TOWARDS AN AUTOMATED HEALING OF 3D URBAN MODELS

J. Bogdahn [a], V. Coors[b]

[a] University of Strathclyde, Dept. of Electronic and Electrical Engineering, 16 Richmond Street, Glasgow G1 1XQ UK - jurgen.bogdahn@strath.ac.uk
[b] HFT Stuttgart – University of Applied Sciences, Faculty C, Schellingstrasse 24, 70174 Stuttgart, Germany – volker.coors@hft-stuttgart.de

Commission IV, WG IV/8

KEY WORDS: 3D city models, data modelling, data quality, validation, model healing

ABSTRACT:

Three dimensional digital city models are more and more used in different domains. These models are acquired in different ways and in very different levels of detail, due to sensor limitations or simply because of requirements of the specific scenario. The tendency towards more sophisticated models including semantic information, additional attributes and links to larger datasets can be observed in recent years. The use of semantically rich standards (e.g. CityGML) for the modelling of urban space beyond the simple representation of the geometrical aspect is one of the key hints towards a rich 'urban information space'. This information space is going to be integrated into other applications or will be the basis for sophisticated systems in the urban domain, like simulation tools. Therefore it will be essential to control and enhance the quality of these models in order to be able to meet the needs of the aforementioned urban systems. This paper is going to introduce an approach, for validation of geometry and topology and first steps towards an automated healing. The authors recommend a test suite that provides modules for specific aspects of the model quality, which can be switched on and off so that customized tests for specific models and specific scenarios can be conducted. Reporting functionality of the tool should be divided into two parts: error report and healing report. In that way the test engineer would have information about which errors occurred, which errors were healed and by which factor the quality of the model was increased. First results of a prototype system developed at HFT Stuttgart will be presented in this paper as a starting point for further research into the field of quality management of 3D city models.

1. INTRODUCTION

Fast development in information and communication technologies made it possible to acquire and model a vast amount of 3D urban data and to produce a digital version of the real world. One very popular example of such a digital world is GoogleEarth, which is also well known among non-specialised users. With Google's initiative 'Cities in 3D' (Google, web ref.) the goal of integrating 3D urban data into the digital globe is even more pushed forward. Many cities and municipalities have already produced 3D city models and use them for a variety of tasks: urban planning, marketing, disaster management, security, etc. This multi-use of one model is desirable because the re-acquisition of project-dependent models would include additional cost and work force. In order to optimize the use of resources most cities try to produce a model that can be used in many different scenarios. For multiple use like simulation, urban planning, marketing, etc. the quality properties of the model need to be defined and validated because all application dependent requirements need to be met. Another example for an application dependent scenario is mobile navigation. 3D urban models used for navigation support have different needs in terms of accuracy and visual quality compared to city models for simulation purposes. Therefore an envisioned quality test tool needs to be flexible and adaptable in terms of validating specific parameters and quality specifications.

In this paper the authors are going to present a possible set of methods in order to define and validate the quality of 3D urban models. As a further step a process of improving the quality of the model will be suggested, which consist of a automatic healing process that is applied after the actual quality check. These two steps, quality check and healing, should remain separated as far as possible in order to sustain the transparency of the overall process. For the user it should always be clear how the quality was before healing, which errors have been corrected and which results have been created after healing. This could be achieved by providing an appropriate report engine, which produces separate quality check and healing reports, for example.

This structured two-level process could be the basis for a certification process for 3D urban models. In that way the user can easily identify appropriate models for his specific application, can get general information about the basic quality of the model and how a possible healing process has increased the quality of the original model. All of this information can help users to identify appropriate models for their specific application or scenario, even before testing the data and importing it into their own tools.

The remaining paper will be structured in the following way: section two provides a brief overview of relevant work and state of the art, section three outlines the quality check and healing process for the part of object geometry. Section four describes the possible quality tool architecture and section five concludes the paper and gives a short outlook on future work.

2. RELATED WORK

Generally quality in terms of 3D urban models is highly application-dependent. Quality parameters need to be identified for each scenario and can differ considerably. Nevertheless,

Gubtill&Morrsion (1995) identified seven elements of geo data quality:

- Lineage
- Positional accuracy
- Attribute Accuracy
- Completeness
- Logical Consistency
- Semantic Accuracy
- Temporal Information

In ISO 19113 (ISO, 2002) quality principles for geo data were defined as well as an evaluation process in ISO 19114 (ISO, 2003). Quality of metadata is also defined in ISO 19115 (ISO, 2003). The Open Geospatial Consortium has also a working group that specifically looks at data quality (OGC, web ref.). However, the aforementioned standards mainly cover 2D geo data, specific problems and characteristics of 3D urban models are not taken into account, as well as healing processes or quality improvement.

