Eurographics Workshop on Urban Data Modelling and Visualization (2014)
V. Tourre and G. Besuievsky (Editors)

Managing temporal change of cities with CityGML

M. Morel and G. Gesquière

LIRIS, UMR CNRS 5205
Université de Lyon, France

Abstract

An increasing number of cities are developing digital models. It becomes thus necessary to take into account changes over time. Interoperability and thus the use of standards is also recommended. In this paper, we propose a new method, based on CityGML to take into account changes in the objects which compose the city. This method is efficient for any kind of changes of the city objects (semantic, geometry, topology or appearance). We then propose an extension of our method in order to consider more frequent changes as it is the case with sensors data that can be linked with part of city objects.

Categories and Subject Descriptors (according to ACM CCS): I.3.5 [Computer Graphics]: Computational Geometry and Object Modeling—I.2.10 [Computer Graphics]: Vision and Scene Understanding—Representation, data structure, and transforms

1. Introduction

In the past ten years managing 3D GIS data for virtual city has become an important topic. These data may be used in numerous applications like urban planning or for helping in the decision making processes. Other applications are proposed in a serious game context where it is often necessary to use geographic information system (GIS) data in order to prepare scenes. Even if sometimes the 3D data are reshaped by designers, using real data can be a significant gain of time in the production of a new game level based on a known city representation. This aspect becomes important if numerous places must be modeled. In serious games like in fire tactic training, users may be faced with completely virtual places created for instance with procedural generation tools like in [YWVW13], [LSWW11] but also places where they will intervene in their real missions. In this last case, it is necessary to establish a scene that represents an exact digital copy of the city. Unfortunately, data aggregation is not sufficient to obtain such a real scene with data extracted from geographic information systems. For instance, in the SIMFOR project, it was necessary to create several states of the same objects [CRDG11]. For a building, this implies to model the required states of the building (e.g. "destroyed", "burned" or "partially destroyed"...). These model states are then available to create the scenario in a dedicated product.

In urban planning, it is also important to follow the changes of the city during time. For instance a new building can replace another one which has been destroyed. Temporal information becomes important to manage the city objects lifecycle. At last, it seems important to give a special attention to interoperability problems. Indeed, the data often comes from several sources (GIS, CAD, modeling tools like Maya or SketchUp). The data must then be made available to other tools like Unity or UDK for games. In the case of urban planning, it is important to be interoperable with products from Autodesk, Bentley or ESRI. This need of interoperability naturally leads to propose a solution based on the work of the Open Geospatial Consortium. Indeed, CityGML is an interesting standard for exchanging objects of virtual cities [Kol09]. Unfortunately, in its current version 2.0, CityGML specification does not take into account the temporal aspect [GKNH12]. Moreover, multiple representations of same objects are only dedicated to the management of different level of details.

Our main contributions in this paper are as follows; first we propose a new solution based on the CityGML specification to take into account the temporal aspects and the state modifications of city objects. This method will be efficient for any kind of changes of the object (semantic, geometry, topology or appearance). Changes will be recorded manually by the user and linked to the objects. We then propose an extension of our method in order to take into account more

DOI: 10.2312/udmv.20141076

frequent changes. Finally, it will be possible to export a static version of the city at any time to use it with any software on the market and guaranteeing the interoperability of our approach.

This paper is structured as follows: Section 2 presents a brief reminder on CityGML specification interesting for this paper. Then, after some definitions, our method is presented (section 3 and 4). Section 5 illustrates this method on an example based on buildings. Finally, we will present future extension of our work.

2. Some CityGML concepts

Three-dimensional city models can be used for instance by public administration for different purposes like environmental simulations or facility management or emergency response [CKPSM06, KGP08]. Thereby, the field of application evolved from traditional applications such as network planning where pure geometric models with low level-of-detail are required, to advanced applications in areas such as tourism. That is, the requirements of city models heavily increased, meaning that besides geometric information there is also a strong need for semantic information and data update [GZ12]. User communities can take advantage of using standards to develop application schemes that follow the rules and reuse the components defined in the abstract standards. An XML scheme encoding following the GML grammar can then be derived from the application scheme and serves as the basis for data exchange. These approaches were for example taken during the development of CityGML 1.0, published by the Open Geospatial Consortium (OGC) in 2008 and upgraded in 2012 [GKNH12]. CityGML specifies a standardized application schema for 3D city models from which a GML 3.1.1 encoding is derived. Semantic, topology, geometry, appearance of city objects are described in the CityGML standard. CityGML is therefore both a conceptual model and an encoding, enabling syntactic and semantic interoperability. More related to this paper, some of its interesting key features are [Kol09]:

