

*Proceedings of the 11th International Conference “Reliability and Statistics in Transportation and Communication” (RelStat’11), 19–22 October 2011, Riga, Latvia, p. 1-9. ISBN 978-9984-818-46-7
Transport and Telecommunication Institute, Lomonosova 1, LV-1019, Riga, Latvia*

3D VISUALIZATION OF AIR QUALITY DATA

¹Roberto San José, ¹Juan Luis Pérez, ²Rosa Maria González-Barras

¹*Environmental Software and Modelling Group, Computer Science School
Technical University of Madrid
Campus de Montegancedo, Boadilla del Monte, 28660 Madrid, Spain*

²*Department of Meteorology and Geophysics
Faculty of Physics
Ciudad Universitaria, 28040 Madrid, Spain*

**Corresponding author: E-mail: roberto@fi.upm.es
Ph.: +34 913367465. Fax: +34 913367412*

The air quality models produce a considerable amount of data. In order for this data to provide useful insight into the workings of the atmosphere it must be visualized in a form that users can readily interpret. Large masses of spatial data have to be managed by the visualization system. The data source can be different, so different file formats have to be supported, for example, netcdf, grid, ascii, etc. New visualization tools have been developed to specific air quality modelling, with only the specific set of functionality needed for air quality analysis. Some of them (VIS5D, VAPOR, VRML, IDV, CITYGML) will be analysed in this paper.

We will show new tools developed by the authors to display air pollution over urban areas. The first one uses the Google Earth technology and the Ferret software together to display the buildings and the air pollution over Internet. Buildings are modelled as 3D Polygons following the COLLADA standard format. Air pollution is displayed like raster images generated by FERRET. All information is included in a KML file. The second one uses the Geographic Information System GvSIG with a 3D extension. It is open source software under GNU/GPL license. The buildings are modelled using the COLLADA standard and textures are included to get realistic images. Air pollution is modelled as spheres with a colour scale and transparency levels.

And finally, the last one tool is based on VRML and it allows displaying buildings as IndexedFaceSet nodes and air pollution as isosurfaces. They are generated using the Marching Cubes algorithm to transform 3D grid data to isosurfaces.

Keywords: 3D Air pollution, Visualization, VRML, GvSIG-3D, COLLADA

1. Introduction

Air pollution modelling is an attempt to describe the functional relation between emissions and occurring concentrations and deposition. Air pollution modelling can produce an analysis of the causes, which have led to these concentrations. Air pollution models play an important role in science, because of their capability to investigate the importance of the relevant processes. Air pollution models are the only method, which can quantify the relation between emissions and concentrations, including the consequences of future scenarios and the determination of the effectiveness of abatement strategies. Considerable work has been done in the area of air quality modelling and computational science to develop high quality air quality modelling systems.

The air quality models produce a considerable amount of data. Raw data can be hard to conceptualise, particularly when the size of the data sets is terabytes. In order for this data to provide useful insight into the different works in the atmosphere, it must be visualized in a form that users can readily interpret.

Visualization systems have been used in the air quality modelling community for some time to make sense of the output from computational mode [1]. However, problems can arise when trying to interpret the results of the simulation. The visualization tools were designed as general visualization systems, to be used for everything from air quality modelling to the design of jet engines. Since they were designed as general purpose systems, only a small subset of the functionality was ever used for Air Quality Modelling. Now, new visualization tools have been developed to specific air quality models, with only the specific set of functionality needed for air quality analysis. Some of these tools will be analysed in this paper.

Air quality visualization has special requirements respect to other kind of information. Large masses of spatial data have to be managed by the visualization system [2]. The data source can be different, so different file formats have to be supported, for example netcdf, grid, ASCII, etc.

A system for air pollution data exploration should ensure the support of all existing dimensions (three spatial, one temporal, multiple thematic and possible scenarios). The presentation and visual analysis of air pollution data cannot be done without geo-referenced coordinates. Multiple visualization techniques should be included in a suitable software solution. For the purposes considered, the most common shading methods, texture mapping facilities, generation of isolines and isosurfaces, slicing functions, display of vector fields, flow visualization, colour editing and reclassification functions, glyph representations, volume rendering and animation are needed. The appropriate use of visualization methods should be assisted by help tools.