Kazar et al. (2008) describe methods in order to validate polygonal 3D geometries inside an Oracle DB. This approach does not include healing or checks for semantic or attribute correctness. Krämer et al. (2007) developed approaches for data quality management in terms of 3D urban models. This approach also allows rudimentary healing capabilities, e.g. correction of topological errors, correction of false orientation of normal vectors, etc. However, the developed algorithms only work on triangulated geometry.

3. QUALITY CHECK AND HEALING

3.1 Concept

Quality can be defined in very different ways for 3D urban models, especially in semantically rich data models like CityGML (OGC, 2008). Especially for multi-purpose 3D models it is necessary to define quality parameter sets, which reflect the actual application scenario. These parameters include geometrical correctness, positional accuracy, semantic accuracy, topology, and textures. In this paper the authors would like to focus on geometrical and topological correctness and healing of geometrical errors as a basis for further research into other aspects outlined above. This is a valid approach as geometrical and topological correctness can be a requirement for other tests, like attribute correctness. For example, if the orientation of normal vectors is used in order to check for a correct 'roof'-classification attribute, then the normal vectors need to be validated first. Otherwise a check for attribute correctness is likely to fail.

In CityGML, the shape of a building is modelled as a set of polygons describing the boundary of a solid. However, it might contain a certain number of errors as a result of data capturing and further data processing. In this paper the focus is on finding errors and automated correction of the building geometry. The automated correction of error is called healing.

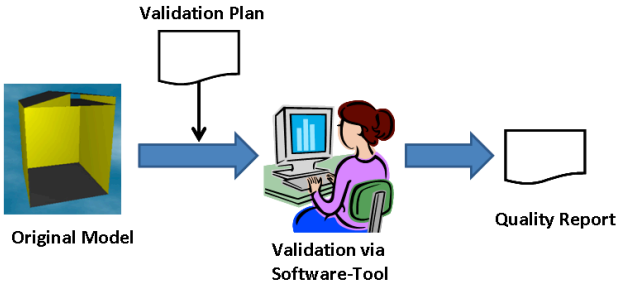

Figure 1: Quality check process with quality report

3.2 Validation and Healing of Polygonal Surfaces

In CityGML a building geometry is usually modelled as a set of Solids, MultiSurface or a set of boundary surfaces. A MultiSurface is a set of Polygons. Boundary surfaces describe the geometry using MultiSurfaces as well. Solids consist of composite surfaces which are again a set of polygons. The fundamental geometry within a building model in CityGML is obviously the Polygon.

Each Polygon is bounded by a planar Linear Ring r. Holes in a Polygon are defined by additional planar Linear Rings. In that case, all Linear Rings have to be in the same plane. However, there might be errors in the data set. Under the assumption that the CityGML file contains invalid polygons we treat the linear rings read from the file (or database) first as a sequence of vertices.

A sequence is an ordered list of elements. Unlike a set, order matters, and the exact same elements can appear multiple times at different positions in the sequence. A finite sequence may be denoted as $(a_0, a_1, \ldots a_n)$, the empty sequence () has no elements.

A finite sequence of points $R = (P_0, P_1, \ldots P_n), n \geq 3$ is a **Linear Ring** if

(i) the first and last point P_0 and P_n represent the same point: $P_0 = P_n$ (**closeness**)

(ii) beside first and last point, two points should not have the same coordinates:
$P_i \neq P_k, i = 0,\ldots,n-1, k = 0,\ldots,n-1, i \neq k$

(iii) two edges (P_i, P_{i+1}) and (P_k, P_{k+1}) $i=0,\ldots n\text{-}1$, $k=0,\ldots n\text{-}1$, $i \neq k$ do only intersect in one start-/ endpoint P_{i+1} if $k = i+1$ or point P_{k+1} if $i = k+1$. No other intersection is allowed (**no self intersection**).

If all points of the Linear Ring lie on a plane defined by three non-colinear points $P_a, P_b, P_c \in R$, we call it **Planar Linear Ring**[1].

A given sequence of three or more points $P = (P_0, P_1, \ldots P_m)$ can be transformed into a Linear Ring $R = (P_0, P_1, \ldots P_n)$ by the following repair operations:

If first and last point are not equal, rule (i) is violated. It can be repaired by simply adding a point P_{n+1} with coordinates (x_0, y_0, z_0) to the sequence. Of course, self intersection has to be tested again after the insertion of the point.

