A FRAMEWORK FOR THE REPRESENTATION OF TWO VERSIONS OF A 3D CITY MODEL IN 4D SPACE

S. Vitalis[1], K. Arroyo Ohori[1], J. Stoter[1]

[1] 3D geoinformation group, Delft University of Technology, Delft, the Netherlands - (s.vitalis, g.a.k.arroyoohori, j.e.stoter)@tudelft.nl

KEY WORDS: 3D, 4D, City models, Semantics, Multi-LoD, Versioning

ABSTRACT:

3D city models are being increasingly adopted by organisations in order to serve application needs related to urban areas. In order to fulfil the different requirements of various applications, the concept of Level of Detail (LoD) has been incorporated in 3D city models specifications, such as CityGML. Therefore, datasets of different LoDs are being created for the same areas by several organisations for their own use cases. Meanwhile, as time progresses newer versions of existing 3D city models are being created by vendors. Nevertheless, the existing mechanisms for representating multi-LoD data has not been adopted by the users and there has been little effort on the implementation of a mechanism to store multiple revisions of a city model. This results in redundancy of information and the existence of multiple datasets inconsistent with each other. Alternatively, a representation of time or scale as additional dimensions to the three spatial ones has been proposed as a better way to store multiple versions of datasets while retaining information related to the corresponding features between datasets. In this paper, we propose a conceptual framework with initial considerations for the implementation of a 4D representation of two states of a 3D city model. This framework defines both the data structure of such an approach, as well as the methodology according to which two existing 3D city models can be compared, associated and stored with their correspondences in 4D. The methodology is defined as six individual steps that have to be undertaken, each with its own individual requirements and goals that have to be challenged. We, also, provide some examples and considerations for the way those steps can be implemented.

1. INTRODUCTION

3D city models are becoming increasingly popular among public and private organisations, such as municipalities and government agencies. They can be used for applications where 3D representations of the urban environment are required, such as the optimisation of community energy planning (Zhivov et al., 2017) and simulation of evacuation scenarios (Choi and Lee, 2009). The amount of detail that has to be incorporated in 3D city models, though, greatly varies according to the specific application that they are intended for. In fact, higher detailed models than required not only fail to produce better results, but they also add unnecessary processing complexity and are more prone to introduce validity and accuracy issues (Biljecki et al., 2014a). Meanwhile, the extraction of lower levels of details (LoD) of a city model from one more detailed dataset is non-trivial and extremely complicated.

For this reason, the concept of LoD has been incorporated in 3D city models specifications (Open Geospatial Consortium, 2012) as a mechanism to store multiple 3D representation of the same real-world objects in one dataset. This has been implemented with the use of identifiers that link together the different geometrical representations of the same city objects. Unfortunately, this approach has been proven difficult when creating and maintaining 3D city models (Steinhage et al., 2010), leading to the inexistence of multi-LoD datasets (Biljecki et al., 2014b). Therefore, there is no mechanism to enforce a reliable and useful way of storing multiple 3D representations of the same objects in the same dataset, while enforcing consistency.

In order to solve this problem, the representation of scale or time as an additional dimension to the three spatial ones has been proposed (van Oosterom and Stoter, 2010). This concept has led to the development of a model for the representation of multiple LoDs of 3D city objects in 4D (Arroyo Ohori et al., 2015c), where different methods of linking corresponding features between them have been proposed.

In this paper, we propose a conceptual framework for the creation of a 4D city model that links two states of city objects starting from two existing 3D city models, either revisions of the same dataset or different LoDs of the same city. The framework consists of two parts: the *modelling* aspect, which describes the way that objects are organized in 4D space; and the *methodology*, which describes the steps that are needed in order to construct a 4D city model from two existing 3D city models.

In this section we provide an introduction to the problem that we have studied, as well as an initial description of our framework. In section 2, we review studies related to the topic discussed by this paper and we present the foundation of knowledge upon which this study is based. In section 3, we define the modeling aspect of the framework and the data structures that we use. In section 4, we describe in detail the process that we propose in order to construct a 4D representation of two existing 3D city models by defining the individual steps and their respective requirements and goals. In section 5, we discuss our findings during the conduction of this study, our suggestions and future plans to expand this research.

2. RELATED WORK

2.1 Representation of nD spaces

The notions of time and scale is an important aspect of 2D/3D spatial data in the GIS field. Spatial data are always based on a time snapshot of reality which is continuously changing. In addition, scale is an important concept when working with city models, because they are by definition an abstraction of reality. Therefore, scale can be considered as a linear attribute between more and less detailed representations of the real world.

There is little effort to implement those linear phenomena as additional dimensions on GIS. The most common solution to storing data across time and scale is by representing different instances through individual 2D or 3D objects connected with each other by common identifiers. But other interesting approaches have emerged where time has been integrated as an additional attribute incorporated in the original 2D/3D data structure (Iwamura et al., 2011).