Different types of legends, scale bars, frames, north arrows, and textual explanations should be part of every system's functionality, but are seldom implemented. Another important feature is a function which assists the user in choosing a meaningful colour or symbol legend compliant with the values that you are displaying.

A decisive criterion for the usability of visualization systems is often found in the implemented strategy for human computer interaction, above all concerning 3D-navigation [3]. During the mapping of 3D spaces to 2D screens, problems occur similar to those in traditional mapping. Additionally, interaction and navigation play an important role. Navigation is conventionally done by mouse movement or keyboard input.

In order to rapidly explore these large masses of spatial data and to quickly detect trends, correlations, and anomalies, the human visual sense is highly suitable. Thus, the underlying geoscientific processes can be recognized more easily by using graphical representations of numerical data than by analysing the raw data itself. The scientific discipline of Visualization in Scientific Computing (ViSC) offers a collection of methods, which can be used for this purpose Hearnshaw, H.M., Unwin, D.J. (Eds.), 1994. Visualization in Geographical Information Systems. Wiley, Chichester, 242 pp.

2. Generic Visualization Tools

Vis5D was the first system for interactive visualization of large 5-D gridded data sets such as those produced by numerical weather models. One can make isosurfaces, contour line slices, collared slices, volume renderings, etc of data in a 3-D grid, then rotate and animate the images in real time. It was the first system to produce fully interactive animated 3-D displays of time-dynamic volumetric data sets and the first open source 3-D visualization system and provide semantic information too. It is possible to define Vis5D files as medium for exchanging information on the Word Wide Web. Links to Vis5D files can be embedded in Web pages (just as links to GIF files can be embedded in web pages), and Mosaic can automatically invoke Vis5D to view them. No further developing activities. Last version April-2002.

Figure 1 is an example of Vis5D visualization of isosurfaces and vertical planes of air pollution produces by the air quality system MM5-CMAQ run by the authors.

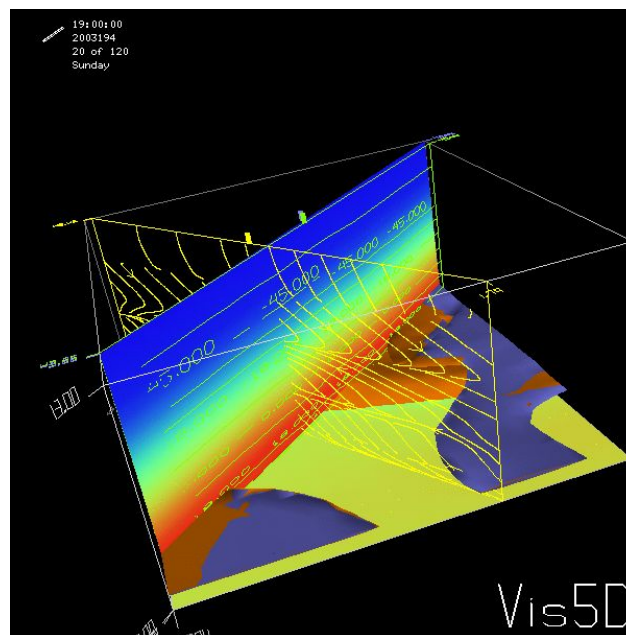


Figure 1. Example of visualization with Vis5D

VAPOR (Visualization and Analysis Platform for Ocean, Atmosphere, and Solar Researchers) is a software package developed at the National Center for Atmospheric Research in collaboration with U.C. Davis and Ohio State University. It can produce images and movies from very large mesh-based datasets, such as wind velocity and other physical fields in two and three dimensions [4]. VAPOR has its own input file format, VDF, but it supports conversion from other formats, such as NetCDF, in particular the files output by Weather Research and Forecasting model (WRF). It is free software but not an open source.

The Virtual Reality Modeling Language (VRML) is a file format for describing 3D interactive worlds and objects. It may be used in conjunction with the World Wide Web. It may be used to create three-dimensional representations of complex scenes such as illustrations, product definition and virtual reality presentations.

VRML is capable of representing static and animated objects and it can have hyperlinks to other media such as sound, movies, and image. Interpreters (browsers) for VRML are widely available for many different platforms as well as authoring tools for the creation VRML files. VRML supports an extensibility model that allows new objects to be defined and a registration process to allow application communities to develop interoperable extensions to the base standard. There is a mapping between VRML elements and commonly used 3D application programmer interface (API) features. Figure 2 is an example of VRML visualization of air pollution over surface and city building produces by the computational fluid dynamics model MICROSYS run by the authors over Las Palmas (Canary Islands).