- Thematic modeling: the model covers a wide range of city objects, including but not limited to buildings, transportation facilities, water bodies, vegetation, terrain, land use, city furniture, etc.
- Multi-scale modeling: CityGML supports five levels of details (LOD). This mechanism facilitates the integration of 2D (at LOD0) and 3D datasets at different scales representing the same real-world entities. The same feature can be represented with different geometries at each scale. CityGML also provides an aggregation and decomposition association between objects that can be used to indicate that an object at a lower LOD has been decomposed into two or more objects at a higher LOD. These LODs also enable applications or simulation models to process the data at the most suitable scale.
- CityGML has been designed as an application independent information model and exchange format for 3D city and landscape models. However, specific applications typically have additional information to be modeled and exchanged. In this case it is possible to incorporate application specific information into the CityGML instance documents. This approach is especially feasible, if the application specific information follows essentially the same structure as defined by the CityGML standard. This is the case, if the application data could be represented by additional attributes of CityGML objects and only some new feature types would have to be defined.
- Generic attributes and objects may appear arbitrarily in the CityGML instance documents, but there is no formal specification of the names, datatypes, and multiplicities. This may reduce semantic interoperability.
- If application specific information are well-structured, it is desirable to represent them in a systematic way, i. e. by the definition of an extra formal schema based on the CityGML schema definitions. Such an XML scheme is called a CityGML Application Domain Extension (ADE). It permits to validate instance documents both against the extended CityGML and the ADE schema and therefore helps to maintain semantic and syntactic interoperability between different systems working in the same application field.
- The temporal part in CityGML is treated only briefly. Objects in the cities can only have a date of construction (creationDate / yearOfConstruction) and destruction (terminationDate / yearOfDemolition). Lifecycle is not taken into account. The future version of CityGML will need to take into account lifecycle. In this paper, we propose an efficient way to reach this goal. Nevertheless, it is necessary to remain compatible with the current version of the CityGML standard, to allow data export.

3. Related work

Using CityGML and giving an access to harmonized data is only a first step towards providing adequate support to city objects management, which requires development of analysis and spatio-temporal data models functionalities. Regarding time management which is the topic of the paper, it has been studied for more more than twenty years. Detailed state of the art can be found in [PTKT04], [WH94]. Few studies describe building lifecycles to stress different states and transformations of the 3D city objects [SLVM08], [DB08]. Objects that compose the city like for instance buildings, bridges, vegetation, terrain change during time. For instance, for a building, different kinds of transformations can be identified. De Luca et al propose to define the following states for a building: creation, destruction, reconstruction, division (the building is separated in several parts), union, variation (modifications) [DLBS*10]. Other transformations are linked to the semantic part of the building, for instance if the owner of a house change [SLVM08]. Describing the build-

ing lifecycle implies to take into account states and spatial changes along a temporal arc [SLVM11]. But this scheme must also be extended to every objects of the city (terrain, vegetation...) and must take into account semantic, topological, appearance and geometric changes. Changes involving buildings can be sudden or gradual. For instance, a change of property is a sudden event. On the other hand changes may be progressive: for example, building demolition is a short event but the construction of a Gothic cathedral is a long event that lasts several centuries. Furthermore, historical building deteriorations may take centuries or millennia [SLVM09]. But this change can also be a data stream linked to sensors (temperature, pollution, ...). It is necessary to enhance previous methods to take into account attribute that can change frequently during time.

Keeping the possibility of exchanging data with other tools and making a snapshot for a given date of the city is essential. Several papers are proposing methods based on CityGML. For instance in [PCD*13], CityGML scheme is modified to add temporal information on buildings. This method does not seem to be dynamic enough; only definite states can be registered. CityGML scheme modification and possible standardized exports is not discussed. The European project BRISEIDE proposes to extend current data / metadata models to be able to benefit from information available at high refresh rate (e.g. from sensors) as well as to be able to analyze evolution of phenomena over time [DACP10]. This approach, based on OGC standards like Sensor Observation Service proposes to connect and to follow changes of these sensors on a 3D context.

