WEB BASED DYNAMIC 3D VISUALIZATION OF CITYGML DATA FOR URBAN SOLAR POTENTIAL

Amol Konde\textsuperscript{a} and Sameer Saran\textsuperscript{b}
\textsuperscript{a}Indian Institute of Remote Sensing, Dehradun, India, akonde22@gmail.com
\textsuperscript{b}Indian Institute of Remote Sensing, Dehradun, India, sameer@iirs.gov.in

Commission VIII, WG VIII/5

KEY WORDS: X3D, JSP, HTML5, X3DOM, CityGML, CityGML4j

ABSTRACT:
The energy requirements of the urban areas and industries are increasing continuously and pollution has become another important issue which gives rise to the need of renewable energy resources. Solar energy systems offer number of advantages over other energy resources as it is a pollution free renewable energy resource. So, this study focuses on the visualization of CityGML 3D model of the buildings and its solar potential attributes. Our approach is centralized on one idea of parsing CityGML objects to X3D scenes and objects at runtime for visualization of urban solar potential attributes. CityGML datasets work as a data store which is accessed via http/web-service requests to create dynamic X3D content as per users input from front end. The job of creating X3D content is done by JAVA beans developed by extending the JAVA classes provided in citygml4j JAVA API. The JAVA beans read CityGML data and map it to X3D content which is then used by JSP pages to create HTML pages at run-time. These web pages are eligible for visualization in HTML5 web browser environments.

1. INTRODUCTION

Currently there is a huge requirement for the interoperable and semantically reach representation models that can accommodate the additional attribute support to run different dynamic scientific models to calculate important features like solar potential, noise levels etc. for smart city objects like roads, buildings etc. and CityGML is one of the best solution to such problem. 3D city models conforming to the standard CityGML provide full coverage of city objects of the entire urban area, including geometry, properties, and the spatial arrangement of buildings, with ability of coherent integration of spatio-semantic attributes, which facilitates the integration of energy-related and ecologic-relevant information on different scales (A. Krüger et al., 2012).

CityGML is an interoperable OGC standard for the representation of 3D urban objects and is a profile of GML (3.1.1) (Peter A. Vretanos, 2006) that facilitates a multi-scale, multi-functional, and semantic 3D city model. It is realized as an open data model and XML-based format for the storage and exchange of virtual 3D city models. It represents both geometric and semantic properties of city objects. With its ability to adapt for different applications with Application Domain Extension, CityGML is still evolving and new attributes/contents have been added into it. CityGML Application Domain Extension provide a way to define application specific attributes and classes so that the results of running simulation models are supported by the CityGML Schema. This ADE schema can then be utilized by the researchers/users working in the similar application field to maintain the semantic and syntactic interoperability.

CityGML supports different Levels-of-Detail (LoD) (Mao, B. 2011), which may arise from independent data collection processes and are used for efficient visualization and efficient data analysis.

LOD0: This is the coarsest level of detail. This can be considered as a 2.5D Digital Terrain Model (DTM) over which an aerial image or a map may be draped.

LOD1: A well-known blocks model, without any roof structures. This level is used for city and region level coverage.

LOD2: Has distinctive roof structures and larger building installations like balconies and stairs. This level of detail is suitable for district/ city level projects.

LOD3: Denotes architectural models with detailed wall and roof structures, doors, windows and bays.

LOD4: This completes a LOD3 model by adding interior structures like rooms, stairs, and furniture.

Fig 1.1 CityGML Levels Of Details (Source: Wate P. et al., 2011)
X3D (Johannes Behr et al., 2009, Johannes Behr et al., 2011) is an ISO open standard file format for 3D visualization. It has architecture that supports real-time rendering of 3D scenes and objects across multiple platforms. X3DOM (Johannes Behr et al., 2009, Johannes Behr et al., 2011), the open-source prototyping effort provided with HTML5, integrates X3D scenes and objects within HTML5 supporting browsers so that the user can operate on 3D scenes directly and also without the need for browser plugin (Yvonne Jung et al., 2012). ‘citygml4j’ (Jack Cheng et al., 2013) is an open source JAVA API developed for manipulating CityGML datasets. It has functions and classes to read, analyse and edit CityGML datasets. JSP is well known for its dynamic web page capabilities.

