Proceedings of the 10th International Conference "Reliability and Statistics in Transportation and Communication" (RelStat'10), 20–23 October 2010, Riga, Latvia, p. 440-446. ISBN 978-9984-818-34-4
Transport and Telecommunication Institute, Lomonosova 1, LV-1019, Riga, Latvia

MACROSCOPIC TRANSPORT MODEL DATA 3D VISUALISATION USING KML

Mihails Savrasovs, Arnis Rudzitis

Transport and Telecommunication Institute
Lomonosova 1, Riga, LV-1019, Latvia
Ph.: +371 29654003. e-mail: mms@tsi.lv

Visualisation and animation play a big role in simulation. They could be used for data presentation and even for model calibration. Regarding transport models, mainly visualisation and animation are applicable for models created at the microscopic level. Both 2D and 3D animation could be constructed and used for presentation purposes. Macroscopic transport models at the moment mainly use only 2D data representation. On the one hand this does not overload the model with details, while on the other hand information presented in 3D is still perceived very well by the human brain.

The main goal of this paper is to demonstrate the possibility of macroscopic model 3D data visualisation using modern information technologies. The demonstration is done using transport model development simulation software PTV VISION VISUM [1] and keyhole markup language (KML) [2]. The Google Earth application is used as the representation environment. The aforementioned tools and technologies were selected for the following reasons: VISUM is a modern simulator which supports COM interface and scripting; KML is an elegant and powerful XML based language for displaying geographical data and visualisation; Google Earth supports KML and provides a flexible environment for visualisation and animation. The process of macroscopic data visualisation could be presented in the following stages: 1) simulation model execution, to obtain data; 2) definition of the object and data 3) script running for KML file construction; 4) KML file loading into Google Earth application; 5) browsing of data.

Keywords: *3D, macroscopic transport model, KML, visualization*

1. Introduction

One of the big advantages of a simulation is the possibility to visualise a processes which happens in a system and to present output data in graphical form. Usually, when talking about animation and visualisation of the results in graphical form we must think about dimension. At the moment mostly 2D and 3D is used for animation and visualisation purposes. This is connected with the limitations of the human brain to perceive graphical data. All popular professional universal simulation tools like ExtendSim, Arena, etc. could produce both 2D and 3D animation during simulation. Going deeper, possibility of animation and visualisation in 2D and 3D is now included in the list of requirements for simulation software [3].

At the same time, in transport modelling we can distinguish microscopic, mesoscopic and macroscopic models. Microscopic models are developed at a level of high detailisation, which is why it is almost standard to have the possibility to animate and visualise in 2D and 3D. The figures below demonstrates 3D animation of the model created in VISSIM simulation software. Also at the same time VISSIM can produce 2D animation (figure 2).

Figure 1. 3D model of transport node

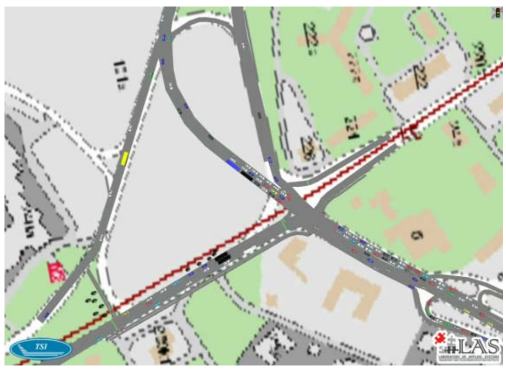

Figure 2. 2D model of transport node

Mesoscopic and macroscopic transport models mostly use 2D visualisation because of the high aggregation level of output data. Most of the well known macroscopic simulation tools like EMME/2, VISUM, CUBE etc present output data in 2D. An example of such presentations can be seen in figure 3.