Lately, several authors proposed ways to exploit higher dimension (nD) space in order to incorporate both scale and time additionally to the basic three spatial dimensions. 4D and 5D objects can be used to express a 3D object in all possible variations through scale and time while keeping all correspondences between its features (van Oosterom and Stoter, 2010). While those studies only investigate the theoretical foundation of those approaches, they do prove the benefits of this technique as a way to keep and maintain multi-LOD or spatio-temporal data while enforcing consistency and validity.

First approaches to the implementation of such nD applications have been initially studied by utilising different geometrical and topological data structures (Arroyo Ohori et al., 2015c,b). In his research, Arroyo Ohori concludes that the modelling of non-spatial characteristics as additional geometrical dimensions can be proven a very powerful technique. More specifically, he has identified that due to the extreme complexity of nD space, geometrical data structures fail to provide robust and efficient representation schemes and computational algorithms. Instead, topological representations seem to be more powerful with regard to storage and calculations of higher than 3D spaces, while they can provide additional insight regarding the data they contain (Arroyo Ohori et al., 2015a).

2.2 Combinatorial Maps

A combinatorial map (C-Map) is a data structure that can represent a subdivision of n-dimensional space to cells (Damiand and Lienhardt, 2014). Cells are all i-dimensional objects that are embedded in space, so 0-cells are vertices, 1-cells are edges, 2-cells are faces, 3-cells are volumes, 4-cells are polychora etc.

In section 2.2.1 we provide a description of the C-Map terminology and definitions. In section 2.2.2 we introduce the concept of *isomorphism* which is later used as part of our methodology.

2.2.1 Structure A C-Map is composed by elements called *darts*, which can be considered as the oriented part of an edge that belongs to every combination of i-cells, $\forall i \in \{0, \ldots, n\}$. Every dart contains links to other darts that are adjacent to it and belong to a neighbouring i-cell. Those links are called β_i, where $0 \leq i \leq n$, and every dart contains one β_i $\forall i \in \{0, \ldots, n\}$. A β_i can be better understood as a link to the dart that is incident to the same combination of cells, except for the i-cell. For example, a β_2 in a 3D C-Map links to the dart that belongs to the same edge (1-cell), of the same volume (3-cell) with the current dart, but is part of the neighbour facet (2-cell).

As mentioned before, darts are oriented parts therefore a C-Map is an oriented data structure. That implies certain rules regarding the validity of a C-Map. Every β_i of a dart, with the exception of β_0 and β_1, has to be oppositely oriented to the dart itself. For example, the origin vertex of a dart is the destination vertex of its β_2.

A C-Map contains a constant that defines the *null* dart, expressed as $\varnothing$. By definition $\varnothing$ is linked with itself for all β_i: $\forall i, 0 \leq i \leq n, \beta_i(\varnothing) = \varnothing$. If a dart d has no neighbouring i-cell, then its β_i is assigned to $\varnothing$ and it is called i-free: $\beta_i(d) = \varnothing$.

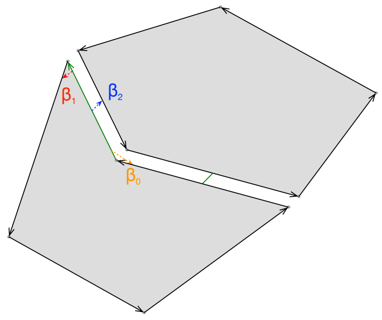

Figure 1. An example of a 2D C-Map. Darts are denoted as black arrows. A dart is highlighted with green color and its β_i pointers are shown.

2.2.2 Map isomorphism Lienhardt (1994), has defined *map isomorphism*, in order to decide if two C-Maps are equal, and *submap isomorphism*, as a way to identify the existence of a pattern of one C-Map in another map. Damiand et al. (2011), have later refined this method for open maps, meaning that isomorphism can apply also to dataset where i-free darts are present.

2.3 Linear Cell Complexes

C-Maps only store information regarding incidence and adjacency between cells, but lack any information regarding geometry, such as shape and size. In other words, they describe the primitives of nD space, but not the ambient space. Therefore, a mechanism for attaching other information on cells has been introduced so that every i-cell can have associated attributes which can allow for the representation of more characteristics of objects. This technique can be used to assign coordinates to all 0-cells, which essentially means that the C-Map can describe geometrical shapes of linear edges through a representation of their boundaries.

An enhanced C-Map data structure that contains information regarding linear geometry is a linear cell complex (LCC). The benefits of a LCC representation based on C-Maps is that it can be easily expanded to as many dimensions as needed, because the underlying structure of boundary representation is not relying on geometrical dimensions. Nevertheless, geometry is still present through the coordinates assigned as attributes to vertices (0-cells).

