Assessment of the LoD specification for the integration of BIM-derived building models in 3D city models

Jasper van der Vaart Jantien Stoter Abdoulaye Diakité
Filip Biljecki Ken Arroyo Ohori Amir Hakim

This is an author's version of the paper. The authoritative version is:

Assessment of the LoD specification for the integration of BIM-derived building models in 3D city models. Jasper van der Vaart, Jantien Stoter, Abdoulaye Diakité, Filip Biljecki, Ken Arroyo Ohori and Amir Hakim. In Thomas H. Kolbe, Andreas Donaubauer and Christof Beil (eds.), *Recent Advances in 3D Geoinformation Science: Proceedings of the 18th 3D GeoInfo Conference*, Lecture Notes in Geoinformation and Cartography, Springer Cham, Munich, Germany, February 2024, pp. 171–191. ISBN: 978-3-031-43699-4 (eBook), 978-3-031-43698-7 (Hardcover), 978-3-031-43701-4 (Softcover), ISSN: 1863-2246, eISSN: 1863-2351.
DOI: 10.1007/978-3-031-43699-4_11

Although level of detail (LoD) is a central concept in 3D city modelling, specifying different LoDs in an unambiguous manner is not straightforward. To resolve this, a set of frameworks have been developed. This paper evaluates the suitability of the LoD framework of Biljecki et al. [2016] for 3D building models that have been generated directly from BIM models. The output of two BIM shell extractors are tested on how well they can be defined by the framework. It was found that although BIM-derived models can be specified by the framework to a certain degree, the framework is not fully capable to also specify lower quality models and to support all the output that may come from BIM shell extractors. This can be resolved by either addressing issues in the shell extractors' output or in the framework itself. The results of this research can be used to improve the LoD framework and to adjust the shell extractors output to better comply with unambiguous definitions of building models at different LoDs and could be a first step to standardise the conversion of BIM models at different LoDs to be used in urban applications.

1 Introduction

Level of Detail (LoD) is a central concept in 3D city modelling [Biljecki et al., 2014]. It describes how much an object has been abstracted from reality. The term originates from 3D computer graphics where it has a slightly different meaning. In computer graphics, this is a rather lenient phenomena primarily used for resource effective rendering [Luebke et al., 2003]. In contrast, in 3D city models, different LoDs are used as collections of rules (i.e. specifications) for static visualisation, acquisition, modelling, generalisation and exchange of 3D data [Biljecki et al., 2014]. The results of city scale analyses are influenced by the method and degree of object abstraction during acquisition and modelling [Biljecki et al., 2018, García-Sánchez et al., 2021, Peronato et al., 2016]. A lenient, poorly applicable or vague ruleset regarding object abstraction can influence results in possibly unpredictable ways and can introduce uncertainty. Uncertainty can also be introduced when evaluating, comparing or exchanging city models that include abstracted objects that have not been clearly defined, see Fig. 1.

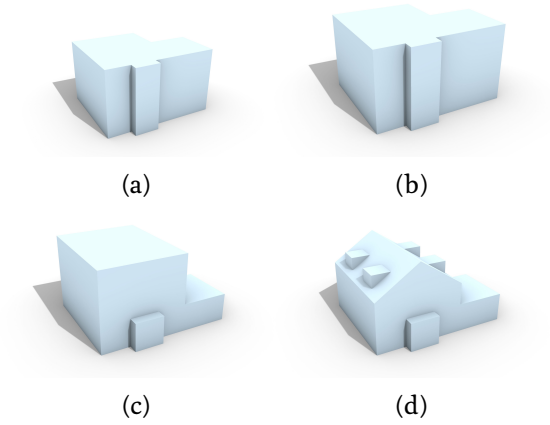
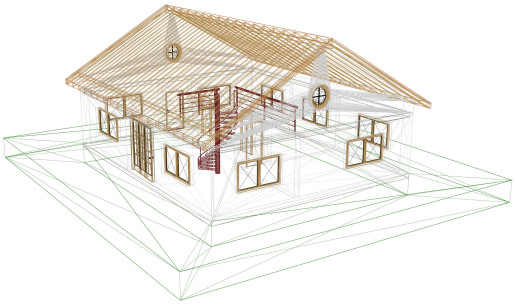


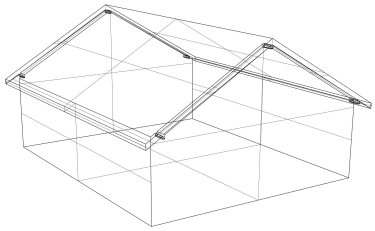
Figure 1: A model without a clear supportive LoD framework can introduce uncertainty. For example, this model could be an abstraction of a building with a single flat roof (1b), with multiple flat roofs at different heights (1c) or it could even represent a building with a gable roof (1d)

To reduce this uncertainty a ruleset is needed to unambiguously define 3D objects at different LoDs. These LoD-definitions are an interplay between the data acquisition methods (e.g. obtained from aerial, terrestrial or mobile measurements), modelling methods (e.g. manually or automatically) and the 3D data requirements of applications. The latter may vary between different applications. For example, LoD1 block models may be sufficient for noise simulations [Stoter et al., 2020], while the accurate roof structures of LoD2 models are needed for solar potential estimation of rooftops [Alam et al., 2016]. More detailed building models with information about windows and doors stored in LoD3 models are important for estimating heat losses [Geiger et al., 2018].