If rule (ii) is violated and two consecutive points P_i and P_{i+1} have the same coordinates, point P_{i+1} is deleted from the

[1] Later in the paper, we need to distinguish between Planar and non-planar Linear Rings to heal holes.

sequence. If non-consecutive points are identical, it is checked for self-intersection first before removing one of the two points.

Self-intersections are not allowed in R (rule (iii)). If P contains self-intersections, there are two possible repair operations as shown in Figure 2. The first approach the repair function is looking for a permutation of the points in P with no self intersection. Points that appear twice in the sequence will be removed in this approach. Unfortunately, the complexity of the algorithm is O(n!) and can only be computed for a small number of points in P.

An alternative approach is splitting the point sequence into two Linear Rings, if two edges (P_i, P_{i+1}) and (P_k, P_{k+1}) intersect in a point P_s. If P_s is not part of the sequence so far, it has to be added. The intersection point will be part of both Linear Rings. If the two edges are identical, one of the edges will be removed from the sequence without splitting it into two Linear Rings.
For 3D urban data models the first approach is chosen if the number of points is small. It is more likely that the order of the points was mixed up during the modelling process.

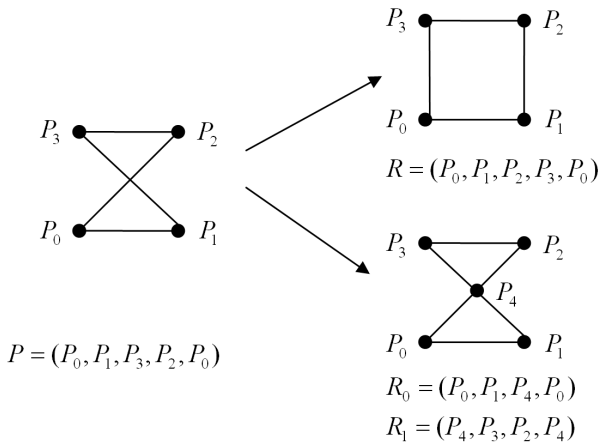

Figure 2: two possible repair-operations of a point sequence where two edges intersect each other. First approach by finding another permutation of points that has no intersections. Second by splitting the sequence into two linear rings

With this set of operations a given sequence of points with three or more different elements can be transformed into a linear ring. A sequence with less than three different points cannot be transformed into a Linear Ring. These sequences are deleted from the model as degenerated polygons.

The now valid Linear Ring defines a boundary of a Polygon, if it is planar. All points of the Linear Ring have to be on the same plane. For validation, a given tolerance value is used to deal with rounding effects of floating point operations and minimal errors during data capturing. If errors occur during the validation, no healing is applied at this stage. It would require the manipulation of point coordinates and several polygons that use the same point are affected by this type of healing.

3.2.1 Valid Polyhedron (Simple Solid)

Let $F = \{F_0, F_1, \ldots F_{n-1}\}$ be a given set of faces with valid polygons as geometry. Each Polygon F_i is defined by a linear Ring R_i. In addition, F_i has a normal vector n_i defined by the order of the points of R_i. For simplification none of the polygons has a hole. If F is a Multisurface geometry in CityGML, no further constraints are given. However, if F is a Solid geometry, it has to be the boundary of a solid. If F is not a valid boundary of a solid, the healing operation should modify F to become such a boundary. We limit the following discussion to simple solids homeomorph to a 3D sphere.

The boundary of a simple solid is an orientable surface. Every polygon is oriented such that its normal vector points to the outside. The boundary surface of a solid homeomorph to a 3D-sphere is a 2D-manifold. Each edge is shared by exactly two polygons.

To validate the set of polygons, firstly connected components are detected. It is assumed that the geometry of a building or building part consists of one component only. If this is not the case, new building parts can be created, one per component. Secondly, it is checked that no polygon intersects another polygon. Duplicate polygons will be removed. Thirdly, the orientation of the polygons are checked and healed. Last but not least, inner polygons have to be removed as they could not be part of a boundary of a solid geometry.

Two polygons are connected if they share a common edge. A connected component *F'* of *F* is a subset of *F* containing all polygons that are connected. A path is a sequence of Polygons $(F_1, F_2, ..., F_n)$ where F_i and F_{i+1} are connected. *F'* is connected if for every Polygon F_i of *F'* a path to any other Polygon F_k of *F'* exists.

Connected components can be simply detected by breath-depth search in an undirected graph data structure where the Polygons are nodes of the graph. Two nodes are connected if the corresponding Polygons share a common Polygon.