LCCs are extremely efficient as they are restricting redundancy of information. Given that 0-cells' coordinates are uniquely defined, independently of the number of incident i-cells, a LCC makes more sparing use of storage. This also increases consistency, as in typical geometric representation a point can be repeated as many times as it can be found on the boundaries of incidents polygons.

2.4 Topological reconstruction of 3D city models

A typical representation of a 3D city model is normally through Simple Features (Open Geospatial Consortium, 2011), but recent studies have been conducted on how to utilise LCCs in order to describe 3D city objects.

Diakité et al. (2014a) have proposed EBM-LCC, a specific implementation of a LCC where attributes are used in order to describe 3D buildings which derive from BIM objects and LoD2 CityGML datasets. (Diakité et al., 2014b) have also developed a method for creating EBM-LCCs from the topological reconstruction of buildings in order to automatically extract different LoDs from one main city object. The method starts by building a "soup" of faces from which, then, the volumes are rebuilt according to their adjacency relationship. Although this study only focused on buildings, it may be applicable to any type of objects found in a city model. The disadvantage of this method is that semantics are not preserved during the proposed process, due to the objects being destroyed during the face "soup" creation.

Vitalis et al. (2018) have developed a method to topologically reconstruct a 3D city model into a LCC based on C-Maps, while preserving the semantics and the structure of city objects from the original city model.

2.5 3D city models matching

Since digital models from different sources are being produced for cities, there has been an emerging need to find similarities and differences between models that are created for the same areas in order to maintain and update those datasets. This subject is quite challenging, as it involves the comparison of 3D geometric shapes which has only been studied for basic simplexes (Cignoni et al., 1996).

A comparison approach to 3D city models have been initially investigated by (Pédrinis et al., 2014). This approach only focuses on how to semantically mark objects that have been altered through time, by projecting building objects to their footprints and, then, linking the 2D geometries together. In this study, a CityGML dataset is compared with a 2D cadastral map and the former is altered with information regarding the amount of changes that have been found between the two datasets.

A more complete comparison between 3D city models of the same location has been studied based on the exploitation of the LCC data structure characteristics (Gorszczyk et al., 2016). The purpose of this study is to identify corresponding features between 3D city objects in the dataset in order to highlight topological and semantics differences between them. The process is executed in three steps: first, objects are associated with each other based on a comparison of their bounding boxes; second, the darts are associated based on their geometry and orientation; finally, an isomorphism function is applied on the associated darts to mark topological or semantic differences. While the proposed method is quite effective for the comparison of very similar datasets it would not work practically for datasets originating from different sources or for datasets with significant differences between objects, such as different LoD of the same city model.

2.6 Representation of Correspondences

While there is extensive research regarding the linking of 2D objects across maps of different scales (Filho et al., 1995; van Oosterom and Meijers, 2013), there is little work done on the representation of corresponding features between different 3D objects. A typical solution to represent linkages between corresponding features of different 3D objects is by the use of identifiers between them, such as in the case of LoD implementation on the CityGML specification (Open Geospatial Consortium, 2012).

Arroyo Ohori et al. (2015c), have proposed a solution to the problem while exploring possibilities regarding the modelling of a multiple LoDs in 4D. They propose four methods to link correspondences across the different versions (Figure 2): (a) *simple linking* provides a very straight-forward approach as only vertices of features that exist in two linked versions are connected to each other; (b) *collapse of unmatched cells* adds links between the vertices of features of the higher LoD object that have not been match with a single vertex on the lower LoD object; (c) *modification of topology* introduces vertices on the lower LoD object in order to provide an equal number of edges on both so that there all features can be matched; (d) *matching all to existing* is similar to the collapse method, but instead it forces for a linking of cells of equal dimension.

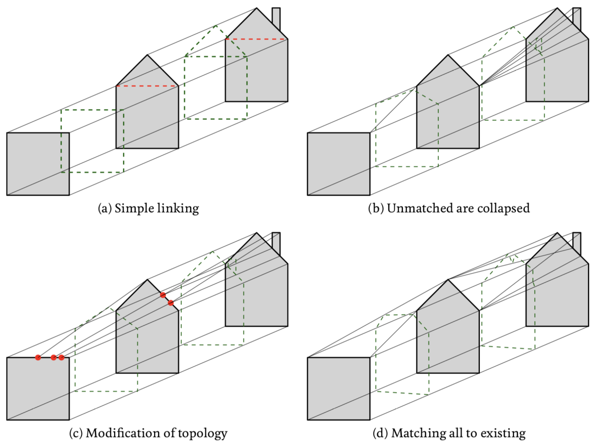
(a) Simple linking
(b) Unmatched are collapsed
(c) Modification of topology
(d) Matching all to existing

Figure 2. A schematic depiction of the four proposed methods for the linking between corresponding features across two objects (Arroyo Ohori et al., 2015c).