Biljecki et al. [2016] refined the LoD specification of the CityGML 2.0 conceptual model to define such a ruleset for buildings, see Fig. 3. This specification has indeed reduced the vagueness of the four main CityGML LoDs. However, this refined framework was established when 3D city models were mainly a product of data acquisition through measurements and observations. In recent years, new ways have been developed to generate 3D data for 3D city models, one of which is the automated abstraction of BIM (Building Information Modeling) models. BIM models contain detailed information about the designs, planning, construction and exploitation of buildings and other constructions. IFC (Industry foundation classes)-files are an open and vendor-neutral standard for exchanging BIM models which contain information related to architecture, engineering and construction projects [Borrmann et al., 2018]. The IFC-files store the majority of the data that is included in the source BIM file. BIM/IFC models are primarily utilised at an architectural scale. Thus, they often cover a smaller area than 3D city models while being more complex. The way buildings and other objects are modelled also differ between BIM and 3D city models. Buildings in BIM models are represented by a large collection of objects, while buildings represented in city scale models are primar-



(a)



(b)

Figure 2: The difference between a BIM model (a) and a city model (b) which are both based on the same building. The BIM model is not only more complex but it contains many differently modelled unique objects while the city model is represented by a single shell shape

ily modelled by their shells, see Fig. 2. BIM data can however be abstracted into generalised building models/shells, which can be integrated in 3D city models to study the impact of their design on the environment and vice versa [Noardo et al., 2022].

At the time the framework by Biljecki et al. [2016] was developed, there was little practical experience available on the use of BIM models to generate abstracted models for their use in downstream 3D geo-applications. Since then, several BIM to GIS conversion methods and tools have been developed (and more are currently under development) to extract building-related concepts from BIM, such as as building shells, building elements, storeys and rooms [Diakité, 2023, van der Vaart, 2022]. These

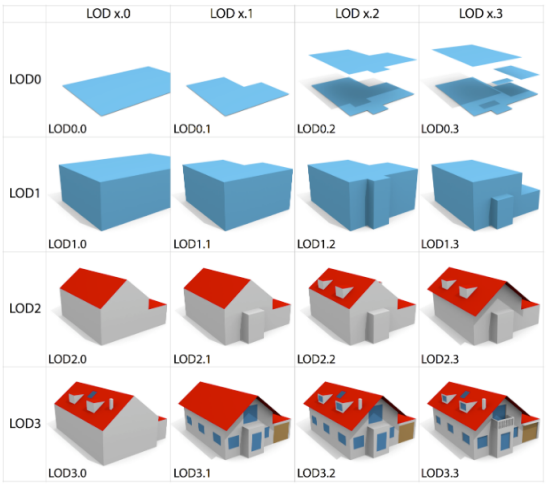


Figure 3: The refined LoD specification of Biljecki et al. [2016].

methods are quite diverse, following different rules and therefore result in vastly different outputs that often do not consider different LoDs.

BIM derived models were considered by Biljecki et al. [2016], but this was exclusively in LoD3.x. BIM models can in practice also be the direct source for further abstracted models than LoD3.x. These further abstracted BIM derived models may be required, because LoD3.x models might be too complex for certain applications. Utilising these too complex models may slow down the process unnecessarily while making the process more sensitive to errors.

One could reason that if the LoD3.3 shell could be generalised from a BIM model, the other LoD shells can be abstractions from this LoD3.3 shell, as is implied by Biljecki et al. [2016]. However, experiences have shown that the quality of the input BIM model can impact the LoD shells that an automated abstraction process is able to extract. Thus, an input BIM model might not be suited to automatically extract LoD3.x output from while being sufficient for LoD2.x output or lower.

For these reasons, BIM derived output might not fit neatly into the existing LoD framework. Very little research on the fitting of BIM derived city models in LoD frameworks has been done. Thus, it is un-

clear if BIM derived output can sufficiently be defined by the current LoD framework or if further refinements are needed to be able to define BIM-derived output in an unambiguous manner.

1.1 Goal & Scope

The goal of this paper is to evaluate how suitable the LoD framework of Biljecki et al. [2016] is for the integration of BIM-derived models in 3D city models. Based on this evaluation refinements can be proposed, or missing knowledge can be highlighted which requires further research. The evaluation in this paper is done for the entire LoD range and not limited to LoD3.x. The results of this research can be utilised to improve the LoD framework as well as BIM-to-Geo conversion methods and could be a first step to standardise the conversion of BIM models to building models at different LoDs to be used in urban applications.

To evaluate the suitability of the framework of Biljecki et al. [2016] for BIM-derived building models, we consider the output of two different BIM shell extractors according the existing framework. These two extractors are the IFC_BuildingEnvExtractor [van der Vaart, 2023b] and BIMShell [Diakité, 2023]. Both extractors are still in development but are in their current state able to extract high quality outer shells from most IFC-files.