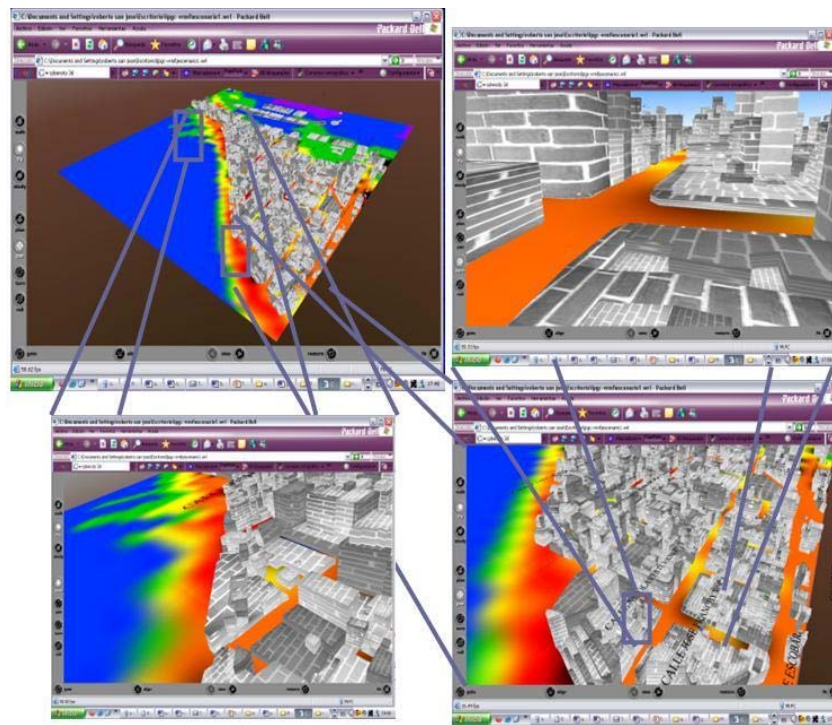


Figure 2. VRML visualization including 3d buildings and 2d pollution information over Las Palmas (Canary Island) developed by UPM

The Integrated Data Viewer (IDV) from Unidata is a Java software framework for analysing and visualizing geosciences data. IDV includes a software library and a reference application made from that software.

The IDV "reference application" is a geosciences display and analysis software system with many of the standard data displays that other Unidata software (e.g. GEMPAK and McIDAS) provide. It brings together the ability to display and work with satellite imagery, gridded data (for example, numerical weather prediction model output), surface observations, balloon soundings, NWS WSR-88D Level II and Level III RADAR data, and NOAA National Profiler Network data, all within a unified interface. It also provides 3-D views of the earth system and allows users to interactively slice, dice, and probe the data, creating cross-sections, profiles, animations and value read-outs of multi-dimensional data sets. The IDV can display any Earth-located data if it is provided in a known format. Figure 3 is an example of visualization with IDV.

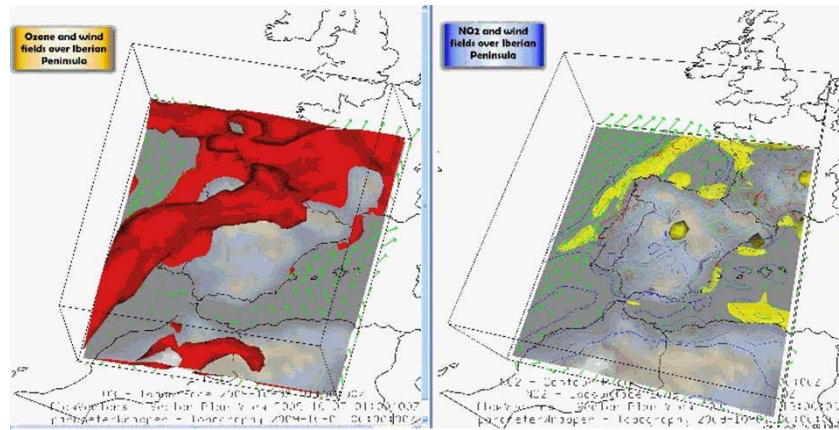


Figure 3. Example de visualization with IDV. Air pollution and Winds vector over Iberian Peninsula. Data from MM5-CMAQ model run by UPM