In the remainder of this article, we present a method based on CityGML, allowing the capture at a given date and export to other softwares while respecting this standard. This method allows to set markers in time as well as linked states. Markers can also be static or evolve dynamically. Our method also allows to cover all objects of the city defined by CityGML standard.

4. Overview and definitions

To describe temporal events, we propose to add two concepts in the CityGML model, *tags* and *flags* which are defined in this section.

Definition 1: A tag is a temporal information linked to a city object. A tag can be a date defined in a standardized way.

A tag always refers to an information (geometry, semantic) until the object is destroyed. In the figure 1 an object $Object_i$ has four tags linked with four standardized dates. If there are no tags, the default informations in CityGML are used. If there is no date information, the building is visible at any time (and can be seen as timeless).

Definition 2: A (static) flag is used to describe the behavior of an object. It is defined for a given tag.

A flag may be composed of semantic and / or geometry informations that have to change for a given tag.

In the figure 1, The flag $Flag1$ is linked to $Tag1$. It corresponds to the construction of the object $Object_i$. For the date $Date2$ defined for tag $Tag2$ the object is modified.This codelist can be easily adjusted according to uses cases investigated, like events such as fire, earthquake, explosion, rubble, parcel empty (and ready for a new building) and the percentage of destruction. For instance, a flag can describe a modification of the building, caused by a partial destruction by fire, leaving the building with 30% of destruction. Related to this, a new set of geometry (and texture) can be attached. In this case, it can be a damaged version of the building with a new set of burned textures.

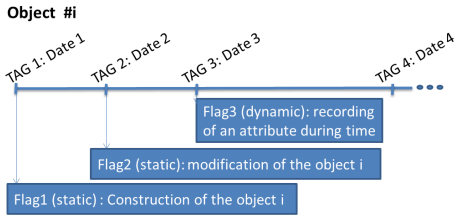

Figure 1: *Lifecycle of a building is described with several tags.*

Definition 3: A dynamic flag contains a data source associated with a given CityGML attribute. This source is a set of possible values, with a date for each ones.

In the figure 1 the flag $Flag3$ is circumscribed between the tags $Tag3$ and $Tag4$. A set of values are registered during this interval. For example, in the case of an application domain extension (ADE), a given attribute like the color of a building wall may be monitored over time. This color may change in accordance with information related to a temperature sensor.

5. Adding temporality in CityGML

The addition of tags and flags allows the management of the lifecycle of the city objects. As it can be seen in figure 2, *_CityObject* is the base class of many types in CityGML. Using tags and flags at *_CityObject* level allows to add temporality to all these sub-types. This modification on the UML scheme is presented in figure 3. City objects can have several tags and flags. A flag or tag is owned by one city object. A flag can be constructed with dynamic values (DynFlag) or only one (StaticFlag). Nevertheless, it is necessary to preserve interoperability and to return to a CityGML file readable by other applications. We therefore choose to make the export available, once a date is set.

We propose to use the concept of generic attribute available in CityGML and to add suffixes on the IDs of the objects. Multiple instances of the same object remain possible

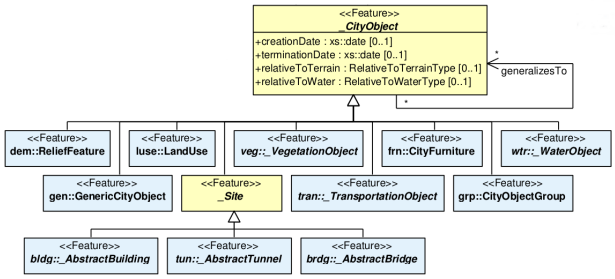

Figure 2: *CityGML's top level class hierarchy. (extracted from [GKNH12]). The subclasses of _CityObject comprise the different thematic fields of a city model covered by separate CityGML extension modules: the terrain, buildings, bridges, tunnels, the coverage by land use objects, water bodies, vegetation, ...*

with this added information. For example if a flag *FLAG*0 is added to a named object *Building*1, the name of the new instance will then become *Building*1_*FLAG*0. For a tag *TAG*0 the new instance will become *Building*1_*TAG*0. Thus, it is possible that an object has from zero to n flags and from zero to n tags. In the particular case where there is no tag and flag, only the default temporal informations contained for objects in CityGML, if they are filled, will operate; (e.g. creationdate and terminationdate for the *_cityObjects*). If no temporal data is available, the object is timeless; It will then appear in the scene for any selected date.

