TU Delft

Modelling of semantically rich multi-LoD GIS data in 4D

PhD Proposal

Stylianos Vitalis

3D geoinformation
Department of Urbanism
Faculty of Architecture and the Built Environment
Delft University of Technology

January 24, 2018

Contents

Abbreviations

nD	n-Dimensional
API	Application Programming Interface
BIM	Building Information Modeling
C-Map	Combinatorial Map
CAD	Computer Aided Design
CDT	Constrained Delaunay Triangulation
CGAL	The Computational Geometry Algorithms Library
CityGML	City Geography Markup Language
CityJSON	City JavaScript Object Notation
DAE	COLLADA Digital Asset Exchange
DCEL	Doubly Connected Edge List
DXF	Drawing Exchange Format
EBM-LCC	Enriched Building Model - Linear Cell Complex
GDAL	Geospatial Data Abstraction Library
GIS	Geographic Information System
glTF	Graphics Library Transmission Format
GML	Geography Markup Language
GMLAS	GML driven by Application Schemas
GUI	Graphical User Interface
IDE	Integrated Development Environment
IFC	Industry Foundation Classes

JSON . . . . . JavaScript Object Notation

KML . . . . . Keyhole Markup Language

LCC . . . . . Linear Cell Complex

LoD . . . . . Level of Detail

NMCA . . . . . National Mapping and Cadastral Agency

NoSQL . . . . . Not-only SQL

NSDI . . . . . National Spatial Data Infratsructure

OBJ . . . . . Wavefront Object

OLAP . . . . . Online Analytical Processing

RDBMS . . . . . Relational DataBase Management System

SFS . . . . . Simple Features Specification

UMnD . . . . . Urban Modelling in Higher Dimensions

XML . . . . . eXtensible Markup Language

Chapter 1

Introduction

1.1 Motivation

As 3D technologies become more mature and countries work towards National Spatial Data Infratsructures (NSDIs), there is an increased need for public and private organisations - such as National Mapping and Cadastral Agencies (NMCAs) - to adopt the use of 3D data and, more specifically, to utilise 3D city models [Stoter et al., 2014]. These datasets are used in applications related to simulation and analysis of real world phenomena in procedures. For example, city models can be used in order to analyse and optimise community energy planning [Zhivov et al., 2017] or to simulate and study the behaviour of a crowd on evacuation scenarios [Choi and Lee, 2009].

The main challenge when processing 3D data is the high complexity of computation and manipulation of geometric information in all three dimensions. In GIS, the calculation of spatial relationships between objects, such as intersection, overlap and touch, are critical for conducting any analysis on the geometries of a dataset. While those concepts have been thoroughly studied in 2D and there are numerous and robust implementations of them, there is limited research on the subject of 3D topological relations calculation [Zlatanova, 2000, Borrmann and Rank, 2009]. The most common implementations for 3D objects' storage are mainly ad-hoc adaptations of data structures that were originally designed for 2D data [Arroyo Ohori, 2016]. Therefore, 3D data are not easy to process and that results in higher uncertainty related to the accuracy and validity of 3D city models [Biljecki et al., 2014a].

Furthermore, the amount of computational resources needed to analyse 3D data in order to process the datasets makes it necessary to optimise the initial model before it can be used for a specific use case. In other words, different applications require various amount of details and it is extremely difficult to extract alternative versions of a city model, each one optimised for specific requirements, from one "master" dataset. Therefore, the collection and use of various models for representing the same city objects has emerged on 3D city

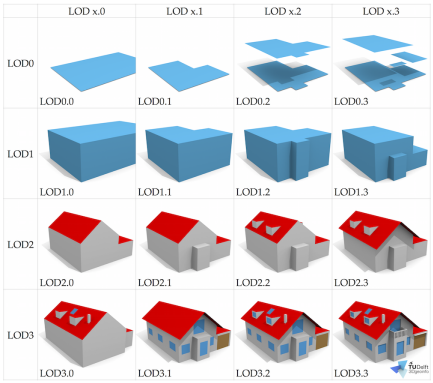


Figure 1.1: The proposed concept of advanced LoD on CityGML specification [Biljecki, 2017].

models [Biljecki et al., 2015a]

For this reason, the concept of LoD has been also incorporated in most implementations of modern 3D city models' representations (Figure 1.1), which allows us to store multiple representations of the same real-world objects in various amount of detail in the same dataset [Open Geospatial Consortium, 2012a, Biljecki et al., 2016a]. The LoD concept has initially been introduced on 3D graphics [Luebke et al., 2002] and has been implemented in most modern 3D graphics' libraries, such as Unity[1] and CesiumJS[2].

Although the concept of LoD is utilised by most current 3D city datasets that have been produced by different authors, there is still a diversity on how they are implemented. This is mostly related to the use of different techniques of data acquisition and processing during the creation of the city models [Aringer and Roschlaub, 2014, Duan and Lafarge, 2016]. The datasets are produced by different data sources and mostly serve very specific purposes, while there is no link between common features of two different models at different LoDs in the same city model. For instance, there is no way of knowing which parts of two models at different LoDs form the same real-world wall of a building.

Current approaches to the subject try to solve the issue by using simple identifiers to link models of different LoDs at the level of city objects. Unfortunately, this is a rather simplistic approach which has been barely adopted, therefore causing issues related to the consistency of city models [Biljecki et al., 2014b] and making the update of such multi-LoD datasets a very challenging process [Steinhage et al., 2010]. Hence, the concept of LoD is mostly used as a meta-data term that describes the amount of detail in a dataset, instead of serving its original purpose of a mechanism to store multiple versions of the city model in one dataset.

For this reason, some initial efforts have emerged that investigate the use of

[1] `https://unity3d.com`
[2] `https://cesiumjs.org`

certain topological structures as a way to link common features between objects of different LoD representations of the same city model [Biljecki et al., 2014b]. As an extension to traditional linking between different 3D objects though identifiers, the representation of LoD as the fourth dimension has been investigated in an early level by Arroyo Ohori et al. [2015a]. While this attempt mostly focuses on the geometrical aspect of a 4D representation of a multi-LoD object, it seems that such an approach could introduce a methodology for storing and maintaining data of different details so that consistency is enforced. Nevertheless, further justification regarding the implementation of such a technique is needed and details related to the data structures and operations that can handle 4D multi-LOD representations remain unexplored.

1.2 Research Objective

My research objective is to: *develop a representation of multiple 3D city models of the same city objects and their linked correspondences with semantics in 4D space.*

The research questions and sub-objectives are further defined in Chapter 3.

1.3 Scope of the Research

- The research is based on combining geometry and semantics in 4D.
- The fourth dimension is mainly intended to represent LoD, but the notion of time might be used as well in order to test and assist the development of the framework that will be proposed. It might be, though, that spatio-temporal data can be incorporated to the datasets through semantics.
- Any research output that results in computer software should be applicable to real-world data.
- Any research output that will be implemented in computer programs, will be released as open source.
- This research will be conducted as part of a larger project examining Urban Modelling in Higher Dimensions (UMnD) where my work will contribute to the overarching goal of developing a 4D data model that stores the application specific LoDs of urban objects as an additional dimension to the three spatial ones. Extraction of 3D objects from the 4D data structure is not a part of this research.

1.4 Structure of this Document

This proposal is divided in 5 chapters:

- This chapter introduces the concept and describes the challenges that need to be undertaken, while briefly defining the topic and the scope of research.
- Chapter 2 presents the related work and the theoretical aspects that are necessary in order to conduct my research.
- Chapter 3 gives a detailed description of the proposed research and presents the methodology that I intend to follow during my PhD.
- Chapter 4 provides a view on the initial results after conducting the first year of my research, as well as some conclusions on my experience with existing software and data.
- Chapter 5 describes the planning, practical aspects and technicalities that are related to my research.

Chapter 2

Related Work

This chapter examines the work that has been conducted on 3D city models and n-Dimensional (nD) spaces, as they affect my research topic. It provides a background overview of the basic notions and theory that are needed in order to understand the scope of this research. Finally, it contains the theoretical foundations that are related to the practical aspects I have undertaken during the first year of my PhD.

Section 2.1 introduces the foundations of geometry and some basic theory needed in order to solve some problems during the manipulation of 3D geometric data, such as the triangulation of polygons. Section 2.2 presents the data structures that have been adapted for the representation of 3D data in computer science and GIS. Section 2.3 introduces the concept of 3D city models, their specifications and the implementation of LoD in them. Section 2.4 reviews current applications that use the nD space in the GIS field. Section 2.5 introduces the foundations of topology and its use on GIS applications. Section 2.6 focuses on the most well established topological data structures that are used in GIS and introduces Linear Cell Complexes and Combinatorial Maps (C-Maps), which will be used extensively during my research. Section 2.7 presents existing research on the field of 3D geometric comparison and, mostly, on city models, which is essential in order to match features between different datasets. Section 2.8 reviews current studies related to the representation of links between correspondences across different objects. Section 2.9 introduces graph databases and their basic concepts, as they will be explored as a way to store city models.

2.1 Geometry

Geometry is a branch of mathematics that studies the rules of relations in space and has been formalised by Euclid [Fitzpatrick and Heiberg, 2007]. Later on, the concept of numerical representation of points through coordinates has been introduced by Descartes [1637] and de Fermat [1679], leading to the foundation of today's concept for geometrical representation named *Analytical* or *Cartesian*

Geometry. While coordinates are an explicit representation of points on an nD space, they need to be combined with several data structures in order to allow for the representation of more complex geometric elements, such as lines, planes and volumes.

Section 2.1.1 introduces some basic theory that is needed in order to challenge certain geometric calculations that will be needed during my research, mostly related to the triangulation of 3D polygons.

2.1.1 Vector Calculus on Planes

A plane is described by the following equation:

$$Ax + By + Cz + D = 0 \tag{2.1}$$

where, the three coefficients (A, B, C) describe the plane's normal $(\vec{n})$.

The normal can be calculated as:

$$\vec{n} = \vec{p_\alpha} \times \vec{p_\beta}, \tag{2.2}$$

where, $\vec{p_\alpha}$ and $\vec{p_\beta}$ are vectors created from two edges of the polygon. But this approach can fail when the points that form the edges are too close to each other due to precision issues with computer calculations, thus rendering this method unreliable for practical applications. Alternatively, Newell's method can be applied to calculate the normal based on all points of the polygon, which is a more robust solution [Tampieri, 1992].

According to Newell, the three coefficients of the plane of the polygon can be calculated as:

$$A = \sum_{i=1}^{n} (y_i - y_{i\oplus 1})(z_i + z_{i\oplus 1}), \tag{2.3}$$

$$B = \sum_{i=1}^{n} (z_i - z_{i\oplus 1})(x_i + x_{i\oplus 1}), \tag{2.4}$$

$$C = \sum_{i=1}^{n} (x_i - x_{i\oplus 1})(y_i + y_{i\oplus 1}), \tag{2.5}$$

where n is the number of points in the polygon and "$\oplus$" stands for addition modulo n.

Every point x that lies on the plane should satisfy the equation:

$$(\overrightarrow{x - p}) \cdot \vec{n}, \tag{2.6}$$

where p is a reference point on the plane, which can be either a random point of a polygon or the "center of gravity" given by:

$$p = \frac{1}{n} \sum_{i=1}^{n} p_i. \tag{2.7}$$

Based on Equation (2.6), D can be computed as:

$$D = -\vec{p} \cdot \vec{n} \tag{2.8}$$

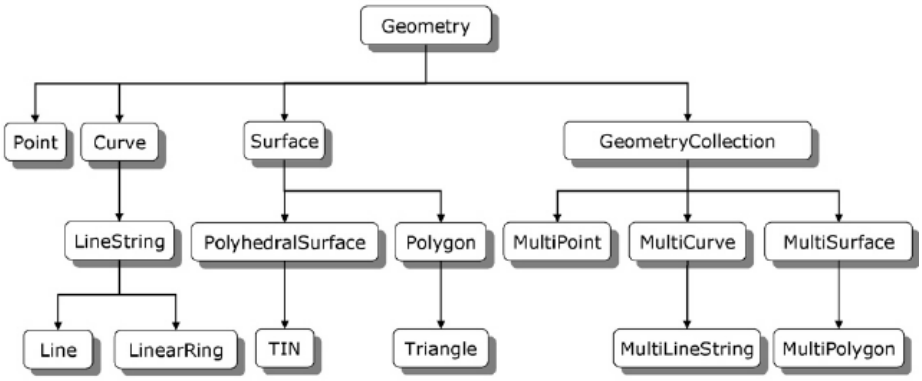


Figure 2.1: A schematic representation of the Simple Features Specification (SFS) data model [Open Geospatial Consortium, 2011].