3. MODELLING MULTIPLE 3D CITY MODELS IN 4D SPACE

In order to represent multi-state 3D objects in 4D space we define a four-dimensional Euclidean space $\mathbb{R}^4$, where the original three spatial dimensions are appended with an extra dimension for representing the version of an object. The version axis may be associated with a characteristic such as the LoD or revisions of a 3D city object. Therefore, every point can be represented by a four-tuple of coordinates (x, y, z, v).

3.1 Data structure

A 4-dimensional LCC is embedded in 4D space in order to represent the structure of objects. Therefore, the space is subdivided into i-dimensional cells, for $0 \leq i \leq 4$, where 0-cells are vertices, 1-cells are edges, 2-cells are faces, 3-cells are volumes and 4-cells are polychora. Every i-dimensional object is described by their $(i-1)$-dimensional boundaries, so an edge is described by its boundary vertices, a face by its boundary edges etc.

3.2 4D Models

It is important to clarify the principles of a 4D model and how it can be associated with a multi-state 3D object representation. An intuitive process is to build upon the description of a simple 3D cube and its fourth-dimensional analogue, a tesseract.

A cube is a 3-cell (volume) which is bounded by 2-cells (faces) perpendicular to each other. In order to fully describe the cube, 6 faces are needed. We could denote the faces with names, in order to identify them: *top, bottom, left, right, front* and *back*. If the cube was intended to describe a pair of two-dimensional features by it's *top* and *bottom* faces, then the other four faces would describe their links. Therefore, the *left, right, front* and *back* may be considered as intermediate faces that connect the *top* and *bottom*.

Analogously, a tesseract (figure 3) is a 4-cell (polychoron) that is bound by 3-cells (volumes). In this case, 8 volumes are needed in order to fully describe the polychoron. Added to the *top, bottom, left, right, front* and *back* volumes, there is a pair of new primitives: the *outer* and *inner* volumes. If the *outer* and *inner* volume are to describe the two 3D objects, then the other 6 volumes can be called intermediate as their purpose is to connect them and describe their links.

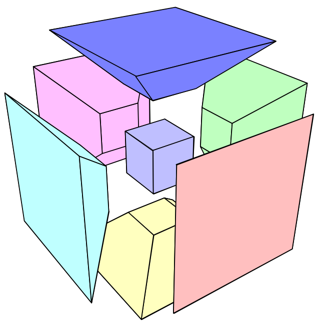

Figure 3. A visualisation of the volumes that form a tesseract, projected in 3D ($\mathbb{R}^3$). The *inner*, *front*, *back*, *left*, *right*, *top* and *bottom* volumes are highlighted with different colours. For visualisation purposes, they have been moved apart from each other. In order to underpin the clarity of the figure, the *outer* volume is not shown.

While a cube or a tesseract are "ideal" cases, they do highlight the pattern that can be used in order to exploit the topological structure of an (i+1)-dimensional object in order to describe a pair of i-dimensional objects. In fact, our intention is to use prismatic 4D objects where there is always an *outer* and *inner* volume that describe the two initial 3D objects. Then, there must be a number of perpendicular volumes that can link them, which can be associated to the number of common features that have been identified on the two objects.

3.3 Semantics

LCCs has an internal mechanism for storing i-dimensional information through the concept of i-attributes. Therefore, it is possible to attach any kind of data to an i-cell. This mechanism can be easily utilised so that the semantic information of the original objects are retained in the final data structure.

In other words, city object information can be associated to volumes as lists of key-value pairs through 3-attributes on the LCC. Any additional information assigned to individual surfaces may be associated with the respective faces similarly as 2-attributes.

4. METHODOLOGY

In order to combine existing datasets in a final 4D city model we established a methodology which prescribes the six steps that need to be taken. This steps are: (a) *Convert 3D City Model to LCCs*, where the initial datasets are topologically reconstructed to LCCs; (b) *Associate objects between LCCs*, where matches of similar city objects between datasets, as volumes in the LCCs, are found; (c) *Create candidate pairs of darts across LCCs*, where matched volumes are compared in order to find candidate pairs of edges, as darts in the LCCs; (d) *Identify corresponding features*, where candidate pairs are "filtered" in order to find paths of corresponding features between the objects; (e) *Construct prismatic 3-cells*, where intermediate volumes are constructed in order to represent the link between the features; (f) *Sew Common Darts across 3-cells*, where object and intermediate volumes are combined in the final 4D city model;

We define those steps as individual processes that have specific requirements as input and goals as output. This, builds a foundation of actions that results in the construction of a 4D model which represent both the original 3D datasets and their correspondences in one 4D topological data structure.

In the remainder of this section we detail the steps with considerations for their implementation.