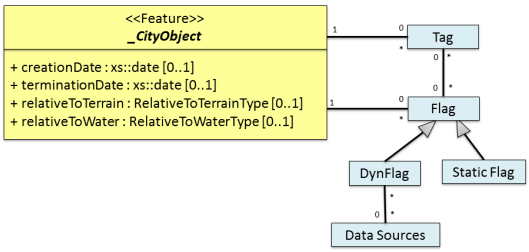

Figure 3: *UML scheme modifications to take into account tags and flags.*

The tag will be joined with a date defined with a generic attribute. Figure 4 shows how the generic attributes are used (in brown). The generic attribute will permit to qualify the flag (name = 'usage', value = 'house'), to set a date to a tag (name = 'date', value = 2012-01-25) and to bind a tag to a flag (name = "flag", value = "building1_FLAG0"). A second tag for the same building may be *Building*1_*TAG*1. The IDs of the objects are chosen unique by construction. These IDs are particularly important to enable the partial (for a given date) or total data export in order to generate again the temporal model after an import in our software. Even if

Figure 4: *Tags and flags are added with modifications of IDs with suffixes and generic attributes strategy.*

the IDs constructions allow easy retrieval of corresponding dates to an object as a building for example, in our inner data structure, an index mechanism (as hash code) can be proposed to enhance the query capacities on temporal and spatial data.

When more frequent property changes are necessary, it is possible to use Dynamics Flags that we propose as an extension of our method. If we propose for example the case of temperature changes at the walls of a building (as in an Application Domain Extension linked to CityGML), we can choose to change the appearance (texture) during time. All necessary states will be stored in a Dynamic Flag. This may simply be a file containing dates / hours and the corresponding texture. This approach may be generalized as a web Service proposed by OGC. A Dynamic Flag is used like a static one. A tag can provide a starting date (the event related to this dynamic flag will be available from this tag). In our current implementation, this dynamic flag must be finished before starting another tag with linked flag.

6. Results

Vcity is a project which contain a software for reading, writing and viewing interoperable format such as CityGML. This platform was originally produced as part of the SIMFOR project where data should be aggregated to create a mock-up for a serious game [CRDG11]. The temporal part was until now untreated.

There are three main components in the Vcity interface (see figure 5). First, there is the 3D view which represents the major part of the interface. It is possible to see imported data and to select data by click. On the left side, there is a panel containing a tree which shows the hierarchical representation of imported data. Each node of this tree can be expanded or selected. A node will be selected in the tree, in

the 3D view and details about the node can be seen in the text area below. In the bottom of the window, a temporal slider is available to move during time and select specific dates.

This application is developed in C++ using Qt5, OpenSceneGraph, GDAL and a upgraded version of libcitygml. CityGML data are provided by french mapping agency IGN. Data can be imported in our software. It is also possible to

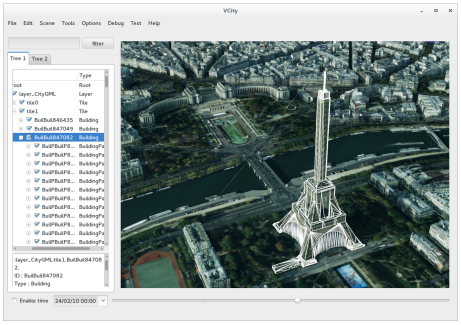

Figure 5: *Vcity interface. The 3D view is on the middle right. On the left data is organized as a tree. Below a text area is showing details about selected node in the 3D scene or in the tree. A temporal slider is available on the bottom.*

add, modify or delete city objects. To demonstrate the easy way to add temporal informations with our method, we propose an example based on a building. First, it is possible to create a new scene or to begin with an already constructed one. A building can be selected to work either by using the 3D view or in the tree view on the left or by name using the filter tool above the tree view panel. Once a building is chosen, interaction must be done by right-clicking on it in the tree view. In this use case, an administrative building will be used. Figure 6 shows an overview of the process followed in this use case. The first step consists in adding flags; the user is invited to fill complementary data. First, a name for the flag can be proposed by the user. This name is checked to avoid duplicate IDs. If no value is proposed, a generic name will be generated. Afterward, attributes and their values can be added. In the case of dynamic flag, several attributes can be added too, but one data source is attached to each of them. This data source can be an array of value, an url or a file... In a data source, each record is a pair (date, value). For instance, for a TexturedSurface mapped on a wall, a color can be paired with a value which can represent the temperature in thermic applications. This scheme can also be used in ADE proposed for numerous use cases.

