Lecture Notes
in Geoinformation and Cartography

LNG&C

Martin Breunig · Mulhim Al-Doori
Edgar Butwilowski · Paul V. Kuper
Joachim Benner · Karl Heinz Haefele *Editors*

3D Geoinformation Science

The Selected Papers of the 3D GeoInfo 2014

Springer

Martin Breunig · Mulhim Al-Doori
Edgar Butwilowski · Paul V. Kuper
Joachim Benner · Karl Heinz Haefele
Editors

3D Geoinformation Science

The Selected Papers of the 3D GeoInfo 2014

Springer

Editors
Martin Breunig
Geodetic Institute
Karlsruhe Institute of Technology (KIT)
Karlsruhe
Germany

Paul V. Kuper
Geodetic Institute
Karlsruhe Institute of Technology (KIT)
Karlsruhe
Germany

Mulhim Al-Doori
School of Engineering
American University in Dubai
Dubai
UAE

Joachim Benner
Institute for Applied Computer Science
Karlsruhe Institute of Technology (KIT)
Karlsruhe
Germany

Edgar Butwilowski
Geodetic Institute
Karlsruhe Institute of Technology (KIT)
Karlsruhe
Germany

Karl Heinz Haefele
Institute for Applied Computer Science
Karlsruhe Institute of Technology (KIT)
Karlsruhe
Germany

ISSN 1863-2246 ISSN 1863-2351 (electronic)
Lecture Notes in Geoinformation and Cartography
ISBN 978-3-319-12180-2 ISBN 978-3-319-12181-9 (eBook)
DOI 10.1007/978-3-319-12181-9

Library of Congress Control Number: 2014957324

Springer Cham Heidelberg New York Dordrecht London

Printed on acid-free paper

Springer International Publishing AG Switzerland is part of Springer Science+Business Media (www.springer.com)

Contents

Improving the Consistency of Multi-LOD CityGML Datasets by Removing Redundancy

Filip Biljecki, Hugo Ledoux and Jantien Stoter

Abstract The CityGML standard enables the modelling of some topological relationships, and the representation in multiple levels of detail (LODs). However, both concepts are rarely utilised in reality. In this paper we investigate the linking of corresponding geometric features across multiple representations. We describe the possible topological cases, show how to detect these relationships, and how to store them explicitly. A software prototype has been implemented to detect matching features within and across LODs, and to automatically link them by establishing explicit topological relationships (with XLink). The experiments ran on our test datasets show a considerable number of matched geometries. Further, this method doubles as a lossless data compression method, considering that the storage footprint in the consolidated datasets has been reduced from their dissociated counterparts.

Keywords Multi-LOD · Topology · XLink · CityGML · Compression

1 Introduction

The OGC standard CityGML (Gröger and Plümer 2012; Open Geospatial Consortium 2012), and other 3D modelling standards such as COLLADA (ISO/TC 184: ISO/PAS 17506:2012; Sony Computer Entertainment Inc. 2008) and ISO's X3D

F. Biljecki (✉) · H. Ledoux · J. Stoter
Section GIS Technology, Delft University of Technology, Delft, The Netherlands
e-mail: f.biljecki@tudelft.nl

H. Ledoux
e-mail: h.ledoux@tudelft.nl

J. Stoter
Kadaster, Product and Process Innovation, Apeldoorn, The Netherlands
e-mail: j.e.stoter@tudelft.nl

J. Stoter
Geonovum, Amersfoort, The Netherlands

M. Breunig et al. (eds.), *3D Geoinformation Science*,
Lecture Notes in Geoinformation and Cartography,
DOI 10.1007/978-3-319-12181-9_1

(ISO/IEC 19775-1:2013) allow the storage of multiple level of detail (LOD) representations of a model, in order to facilitate the multi-scale use of the models and to improve the computational efficiency of spatial operations (Biljecki et al. 2014a).