4.1 Convert 3D City Model to LCCs

- Expected input: 2x 3D city models in SFS.
- Targeted output: 2x 3D LCCs with semantics.

In order to process the 3D objects in a meaningful manner, they have to be converted in a similar data structure as the final result. 3D city models are being expressed as Simple Features, which is based mostly on geometry. Instead, a LCC representation can be scaled easier to the 4th dimension and allows for the integration of all three components of a city model: geometry, topology and semantics. In fact, this integration of information provides a better foundation for the processing that will occur in the next steps.

An important concern of this step is that semantics and the original structure of objects in the city model is retained, so that those information can be used in the following steps of the methodology. Section 2.4 refers to the studies that can be used in order to accomplish that.

4.2 Associate objects between LCCs

- Expected input: 2x 3D LCCs with semantics.
- Targeted output: Pairs of 3-cells accross the LCCs

The aim of this step is to establish associations between common city objects across the two models. Given that city objects of the original 3D city models have been converted to 3-cells during the topological reconstruction, this step is about finding pairs of 3-cells that correspond to the same real-world objects, for example the same building.

This can be straightforward when semantics which can provide linkage information between city objects exist in both dataset. For instance, in case there are common identifiers between the city objects in the models, this can be considered as a trivial solution to associating 3-cells, given that the identifiers are retained during the topological reconstruction (section 4.1). If linking information is not present, then the object association could still be possible based on semantics, as long as there is enough information in both models so that the objects can be linked. That would involve some heuristic rules that identifies common objects according to equal or similar attributes. For example, in the case of buildings where the address is present in both datasets, a normalised text of the address can be used.

In case there is not enough semantic information in both datasets, geometry can be used to associate the volumes. Again, this involves some heuristic approach where objects will be associated according to criteria related to their geometric characteristics. Those criteria may vary, according to the application and the specifics of the two datasets.

When the geometry of the two models is known to be vastly different, for instance between two LoDs of the same area, then some loose criteria can be used. For example, the percentage of the volume of their intersection in respect to their original volume. But in cases where the two models are expected to have many similarities, such as different versions of the same city model with minor changes, then the criteria can be more strict, for example if they share a common side of their bounding box.

It is important to verify the compatibility of the two datasets for any geometric comparison. Sometimes, the models need to be pre-processed in order to proceed to the association of volumes, as described before. For instance, when datasets belong to different spatial reference systems (SRS) a reprojection has to be performed. In addition, the reference of the elevation of the datasets has to be taken into account. Many times, 3D city models may not contain any terrain elevation data into consideration, meaning that building floors are all “flattened” to zero height. Heterogeneous elevation information may vastly alter the way the comparison of volumes is undertaken. Thus, it might require a pre-processing action, such as the “flattening” of both datasets to zero elevation, or the alteration of the way objects are associated, e.g. by the comparison of the footprints of buildings.

We need to clarify that although the output of this step is composed by pairs of volumes, this does not imply a one-to-one relationship between the matched 3-cells. In fact, given that during the topological reconstruction a city object might have resulted in multiple 3-cells, it is possible that one 3-cell on one dataset is associated with many 3-cells from the other. Therefore, one volume may be contained in more than one pairs.

4.3 Create candidate pairs of darts across LCCs

- Expected input: Pairs of 3-cells
- Targeted output: Pairs of darts

After volumes have been associated, there is a need to identify edges of corresponding features between them. This is done in two steps, where the first is to construct a list of candidate pairs of darts between the two LCCs. The second step (section 4.4) is to filter the candidate pairs and construct the final matches.

At this stage, it is unlikely that there is semantic information which could provide details on potential matches of darts. Therefore, this is a step that has to be exclusively based on heuristic rules in order to find potential candidates.

The methodology according to which this could be established is closely related to the datasets available and the particular application needs. In cases of very similar datasets, such as in a versioning application, this could be as straightforward as edges with the same coordinates at their endpoints. Alternatively, in cases of datasets with significant differences, more loose criteria can be used, such as lines with a closest distance under a threshold.

4.4 Identify corresponding features

- Expected input: Pairs of darts
- Targeted output: Pair of dart paths

The aim of this step is to establish the final pairs of darts that will be linked together. This can be alternatively considered as a filtering of the candidate darts from the previous step. In addition, it is intended to construct a pair of paths that outline the corresponding features between the objects. In other words, some candidate pairs of darts will be rejected as incompatible while others will conclude to final closed loops of darts that will have to be linked.

This step involves the application of a traversal algorithm and a predicate function, similar to the original isomorphism algorithm (section 2.2.2). The algorithm starts the traversal according to the input candidate pairs. In every step, the predicate function is applied to the pair of darts. Then, the next pair of darts is picked and evaluated.