In the example, there are two flags: the first one sets the building usage to "private" and is used by *TAG*0. The second one adds a fire event used by *TAG*1. The last tag *TAG*2 does not contain a flag; the building does not exist after this date, even if the date linked to this tag is before the termination date (stored in the original building). After adding flags, tags must be positioned. Tags can have a name and is linked

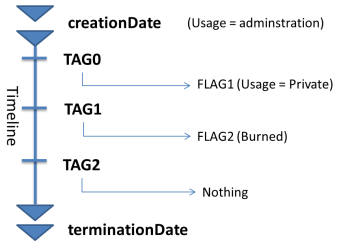

Figure 6: *Timeline of the use case. There is four steps. First, the creation of the "administrative" building. Then it is modified at TAG*0 *and its usage goes to "private". After, at TAG*1 *there is a fire and the building is left for some times. Finally, at TAG*2 *the parcel is freed.*

to a flag. Once at least one tag has been added to a building, it is considered as a temporal building. It means that a request with a date on an attribute can be made on this building. Depending on the date, the value given will be chosen from the original building, a flag or a dynamic flag. If no flag is attached to a tag, it will be considered as the ending of the building and its location will be freed; there is no information linked to this building. This tag may correspond to the terminationDate in CityGML. After these steps, it may be necessary to modify other objects like for instance terrain. After a building destruction, the terrain may not fit this destruction. In that case, the terrain should be flagged too with a version of its geometry which shows the terrain destruction or deformation.

Here is an example of temporal evolution of a building in figure 7

Figure 7: *Screenshot for TAG*0 *(left) and TAG*1 *(right) in the timeline (figure 6).*

7. Conclusion and future works

In the context where most cities have a digital models, it is necessary to take into account changes that appear during time. Temporal aspect is at the center of many works for years. Interoperability and thus the use of standards is also recommended. In this paper, we propose a new method,

based on CityGML to take into account changes in the objects which compose the city. Our main contributions in this paper are as follows; first we propose a new solution based on the CityGML specification to take into account the temporal aspects and the state modifications of city objects. This method is efficient for any kind of changes of the object (semantic, geometry, topology or appearance). Then we propose a new kind of flag named Dynamic Flag to take into account frequent modification of an attribute during time as it is the case with sensor that can be linked with parts of city objects. This method has been constructed with existing elements already defined in CityGML (generic attributes and suffixes added to the IDs of objects). Tags and Flags must be defined manually in the interface. The modification of the geometry is made in other commercial softwares where CityGML format is available. We are currently working on the possibility to export a static version of the city at any time in order to use it in any software on the market and guaranteeing the interoperability of our approach. This method relies on CityGML 2 standard. It allows prefiguring the changes that will be needed in order to better take into account the temporal aspect in CityGML 3 in the coming months. It also provides a reference within an implementation in the vcity project.

In the future works, we would like to make an automatic process in order to propose values for tags and flags. Additional tools must be developped to detect changes between two versions of a same area provided by two different acquisitions (ID objects may be differents and cannot be taken into account). An other goal is to study the possibility to add uncertainty in the temporal management, which is important for instance in archeology. We also want to look at the visualization of these spatial and temporal data aspect in order to provide real assistance in decision-making processes.

8. Acknowlegment

This work was performed within the BQI program of the Université Lyon 1. CityGML data are provided by IGN (Bati3D).

References

[CKPSM06] CZERWINSKI A., KOLBE T. H., PLÜMER L., STÖCKER-MEIER E.: Spatial data infrastructure techniques for flexible noise mapping strategies. In *Shaker Verlag* (2006), Klaus Tochtermann A. S. E., (Ed.), Shaker Verlag, pp. 99–106. ISBN 978-svn3-8322-5321-9.