1.2 The Software Tools

The IFC_BuildingEnvExtractor [van der Vaart, 2023b] extracts multiple differently abstracted LoD shells from IFC-files. The classification/abstraction of the shells has been developed in such a way that the output complies as much as possible with the LoD specification of [Biljecki et al., 2016]. The possible number of extractable shells depends on the quality of the input model, see fig. 4. The tool defines three levels of extraction where the extraction methods are similar per level. Low-level shells (LoD0.0 & 1.0) are generated by approximating a smallest bounding box around the input model. These shells can be extracted from any valid IFC-file regardless of how well it is constructed. Mid-level shells (LoD0.2, 1.2, 1.3 & 2.2) are generated by isolating the roof structure of the input model and either projecting or downward extruding it. Due to this process being based on the roofing structure, an accurate output requires an input BIM model that has the roof well modelled. High level shells (LoD3.2) are generated from the surfaces of objects that together construct the outer envelope of the BIM model. These shells can only be extracted accurately if the objects constructing the outer shell are well constructed in the original BIM model.

The tool only considers a subset of object types available in the IFC-format. When the default settings are used only the space dividing objects [van der Vaart, 2022] are used for the extraction of the shells. These space dividing objects are: IfcBeam, IfcColumn, IfcCovering, IfcCurtainWall, IfcDoor, IfcMember, IfcPlate, IfcRoof, IfcSlab, IfcWall and IfcWindow objects. Additionally, if the user desires additional object types to be included they are able to set this manually when running the tool. Aside from this customisation, the tool requires limited user input: an input path, output path, and voxel size. The voxel size is used to generate a voxel grid to roughly filter the objects that are being processed. This parameter has usually negligible effect on the end result, but fine tuning it to certain types of buildings can improve computation speed.

BIMShell is another tool that also extracts building shells from BIM files, but currently it does not constrain itself to any LoD specification, see fig. 5. Its main purpose is to reduce the size of the original BIM model and to automatically remove internal elements, while preserving as much as possible of their external appearance. As such, it is mainly meant for visualisation and applications that do not need more than the visual resemblance of a building model. The current version supports IFC and several other geometric and CAD formats as input (obj, 3ds, fbx, etc.). It produces two shells: a *raw shell* that corresponds to a collection of the exterior faces of the input model de-

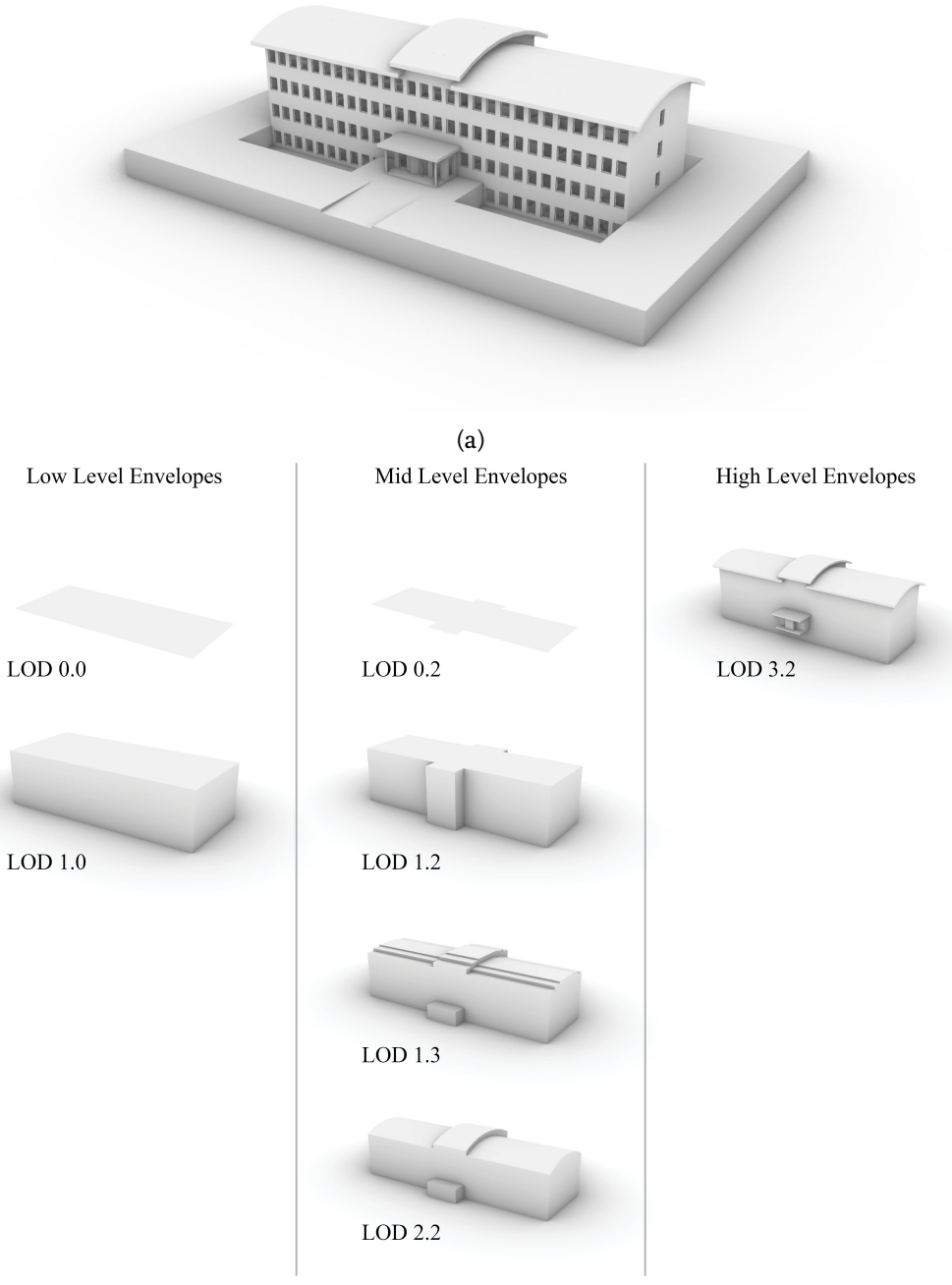
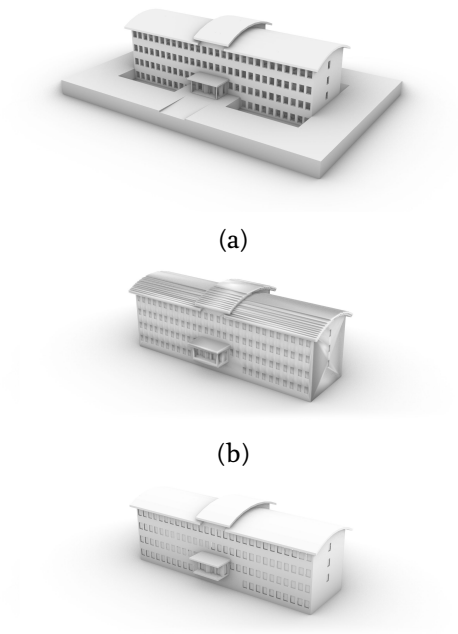