CITYGML is a standard that enables the creation and exchange of 3D data, applied to cities and the urban environment, as well as topography. CityGML is realised as an open data model and XML-based format for the storage and exchange of virtual 3D city models. It is implemented as an application schema for the Geography Markup Language version 3.1.1 (GML3), the extendible international standard for spatial data exchange issued by the Open Geospatial Consortium (OGC) and the ISO TC211. CityGML is an official OGC Standard and can be used free of charge. CityGML has 5 levels of representation defined, starting at level 0, appropriate to represent large areas with little detail, and ending at level 4, recommended for use inside buildings and areas with lots of detail. Furthermore, these representations can be combined without conflict, thus highlighting those areas most important representation.

Using CityGML is possible to represent the most important elements of the cities, as buildings and roads, but also the urban material, such as trees. The buildings receive a special treatment; CityGML can incorporate elements as windows, doors, decorative items, etc. It is also possible to incorporate the field and its features, using a dot notation, triangles or curves, as well as bodies of water, such as rivers and seas. CityGML allows the inclusion of generic objects, i.e., objects no defined in the standard and must be specified by the designer.

But CityGML not only has graphics features, also allows the inclusion of semantic information. Unlike other standards for representing 3D, CityGML defined classes used to build the city. Thus, a building is not just a set of polygons, but a set of polygons defined as walls, ceiling, floor, windows, doors, etc. Each element of the city has its own class (and a generic class for elements not defined in the standard), so that the buildings are defined as buildings, the roads are defined as road, etc. But also, with CityGML can add features of each element, such as year of construction of each building for example.

We have used this standard for the generation of cities, including buildings (with doors and windows), the ground and trees. We have added the pollution data in the form of soil colour, thus achieving a 2D representation. Using the standard features, different representations are been included, each indicated the expected values at a given instant of time. This value is indicated by the colour of the ground, and including the numerical value in soil characteristics. The final view can be viewed in Figure 4.

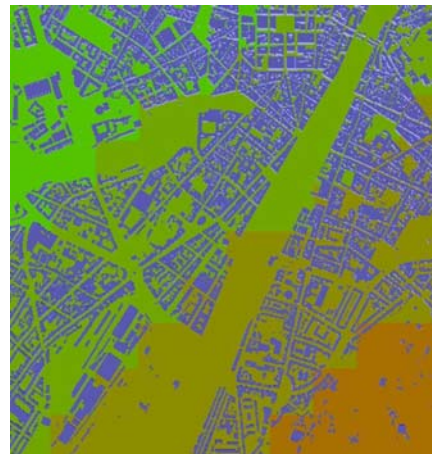


Figure 4. CityGMI visualization including 3d buildings and 2d pollution information over Florence (Italy) developed by UPM

3. Specific Visualization Tools

Now we go to explain specific tools developed by the authors to display air pollution over urban areas, which have been integrated in the simulation systems.

COLLADA-FERRET IN GOOGLE EARTH

Google Earth is a 3D viewer, developed by Google. Google Earth shows a 3D representation of the world, where you can see various elements such as pictures of prominent places, explanations on the environment, political maps, 3D object, animations, etc Google Earth. Also allows the incorporation of 3D buildings, adding its location in the world. It also allows the incorporation of animation, a number of models along the time the time, which may indicate the actual dates and times.

Google Earth uses KML or KMZ files. Kml file is a file which includes the geographic location data information to be viewed, as well as data concerning dates and times of the animations. Kml file may also include links to images or Collada files, which contains a 3D model. In this case, Google Earth reads and interprets the contents of the Collada file and sets it to the location defined in the Kml file. However, Google Earth defines their own animations, so you cannot show the animations included in the Collada file. These animations are based on a series of moments, which defines the elements that are appropriate at each moment in each point highlighted on the map.

Kmz file is a zip file of a set of files that make up a single model. The first file in the zip file must be a Kml file, because the application uses this to set reference points. The Kmz file may also include all the images and Collada files needed for visualization. Thus, the zip file is used to form a common file with everything you need for the representation of the model.