2. ENERGY ADE

Energy ADE is a CityGML Application Domain Extension developed to support indicators and indexes of the urban energy systems. The Energy ADE was developed as an ADE for energy assessment that forms the basis of Energy Atlas Berlin data model (A. Krüger et al., 2012). It is developed to provide an interoperable standard between various energy simulations and their applications.

3. VISUALIZATION FRAMEWORK

The framework is proposed here by considering the extensibility of CityGML i.e. the Application Domain Extensions into account. The indicators and attributes calculated as a result of running different models and simulations and that are supported by CityGML ADE schema can be utilized to create instance document of the CityGML ADE. And at last the instance document can be visualized by implementation of the framework proposed here.

The implementation architecture of the proposed framework (fig 3.1) consist of three steps. The first steps involves reading the CityGML document, in the second step we generate the X3D graphic elements and the third steps renders the generated graphic elements on the web browser.

3.1 Reading CityGML

CityGML4j is a powerful API for reading, analysing and editing CityGML datasets. It provides classes that can be used to traverse the whole CityGML file for each feature. Also it provides function that can be used for query processing on CityGML dataset. Here we used this JAVA API to read the city objects members into java classes/beans. Once the city objects are read the processor java classes are invoked for each city object type to process the city objects. Also using CityGML4j API we can read and process the Application Domain Extension instance documents that adds the extensibility feature to the proposed architecture to add support for different Application Domain Extensions etc.

3.2 Generation of graphics elements

The Processors initiate the specific feature generators and call the generate method of generator class to create the 3D graphic elements for the features to provide W3DS services i.e. to return the X3D elements to the front end. At this stage when we generate the graphic elements different generalization algorithms can be implemented as per the requirement of the application to improve the performance so that the graphic elements are rendered faster. For generalization purpose various libraries are available like JTS, Computational Geometry Algorithms Library etc.

3.3 Rendering of graphics elements

With the advancement of X3DOM support in browsers like Google chrome that support X3D content in-line with the html5 tags, we could directly embed the X3D scenes in the web pages dynamically. This feature facilitates the users to fetch the required 3D content to the web browser without any additional plugin requirements. In our scenario the X3D graphic elements generated are rendered to the web browser using server side technology like JSP to dynamically generate the HTTP response. The X3D elements are added to the web page recursively at server side such that each having the on click event to execute the java-script function to display its attributes. Once the web page is ready, it is sent as http response to the front end where it is displayed on html5/X3DOM supporting browser.
4. RESULTS AND DISCUSSION

The output of the 3D visualization of the CityGML datasets using the implemented framework is presented in the figures 4.1 and 4.2. Fig 4.1 represents the tilted view of the test dataset and the Fig 4.2 represents the nadir view of the Test dataset.

Google chrome browser that supports html5 was used for visualizing the dataset. It was possible to execute the functions like zoom-in and out, pan, rotate around all the three axis. Here we used LoD1 and LoD2 datasets for visualization and assigned different colours for LoD1 and LoD2 buildings so that we can identify the LoD1 and LoD2 buildings distinctly. Use of HTML5/X3DOM supporting browser provided following advantages,

1. No Plugin required to visualize the dataset.
2. Easily execute the functions like Zoom-in, Zoom-out, Pan, and rotate.
3. Event based JavaScript functions for city Object can be easily developed etc.

In this case study, we were able to visualize the data in effective manner and simulate the buildings with different level of details (LoD1 and LoD2) as per the requirement.

Fig 4.1 Inclined view of the CityGML Dataset of the village Waldbruckeck, Germany (Source: CityGML Datasets)

Fig 4.2 Nadir view of the CityGML Dataset of the village Waldbruckeck, Germany (Source: CityGML Datasets).

5. CONCLUSION AND FUTURE WORK

5.1 Conclusion

The proposed framework suggests a plugin independent visualization framework for CityGML Datasets and it can be easily extended to support different CityGML Application Domain Extensions like Energy, Solar etc.

But, this framework still has some limitations which need to be addressed and tackled such as it is still impractical to use this for heavy datasets. LoD3 and LoD4 datasets are not supported by this framework.

5.2 Future Work

The future work will focus on the visualization of additional indicators and attributes generated as a result of running variety of simulations for various applications like Energy, Noise, Land Use/Land Cover etc. Also the geometric manipulations and generalization functions will be added to the framework to improve the performance and also to visualize LoD3 and LoD4 Models using Java Topological Suit (JTS).
REFERENCES