2.2 3D Geometrical Representations

In computer science several data structures have been proposed and implemented for the purpose of 3D modelling in uses such as 3D graphics, Computer Aided Design (CAD) or GIS applications. Most notably, modern computer hardware and low-level Application Programming Interfaces (APIs) have been using 2D simplicial complexes (triangles) for the representation of 3D geometries [Segal and Akeley, 2009]. But for higher level applications, such as storage and editing of 3D Models, more comprehensive representation have been used.

3D Models can be divided in two categories: (a) *solids*, are models that define the volume of the object that they represent and (b) *boundaries*, are models that represent the boundary surfaces of an object, but not its volume.

The most common 3D models representation in computer graphics for higher level 3D applications is *polygonal modeling*, where points in 3D space are linked by lines in order to form polygons [Russo, 2006]. The main advantage of this technique is that it is extremely efficient for visualisation purposes, as polygons can be easily and efficiently triangulated, thus making 3D models compatible with modern graphics acceleration hardware. Furthermore, they are more storage efficient, as less vertices are needed to be repeated when storing a polygon instead of the triangles that form it. However, the polygons must be planar, therefore curved surfaces can only be approximated by many polygons.

In GIS applications a polygonal modelling approach has been adapted for the geometrical representation regarding storage and exchange of geographic data, which has been formalised through the SFS [Open Geospatial Consortium, 2011]. According to the SFS, a polygon is stored as the list of points that form its boundaries. In a similar fashion, 3D objects such as volumes can be defined by the polygons the bound them (Figure 2.1).

Even though SFS has been implemented by almost all GIS applications, most software is still lacking in support for visualisation and processing of complex

geometric elements, such as multi-surfaces, which are needed for representing 3D objects. In an effort to overcome such limitations, many applications use the term "3D" while they are implementing a 2.5D approach [Arroyo Ohori, 2016]. For example, many times buildings are represented as the 2D polygons of their footprint and a single number that represents their height. Those representations are making the transition from a 2D to a 3D era much more challenging, as the term 3D is used loosely which makes it difficult to distinguish true 3D applications from pseudo-3D ones.

2.3 3D City Models

3D city models are 3-dimensional geometrical representations of the urban environment [Billen et al., 2014]. They can be considered a specialisation of 3D models (as described in Section 2.2) according to two perspectives. Firstly they model the 3D structure of the real world and, particularly, of urban areas. Secondly they contain semantic information for the 3D objects they describe, which essentially defines them as GIS data.

It is evident that 3D city models are being increasingly adopted by more cities and organisations, in order to facilitate the management of heterogeneous information in one place. For example, the Port of Rotterdam is working on a 3D Spatial Data Infratsructure (SDI) in order to support the information flow of its organisation and to provide a platform for information exchange with the stakeholders of the entire region [Zlatanova and Beetz, 2012]. The Dutch Kadaster is also working towards that direction through the development of a 3D map[1] for the whole country. The first cadastral registration of rights through a 3D plan has also been accomplished already in the Netherlands [Stoter et al., 2017].

Many applications of 3D city models can be found in various fields: visibility analysis, energy demand estimation, traffic planning, property registration, noise simulation, etc. For instance, Kaden and Kolbe [2014] have concluded a city-wide estimation of the energy demands (heating, electricity and warm water) of buildings by utilising the 3D model of the city of Berlin. Stoter et al. [2008] have generated a 3D noise map in order to better assess the noise impact on the urban environment, compared to traditional 2D simulations. Varduhn et al. [2014] try to improve assessment of flood risk and potential damage by using 3D information from a city model, which can assist risk management, evacuation planning, and utility management.

The following sections provide more details about 3D city models and their implementations. Section 2.3.1, reviews various specifications that are used for the representation of 3D city objects and describes the main 3D city model formats. Section 2.3.2 focuses on the implementation of the LoD concept in 3D city models and mostly in CityGML. Section 2.3.3 provides a brief overview of multi-LoD dataset existence. Finally, Section 2.3.4 reviews the software tools and libraries that can be used to manipulate 3D city model files.

[1] `https://www.kadaster.nl/-/3d-kaart-nl`

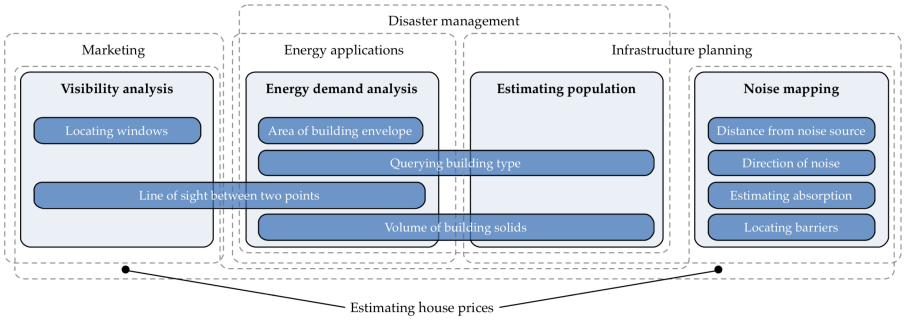


Figure 2.2: An overview of the use cases of 3D city models applications [Biljecki et al., 2015b].

2.3.1 Specifications of 3D City Objects

There are several standards that have been developed for the description of 3D urban objects. For civil engineering and construction purposes, Industry Foundation Classes (IFC)[2] and Drawing Exchange Format (DXF)[3] are the most prominent, utilised in Building Information Modeling (BIM) and CAD software. For visualisation purposes, COLLADA Digital Asset Exchange (DAE)[4] and Wavefront Object (OBJ)[5] are used extensively as exchange formats between specialised software and typical 3D graphics viewers. Web 3D GIS applications and libraries, such as Google Earth[6], are using Keyhole Markup Language (KML)[7] and Graphics Library Transmission Format (glTF)[8] in order to represent and store 3D geometric data mostly for visualisation purposes.

The most concrete and complete standard for the description of 3D city models is the CityGML specification [Open Geospatial Consortium, 2012a]. It forms a specialisation of the Geography Markup Language (GML) schema [Open Geospatial Consortium, 2012b], therefore it uses the same geometric representation according to the SFS. In addition, it allows for the storage of topological relationships through identifiers between geometries. CityGML contains definitions for several feature classes, such as transportation networks, vegetation, water bodies, terrain and city furnitures, in groups of 13 thematic models. It may, also, contain spatio-temporal data through the 'dynamizer' mechanism which has been proposed by Chaturvedi and Kolbe [2016].

3DCityDB is a Relational DataBase Management System (RDBMS) schema definition based on the CityGML data model, which intends to overcome the limitation of the later due to its file format nature. Essentially, it delivers the

[2] http://www.buildingsmart-tech.org/specifications/ifc-overview
[3] https://www.autodesk.com/techpubs/autocad/dxf/reference/
[4] https://www.khronos.org/collada/
[5] https://www.cs.cmu.edu/~mbz/personal/graphics/obj.html
[6] https://www.google.com/earth/
[7] https://developers.google.com/kml/
[8] https://github.com/KhronosGroup/glTF

features of CityGML to the PostGIS[9] and Oracle Spatial[10] databases, which can better fulfil the purpose of permanent storage, maintenance and partial exchange of information through multiple and remote sources.

Although CityGML and 3DCityDB are well established in the field of 3D city models, they do have certain shortcomings. First, CityGML s based on the eXtensible Markup Language (XML) storage mechanism of objects, which is extremely verbose and complicated, as it is difficult to map its basic tree model to the type system of most programming languages. Second, the effort of CityGML and 3DCityDB to anticipate all possible urban types and characteristics in a hierarchic tree makes its data model extremely complex and difficult to understand.

As an alternative to a more lightweight and flexible solution for city models' representation, City JavaScript Object Notation (CityJSON)[11] was lately introduced. It is based on the JavaScript Object Notation (JSON) encoding format, which is more storage efficient, flexible and web oriented. It is, also, easily parsed by high-level weakly typed languages, such as JavaScript or C#, which can lead to more straightforward implementation of higher-level applications, such as 3D GIS programs. CityJSON keeps all city classes and only provides a minimal set of required attributes, while leaving to the author of every specific dataset the flexibility to add more features as needed for the specialised application that it is intended for.

2.3.2 LoDs in City Models

LoD is a well defined concept in the topic of 3D graphics, where it serves the purpose of more efficient rendering when different amounts of details are needed for 3D models [Luebke et al., 2002]. A similar concept has been incorporated to the CityGML specification, where five consecutive LoDs are declared (Figure 2.3), while there are ambiguities regarding their definition. Biljecki et al. [2016a] has proposed a more detailed subdivision of scale through a two level system. LoD0-3 is used to describe individually the horizontal and vertical amount of details represented in the city model (Figure 1.1). Furthermore, Löwner et al. [2016] proposes a more arbitrary definition of the LoD concept, where every user can define the level that meets the specific application.

The incorporation of LoDs in 3D city model is of significant importance, taking into consideration the particularities of the field. First of all, 3D geometry calculations are complex and there is still a lack of mature and robust algorithms to manipulate 3D objects [Zlatanova, 2000]. That makes it extremely difficult to extract geometric shapes of less detail from a higher LoD model. Secondly, applications have very different requirements and there is evidence that the use of higher LoD objects doesn't always produce better results [Biljecki et al.,

[9] `https://postgis.net/`

[10] `http://www.oracle.com/technetwork/database/options/spatialandgraph/overview/index.html`

[11] `http://www.cityjson.org`

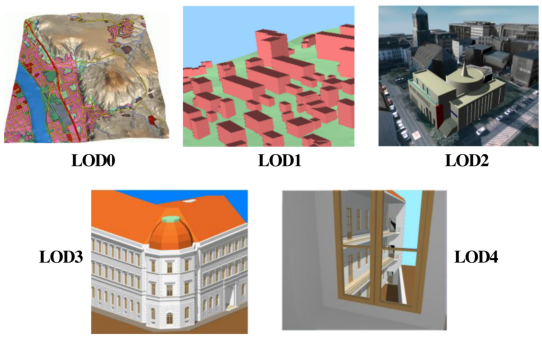


Figure 2.3: The five LoDs as defined in CityGML [Open Geospatial Consortium, 2012a].

2017]. In fact, Biljecki et al. [2016b] concludes that there is a higher probability for errors to be found in datasets with more detail.

2.3.3 Multi-LoD Datasets

Although there has been a consensus about the need of a better LoD implementation in 3D city models, today's datasets are, almost universally, describing only a single LoD of the city they represent. According to Biljecki et al. [2014b], this is due to a lack of consistency, such as redundancy in the acquisition, modelling and storage processes, and to the lack of robust solutions on the subject of 3D generalisation, which could provide more uniform and consistent multi-LoD datasets.

2.3.4 Software Support for 3D City Model Files

In order to extract information from a city model for a specific application many transformations need to be done. Proprietary tools, such as FME[12] and ArcGIS[13], offer limited support for CityGML handling.

Full support for most up-to-date operations of CityGML is provided by CityGML4J[14], an API for the Java programming language. In addition, it can read and write CityJSON files. While it is the most robust library for CityGML file manipulation, it only offers limited options of other GIS formats to import and export data. Java is, also, lacking in libraries that support LCC manipulation, needed for the purpose of this research.

Libcitygml[15] is a C++ library that supports reading of CityGML files. It is well maintained, but it does not support writing of files and it still lacks full support of all features of the CityGML specification.