Although 3D GIS datasets may contain multiple LODs, multi-LOD datasets are almost non-existent in practice, and they are seldom linked beyond an administrative link between object identifiers (i.e. they share the same building ID) (Biljecki et al. 2013). In our opinion, this situation is caused by the following deficiencies: (1) lack of consistency, i.e. there is redundancy in the acquisition–modelling–storage process; (2) when using multi-LOD datasets, it is not always clear when and how to switch between LODs as it is the case in computer graphics; and (3) 3D generalisation specifications and implementations are not fully developed, limiting the generation of LODs other than the one being primarily constructed from the acquired dataset.

In this paper we focus on the first shortcoming by investigating the possible improvements in the consistency and storage of multi-LOD datasets, with a theory that is applicable also to single-LOD representations. It is our experience that in practice, besides exemplary models, single-LOD datasets do not contain the explicit representation of topological relationships, hence developing a joint method that is beneficial for both possibilities is important. We observe and take advantage of the practical fact that many of the stored geometries (primarily polygons) in 3D datasets are geometrically equal both within a single LOD and across multiple LODs. By determining the topological relationships between such reoccurring geometries and storing them explicitly, the consistency of 3D models can be increased, as we show in this paper. However, while developing the method, we have realised that in practice most of the geometries that are reoccurring are not identical and cannot be readily matched. Therefore, we have investigated other cases and covered them as well.

This paper is focused towards CityGML and its LOD concept, however, most of the developed work is applicable to other formats. Our work consists of the following contributions: (1) we have investigated and described several cases of reoccurring geometries and introduce a terminology to distinguish them; (2) we have developed robust algorithms that efficiently index the geometries in CityGML datasets and that take advantage of the geometries that reoccur by explicitly storing their topological relationships; (3) we have developed a software prototype that analyses CityGML data and automatically computes explicit topological links between matching geometric features through the XML Linking Language (XLink) mechanism; and (4) we show with experiments that a considerable subset of data can be matched. We have tested the method on a synthetic dataset that contains buildings in LOD1, LOD2, and two variants of LOD3.

Because matched geometries are stored only once, the consolidated dataset is compressed without loss of information.

In Sect. 2 we explain the advantages of establishing explicit topological representations between geometric features. In Sect. 3 we describe possible topological cases, introduce our terminology, and present the algorithms that index, match, and consolidate the geometry of 3D datasets, primarily CityGML. The consolidation of the data is a technical challenge because it is done on a hierarchical data structure, and there might not be an optimal algorithm that is suited for all CityGML datasets.

The implementation and the results, presented in Sect. 4, show that after linking a higher degree of the consistency of the data is achieved, contributing to an efficient storage and maintenance. For instance, if the geometry of a feature is altered in one LOD, thanks to the established explicit topological representations this change may propagate through other LODs.

2 Background and Related Work

Consistency of 3D city models is an important topic in GIS, in which topology plays a prominent role (Gröger and Plümer 2009; Ledoux and Meijers 2011). Current research efforts focus on the relationships of features within the same representation, e.g. the validation of solids (Ledoux 2013), and making use of topological data structuring to improve rendering performance of 3D city models on mobile devices (Ellul and Altenbuchner 2014). To the extent of our knowledge, there is no related work to detect and link geometric features across multiple representations.

While this paper generally describes a way how to increase the consistency and to compute topological relationships in a model, it is focused on the maintenance and storage of (3D) GIS datasets, which is a topical subject in academia and industry (Aringer and Roschlaub 2014; Steinhage et al. 2010; Stoter et al. 2011). Updates of models often introduce errors (Gröger and Plümer 2012), so increasing consistency is one of the prerequisites for an efficient workflow.

In this section we describe the redundancy and benefits of an established topology with respect to the scalability of the models: for single representations (single-LOD datasets), and for multiple representations (multi-LOD datasets).

2.1 Single-LOD Datasets

Research that has been done in this topic is focused on the relationship of real-world features within the same representation (Gröger and Plümer 2005, 2011). For instance, the topology of two coinciding polygons, such as a wall shared by two buildings. The consistency that is achieved by establishing explicit topological relationships in practice simplifies the maintenance of the data and reduces the redundancy in the storage.

