Framework for The Generation of Realistic Traffic Simulation Using PostGis

Abstract:
To have vehicles, wandering around to obtain their GPS signals is expensive and time consuming[1] With this framework, the reader will understand how is possible to generate realistic GPS signals of vehicles with PostGis[2] and Java. It will also be explained, what defines a realistic GPS signal.

Authors: Evaldas Jankus, Filipe Emanuel Barroso and Giedrius Zakaitis

2011 May 27

Table of Contents

1. Introduction

Traffic management is playing a very important role in planning city or town infrastructure.
As the number of vehicles is increasing, planning all traffic scheme is becoming very challenging. It is very important to ensure that network of the roads would be optimized in a sense of money saving as much as possible. It would be very helpful for engineers, to have data, which could show how the vehicles would move on given map. So, there is a need of GPS data, based on real-world vehicles behavior. [1][3]
It can be achieved by two ways:

a) Giving simulator real data
b) Use synthetic data

The difference between real and synthetic data is that synthetic data is a data generated by computer, which looks as realistic as it could (this data represent real life situations). First approach would be very expensive and there would not be enough drivers to simulate, for example, rush hour. On the other hand, this data would be very realistic. Hover, one of the best advantages of synthetic is that user could choose as many vehicles as he wants (depending on computer hardware, which is relatively cheap nowadays), trying to see, what happens if there will be 10 or even 100 times more vehicles than usual. Traffic simulator tool could be useful to simulate vehicles traffic on any map, even the one, which does not exist in real world, for example, planning new city's infrastructure, in that case traffic simulator tool is especially important in sense of saving the money.

2. Related work

- [4]"Continuum Traffic Simulation" by J. Sewall ; D. Wilkie ; P. Merrell and M. C. Lin
- [5]"Virtual Traffic Simulation " by Jonathan Ruttle
- [6]"Application of multi-agent systems in traffic and transportation ," by Burmeister, B. ; Haddadi, A. and Matylis, G.

All these works are complex, our work propose a more lite version. For example: [4] works with a long period of time and in a more macro state; [5] shows the results in a CGI output for visualization and [6] shows a different method of generating GPS signals of vehicle by using an agents system. But with this work it is shown that it is not necessary and it is more usefull if given certain parameters and a universal output (KML) it will not be necessary to be a schoolar to test various case scenarious of traffic and analize the output.

3. Traffic simulation

All traffic simulators can be divided into three types:[5]

	Advantages	**Disadvantages**
[5][4] .Macroscopic traffic simulation	• Traffic can be simulated in large area • Good for simulating major roads, highways	• Vehicles specifications cannot be detailed, traffic is simulated as continuous flow
[5]. Mesoscopic traffic simulation	• Good for simulating problematic areas: intersections,roundabouts, changing of lanes in highway.	• Only particular problematic area can be simulated • Speed, density and flow information is needed.

	• Can be used to see where traffic jam will occur • Does not need much computing • Large time-span of traffic can be simulated in short time.	
[5].Microscopic traffic simulation	• Simulation can be very detailed • Each vehicle has its own characteristics • Good for simulating city center or other areas, where every vehicle specification is important • Generated data is very similar to real life driver behavior	• Depending on details level, a lot of computing por is required. • Cannot be simulated in large area. • To achieve, that simulated vehicles behave like real, very sophisticated algorithms is needed

● **[5].Macroscopic** - it is continuum-based[1] traffic data generator. It is used to generate aggregated values continuous flow of vehicles, not according to details of each vehicle. Because of that reason, it does not require much computing resources and large networks of traffic can be generated.

● **[5].Mesoscopic** - traffic simulation based on speed-flow-density relationship. It could be described as simple formula: Flow (vehicles per hour) = Speed * Density, if there are not many vehicles (low density), drivers accelerates, otherwise, if traffic density is very high, drivers must decelerate. This method is best, when it is necessary to determine, where the traffic congestion will occur, because a large time span can be computed within seconds.

● **[5].Microscopic** - this method concentrates most on details of each vehicle (driving style, type of the vehicle, behavior at the intersections, acceleration and deceleration), this type of simulation is very detailed, although it requires a lot of computing power, so only small area can be simulated.

These types of simulation can be joined together. One of the main reasons to join these types of simulation is real life representation. Each style separately can make only a plain view of the traffic simulation (not realistic enough), for example macroscopic type could be used to reflect traffic of vehicles in a large scale of area, and for more detailed view could be used microscopic type of simulation. Besides when a traffic is in a large area and consist of a big density of vehicle simulation would be impossible with using only microscopic methods because calculations would be too difficult and it would take too much time to calculate necessary attributes (speed, deceleration, acceleration, etc) without using other types of simulations like mesoscopic.

