UNIVERSITY OF PIREAUS

DEPARTMENT OF INFORMATICS
POST GRADUATE STUDIES PROGRAM
ADVANCED INFORMATION SYSTEMS

Hermoupolis : A Trajectory Generator for Simulating Generalized Mobility Patterns

MSc Thesis
by
Ntrigkogias Christos

Supervisor : Lecturer Pelekis Nikos

Piraeus Oct 2013

Three - member Inquiry Committee

(Signature)	(Signature)	(Signature)
Pelekis Nikos Lecturer	Yannis Theodoridis Professor	Ioannis Siskos Professor

Dedicated to

my beloved wife Viki
my son Vasilio
my daughter Glykeria
my son Dimitrio

- Nothing less than a big "THANK YOU"

Disclaimer

This thesis has been prepared as part requirement for acquiring the MSc Degree in Advanced Informatics System at University of Piraeus in Greece.

This thesis has been constructed under the supervision of Lecturer Nikos Pelekis and can be freely distributed to examiners or everyone else that find it helpful without any permission of the author.

Author: Christos V. Ntrigkogias Piraeus Oct 2013

Abstract

The real-world process of generating a large spatio-temporal data set performs a very challenging problem. From the one hand it is a very expensive process, requiring sometimes state of the art software tools and very complicated hardware (sensors, servers, GPS infrastructure etc.);while from the other hand the recorded trajectories sometimes cannot represent any special traffic or movement patterns since they refer to specific vehicles or people that cannot be easily integrated to groups. thus there seems to be a serious need for large amounts of real life mobility data To achieve this, a big number of moving object generators have been developed, in order to help even people that are not familiarized with the virtual creation of such objects, to evaluate their performance, in general, through their mobility patterns.

This thesis tries to compare the previous work on data generators, analyze them and propose a modern data generator that among its other common features can generate "enriched" (semantic) trajectories which are derived from a given general semantic scenario and follow real life mobility patterns.

Περίληψη

Η δυνατότητα του πραγματικού κόσμου να παράγει μεγάλα σύνολα χωρο χρονικών δεδομένων αποτελεί ένα πολύ δύσκολο ζήτημα.Από τη μία πλευρά είναι μια πολύ δαπανηρή διαδικασία , απαιτώντας ενίοτε λογισμικά υπερσύγχρονα και πολύ περίπλοκο υλικό (αισθητήρες , servers , GPS, υποδομές κλπ.)? , Ενώ από την άλλη πλευρά, οι τροχιές που καταγράφονται μερικές φορές δεν μπορεί να αντιπροσωπεύουν ειδικά μοτίβα κυκλοφορίας ή κίνησης , δεδομένου ότι αναφέρονται σε συγκεκριμένα οχήματα ή άτομα που δεν μπορούν εύκολα να ενσωματωθούν σε ομάδες . Έτσι φαίνεται να υπάρχει μια σοβαρή ανάγκη για μεγάλες ποσότητες των πραγματικών δεδομένων της πραγματικής κίνησης. Για να επιτευχθεί αυτό , ένας μεγάλος αριθμός γεννητριών κινούμενων αντικειμένων έχουν αναπτυχθεί , προκειμένου να βοηθήσουν ακόμη και άτομα που δεν είναι εξοικειωμένα με την εικονική δημιουργία τέτοιων αντικειμένων , με σκοπό να καταφέρουν αξιολογούν τις επιδόσεις τους , σε γενικές γραμμές , μέσω των προτύπων κινητικότητας τους.

Η εργασία αυτή προσπαθεί να συγκρίνει την προηγούμενη εργασία στις γεννήτριες των σημασιολογικών δεδομένων , να τις αναλύσει και να προτείνει μια σύγχρονη γεννήτρια δεδομένων που ανάμεσα στα άλλα κοινά χαρακτηριστικά της μπορεί να δημιουργήσει «εμπλουτισμένες» (semantic) τροχιές οι οποίες προέρχονται από ένα γενικό σημασιολογικό σενάριο και οι οποίες ακολουθούν πραγματικά πρότυπα κινητικότητας.

Acknowledgements

This thesis has been a big challenge for me. When I first involved with it I did not fully understand my position. I have been the supervisor many times before in my work but until now I did not have the opportunity to be the low level programmer and to have to implement something on my own. They are completely different things.

Firstly, I would like to thank my supervisor, Lecturer Nikos Pelekis for his excellent guidance and the continued support during our cooperation on the first hand and his endless patience shown in all those difficult situations I passed through on the other hand.

I would also like the other members of the lab Panagiotis Tambakis, Stelios Sideridis, Despoina Kopanaki and Professor Yannis Theodoridis for their continued support and encouragement.

Last but not least I would like to thank my beloved family. My wonderful wife Viki for her endless understanding and her encouragement and my little kids for all the moments they needed me and I was not there..

Thank you again

TABLE OF CONTENTS

1 Introduction 11

2 Related Work 13

3 Requirement gathering and analysis 23
3.1 Development Process model 23
3.2 Problem Formulation 24

4 System Design 27
4.1 Raw trajectory τ 27
4.2 Raw sub-trajectory τ' 27
4.3 Semantic Trajectory τ_{sem} 28
4.4 Network 28
4.5 POIs Db table 29
4.6 Moving Objects 29
4.7 External Impacts (External Objects) 29

5 Implementation 31
5.1 Brinkhoff generator analysis 31
5.2 Our Proposed Algorithm 33

6 Hermoupolis Framework 38
6.1 Input elements analysis 38
6.1.1 Generator's network 38
6.1.2 DB Network Creation - POIs 44
6.1.3 Semantic Database 45
6.2 Computing the motion 52
6.2.1 Moving Objects creation. 52
6.2.2 STOP episode 52
6.2.3 MOVE episode 53
6.3 The Report 53

7 Validation Study 56
7.1 The Input 56
7.2 The Computation of motion 63
7.3 The Report 66

8 Conclusion – Feature Work 69

References 70

FIGURE TABLE

Figure 1. Movement using GSTD....13
Figure 2. Movement using extended GSTD....14
Figure 3. Movement using C4C generator....15
Figure 4. Movement using G-TERD....15
Figure 5 . Oporto generator....16
Figure 6. Brinkhoff generator....17
Figure 7. Conversion of simple, plain network data into a complete description (SUMO generator)....18
Figure 8. Node Selection Strategies using BERLIN- MOD....19
Figure 9. Validity of ST–ACTS in terms of TACs....20
Figure 10. MWGen - Experimental results....21
Figure 11. Minnesota data generator....22
Figure 12. Waterfall model....23
Figure 13. BRINKHOFF generator in action....32
Figure 14.Quantum GIS editing Attiki shapefile....39
Figure 15. ShapeNetworkFileManager modified methods used for converting shapefiles in network files (.node,.edge,.info)....39
Figure 16. ShowNetworkMap Class Diagram....41
Figure 17.LoadDrawables Class Diagram....42
Figure 18.ShowMap Class Diagram....44
Figure 19. Georaptor - a spatial extension of sql developer....45
Figure 20.Semantic Db output that will be used as input for generator (part 1)47
Figure 21.Semantic Db output that will be used as input for generator (part 2)48
Figure 22. Semantic Db output that will be used as input for generator (part 3) 50
Figure 23. Loading Scenario File....50
Figure 24. Classes that store the input semantic Scenario....51
Figure 25. Semantic Trajectories Report Table....54
Figure 26.Visualization of generated objects....55
Figure 27.Semantic File....58
Figure 28.Attika_Network....59
Figure 29.Visualization of the database table containing the POIs in Attiki (part 1)....60
Figure 30. Visualization of the database table containing the POIs in Attiki (part 2)....61
Figure 31. Visualization of the database table containing the POIs in Attiki (part 3)....62
Figure 32. Visualization of the database table that contains the POIs in Attiki (part 4)....63
Figure 33. Hermoupolis after loading the scenario file....64
Figure 34.Hermoupolis in time 79....65

Figure 35.Hermoupolis after generation.....65
Figure 36.Semantic output for object id 7 (part1).....66
Figure 37.Semantic output for object id 7 (part2).....67
Figure 38.Semantic output for object id 7 (part 3).....68

1 Introduction

The real-world process of generating a large spatiotemporal data set performs a very challenging problem. From the one hand it is a very expensive process, requiring sometimes state of the art software tools and very complicated hardware (sensors, servers, GPS infrastructure etc.);while from the other hand the recorded trajectories sometimes cannot represent any special traffic or movement patterns since they refer to specific vehicles or people that cannot be easily integrated to groups. it is undisputed that the pros deriving from the GPS equipped mobile devices technology, such as smart phones, GPS navigation devices, tablets etc. along with the vast spread of those devices and the development of the appropriate techniques for storing, processing, querying, and mining such data, has led to the generation of monstrous amounts of GPS-like data. Nevertheless, converting this kind of data into a meaningful form in terms of mobility is not an easy task. During this procedure a lot of spatiotemporal algorithms and data structures are used. To achieve this, simple and comprehensible methods of performance evaluation, must be used . Thus huge amounts of real life mobility data are necessary but unfortunately they cannot be easily obtained. The use of these techniques is widespread, since they accommodate the study of the movement of objects in the space over time. Therefore, the evaluation of spatiotemporal algorithms is required because through them one can understand, and hence resolve problems created by the movement of objects such as traffic problems, transportation of groups of endangered animals etc. As a result, a big number of moving object generators have been developed, in order to help even people that are not familiarized with the virtual creation of such objects, to evaluate their performance, in general, through their mobility patterns.

The generators proposed so far in the literature are divided in two categories: *microscopic* and *macroscopic*. Microscopic generators deal with single vehicles. The models produced by the microscopic generators are mostly used to evaluate traffic efficiency regarding speed and travel time. They basically focus on simulating traffic signal regulation, route control and traffic condition estimation. Each of these generators uses its own control strategies and algorithms. On the other hand, macroscopic generators deal with the traffic flow (and not with single vehicles) at a higher abstraction level than the microscopic generators . They are employed to evaluate traffic flow as a whole without consideration of the characteristics and features of individual vehicles in the traffic. The proposed data generator, even though it holds a lot of the characteristics that the microscopic generators hold, it is considered to be a macroscopic generator.

An interesting category of algorithms and techniques performed on mobility data aims at the behavioral analysis of people moving in an urban environment and performing their daily activities. The outcome of such an analysis could assist in “smart” and efficient city planning and thus having a great impact in the improvement of the everyday life of people living in an urban environment. Unfortunately, real raw mobility data lack essential behavioral info. Moreover, real mobility data that incorporate behavioral information are hard to acquire. In most cases,

such datasets are small in size and cannot be considered as “big”. Taking into account all of the above, it can be inferred that the evaluation of the algorithms and techniques mentioned above is not a straightforward task. Furthermore, the same applies for the evaluation of the efficiency and scalability of a data management infrastructure, designed for this kind of data.

2 Related Work

One of the first moving object generators is GSTD[1] (Theodoridis et al. 1999),which creates moving points and moving rectangles in one area without network constraints.

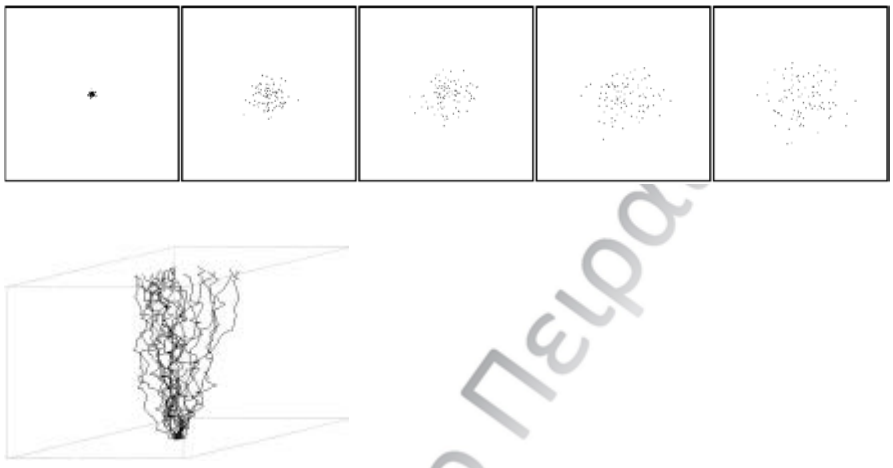
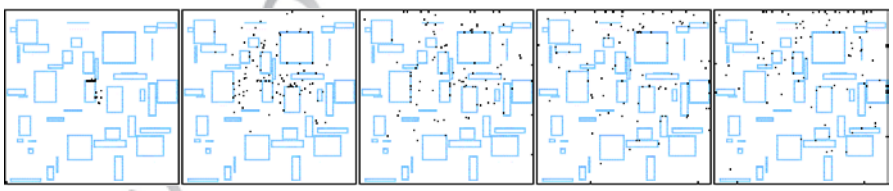


Figure 1. Movement using GSTD

During the time the objects generated can change positions and size.

A newer version of this generator [2] (Pfoser and Theodoridis 2003), gives the user more features, allowing greater agility.



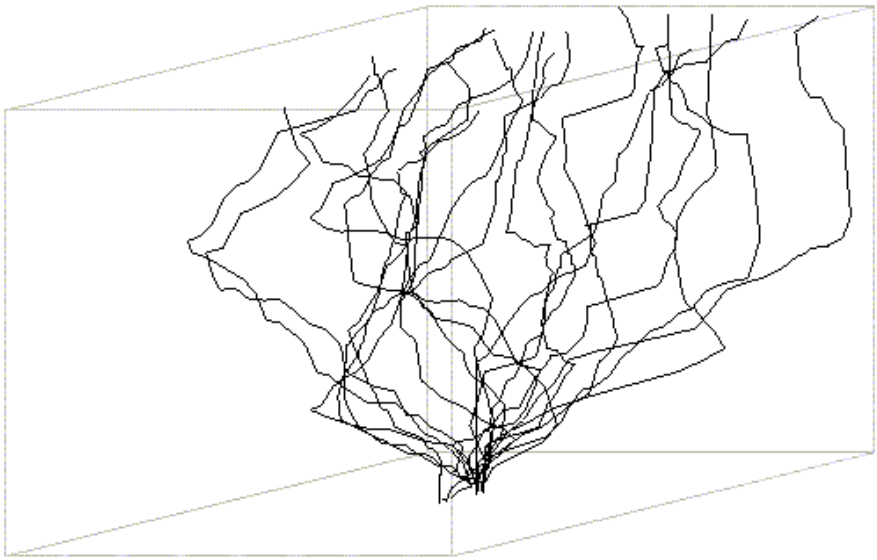


Figure 2. Movement using extended GSTD

An improvement of this generator is that the parallelogram generated acts as an obstacle for other objects, so the movement is not completely independent.
The purpose of another generator was to evaluate the algorithm presented in (Pelekis et al. 2009). This generator, which is based in GSTD presented above, sets a number of points and generates moving objects that during their trajectory must pass through those points (a kind of stop). During their motion, the objects are divided into groups, since all objects must necessarily pass through the points that have been set, so it can then evaluate the clustering algorithm is the purpose of this work. These features help to partially display real situations - routes. Given that, in the real world there are objects moving on predefined tracks (cars on roads, trains on tracks, etc.), the objects created by the generator shall be unsuitable for measuring performance of algorithms on moving objects over a network.