[12] `https://www.safe.com/fme/fme-desktop/`
[13] `https://www.esri.com/arcgis/about-arcgis`
[14] `https://www.3dcitydb.org/3dcitydb/citygml4j/`
[15] `https://github.com/jklimke/libcitygml`

Geospatial Data Abstraction Library (GDAL)[16] is an extremely versatile piece of software, given that it provides support to read and write a wide range of file formats or spatial databases. But its original GML driver was extremely limited and it can only load files with a very simple structure, hence not CityGML files. GDAL has recently introduced a new driver with full support for GML files, the GML driven by Application Schemas (GMLAS). This new driver is able to process files with complex schema definitions, such as CityGML files.

2.4 nD GIS Modelling

Before we examine the use of nD on GIS applications, it is important to clarify what the term *dimension* stands for. Many times terms such as *4D* and *5D* are used without containing the true meaning of the equivalent nD geometrical spaces, but instead they describe the fact that spatio-temporal data are stored and processed. This originates from the use of the term *dimension* as it is being introduced in databases and, more specifically, in Online Analytical Processing (OLAP) systems [Chaudhuri and Dayal, 1997]. During this research, though, I am only focusing on the implementation of actual nD geometrical spaces in order to store data related to scale and time additionally to the basic spatial dimensions.

There is a high correlation between 2D/3D spatial data and the notions of time and scale in the GIS field. For instance, in his study Mandelbrot [1967] explains the importance of scale when measuring spatial data. In fact, scale is an important concept when we are working with 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 GISs. 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 nD spaces in order to incorporate both scale and time additionally to the basic three spatial dimensions. In their study van Oosterom and Stoter [2010] proposed a model where 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 Meijers, 2013]. 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

[16] `https://www.gdal.org`

[Arroyo Ohori et al., 2015a,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 spaces, 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., 2015c].

2.5 Topology

Topology is a branch of mathematics which studies the continuity of functions between closed sets. In some form, it can be considered as an *abstract geometry*, meaning that it focuses on attributes of spaces that are independent of their geometric shape. For example, from a topological point of view a cube and a sphere are homeomorphic shapes, meaning that we can bend one to make it look like the other. In addition, size is of no concern in topology.

One of the first topological problems is considered to be defined by Euler [1736], when he studied the problem of the Seven Bridges of Königsberg (Figure 2.4). He described the layout of the city, a mainland that was separated from two big islands by the Pregel River, all connected through seven bridges. They were placed in such an order that it would be impossible to walk through all them once and only once. This problem only focuses on the graph representation of the city's structure, without being affected by attributes related to the shape and size of it. Later on, Euler also figured out what is considered probably the first topological theorem, the polyhedron formula ($V - E - F = 2$) that describes the relationship between the number of vertices, edges and faces of a polyhedron.

Topology's basic foundation was laid by Georg Cantor when he established set theory and started applying it on points of Euclidean spaces. Later on, those ideas were evolved with the works of Fréchet [1906], Hausdorff [1914] and Kuratowski [1922] to become what we know today as topological space.

Topology is an important tool on GIS science, as it provides powerful tools to study the relationships between geometries. It offers a set of rules which can be used to subdivide spaces in a more formal way. This offers insights and solutions to problems that are difficult or impossible to solve through basic geometrical tools. For instance, a topology can be built based on a cadastral map in order to provide meaningful information regarding neighbouring relations, the validity of parcels or the existence of empty spaces (spaces without ownerships). Hence, topologies in GIS can assist with data integrity and consistency.

GIS topological structures are a combination of the mathematical concept embedded with the geometrical information that defines the objects' shapes. They can be described as enhanced boundary representation, meaning that they are non-geometric data structures which are associated with geometrical

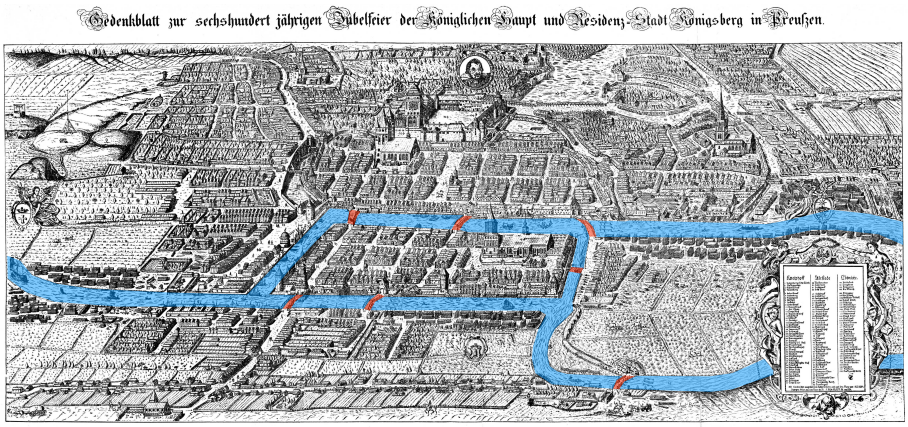


Figure 2.4: The problem of Seven Bridges of Königsberg, as described by [Euler, 1736]. Based on an image from a 1613 engraving by Joachim Bering.

data that allow them to carry both spatial and non-spatial information.

In traditional GISs, topological structures are created by applying calculations on the original geometric data, which are normally described by the SFS. But, as mentioned before, GIS has only recently adopted the use of 3D data. Therefore, most related work on the application of topological data structures is focused on the 2D space.

2.6 Topological Data Structures

A GIS topological structure is a subdivision of an nD space into partitions. These i-dimensional partitions ($\forall i : 0 \leq i \leq n$) are called *$i$-cells*. Therefore, a 0-cell is a vertex, a 1-cell is an edge, a 2-cell is a facet, a 3-cell is a volume etc. Every i-cell is bounded by $(i-1)$-cells (for $0 < i \leq n$), which defines the *incidence* relationship between the bounded and bounding cells. Additionally, every i-cell can have neighbouring cells of the same dimension when the two cells share a common incident $(i-1)$-cell, which defines the *adjacency* relationship. For example, two surfaces (2-cells) are adjacent when they share a common edge (1-cell) across their boundaries.

The above description defines the basic data model of every GIS topology, from which it becomes clear that the representation of objects can be achieved through their boundaries. This means that starting from vertices (0-cells), $\forall i : 0 \leq i \leq n$, all objects of $(i+1)$-dimension can be built through their bounding i-cells. Practically, we can define edges (1-cells) from vertices (0-cells), facets (2-cells) from edges (1-cells) etc.

It is important to underline the independence of dimensionality between the topological primitives and the ambient space. A topological structure can describe the relationships between geometries of higher dimensions. For instance,

a 2D topological structure can represent cells of up to 2 dimensions (vertices, edges and facets) on a 3D space, so that every vertex is assigned three coordinates (x, y, z).

The following sections describe the topological data structures that are used in GIS. Section 2.6.1 provides a review of the ordered topological structures and lists the most popular implementations of them on GIS software. Section 2.6.2 introduces the Combinatorial Map (C-Map) data structure and the concept of a Linear Cell Complex (LCC). Finally, 2.6.3 reviews the usage of LCCs for 3D city model representations and applications.

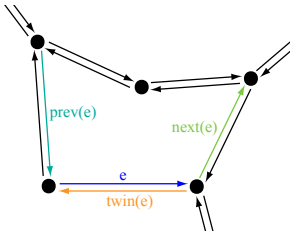
2.6.1 Ordered Topological Structures

The most commonly used data structure for representing topological relationships is the *Doubly Connected Edge List (DCEL)* or *half-edge* data structure (Figure 2.5), originally proposed by Muller and Preparata [1978]. In this data structure, the space is subdivided through half-edges, meaning that each edge is split into two oriented units with opposite direction. A half-edge represents the part of an edge that is incident to the face on the left side of the edge. It contains information regarding the next, previous and opposite (or twin) half-edge, which is enough data in order to access all elements of the whole DCEL starting from any half-edge element. In addition, every half-edge keeps a pointer to the origin point coordinates, therefore geometrical information are stored in the data structure.

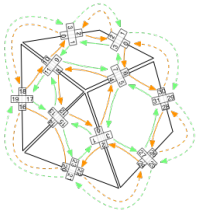
Although it was originally proposed as a way to represent 3D polyhedral objects, the half-edge data structure has been mostly successful in 2D. It is implemented in most GIS applications, such as ArcGIS[17], QGIS[18] and AutoCAD Map 3D[19], for representing 2D GIS topologies. It has been proven an extremely efficient and robust data structure to implement, as it uses limited memory space and there are robust algorithms in order to navigate and alter the structure with low computational complexity.

Another similar data structure that has been proved popular, is the *quad-edge* data structure invented by Guibas and Stolfi [1985]. It shares a common idea with DCELs as every edge is represented by two opposite-direction elements, but in this case both the main graph and its dual are represented. Practically, every quad-edge not only keeps pointers to the next and previous quad-edges of its incident polygon, but it also keeps another pair for the opposite polygon. This makes adjacency queries more efficient and allows altering operations on the data structure to be applied easily, but the data structure requires more storage.

[17] `https://www.esri.com/arcgis/about-arcgis`
[18] `http://www.qgis.org`
[19] `https://www.autodesk.com/products/autocad-map-3d/overview`



(a) A DCEL



(b) A quad-edge graph

Figure 2.5: Examples of known 2D/3D topological data structures

2.6.2 Linear Cell Complexes

While the data structures defined in Section 2.6.1 have been proven robust and are implemented and used in most modern GIS applications, they do remain somewhat limited with regard to their scalability across dimensions. Most of them have been designed for 2D or 3D spaces and they are mostly applied and used only for 2D data, while it is impossible to use them in higher dimensions.

A generalized model independent of dimensions was originally proposed by Edmonds [1960], which later evolved to the data structure known as *C-Maps* defined by Vince [1983]. A C-Map is a data structure that can represent a partition of an n-dimensional space. A C-Map is composed by elements called *darts*, which can be considered as the 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 belongs to a neighbouring i-cell. Those links are called β_i, where $0 \leq i \leq n$, and every dart contains one β_i $\forall i \in \{1, \ldots, n\}$. A β_i can be better understood like 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).

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

C-Maps only store information regarding incidence and adjacency between the cells that it describes, but lack any information regarding their geometry, such as shape and size. In other words, they describe the primitives of the 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 linear geometrical shapes through a representation of their boundaries. As stated before, the coordinates can be of d-dimensions (compared to the nD C-Map), where $d \geq n$.

This enhanced data structure that is primarily a topology but encapsulates

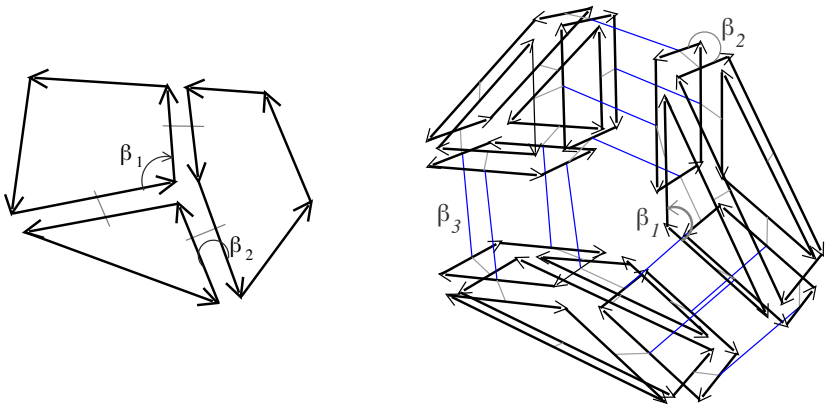


Figure 2.6: Examples of C-Maps [Damiand and Teillaud, 2014].

linear geometry as well, is called a LCC. The benefits of a LCC is that it can be easily expanded to as many dimensions as needed, because the underlying structure is not relying on geometrical dimensions. Nevertheless, geometry is still present through the coordinates assigned as attributes on 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, an 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.

A LCC can be considered as similar to a DCEL, as a half-edge can be conceived as a dart on a 2D/3D C-Map where β_1 is the pointer to the next half-edge and β_2 the twin.

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.