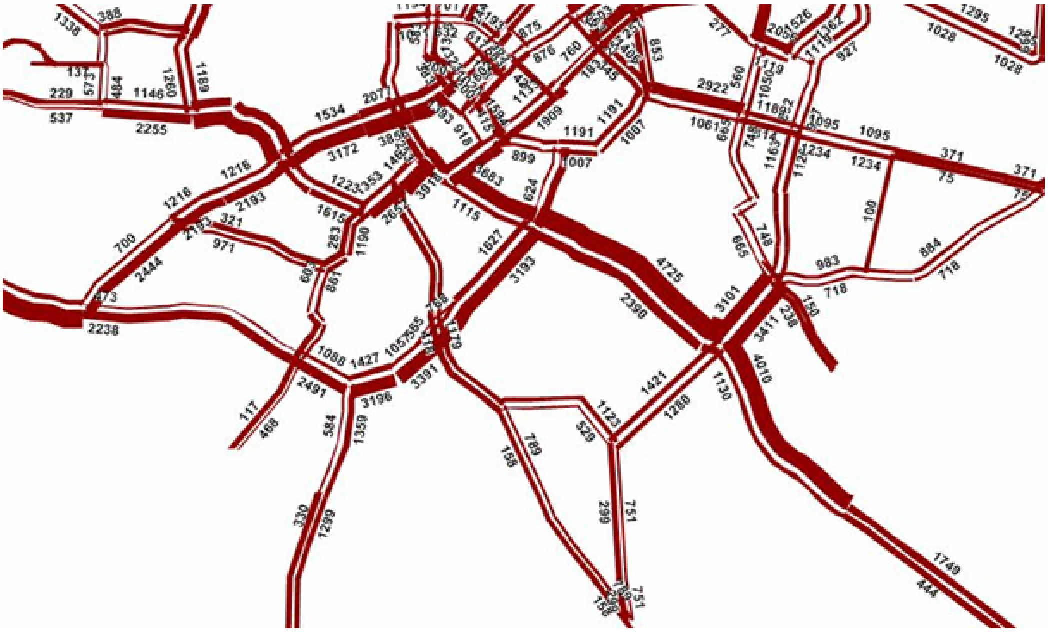

Figure 3. An example of data presentation in macroscopic models (traffic volumes)

Such kinds of data visualisation and presentation unfortunately have a lot of disadvantages. PTV VISION VISUM software can be used to show these disadvantages:

- Limitations on browsing simulation results without simulation software – this point means that simulation results can be browsed without limitations only with the software. If the results must be presented to some other people, who do not have software, pictures and shape files must be used. Unfortunately pictures and shape files are not interactive enough for browsing.
- Limitations on data sharing across users – if it is necessary to share results across users a lot of image and shape files must be sent to all users. First of all these files are not interactive enough, second, users will be overloaded with information.
- Limitations on data presentation – simulation results are presented as 2D diagrams, but as previously stated, the human brain can perceive 3D data presentations without problems.

These three limitations lead us to propose some tools or add-ins for traffic simulation software. Of course, each software has its own architecture and object hierarchy. This is why the example presented will be based on PTV VISION VISUM software. This software is well known across traffic simulation professionals and is one of the leaders in this field.

2. Keyhole Markup Languague

Analysing the previously described limitations of the traffic flow simulation software for data presentation, visualisation and sharing, it was decided to use the keyhole markup language.

Keyhole Markup Language (KML) is an XML-based language schema for expressing geographic annotation and visualization on Internet-based, two-dimensional maps and three-dimensional Earth browsers. KML was developed for use with Google Earth, which was originally named Keyhole Earth Viewer. It was created by Keyhole, Inc, which was acquired by Google in 2004. The name ‘Keyhole’ is homage to the KH reconnaissance satellites, the original eye-in-the-sky military reconnaissance system first launched in 1976. KML is an international standard of the Open Geospatial Consortium. Google Earth was the first program able to view and graphically edit KML files [4].

A KML file specifies a set of features (place marks, images, polygons, 3D models, textual descriptions, etc.) for display in Google Earth, Maps and Mobile, or any other 3D Earth browser (geobrowser) implementing the KML encoding. Each place always has a longitude and latitude. Other data can make the view more specific, such as tilt, heading and altitude, which together define a ‘camera view’[4]. A KMZ file consists of a main KML file and zero or more supporting files that are packaged into one unit using a Zip utility, called an archive. The KMZ file can then be stored and emailed as a single entity. A NetworkLink can fetch a KMZ file from a web server. When the KMZ file is unzipped, the main .kml file and its supporting files are separated into their original formats and directory structure, with their original filenames and extensions. In addition to being an archive format, the Zip format is also compressed, so an archive can include only a single large KML file. Depending on the content of the KML file, this process typically results in 10:1 compression. A 10 Kbyte KML file can be served with a 1 Kbyte KMZ file [5].

```
<?xml version="1.0" encoding="UTF-8"?>
<kml xmlns="http://www.opengis.net/kml/2.2">
  <Placemark>
    <name>Simple placemark</name>
    <description>Attached to the ground. Intelligently places itself
       at the height of the underlying terrain.</description>
    <Point>
      <coordinates>-122.0822035425683,37.42228990140251,0</coordinates>
    </Point>
  </Placemark>
</kml>
```