Because of vehicle density, speed and flow it is possible to face a problem of calculations. Mesoscopic type of simulation perfectly suits to solve this problem.

1 Continuum-based traffic is continuous flow of the vehicles, when given number of car objects is created and when they reach end of the map object is deleted and new one in the beginning created.

In this project Macroscopic and Microscopic methods will be used, to achieve detailed characteristics of vehicles and to be able simulate traffic on large area. Mesoscopic methods are more appointed to junctions and interactions between vehicles. In order to that just a little part of those methods are used in project (acceleration and deceleration issues).

4. Running example

Let's discuss a concrete example Læsø island.

Figure 1: Laeso island [source: maps.google.com]

As mentioned before, the macroscopic goal is to model in a big area without getting into details too much. Simplify macroscopic problem of vehicles movements in island is reflected in *Figure 1*. Different color circles are special zones marked on the map to reduce number of the details.

Special zones:

- Red circle (Work area)
- Blue circle (Home area)
- Green circle (Leisure area)

There is a need to have special zones. Zoning map helps avoid additional calculation, which needed to display realistic traffic situations. The use of special zones are showed below.

- In the morning all vehicles from home area will move to work area.
- On rush hours most of the vehicles will travel from work area to home area and some of the vehicles will move to Leisure area.
- On the weekends most of the vehicles will travel from home area to leisure area and just a few vehicles will turn to work area.

If there is a need to represent traffic as realistic as possible, besides macroscopic side of the project, also need to discuss microscopic side. Microscopic side of view is paying attention more into details, but in that case mapping of the traffic gets more difficult in larger areas. So, for this point, there is a simplified representation of part of the island, in a picture below.

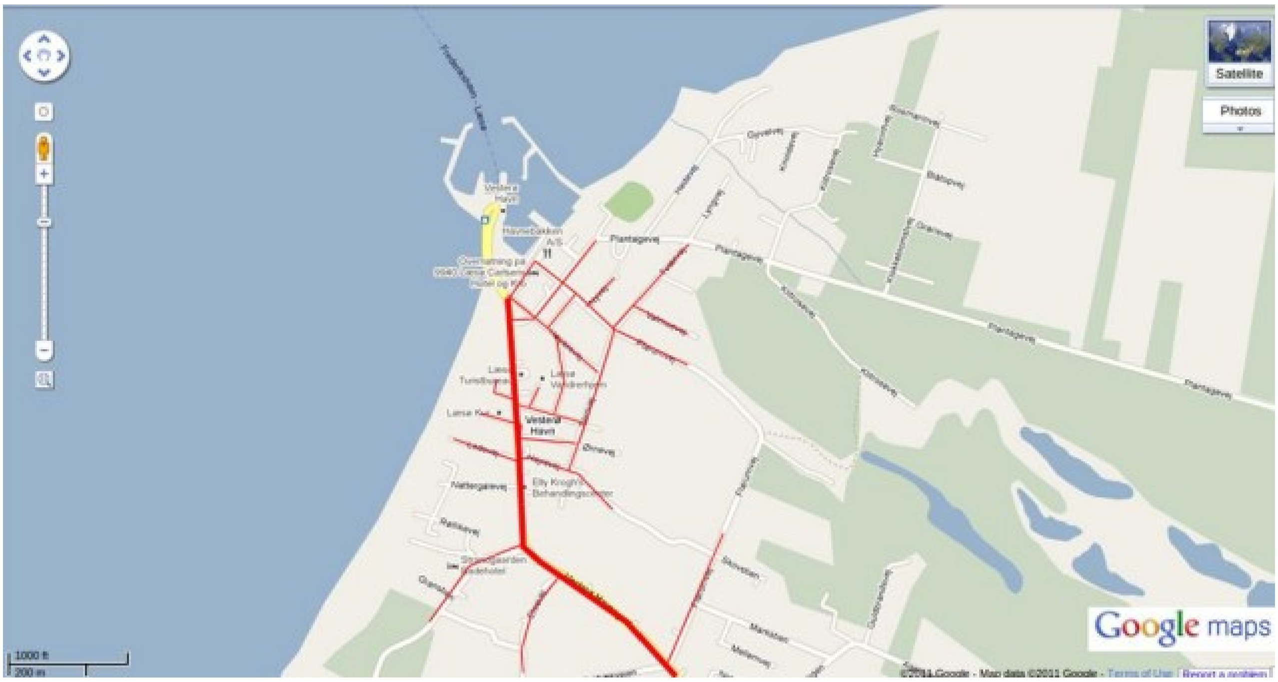

Figure 2: Laeso island [source: maps.google.com]

Considering only the roads, highlighted in red and remembering that talking about work zone. It is expected that there is a big amount of vehicles traffic in the morning and in the evening.