Another embodiment of the GSTD generator is CENTRE (Cellular Network Trajectories Reconstruction Environment) [3] (Giannotti et al. 2005), which examines movements of objects in mobile networks, using data from cell phone providers and simulating larger numbers of objects. It uses some characteristics of GSTD, but the main difference is that groups can be defined according to their special characteristics (e.g. direction, speed, agility and obstacles). For example, cars can be obstructed by rectangular obstacles (buildings) while people not. So people have arbitrary motion in space, but with less speed.

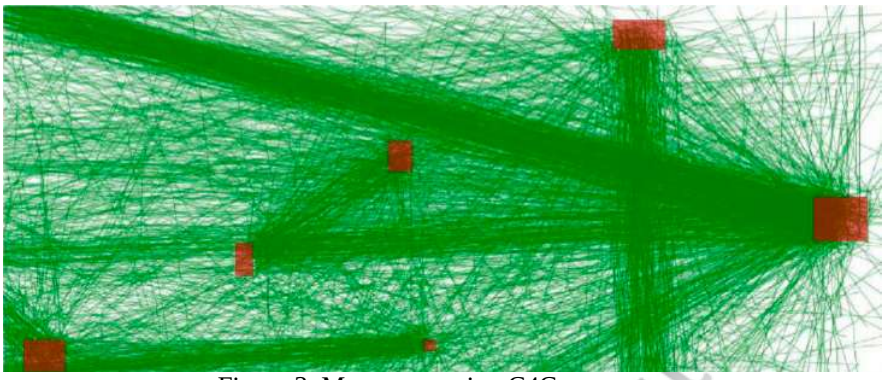


Figure 3. Movement using C4C generator

The G-TERD (Tzouramanis et al. 2006) [4] is a generator that produces large sets of objects that constantly move and change sizes in two-dimensional regions. These items are not merely points but define areas which have shape, size and color for the display of the different properties that are between them. The user has the ability to modify some of their parameters, such as maximum speed, angle of movement, interaction with other objects, the initial position of the object, etc. This generator could easily be used to monitor objects. But even here the movement is unrestricted, so it doesn't serve for observing objects moving over a network.

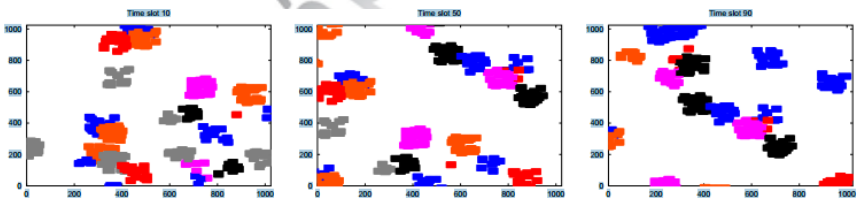


Figure 4. Movement using G-TERD

Oporto (Saglio and Moreira 2001)[5] is another data generator which simulates the scenario with fishing boat. Like previous generators, Oporto offers many features to the user. Nevertheless, the motion of object sis arbitrary in two-dimensional space, which does not allow us to study the motion of objects over a network.

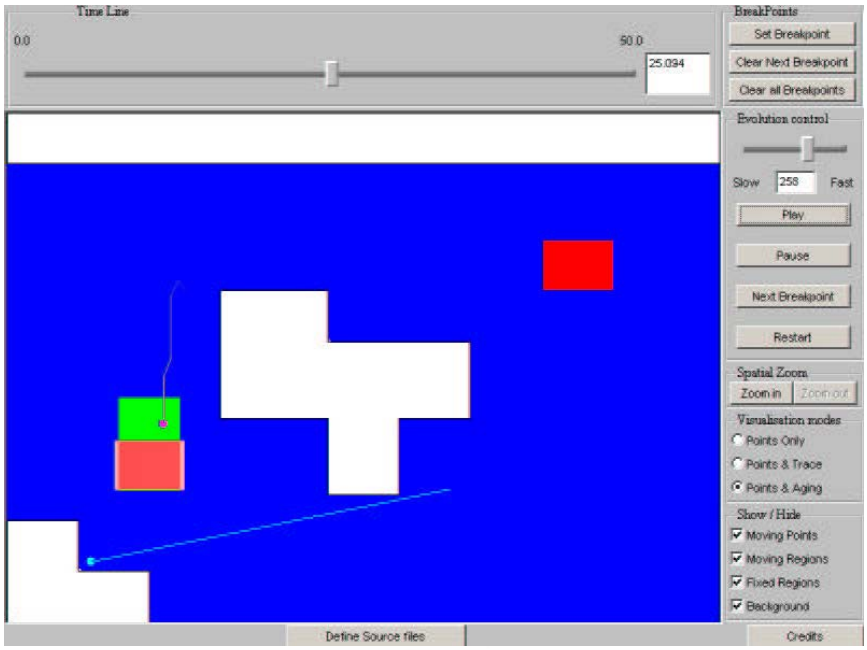


Figure 5 . Oporto generator

An early attempt to create a network -constraint data generator was made by Dr. Thomas Brinkhoff (Brinkhoff et al. 2002)[6]. He created an open-source generator, with graphical interface for objects moving over a network. During the simulation, new objects are created in random locations on the grid and disappear when (and if) they reach their destination. These objects are separated at their creation in groups (can be displayed in different colors) to highlight different properties may have. The speed and the path followed by each object depend on the group to which it belongs, but also from the point located at all times. This is because each edge of the network has a maximum capacity, which if exceeded, affects the speed of moving objects in it. The basic functions, such as the number of moving objects or timing simulation of the generator can be regulated by a limited set of parameters that are easily changed, not necessarily from experienced users. But to make some other major changes, such as the speed of the objects or the initial and final position, the user will need to be more specific, since you will need to refer to the code. The original generator intends to represent specific objects and paths not monitor specific objects for some time, which does not allow proper measurement network performance.

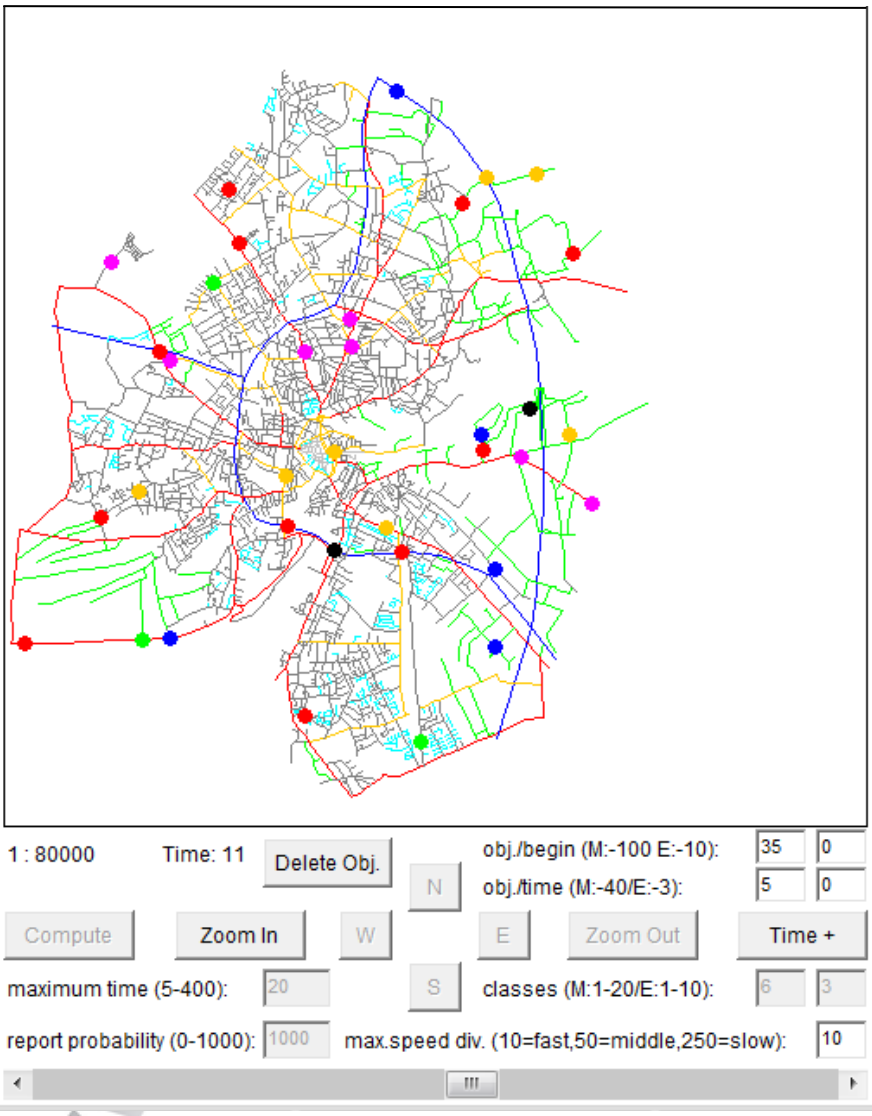


Figure 6. Brinkhoff generator

Another example of moving object simulator over a network is SUMO (Simulation of Urban Mobility) (Krajzewicz et al. 2002)[7], an open source generator that simulates the movements of private and public transport. In the simulations, vehicle movements are represented in a network that does not allow conflicts; the streets are two-way, etc., thus completing the picture of a real network. An addi-

tional feature of the generator SUMO is that the behavior of moving objects generated is based on the proper interaction between them. So the movement of each object that is created depends on the motion of other objects in the network design. This, however, limits quite the scholar, since if it wanted to simulate some extreme movements in the network will be quite difficult, since the movements of other objects on the network will adjust on it. This feature makes the network not function as a real network.

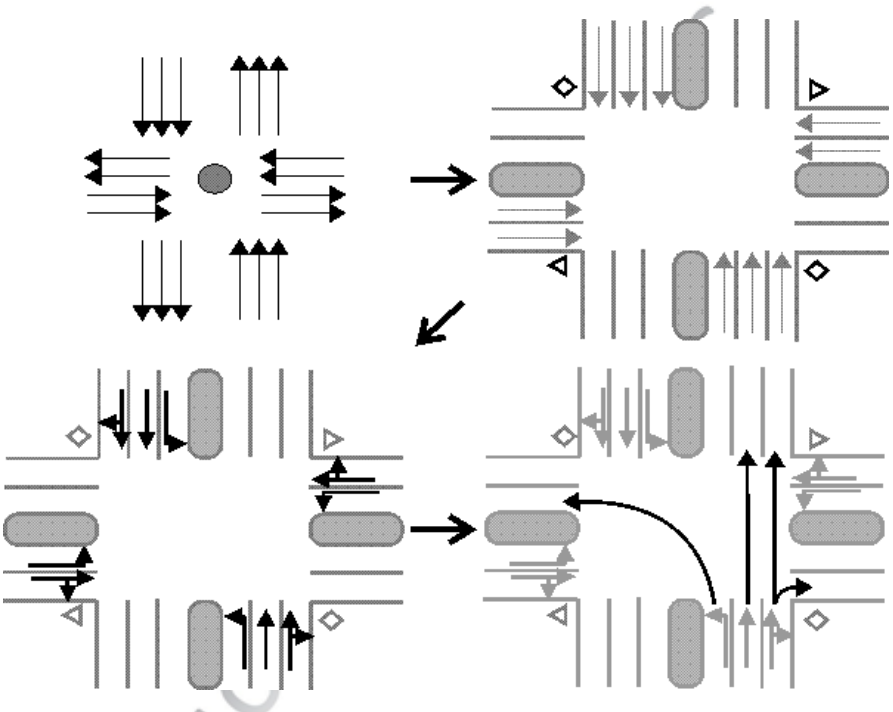


Figure 7. Conversion of simple, plain network data into a complete description (SUMO generator)

The most recent study is (Düntgen et al. 2008)[10], which simulates movements of objects over the network of the city of Berlin for a given period of time and capture their positions. Instead of implementing a standalone dedicated generator software, the infrastructure of Secondo MOD [11] is utilized. The traffic in this generator, is long term, i.e. simulates movements of objects for long periods, for example a day or more, in contrast to the other generators which have shorter time strokes.

Figure 8. Node Selection Strategies using BERLIN- MOD

Besides generators studying the natural movement of objects on maps, which are the majority, there is a different generator that is studying social and geo-demographic data. The ST-ACTS (Gidofalvi and Pedersen 2006)[12] considers how objects move through space because they have to perform a particular activity, which can only be performed at a particular point, both in space and in time. Through the experiments made with this generator we study the distribution of activities in space that can help develop appropriate spatiotemporal data management systems and data mining techniques.

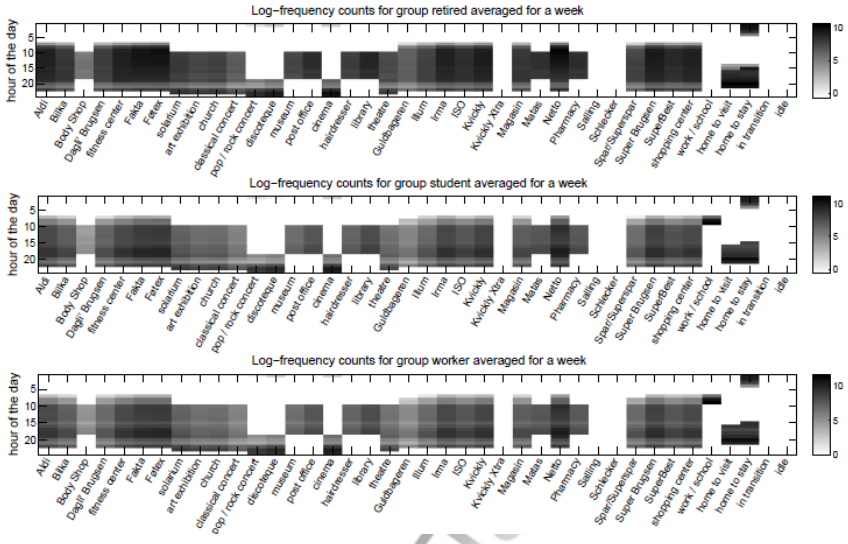


Figure 9. Validity of ST–ACTS in terms of TACs

Another data generator, which attempts to combine the generation of moving objects traveling through different environments and with multiple transportation modes, is MWGen (Xu and Güting, 2012) [13]. This generator basically tries to generate moving objects in different environments where the precise location in each environment and transportation modes are managed.

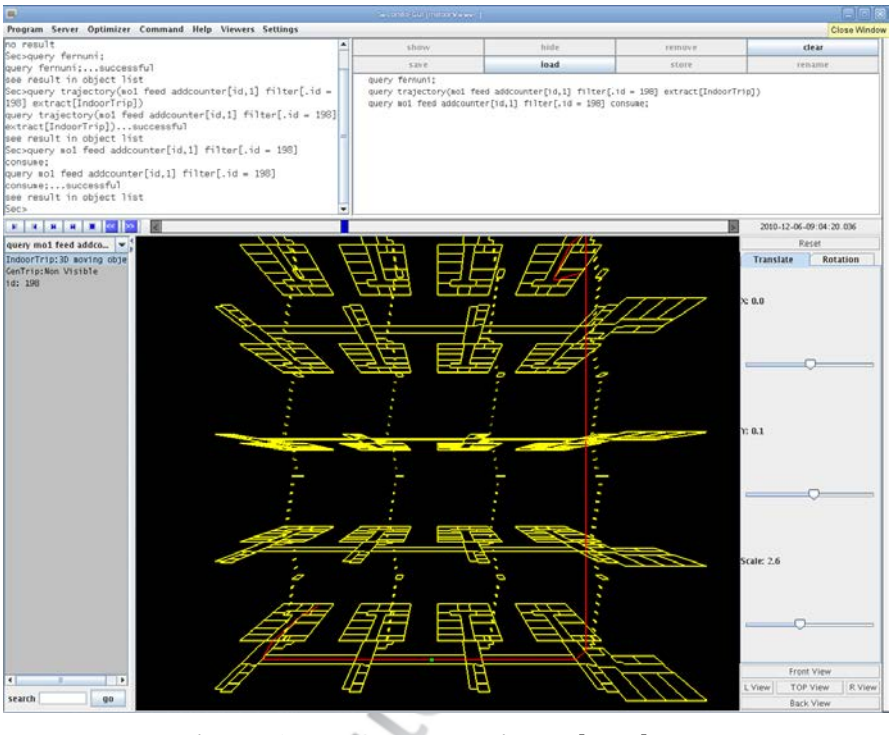


Figure 10. MWGen - Experimental results

The GAMMA framework (Hu and Lee, 2005) [14] represents moving object behavior as trajectories in the spatiotemporal. The generation process is considered as an optimization problem and is solved by the utilization of a genetic algorithm. In fact this framework is a generator "by example", which means that it receives as input a set of trajectories and tries to generate similar "activity" trajectories that enclose real–life activity patterns. The generated trajectories will be similar to the input trajectories. So in order for the generator to simulate spatiotemporal activities of an entire population, a representative sample is needed.