Figure 4: The possible extracted shells of the IFC_BuildingEnvExtractor are categorised in three different levels of quality. Low level shells are accurately created from any quality model. Mid level shells are only accurately created from models that have well defined roofing structures. High level shells are only created from models that have well defined objects that construct the outer shell. All shells are generated from the model in Fig. 4a



(a)

(b)

(c)

Figure 5: All the possible extracted shells of BIMShell. 5b is a voxelized shape. 5c is a collection of exterior extracted faces. The shells are generated from the model in Fig. 5a

tected through a ray tracing process, and a *voxel shell* that is, as its name suggests, a watertight shell built from a voxelization of the input. The few parameters required by the tool include an error tolerance corresponding to the resolution of the voxel grid used. It does not assume any condition and it does not require any constraint on the input models except that gaps that are larger than the tolerance are likely to cause the production of false shells (e.g. the output includes indoor faces).

2 Methodology

The evaluation of the LoD framework for BIM-derived building models is done by comparing output models created by the two software tools to models defined in the current framework. Biljecki et al. [2016] give a clear collection of rules to which every LoD has to adhere which makes it easy to see if the outputted model can be classified as one of the specified LoDs. This enables us to determine if an output model does not comply with the framework, why it does not comply, and how this can be resolved (by either addressing issues in the shell extractors' output or in the framework itself).

The primary evaluation that was done is testing a set of output models against the different rules per LoD. Figure 6 shows the followed rules as described in Biljecki et al. [2016]. When considering the different LoDs of the existing framework, the written rules were followed as much as possible instead of interpreting the rules from the visual representation of the framework (Fig. 3). This is done to reduce subjectivity in the classification process.

This evaluation can be easily executed with the output of the IFC_BuildingEnvExtractor because its output models are already classified in LoDs according to the framework. This is not the case for BIMShell. Therefore, prior to testing BIMShell's output to the LoD rules, we first needed to define the most likely LoD of the output. This is done by testing the BIMShell output to the rules of every LoD in the framework. The output is considered the LoD with which it complies the most. Often it does not comply with all the rules related to the LoD it is classified in, as will be seen further.

Biljecki et al. [2016] state that the 3D LoD shells are volumetric. This suggests that the 3D shapes are required to be valid watertight solids. To evaluate if the models are watertight solids, two different approaches were taken. Firstly, the output models were evaluated manually/visually to see if there were any glaring issues present. Secondly, the output models were checked by a small c++ program that searches for matching edges to guarantee water-tightness. This program relies on the CJT library [van der Vaart, 2023a] to open CityJSON files and OpenCASCADE library [Open Cascade, n.d.] to open OBJ files and check if the edges have neighbours.

The settings used for IFC_BuildingEnvExtractor in all the evaluations were the default settings with a

Requirements	Refined levels of detail															
	0.0	0.1	0.2	0.3	1.0	1.1	1.2	1.3	2.0	2.1	2.2	2.3	3.0	3.1	3.2	3.3
Individual buildings		•	•	•		•	•	•	•	•	•	•	•	•	•	•
Large building parts (>4 m, 10 m^2)		•	•	•		•	•	•	•	•	•	•	•	•	•	•
Small building parts, recesses and extensions (>2 m, 2 m^2)			•	•			•	•		•	•	•	•	•	•	•
Top surface(0)			S	M	S	S	S	M								
Explicit roof overhangs (if >0.2 m)												•		•	•	•
Roof superstructures(1) (larger than 2 m, 2 m^2)											•	•	•		•	•
Other roof details (e.g. chimneys >1 m)													•		•	•
Openings(2) (>1 m, 1 m^2)													R	W	•	•
Balconies (>1 m)														•	•	•
Embrasures, other façade and roof details, and smaller windows (>0.2 m)																•