In our case we have created a 3D model of the city of Madrid, using the *Google SkechUp* software to become Shapefile files (GIS polygonal data) to Collada file, containing all the buildings in the area selected. This information is showed on the

Figure 5.

On the other hand, we have been obtained images on the area at different points in time, representing the evolution of temperature over a day in the selected area. These images were obtained with *Ferret visualization tool*., Subsequently a Kml file was created. This Kml file includes a link to the Collada file from the city of Madrid, and it incorporates the coordinates where it is. Also, we defined the image to display at every moment of time and location, giving rise to the animation. Finally, all these files have been compressed in a zip file, leading to a Kmz file. Google Earth is able to read this file, and using the coordinates located in Kml file, add the 3D model of Madrid and the first of the images. Google Earth incorporates the progress bar indicating the status of the animation and the dates indicated in the Kml file. The background image changes as the animation progresses.

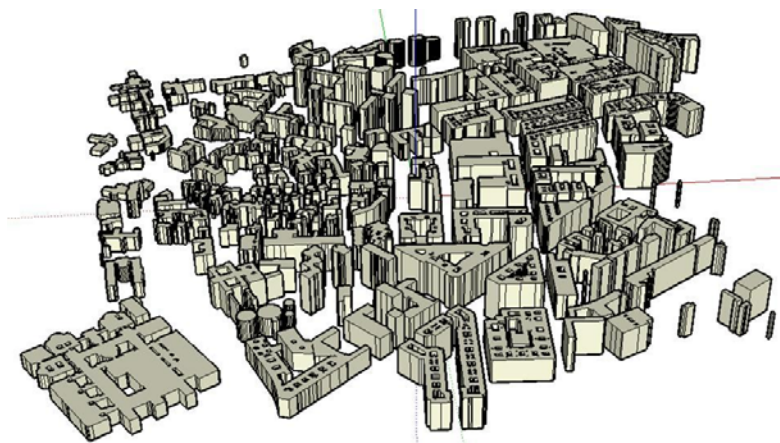


Figure 5. 3D buildings displayed using COLLADA. Madrid (Spain) area

The different tools that have been integrated to display buildings and atmospheric information in Google Earth are showed on Figure 6 and the flow chart of the integration is showed. And finally the results are showed on Figure 7:

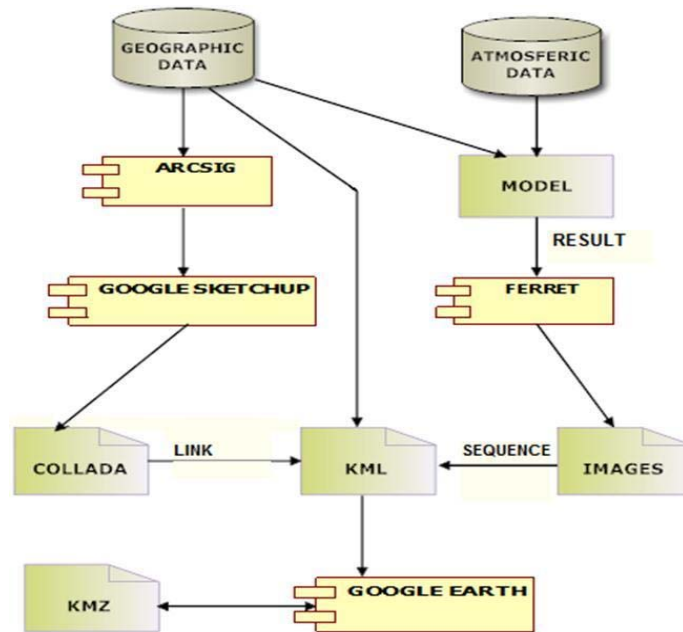


Figure 6. Flow chart to the integration of the different tools

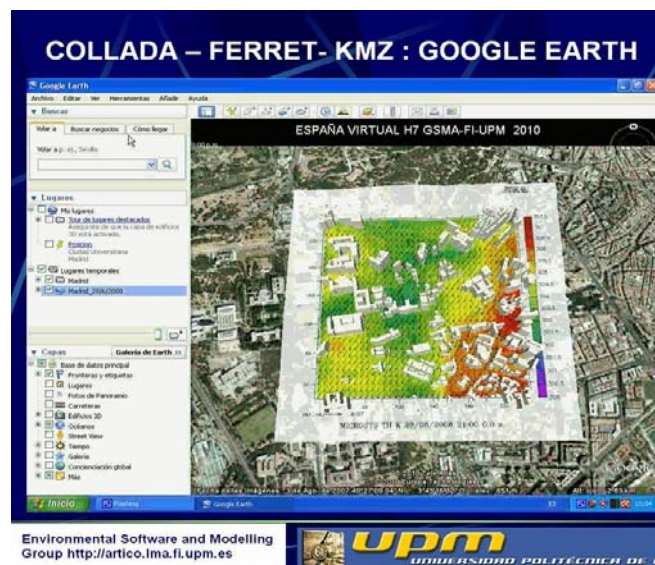


Figure 7. Google Earth visualization, combining FERRET and COLLADA

GvSig-3D

GvSIG is a friendly user Geographic Information System implemented in Java. It has the capability to read the major of vector and raster formats, allowing managing spatial data and perform different type of analysis process. It is a free and open source application, so developers can develop new modules, for example a new feature that allows the visualization of atmospheric data. GvSIG has an extension called 3D. It adds the possibility of working with 3D views and 3D objects, view and navigate through them. The core of the 3D extension is the framework OSGVP (OpenSceneGraph Virtual Planets). OSGVP is implemented in C++ and it was exported to API Java using JNI technology.

The first step is the 3D rendering of the city buildings. We use 2D geographical data to make a 3D model of the buildings, which will be showed in gvSIG-3D. COLLADA is the representation format selected. From the vectorial geographic database, using the ARGGIS software we can get the 3D model in skp format. The “skp model” is exported to a COLLADA format using the SketchUp tool. Finally to improve the 3D model of the building, a JAVA application has been developed to add textures to the buildings Figure 8. The final visual aspect is showed in the Figure 9.

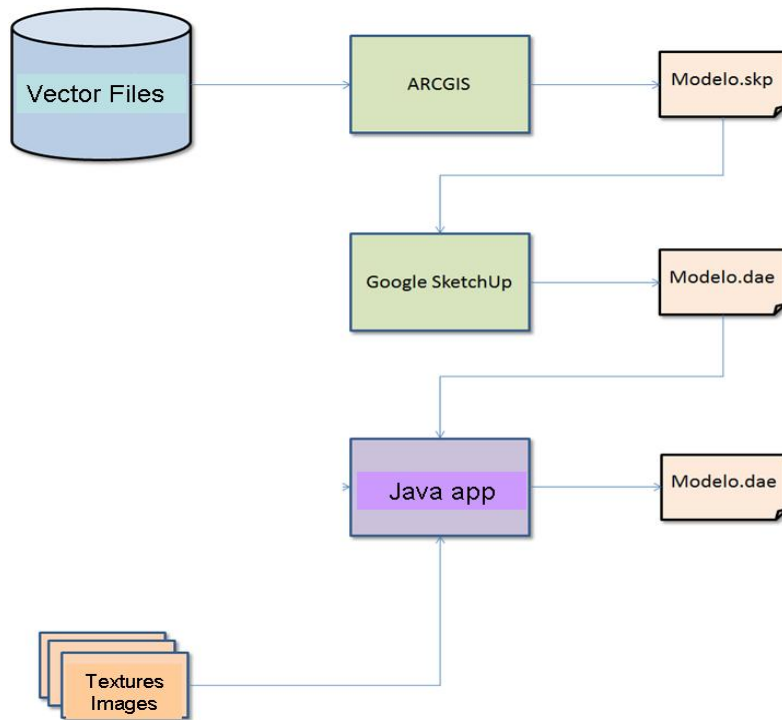


Figure 8. Flow chart of the application to produce 3D buildings in gvSIG-3D



Figure 9. 3D buildings in gvSIG-3D. Madrid area

The class “osgfeatures” allows drawing vectorial data: points, lines, polygons and spheres and other simple geometric figures. Different colour can be assigned to the figures and also can be applied a certain level of transparency. We have chosen the spheres as figure to represent the atmospheric data. Each grid cell data will be modelled by a sphere with a colour based on a scale. Transparency will be used to see the city buildings and the pollution.