MATSim is a micro simulator that provides a framework for fast, dynamic, large-scale agent-based transport simulations, such as demand-modeling, agent-based mobility-simulation (traffic flow simulation), re-planning, a controller to iteratively run simulations as well as methods to analyze the output generated. It supports large scenarios with several millions agents or network with hundreds of thousands of streets. Throughout the simulation, it gathers numerous key values from the simulation and outputs them in order to provide the user with a quick overview of the status of the simulation.

An entirely different approach is the MinnesotaTraffic Generator which is not actually a data generator but it's a web application that acts as an interface layer on top of the Brinkhoff or BerlinMOD generators. The main contribution of this tool is that the end user avoids the installation and complicated configuration steps

necessary to get either Thomas-Brinkhoff or BerlinMod both up and running, and +\he is able to work on a user specified region.

Figure 11. Minnesota data generator

3 Requirement gathering and analysis

3.1 Development Process model

There are many software development methods that are employed during development process of software, these approaches are also referred as "Software Development Process Models" (e.g. *Waterfall model, incremental model, V-model, iterative model,* etc.). Each model follows its own life cycle in order to ensure success. For the proposed data generator we partially followed the Waterfall model. This model is a popular version of the systems development life cycle model for software engineering. Often considered the first approach to the systems development life cycle, the waterfall model describes a linear and sequential development method. Development in a Waterfall model basis has separate tasks for each phase of development like a waterfall on a mountain. Once the water has flowed over the edge of the crag and has begun its journey down, it cannot turn back. It looks like waterfall development. Once a part of development process is completed, the development proceeds to the next part and there is no return.

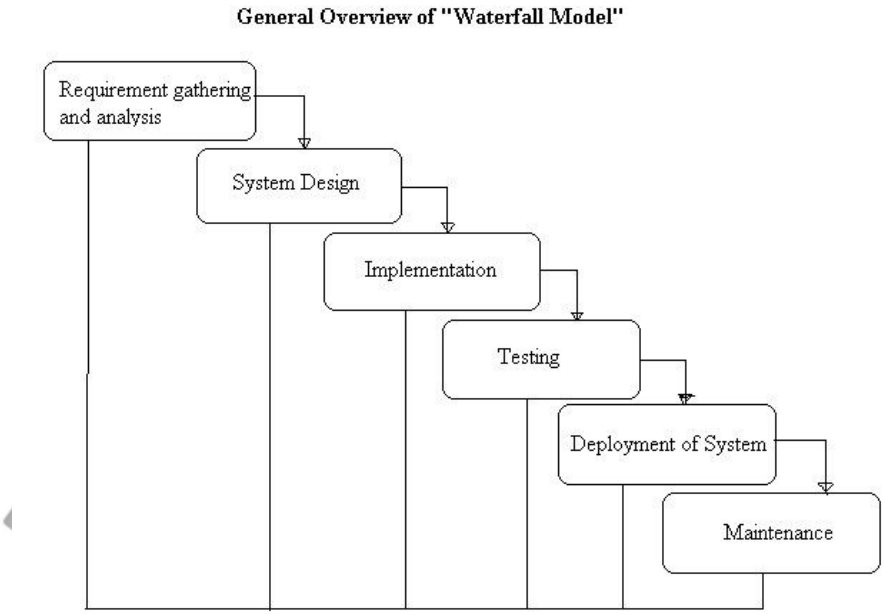


Figure 12. Waterfall model

3.2 Problem Formulation

Generally, moving object data generators can be categorized with respect to the characteristics they hold. These characteristics are:

- *User Defined Input.* The generator takes as input movement features, such as speed, agility, direction etc.
- *Obstacle Areas.* Another important feature that some generators hold is whether they can represent obstacles which have to be avoided by the moving objects.
- *Crossing Points*. Many generators generate moving objects which, during their trajectory, must pass through some predetermined points (Crossing Points).
- *Civil Environment Simulation.* How realistic can a generator simulate civil environment conditions. This can be done by its ability to handle such data (Land use, Points of Interest etc).
- *Network-Based Movement.* Generated moving objects can either move freely in the space or move according to a network. Moreover, several generators not only create objects that their movement is constrained to a subjacent network but also take into account supplementary attributes of the network, such as the capacity, the maximum speed etc. There are also data generators that obligate the generated moving objects to move accordingly to additional network constraints such as maximum speed ,max network capacity etc.
- *Moving Objects Interaction.* The case in which each generated moving object interacts with others resulting in its movement parameters alternation.
- *Moving Objects Categorization.* Another data generator feature is their ability to categorize generated moving objects in groups that are characterized by dissimilar mobility characteristics and as a result conform to different standards.
- *Semantics Info*. Rarely, data generators have the ability to handle with information about the general activity of the generated moving object. Thus, they can support multilevel analysis of their behavior.

- *Generation-by-example*. Another not so common feature of recent data generators is their capability of taking as input sets of real life trajectories and generating objects that follow the mobility patterns that are given as input.
- *Long Term Generation.* An extra feature that certain generators own is the constant generation of moving object during long time periods. It can be lead to reliable conclusions if the simulation refers to behavior analysis of big datasets lasting long time periods.
- *Synchronized Output* .Given a set of semantic patterns as input some generators are able of producing logs, after completing the generation, that visualize not only the semantic trajectories but also the raw trajectories.

Moving Object Generator	User Defined Input.	Obstacle Areas.	Crossing Points	Civil Environment Simulation	Network-Based Movement	Moving Objects Interaction	Moving Objects Categorization	Semantics Info	Generation-by-example
GSTD family	√	√	√			√	√		
CENTRE	√	√					√		
Brinkhoff	√				√	√	√		
SUMO	√				√	√	√		
GAMMA								√	√
BerlinMOD			√	√	√	√		√	
MWGen					√		√		
Hermoupolis			√	√	√	√	√	√	

Table 1.Data Generators by Feature Comparison

The goal of building a moving object generator that complies with previous stated algorithms could be achieved if it could incorporate as much as possible of the above characteristics. To the best of our knowledge, such a generator has not been proposed in the literature. Such a synthesizer could be produced by combining a moving object data generator and an activity generator, which will produce the synchronized output (both network constrained trajectories and "semantically" enriched trajectories).

4 System Design

Informally, the proposed data generator will receive as input a set of mobility behavior “profiles”, formally defined in section, a database table containing points of interest (POIs), and the underlying road network (N). Our approach utilizes the Brinkhoff generator and makes it capable of generating moving objects which during their trajectory (trip) have to pass through some points of interest (areas, buildings etc).These predefined points are given by the set of mobility profiles. By adding this functionality, the conception of generating objects enriched with activity information is being embedded in the generator. As a result of that, both the raw network constrained trajectories and the “semantically” enriched trajectories generated, are synchronized. Furthermore, we tried to insert the temporal element into the generator, since the given mobility profiles contain such info, by extending it in order to ensure that each moving object will stop to predetermined point for the current period of time.

In order to become clear what our proposed generator should do we must give additional definitions which further analyzing and explaining the input information.
More particularly,

4.1 Raw trajectory τ

A *raw trajectory* of a moving object is a triple where is the unique identifier of the moving object, is the unique identifier of the trajectory and is a sequence of spatiotemporal points with . At this point, it has to be clarified that between consecutive points and is assumed.

4.2 Raw sub-trajectory τ'

A *raw sub-trajectory* is a portion of a raw trajectory and it is defined by the tuple where , and are the unique identifiers of the moving object, trajectory and sub-trajectory ly. is the part of the trajectory between two timestamps and with .

4.3 Semantic Trajectory τ_{sem}

Having defined a *raw trajectory* and a *raw sub-trajectory* we can proceed to the definition of the semantic trajectory. Informally, a *semantic trajectory* can be defined as a sequence of successive STOP and MOVE episodes.Formally, a STOP or MOVE corresponds to a sub-trajectory. A sub-trajectory is considered as a STOP if and only if its spatial (temporal) projection obeys a predefined spatial (temporal, resp.) constraint C_{space} (C_{time}, resp.). In any other case is considered a MOVE. An episode also corresponds to a sub-trajectory and is defined as a tuple (, , , ,), where defines whether the episode is a STOP/MOVE. MBB corresponds to and is a tuple (, ,) where holds the spatial properties and is the bounding rectangle of and [,] holds the temporal properties and is the temporal period in which exists. Furthermore, is a … which incorporate the semantic information. For example, for STOPs it could be the POI and the corresponding activity that the object had there, while for MOVEs could be the road type and the mean of transportation used to traverse the specific road. Finally, ly, is a link to .

Consequently, a *semantic trajectory* of a moving object is defined as a triple where , is the unique identifier of the moving object and the unique identifier of the semantic trajectory of the moving object, respectively, and is a sequence of consecutive (w.r.t time) episodes belonging to the same trajectory .

4.4 Network

Also another very important element, that data generator takes as input, is the underlying road network and its characteristics. So The road network is actually a graph composed of a set of vertices , a set of connecting those vertices. Each edge of the road network is described by a number of attributes like , the maximum capacity of each edge with . Additionally, there exists a function which restricts the maximum speed of by a further limit, if during a time interval the edge usage ( ) is greater than the maximum capacity of . An additional important element of the generator is the set of points of interest of the area that we would like to simulate.

4.5 POIs Db table

The last element that our proposed data generator takes as input is the set of POIsEach point of interest is a tuple that consists of where is the unique identifier of each , is a spatial point denoting its location and is a set of tags stating its underlying utility.

4.6 Moving Objects

How is moving objects being modeled?

Each one belongs to an object class *objClass(obj)* and for each object class a maximum speed *objClassMaxSpeed(objClass)* is defined. Furthermore the speed of an object on an edge *objSpeed(obj,edge,time)* is restricted by the maximum speed of its object class and the maximum speed on the edge.

objSpeed(obj,edge,time) < objClassMaxSpeed(objClass(obj))

objSpeed(obj,edge,time) < edgeMaxSpeed(edge,time)

4.7 External Impacts (External Objects)

In real life every movement is influenced by external events (weather conditions etc) .In order to simulate those events we are trying to introduce the external objects which have grouped by similar characteristics. So they are grouped by:

- *lifetime* We can produce external objects active from the beginning to the end others that are created at *timestamp1* and deleted at *timestamp2 (timestamp1< t timestamp2).*

- *Mobility*. We have static and moving external objects.
- *Shape.* External objects having a static shape and others that change their spatial extension over the time.

The basic properties of such external objects are:

- The position and extension of the object *area(extObj,time)*. The attribute *area* is time dependable.
- The affiliation to an object class *objClass(extObj)*, which describes basic properties like the speed and the lifetime of an external object.
- The speed reduction in the area of an external object is determined by the attribute *decreasingFactor* of the corresponding object class.

Now, a further limit to the maximum speed on an edge can be defined:

edgeMaxSpeed(time,edge) < edgeClassMaxSpeed(edgeClass(edge)) ■ minimum ({decreasingFactor(objClass(extObj)) with loc(edge) Π area(extObj,time) Φ 0})

5 Implementation

5.1 Brinkhoff generator analysis

Brinkhoff generator in its initial state is described below.