Figure 6: Summary of the LoD framework rules by Biljecki et al. [2016].
(0) Applicable only to LoD0.y and LoD1.y: S—Single top surface; M—Multiple top surfaces if the difference in height of the extruded building elements is significant (larger than 2 m).
(1) It includes dormers and features of comparable size and importance (e.g. very large chimneys).
(2) R—only openings on roofs; W—only openings on walls. In R, openings on dormers are not required

voxel size of 1 by 1 meter. The settings used for BIMShell in all the evaluations were a 0,2m tolerance with a detailed raw shell look.

The input IFC models that have been used were picked to represent different building shapes while at the same time being fairly small and simple. This allows us to evaluate different building shapes while excluding potential issues with the created shells caused by exceptional cases that the software can not deal with. The evaluation assesses how the output models fit in the LoD framework. It does not evaluate the performance of the software tools. All models are openly available online, or can be easily recreated when desired.

An overview of the models can be found in Table 1. In the Appendix a visual representation of the models is included.

3 Results

Following the described method, the results of these analyses are split in two parts. The first part covers the fitting of the output of BIMShell into a suitable LoD from the framework of Biljecki et al. [2016], see Table 2. The second part tests how well the output of BIMShell and the IFC_BuildingEnvExtractor fit to the classified LoD as defined in the framework, see table 3, 4 and 5. This part also evaluates if the output of both tools are solid 3D shapes, see Tables 6 and 7.

3.1 Fitting of the BIMShell Output

We can see in Table 2 that both the raw shell and the voxel shell output of BIMShell comply with all the rules defined by Biljecki et al. [2016] for LoD3.3. This would suggest that both could be classified as LoD3.3. There is however one nuance that challenges this conclusion. The raw shell output of BIMShell follows the shape of the building very closely and does not exclude any detail. It also complies well with the description of LoD3.3: "an architecturally detailed model ... that contains features of size larger than 0.2 m, including embrasures of windows (i.e. making windows 3D), awnings and similar features of comparable size." Biljecki et al. [2018]. However, the voxel shell does not closely follow the shape of the building, nor its roofing structure. It results in a shape that has multiple flat top surfaces that represent the roofing structure but is not identical to it. So, while the raw shell could be considered LoD3.3, the voxel shell cannot. The way the roofing structure is modelled in the voxel shell output would presume it to be more closely related to LoD1.x. However, the rest of the shell is considered too detailed to fit with LoD1.x. In the end we were unable to find a suitable LoD in the framework and are thus unable to further analyse the framework's suitability of the voxel shell output of BIMShell. Supporting voxel shells in the BIM-based LoD framework, might therefore require the addition of yet another LoD.

3.2 Evaluation of the Framework Fit for the Output

Table 3 shows if the output of BIMShell fully fits in the framework. Due to the challenging classification of the voxel shell we only evaluated the fit of the raw shell to the framework's rules. It can be seen that when considering all the set rules it can not be properly considered LoD3.2 according to the framework. Table 6 summarises the main issue, i.e. none of the raw shells are closed solids. Gaps can exist, which mainly occur at places where small details were present in the input model e.g., windows and doors. If we exclude the volumetric/solid requirements the framework is much better suited to define the output of BIMShell.

Table 4 shows if the output of the IFC_BuildingEnvExtractor fully fits in the framework. As can been seen, many of the output models do not completely fit in the framework. The exception are two of the three evaluated office buildings. These two models show an output that is fully

Table 1: Summary of the used input BIM models. * object count done with BimVision. ** 1 = freestanding house, 2 = terraced house, 3 = office building

Input Model Name	Object Count*	Building Type**	Storeys	Has Overhang (Y/N)	Source
AC20-FZK-Haus	105	1	2	✓	KIT [n.d.]
AC20-Institute-Var-2	896	3	3	✓	KIT [n.d.]
AC-20-Smiley-West-10	972	2	4	✓	KIT [n.d.]
RE16_E3D_Building	552	3	2	×	IBPSA [2021]
DigitalHub	775	3	3	×	RWTH E3D [2022]
RAC_basic_sample	450	1	2	✓	Revit

Table 2: Evaluation of the classification of the output of BIMShell. S—Single top surface; M—Multiple top surfaces; P-Precisely followed top faces

Requirement	raw shell	voxel shell
Individual buildings	✓	✓
Large building parts (>4 m, 10 m^2)	✓	✓
Small building parts recesses and extensions (2 m, 2 m^2)	✓	✓
Top surface(0)	P	M
Explicit roof overhangs (if >0.2 m)	✓	✓
Roof superstructures (1) (larger than 2 m, 2 m^2)	✓	✓
Other roof details (e.g. chimneys >1 m)	✓	✓
Openings(2) (>1 m, 1 m^2)	✓	✓
Balconies (>1 m)	✓	✓
Embrasures, other façade and roof details, and smaller windows (>0.2 m)	✓	✓

Table 3: Evaluation of the fit of the BIMShell raw shell output models to the existing LoD framework. * disregard of the volumetric/solid requirement