Storage

Feng et al. [2013] studied the concept of storing LCCs by proposing a solution for compact storage of C-Maps for mesh data, but his solution is limited to 3D spaces. Damiand and Teillaud [2014] describes an implementation of C-Maps storage and manipulation which is dimension independent and, therefore, can serve the purpose of storing 4D LCCs with attributes. This study led to the

implementation of the C-Map[20] and LCC[21] packages in The Computational Geometry Algorithms Library (CGAL).

2.6.3 Linear Cell Complexes on 3D City Models

As described in Section 2.3.1, a typical representation of a 3D city model is normally through the SFS, but recent studies have been conducted on how to utilise LCCs in order to describe 3D city objects. Horna et al. [2015] explore such approaches in the use of modelling and simulation studies, where he describes the advantages of topological representations for 3D buildings. He also proposes some basic algorithms for the manipulation of such data when stored in 3D LCCs, as well as traversals that can be applied to building objects in order to improve calculations. An example of how to exploit topological characteristics is during the propagation of rays when calculating their propagation, when adjacency information can be used in order to improve the performance of a simulation (Figure 2.6).

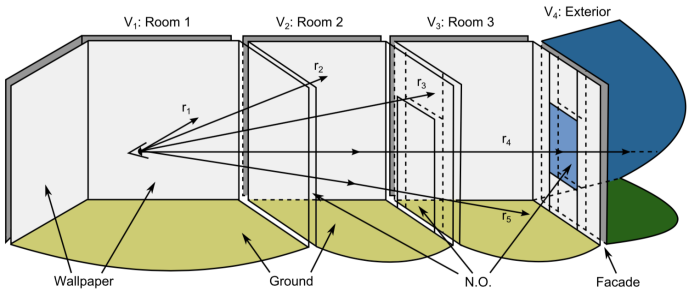


Figure 2.7: An example of ray-casting calculation that uses adjacency information on a building stored in a topological structure [Horna et al., 2015].

Diakite et al. [2015] have proposed Enriched Building Model - Linear Cell Complex (EBM-LCC), a specific implementation of an LCC where attributes are used in order to describe 3D buildings which derive from BIM objects and LoD2 CityGML datasets. He also develops a method for creating EBM-LCCs from the topological reconstruction of buildings in order to automatically extract different LoDs from one main city object [Diakité et al., 2014]. The method is starting 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 on a city model. But, instead, semantics are not preserved during the proposed process.

[20] `https://doc.cgal.org/latest/Combinatorial_map/`
[21] `https://doc.cgal.org/latest/Linear_cell_complex/`

2.7 Comparing City Models

While 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 city models. 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 change that has been found between the two datasets.

A true comparison between city models in 3D has been studied by Gorszczyk et al. [2016]. He has exploited the characteristics of the LCC data structure to identify common features between 3D city objects on the dataset in order to highlight topological and semantics differences between them (Figure 2.8). 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, such as those examined in this study, it would not work practically for datasets originating from different sources or for datasets with heavy differences between objects, such as different LoDs of the same city model.

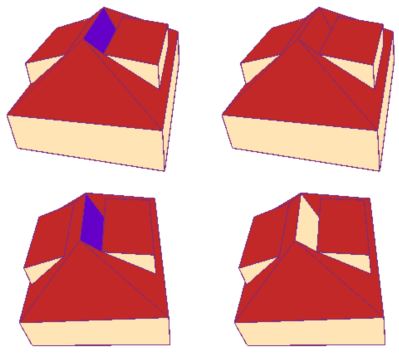


Figure 2.8: A difference of semantics between two 3D city models, found with the proposed methodology by Gorszczyk et al. [2016].

2.8 Link representation of 3D Objects across Scale

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 common features between different 3D objects. A typical solution to such a problem is by the use of identifiers between common objects, such as in the case of LoD implementation on the CityGML specification [Open Geospatial Consortium, 2012a].

Arroyo Ohori et al. [2015a] has proposed some solutions to the problem while exploring possibilities regarding the modelling of a multiple LoDs in 4D. He proposes four approaches on the way common features are linked across the different versions (Figure 2.9). Each of the four alternatives comes with its own benefits, but introduces certain limitations when applied. For instance, *simple linking* provides a very straight-forward approach as only features that exist in two linked versions are connected to each other. While this allows for a more straight-forward representation of the relationship between objects, it does not provide a continuous transition across the LoD axis.

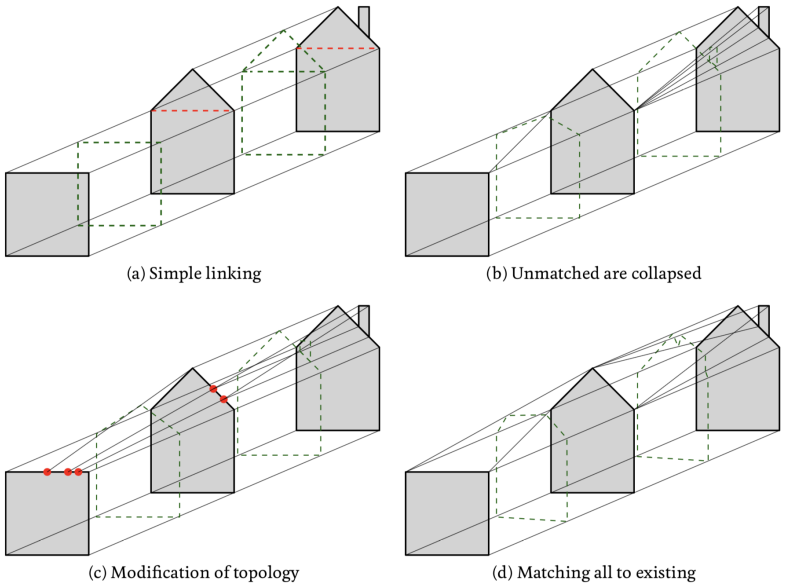


Figure 2.9: Four proposed solutions for the encoding of correspondences between common features across different objects [Arroyo Ohori et al., 2015a].

2.9 Graph Databases

While simple computer file and RDBMS has served the purposes of persistent storage and exchanging of data for decades, modern applications are introducing

huge challenges that can not be easily served by them [Corbellini et al., 2017]. In order to solve those problems, such as the storage and manipulation of big-data, a new generation of database systems has been started to gain the attention on computers, given the name Not-only SQL (NoSQL). A specific type of NoSQL databases is *graph databases*, which uses a graph structure to store data as nodes and their links as edges [Lal, 2015].

Chapter 3

Proposed Research

3.1 Problem Definition

Scale is a very important factor in modelling and can seriously alter the outcome of an application [Mandelbrot, 1967]. The concept of LoD has been introduced for many years in 3D graphics [Luebke et al., 2002] and later incorporated and enhanced in 3D city models, in order to reflect scale's influence on 3D data representation. This is mainly due to the fact that it has been proven that different applications need various amounts of detail and it is not always the case that more detail can provide better results [Biljecki et al., 2017]. Unfortunately, the use of the LoD mechanism, as it is being used today, is prone to introducing inconsistencies in datasets and difficulties on the maintenance of those datasets [Aringer and Roschlaub, 2014].

van Oosterom and Stoter [2010] has proposed a solution to this problem by applying time and scale to 2D and 3D GIS datasets as additional dimensions in nD spaces. Arroyo Ohori et al. [2015b] has investigated this idea and proposed C-Maps as the best likely candidate for the representation of LoD as the fourth dimension.

While studies of this subject have remained on either a conceptual level or as a proof of concept, there are still questions to be answered in order to develop a complete framework for storing and manipulating multi-LoD city models in 4D for real world data. This involves the process of matching of common features between 3D objects of different datasets (Section 2.7) and the evaluation of alternative methods on linking common features between objects as identified by the previous process (Section 2.8).

3.2 Research Objectives

Based on the problem that has been defined before, my main research objective is to:

Develop a representation of multiple 3D city models of the same city objects and their linked correspondences with semantics in 4D space.

This objective can be further defined through the following research questions:

(i) What kind of representation can we use in order to store and manipulate efficiently objects in 4D space?

(ii) What is the best way to encode city objects and semantics in such a data structure?

(iii) How can we identify common features between objects of different LoDs and how can they be represented in a continuous way in 4D?

(iv) How can we construct a complete 4D city model containing all the geometric and semantic information of its LoDs?

(v) What is the most efficient way for permanent storage and exchange of 3D and 4D city models?

3.3 Methodology

In order to solve those questions, I have defined four sub-objectives that I will undertake during my research. In this chapter, I will describe them as challenges of my research that can be undertaken through individual processes.

In Section 3.3.1 I focus on the study of research questions (i) and (ii). In Sections 3.3.2, 3.3.3 and 3.3.4 I propose a methodology that will answer the research questions (iii), (iv) and (v), respectively. In Sections 3.3.5 and 3.3.6 I describe the research that I am planning to conduct in collaboration with Anna Labetski (PhD Candidate) and Ken Arroyo Ohori (Post-doc researcher) as part of our common involvement in the UMnD project.

3.3.1 Topological Reconstruction of 3D City Models

As discussed in Section 2.4, it is evident that topological data structures are more capable regarding their ability to represent nD objects. LCCs based on C-Maps are a very efficient and powerful solution to storing data in nD spaces [Arroyo Ohori et al., 2015c]. Meanwhile, the majority of available 3D city models worldwide are stored and exchanged in geometrical formats and, mostly, according to the SFS as it is incorporated in CityGML.

The first part of my research is to investigate the best approach regarding the conversion of geometry from the SFS model to the LCC data structure.

Converting SFS geometry to LCC

Diakité et al. [2014]'s method can be used to convert a building to an LCC. This work needs to be extended in order to achieve the topological reconstruction of

all possible classes found on a CityGML dataset. In addition, this approach has to be further refined in order to take into consideration the semantics of a city model.

My research will focus on the best possible method to topologically reconstruct a city model in a lossless way, so that semantics are maintained. Semantics incorporation of the topological reconstruction greatly complicates the procedure, as the retaining of information of objects also implies the need to preserve the original city model structure as much as possible. For example, in a typical conversion of a geometrical dataset through an incremental construction, two originally individual city objects can merge into one 3-cell, as soon as they are topologically valid. This, of course, would eliminate the information of one of the two original objects, making the conversion lossy.

I will also work on the effective storing of semantic information in the resulting topology. My intention is to use the functionality of i-attributes, as described by Damiand and Teillaud [2014], in order to assign the information to the respective i-cells. City object's information, such as the type of the object or attributes of it in the CityGML data, can be represented by 3-attributes, while 2-attributes can be used for information related to individual surfaces of an object.

Initially, I will work on the topological reconstruction of buildings to 3-cells. When this is accomplished in time, I will work on the conversion of roads to 2-cells and, then, representation of linear network to 1-cells.

3.3.2 Linking Common Features Between Datasets

A major challenge in undertaking my research is the lack of multi-LoD datasets with correspondences between common features of different versions of the same object. For this reason, there is a need to develop a process for the comparison of datasets in order to find common features that will be linked together in the final 4D objects (polychora).

LCCs Matching

Comparing city models in order to find common objects or features between two datasets is an essential operation for building a multi-LoD dataset. For this reason, a method to apply matching on existing datasets is important to be established.

I intend to generalise Gorszczyk et al. [2016]'s methodology on the subject. His study is undertaken in three steps: objects detection, edge matching and comparison of matches. Those steps can be considered as abstract components of a more generic process, in order to be able to adapt it on the particular needs of the investigated pair of datasets. I will implement variations of those processes on different real world datasets.

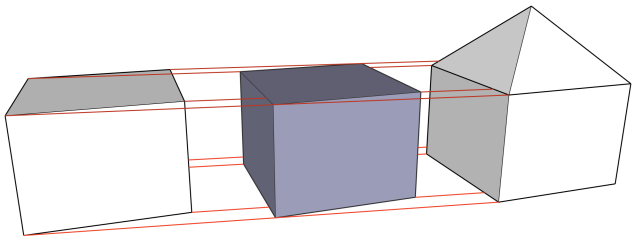
The result of this step must be to find correspondences between the pairs of city models. The correspondences will be realised as pairs of darts that describe the same features, for instance the same facade of a building.