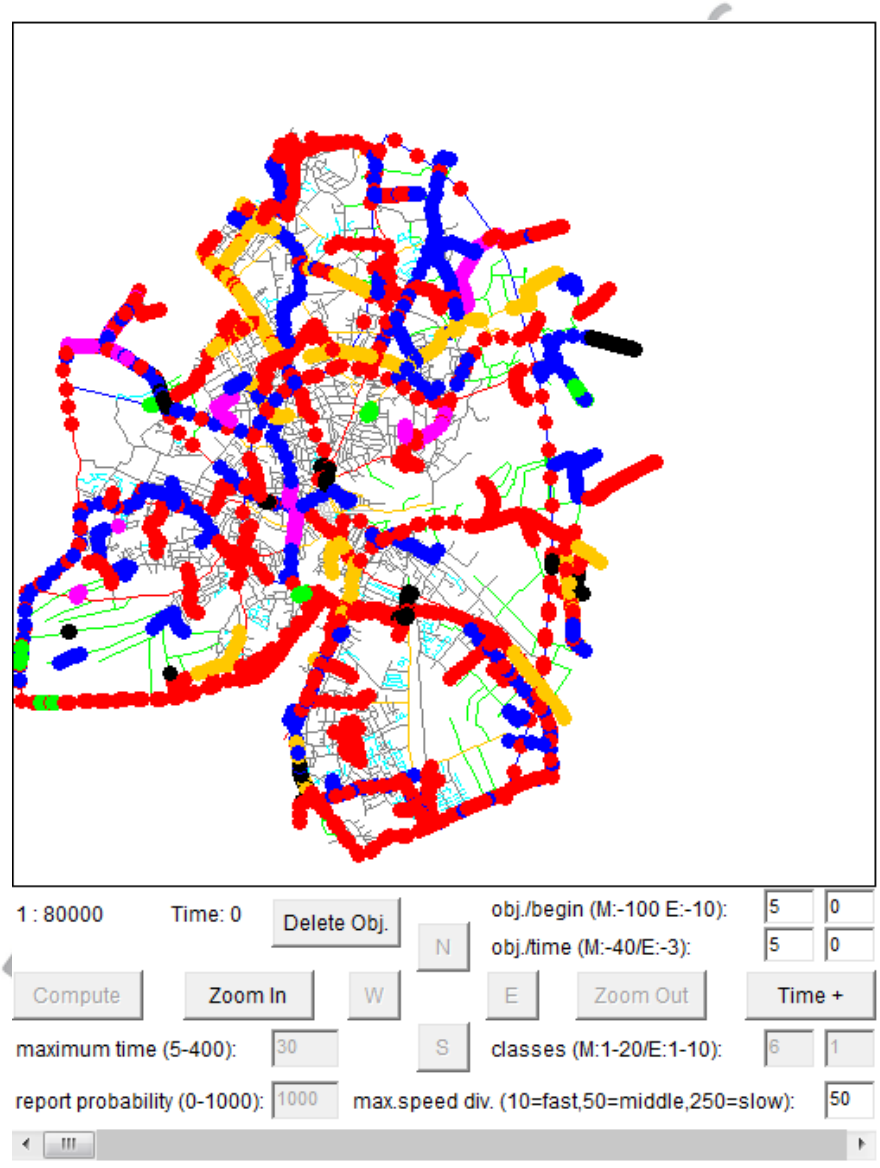
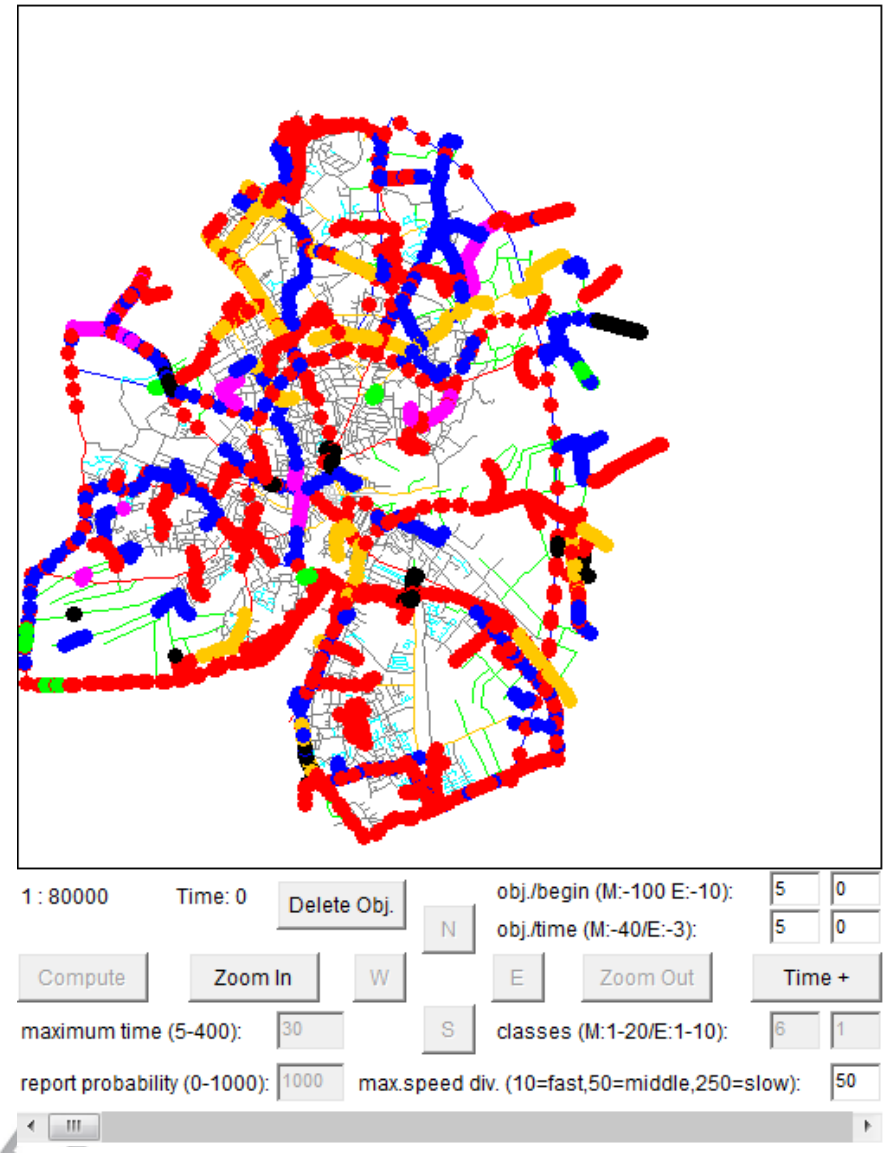


Figure 13. BRINKHOFF generator in action



The moving object generator provided by Thomas Brinkhoff is based on the observation that objects movement follows network pattern (vehicles, people moving in a city even aircrafts flight obeys to network rules although invisible). So, almost no objects can be moved outside of the network. The generator's time model is as follows: We assume that the whole movement period is divided by a number *(m) of* time stamps. At each time stamp, new

moving objects are generated and previous objects are moved or are deleted (if they reach their destination).There are many classes to which moving objects belong and they specify the behavior of each moving object (e.g. max speed).Also each edge contains a number of edges and specifies some of their characteristics (speed limit, capacity). In case the number of objects moving on an edge exceeds its maximum capacity, edge's speed limit will be decreased.

In addition, extra features (external objects) are implemented in order to influence movement (weather conditions etc).Some of them, exist over the total period when others are generated in a time stamp and deleted later. During their influence on a moving object they change its speed according to their class's speed limit.

How many new objects are generated per time stamp or which is the start position or the destination position depends on time. Although a moving object's route is computed once at the beginning during its creation, it will change in the next time stamps if an external object or other moving objects influence it.

As stated in section [3.2] Brinkhoff data generator implements a number of characteristics that make a moving object data generator more efficient.

(i), (ii). Allows user to define movement features such as maximum speed, agility, direction and influence the moving object's movement. So this data generator can not only create objects that their movement is constrained to a subjacent network but also takes into account many other attributes of the network, such as the capacity, the maximum speed etc, that influence created moving object's movement.

(iii). Furthermore this generator create moving objects that can fully interact and affect or not other object's movement.

(iv). A final characteristic that this generator adopts is that its generated objects are divided into groups (classes) with different features and mobility patterns.

5.2 Our Proposed Algorithm

As already stated above, the input of the semantic "aware" generator is a Semantic Mobility Database , the number of objects per "profile" to be generated, a road network , a spatial database containing points of interest , the desired sampling rate of the output during STOP episodes, the max-

imum simulation time and the maximum speed per profile per episode and. The latter two parameters are either given as input by the user or can be derived from the "profiles" that are given as input. The output of the generator is semantic trajectories and the corresponding raw network constrained trajectories.

Briefly, the methodology is as follows. Initially, the generator loads the road network (line 1). Then, for every object it calculates its respective STOP nodes with respect to the STOP episodes of the input, (line 3-7). The Brinkhoff generator for every object computes a position , by using a two-dimensional distribution function, assuming a uniform distribution, or a network distribution function, and then determines the nearest node of the network which plays the role of the .In contrast to that, the current approach in order to compute the STOPs, takes into account the episode's MBB, its semantic annotation and the set of points of interests along with their underlying utility. Actually, the set is stored in a database. Each tuple consists of the fields properly indexed with an R-Tree index on the field and a B-Tree index on the field. In addition to the set, the road network is also stored in a database. Each tuple contains the fields properly indexed with a R-Tree index on the field. Considering the above, the POIs where the overlaps the and the equals the , are returned (line 4-5). Subsequently, the nearest node of those POIs is retrieved from the road network relation where by posing a 1-NN query (line 6). If there are more than one node in the result set then we select randomly one of them (line 7). On the contrary, the Brinkhoff generator's destination node of each object is determined by a separate function than the origin node, getting the starting node and preferred length of the route, which is a user-defined function, as additional parameters. Having determined all the nodes of every object, it's a trivial task to compute the route between two consecutive STOP nodes, which is actually the for every object, by calling the Brinkhoff generator's functionality for routing (lines 11). The routing algorithm used by the generator is actually a variation of the A* algorithm. At this point, since the moving objects are initialized, the STOP nodes for each object are defined and the MOVEs for moving between the consecutive STOP nodes is computed, it's time for the actual simulation to take place. Therefore, the simulation time for the current episode is initialized (line 13) and while it hasn't exceeded the maximum simulation time (line 14), for every object (line 15), if its current position is not a STOP node (line 20) then, if there is a strong deviation of the current speed from the expected speed (e.g. if the car is in a traffic jam) and enough time has passed since the last computation of the route then a different route should be computed (lines 21-24), on any other case continue the predefined routing (line 25). The expected speed is calculated by taking into account the maximum speed of the profile for the specific episode, the maximum allowed speed and the maximum capacity of the edge. If the current position is a STOP node and the moving object hasn't stayed long enough to with respect to the duration of , then the current position is set as the next position (lines 44-45). If the

moving object has stayed long enough to with respect to the duration of then move to the next STOP (lines 47). ++

Algorithm
Input: set of Profiles , number of objects per profile , road network , a set of points of interests , the maximum simulation time , the maximum speed per profile
Output: Synchronized Semantic and GPS Trajectories

```
1   load( )
2   for i in 1..NumOfObjects do
3     for j in 1..              do
4                       overlaps
5                    =
6
7
8
9                       =
10     if j > 1 then
11
12
13          = 0
14    while                     do
15    for i in 1..NumOfObjects do
16     if                      then
17      if                          >       and
18                                    then
```

```
19
20              =
21                 ++
22       else            =
23                 ++
24      else
25       if
26         if                          >       and
27                                    then
28
29              =
30                 ++
31         else             =
32                 ++
33      else
34       if
35         if                          >       and
36                                    then
37
38              =
39                 ++
40         else
41              =
```

```
42                ++
43        else
44         if                                    then
45              =
46                ++
47         else          =
48                ++
49      Report
50          ++
51  for i in 1..NumOfObjects do
52    for j in 1..              do
53    for k in 1..NumOfReportedPos do
54      if
55              =
56      else
57              =
```

6 Hermoupolis Framework

6.1 Input elements analysis

As seen above Brinkhoff generator needs enrichment in order to become competitive. It lacks of many characteristics that a semantic aware generator should have. This is what we have done. Our proposed generator although based in Brinkhoff generator engine, enriches it with many new features and turns it into a very functional semantic "aware" generator. Below is the analysis of elements that are

6.1.1 Generator's network

The proposed generator is able to read the network representation from files or from an Oracle database. In our implementation it needs, two binary files describing the nodes and the edges of the network. In order to use existing files from OSM , tools can be used to transform them into this file format. Currently, two open source tools exist which allow using TIGER/Line Files.

The conversion from shapefiles to the proper format for the network source files is described below.

The shapefiles to be used have been downloaded from open street maps (OSM) have to be properly modified using open source tools (like quantum gis desktop, MapWindow gis etc) in order to contain proper information about predefined region.(in our case Attiki).

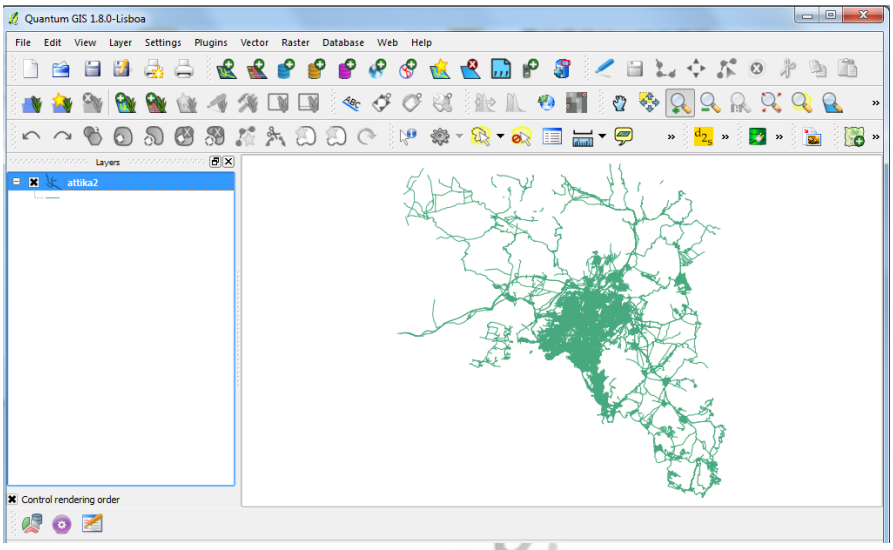


Figure 14.Quantum GIS editing Attiki shapefile

The tool used for converting the shapefiles to proper format is the one available to BRINKHOFF website, called shapeNetworkFileManager. The sw has been written in java and exports two files .node and .edge describing all the nodes and edges for the network that will be constructed, in a simple syntax. In order to achieve more detailed conversion and obtaining more information from the source files (OSM data) that will finally passed to the generator, the sw has been enriched with new methods and some of the old have been modified. Thus, to its last edition it exports not only the old .node and .edge files but also a .info that contains all the initial information needed.

Operations

```
public ShapeNetworkFileManager( )
public void  main( String args[0..*] )
private int  intoX( double longitude, double min, double ext )
private int  intoY( double latitude, double max, double ext )
public void  deleteNetwork( )
public void  readShapeFile( )
public void  setResolution( )
public void  writeNetworkFile( )
```

Figure 15. ShapeNetworkFileManager modified methods used for converting shapefiles in network files (.node,.edge,.info)

Thereafter the generator loads the created network. It has various methods implemented that not only convert the initial x, y coordinates derived from the OSM files to a different format, suitable for the generator, but can also store the original values for future usages. Figure 16 shows the class diagrams of the java packages that are responsible for reading the network files and create the network map that is displayed to generator's graphical user interface.