Input Model Name	Compliant LoD3.2 (Y/N)	Compliant LoD3.2* (Y/N)
AC20-FZK-Haus	✗	✓
AC20-Institute-Var-2	✗	✓
AC-20-Smiley-West-10	✗	✓
RE16_E3D_Building	✗	✓
DigitalHub	✗	✓
RAC_basic_sample	✗	✓

Table 4: Evaluation of the fit of the IFC_BuildingEnvExtractor output models to the existing LoD framework

Input Model Name	Compliant LoD1.0 (Y/N)	Compliant LoD1.2 (Y/N)	Compliant LoD1.3 (Y/N)	Compliant LoD2.2 (Y/N)	Compliant LoD3.2 (Y/N)
AC20-FZK-Haus	✗	✗	✗	✓	✗
AC20-Institute-Var-2	✗	✗	✗	✓	✗
AC-20-Smiley-West-10	✗	✗	✗	✓	✗
RE16_E3D_Building	✓	✓	✓	✓	✗
DigitalHub	✓	✓	✓	✓	✗
RAC_basic_sample	✗	✗	✗	✓	✗

consistent with significantly more LoDs than the other input models.

None of the output models are compliant with the LoD3.2 shell. However, when we ignore the wall and roof opening requirements of the framework (see subsection 4.4), the LoD3.2 output does fit the framework very well, see table 5 and 7.

4 Discussion & Evaluation of the Framework

The evaluation of the shell output highlighted a set of issues with the current framework when attempting to define BIM-derived building models in an unambiguous way. These issues are addressed and discussed in the following subsections.

4.1 Voxels

The fitting of BIMShell's voxel shell highlights the first issue of the framework of Biljecki et al. [2016] for BIM derived models: it does not mention voxelized shapes. This makes it impossible to classify the voxel shell output of BIMShell to a certain LoD within the framework. This shell was considered to be deviating so far from the defined LoDs that it could not be fit in a meaningful manner and this shell's classification was discarded.

The absence of voxelization in the framework could be seen as an issue. Voxelization can be utilized to create watertight geometries which can even be used to approximate shapes from incomplete and/or faulty BIM data sources [Huang et al., 2020, Mulder, 2015]. IFC-models have shown to often contain geometric issues that make it difficult for envelope extractors to generate high LoD models. These issues range from misplaced objects [van der Vaart, 2022], to missing or corrupted geometry [Krijnen et al., 2020, Arroyo Ohori et al., 2018]. With the help of voxelization, shell shapes could still be approximated when these issues occur. However, the fact that voxelization and voxel related shapes are a solution for creating shells from erroneous input models does not mean that these outputted shells are desirable in further analysis. The effects of these models on analysis (and therefore the need of such models in urban applications) has to be examined first, before it can be decided if these models should have their place in city scale models and are important to include in the LoD framework.

4.2 Non-Watertightness

The raw shell output of BIMShell follows the framework geometrically quite well. The major exception is that the raw shell is not a volumetric solid, but a collection of surfaces. This is however an issue that cannot be attributed solely to the framework. A shell is needed to separate the outside from the inside of an building. A somewhat closed shape is also required to enable geometrical analysis in 3D such as volume-calculation and CFD-modeling.

BIMShell also occasionally outputs models that have seemingly closed shapes but include some residual surfaces of interior objects. The outer shell of the shape is closed, or close to being closed, but can not be considered solid due to these residual surfaces. This means that the boundary between interior and exterior is present, but it is not completely explicit where it lies, see fig. 7. Possibly for exterior analysis (e.g. exterior wind and sunlight analysis) these models could still be suitable. More research should be done on further usefullness of models that are not completely closed in urban applications to be definite about the inclusion of these models in the framework.

4.3 Footprints and Roof Outlines

The evaluation of the output of IFC_BuildingEnvExtractor shows that there are some ambiguities with the lower level LoD shell classification. The main issue is the use of footprints and roof outlines. Biljecki et al. [2016] imply that LoD0.x, LoD1.x and Lod2.x models are based on the footprint, while LoD0.x may

Table 5: Evaluation of the fit of the IFC_BuildingEnvExtractor output models to the existing LoD framework. The requirements related to wall and roof openings are ignored

Input Model Name	Compliant LoD1.0 (Y/N)	Compliant LoD1.2 (Y/N)	Compliant LoD1.3 (Y/N)	Compliant LoD2.2 (Y/N)	Compliant LoD3.2 (Y/N)
AC20-FZK-Haus	✗	✗	✗	✓	✓
AC20-Institute-Var-2	✗	✗	✗	✓	✓
AC-20-Smiley-West-10	✗	✗	✗	✓	✓
RE16_E3D_Building	✓	✓	✓	✓	✓
DigitalHub	✓	✓	✓	✓	✓
RAC_basic_sample	✗	✗	✗	✓	✗

Table 6: Results of the solid check for the output of the BIMShell

Input Model Name	Solid Raw Shell (LoD3.2) (Y/N)	Solid Voxel Shell (Y/N)
AC20-FZK-Haus	✗	✓
AC20-Institute-Var-2	✗	✓
AC-20-Smiley-West-10	✗	✓
RE16_E3D_Buildin	✗	✓
DigitalHub	✗	✓
RAC_basic_sample	✗	✓