After running traffic simulation program, user will be able to know:

- How vehicle will act in a certain situation at a certain hour.
- Which is the best way to go at a certain hour.

This is perfect tool for improving traffic in the roads.

5. Traffic

It is important to simulate traffic on a big road in a sense of vehicles because most of the traffic on a rush hour simulated there. The traffic on a big road suppose to be a salvation to people who want to get somewhere quick. So, always there is a need to monitoring the traffic on that side. Part of the main idea of this traffic is to get more quickly to the destination, but for example, at rush hour the bigger roads are always filled by vehicles. In that case the traffic in there supposes to be more slow. In order to avoid traffic congestion it is better to take a little circle around of the small road (sometimes to take a longer path can take you faster to your destination).

Traffic in a small road is less crowded by vehicles. Most of the vehicles, which are taking small roads, are driven by local people which living in that region so traffic there is not big.

5.1. Path-finding

Before it's possible to generate any type of data, it's required to know which path a certain vehicle will take. This means, to avoid the vehicle to go through the same street more than once to the destination. A certain vehicle should stick to the shortest path in the road depending on the current situation. If chosen trip path is crowded by vehicles than there is a need to find another trip path (with the less traffic on the road).

The algorithm use to find the shortest path is *Dijkstra's algorithm[7]. Dijkstra's [7]* uses graph theory to find shortest path. It searching for all known path and if the path, it founds before, is shorter it replaces the path with the one it has already. After a few experiments it was decided that this algorithm is fastest and most efficient.

5.2. Road classification

In the traffic simulation tool, it is important to know different types of roads in sense of simulating shortest and less time-consuming path. Since known GPS data consist that kind of information, it is possible to group the roads:

- Big roads (highway)
- Middle roads (roads smaller than highway, but bigger than 500 meters)
- Small roads (roads 0> and <500 meters)

This classification was defined after looking at the data that was provided, in the shapefile related to North of Jutland of Denmark.

6. Vehicles

Vehicle in traffic simulation tool are modeled as a dot. The main goal of this program is to simulate traffic as realistic as possible and to tell user which road to choose or to watch critical situations on the roads (than there is too much vehicles on the road).

6.1. No overlapping vehicles

For the data to be realistic it is necessary to make sure that there is no overlapping vehicles. This should be done to all the population of the vehicles, by checking each of vehicles position and generating other vehicles according to it and leaving some space. If vehicles overlaps the event is treating like and accident and vehicles are destroyed (erase from a map and appear in random place in a map).

6.2. Actions

Vehicles have six actions which can be performed:

- Move forward
- Turn left
- Turn right
- Accelerate
- Decelerate
- Stop/Stay

Most important actions are acceleration and deceleration. These two actions will make sure that vehicles would not overlap, stop in a right time and start to move than there is a need. The problem is to decide than and where to start to stop the vehicle before intersection. Movements *turn left* and *turn right* are used only in between. If the road will be straight turn movements will not be used at all. So, only four moves of six will be used in every trip.

6.3. Speed

The signals that are generate must make sense, it won't be possible a sequence of signals that represent a vehicle riding 500km/h in the city center or that in a second there in one point and in the other second 50km ahead. It's required a special attention while generating the data so that this don't happen and this is easily avoidable.

7. GPS data

Data which can tell the whole trip path calls GPS data. GPS data can be divided into two types:

- Real data
- Synthetic or realistic data

Real data generated by GPS tool. That kind of data is closest to real life data, which everyone tries to achieve. Unfortunately for now it is impossible to get real GPS data. Real life data is extremely detail, it includes all real life factors like drivers mood, health, weather, wind, etc. So, basically, real GPS data type is very detailed synthetic GPS data.

Synthetic or realistic GPS data is data generated by computer. This data is less detailed because it uses less real life factors. The reason is simple if this data would try to use too much detail computers wouldn't be able to handle it or it would require extremely powerful computer. For more detailed generating of synthetic GPS data read in a section below.

In this project, the main goal is to generate synthetic data with as much details as possible to make received information as realistic as it could be.

7.1. Generating GPS data

Since this project main goal is to generate synthetic data as realistic as possible that is why in this section it would be talked about generating of synthetic data.