For each connected component *F'* the number of Polygons shared by an edge is counted per edge. It has to be two for each edge if *F'* is the boundary of a simple. A Linear Ring of edges that share only one Polygon is the boundary of a hole. A hole can be healed by inserting a set of new Polygons to close it. If the Linear Ring is planar, one Polygon is sufficient. However, it could be that the Linear Ring is not planar, so more than one Polygon is necessary to close the hole.

A Linear Ring of edges that share three or more Polygons indicate inner Polygons. In the current status of the implementation, these inner Polygons are not removed automatically but the user has to confirm it.

If each edge bounds exactly two Polygons, the orientation of the Polygons is validated. Each edge $e = (P_i, P_j)$ has two half-edges $he_1 = (P_i, P_j)$ and $he_2 = (P_j, P_i)$. If all Polygons in *F'* have the same orientation, each half-edge is used only once. If one half-edge is used twice, one of the Polygons that share this edge is incorrectly orientated. The Polygon orientation can be healed by changing the order of the points of the Linear Ring that bounds the Polygon. Of course, changing the orientation of a Polygon might have an impact on other Polygons. The changes are propagated through all Polygons until a stable solution is found (Edelsbrunner, 2001). After this, all Polygons have the same orientation, either all clockwise or all counter-clockwise. A simple ray from any point cast will detected the inside and outside of the solid. This is used to find the counter-clockwise orientation, where the Polygon normal points to the outside of the solid.

3.2.2 The validation and healing process

The healing process (Figure 3), aforementioned described with focus on geometry, is an extension of the validation process (Figure 1). In this process the validation can be regarded as the detection of errors in the model. When detected, the tool tries to correct the error automatically and provides a correction recommendation. Depending on how interactive the process is configured the user is able to accept or reject this recommendation. Nevertheless, the correction of the model can also be done automatically without user interaction.

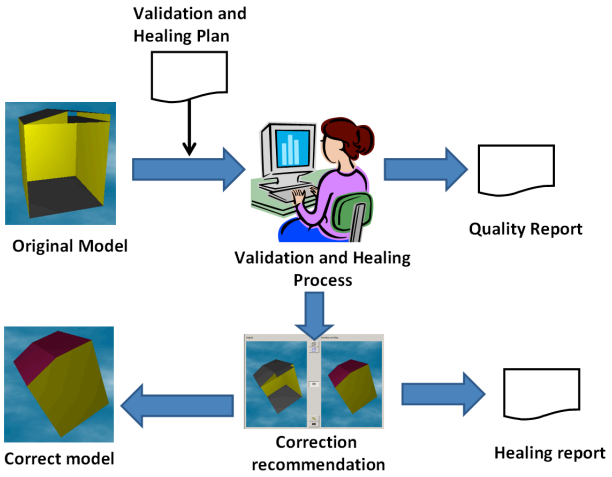

Figure 3: The quality check and healing process with 2 separate reports

The reports in this process are subdivided into two parts: a quality report which includes the errors in the original model and the healing report including the un-/corrected errors. In this way the user of the model can determine the level of quality of the original model and how many errors were found, and he also knows how many of these errors could be corrected. By providing e.g. the IDs of objects which could not be corrected automatically, the user has the information where he needs to correct the model manually. The information in the healing report can also include information about the factor by which the quality was increased or a summary of the most common errors. This information could be used as feedback for the initial modelling process and might influence and improve its quality management methods.

4. TOOL ARCHITECTURE AND 3D MANAGEMENT FRAMEWORK

The outlined approach of validation and healing of 3D urban models is implemented in a standalone Java-Software called QSCity3D at University of Applied Sciences Stuttgart as a software project by a group of computer science students under supervision of the authors. Based on this experience, the architecture of an envisioned quality test tool should be flexible and adaptable to specific quality test cases and scenarios. As already mentioned throughout the previous sections, the quality of a 3D urban model can be defined very differently for specific scenarios in which the models are used. The test tool should be able to test certain criteria and leave out other criteria when these are not relevant. Internally tests should be able to be turned on and off, preserving a general workflow that is valid for all tests. In this way it would be possible to hide the complexity of the workflow and different test cases from the user and the tool could be configured by a simple interface/config file in order to prepare an appropriate test setting. This approach of a more or less closed test system that can be configured through a simple interface can be used to implement a stand-alone GUI application, as well as producing a test library. This library can be used as a black box test module, which can be integrated into existing applications, like 3D management frameworks.

The integration of the test library can be done in specific ways and on certain levels of these frameworks. Basically the concept would be to 1) configure the test tool in the appropriate way to switch on/off certain tests, 2) provide the input model (format transformations might be necessary), 3) read the healed model and 4) analyse the quality- and healing protocol. In (Figure 4) one possible scenario for integrating test capabilities into a data management framework is depicted.