Table 7: Results of the solid check for the output of the IFC_BuildingEnvEx- tractor

Input Model Name	Solid LoD1.0 (Y/N)	Solid LoD1.2 (Y/N)	Solid LoD1.3 (Y/N)	Solid LoD2.2 (Y/N)	Solid LoD3.2 (Y/N)
AC20-FZK-Haus	✓	✓	✓	✓	✓
AC20-Institute-Var-2	✓	✓	✓	✓	✓
AC-20-Smiley-West-10	✓	✓	✓	✓	✓
RE16_E3D_Building	✓	✓	✓	✓	✓
DigitalHub	✓	✓	✓	✓	✓
RAC_basic_sample	✓	✓	✓	✓	✗

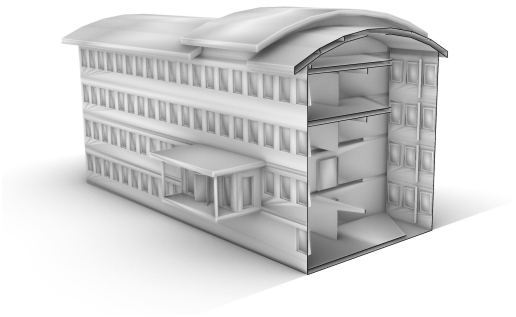


Figure 7: Section of the BIMShell output of the AC20-Institute-Var-2 model. This section shows a seemingly closed shell while still having a subset of interior faces

optionally include the roof outline. However, at LoD2.x it is stated that "When roof overhangs are not available, the walls are usually obtained as projections from the roof edges to the ground, inherently increasing the volume of the building". This exception was added due to the cost of collecting the required data to accurately determine the overhang. This exception is not mentioned in subsequent parts of the paper that presents the framework. It is thus not completely clear if a valid LoD2.x model is based on the footprint, the projected roof outline, or if both options are framework compliant. In any case, if only one is allowed it is not clearly dictated by the framework. If both are allowed it introduces even more vagueness because the user may not be aware of such a openness to this LoD and it cannot be specified which is used, the footprint or the roof outline. If the input model has a roofing structure with overhang, the geometry of the projected roof structure will differ from its footprint, see Fig. 8 and this can have an impact on the analysis in which the models are used. A solution to distinguish between both options is to include the explicit use of footprints or projected roof edges for extruded objects in the framework.

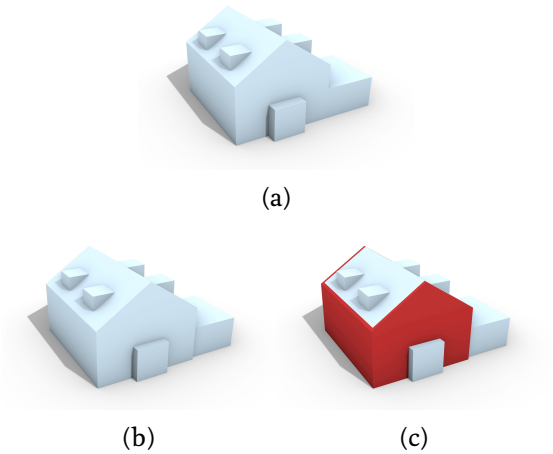
For the evaluation, we used the roof outline as base for the LoD2.x. For the LoD1.x the footprint was considered to be the base. Although the paper presenting the framework is not explicit about the flexible use of footprints or projected roof outlines in lower LoDs it seems to imply that this flexibility is not present. Due to this we presume that that lower LoDs should exclusively be based on the footprint to be considered compliant.

However, the LoD0.x and LoD1.x shells show that they would have been valid if they were allowed to be based on the roof outline instead of the footprint. Extending the reasoning of Biljecki et al. [2016] for LoD2.x, it could be said that creating LoD0.x and LoD1.x from BIM models based on only footprints would bring a larger cost with it as well. The computations to successfully create compliant shapes for footprints from BIM would cost a lot more time to execute, and might become unreliable or ineffective. It is easier to extract roof outline and extrude these downwards. Additionally the BIM model could be required to be made more precisely and robust, costing time and money. This could possibly be avoided if LoD0.x and LoD1.x were also allowed to be based on the projected roof outline.

Extending the flexibility of footprint/roof outline use, without clearly distinguishing between both, to LoD0.x and LoD1.x might make the fitting of BIM output into the framework more easy. It will however also introduce more ambiguity since it is not clear which 2D geometry has been used for the extrusion. This ambiguity is already present in practice, since LoD1.x and LoD2.x models are sometimes generated from 2D polygons that represent the projected roof outline and not the footprints e.g., 3D BAG [TU Delft and 3DGI, n.d.] while this distinction is not supported in the framework. We therefore recommend to explicitly add this refinement regarding the used 2D primitive (footprint or roof outline) to the framework.

4.4 Missing Semantic & Opening Information

The LoD3.x output of IFC_BuildingEnvExtractor does not include the geometry related to wall and roof



(a)

(b) (c)

Figure 8: Two LoD2.3 models displayed in 8a and 8b. Where 8b is a extrusion of the roofing structure and 8a is the same shape, but clipped to comply with the building's footprint. This results in the displayed in 8c highlighted shape to be removed

openings that are required by the framework. BIMShell has these surfaces present, but it does not semantically classify these surfaces.