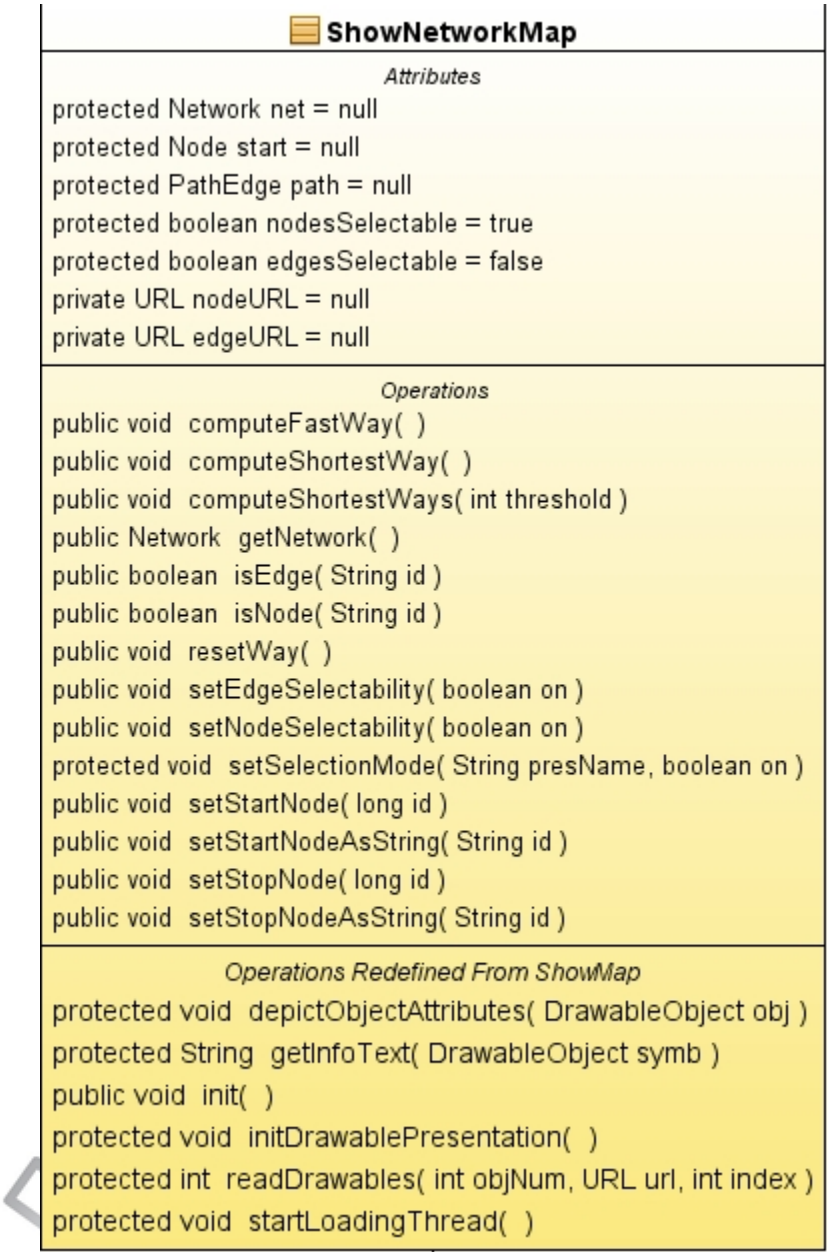


Figure 16. ShowNetworkMap Class Diagram

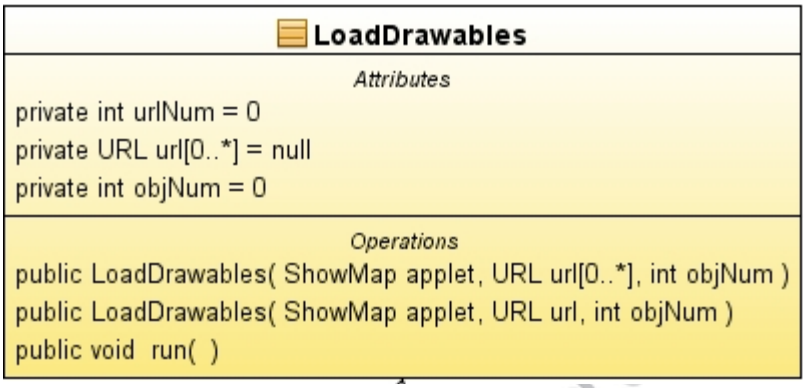


Figure 17.LoadDrawables Class Diagram

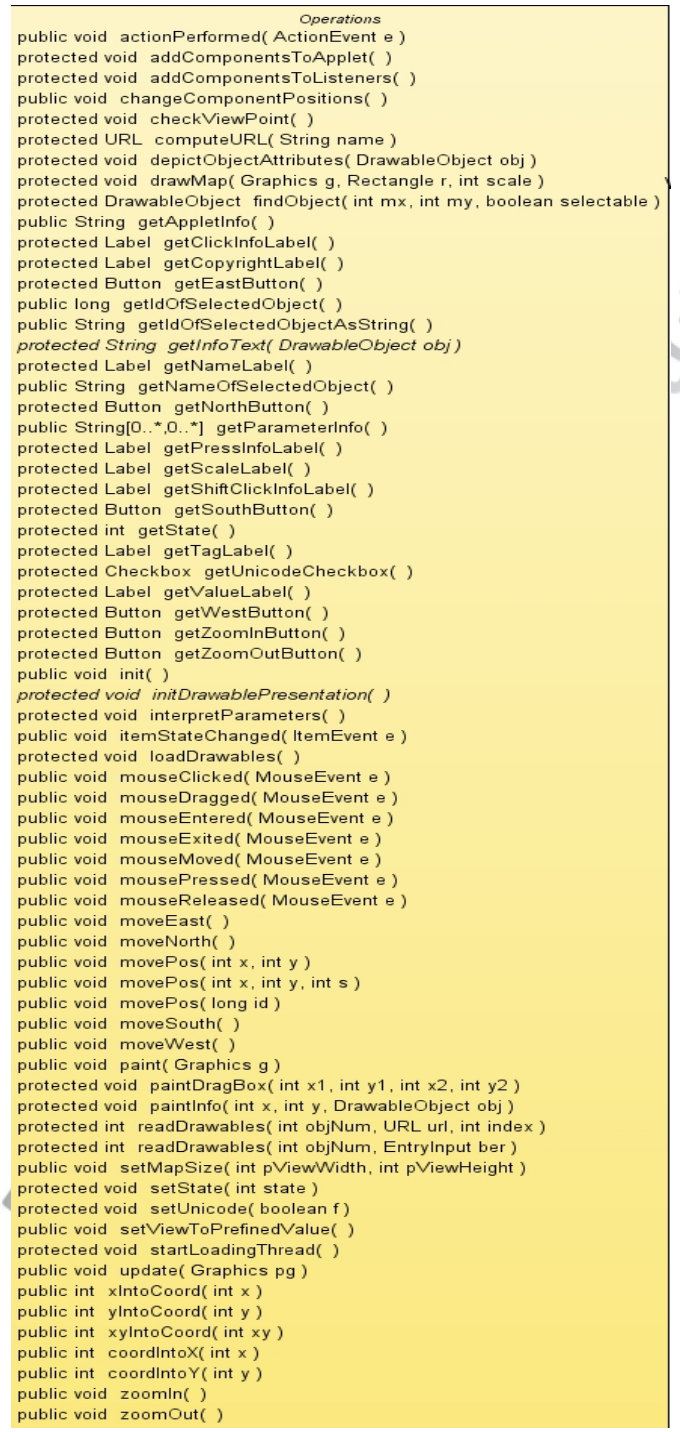

Operations
public void actionPerformed(ActionEvent e)
protected void addComponentsToApplet( )
protected void addComponentsToListeners( )
public void changeComponentPositions( )
protected void checkViewPoint( )
protected URL computeURL(String name)
protected void depictObjectAttributes(DrawableObject obj)
protected void drawMap(Graphics g, Rectangle r, int scale)
protected DrawableObject findObject(int mx, int my, boolean selectable)
public String getAppletInfo( )
protected Label getClickInfoLabel( )
protected Label getCopyrightLabel( )
protected Button getEastButton( )
public long getIdOfSelectedObject( )
public String getIdOfSelectedObjectAsString( )
protected String getInfoText(DrawableObject obj)
protected Label getNameLabel( )
public String getNameOfSelectedObject( )
protected Button getNorthButton( )
public String[0..*,0..*] getParameterInfo( )
protected Label getPressInfoLabel( )
protected Label getScaleLabel( )
protected Label getShiftClickInfoLabel( )
protected Button getSouthButton( )
protected int getState( )
protected Label getTagLabel( )
protected Checkbox getUnicodeCheckbox( )
protected Label getValueLabel( )
protected Button getWestButton( )
protected Button getZoomInButton( )
protected Button getZoomOutButton( )
public void init( )
protected void initDrawablePresentation( )
protected void interpretParameters( )
public void itemStateChanged(ItemEvent e)
protected void loadDrawables( )
public void mouseClicked(MouseEvent e)
public void mouseDragged(MouseEvent e)
public void mouseEntered(MouseEvent e)
public void mouseExited(MouseEvent e)
public void mouseMoved(MouseEvent e)
public void mousePressed(MouseEvent e)
public void mouseReleased(MouseEvent e)
public void moveEast( )
public void moveNorth( )
public void movePos(int x, int y)
public void movePos(int x, int y, int s)
public void movePos(long id)
public void moveSouth( )
public void moveWest( )
public void paint(Graphics g)
protected void paintDragBox(int x1, int y1, int x2, int y2)
protected void paintInfo(int x, int y, DrawableObject obj)
protected int readDrawables(int objNum, URL url, int index)
protected int readDrawables(int objNum, EntryInput ber)
public void setMapSize(int pViewWidth, int pViewHeight)
protected void setState(int state)
protected void setUnicode(boolean f)
public void setViewToPrefinedValue( )
protected void startLoadingThread( )
public void update(Graphics pg)
public int xIntoCoord(int x)
public int yIntoCoord(int y)
public int xyIntoCoord(int xy)
public int coordIntoX(int x)
public int coordIntoY(int y)
public void zoomIn( )
public void zoomOut( )

Figure 18.ShowMap Class Diagram

In current state, after reading the original network files generator exports two files containing the information about the generated edge's and node's characteristics (node id, edge id, edge capacity, edge max speed etc) in order to be used by oracle RDMS to create the spatial network with the exact attributes as the aforementioned network.

6.1.2 DB Network Creation - POIs

In order to store the set of Points of interest and to create the spatial network a spatial database is needed. Although many open source databases exist, Oracle RDBS spatial capabilities are outstanding. Its geospatial data features support complex geographic information systems (GIS) applications, enterprise applications, and location-based services applications, augmenting the Oracle Database Locator feature, which provides storage, analysis, and indexing of 2D location data accessible through SQL and standard programming languages.

Some of these spatial features are:

- Spatial manipulation and analysis such as buffer generation, linear referencing
- Complete geocoding engine and routing engines
- Storage, indexing, and analysis of image and gridded raster data
- A persistent topology data model and schema for land management applications

As mentioned above data generator takes as input a spatial database containing points of interest (POIs).The source for obtaining the data in order to create such a database has been decided to be openstreet maps (OSM) which contains a very large amount of information for every place in the world. Except from the table containing the POIs, a spatial network of the same area with the network that will be used as input for loading from the generator, must be created. There are two reasons for the creation. Firstly it will be used by the spatial queries and the other reason is for testing purposes.

There are a few things to be done before database creation. Data derived from OSM, contain sometimes incomplete information (speed limitation for each street etc) and thus must be filtered and adjusted to the real traffic conditions. Many open source tools are available for osm data files editing. One of the most efficient is the QGIS for desktop use. After data preparation they can be inserted to spatial database using the oracle's sql developer and especially georaptor (spatial extension).

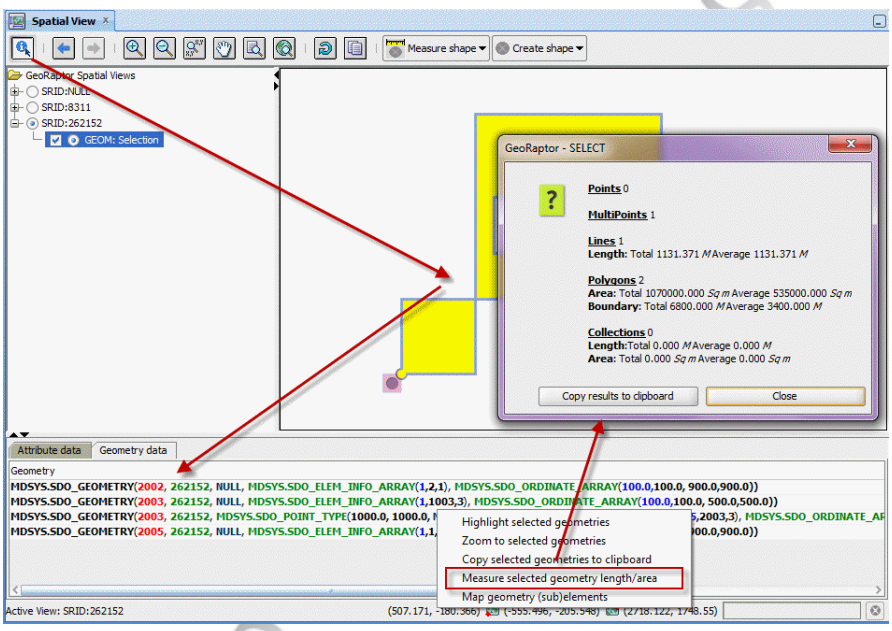


Figure 19. Georaptor - a spatial extension of sql developer

6.1.3 Semantic Database

Currently the generator takes as input the output of the semantic db in a .csv format. This output contains all semantic elements that are needed for the scenario to be created and loaded by the generator. Figure 18 shows the typical structure of this semantic output. Below is a column by column analysis of the semantic DB aiming to better understanding of the scenario elements.

- Columns A,B,D (object_id, sem_traj_id, sem_traj_tag) combined constitute the unique identifier of a semantic trajectory.

- Subsequently columns A-C (object_id, sem_traj_id, epis_id) combined constitute the unique identifier of a semantic sub_trajectory. A sequence of semantic sub_trajectories compose a semantic trajectory as properly defined in section.
- Column E (srid) gives the Spatial Reference System Identifier (SRID) used in semantic. It is required since we must be sure that the coordinates given by the semantic db are in agreement with those used for network creation, else every spatial query will return false results.

  All the remaining columns give the rest of information for a semantic sub_trajectory definition.
  More detailed.
- Column F (epis_def_tag) defines the kind of the sub_traj. (Stop - Move)
- Column G (epis_epis_tag) gives a description of the stop area (if sub_traj is a STOP) and the description of the MOVE in case of a MOVE sub_traj, while column H (epis_activ_tag) describes the activity during a STOP episode and the means of transport in case of a MOVE.
- Columns F-X give the dimensions of the spatiotemporal boxes of each sub_traj. For example epis_minx, epis_miny give the value of the bottom left point of the spatial rectangle.
- Column Y (distance_covered) provides the value of the Euclidean distance covered by each episode. This value can also be computed by the pre given spatial dimensions of the sem_sub_traj.
- Column Z (duration_sec) provides the value of the sem_sub_traj duration that could also be computed by the pre given temporal dimensions of the sem_traj.
- Columns AA,AB,AC (top_speed, avg_speed, speed_var) give information about the speed and its variations during a sem_sub_traj.
- Lastly the column AD (number of profiles) gives the value for the number of sem_sub_traj we want to generate for each given semantic sub trajectory. (number of generated moving objects)

object_id	sem_traj_	epis_id	sem_traj_	srid	epis_def_t	epis_epis_	epis_activ_	epis_minx	epis_miny	epis_miny	epis_minn	ep
1	1	1	straj1	2100	STOP	Home	Relaxing	468993	4201747	2013	4	
1	1	2	straj1	2100	MOVE	Transporta	Walking	468993	4201747	2013	4	
1	1	3	straj1	2100	STOP	School	Studying	468993	4201747	2013	4	
1	1	4	straj1	2100	MOVE	Transporta	Walking	468993	4201747	2013	4	
1	1	5	straj1	2100	STOP	Home	Relaxing	468993	4201747	2013	4	
1	2	1	straj1	2100	STOP	Home	Relaxing	478993	4201747	2013	4	
1	2	2	straj1	2100	MOVE	Transporta	Bicycle	473993	4201747	2013	4	
1	2	3	straj1	2100	STOP	University:	Studying	473993	4201747	2013	4	
1	2	4	straj1	2100	MOVE	Transporta	Bicycle	473993	4201747	2013	4	
1	2	5	straj1	2100	STOP	Cafe	Socializing	473993	4201747	2013	4	
1	2	6	straj1	2100	MOVE	Transporta	Bicycle	473993	4201747	2013	4	
1	2	7	straj1	2100	STOP	Home	Relaxing	478993	4201747	2013	4	
1	3	1	straj1	2100	STOP	Home	Relaxing	463993	4196747	2013	4	
1	3	2	straj1	2100	MOVE	Transporta	Bus	463993	4196747	2013	4	
1	3	3	straj1	2100	STOP	Bank	Banking	473993	4201747	2013	4	
1	3	4	straj1	2100	MOVE	Transporta	Bus	463993	4196747	2013	4	
1	3	5	straj1	2100	STOP	Cafe	Socializing	473993	4196747	2013	4	
1	3	6	straj1	2100	MOVE	Transporta	Bus	463993	4196747	2013	4	
1	3	7	straj1	2100	STOP	Home	Relaxing	463993	4196747	2013	4	
1	4	1	straj1	2100	STOP	Home	Relaxing	468993	4201747	2013	4	
1	4	2	straj1	2100	MOVE	Transporta	Car	468993	4196747	2013	4	
1	4	3	straj1	2100	STOP	Bank	Working	468993	4196747	2013	4	
1	4	4	straj1	2100	MOVE	Transporta	Car	468993	4196747	2013	4	
1	4	5	straj1	2100	STOP	Home	Relaxing	468993	4201747	2013	4	
1	5	1	straj1	2100	STOP	Home	Relaxing	478993	4201747	2013	4	
1	5	2	straj1	2100	MOVE	Transporta	Car	468993	4196747	2013	4	
1	5	3	straj1	2100	STOP	Bank	Working	473993	4196747	2013	4	
1	5	4	straj1	2100	MOVE	Transporta	Car	468993	4196747	2013	4	
1	5	5	straj1	2100	STOP	Cinema:Vil	Shopping	468993	4196747	2013	4	
1	5	6	straj1	2100	MOVE	Transporta	Car	468993	4196747	2013	4	
1	5	7	straj1	2100	STOP	Home	Relaxing	478993	4201747	2013	4	
1	6	1	straj1	2100	STOP	Home	Relaxing	473993	4206747	2013	4	
1	6	2	straj1	2100	MOVE	Transporta	Bus	473993	4201747	2013	4	
1	6	3	straj1	2100	STOP	Cafe	Socializing	478993	4201747	2013	4	
1	6	4	straj1	2100	MOVE	Transporta	Bus	473993	4201747	2013	4	
1	6	5	straj1	2100	STOP	Home	Relaxing	473993	4206747	2013	4	

