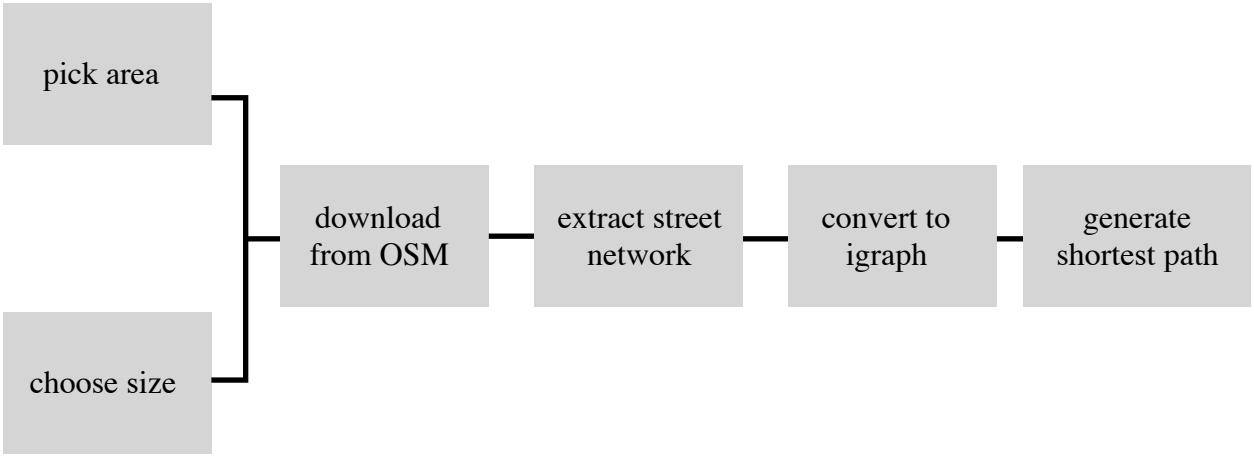
Pedestrian trajectory generator

Candidate Number: WXHX4

I. Introduction

The problem of inferring pedestrian movement in the city has been the main focus in urban planning as the acquired model is considered as an artificial laboratory where planners can test hypotheses that are not easy to explore in the real world (Torrens 2015). With the exponential growth of computational power and the rapid advancement of statistical technique, machine learning has become the major approach to infer pedestrian movement (Lerner 2007). To acquire plausible model, the vast amount of pedestrian trajectory, or training data is essential. While the current pervasiveness of mobile devices enables the collection of mobile location data at the unprecedented scale and granularity, the challenges invariably remain in associating movement path with the reason "why" people moved as the recorded trajectory. To illustrate, the GPS data of those who are commuting to her office and those who are on the way to a nightclub cannot be used together as the training data to model pedestrian movement: the former will most likely choose the shortest path and the latter will likely be a random walk. With the purpose of facilitating machine learning approach toward human movement prediction in the city, this tool aims at being a pipeline for generating ready-to-be-used training dataset. Therefore, assuming that people walking along the shortest route are moving in order merely to reach the goal point, this tool generates the shortest path from an arbitrary start point to a goal point in the selected area.

II. Process



1. Users Input:

1a. Users pick the area where the pedestrian trace data is created by specifying the bounding box (minimum longitude, minimum latitude, maximum longitude, maximum latitude).

ex) area <- (-0.1049, 51.5113, -0.0815, 51.5207)

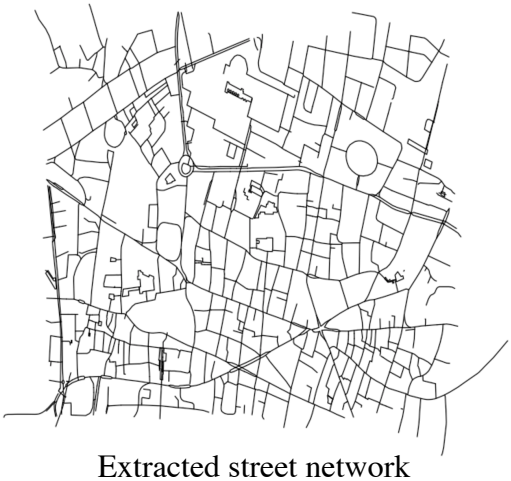
1b. Users choose the number of the data generated.

ex) number <- 400

Selected area

2. Download the geographic data on OpenStreetMap in the selected area by using R package Osmar.

3. As only the street data is needed for the trace generation, the street network is extracted from the downloaded geographical data. It is done by selecting data that is tagged as “highway” which is the main key used for identifying any kind of road, street or path on OpenStreetMap.

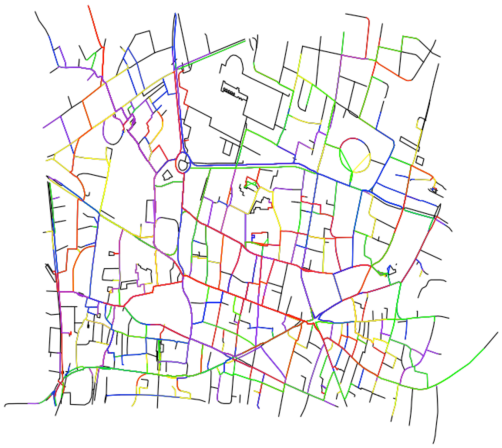


Extracted street network

4. The highway-osmar object is converted into a graph object in order to compute the shortest path. In the graph representation, each node represents an intersection and edge corresponds to a road.

Graph object

5. Using R package Igraph, the shortest path between randomly specified starting nodes and goal nodes are computed. Various algorithms such as Dijkstra, Bellman-Ford and Johnson can be selected. By default, Igraph attempts to choose the fastest suitable algorithm.



Generated training data. Different colors are introduced only for distinguishing each path.

III. Summary

The output pedestrian trajectory data can be downloaded as a text file, consisting of nodes that artificial pedestrians selected to walk on. The associated street network can also be downloaded as CSV in the form of an adjacent matrix, with each element indicating whether pairs of nodes are connected. This API produces the simple but useful dataset for pedestrian movement prediction based on machine learning approach, enabling users to specify an arbitrary area for data generation.

Reference

Torrens, P. M. (2015) Slipstreaming human geosimulation in virtual geographic environments. *Ann. GIS 2015*, 325–344.

Lerner, A., & Chrysanthou, Y., & Lischinski, D. (2007) Crowds by Example. *Computer Graphics Forum* **26**, 655-664.

Manuel J. A. E., & Schlesinger, T. (2012) osmar: OpenStreetMap and R. *R Journal*.

Csardi. G., & Nepusz T. (2006) The igraph software package for complex network research, *InterJournal, Complex Systems* **1695**. http://igraph.sf.net

Lin, Z., & Feygin, S. A. (2016) Simulating Human-like Navigation in Urban Transportation Environments with Multi-agent Deep (Inverse) Reinforcement Learning.