Linking Correspondences in 4D

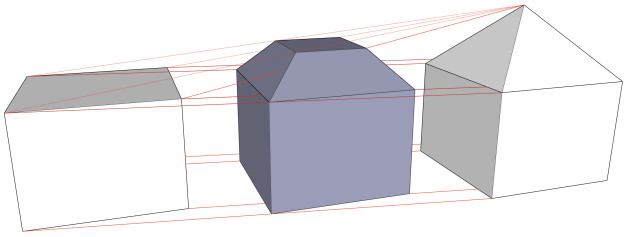
Arroyo Ohori et al. [2015b] have proposed several methods for linking correspondences between different LoDs of the same object in 4D. I intend to apply those methods on real world datasets, based on the outcome of LCCs matching as described in Subsection 3.3.2.

My research will focus on the details that need to be investigated regarding the applications of the proposed methods. For instance, when the "modification of topology" method is applied there are alterations that have to be done on the geometry and topology of the two 3D objects. Such operations raise practical questions, such as the investigation of the methodology to determine the new points to be introduced in the model.

In addition, I intend to work towards the systematic evaluation of the proposed methods of linking, in order to identify their benefits and caveats when applied in practice. This can be further realised after the extraction of intermediate objects has been established (related to Subsection 3.3.6), as the resulted 3D objects from this operations would be greatly affected by the implemented method of linking (Figure 3.1)



(a) Simple linking



(b) Matching existing cells

Figure 3.1: A demonstration of how extracted 3D objects from one 4D object is affected by the linking method [Arroyo Ohori et al., 2015a].

3.3.3 Merging Linked 3D Objects in 4D City Objects

This part of my research is about the construction of the final 4D object. I will work on the base of Arroyo Ohori et al. [2015a]'s work, with the aim to produce a final "prismatic" polychoron where the two original 3D object are the 4D equivalent of a "base" and "top" face of a 3D prism. We denote the fourth dimension as l and every object's point is assigned the LoD number as the value of l.

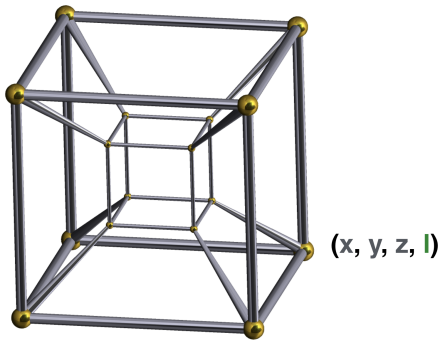


Figure 3.2: Simple example of a 4D object with (x, y, z, l) coordinates, projected to R^3.

4D LCC Construction

The construction of the final 4-cell that contains both 3D objects and its correspondences is a challenging operation. In order to construct a complete polychoron, a 3-cell has to be created for every pair of faces between the two objects. I will focus on the designation of those faces through the corresponding links and the methodology to define and construct those intermediate 3-cells.

My research will focus, also, on other practical aspects that need to be tackled during this construction. In order to construct the final 4D object, intermediate 3-cells need to be created in order to link the two objects of lower and higher LoD. This process involves some alterations to the original objects. For instance, in order to sew common 2-cells between two 3-cells, a reverse of orientation might be needed in order for the final intermediate 3-cell to be a valid volume.

3.3.4 Storing and Exchanging 4D City Models

We can define the problem of persistent storage for archiving and exchanging purposes as a subject related to two individual research fields. Mainly, city models are today studied as 3D data structures where research has concluded on the creation and continuous development of several specifications (Section

2.3.1). But it is evident that their geometrical representation (mostly based on the SFS or 3D graphics-oriented data structures) fails to efficiently describe nD objects, for which a LCC storage file format is needed.

Persistent Storage

In my research, I will work on top of Damiand and Teillaud [2014]'s work in order to implement LCCs stored as C-Maps in the CGAL library. This allows for some basic persistent storage option, as this implementation offers the ability to read and write the respective data in a file.

My research will also try to evaluate potential incorporation of the C-Map data model in existing popular 3D city models formats. First, I will focus on the development of an implementation of the resulting 3D topological model, as produced by the output of Section 3.3.1, in the CityGML and CityJSON specifications. I will, then, try to evaluate the benefits of using this storage model either in order to replace or to enhance the original geometrical representation.

Graph Databases for LCC Storage

Given that the C-Map data structure is essentially a graph structure as well, I indent to investigate the possibility of using graph databases and, more specifically, Neo4J[1] (which is an open source graph database implementation) as a way to store and query 4D city model data in the persistent layer.

More specifically, I intend to implement darts and attributes as nodes and the β_i pointers as relationships. Then, I will do a systematic evaluation of the performance of this representation, given that graph databases are considered extremely efficient for querying big and complex datasets, where traditional city model specifications, such as CityGML and CityJSON are lacking.

3.3.5 Creating 4D City Objects by Extrusion and Generalisation

After the creation of 4D objects has been achieved from heterogeneous datasets, as described in Section 3.3.5, I will work with the outcome of Anna Labetski's research on 3D generalisation, which is also part of the UMnD project.

My intention is to firstly apply an extrusion of a higher LoD 3D object to 4D, based on Arroyo Ohori et al. [2015d] methodology for arbitrary nD extrusion of LCCs. Then I will apply the generalisation techniques, proposed by Labetski, in the "base" 3D object of the prismatic polychoron in order to achieve a multi-LoD 4D object.

[1] `https://neo4j.com/`

3.3.6 Visualising Intermediate 3D Objects Extracted from 4D

Ken Arroyo Ohori will work as a post-doc researcher on the concept of the extraction of 3D objects from the 4D data structure that I will develop. As mentioned during Section 3.3.2, the method through which the 3D objects are linked greatly affects the outcome of this process.

My intention is to work closely with Arroyo Ohori on the evaluation of the different methods that can be applied. I will, also, work on the practical aspect of the visualisation, such as the triangulation of 3D polygons, that I have already undertaken through my experience so far.

Triangulation of 3D polygons

As described in Section 2.2, in order to visualise 3D data in a modern computer, they need to be provided to the graphics hardware as a 2D simplicial complex (triangles). The triangles are provided as lists of points and indexes that provide the necessary information for the mesh to be rendered by the graphics card. Given that 3D spatial data are, mostly, described according to the SFS model, which describes polygons, there is a need to triangulate them. While 3D triangulation algorithms exist, they normally break the structure of the original polygons, providing only triangulated convex volumes according to the provided points, as a result. In order to achieve a triangulation of polygons in a 3D space while keeping their original structure, I intend to implement a practical solution for applying 2D Constrained Delaunay Triangulation (CDT) [Chew, 1989] on 3D polygons.

This can be achieved in four steps: (a) the best-fitting plane for the polygon is calculated; (b) the points of the polygon are projected from 3D space to a local 2-axis coordinate system defined on the plane; (c) the projected points, which form a 2D polygon, are triangulated according to the CDT so the list of indexes is calculated; and (d) the global position of the projected points in 3D space is calculated and those 3D coordinates, along with the indexes produced before, are provided for rendering.

Chapter 4

Preliminary Results and Conclusions

As part of my first year's research I have had the opportunity to investigate practical aspects of 3D city models and to experiment with the initial steps of my proposed methodology. Here are the preliminary results of my research.

4.1 A Framework for Creating 4D City Models

This part of my research is about the conceptual study of the necessary steps that needs to be taken in order to produce a 4D city model from existing 3D datasets. My proposal is a combination of the work of Gorszczyk et al. [2016], on the comparison of two 3D city models through LCCs, and Arroyo Ohori et al. [2015a], on the methodology to link correspondences between different LoDs of the same city object.

In order to combine existing datasets we need to define a clear framework, which defines the individual goals that need to be achieved through the process of the 4D city model creation. For this purpose, I propose the following pipeline:

In step (I), the existing 3D city models are topologically reconstructed in order to be converted as LCCs. This step is described further on detail in Section 4.2.

Steps (II), (III) and (IV) are based on the approach of Gorszczyk et al. [2016]. Initially, objects have to be associated, if possible, as volumes according to some criteria. The criteria could be loosen or tightened according to the characteristics of the provided datasets. For instance, when the two initial city models share many similarities, like LoDs originating from the same raw data, then the association can be defined according to sharing common surfaces. In cases where the two datasets are extremely different, then more abstractive criteria can be applied, e.g. the ratio between the intersected volume of two objects and their original volume.

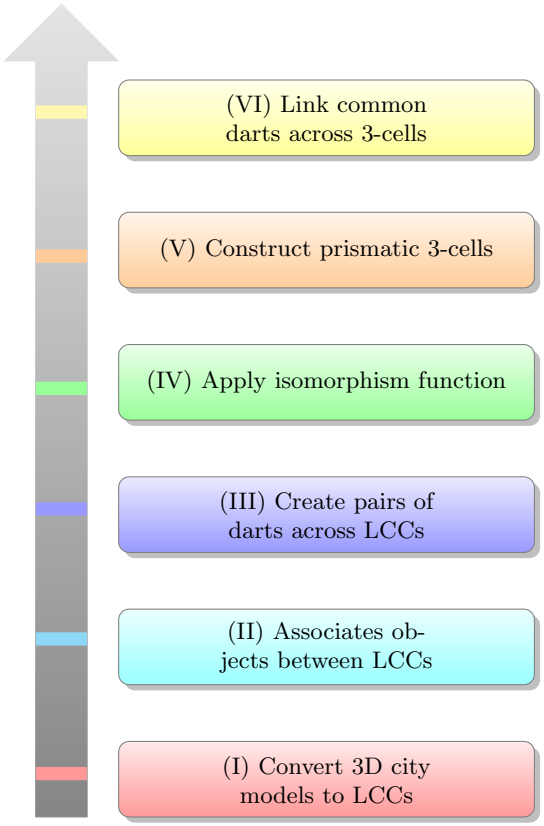


Figure 4.1: The pipeline with the steps that can produce the 4D city objects.

After the association association of objects is achieved, steps (III) and (IV) need to be undertaken in order to gain a list with correspondences of darts. The list is produced by first creating candidate pairs of darts, according to their geometry, and then applying the isomorphism function in order to filter the true common darts. Again, the geometric criteria for those steps can be set as loose or as tight as it is needed, according to the specific features of the datasets. If the two models share parts of the same geometry, then the criteria for assigning darts could be if they share the same source and destination coordinates. But in cases where the models are from different sources, a function that would find the closest darts between the two objects could be used instead.

Step (V) is where the prismatic 3-cells are created. This means that the initial 3D objects must be assigned a fourth coordinate, l, according to their LoD, in order to fit in 4D space. During this stage, knowing the correspondences between the two prismatic objects, there might be a need for some changes to be made to the objects, in case the *Modification of topology* method according to Arroyo Ohori et al. [2015a] (Figure 2.9) is going to be applied.

Finally, step (VI) will be undertaken to sew the two 3-cells in one 4D object. For this purpose, an intermediate 3-cell must be created between every common 2-cell across the two objects. This involves some modifications that might have to be taken into account, such as the orientation of the darts. As described in Section 2.6.2, LCCs are ordered topological structures which means that orientation of the darts define characteristics of the i-cells they describe. This means that, in order for the final 4D object to be valid there might have to be a flip of the orientation of the darts on one of the two prismatic objects.

4.2 Converting 3D City Models to Linear Cell Complexes

I have began my research by working on the topological reconstruction of existing 3D City Models, as described in Section 3.3.1. Most datasets are available in the form of CityGML files, therefore my work focused on the conversion of a city model in CityGML to a LCC. Arroyo Ohori et al. [2014] provides an abstract approach for n-dimensional spaces, but my research focuses on more practical aspects of converting 3D city models while keeping their original structure and semantics.

In order to convert a CityGML model to an equivalent LCC, it is required that darts are created on a C-Map while keeping track of those that are 2-free and 3-free in order to sew them respectively with newly created darts that are adjacent to them. I have implemented this procedure in incremental steps described by individual algorithms. Hence, every i-dimensional object is processed by the respective algorithm.