Figure 20.Semantic Db output that will be used as input for generator (part 1)

is_mind	epis_minh	epis_minn	epis_mins	epis_maxx	epis_maxy	epis_maxy	epis_maxn	epis_maxc	epis_maxl	epis_maxr	epis_maxs	distanc
8	0	0	0	473993	4206747	2013	4	8	7	50	0	
8	7	50	0	473993	4206747	2013	4	8	8	10	0	
8	8	10	0	473993	4206747	2013	4	8	14	0	0	
8	14	0	0	473993	4206747	2013	4	8	14	20	0	
8	14	20	0	473993	4206747	2013	4	8	23	59	59	
8	0	0	0	483993	4206747	2013	4	8	9	0	0	
8	9	0	0	483993	4206747	2013	4	8	9	30	0	
8	9	30	0	478993	4206747	2013	4	8	18	0	0	
8	18	0	0	483993	4206747	2013	4	8	18	30	0	
8	18	30	0	478993	4206747	2013	4	8	20	30	0	
8	20	30	0	483993	4206747	2013	4	8	21	0	0	
8	21	0	0	483993	4206747	2013	4	8	23	59	59	
8	0	0	0	468993	4201747	2013	4	8	8	20	0	
8	8	20	0	478993	4206747	2013	4	8	9	0	0	
8	9	0	0	478993	4206747	2013	4	8	17	0	0	
8	17	0	0	478993	4206747	2013	4	8	17	15	0	
8	17	15	0	478993	4201747	2013	4	8	20	0	0	
8	20	0	0	478993	4206747	2013	4	8	20	30	0	
8	20	30	0	468993	4201747	2013	4	8	23	59	59	
8	0	0	0	473993	4206747	2013	4	8	8	40	0	
8	8	40	0	473993	4206747	2013	4	8	9	0	0	
8	9	0	0	473993	4201747	2013	4	8	17	0	0	
8	17	0	0	473993	4206747	2013	4	8	17	20	0	
8	17	20	0	473993	4206747	2013	4	8	23	59	59	
8	0	0	0	483993	4206747	2013	4	8	8	40	0	
8	8	40	0	483993	4206747	2013	4	8	9	0	0	
8	9	0	0	483993	4201747	2013	4	8	17	0	0	
8	17	0	0	483993	4206747	2013	4	8	17	15	0	
8	17	15	0	473993	4201747	2013	4	8	18	0	0	
8	18	0	0	483993	4206747	2013	4	8	18	30	0	
8	18	30	0	483993	4206747	2013	4	8	23	59	59	
8	0	0	0	478993	4211747	2013	4	8	10	0	0	
8	10	0	0	483993	4206747	2013	4	8	10	30	0	
8	10	30	0	483993	4206747	2013	4	8	13	30	0	
8	13	30	0	483993	4206747	2013	4	8	14	0	0	
8	14	0	0	478993	4211747	2013	4	8	23	59	59	

Figure 21.Semantic Db output that will be used as input for generator (part 2)

.c duration_s	top_speed	avg_speed	speed_var	number of profile
28250	5	500		
1210	5	500		
21000	5	500		
1220	5	500		
34799	5	500		
32400	20	500		
1830	20	500		
30600	20	500		
1830	20	500		
7230	20	500		
1800	20	500		
10799	20	500		
30020	40	750		
2400	40	750		
28800	40	750		
915	40	750		
9900	40	750		
1830	40	750		
12599	40	750		
31240	60	1250		
1200	60	1250		
28800	60	1250		
1220	60	1250		
23999	60	1250		
31240	60	1250		
1200	60	1250		
28800	60	1250		
915	60	1250		
2700	60	1250		
1830	60	1250		
19799	60	1250		
36000	40	750		
1830	40	750		
10830	40	750		
1800	40	750		
35999	40	750		

Figure 22. Semantic Db output that will be used as input for generator (part 3)

There is also a button in the interface that helps the user select the proper semantic file. After loading the information of the input file it calls a specific method that compiles it and stores it in various structures (jtextfields etc) for future use. The procedure with the structures are shown in figures 19, 20.

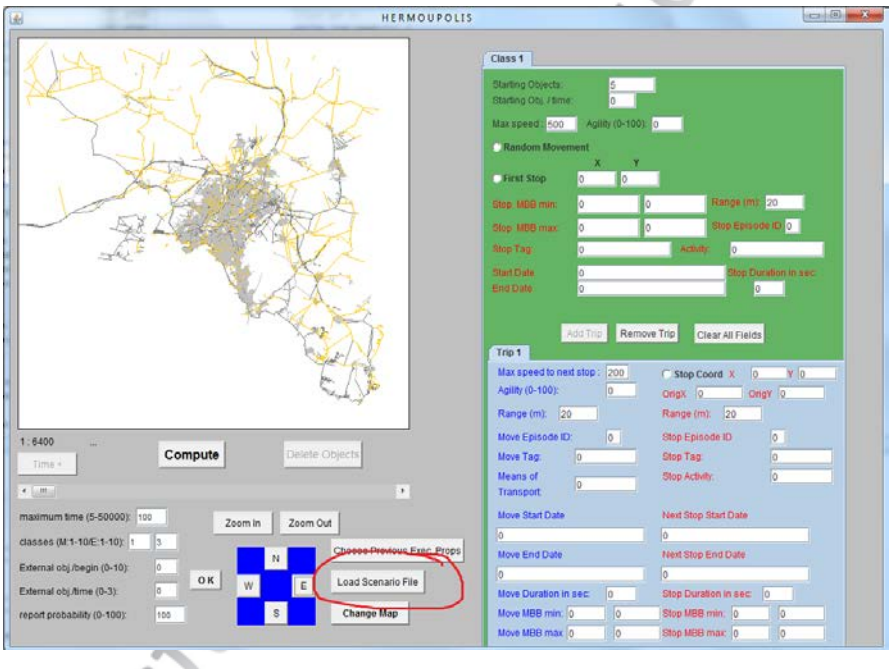


Figure 23. Loading Scenario File

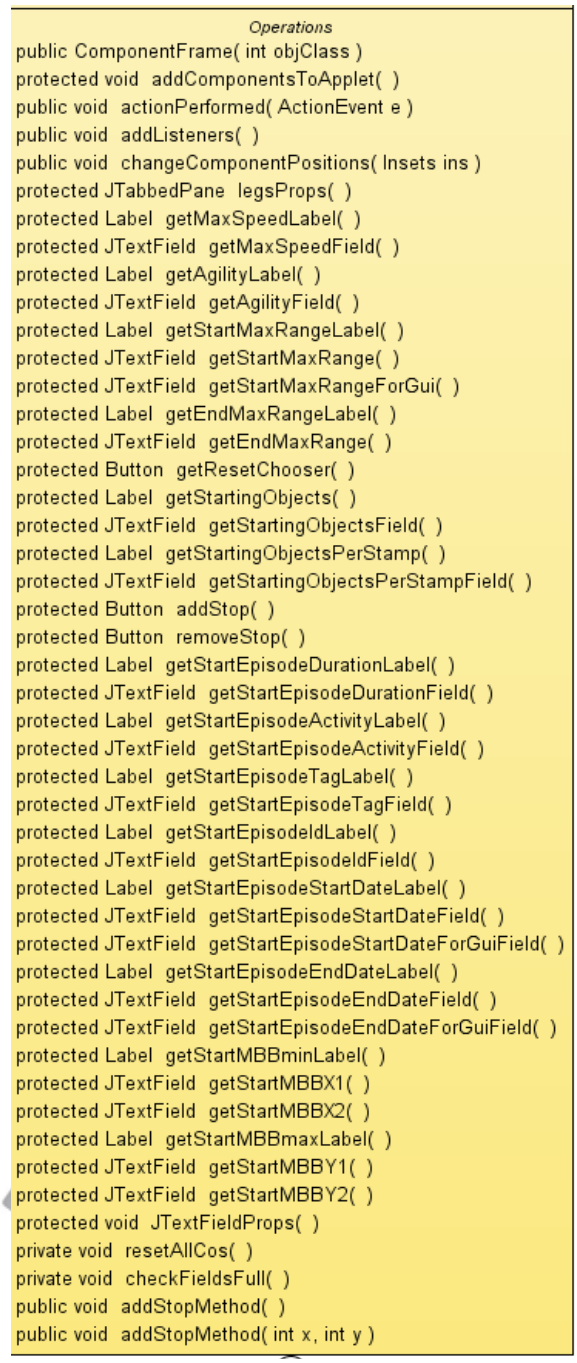


Figure 24. Classes used by generator to store the input semantic Scenario

6.2 Computing the motion

After reading the scenario file and storing the information in the appropriate structures (jtextfields) as described before, the generator is ready to generate the semantic trajectories but does nothing until user decides to do so. There is a method called by a button (user action required) that does the following.

6.2.1 Moving Objects creation.

In the first step the generator initializes everything. Then it creates the predefined number of moving objects divided in classes that each one adopt the characteristics of the semantic sub trajectories given by the preloaded scenario file. Besides a number of other attributes each moving object stores in various structures, it also stores the trips that must follow during its lifetime that each one consist of a couple of STOP and MOVE. Furthermore the STOP implementation is as follows.

6.2.2 STOP episode

As said before every trip consists of a STOP and a MOVE episode. But what does a STOP episode means for a moving object? The created moving objects ,as described in section 6.2.2, stores the information about their routes that consist of STOP and MOVE episodes. More analytically, in order to compute the corresponding network node for every STOP, a 1-NN query is posed to the table containing the POIs that returns a list of the NN nodes of the POIs that are contained to STOP episode's spatial box. If the returned list has more than one element then one of them is randomly selected. Obviously, since the aforementioned procedure is repeated many times (equal to the number of generated moving objects) many objects will store different nodes for the same STOP episode.

Besides spatial info the aforementioned semantic file that generator takes as input contains temporal info for each episode which means that the moving object should remain at the same STOP node for a predefined (simulation time) duration.(Calls a method that simply returns the same node as the next node of moving object's route. Also another user defined variable called *StopSimulationTime* is used in order to define the number of samples that the STOP duration is divided, the simulation time units that an object should remain at the same node.

6.2.3 MOVE episode

After computing all the STOP nodes of every object, computing the corresponding MOVE episode (actually the route between two consecutive STOP nodes) is implemented by calling the BRINKHOFF generator's routing functionality. It is implemented by pressing the compute button which calls the compute () method. The critical point here is that the routing algorithm for every moving object should be called only after the number of simulation time samples that the object should not move from the current stop node.

The routing algorithm used by the generator is actually a variation of the A* algorithm. If moving object's current position is not a STOP node then, if there is a strong deviation of the current speed from the expected speed (e.g. if the car is in a traffic jam) or form an external object (very strong winds etc) and enough time has passed since the last computation of the route so the algorithm computes a different route, on any other case continue the predefined routing. There are two boolean methods describing the two cases that the algorithm computes a newer route than the predefined one.

1. *computeNewRouteByEvent (time, timeOfLastComputation).*This function allows simulating external events. In a simple version, it may return "true" if *time* minus *timeOfLastComputation* is larger than a given threshold.

2. *computeNewRouteByComparison (time, timeOfLastComputation, currentSpeed, expectedSpeed)*

This function allows simulating the reaction in the case of a strong deviation between the current speed and the expected speed on an edge.

The expected speed is calculated by taking into account the maximum speed of the profile for the specific episode, the maximum allowed speed and the maximum capacity of the edge.

The STOP node of the first STOP episode and the STOP node of the last STOP episode for every moving object compose the beginning and the end of the semantic sub_trajectory.

6.3 The Report

The generator in its current state supports three types of reporting:

- The report of the raw trajectories in a simple text file,

- The report of the semantic trajectories in a semantic database table and (figure 21)

- An ad-hoc visualization of the generated moving objects (figure 22)

Columns | Data | Constraints | Grants | Statistics | Triggers | Flashback | Dependencies | Details | Partitions | Indexes | SQL

Actions...

	COLUMN_NAME	DATA_TYPE	NULLABLE	DATA_DEFAULT	COLUMN_ID	COMMENTS
1	TIME	NUMBER(30,0)	Yes	(null)	1	(null)
2	STOPEPISODEID	NUMBER(30,0)	Yes	(null)	2	(null)
3	ID	NUMBER(30,0)	Yes	(null)	3	(null)
4	REPNUM	NUMBER(30,0)	Yes	(null)	4	(null)
5	OBJCLASS	VARCHAR2(30 CHAR)	Yes	(null)	5	(null)
6	STOPDATE	VARCHAR2(50 CHAR)	Yes	(null)	6	(null)
7	EDGEID	NUMBER(30,0)	Yes	(null)	7	(null)
8	X	NUMBER(30,0)	Yes	(null)	8	(null)
9	Y	NUMBER(30,0)	Yes	(null)	9	(null)
10	SPEED	NUMBER(30,0)	Yes	(null)	10	(null)
11	DONEDIST	NUMBER(30,0)	Yes	(null)	11	(null)
12	NEXTNODEID	NUMBER(30,0)	Yes	(null)	12	(null)
13	NEXTNODEX	NUMBER(30,0)	Yes	(null)	13	(null)
14	NEXTNODEY	NUMBER(30,0)	Yes	(null)	14	(null)
15	REPORTPROBABILITY	NUMBER(30,0)	Yes	(null)	15	(null)
16	STOPNUM	NUMBER(30,0)	Yes	(null)	16	(null)
17	REPORTER_TAG	VARCHAR2(30 CHAR)	Yes	(null)	17	(null)
18	EPISODE_ACTIVITY	VARCHAR2(30 CHAR)	Yes	(null)	18	(null)
19	EPISODE_TAG	VARCHAR2(30 CHAR)	Yes	(null)	19	(null)