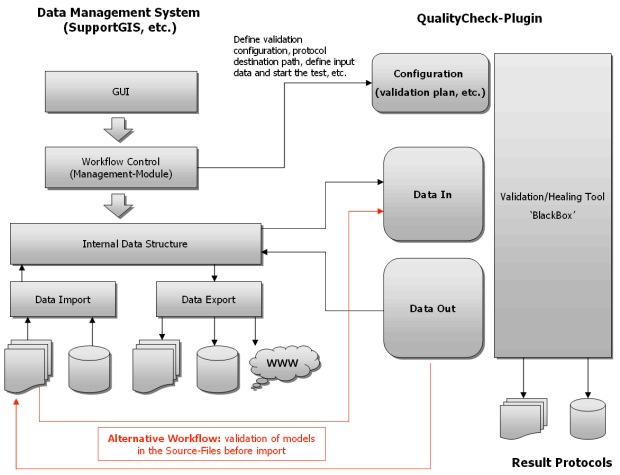

Figure 4: Possible integration of the Quality/Healing-'blackbox' into 3D model management systems

The development of a closed system type of library which can be used for a stand-alone version of the tool as well as to integrate test capabilities into other application appears to be a sensible approach. Many existing applications with focus on 3D data management or manipulation would benefit from quality test capabilities. Integrated quality test functionality would also optimize workflows based on existing applications and increase acceptance among users. Nevertheless, the development of a stand-alone test suite based on CityGML files is supported as well.

5. CONCLUSIONS

As 3D urban models are more and more used in different ways and integrated into application scenarios, the quality of these models needs to be validated and improved. The quality is essential in scenarios where application results are directly related to the correctness of the 3D model, e.g. simulations. In this paper the authors presented a concept for data quality validation, error correction and sophisticated reporting functionality. The envisioned approach includes a flexible quality test suite, which can be adapted to specific scenarios and quality requirements. This is very important as quality of 3D urban models can be a mix of very different parameters that are different for individual scenarios. A possible approach for this would be to define certain levels of quality, defining specific

test sets for a majority of use cases of models. Nevertheless, a future quality test suite should be able to switch on/off certain quality criteria and tests in order to produce realistic results for specific test cases. First results of a test application were briefly outlined in this paper as a basis for further research into this field. Especially automatic error detection and healing in geometry is still a field for further research. However, as semantically rich data models like CityGML are more and more in use, also test and validation processes for semantics, attributes, etc. need to be developed in the future.

The integration of a future quality test suite into existing data management or authoring applications seems to be a sensible approach as outlined in section four. This would increase usability and efficiency of workflows as well as the acceptance among users. Nevertheless a possible test library can also be used to develop a stand alone test application, which can be used independently, for example by model end users.
The reporting functionality can also be used to award certain quality certificates in order to proof the level of quality to potential users of the model and to provide information about certain fields of use that the model is appropriate for.

6. ACKNOWLEDGEMENTS

We would like to thank the Marie-Curie Research Training Network and the CityNet-Project for their support. Part of the work is funded by the German Ministry for Education and Research within the Project CityDoctor. We also would like to thank the City of Stuttgart for providing test data during the development of the QS-City3D prototype.

7. REFERENCES

Edelsbrunner, H., 2001. Geometry and Topology for Mesh Generation. Cambridge University Press.

Google Inc., Cities in 3D Program, http://sketchup.google.com/intl/en/3dwh/citiesin3d/ (accessed 15.05.2010).

Guptill, S.C., Morrison, J., 1995. Elements of Spatial Data Quality. Pergamon.

International Organization for Standardization (ISO), 2002. ISO 19113 Standard: Geographic information -- Quality principles.

International Organization for Standardization (ISO), 2003. ISO 19114 Standard: Geographic information -- Quality evaluation procedures.

International Organization for Standardization (ISO), 2003. ISO 19115 Standard: Geographic information -- Metadata.

Kazar, B. M., Kothuri, R., van Oosterom, P., Ravada, S., 2008. On Valid and Invalid Three-Dimensional Geometries. In: Fendel, Oosterom, Penninga, Zlatanova (Eds.), Advances in 3D Geo Information Systems, Springer Press.

Krämer, M., Haist, J., Reitz, T., 2007. Methods for Spatial Data Quality of 3D City Models. Eurographics Italien Chapter Conference

Open Geospatial Consortium (OGC). Data Quality Working Group. http://www.opengeospatial.org/projects/groups/dqwg (accessed 14.05.2010)

Open Geospatial Consortium (2008): City Geography Markup Language (CityGML).