In the isomorphism algorithm, the predicate is actually an evaluation of the darts compatibility. Originally, this is undertaken by the comparison of the mappings between the darts in every iteration, meaning that the two darts must be equally i-free, for $0 \leq i \leq 3$. While this function fully complies with the definition of the topological isomorphism, an enhanced predicate is needed in order to take into account the additional geometric and semantic information of the dataset. Therefore, the compatibility of darts should also incorporate the i-attributes between the darts (given that geometry is assigned to 0-attributes and semantics to i-attributes for $i \geq 1$).

In cases of datasets with much resemblance, the modifications might have to be subtle. The predicate function can be “strict” when searching for dart mappings, given that the topology of corresponding features is expected to be mostly identical. The same applies to the way the traversal is performed. Same as in the original isomorphism algorithm, a simple traversal should be sufficient. That means that the darts are compared one-by-one by visiting their β_1.

Nevertheless, more relaxed implementations of the traversal and the predicate function are needed for dataset where more than a few changes are present. For example, the compatibility criterion

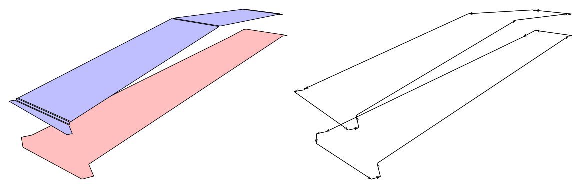
Figure 4. The output of this step should be a pair of paths that meat the defined criteria. In this figure, the roof of two different LoDs of the same building is shown as 2-cells (faces) on the left and the output of this step is shown as a path of darts on the right. Notice how the darts that may belong to different faces of the original objects, meaning that the traversal did not follow only the β_1s.

could be implemented as a threshold between the origin vertex coordinates of the darts. As for the traversal, a more sophisticated search with a deep search on the two LCC graphs will probably be needed, given that the topology of the corresponding features is expected to differ. For instance, in cases of objects of different LoDs the darts of one face of the lower LoD model can be associated with the darts of multiple faces of the higher LoD (figure 4). Therefore, the traversal will have to follow both β_1 and β_2 in order to identify a pair of paths that can be associated.

4.5 Construct prismatic 3-cells

- Expected input: Pairs of dart paths
- Targeted output: 3-cells (volumes) linking the features in 3D

This step is about the creation of the intermediate 3-cells (volumes) that link the identified corresponding features. As described in section 4.4, the corresponding features can be represented by the two paths of darts that bounds them. A prismatic intermediate volume, then, can be constructed where the bottom and top sides are the features described by those paths.

Initially, all vertices of the two LCCs are assigned the addition v coordinate that represents their position in the fourth dimension. Then, the pairs of paths are used in order to identify faces that will be linked. Finally, those faces are connected by vertical faces across the v-axis in order to create the intermediate volumes (figure 5).

At this stage, the faces that represent the vertical correspondences can be constructed according to the linking methods described in section 2.6. We have distinguished the following three cases.

The *simple linking* (figure 5a) can be represented by the creation of complex faces that connect parts of the two paths. While this can be trivial, it may result in the construction of non-planar vertical faces.

The *collapse of unmatched cells* (figure 5b) can be constructed by triangulating across the edges of the two paths. In this case, every dart of the top path that is associated to a dart of the bottom path would create a quadrilateral face. The darts that could not be associated, should be triangulated with the closest point of the other path. This is not a trivial process, as the geometric complexity of the paths might introduce some degenerate cases where a dart cannot be triangulated with any point of the other path as it will intersect other darts. In this case, this darts have to be identified and triangulated first with a point of the same path. Then, a new dart will be created that can be associated with the previous point or dart of its two sides. Figure 5b depicts such a case.

The *modification of topology* (figure 5c) can be constructed by projecting the vertices of the two paths to each other and, then, creating quadrilateral polygons. This involves heavy modification of the original objects as well, as the original faces have to follow the topological changes that occur on the bottom and top sides of the intermediate volume. This method is subject to the same issue as mentioned before. There may be a dart of the top path that cannot be associated with any dart of the bottom without creating a self-intersecting face. Similarly with collapse method, this can be avoided by a triangulation that would produce a new dart which can be find a valid match on the other path.