Figure 25. Semantic Trajectories Report Table Structure

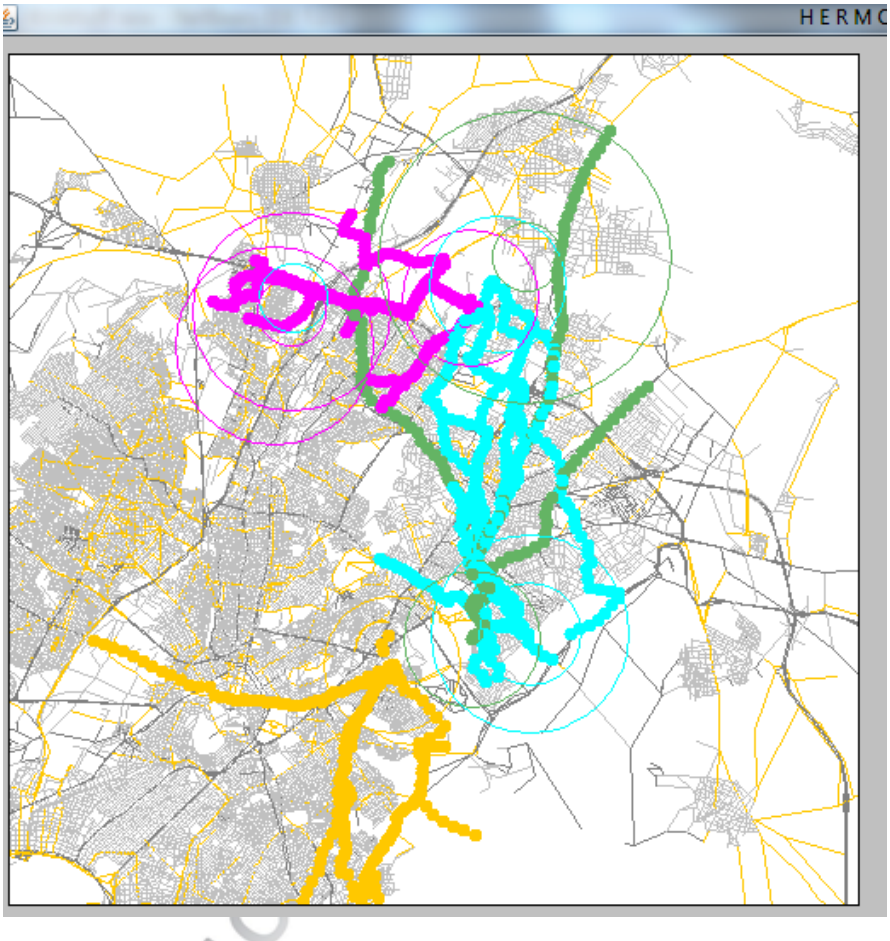


Figure 26.Visualization of generated objects

7 Validation Study

7.1 The Input

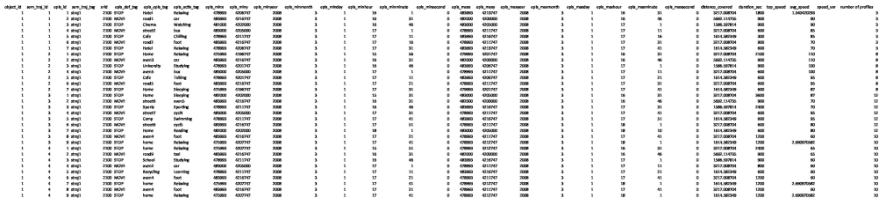
After completing requirement analysis and the design and implementation of our generator we validate all those things we claimed before by the following example.

Input: set of Profiles , number of objects per profile , road network , a set of points of interests , the maximum simulation time , the maximum speed per profile

- Semantic file (semantic DB output) including number of objects for the given profile (we will run the example for only one profile),the maximum simulation time and the maximum speed for the profile (figure 23).

- Attiki Road Network (figure 24).
  We used a very useful tool available from Oracle for spatial network visualization and analysis that is called *Oracle Spatial Network Data Model Editor.*

- POIs TBL (figure 25).



object_id	sem_traj_id	epis_id	sem_traj_tag	srid	epis_def_tag	epis_epis_tag	epis_activ_tag	epis_minx	epis_miny	epis_minyear	epis_minmonth
1	1	1	straj1	2100	STOP	Hotel	Relaxing	478993	4206747	2008	3
1	1	2	straj1	2100	MOVE	road1	car	483993	4216747	2008	3
1	1	3	straj1	2100	STOP	Cinema	Watching	481000	4202000	2008	3
1	1	4	straj1	2100	MOVE	street2	bus	483000	4205000	2008	3
1	1	5	straj1	2100	STOP	Cafe	Chilling	478993	4211747	2008	3
1	1	6	straj1	2100	MOVE	road2	foot	483993	4216747	2008	3
1	1	7	straj1	2100	STOP	Hotel	Relaxing	478993	4206747	2008	3
1	2	1	straj1	2100	STOP	Home	Relaxing	475993	4196747	2008	3
1	2	2	straj1	2100	MOVE	aven3	car	483993	4216747	2008	3
1	2	3	straj1	2100	STOP	University	Studying	478993	4201747	2008	3
1	2	4	straj1	2100	MOVE	aven5	bus	483000	4205000	2008	3
1	2	5	straj1	2100	STOP	Cafe	Talking	478993	4201747	2008	3
1	2	6	straj1	2100	MOVE	road5	foot	483993	4216747	2008	3
1	2	7	straj1	2100	STOP	Home	Sleeping	475993	4196747	2008	3
1	3	1	straj1	2100	STOP	Home	Sleeping	482000	4202000	2008	3
1	3	2	straj1	2100	MOVE	street6	aven5	483993	4216747	2008	3
1	3	3	straj1	2100	STOP	Sports	Sporting	478993	4211747	2008	3
1	3	4	straj1	2100	MOVE	street7	cycl4	483000	4205000	2008	3
1	3	5	straj1	2100	STOP	Camp	Swimming	478993	4211747	2008	3
1	3	6	straj1	2100	MOVE	street6	cycl5	483993	4216747	2008	3
1	3	7	straj1	2100	STOP	Home	Reading	482000	4202000	2008	3
1	3	8	straj1	2100	MOVE	aven4	foot	483993	4216747	2008	3
1	3	9	straj1	2100	STOP	home	Relaxing	475993	4207747	2008	3
1	4	1	straj1	2100	STOP	home	Relaxing	475993	4207747	2008	3
1	4	2	straj1	2100	MOVE	road9	taxi	483993	4216747	2008	3
1	4	3	straj1	2100	STOP	School	Studying	478993	4211747	2008	3
1	4	4	straj1	2100	MOVE	aven2	car	483000	4205000	2008	3
1	4	5	straj1	2100	STOP	Recycling	Learning	478993	4211747	2008	3
1	4	6	straj1	2100	MOVE	aven4	foot	483993	4216747	2008	3
1	4	7	straj1	2100	STOP	home	Relaxing	475993	4207747	2008	3
1	4	8	straj1	2100	MOVE	aven4	foot	483993	4216747	2008	3
1	4	9	straj1	2100	STOP	home	Relaxing	475993	4207747	2008	3

Figure 27.Semantic file fiven to generator as input (part 1)

epis_minday	epis_minhour	epis_minminute	epis_minsecond	epis_maxx	epis_maxy	epis_maxyear	epis_maxmonth	epis_maxday	epis_maxhour	epis_maxminute
1	16	1	0	483993	4213747	2008	3	1	16	31
1	16	31	0	482000	4200000	2008	3	1	16	46
1	16	46	0	483000	4205000	2008	3	1	17	1
1	17	1	0	478993	4211747	2008	3	1	17	11
1	17	11	0	483993	4216747	2008	3	1	17	16
1	17	16	0	478993	4211747	2008	3	1	17	31
1	17	31	0	483993	4213747	2008	3	1	17	41
1	15	56	0	479993	4201747	2008	3	1	16	31
1	16	31	0	482000	4200000	2008	3	1	16	46
1	16	46	0	483993	4206747	2008	3	1	17	1
1	17	1	0	478993	4211747	2008	3	1	17	11
1	17	11	0	483993	4206747	2008	3	1	17	21
1	17	21	0	478993	4211747	2008	3	1	17	31
1	17	31	0	479993	4201747	2008	3	1	17	41
1	15	31	0	483000	4205000	2008	3	1	16	31
1	16	31	0	482000	4200000	2008	3	1	16	46
1	16	46	0	483993	4216747	2008	3	1	17	31
1	17	31	0	478993	4211747	2008	3	1	17	41
1	17	41	0	483993	4216747	2008	3	1	17	51
1	17	51	0	478993	4211747	2008	3	1	18	1
1	18	1	0	483000	4205000	2008	3	1	18	10
1	17	21	0	478993	4211747	2008	3	1	17	41
1	17	41	0	478993	4212747	2008	3	1	18	1
1	15	51	0	478993	4212747	2008	3	1	16	31
1	16	31	0	482000	4200000	2008	3	1	16	46
1	16	46	0	483993	4216747	2008	3	1	17	1
1	17	1	0	478993	4211747	2008	3	1	17	11
1	17	11	0	483993	4216747	2008	3	1	17	21
1	17	21	0	478993	4211747	2008	3	1	17	41
1	17	41	0	478993	4212747	2008	3	1	18	1
1	17	21	0	478993	4211747	2008	3	1	17	41
1	17	41	0	478993	4212747	2008	3	1	18	1

Figure 28.Semantic file fiven to generator as input (part 2)

epis_maxsecond	distance_covered	duration_sec	top_speed	avg_speed	speed_var	number of profiles
0	3217,008704	1800		1,340420293		3
0	5692,114735	900		90		3
0	1586,597814	900		90		3
0	3217,008704	600		85		3
0	1614,582349	300		85		3
0	3217,008704	900		70		3
0	1614,582349	600		70		3
0	3217,008704	2100		110		8
0	5692,114735	900		110		8
0	1586,597814	900		100		8
0	3217,008704	600		100		8
0	1614,582349	600		95		8
0	3217,008704	600		95		8
0	1614,582349	600		87		8
0	3217,008704	3600		87		12
0	5692,114735	900		70		12
0	1586,597814	2400		70		12
0	3217,008704	600		95		12
0	1614,582349	600		95		12
0	3217,008704	600		80		12
0	1614,582349	600		80		12
0	3217,008704	1200		60		10
0	1614,582349	1200		2,690970582		10
0	3217,008704	2400		65		10
0	5692,114735	900		65		10
0	1586,597814	900		60		10
0	3217,008704	800		60		10
0	1614,582349	600		60		10
0	3217,008704	1200		60		10
0	1614,582349	1200		2,690970582		10
0	3217,008704	1200		60		10
0	1614,582349	1200		2,690970582		10

Figure 29. Semantic file fiven to generator as input (part 3)

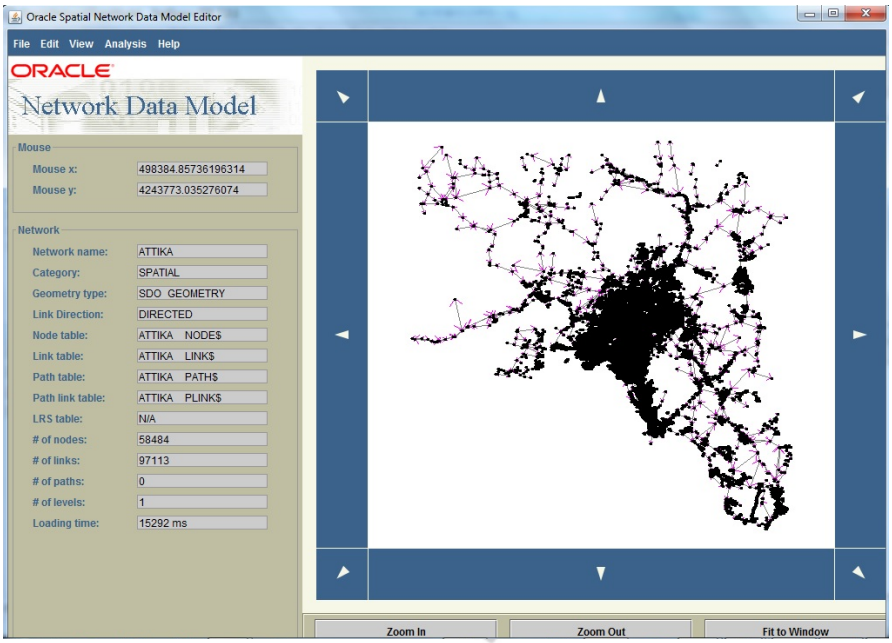


Figure 30.Attika_Network

CATEGORY	ID	NAME	G
Government and Public Services	5149	Tram stop:??????? ??????	<
Government and Public Services	5150	School:72? ????????	<
Government and Public Services	5151	Place of Worship:???? ...	<
Government and Public Services	5152	School:121? ???????? ?...	<
Government and Public Services	5153	Home	<
Government and Public Services	5154	Home	<
Government and Public Services	5155	Place of Worship:???? ...	<
Government and Public Services	5156	Public telephone	<
Government and Public Services	5157	Home	<
Government and Public Services	5158	Home	<
Government and Public Services	5159	Bus Stop:???	<
Government and Public Services	5160	Home	<
Government and Public Services	5161	School	<
Government and Public Services	5162	Home	<
Government and Public Services	5163	Tram stop:??? ???? ??????	<
Government and Public Services	5164	Home	<
Government and Public Services	5165	Court house:?????? ?????	<
Government and Public Services	5166	Subway Entrance	<

Figure 31.Visualization of the database table containing the POIs in Attiki (part 1)

GEOM
<gml:Point srsName="SDO:2100" xmlns:gml="http://www.opengis.net/gml"><gml:posList srsDimension="2">
<gml:Point srsName="SDO:2100" xmlns:gml="http://www.opengis.net/gml"><gml:posList srsDimension="2">
<gml:Point srsName="SDO:2100" xmlns:gml="http://www.opengis.net/gml"><gml:posList srsDimension="2">
<gml:Point srsName="SDO:2100" xmlns:gml="http://www.opengis.net/gml"><gml:posList srsDimension="2">
<gml:Point srsName="SDO:2100" xmlns:gml="http://www.opengis.net/gml"><gml:posList srsDimension="2">
<gml:Point srsName="SDO:2100" xmlns:gml="http://www.opengis.net/gml"><gml:posList srsDimension="2">
<gml:Point srsName="SDO:2100" xmlns:gml="http://www.opengis.net/gml"><gml:posList srsDimension="2">
<gml:Point srsName="SDO:2100" xmlns:gml="http://www.opengis.net/gml"><gml:posList srsDimension="2">
<gml:Point srsName="SDO:2100" xmlns:gml="http://www.opengis.net/gml"><gml:posList srsDimension="2">
<gml:Point srsName="SDO:2100" xmlns:gml="http://www.opengis.net/gml"><gml:posList srsDimension="2">
<gml:Point srsName="SDO:2100" xmlns:gml="http://www.opengis.net/gml"><gml:posList srsDimension="2">
<gml:Point srsName="SDO:2100" xmlns:gml="http://www.opengis.net/gml"><gml:posList srsDimension="2">
<gml:Point srsName="SDO:2100" xmlns:gml="http://www.opengis.net/gml"><gml:posList srsDimension="2">
<gml:Point srsName="SDO:2100" xmlns:gml="http://www.opengis.net/gml"><gml:posList srsDimension="2">
<gml:Point srsName="SDO:2100" xmlns:gml="http://www.opengis.net/gml"><gml:posList srsDimension="2">
<gml:Point srsName="SDO:2100" xmlns:gml="http://www.opengis.net/gml"><gml:posList srsDimension="2">
<gml:Point srsName="SDO:2100" xmlns:gml="http://www.opengis.net/gml"><gml:posList srsDimension="2">
<gml:Point srsName="SDO:2100" xmlns:gml="http://www.opengis.net/gml"><gml:posList srsDimension="2">
<gml:Point srsName="SDO:2100" xmlns:gml="http://www.opengis.net/gml"><gml:posList srsDimension="2">
<gml:Point srsName="SDO:2100" xmlns:gml="http://www.opengis.net/gml"><gml:posList srsDimension="2">
<gml:Point srsName="SDO:2100" xmlns:gml="http://www.opengis.net/gml"><gml:posList srsDimension="2">
<gml:Point srsName="SDO:2100" xmlns:gml="http://www.opengis.net/gml"><gml:posList srsDimension="2">
<gml:Point srsName="SDO:2100" xmlns:gml="http://www.opengis.net/gml"><gml:posList srsDimension="2">