The conversion starts by calling the main body for a given city model. This algorithm keeps the global data through the process (the resulting LCC and the two indexes for 2-free and 3-free darts), while it iterates through all root city object nodes of the original city model. For every root city object the `ReadCityObject` algorithm is run, which appends the LCC and the indexes according to the provided city object.

`ReadCityObject` iterates through all immediate geometries of the city object and calls the `ReadGeometry` subroutine for every geometry. It also iterates through all child city objects calling, recursively, itself in order to process them. After a polygon has been converted to the equivalent 2-cell, then the algorithm will attempt to 3-sew any 2-cells that share the same geometry.

`ReadGeometry` converts a geometry (polygon) to a 2-cell in the destination LCC. It loops through the edges that bound the polygon and calls `GetEdge` in order to construct and sew together the respective 1-cells that are needed in order to describe the 2-cell. After every edge is created, the algorithm will find 2-free darts that can be 2-sewed to the new ones and will apply the 2-sew operation.

The `GetEdge` algorithm creates an 1-cell based on two end points. An 1-cell needs two darts in order to be described, therefore `GetVertex` is used in order

to either find an 1-free dart or to create a new one dart with the coordinates that are on every point.

The main algorithm has two variations: one where the index of 2-free darts is cleared after every root city object is finished and one where the 2-free darts remain available through the whole process. The first approach can be described as *semantically-oriented*, since it forces 2 polygons to be sewed only in case they belong of the same root city object, and the second approach as *geometrically-oriented*, as it will allow for previously individual volumes to become one 3-cell in case they share a common edge across one of their bounded polygons.

The described methodology has been implemented in a computer program using the C++ programming language. For the purpose of CityGML reading, I used the *libcitygml*[1] library. For the basic LCC representation I used the CGAL LCC package[2]. Visualisation was accomplished through a modified version of the LCC demo provided with the CGAL software package, which is based on the Qt5 graphical user interface and the *libqglviewer*[3] component for viewing 3D graphics.

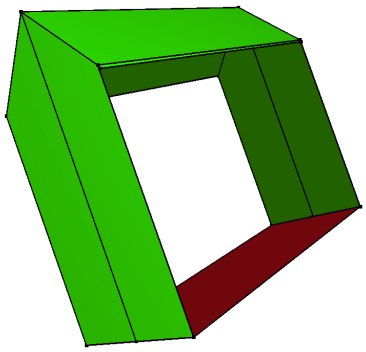
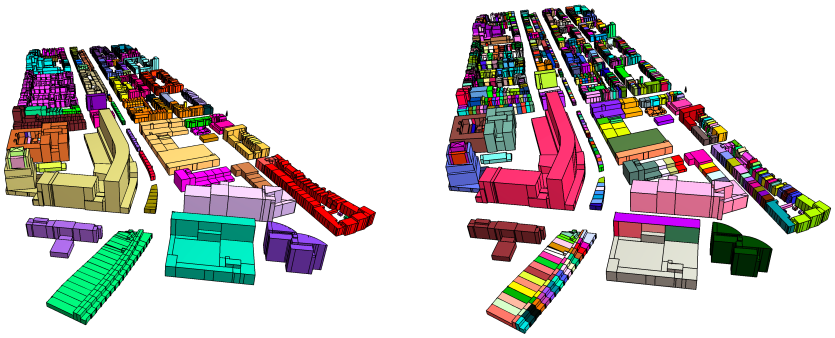


Figure 4.2: An individual house from the Agniesebuurt dataset, visualised after it is topologically reconstructed to an LCC. Every individual 3-cell is distinguished by different colours.

The implemented software program was, then, run against the city model of Agniesebuurt, a neighbourhood of Rotterdam offered as open dataset by the municipality[4]. Initially, I applied the reconstruction for an individual building (Figure 4.2). The resulting LCC underlines some specific features of this dataset. First, the walls between neighbouring buildings are missing. Second, the geometry is topologically invalid, as the building has resulted in 2 volumes (3-cell).

[1] `https://github.com/jklimke/libcitygml`
[2] `http://doc.cgal.org/latest/Linear_cell_complex/index.html`
[3] `http://libqglviewer.com/`
[4] `https://www.rotterdam.nl/werken-leren/3d/`



(a) LCC A: Geometrically-oriented reconstruction

(b) LCC B: Semantically-oriented reconstruction

Figure 4.3: Examples of the two different approaches for the topological reconstruction of an example city model

		Number of				
Model	Size (MB)	Darts	0-cells	1-cells	2-cells	3-cells
LCC A	4.4	59188	22280	36710	12876	469
LCC B	4.7	59188	25771	38810	12876	1148

Table 4.1: The values that represent the size of the resulting linear cell complexes of Agniesebuurt (Rotterdam) dataset

Finally, the two variations of the reconstruction were applied to the complete city model. Figure 4.3 demonstrates the difference between the two resulting LCCs, the geometrically-oriented (LCC A) and semantically-oriented (LCC B). Due to the fact that neighbouring walls are missing, the geometrically-oriented reconstruction sewed together all adjacent buildings in one volume (3-cell), except for non topologically-valid objects. The semantically-oriented reconstruction enforced the separation of individual buildings, which resembles better the original structure of the city model. This is also evident in Table 4.1, which present statistical aspects of the resulting LCCs.

4.3 Evaluating Software Support for 3D City Models

In order to achieve the proposed methodology, I have investigated existing 3D city models by developing and using computer software against available open datasets. Through this experience, I have gained a better understanding of practical aspects related to the manipulation of geometry and semantics on city

models.

4.3.1 Manipulation of CityGML by GIS software

Semantics are an important component of 3D city models. They are incorporated in CityGML through the schema definition that is described by the specification. Loading and handling CityGML datasets in a computer program, though, is not a trivial process, as described in Section 2.3.4. There are few GIS software applications and libraries that feature reading and saving CityGML files.

GDAL's lately introduced GMLAS driver, which is capable of reading CityGML, provides an opportunity for the manipulation of 3D city models by its set of tools and all software that are utilising it. This, essentially, means potential support of CityGML by all well-known open-source applications, such as QGIS[5] and GRASS[6], as long as they can represent and process 3D geometries.

Initially, the GMLAS driver had trouble to parse CityGML files. I have worked closely with GDAL developers in order to test the driver against several CityGML datasets and provide useful information in order to fix issues related to city models' manipulation by the library. I have, also, provided some guide and examples[7] on how to use GDAL with GMLAS in order to convert CityGML datasets in other formats, like ESRI Shapefile and GeoJSON, which makes it possible to process the data in traditional GIS application, such as QGIS.

4.3.2 3D geometry on GIS applications

As described in Section 2.3.4, there is limited support for the CityGML and CityJSON city models' formats. This is especially true for the case of 3D visualisation.

Proprietary tools, such as FME, ArcGIS, TerrainView[8] and FZK Viewer[9], provide visualisation of the 3D geometry of a city model and can read CityGML files, but they are closed source and, mostly, only support Windows. The only open source alternative to them is Azul[10] which supports both CityGML and CityJSON, but is exclusive to Mac computers.

In order to fill this gap of open source multi-platform software with the ability to manipulate city model data in a GIS application, I have worked on the support of 3D geometrical data on QGIS. There has been recent development for the implementation of 3D visualisation in QGIS, but it only worked for 2.5D data and it was unable to process vertical objects (polygons with more than one points of the same X and Y coordinates). I have implemented a more robust solution on the subject, by applying the methodology described on Section 3.3.6.

[5] https://www.qgis.org

[6] https://grass.osgeo.org/

[7] https://3d.bk.tudelft.nl/svitalis/citygml/gdal/2017/07/24/messing-around-with-citygml-on-gdal-2.2.html

[8] http://www.viewtec.net/

[9] https://www.iai.kit.edu/

[10] https://github.com/tudelft3d/azul

For step (a) I have implemented the Newell's method (described in Section 2.1.1) in order to calculate the normal ($\vec{n}$) of a 3D polygon based on its point. Given that the normal provides the (A, B, C) coefficients of the plane's equation. Then D can be calculated by (2.8).

For step (b), a local 2-axis coordinate system must be declared on the plane. This can by defined two arbitrary vectors on the plane, as long as they are perpendicular. For my implementation, I take the vector between the "center of gravity" ($\vec{p}$) and the first point of the polygon ($\vec{p_1}$) in order to define one axis, so:

$$\vec{x} = \vec{p_1} - \vec{p} \tag{4.1}$$

Then, I calculate the other axis as the vector that is perpendicular to this vector and the plane's normal: so:

$$\vec{y} = \vec{x} \times \vec{n} \tag{4.2}$$

Given the two vectors, the local 2D coordinates of any 3D point of the polygon projected to the plane can be computed as the dot product of the vector from the "center of gravity" to this point with the respective axis vector. Hence, for any point $\vec{p_i}$ the coordinates are:

$$x_{local} = (\vec{p_i} - \vec{p}) \cdot \vec{x} \tag{4.3}$$

$$y_{local} = (\vec{p_i} - \vec{p}) \cdot \vec{y} \tag{4.4}$$

Step (c) was already implemented in QGIS, therefore I do not further describe this here.

Step (d), then, is also quite straightforward. The 3D coordinates of the local points can be calculated through:

$$\vec{p_{new}} = x_{local}\vec{x} + y_{local}\vec{y} \tag{4.5}$$

The above methodology has been implemented with C++ in the QGIS source code and a pull request[11] has been submitted to the official source code repository[12] of the project. The contribution has been accepted and will be part of the next stable version of QGIS (3.0), which is planned to be releases in February 2018.

[11] `https://github.com/qgis/QGIS/pull/5295`
[12] `https://github.com/qgis/QGIS`

Chapter 5

Planning and Practical Aspects

This chapter describes my research plan and the practical aspects of the proposed research outlined in the previous chapters.

5.1 Timetable

The timeline of my research over the course of the 4 years that it lasts, is presented by Figure 5.1. This is defined by the topics that I described during my methodology, which are split in more time-oriented objectives.

I intend to keep up with this timetable as much as possible, although my plan is flexible and it can adapt to the influences that will be introduced by the flow of my research. This can be further needed due to the additional tasks and activities that I will have to undertake as part of my employment with the university and the needs of the group. Therefore, it can be that attendance to conferences, summer schools and courses, research visits and academic responsibilities might alter the schedule in certain cases.

5.1.1 Short Term First Year Plan

During my first year I worked on my literature review and started working on practical aspects of geometrical and topological manipulation. The key events during this year are listed in Table 5.1.

5.2 Software and Libraries

My research is greatly relying on the implementation of solutions. For this reason, various software programs and libraries are being used and will be further utilised, as listed in Table 5.2.

