Then, the object moves towards the selected transfer and the process repeats until the counter for the number of transfers exceeds $max\_number\_of\_transfers$. Moveover, every time an object reaches a transfer, it will dwell on the place for some time which is decided by randomly picking up a value from the range [0, $max\_dwelling\_time$].

For simulating the mutual influence of a set of trajectories, we use Brinkhoff's traffic model for outdoor movements and extend it for indoor movements. In Brinkhoff's generator [3], each edge of a road network is associated with several parameters, e.g., max_capacity, max_speed, and the number of objects, which compose a traffic model. In accordance to the traffic model, the movement speed goes down if the number of objects on a street exceeds the capacity. For indoor spaces, we determine the speed of a partition (e.g., a room) by the number of its associated objects and the total length of door-to-door edges (e.g., the green one exemplified in Figure 4) within the partition. Then, within a partition, an object moves slowly if its corresponding crowd density is high.

IV. Conclusion and Future Works

UrbanGen is the first trajectory generator that integrates the in- and outdoor movement. Throughout the demonstration, we show how an integrated in- and outdoor topology is constructed and how trajectories can be generated conforming such topology constraints. We also demonstrate how the trajectory can be parameterized and visualized. In the future, a web-based interface like [5] will be implemented such that the outdoor area can be specified and selected by users on a map.

References

[1] X. Xie, H. Lu, and T. B. Pedersen. Efficient distance-aware query evaluation on indoor moving objects. ICDE 2013.

[2] C. S. Jensen, H. Lu, and B. Yang. Graph Model Based Indoor Tracking. MDM 2009

[3] T. Brinkhoff. A Framework for Generating Network-based Moving Objects. GeoInformatica, 6(2), 2002

[4] C. Düntgen, T. Behr, and R. H. Güting. BerlinMOD: a Benchmark for Moving Object Databases. VLDB Journal, 18(6), 2009.

[5] M. F. Mokbel, L. Alarabi, J. Bao, A. Eldawy, et al. MNTG: an extensible web-based traffic generator. SSTD 2013

[6] C. Huang, P. Jin, H. Wang, N. Wang, et al. IndoorSTG: A Flexible Tool to Generate Trajectory Data for Indoor Moving Objects. MDM 2013

[7] H. Li, H. Lu, X. Chen, G. Chen, et al. Vita: A Versatile Toolkit for Generating Indoor Mobility Data for Real-World Buildings. VLDB 2016

[8] S. Hussein, H. Lu, and T. B. Pedersen: Reasoning about RFID-tracked Moving Objects in Symbolic Indoor Spaces. SSDBM 2013

[9] M. Boysen, C. Haas, H. Lu, and X. Xie, A. Pilvinyte. Constructing indoor navigation systems from digital building information. ICDE 2014.

[10] D. Xu, P. Jin, X. Zhang, J. Du, et al. Extracting Indoor Spatial Objects from CAD Models: A Database Approach. DASFAA Workshops 2015.