Figure 1 shows an example of the benefit with respect to the maintenance of a 3D model. The left model (Fig. 1a) shows a building with a wall that contains a window. The polygon representing the wall is shown in red, and it contains a hole (inner ring), which is filled by another polygon representing the window. The interior ring of the wall polygon corresponds to the exterior ring of the polygon representing the window. In a model without established explicit topological relationships, the two features are not linked in any way. When the geometry of a part of the object is updated, e.g. the window is enlarged, the change does not affect

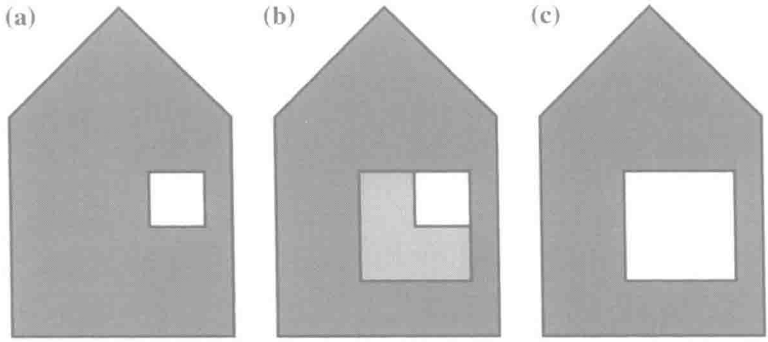


Fig. 1 The determined explicit topological relationships in a dataset has a significant benefit to its maintenance. This example shows the benefit on a wall with a window that is being enlarged. **a** Model of a building before the update (enlargement of the window). **b** Updated model without links resulting in inconsistency. **c** Updated model with established links (desired case)

the related geometry (i.e. hole of the wall), leading to redundancy in the process (see Fig. 1b). In a model with established topological representations, the change properly propagates to the related features (see Fig. 1c for the desirable outcome).

2.2 Multi-LOD Datasets

On top of the redundancy in a single representation, that multiplies with each new representation in a multi-LOD dataset, there is also additional redundancy. For instance, should a feature be changed, in practice the update must be done manually for each representation. Further, because of the increasing complexity of the models, the size of the datasets substantially increases with the increase of the LOD, making the storage less feasible. The surge in the size is not only caused by the growth of the amount of details, but also because of the redundancy that could be removed, as we show later in the paper.

Therefore, despite the fact that this option is available in CityGML, models are usually derived in a single LOD representation. While multi-LOD datasets are rare, when they are available they are usually produced by generalisation from finer LODs (e.g. see Akmalia et al. 2014; Baig and Abdul-Rahman 2013; Zhao et al. 2012), for instance, as a bounding box of an LOD2 (El-Mekawy and Östman 2011), or of an LOD3 including features such as antennas on roofs (Mao et al. 2012). This is beneficial for this research, since it results in datasets where many of the geometries are preserved, and are identical in more than one representation. For instance, the ground surface of a building (i.e. Ground Surface) is usually identical in all representations.

Detecting and linking such occurrences would be a first step towards complementing the discussed practical shortcomings.

3 Methodology

We have developed a method that searches for matching geometries in the datasets and links them. Figure 2 shows the desirable outcome of the algorithms with an example of three LODs where some of the geometries are reoccurring and are consequently linked.

While examining multi-LOD datasets we have realised that there are different cases of corresponding geometry, not only polygons that are identical and that can be directly referenced. For instance, polygons that share the exterior ring, but their interior is different (common in CityGML LOD3 where openings are allowed, see Fig. 2). Further, specific cases such as two equal polygons whose starting point is different should also be handled.

3.1 Terminology

In this paper we focus on the two geometric feature types: polygons and linear rings. We consider two or more geometric primitives *identical* if they are topologically and geometrically equivalent, i.e. they can be readily linked and re-used. The geometric representations of the ground plane of a building in two LODs are usually identical. Two or more primitives are *partially identical* if they are not identical and if their relation has one or more of the following properties which prevent them to be identical:

- The orientation of their vertices is different, i.e. their normals are reversed. For instance, two buildings share the same wall.