Figure 4. An example of a simple KML file

All described above, concerns that use of KML can successfully solve all the limitations:

- KML or KMZ files can be browsed using free software, which can be downloaded from the Internet. One of the most popular is Google Earth. Even an internet browser can be used by installing Google Earth add-in. KML and KMZ files can store enough information to make it interactive.
- Data transmission to users using KML and KMZ files is very simple. To view all data only one KMZ file need be sent by e-mail or put into a FTP server for downloading. In case of data high change time, the KMZ file can be launched directly from a website using a browser and played using Google Earth plug-in for browsers.
- Using KML, data can be presented in 2D and 3D.

3. Macroscopic Transport Model Data 3D Visualisation

The process of macroscopic transport model data visualization and presentation can be described in the following stages: 1) simulation model execution, to obtain data; 2) definition of the objects and data 3) script running for KML file construction; 4) KML file loading into Google Earth application; 5) browsing of data. The figure below (figure 5) demonstrates the general workflow.

The following comments can be given for each step of the workflow.

1. *The first step of the workflow is a simulation model execution or a model file launching*. This step is natural, because the data must be obtained before it can be visualised.,. This can be done by executing a model. However, the structure of the simulated system (in our case a transport system) could also be interesting for presentation. This is why the second part of this stage is only opening the model file in simulation software.
2. *The definition of the objects and data which should be visualised and presented.* Macroscopic models have a number of objects such as links, nodes, zones, stops, etc. The objects for visualisation must be selected. Each object has a number of parameters. Some parameters require model prior execution to be filled in with data; some of parameters are filled during system structure construction. The examples of possible parameters could be drawn here:
   a. Links – geographical location and geometry, volume of traffic, capacity, loading level, allowed speed, allowed transport systems, user parameters etc.;
   b. Nodes – geographical location, traffic turning volumes, user parameters etc.;
   c. Zones – geographical location and geometry, centre, name, connectors, area, user parameters (population, number of work places and so on) etc;
   d. Stops – geographical location, passengers volumes, catchment area, passengers transfers, user parameters, etc.

   To define objects and data a special interface should be provided for users. Also this interface should include the possibility of colouring different data and drawing different types of charts.

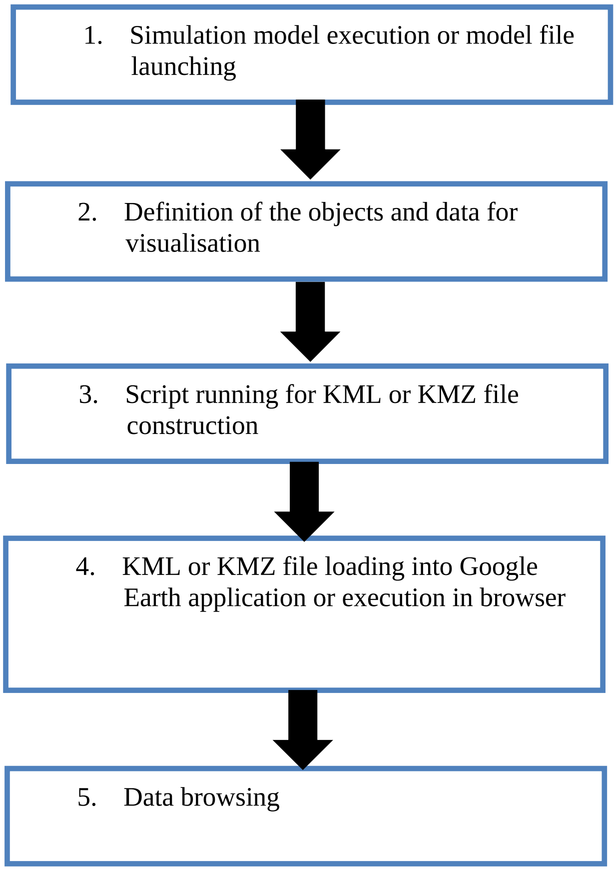

Figure 5. **Main workflow**

3. *A KML or KMZ file should be constructed according to selected objects and parameters*. The files by themselves could be generated using two possibilities:
    a. Using an external application written in any programming or scripting language. This is possible, because VISUM has a COM interface. The advantage of this is that you can use any programming language which you know. The disadvantage is slow communication via COM interface with VISUM.
    b. Using VISUM add-ins written in Python scripting language. The advantages are module integration into VISUM, which allows you to it in any time without launching any external application and a higher communication speed with VISUM than in first case. The disadvantage is that you are restricted to using Python as the scripting language.