GPS data generating starts with shapefile (.shp) gained from GPS tool. All data inside is represented in UTM[2] format. To generate PGS data from given shapefile first of all there is a need to insert all data in SQL database. In order to do that, conversion from .shp file to .sql file is needed. There are two methods to perform that kind of conversion:

- Use special programs
- Use queries and scripts

Since first method is easier and less confusing, it was chosen. After shapefile importation in SQL database is finished, all is left, is to generate GPS data. In order to do that there is a need to use various algorithms. One of them is *Dijkstra's algorithm* mentioned before. This algorithm is most important to generate GPS data. Besides Dijkstra's algorithm, there is a need to use some additional SQL database library like pg_routing.

7.2. GPS data accuracy

For GPS data to be realistic some percent of data should be a little bit inaccurate. This is because on foggy and rainy days real GPS transmitters cannot send very accurate data too. This is easy to implement. It can be done just by adding some small random value to some percentage of the data.

8. Program management

8.1. Trip logic

Since this work is related in simulating the trips of a large group of people in their vehicles, it

---

2 UTM – is the Universal Transverse Mercator, which uses global measure unit meters. World map is divided in squares, and moves like simple coordinates (x,y) from the start coordinates. [0]

doesn't make sense to have random generated departures and arrival points and so it is required that had a certain logic in it. So that in the morning our data will simulate the departure of the people leaving their homes and not leaving their workplace. Even with exceptions it's important to simulate this even for a small part.

8.2. Program core

Pseudo code: "Main Algorithm":

Start:

{Input}

```
UserInput user_input = new UserInput() //All the information given by the user
```

{Arguments}

```
Vehicles vehicles = new Set(Vehicles, user_input.getNumberOfVehicles() );
double total_duration = user_input.getDuration();
Zones origin_zones = user_input.getOriginZones();
Zones destination_zones = user_input.getDestinationZones();
double frequency = user_input.getFrequency();
```

{Phase 1}

```
Foreach(Vehicle v in vehicles){
	Zone origin_zone = SelectRandomZone(origin_zones);
	Zone destination_zone = SelectRandomZone(destination_zones);
	GPSSignal origin_position = SelectRandomPositionIn(origin_zone);
	GPSSignal destination_position =
SelectRandomPositionIn(destination_zone);
	v.shortestpath = ShortestPath(origin_position, destination_position);
}
```

{Phase 2}

```
int time = frequency;
While(time <= totalDuration){
	Foreach(Vehicle v in vehicles){
		v.move(time);
	}
	time += frequency;
}
```

{Output}

```
OutputUtil.writeKML( Vehicles.getTrips() );
OutputUtil.saveToDatabase( Vehicles.getTrips() );
```

:End

9. Conclusion

Using Postgis with support of Java, is possible to generate, a simulation of GPS signals of vehicles.

It is also possible to create an algorithm that simulates the traffic of *n* vehicles during a period of time.

Since PostGis is a module for Postgres, the performance of the queries can be and are improved to a accepted running time and this show us that it is possible to simulate a large number of vehicles in a long period of time. Just by using the tools that already exist.

That it not required, to create a large program, to be able to simulate the traffic and it is possible with a small dimension program to generate GPS signals that simulate realistically the movement of vehicles.

9.1. Limitations and Future work

This solution, is database dependent and it is required a special attention, when adding new functionalities, so that they do not reduce the performance. While there were not applied some restrictions to the movement of the vehicles, for example, acceleration or vehicles stopping at traffic lights. These can be implemented by adding modules to the core program.

This project if continued to be work on, can turn to be a framework for scholars or users, experimenting with traffic simulation. Since this is an open-source project, a community can grow around this work. Because this topic of traffic simulation is always possible to continue to improve.

The result would be a framework that organizations small or big would use with a very reduced cost. Without having to turn to gathering GPS signals from devices installed on real vehicles.

10. Reference

References

1: Todd Litman, The Costs of Automobile Dependencyand the Benefits of Balanced Transporta,
2: , Postgis Webpage, , http://postgis.refractions.net/
3: Peter Jones and Arild Hervik, Restraining car traffic in European cities: An emerging role for road prici, 1992
4: Sewall, J.; Wilkie, D.; Merrell, P. and Lin, M. C., Continuum Traffic Simulation, 2010
5: Jonathan Ruttle, Virtual Traffic Simulation, 2008
6: Burmeister, B.; Haddadi, A.; Matylis, G., Application of multi-agent systems in traffic and transportation, 1997
7: , Dijkstra's algorithm, , http://en.wikipedia.org/wiki/Dijkstra's_algorithm
8: , Universal Transverse Mercator coordinate system, 2011, http://en.wikipedia.org/wiki/Universal_Transverse_

Appendix