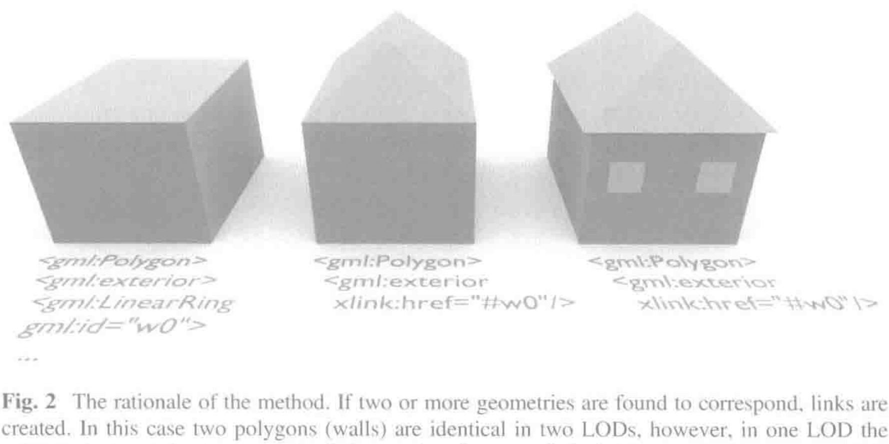


Fig. 2 The rationale of the method. If two or more geometries are found to correspond, links are created. In this case two polygons (walls) are identical in two LODs, however, in one LOD the polygon has a hole, hence, only their exterior rings are linked. How and where to establish the links while balancing the maximisation of links and the topological structure is the main concern of the consolidation process

- They constitute different aspects of their parent primitive, e.g. two linear rings correspond, but one forms the exterior ring of its parent polygon, while the other forms the interior ring to describe a hole. A prominent example of this case is a wall with a hole that represents an opening—window or door which is stored separately.
- The number of points in their rings is not equal while the shape and location are identical. This is caused by *redundant* points p_i, where p_{i-1}, p_i, p_{i+1} are collinear. The removal of such points would not compromise the shape and location of the polygon.
- The starting point in the linear ring is different. This discrepancy might be easily detected and corrected *on-the-fly*, hence it will not be particularly emphasised in the continuation of the paper.

Two geometries *match* if they are either identical or partially identical. When a match of two or more geometries is found, one is selected as the *resource*, and the rest are *linked* to it.

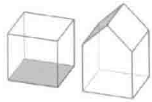
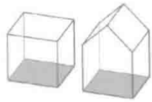
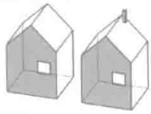
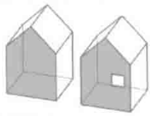
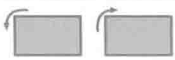
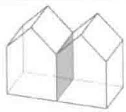
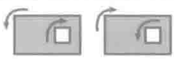
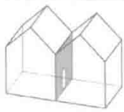
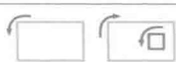
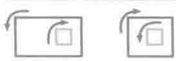
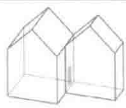
3.2 Topological Relationships

In this section we show the possible cases of the *matching* geometry, i.e. topological relationships between polygons, and their constituting components—exterior and interior ring(s). We have investigated the possible cases, and their occurrences in real-world datasets, which we show in Table 1. The sign = denotes identical primitives,—partially identical, ≠ non-matching, and ∅ denotes no geometry. The matched primitives in each case are shown in red. In the example column, when two objects are separated, the example refers to the multi-LOD case.

While more "permutations" are possible, they are not valid according to GML, hence the list does not take into account invalid cases. One such example are two exterior rings that are identical, but their interior rings have a reversed orientation (which is shown in the last line as an exceptional example).