Figure 32. Visualization of the database table containing the POIs in Attiki (part 2)

477966.591555584 4190338.75018413 </gml:posList></gml:Point>
478014.557046018 4206479.23150097 </gml:posList></gml:Point>
477996.236173924 4197163.89591997 </gml:posList></gml:Point>
478023.929974299 4205265.43915141 </gml:posList></gml:Point>
478051.386078227 4214633.37068902 </gml:posList></gml:Point>
478065.265796758 4217735.99335485 </gml:posList></gml:Point>
478005.066897531 4195161.29840597 </gml:posList></gml:Point>
478024.501401623 4197790.76762394 </gml:posList></gml:Point>
478075.138689229 4214624.71923528 </gml:posList></gml:Point>
478118.64451403 4230311.25539884 </gml:posList></gml:Point>
478086.059644078 4204230.46528058 </gml:posList></gml:Point>
478120.550802681 4216015.38998947 </gml:posList></gml:Point>
478095.282135544 4197867.10317973 </gml:posList></gml:Point>
478146.994006436 4214892.06280035 </gml:posList></gml:Point>
478085.92905324 4188854.65280996 </gml:posList></gml:Point>
478170.647662021 4214882.723958 </gml:posList></gml:Point>
478146.137703254 4204295.30207967 </gml:posList></gml:Point>
478152.259574302 4203260.10203159 </gml:posList></gml:Point>
478158.966855834 4203229.39456161 </gml:posList></gml:Point>
478148.986270945 4198517.17512043 </gml:posList></gml:Point>
478173.69407099 4200268.56321807 </gml:posList></gml:Point>
478185.985783387 4203231.05380628 </gml:posList></gml:Point>
478237.65284938 4215905.33233134 </gml:posList></gml:Point>

Figure 33. Visualization of the database table containing the POIs in Attiki (part 3)

NN_NODE_ID	NN_NODE_X	NN_NO...
385908249	485634	4203050
387190192	484980	4199567
386249082	486222	4190426
386138487	485659	4203438
384330722	485859	4220385
386090371	485621	4219684
386507575	485859	4220385
386678318	485528	4210485
384330722	485859	4220385
398496683	485859	4220385
386890462	485859	4220385
387486994	485484	4205802
387859356	485859	4220385
386198913	485477	4209196
387605316	485859	4220385
386198913	485859	4220385
387770905	485060	4197491
388114879	486222	4190426
388114879	485859	4220385
387794668	485621	4219684
387922810	485659	4203438
388114879	485859	4220385
387486994	485859	4220385
386971956	485859	4220385

Figure 34. Visualization of the database table that contains the POIs in Attiki (part 4)

7.2 The Computation of motion

After loading the above Network and the semantic file the generator is at the status shown in fig 26 waiting for semantic trajectories generation.

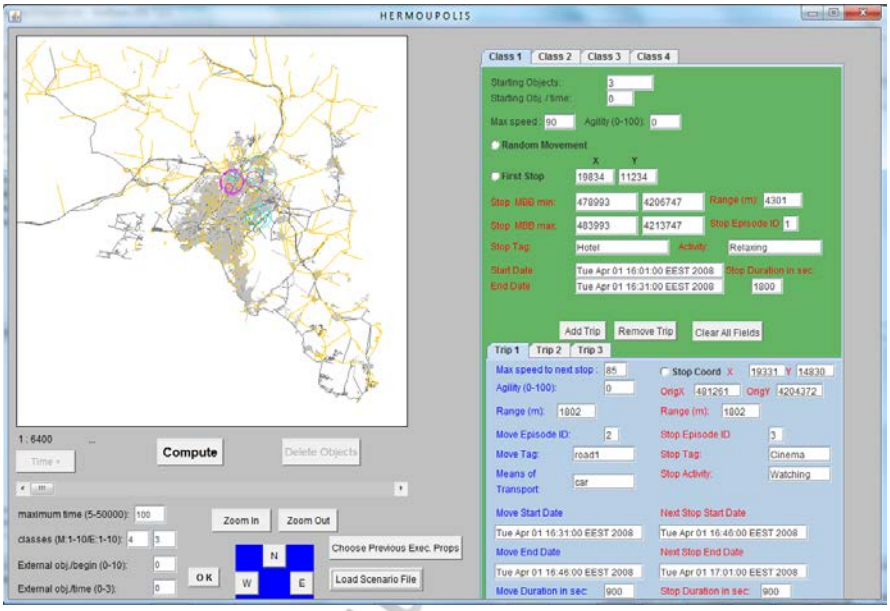


Figure 35. Hermoupolis after loading the scenario file

After calling the compute() method and finished with semantic trajectories generation the generator looks like figure 27.Notice the max time for simulation that has been set to 400 in order to complete all the moving objects their trajectories.

Figure 36.Hermoupolis in time 79

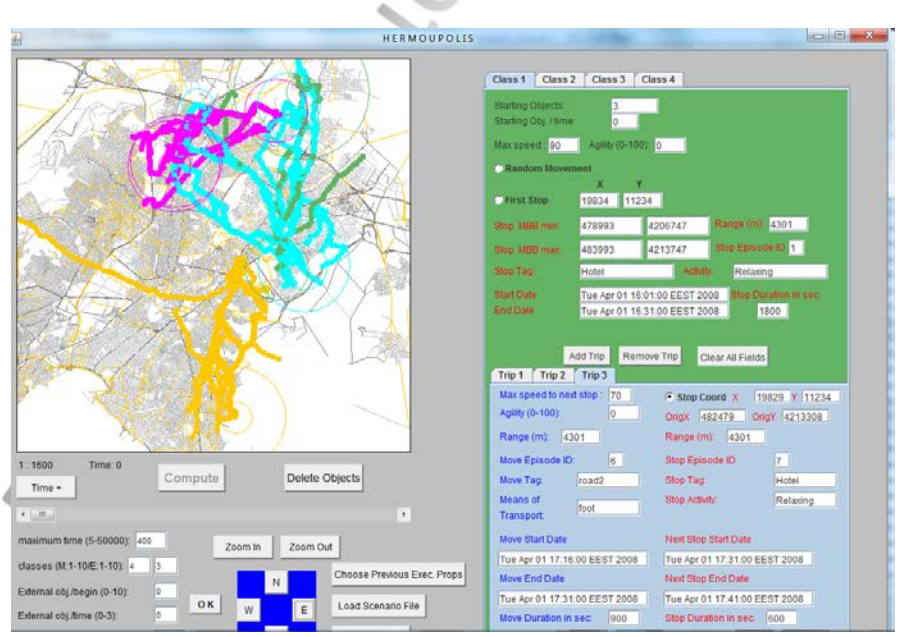


Figure 37.Hermoupolis after generation

7.3 The Report

Figures 36,37,38 show the semantic output after semantic trajectories generation for moving object id 7.

STOPEPISODEID	REPNUM	OBJCLASS	REPORTPRO...	SPEED	NEXTNO...	NEXTNO...	NEXTNO...	EDGEID
1	2	1	100	95	420024543	481819	4205001	79586
1	3	1	100	95	420024543	481819	4205001	79586
1	4	1	100	95	420024543	481819	4205001	79586
1	5	1	100	95	420024543	481819	4205001	79586
1	6	1	100	95	420024543	481819	4205001	79586
1	7	1	100	95	420024543	481819	4205001	79586
1	8	1	100	95	420024543	481819	4205001	79586
1	9	1	100	95	420024543	481819	4205001	79586
1	10	1	100	95	420024543	481819	4205001	79586
1	11	1	100	95	420024543	481819	4205001	79586
1	12	1	100	95	420024543	481819	4205001	79586
1	13	1	100	95	420024543	481819	4205001	79586
1	14	1	100	95	420024543	481819	4205001	79586
1	15	1	100	95	420024543	481819	4205001	79586
1	16	1	100	95	420024543	481819	4205001	79586
1	17	1	100	95	420024543	481819	4205001	79586
1	18	1	100	95	420024543	481819	4205001	79586
1	19	1	100	95	420024543	481819	4205001	79586
1	20	1	100	95	420024543	481819	4205001	79586
1	21	1	100	95	420024543	481819	4205001	79586
1	22	1	100	95	420024543	481819	4205001	79586
1	23	1	100	95	420024543	481819	4205001	79586
1	24	1	100	95	420024543	481819	4205001	79586

Figure 38.Semantic output for object id 7 (part1)

X	Y	DONEDIST	STOPNUM	TIME	ID
481819	4205001	0	0	1	7
481819	4205001	0	0	2	7
481819	4205001	0	0	3	7
481819	4205001	0	0	4	7
481819	4205001	0	0	5	7
481819	4205001	0	0	6	7
481819	4205001	0	0	7	7
481819	4205001	0	0	8	7
481819	4205001	0	0	9	7
481819	4205001	0	0	10	7
481819	4205001	0	0	11	7
481819	4205001	0	0	12	7
481819	4205001	0	0	13	7
481819	4205001	0	0	14	7
481819	4205001	0	0	15	7
481819	4205001	0	0	16	7
481819	4205001	0	0	17	7
481819	4205001	0	0	18	7
481819	4205001	0	0	19	7
481819	4205001	0	0	20	7
481819	4205001	0	0	21	7
481819	4205001	0	0	22	7
481819	4205001	0	0	23	7
481819	4205001	0	0	24	7
481819	4205001	0	0	25	7
481819	4205001	0	0	26	7

Figure 39.Semantic output for object id 7 (part2)

STOPDATE	REPORTER_TAG	EPISOD...	EPISOD
Tue Apr 01 15:31:57 EEST 2008	Stop_0	Sleeping	Home
Tue Apr 01 15:32:55 EEST 2008	Stop_0	Sleeping	Home
Tue Apr 01 15:33:53 EEST 2008	Stop_0	Sleeping	Home
Tue Apr 01 15:34:50 EEST 2008	Stop_0	Sleeping	Home
Tue Apr 01 15:35:48 EEST 2008	Stop_0	Sleeping	Home
Tue Apr 01 15:36:45 EEST 2008	Stop_0	Sleeping	Home
Tue Apr 01 15:37:43 EEST 2008	Stop_0	Sleeping	Home
Tue Apr 01 15:38:41 EEST 2008	Stop_0	Sleeping	Home
Tue Apr 01 15:39:38 EEST 2008	Stop_0	Sleeping	Home
Tue Apr 01 15:40:36 EEST 2008	Stop_0	Sleeping	Home
Tue Apr 01 15:41:33 EEST 2008	Stop_0	Sleeping	Home
Tue Apr 01 15:42:31 EEST 2008	Stop_0	Sleeping	Home
Tue Apr 01 15:43:29 EEST 2008	Stop_0	Sleeping	Home
Tue Apr 01 15:44:26 EEST 2008	Stop_0	Sleeping	Home
Tue Apr 01 15:45:24 EEST 2008	Stop_0	Sleeping	Home
Tue Apr 01 15:46:21 EEST 2008	Stop_0	Sleeping	Home
Tue Apr 01 15:47:19 EEST 2008	Stop_0	Sleeping	Home
Tue Apr 01 15:48:17 EEST 2008	Stop_0	Sleeping	Home
Tue Apr 01 15:49:14 EEST 2008	Stop_0	Sleeping	Home
Tue Apr 01 15:50:12 EEST 2008	Stop_0	Sleeping	Home
Tue Apr 01 15:51:09 EEST 2008	Stop_0	Sleeping	Home
Tue Apr 01 15:52:07 EEST 2008	Stop_0	Sleeping	Home
Tue Apr 01 15:53:05 EEST 2008	Stop_0	Sleeping	Home
Tue Apr 01 15:54:02 EEST 2008	Stop_0	Sleeping	Home
Tue Apr 01 15:55:00 EEST 2008	Stop_0	Sleeping	Home
Tue Apr 01 15:55:57 EEST 2008	Stop_0	Sleeping	Home

Figure 40.Semantic output for object id 7 (part 3)

8 Conclusion – Feature Work

We tried to build a moving object generator that complies with previous stated algorithms and achieves as much as possible of the above characteristics. As far as we know a generator like HERMOUPOLIS has not been proposed yet in the literature.

We have taken an already capable generator (Brinkhoff generator) and tried to enrich it in a way that it could generate both raw and enriched (semantic) trajectories based on general information that are given as input. Also it tries to simulate the real life motion of object groups besides unexpected changes that could be caused by various reasons.

This kind of generation will help one to understand, and hence resolve problems created by the movement of objects involved with traffic, analysis and thus rescue of groups of endangered animals etc. We hope that not only researchers but everyone that needs to produce such kind of data will find our generator very helpful.

The next step that should be done as a future work is to give the people the ability to access the generator through a web interface besides placing their own datasets and accessing the output semantic data.

References

1. *Theodoridis, Y., J.R.O. Silva, and M.A. Nascimento*. On the Generation of Spatiotemporal Datasets. *in* 6th International Symposium on Large Spatial Databases. *1999. Hong Kong.*
2. Pfoser, D. and Y. Theodoridis, *Generating semantics-based trajectories of moving objects.* Comput. Environ. Urban Syst., 2003. **27**(3): p. 243-263.
3. Giannotti, F., et al., *Synthetic generation of cellular network positioning data.* GIS '05, 2005(Proceedings of the 13th annual ACM international workshop on Geographic Information Systems): p. 8.
4. Tzouramanis, T., M. Vassilakopoulos, and Y. Manolopoulos, *On the generation of time-evolving regional data.* Geoinformatica, 2002. **6**(3): p. 207-231.
5. Saglio, J.-M. and J. Moreira, *Oporto: a realistic scenario generator for moving objects.* Geoinformatica, 2001. **5**(1): p. 71-93.
6. *Brinkhoff, T.,* A framework for generating network-based moving objects. *Geoinformatica, 2002. **9**(1): p. 153-180.*
7. *Krajzewicz, D., et al.* SUMO(Simulation of Urban MObility): an open-source traffic simulation. *in* 4th Middle East Symposium on Simulation and Modelling. *2002. SCS European Publishing House*.
8. Nanni, M. and D. Pedreschi, *Time-focused clustering of trajectories of moving objects.* Journal of Intelligent Information Systems, 2006. **27**(3): p. 267-289.
9. *Pelekis, N., Kopanakis, I., Kotsifakos, E., Frentzos, E., and Theodoridis, Y.* Clustering Trajectories of Moving Objects in an Uncertain World. *in* International Conference on Data Mining (ICDM09). *2009. Miami, FL,USA.*
10. Duntgen, C., T. Behr, and R.H. Guting, *BerlinMOD: a benchmark for moving object databases.* The VLDB Journal — The International Journal on Very Large Data Bases, 2008 **18**(6): p. 34.
11. *Dieker, S. and R.H. Guting.* Plug and Play with Query Algebras: SECONDO-A Generic DBMS Development Environment. *in* IDEAS '00 Proceedings of the 2000 International Symposium on Database Engineering & Applications *2000.*
12. *Gidofalvi, G. and T.B. Pedersen.* ST--ACTS: a spatio-temporal activity simulator. *in* 14th annual ACM international symposium on Advances in geographic information systems. *2006.*
13. *J. Xu and R.H. Güting. MWGen:* A Mini World Generator, 13th International Conference on Mobile Data Management (IEEE MDM), pages 258-267, *2012.*
14. *Haibo* Hu, *Dik Lun* Lee:GAMMA*:* A Framework for Moving Object Simulation. *SSTD* 2005*: 37-54.*