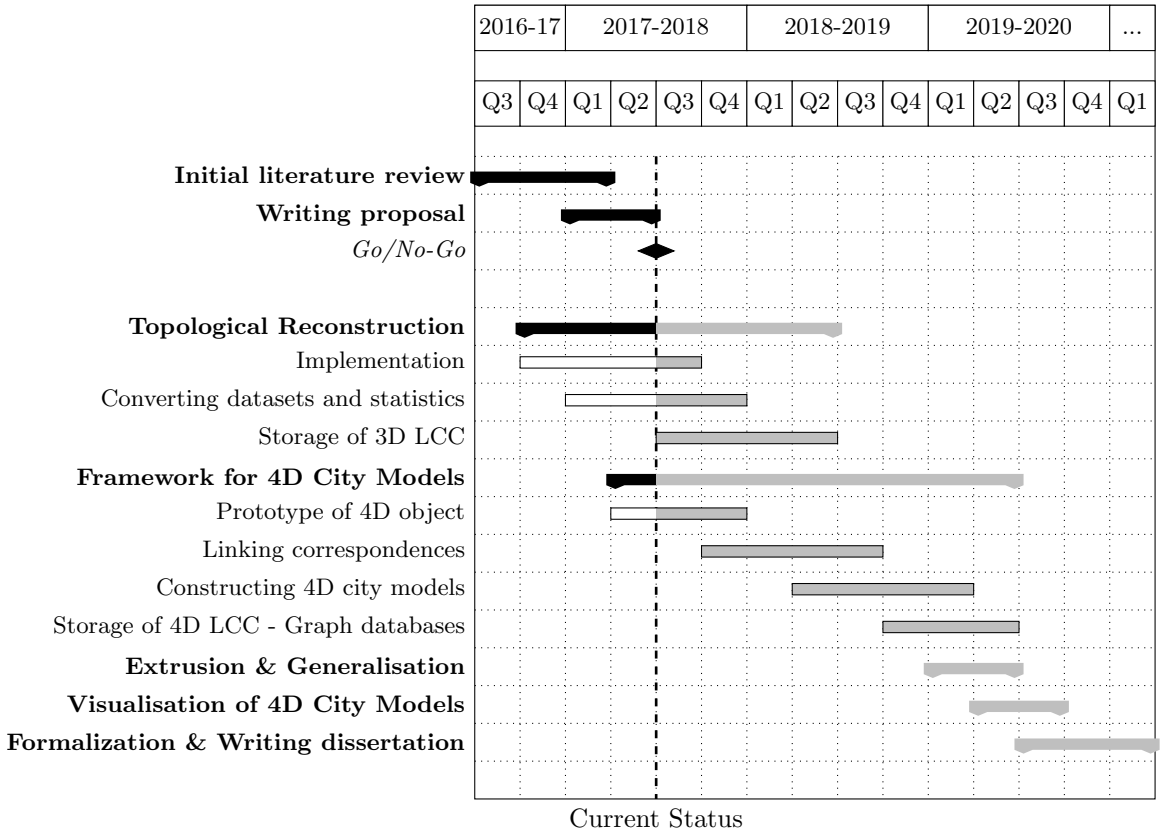


Figure 5.1: Planning Overview

5.3 Publications in Progress

- AGILE 2018 short paper: Topological Reconstruction of 3D City Models with semantics
- GIScience 2018 short paper: A Framework for the Construction of 4D City Models
- 3D Geoinfo 2018 paper: A prototype 4D object representing multiple LoDs

5.4 Graduate School

According to the Doctoral Regulations & Implementation Decree of TU Delft, in conjuction with my research I have to follow the Doctoral Education (DE) programme with the university's Graduate School (GS). During the four years

Date	Event
03-2017	Start of my Research
04-2017	First sub-objectives defined
05-2017	First results on Topological Reconstruction
06-2017	Attended Geometry Understanding in Higher Dimensions symposium
07-2017	Submitted paper on UDMV 2017 (event cancelled)
07-2017	Attended FOSS4G Europe 2017
07-2017	Closely working with GDAL developers on GMLAS
09-2017	QGIS 3D contribution merged in to master (to be released with 3.0)
10-2017	Began supervising a Master's student
11-2017	Presented my topic on NCG Symposium

Table 5.1: Key events during the first year of my research

of my PhD activities, I will have to earn 45 GS credits (1 credit equals 8 hours of coursework and 4 hours of preparation/assignments). Those credits must be fulfilled by 3 individual categories: (a) Discipline-related skills, which can be acquired by attending courses, schools and workshops related to my topic; (b) Research skills, which can be earned by courses provided by the university's GS organisation or by Learning on-the-Job activities; and (c) Transferable skills, which can be earned by GS courses.

In Table 5.3, I list all courses and activities that I have completed so far or that I intend to undertake in the future in order to earn the required amount of GS credits during the course of my PhD.

5.5 Acknowledgments

The research leading to this paper has received funding from the European Research Council under the European Union's Horizon2020, ERC Agreement no. 677312 UMnD: Urban modelling in higher dimensions.

Technology	**Purpose**
Programming Languages	
C++	Computation Geometry and High-Performance computing
Python	Prototyping of Software
HTML/JavaScript	Web applications
Libraries	
CGAL	3D Geometry Manipulation, LCCs, C-Maps
GDAL	Parsing of CityGML, Exchange with other formats
libcitygml	Parsing of CityGML (obsolete)
Qt5 (3D)	Graphical User Interface (GUI) and 3D visualisation
Software	
Azul	3D City Models inspection on Mac
QGIS 3.0	GIS data inspection (cross-platform)
GDAL CLI tools	CityGML & GIS data transformation
Qt Creator	Integrated Development Environment (IDE)

Table 5.2: Key events during the first year of my research

Category	Name	Credits	Status
Discipline Skills	Geometric Modelling (MSc Course)	5	Complete
	GMLAS made easy in GDAL [...] (Workshop)	0.5	Complete
	Hands-on on geometric software (Workshops)	4	Planned
	TBD	5.5	Planned
Research Skills	The Informed Researcher: Information [...]	1.5	Planned
	Project Management of Your PhD Project	1.5	Planned
	Effective Management of your PhD Research	1	Planned
	Speedreading and Mindmapping	1.5	Planned
	Using Creativity to Maximize [...]	1.5	Planned
Learning on-the-Job	Addressing a small audience	0.5	Complete
	Preparing and giving lecture in a MSc Course	2	Complete
	Writing the first conference paper	1	In Progress
	Supervising a MSc student	2	In Progress
	Work consultation with research partner	0.5	Planned
	Writing an international, peer-reviewed journal article	2	Planned
Transferable Skills	PhD Start-Up	2	Complete
	Dutch for Foreigners	3	Planned
	Voice Training	1	Planned
	The Art of Presenting Science	3	Planned
	Writing a Dissertation	3	Planned
	Career Development [...]	1	Planned
	How to Become Effective in a Network Conversation	1	Planned
	Social Media for PhD Candidates	1	Planned

Table 5.3: The list of course and activities that I have completed or I am planning to attend in order to gather the necessary GS credits.

Bibliography

K. Aringer and R. Roschlaub. Bavarian 3d building model and update concept based on LiDAR, image matching and cadastre information. In *Lecture Notes in Geoinformation and Cartography*, pages 143–157. Springer International Publishing, 2014. doi: 10.1007/978-3-319-00515-7_9.

Ken Arroyo Ohori. *Higher-dimensional modelling of geographic information*. PhD thesis, Delft University of Technology, apr 2016. ISBN: 978-1-326-59638-5.

Ken Arroyo Ohori, Guillaume Damiand, and Hugo Ledoux. Constructing an n-dimensional cell complex from a soup of (n-1)-dimensional faces. In Prosenjit Gupta and Christos Zaroliagis, editors, *Applied Algorithms. First International Conference, ICAA 2014, Kolkata, India, January 13-15, 2014. Proceedings*, volume 8321 of *Lecture Notes in Computer Science*, pages 37–48. Springer International Publishing Switzerland, Kolkata, India, jan 2014. ISBN: 978–3–319–04125–4 (Print) 978–3–319–04126–1 (Online) ISSN: 0302–9743 (Print) 1611–3349 (Online).

Ken Arroyo Ohori, Hugo Ledoux, Filip Biljecki, and Jantien Stoter. Modeling a 3d city model and its levels of detail as a true 4d model. *ISPRS International Journal of Geo-Information*, 4(3):1055–1075, jul 2015a. doi: 10.3390/ijgi4031055.

Ken Arroyo Ohori, Hugo Ledoux, and Jantien Stoter. Storing a 3D city model, its levels of detail and the correspondences between objects as a 4D combinatorial map. In Alias Abdul Rahman, Umit Isikdag, and Francesc Antón Castro, editors, *ISPRS Joint International Geoinformation Conference 2015*, volume II–2/W2 of *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, pages 1–8, Kuala Lumpur, Malaysia, oct 2015b. ISPRS. doi: 10.5194/isprsannals-ii-2-w2-1-2015. ISSN: 2194–9042 (Print), 2194–9050 (Internet and USB).

Ken Arroyo Ohori, Hugo Ledoux, and Jantien Stoter. 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):825–849, feb 2015c. doi: 10.1080/13658816.2014.999683.

Ken Arroyo Ohori, Hugo Ledoux, and Jantien Stoter. A dimension-independent extrusion algorithm using generalised maps. *International Journal of Geographical Information Science*, 29(7):1166–1186, mar 2015d. doi: 10.1080/13658816.2015.1010535.

F Biljecki, H Ledoux, and J Stoter. 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): 31, 2014a. doi: 10.5194/isprsannals-ii-2-31-2014.

Filip Biljecki. *Level of detail in 3D city models*. PhD thesis, 2017.

Filip Biljecki, Hugo Ledoux, and Jantien Stoter. Improving the consistency of multi-LOD CityGML datasets by removing redundancy. In *Lecture Notes in Geoinformation and Cartography*, pages 1–17. Springer International Publishing, nov 2014b. doi: 10.1007/978-3-319-12181-9_1.

Filip Biljecki, Gerard BM Heuvelink, Hugo Ledoux, and Jantien Stoter. Propagation of positional error in 3d gis: estimation of the solar irradiation of building roofs. *International Journal of Geographical Information Science*, 29 (12):2269–2294, 2015a. doi: 10.1080/13658816.2015.1073292.

Filip Biljecki, Jantien Stoter, Hugo Ledoux, Sisi Zlatanova, and Arzu Çöltekin. Applications of 3d city models: State of the art review. *ISPRS International Journal of Geo-Information*, 4(4):2842–2889, 2015b. doi: 10.3390/ijgi4042842.

Filip Biljecki, Hugo Ledoux, and Jantien Stoter. An improved LOD specification for 3d building models. *Computers, Environment and Urban Systems*, 59:25–37, sep 2016a. doi: 10.1016/j.compenvurbsys.2016.04.005.

Filip Biljecki, Hugo Ledoux, Jantien Stoter, and George Vosselman. The variants of an LOD of a 3d building model and their influence on spatial analyses. *ISPRS Journal of Photogrammetry and Remote Sensing*, 116:42–54, jun 2016b. doi: 10.1016/j.isprsjprs.2016.03.003.

Filip Biljecki, Hugo Ledoux, and Jantien Stoter. Does a finer level of detail of a 3d city model bring an improvement for estimating shadows? In *Advances in 3D Geoinformation*. Springer, January 2017. ISBN 978-3-319-25689-4. doi: 10.1007/978-3-319-25691-7_2. URL `http://dx.doi.org/10.1007/978-3-319-25691-7_2`.

R. Billen, A.-F. Cutting-Decelle, O. Marina, J.-P. de Almeida, Caglioni M., G. Falquet, T. Leduc, C. Métral, G. Moreau, J. Perret, G. Rabin, R. San Jose, I. Yatskiv, and S. Zlatanova. 3d city models and urban information: Current issues and perspectives. In R. Billen, A.-F. Cutting-Decelle, O. Marina, J.-P. de Almeida, Caglioni M., G. Falquet, T. Leduc, C. Métral, G. Moreau, J. Perret, G. Rabin, R. San Jose, I. Yatskiv, and S. Zlatanova, editors, *3D City Models and urban information: Current issues and perspectives – European COST Action TU0801*. EDP Sciences, 2014. doi: 10.1051/tu0801/201400001.

André Borrmann and Ernst Rank. Topological analysis of 3d building models using a spatial query language. *Advanced Engineering Informatics*, 23(4): 370–385, 2009. doi: 10.1016/j.aei.2009.06.001.

Kanishk Chaturvedi and Thomas H. Kolbe. Integrating dynamic data and sensors with semantic 3d city models in the context of smart cities. In Efi Dimopoulou and Peter van Oosterom, editors, *Proceedings of the 11th International 3D Geoinfo Conference*, volume IV-2/W1 of *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, Athens, Greece, 2016. ISPRS. doi: 10.5194/isprs-annals-IV-2-W1-31-2016. URL `http://www.isprs-ann-photogramm-remote-sens-spatial-inf-sci.net/IV-2-W1/31/2016/`.

Surajit Chaudhuri and Umeshwar Dayal. An overview of data warehousing and olap technology. *SIGMOD Rec.*, 26(1):65–74, March 1997. ISSN 0163-5808. doi: 10.1145/248603.248616. URL `http://doi.acm.org/10.1145/248603.248616`.

L. Paul Chew. Constrained delaunay triangulations. *Algorithmica*, 4(1-4):97–108, jun 1989. doi: 10.1007/bf01553881.

Jinmu Choi and Jiyeong Lee. *3D Geo-Network for Agent-based Building Evacuation Simulation*, pages 283–299. Springer Berlin Heidelberg, Berlin, Heidelberg, 2009. ISBN 978-3-540-87395-2. doi: 10.1007/978-3-540-87395-2_18.