The case 0 is the (usual) case where two primitives have no topological relation in the context of this research. Identical features rarely occur within the same LOD (in contrast to partially identical), so the cases 1 and 2 where the geometries of two features are identical are often present in multi-LOD datasets. Case 3 is also typical in multi-LOD datasets (again, see Fig. 2), while case 4 is more common in single-LOD datasets where two buildings share the same wall. The fifth case extends the previous with holes. Cases 6 and 7 are unusual in the real-world, and case 8 is similar to the fifth case in the occasion when the polygon is not identical. Cases 9 and 10 are cases of interchangeable roles of the exterior and interior rings, and

Table 1 Cases of topological relationships of rings and polygons

Case	Ring Exterior	Ring Interior	Polygon	Graphical explanation	Real-world example
0	≠	≠	≠		
1	=	∅	=		
2	=	=	=		
3	=	≠	≠		
4	–	∅	–		
5	–	–	–		
6	–	≠	≠		Unidentified
7	≠	=	≠		Unidentified
8	≠	–	≠		
9	⟵ = ⟶		≠		Unidentified
10	⟵ – ⟶		≠		
Invalid	=	–	?		Not possible

The curved arrows denote the ringâ€™s orientation, while the long horizontal arrows indicate that there is a relation between one polygonâ€™s exterior to another polygonâ€™s interior

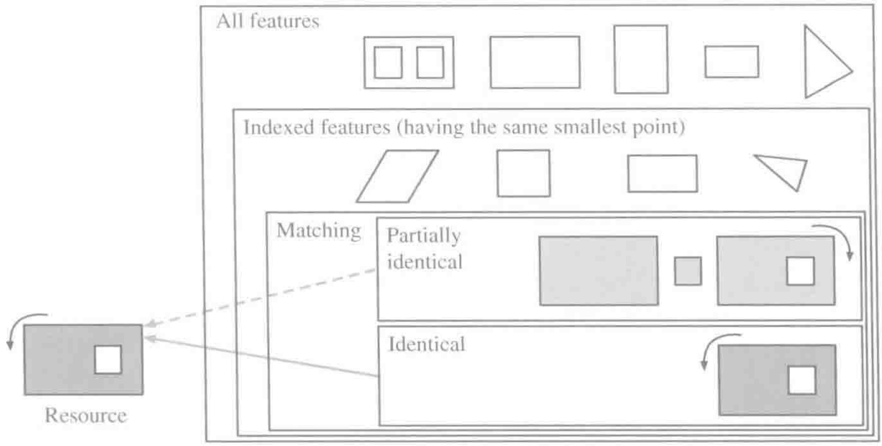


Fig. 3 Simplified classification of the relationships and workflow of the method. The primitives in the dataset are first indexed, and then tested for matches, of which there are different categories

might be rather considered as extended cases. Case 9 is uncommon, and case 10 is usually occurring in finer single-LOD datasets as the relation between a wall and an opening.

3.3 Overview of the Method

The workflow of the method to match regards both matching primitives within the same LOD and across multiple LODs:

1. Indexing. Index all points, linear rings, and polygons in all LODs for efficiency (Sect. 3.4).
2. Matching. Detect matching geometries and flag them. Because of different topological relations, which have been introduced in the previous section, the detection of the matching geometries is done in multiple phases (Sect. 3.5).
3. Consolidation. Analyse the matched geometry and remove redundant data by replacing them with a link to one other matching representation resource, with modifications if necessary (Sect. 3.6).

Figure 3 shows the simplified workflow of the method and the relation between the features when searching for matches. Because all features are indexed, the searching algorithm has a considerably reduced subset of potential matches. After ruling out non-matching features in the indexed subset, the algorithms detect the matched features and classify them according to the cases presented in the previous section.

The developed algorithm and the implemented software prototype are focused towards ideal cases where the vertices of the geometry are identical across LODs and where the geometries fully correspond. This is useful for datasets produced with generalisation, however, when used on datasets with a different lineage it might not produce results to the same extent. This could be solved by introducing the snapping of the points according to a tolerance threshold, and more sophisticated matching of similar geometries. The automatic matching of the representations that are acquired with different techniques would require employing more advanced algorithms such as (Arkin et al. 1991) and (Zheng et al. 2014), extending related work done in cartography (e.g. Dilo et al. 2009; Zhang et al. 2014) to 3D GIS, and would probably result in a lossy compression (partial data discarding).