[CRDG11] CHAMBELLAND J., RAFFIN R., DESBENOIT B., GESQUIÈRE G.: Simfor: towards a collaborative software platform for urban crisis management. In *IADIS Computer Graphics, Visualization, Computer Vision and Image Processing (CGVCVIP'2011), Rome, Italy* (02 august 2011).

[DACP10] DE AMICIS R., CONTI G., PRANDI F.: An integrated framework for spatio-temporal data management: the project briseide - bridging services information and data for europe. In *In proceedings of WebMGS 2010 - 1st International Workshop on Pervasive Web Mapping* (Como, Italy, 2010).

[DB08] DUDEK I., BLAISE J.-Y.: Visual assessment of heritage architecture life cycles. In *Journal Of Universal Computer Science* (Graz, Autriche, Sept. 2008), J.UCS, (Ed.), Know-Center, pp. 349–357. ISSN 0948-695x.

[DLBS*10] DE LUCA L., BUSARAYAT C., STEFANI C., RENAUDIN N., FLORENZANO M., VÉRON P.: An iconography-based modeling approach for the spatio-temporal analysis of architectural heritage. In *Shape Modeling International Conference (SMI), 2010* (June 2010), pp. 78–89.

[GKNH12] GRÖGER G., KOLBE T. H., NAGEL C., HÄFELE K.-H.: *OGC City Geography Markup Language(CityGML) Encoding Standard*. Tech. rep., Open Geospatial Consortium, Apr. 2012.

[GZ12] GOETZ M., ZIPF A.: Towards defining a framework for the automatic derivation of 3d citygml models from volunteered geographic information. *IJ3DIM 1*, 2 (2012), 1–16.

[KGP08] KOLBE T. H., GRÖGER G., PLÜMER L.: Citygml - 3d city models and their potential for emergency response. In *Geospatial Information Technology for Emergency Response*, Zlatanova S., Li J., (Eds.), no. Bishr 1998 in International Society for Photogrammetry and Remote Sensing (ISPRS) Book Series. Taylor & Francis, London, 2008, pp. 257–274.

[Kol09] KOLBE T. H.: Representing and exchanging 3d city models with citygml. In *Proceedings of the 3rd International Workshop on 3D Geo-Information, Lecture Notes in Geoinformation & Cartography* (Seoul, Korea, 2009), Lee J., Zlatanova S., (Eds.), Springer Verlag, p. 20.

[LSWW11] LIPP M., SCHERZER D., WONKA P., WIMMER M.: Interactive modeling of city layouts using layers of procedural content. *Computer Graphics Forum (Proceedings EG 2011) 30*, 2 (Apr. 2011), 345–354.

[PCD*13] PFEIFFER M., CARRÉ C., DELFOSSE V., HALLOT P., BILLEN R.: Virtual leodium: From an historical 3d city scale model to an archaeological information system. *ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information Sciences II-5/W1* (2013), 241–246.

[PTKT04] PELEKIS N., THEODOULIDIS B., KOPANAKIS I., THEODORIDIS Y.: Literature review of spatio-temporal database models. *Knowl. Eng. Rev. 19*, 3 (Sept. 2004), 235–274.

[SLVM08] STEFANI C., LUCA L. D., VÉRON P., M. F.: Reasoning about space-time changes: an approach for modeling the temporal dimention in architectural heritage. In *Proceedings of the IADIS International Conference on Computer Graphics and Visualization 2008* (Amsterdam, Netherlands, july 2008).

[SLVM09] STEFANI C., LUCA L. D., VÉRON P., M. F.: Time indeterminacy and spatio-temporal building transformations: an approach for architectural heritage understanding. *International Journal on Interactive Design and Manufacturing* (2009). ISSN 1955-2513 (Print) 1955-2505 (Online) Springer.

[SLVM11] STEFANI C., LUCA L. D., VÉRON P., M. F.: A tool for the 3d spatio-temporal structuring of historic building reconstructions. In *Digital Media and its applications in Cultural Heritage* (2011).

[WH94] WACHOWICZ M., HEALEY R. G.: Towards temporality in gis. *Innovations in GIS 1* (1994), 105–115.

[YWVW13] YANG Y.-L., WANG J., VOUGA E., WONKA P.: Urban pattern: Layout design by hierarchical domain splitting. *ACM Transactions on Graphics (Proceedings of SIGGRAPH Asia 2013) 32* (2013).