   If it is necessary to generate a KMZ file, the archiving tool must be applied from the programming environment.
4. *KML or KMZ file loading into Google Earth application or execution in browsers.*
5. *Data browsing*. Data browsing can be done using the Google Earth application or the Google Earth plug-in for browsers. Of course third party software which can read KMZ or KML files can be used.

4. Case-Studies

In this part of paper a number of examples of using KML for visualisation are presented. For demonstration the algorithm described above was used.: PTV VISION VISUM 11.0, Python and Google Earth Application wereused as tools.

Case-study 1

The first case-study presents the results of a visualisation of 15 minute catchment areas for public transport stops in the city of Riga. The stops are the centres of blue circles. This example does not separate stops by mode of public transport. The visualisation can be seen below (Fig.6)

Figure 6. Visualisation of catchment areas of public transport stops using

Case-study 2

The 2nd case-study demonstrates macroscopic data for Pfullingen (a small German city). The following information is presented in 3D visualisation: circles show catchment areas; the colour of the circle shows the type of public transport which uses the stop (blue colour- bus stops, red – train stops); the height of the cylinder is the number of passenger travelling from this stop, the radius of the base is the catchment area. The visualisation can be seen below (Fig.7)

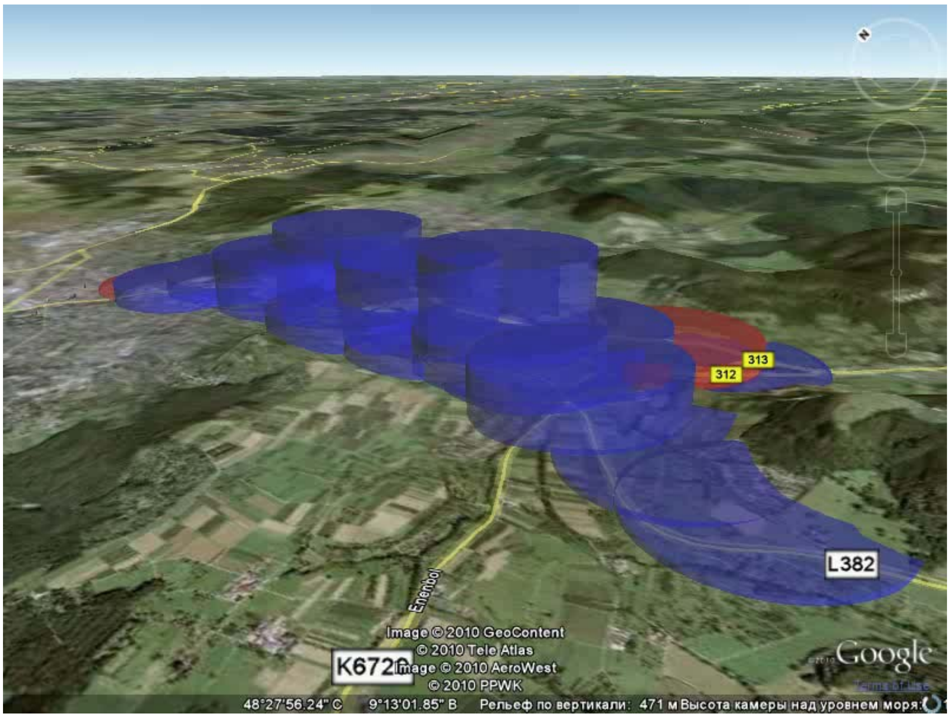

Figure 7. Visualisation of catchment areas, number of passengers and public transport type

Case-study 3

The 3rd case-study shows the volume of traffic. As in the previous example, the visualisation was constructed for Pfullingen. The data is presented for each link of the model. The height of the parallelepiped shows the volume of traffic travelling via a concrete link. The visualisation can be seen below (Fig. 8).

Figure 8. Traffic volume visualisation

Case-study 4

The last case-study demonstrates the loading level of crossroads, according to the LOS. The height of the cylinder shows the delay time of journeys via each crossroads, and the colour shows the level of LOS (red- critical LOS level, yellow and green – heightened and normal LOS). The visualisation was constructed for selected crossroads in the city of Riga.