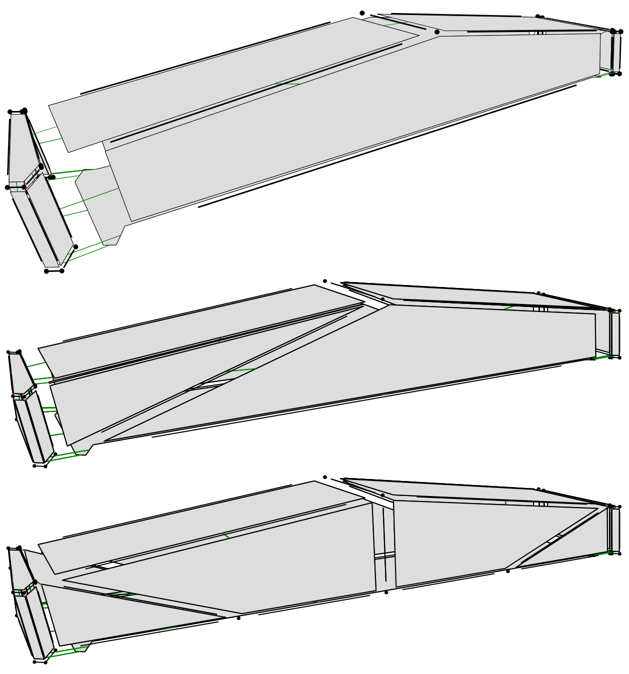

Figure 5. The intermediate volume between the roofs of two models. The volume is created after the darts have been associated with each other according to the simple-linking (top), collapse to closest (middle) and topology modification (bottom) method.

4.6 Sew Common Darts across 3-cells

- Expected input: Pairs of dart paths and intermediate volumes
- Targeted output: Final 4-cells with all correspondences incorporated.

The final step is the construction of the final 4D data structure. At this point, the LCC structure is filled with a soup of the original 3-cells (the outer and inner volumes) and the intermediate 3-cells.

As mention in section 2.2.1, a C-Map is an oriented data structure. Therefore, in order to 3-sew them in 4D object, there has to be some modification of their orientation so that adjacent-to-be darts have opposite orientations before they are sewed. This only has to happen in one of the two volumes. We can choose arbitrarily that the dataset that is given the higher v coordinates has to be flipped. This can be easily established by swapping all β_0 to β_1 and vice-versa.

After the faces are all correctly oriented, the 3-sew operations can be applied according to the geometrical information. It is expected that all faces of the two datasets will find mirrored faces on intermediate volumes, therefore this operations is trivial.

5. CONCLUSIONS

In this paper, we presented a framework for the construction of 4D city models from a pair of existing 3D city models. This involves the identification of corresponding features between the datasets and the creation of a final data structure that represents the correspondences in the fourth dimension. The purpose of this framework is to divide this process to individual steps with specific requirements and goals that can be implemented accordingly.

By listing the steps of our methodology and providing the examples above we have shown that it is possible to construct a 4D representation of two 3D city models by utilising the LCC data structure. We have also highlighted the potential challenges and issues that need to be resolved in order to achieve such a result.

At its current form, the methodology is subject to limitations. First, the input should be topologically valid. While this might not produce any obstacles to the process, the existence of topologically invalid objects would possibly conclude to a non-watertight output. Second, it is designed to work exclusively for two datasets. How this can be extended for more than two datasets, needs further study. For instance, when applying the topological-modification linking method, every modification to one version of the object would require the respective altering of all other 3D objects related to it in the dataset.

Further research is needed on the implementation of every step. As applications and data vary, there are many possibilities for how to accomplish this. Nevertheless, there are common issues that need to be addressed and we have provided first considerations on how they can be approached.

There is, also, room for incorporating the output of other research fields as part of this methodology. Machine learning could be applicable in steps where heuristic approaches were mentioned before.

In the future, we plan to develop specific software implementations that can undertake the necessary actions to accomplish this framework. More precisely, we intend to construct 4D city models by applying the different linking methods in order to evaluate their characteristics.

ACKNOWLEDGEMENTS

This project has received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (grant agreement No 677312 UMnD).

References

Arroyo Ohori, K., Ledoux, H. and Stoter, J., 2015a. An evaluation and classification of nD topological data structures for the representation of objects in a higher-dimensional GIS. *International Journal of Geographical Information Science* 29(5), pp. 825–849.

Arroyo Ohori, K., Ledoux, H. and Stoter, J., 2015b. Storing a 3D city model, its levels of detail and the correspondences between objects as a 4D combinatorial map. In: A. A. Rahman, U. Isikdag and F. A. Castro (eds), *ISPRS Joint International Geoinformation Conference 2015*, ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences, Vol. II–2/W2, ISPRS, Kuala Lumpur, Malaysia, pp. 1–8. ISSN: 2194–9042 (Print), 2194–9050 (Internet and USB).

Arroyo Ohori, K., Ledoux, H., Biljecki, F. and Stoter, J., 2015c. Modeling a 3d city model and its levels of detail as a true 4d model. *ISPRS International Journal of Geo-Information* 4(3), pp. 1055–1075.

Biljecki, F., Ledoux, H. and Stoter, J., 2014a. Error propagation in the computation of volumes in 3d city models with the monte carlo method. *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences* 2(2), pp. 31.