3.4 Algorithm for Indexing the Geometry

In order to make the search for the matching geometry more efficient and enable the consolidation of larger datasets, the polygons and their rings are first indexed. We have decided to build an index where each polygon's ring is indexed according to its smallest point. The smallest point is the point with the smallest coordinate value, i.e. the one that is closest to $(-\infty, -\infty, -\infty)$. Each valid ring has one such point, hence, it can serve for the purpose of indexing. This point should not be confused with the starting point of the ring, which is not relevant here.

This step considerably reduces the search time for matches since in practice only a few non-matching polygons share the same smallest point. As an example, Fig. 4 shows two different polygons that share the same smallest point. The comparison of such polygons for discarding the different geometries is described in the next step.

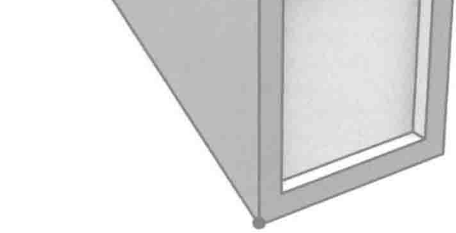


Fig. 4 Multiple different polygons (shown here in *red*) may have the same smallest point (shown in *black*). In this case the two polygons are part of a dormer of a building

3.5 Algorithm for the Detection of Matching Geometries

After indexing, the rings (both exterior and interior) are queried for their relations. The algorithm first removes vertices that are redundant (i.e. being collinear with its preceding and succeeding points). The algorithm is given in Algorithm 1.

Algorithm 1 Algorithm for the detection of matching geometries.

Input: Indexed features, where each ring has its smallest point indexed (Sec. 3.4)
Output: Topological relationships between the geometries
1: Iterate all linear rings and remove all redundant points
2: **for** each ring r_j of each polygon **do**
3:   **for** each ring r_k that shares the same smallest point **do**
4:     relation = 1 {Assume identicalness until proven otherwise}
5:     **if** r_j and r_k are identical features **then** {Avoid comparing the ring to itself}
6:       Go to 3
7:     **end if**
8:     **if** the relation between the two rings has already been checked **then**
9:       Go to 3
10:     **end if**
11:     **if** the number of points of r_j and r_k are equal **then**
12:       Reinstate the points{i.e. reorder them so the first point coincides}
13:       **while** relation = 1 **do**
14:         **for** each point p_j^i in r_j **do** {Start from the second point because the first one is identical in any case (index)}
15:           **if** $p_j^i \neq p_k^i$ **then** {Points not identical, so there is no match}
16:             relation = 0
17:           **end if**
18:         **end for**
19:       **end while**
20:       **if** relation = 0 **then** {If the match has not been found, try for the other orientation}
21:         reverse = True {Assume that the linear rings are inverted until proven otherwise}
22:         n = number of points
23:         **while** reverse = True **do**
24:           **for** each point p_j^i in r_j **do** {Start from the second point because the first one is equal in any case (index)}
25:             **if** $p_j^i \neq p_k^{n-i}$ **then** {Points not identical when the rings are reversed}
26:               reverse = False
27:             **end if**
28:           **end for**
29:         **end while**
30:         **if** reverse = True **then** {the relation between the rings is reversed}
31:           relation = -1
32:         **end if**
33:       **end if**
34:     **else** {If the number of points is different, the features are not identical}
35:       relation = 0
36:     **end if**
37:     **if** relation = 1 **or** relation = -1 **then** {If the rings are identical or partially identical, store this information}
38:       Store the information about the relation $r_j \leftrightarrow r_k$
39:     **end if**
40:   **end for**
41: **end for**

3.6 Algorithm for the Consolidation of the Data

After detecting matching geometries, the last phase of the method involves consolidating the data (linking), i.e. analysing the relationships and determining the level of the relationship that can be linked. A straightforward solution would be to directly link the matching rings. However, because of the different cases and hierarchies, this phase is not forthright, and it can be solved in multiple ways. For instance, if two rings that form the exterior of two polygons match, this does not necessarily mean that the polygons can be matched right away, since the interior may be different (e.g. see case 3 in Table 1). Further, one of these rings may be an interior of another polygon, that is further related to another polygon in another way. Therefore, maximising the number of links that are established is the main concern when designing such algorithm, and cannot be solved simply by determining the frequency of the occurrences and selecting the topmost ring. Further, the development of the algorithm is associated with the content of a targeted dataset, as an algorithm might not be equally beneficial when employed for consolidating two different datasets. This problem is related to the field of data compression (Huffman 1952; Salomon et al. 2007).