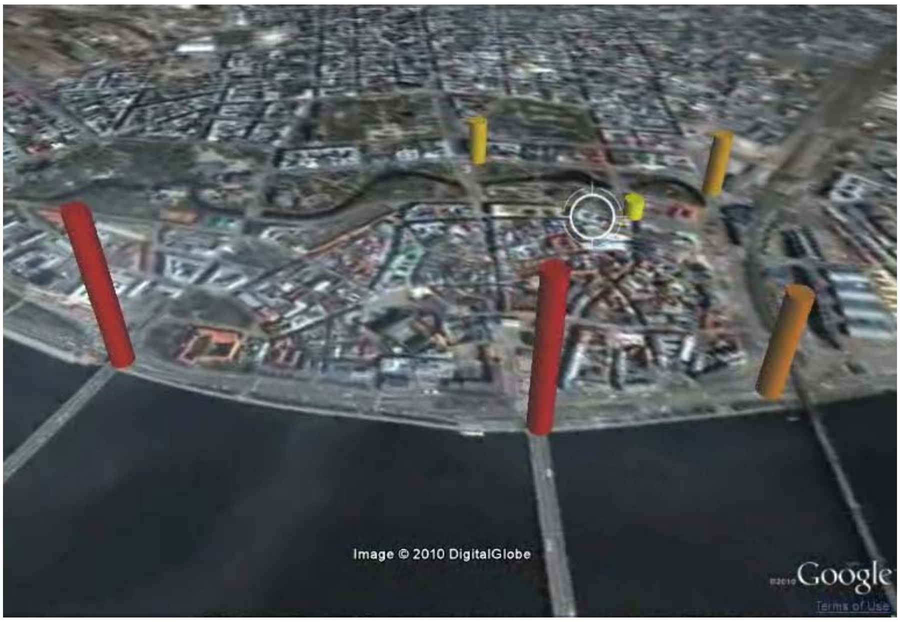

Figure 9. LOS level and delay time visualisation

5. Conclusions

- One of the advantages of a simulation is the possibility to present processes and data in visual form via animation and graphical representation in 2D and 3D format.
- Classically macroscopic transport models present results in 2D format. However, 3D can be used because information presented in 3D is still perceived very well by the human brain. Most modern software for macroscopic simulation does not have the possibility to present data in 3D.
- This article demonstrates the possibility of using KML (keyhole markup language) for 3D visualisations of the data of the model. Examples of using KML for visualisation are presented using PTV VISION VISUM software, Python scripting language and Google Earth application.
- Such an approach of data visualisation can help to solve some limitations of existing modern simulation software. These include limitations on browsing modelling results without simulation software; limitations on data sharing across groups of users and limitations on data presentation.
- The development of the examples revealed some problems which were successfully solved in the frame of this article. The following problems are mentioned here:
    - Longitude and latitude coordinates must be used for the positioning of an object. Despite the fact that most simulation software has the possibility to use different types of coordinates, many transport models are constructed in the scale of software coordinates. This problem was solved by the development of a small module which maps the software coordinates with longitude and latitude. Of course this operation requires manual data entering.
    - KML file format does not allow geometric shapes to be inputted directly. The geometric shapes can be constructed as a sequence of lines. For example, to construct a circle, a number of lines should be used. Such a situation of course makes the process of visualisation not so simple. The problem was solved successfully by integrating special free Python library, which allows geometrical shapes to be constructed in more simple notation (for the circle: centre and radius).

Acknowledgments

The article was written with the financial assistance of the European Social Fund. Project Nr. 2009/0159/1DP/1.1.2.1.2/09/IPIA/VIAA/006 (The support in Realisation of the Doctoral Programme "Telematics and Logistics"of the transport and Telecommunication Institute).

References

1. PTV VISION VISUM Official Manual, 2008.
2. KML Tutorial - http://code.google.com/intl/en EN/apis/kml/documentation/kml_tut.html
3. Banks, Jerry, Carson II, John S., Nelson, Barry L., Nicol, David M.. *Discrete-Event System Simulation*. 3th Edition. Prentice Hall, 2001. 594 p.
4. http://en.wikipedia.org/wiki/Keyhole_Markup_Language
5. http://code.google.com/intl/lv-LV/apis/kml/documentation/kmzarchives.html