Biljecki, F., Ledoux, H. and Stoter, J., 2014b. Improving the consistency of multi-LOD CityGML datasets by removing redundancy. In: *Lecture Notes in Geoinformation and Cartography*, Springer International Publishing, pp. 1–17.

Choi, J. and Lee, J., 2009. *3D Geo-Network for Agent-based Building Evacuation Simulation*. Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 283–299.

Cignoni, P., Rocchini, C. and Scopigno, R., 1996. Metro: Measuring error on simplified surfaces. Technical report, Paris, France.

Damiand, G. and Lienhardt, P., 2014. *Combinatorial Maps: Efficient Data Structures for Computer Graphics and Image Processing*. 1st edn, A. K. Peters, Ltd., Natick, MA, USA.

Damiand, G., Solnon, C., de la Higuera, C., Janodet, J.-C. and Samuel, É., 2011. Polynomial algorithms for subisomorphism of nD open combinatorial maps. *Computer Vision and Image Understanding* 115(7), pp. 996–1010.

Diakité, A. A., Damiand, G. and Gesquière, G., 2014a. Automatic Semantic Labelling of 3D Buildings Based on Geometric and Topological Information. In: *Proc. of 9th International 3DGeoInfo Conference (3DGeoInfo)*, 3DGeoInfo conference proceedings series, Karlsruhe Institute of Technology, Dubaï, United Arab Emirates.

Diakité, A. A., Damiand, G. and Van Maercke, D., 2014b. Topological Reconstruction of Complex 3D Buildings and Automatic Extraction of Levels of Detail. In: V. T. Gonzalo Besuievsky (ed.), *Eurographics Workshop on Urban Data Modelling and Visualisation*, Eurographics Association, Strasbourg, France, pp. 25–30.

Filho, W. C., Figueiredo, L. H. D., Gattass, M. and Carvalho, P. C., 1995. A topological data structure for hierarchical planar subdivisions. In: *4th SIAM Conference on Geometric Design*.

Gorszczyk, B., Damiand, G., Servigne, S., Diakité, A. and Gesquière, G., 2016. An Automatic Comparison Approach to Detect Errors on 3D City Models. In: V. Tourre and F. Biljecki (eds), *Eurographics Workshop on Urban Data Modelling and Visualisation*, The Eurographics Association.

Iwamura, K., Muro, K., Ishimaru, N. and Fukushima, M., 2011. 4d-GIS (4 dimensional GIS) as spatial-temporal data mining platform and its application to managementand monitoring of large-scale infrastructures. In: *Proceedings 2011 IEEE International Conference on Spatial Data Mining and Geographical Knowledge Services*, IEEE.

Lienhardt, P., 1994. N-dimensional generalized combinatorial maps and cellular quasi-manifolds. *International Journal of Computational Geometry & Applications* 04(03), pp. 275–324.

Open Geospatial Consortium, 2011. OpenGIS Implementation Standard for Geographic information - Simple feature access.

Open Geospatial Consortium, 2012. City Geography Markup Language (CityGML) Encoding Standard, version: 2.0.0.

Pédrinis, F., Morel, M. and Gesquière, G., 2014. Change detection of cities. In: *Lecture Notes in Geoinformation and Cartography*, Springer International Publishing, pp. 123–139.

Steinhage, V., Behley, J., Meisel, S. and Cremers, A. B., 2010. Automated updating and maintenance of 3d city models. In: *ISPRS joint workshop on" Core Spatial Databases-Updating, Maintenance and Services-from Theory to Practice*, pp. 1–6.

van Oosterom, P. and Meijers, M., 2013. Vario-scale data structures supporting smooth zoom and progressive transfer of 2d and 3d data. *International Journal of Geographical Information Science* 28(3), pp. 455–478.

van Oosterom, P. and Stoter, J., 2010. 5D data modelling: Full integration of 2D/3D space, time and scale dimensions. In: S. I. Fabrikant, T. Reichenbacher, M. van Kreveld and C. Schlieder (eds), *Geographic Information Science: 6th International Conference, GIScience 2010, Zurich, Switzerland, September 14-17, 2010. Proceedings*, Springer Berlin Heidelberg, pp. 311–324.

Vitalis, S., Ohori, K. A. and Stoter, J., 2018. Topological reconstruction of 3D city models with preservation of semantics. In: A. Mansourian, P. Pilesjö, L. Harrie and R. von Lammeren (eds), *Geospatial Technologies for All : short papers, posters and poster abstracts of the 21th AGILE Conference on Geographic Information Science*, Lund University. Accessible through https://agile-online.org/index.php/conference/proceedings/proceedings-2018.

Zhivov, A. M., Case, M. P., Jank, R., Eicker, U. and Booth, S., 2017. Planning tools to simulate and optimize neighborhood energy systems. In: *Green Defense Technology*, Springer.