We have designed a top-down approach that first iterates polygons with holes, comparing the matched rings, and builds a hierarchy of features, continuing to polygons without holes. This is particularly beneficial for cases 1, 2, 3 and 10, which are the most common. The algorithm is given in Algorithm 2.

4 Implementation and Results

4.1 Test Data

Because multi-LOD datasets are rare in practice, there is a difficulty to find input material for the testing of the prototype. The publicly available CityGML datasets that contain more than one LOD representation are limited to one or a few buildings (e.g. Häfele 2011).

We have used a dataset in multiple LODs that was automatically generated from a parametrised description by the engine "Random3Dcity", which was presented in Biljecki et al. (2014b). The dataset contains 100 buildings. Some of the LODs are represented in two ways, as a <gml:Solid> or as semantically structured surfaces (<gml:boundedBy>). This is done in order to extend the experiments by comparing the different variations of the models that are valid (Benner et al. 2013; Löwner et al. 2013). Further, two variants of LOD3 are available, in order to take into account the different levels of complexity that may occur in the same LOD range (see Biljecki et al. 2014a for further details). The dataset contains six representations of each building, that are shown in Fig. 5.

Algorithm 2 Algorithms for consolidating the data.

Input: Topological relationships, obtained from the previous algorithm in Sec. 3.5
Output: Consolidated dataset with established links

```
1: for each polygon P_0 that contains the interior and has at least one ring matching to a ring of at
   least one other polygon P_i do {Both exterior and interior rings are taken into account}
2:    if P_0 and P_i were already matched then
3:       Go to 1
4:    end if
5:    if P_i comprises only polygons with holes and are identical to P_0 then
6:       Establish P_0 as the resource, and link P_i
7:       Remove the primitives from further consideration
8:    end if
9:    if the exterior of P_i is identical to P_0 then
10:      Establish the exterior of P_0 as the resource, and link P_i
11:      Remove the primitives from further consideration
12:   end if
13:   if P_i is partially identical to P_0 then
14:      Establish the exterior of P_0 as the resource with necessary modifications, and link P_i
15:      Remove the primitives from further consideration
16:   end if
17: end for
18: for each polygon P_0 that does not contain the interior and has at least one ring matching with
    a ring of at least one other polygon P_i do
19:   if P_0 and P_i were already matched then
20:      Go to 18
21:   end if
22:   Repeat the steps above without the operations on the interior
23: end for
24: Store the consolidated dataset with determined topological relationships
```

LOD1s The standard LOD1 block model, represented as a <gml:Solid>. The top of the block model represents the height of the building at the eaves. The footprint represents the *real* footprint of the building (*cf.* cadastral records).

LOD2s A model with simple standardised roof shapes. The footprint is the same as in LOD1. It is also stored as a solid.

LOD2b The semantically enriched boundary representation from which the previous model was generated.

LOD3s A solid obtained from the architecturarly detailed model. In comparison to the LOD2, it includes dormers and other objects that contribute to the internal volume of the building.

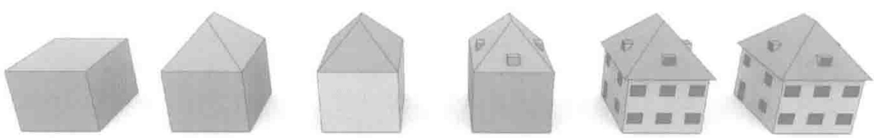


Fig. 5 Visualisation of the six representations that are available in the test data. Their order from the left is the same as in the description in the text