Paolo Cignoni, Claudio Rocchini, and Roberto Scopigno. Metro: Measuring error on simplified surfaces. Technical report, Paris, France, France, 1996.

Alejandro Corbellini, Cristian Mateos, Alejandro Zunino, Daniela Godoy, and Silvia Schiaffino. Persisting big-data: The NoSQL landscape. *Information Systems*, 63:1–23, jan 2017. doi: 10.1016/j.is.2016.07.009.

Guillaume Damiand and Monique Teillaud. A generic implementation of dD combinatorial maps in CGAL. *Procedia Engineering*, 82:46–58, 2014. doi: 10.1016/j.proeng.2014.10.372.

Guillaume Damiand, Christine Solnon, Colin de la Higuera, Jean-Christophe Janodet, and Émilie Samuel. Polynomial algorithms for subisomorphism of nD open combinatorial maps. *Computer Vision and Image Understanding*, 115(7):996–1010, jul 2011. doi: 10.1016/j.cviu.2010.12.013.

P. de Fermat. *Varia Opera Mathematica D. Petri Fermat, Sentoris Tolosani. Accesserunt Selectae Quaedam Ejusdem Epistolae ... de Rubus Ad Mathematicas Disciplinas, Aut Physicam Pertinentibus Scriptae*. Apud Joannem Pech, 1679. URL `https://books.google.nl/books?id=-Bx3jwEACAAJ`.

R. Descartes. *Discours de la méthode, Les passions de l'âme, Lettres*. Les grands classiques illustrés. Éditions du Monde Moderne, 1637. URL `https://books.google.nl/books?id=VTMMAAAAIAAJ`.

Abdoulaye A. Diakite, Guillaume Damiand, and Gilles Gesquie‘re. Automatic semantic labelling of 3d buildings based on geometric and topological information. 2015.

Abdoulaye Abou Diakité, Guillaume Damiand, and Dirk Van Maercke. Topological Reconstruction of Complex 3D Buildings and Automatic Extraction of Levels of Detail. In Vincent Tourre Gonzalo Besuievsky, editor, *Eurographics Workshop on Urban Data Modelling and Visualisation*, pages 25–30, Strasbourg, France, April 2014. Eurographics Association. doi: 10.2312/udmv.20141074. URL `https://hal.archives-ouvertes.fr/hal-01011376`.

Liuyun Duan and Florent Lafarge. Towards large-scale city reconstruction from satellites. *Computer Vision – ECCV 2016*, January 2016. doi: 10.1007/978-3-319-46454-1_6. URL `http://dx.doi.org/10.1007/978-3-319-46454-1_6`.

J.R. Edmonds. *A Combinatorial Representation for Oriented Polyhedral Surfaces*. 1960. URL `https://books.google.nl/books?id=vo2ENwAACAAJ`.

L Euler. Solutio problematis ad geometriam situs pertinentis. 8:128–140, 01 1736.

Xin Feng, Yuanzhen Wang, Yanlin Weng, and Yiying Tong. Compact combinatorial maps: A volume mesh data structure. *Graphical Models*, 75(3):149–156, may 2013. doi: 10.1016/j.gmod.2012.10.001.

Waldemar Celes Filho, Luiz Henrique De Figueiredo, Marcelo Gattass, and Paulo Cezar Carvalho. A topological data structure for hierarchical planar subdivisions. In *In: 4th SIAM Conference on Geometric Design*, 1995.

R. Fitzpatrick and J.L. Heiberg. *Euclid's Elements*. Richard Fitzpatrick, 2007. ISBN 9780615179841. URL `https://books.google.nl/books?id=7HDWIOoBZUAC`.

M. Fréchet. *Sur quelques points du calcul fonctionnel, par M. Maurice Fréchet, ...* 1906. URL `https://books.google.gr/books?id=WR6smgEACAAJ`.

Benjamin Gorszczyk, Guillaume Damiand, Sylvie Servigne, Abdoulaye Diakité, and Gilles Gesquière. An Automatic Comparison Approach to Detect Errors on 3D City Models. In Vincent Tourre and Filip Biljecki, editors, *Eurographics Workshop on Urban Data Modelling and Visualisation*. The Eurographics Association, 2016. ISBN 978-3-03868-013-0. doi: 10.2312/udmv.20161416.

Leonidas Guibas and Jorge Stolfi. Primitives for the manipulation of general subdivisions and the computation of voronoi. *ACM Transactions on Graphics*, 4(2):74–123, apr 1985. doi: 10.1145/282918.282923.

Felix Hausdorff. *Grundzüge der mengenlehre.* Von Veit, Leipzig, 1914.

Sébastien Horna, Guillaume Damiand, Abdoulaye Diakité, and Daniel Meneveaux. Combining Geometry, Topology and Semantics for Generic Building Description and Simulations. In Filip Biljecki and Vincent Tourre, editors, *Eurographics Workshop on Urban Data Modelling and Visualisation*. The Eurographics Association, 2015. ISBN 978-3-905674-80-4. doi: 10.2312/udmv.20151343.

Kazuaki Iwamura, Keiro Muro, Nobuhiro Ishimaru, and Manabu Fukushima. 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, jun 2011. doi: 10.1109/icsdm.2011.5969001.

Robert Kaden and Thomas H. Kolbe. Simulation-based total energy demand estimation of buildings using semantic 3d city models. *International Journal of 3-D Information Modeling*, 3(2):35–53, apr 2014. doi: 10.4018/ij3dim.2014040103.

Casimir Kuratowski. Sur l'opération Ā de l'analysis situs. *Fundamenta Mathematicae*, 3(1):182–199, 1922. ISSN 0016-2736.

Mahesh Lal. *Neo4j Graph Data Modeling*. Packt Publishing, 2015. ISBN 1784393444. URL `http://www.ebook.de/de/product/24783377/mahesh_lal_neo4j_graph_data_modeling.html`.

P. Lienhardt. N-dimensional generalized combinatorial maps and cellular quasi-manifolds. 1994. International Journal on Computational Geometry and Applications, Vol. 4, n 3, pp. 275-324.

M.-O. Löwner, G. Gröger, J. Benner, F. Biljecki, and C. Nagel. PROPOSAL FOR a NEW LOD AND MULTI-REPRESENTATION CONCEPT FOR CITYGML. *ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information Sciences*, IV-2/W1:3–12, October 2016. doi: 10.5194/isprs-annals-iv-2-w1-3-2016.

David Luebke, Benjamin Watson, Jonathan D. Cohen, Martin Reddy, and Amitabh Varshney. *Level of Detail for 3D Graphics*. Elsevier Science Inc., New York, NY, USA, 2002. ISBN 1558608389.

B. Mandelbrot. How long is the coast of britain? statistical self-similarity and fractional dimension. *Science*, 156(3775):636–638, may 1967. doi: 10.1126/science.156.3775.636.

D.E. Muller and F.P. Preparata. Finding the intersection of two convex polyhedra. *Theoretical Computer Science*, 7(2):217–236, 1978. doi: 10.1016/0304-3975(78)90051-8.

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

Open Geospatial Consortium. City Geography Markup Language (CityGML) Encoding Standard, version: 2.0.0, 2012a. URL http://www.opengeospatial.org/standards/citygml.

Open Geospatial Consortium. OpenGIS Geography Markup Language (GML) Encoding Specification, version: 3.1.1, 2012b. URL http://www.opengeospatial.org/standards/gml.

F. Pédrinis, M. Morel, and G. Gesquière. Change detection of cities. In *Lecture Notes in Geoinformation and Cartography*, pages 123–139. Springer International Publishing, nov 2014. doi: 10.1007/978-3-319-12181-9_8.

M. Russo. *Polygonal Modeling: Basic and Advanced Techniques*. Wordware game and graphics library. Jones & Bartlett Learning, 2006. ISBN 9781598220070. URL https://books.google.nl/books?id=yBd7Ldw-8dAC.

Mark Segal and Kurt Akeley. The OpenGL Graphics System: A Specification (Version 3.1). Technical report, The Khronos Group Inc., March 2009.

Volker Steinhage, Jens Behley, Steffen Meisel, and Armin B Cremers. Automated updating and maintenance of 3d city models. In *ISPRS joint workshop on" Core Spatial Databases-Updating, Maintenance and Services-from Theory to Practice*, pages 1–6, 2010.

Jantien Stoter, Henk de Kluijver, and Vinaykumar Kurakula. 3d noise mapping in urban areas. *International Journal of Geographical Information Science*, 22(8):907–924, aug 2008. doi: 10.1080/13658810701739039.

Jantien Stoter, Carsten Roensdorf, Rollo Home, Dave Capstick, André Streilein, Tobias Kellenberger, Eric Bayers, Paul Kane, Josef Dorsch, Piotr Woźniak, Gunnar Lysell, Thomas Lithen, Benedicte Bucher, Nicolas Paparoditis, and Risto Ilves. 3d modelling with national coverage: Bridging the gap between research and practice. In *Lecture Notes in Geoinformation and Cartography*, pages 207–225. Springer International Publishing, nov 2014. doi: 10.1007/978-3-319-12181-9_13.

Jantien Stoter, Hendrik Ploeger, Ruben Roes, Els van der Riet, Filip Biljecki, Hugo Ledoux, Dirco Kok, and Sangmin Kim. Registration of multi-level property rights in 3d in the netherlands: Two cases and next steps in further implementation. *ISPRS International Journal of Geo-Information*, 6(6):158, may 2017. doi: 10.3390/ijgi6060158.

Filippo Tampieri. Newell's method for computing the plane equation of a polygon. In *Graphics Gems III (IBM Version)*, pages 517–518. Elsevier, 1992. doi: 10.1016/b978-0-08-050755-2.50112-3.

Peter van Oosterom and Martijn Meijers. Vario-scale data structures supporting smooth zoom and progressive transfer of 2d and 3d data. *International Journal of Geographical Information Science*, 28(3):455–478, dec 2013. doi: 10.1080/13658816.2013.809724.

Peter van Oosterom and Jantien Stoter. 5D data modelling: Full integration of 2D/3D space, time and scale dimensions. In Sara Irina Fabrikant, Tumasch Reichenbacher, Marc van Kreveld, and Christoph Schlieder, editors, *Geographic Information Science: 6th International Conference, GIScience 2010, Zurich, Switzerland, September 14-17, 2010. Proceedings*, pages 311–324. Springer Berlin Heidelberg, 2010. doi: 10.1007/978-3-642-15300-6_22.

Vasco Varduhn, Ralf-Peter Mundani, and Ernst Rank. Multi-resolution models: Recent progress in coupling 3d geometry to environmental numerical simulation. In *Lecture Notes in Geoinformation and Cartography*, pages 55–69. Springer International Publishing, nov 2014. doi: 10.1007/978-3-319-12181-9_4.

Andrew Vince. Combinatorial maps. *Journal of Combinatorial Theory, Series B*, 34(1):1–21, feb 1983. doi: 10.1016/0095-8956(83)90002-3.

Alexander Michael Zhivov, Michael Patrick Case, Reinhard Jank, Ursula Eicker, and Samuel Booth. Planning tools to simulate and optimize neighborhood energy systems. In *Green Defense Technology*. Springer, January 2017. ISBN 978-94-017-7598-4. doi: 10.1007/978-94-017-7600-4_8. URL `http://dx.doi.org/10.1007/978-94-017-7600-4_8`.

S. Zlatanova and J. Beetz. 3d spatial information infrastructure: The case of port rotterdam. In T. Leduc, G. Moreau, and R. Billen, editors, *Usage, Usability, and Utility of 3D City Models – European COST Action TU0801*. EDP Sciences, 2012. doi: 10.1051/3u3d/201203010.

Sisi Zlatanova. On 3d topological relationships. In *Database and Expert Systems Applications, 2000. Proceedings. 11th International Workshop On*, pages 913–919. IEEE, 2000. doi: 10.1109/DEXA.2000.875135.