The colour is set with RGB codes. To achieve smooth colour scale will be used a 3 linear functions, which give us the value of red, green and blue to each grid cell data. For the definition of the colour of linear functions we calculate the maximum, minimum and average value. The first section of the function is from the minimum to the average and the second from the average to the maximum.

Each 3D spatial data is represented by a sphere, which the main properties are the position and the colour. All spheres of the domain are grouped into a root node, which is added to the OSG layer, using the Java casting property

Now we show several examples of the representations obtained, data are based of simulations develop by us using the EULAG model over a Madrid area of 1Km by 1Km. The results show temperature data, with red colour to higher values and light green to the lowest values, from different points of view /see Figure 10/.

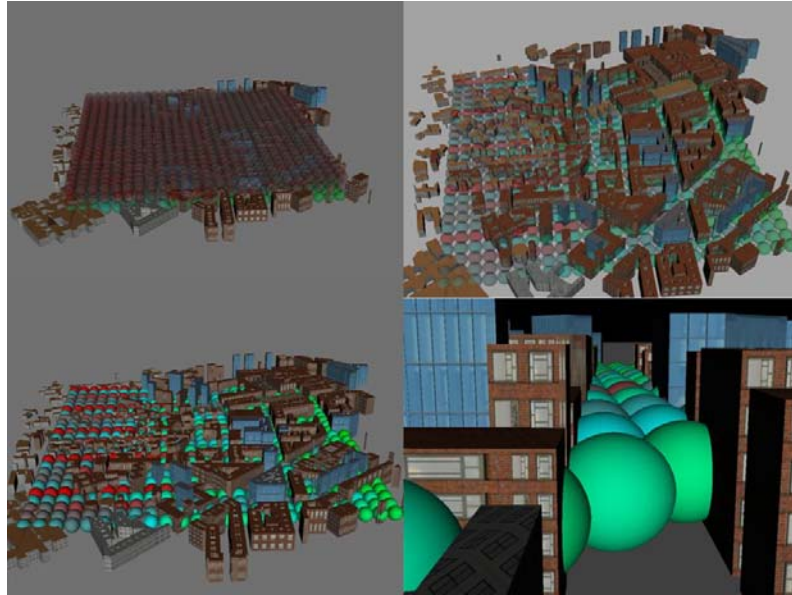


Figure 10. 3D temperature data and buildings in gvSIG-3D, 3 vertical levels (upper-left), surface (upper-right), without transparency (bottom-left) and zoom (bottom-right)

VRML-ISOSURFACES

To represent the city buildings in VRML we have used “IndexedFaceSet” nodes. This type of nodes is a collection of faces that allow building any type of geometric figures as buildings. One modelled all sides of the buildings using the node described above.

The isosurfaces have been generating using the “marching cubes” algorithm [5]. It is often used for building a polygonal mesh from 3D scalar data, as temperature or concentrations of pollutants. The modelling domain is spited into cubes with the corresponding value in each vertex. The algorithm runs through all cubes and if one or more corners of the cube have values less than the isovalue selected, then this cube is a part of the isosurface. Having determined the sides of the cube are intersected by the isosurface, the algorithm creates triangular polygons, which divide the cube into regions that are inside the isosurface and others are not.

In this phase, we have two components, buildings and isosurfaces, now the final step is the integration of buildings and isosurfaces to generate the final product, which can be viewed by the end users with the corresponding VRML viewer. An example of the final results is shown on Figure 11.

The user can navigate through the streets and see the distribution of the atmospheric data between the buildings as the Figure 12 shows.

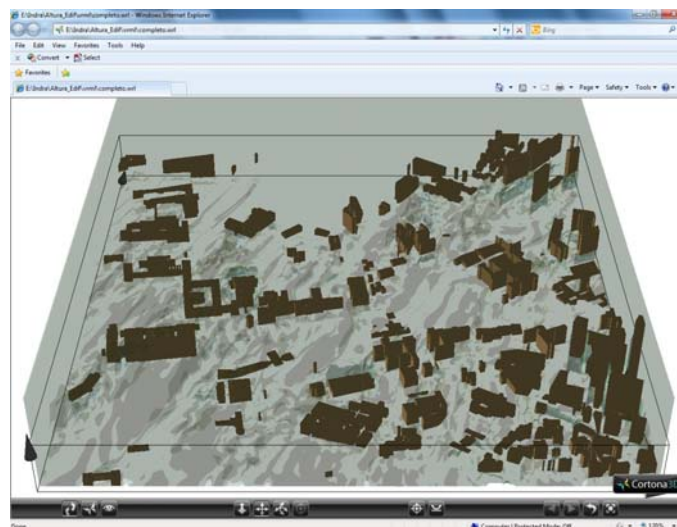


Figure 11. Example of visualization with VRML of buildings and isosurfaces of temperature over Madrid area

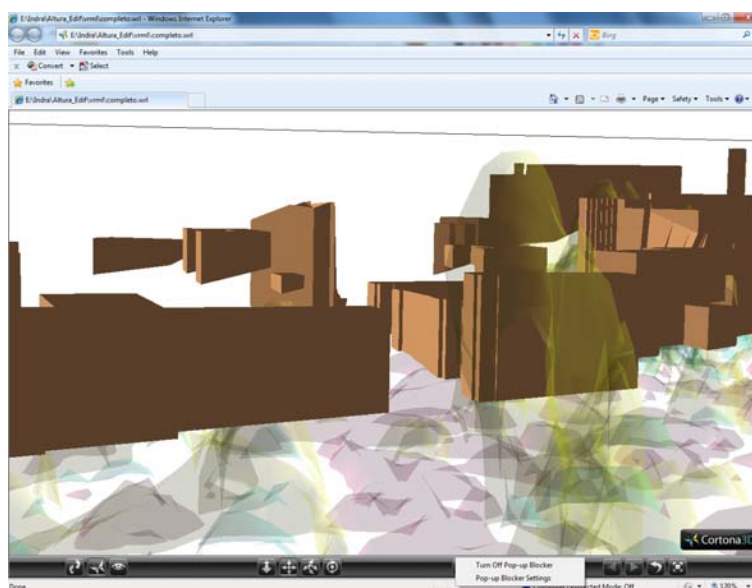


Figure 12. Example of interaction of the user with VRML

4. Conclusions

This paper provides several solutions to the atmospheric scientists for the analysis of large amounts of data, which are generated by the atmospheric simulation systems. Instead of terabytes of raw number they can utilize the 3D interaction and visualization capabilities provided by the different tools. Two groups of tools have been presented. The first one is generic tools that can be used directly with the simulation results. The second one is specific applications developed by the authors, which are the integration of different tools and technologies.

The second group has more focus on 3D urban environments, which combines urban morphology (buildings), and atmospheric data. In the case of the buildings, the COLLADA format is showed as one of the most suitable, because it can be used in different systems as gvSIG-3D and Google Earth. The atmospheric data representation is the most complex part. In this paper we develop two different ways to show atmospheric data, with spheres in gvSIG-3D and isosurfaces with VRML, respectively. The latter one has more possibilities, so this one will be more developed in the future.

As to future work in gvSIG-3D, we should improve the type of representation and try to get representations with isosurfaces or semi-transparent volumes. To achieve this objective it will be necessary to extend the osgVP library.

References

1. Dobashi, Y., Kaneda, K., Yamashita, H., Okita, T. and Nishita, T. A simple, efficient method for realistic animation of clouds. In: *SIGGRAPH '00: Proceedings of the 27th annual conference on Computer graphics and interactive techniques*. New York, NY, USA: ACM Press/Addison-Wesley Publishing Co, 2000, pp. 19–28.
2. Dykes, J.A. Exploring spatial data representations with dynamic graphics, *Computers and Geosciences*, Vol. **23**(4), 1997, pp. 345–370.
3. Chen, J., Fang, Y., Loftin, R., Leiss, E., Lin, C. and Su, S. An immersive virtual environment training system on real-time motion platform. In: *Proc. of the Computer Aided Design and Computer Graphics, part 2*, 2001, pp. 951–954.
4. Gruchalla, M. Rast, Bradley, E., Clyne, J., Mininni, P. *Visualization-Driven Structural and Statistical Analysis of Turbulent Flows*, Procs. IDA 2009: Lecture Notes in Computer Science 5772. Springer, 2009, pp 321-332.
5. Goodchild, M.F., Battenfield, B. and Wood, J. Introduction to visualizing data validity. In: *Visualization in Geographical Information Systems / H.M. Hearnshaw and D.J. Unwin (Eds.)*. New York: Wiley, 1994. pp. 141–149.