This highlights an interesting issue with the framework for BIM-derived models. It is stated that openings have to be modelled, but it does not state how to correctly model these openings. The best option to do so is to model these openings both geometrically and semantically. In earlier work it is stated that indeed every object should be modelled both geometrically and semantically [Biljecki et al., 2014]. However, in the paper presenting the framework it is explicitly mentioned that semantic data is not used for its construction [Biljecki et al., 2016]. It is thus unclear if the openings should be modelled according to the earlier described method or if they are only required to be modelled geometrically.

Since the framework mentions it actively disregards semantics, we did so as well for the evaluation of LoD3.x output. However, if we would consider the framework to require the openings to be modelled both geometrically and semantically, both tested software tools would not generate valid output. One could reason that both the tools would then generate LoD2.3 as highest quality output because of these missing elements. But both outputs do include small roof details and balconies which are all exclusively part of an higher LoD (higher than LoD2.x).

IFC_BuildingEnvExtractor does not model the openings at all, not semantically and also not geometrically. LoD3.x models are often used for evaluations that require wall and roof openings to be properly modelled, such as heat loss estimations. However, there has been little research regarding the need for LoD3.x shapes in applications that do not require these openings. This is required to decide if LoD3.x models with detailed building elements but without openings should be supported by the framework.

5 Conclusions & Proposed Refinements

In this paper we evaluated how suitable the framework of Biljecki et al. [2016] is to unambiguously specify BIM-derived building models. This research was done by comparing the output of two BIM shell extractors with the LoDs defined in the framework: IFC_BuildingEnvExtractor and BIMShell.

The results highlight two issues that arise when attempting to fit BIM-derived building models in the framework and to define these models in an unambiguous manner. These issues also highlight some other general needs to resolve the ambiguity of the framework.

These are the definition of wall/roof openings in case of LoD3.x models and the definition of the used 2D primitive (roof outline or footprint) for LoD0.x, LoD1.x and LoD2.x models. The first issue can easily be resolved by clearly describing the rules depending on the data needs from urban applications (either always requiring openings at LoD3.x that are both geometrically and semantically modelled or

also allowing LoD3.x models containing detailed building elements without containing openings). The second issue requires to decide if both the roof outline and footprint should be allowed to be used as a base for LoD2.x models. If so, a clear and concise way of signifying which source is used has to be included. Additionally, the flexibility of roof outline and footprint use should be considered for lower LoDs as well. The IFC_BuildingEnvExtractor outputs LoD1.x shells that are based on the projected roof outline instead of the footprint. This is not framework compliant. However, roof outline based shapes will be more easy to generate from BIM-derived models, also for LoD1.x models. Additionally, roof outline based LoD1.x models are already used in practice. Apart from including the distinction of the used 2D primitive in the framework, more research is needed on the effect of using either of them in city scale analysis.

Finally, the framework does not have a place for voxelized shapes and non solid volumetric shapes. Further research is needed to see if such models are required in urban applications.

The results of this research can be used to improve the LoD framework and to adjust the shell extractors output to better comply with unambiguous definitions of building models at different LoDs. This could be a first step to standardise the conversion of BIM models at different LoDs to be used in urban applications.

Acknowledgements

This project has received funding from the European Union's Horizon Europe programme under Grant Agreement No.101058559 (CHEK: Change toolkit for digital building permit).

References

Nazmul Alam, Volker Coors, Sisi Zlatanova, and PJM Oosterom. Resolution in photovoltaic potential computation. *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, 4:89–96, 2016.

Ken Arroyo Ohori, Abdoulaye Diakité, Thomas Krijnen, Hugo Ledoux, and Jantien Stoter. Processing bim and gis models in practice: experiences and recommendations from a geobim project in the netherlands. *ISPRS International Journal of Geo-Information*, 7(8):311, 2018.

Filip Biljecki, Hugo Ledoux, Jantien Stoter, and Junqiao Zhao. Formalisation of the level of detail in 3d city modelling. *Computers, environment and urban systems*, 48: 1–15, 2014.

Filip Biljecki, Hugo Ledoux, and Jantien Stoter. An improved lod specification for 3d building models. *Computers, Environment and Urban Systems*, 59:25–37, 2016.

Filip Biljecki, Gerard BM Heuvelink, Hugo Ledoux, and Jantien Stoter. The effect of acquisition error and level of detail on the accuracy of spatial analyses. *Cartography and Geographic Information Science*, 45(2): 156–176, 2018.

André Borrmann, Jakob Beetz, Christian Koch, Thomas Liebich, and Sergej Muhic. Industry foundation classes: A standardized data model for the vendor-neutral exchange of digital building models. *Building information modeling: Technology foundations and industry practice*, pages 81–126, 2018.

Abdoulaye Diakité. Bimshell. `https://bimshell.app/`, 2023. Accessed: 04-07-2023.

C García-Sánchez, S Vitalis, I Paden, and J Stoter. The impact of level of detail in 3d city models for cfd-based wind flow simulations. *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, pages 67–72, 2021.

Andreas Geiger, Joachim Benner, Karl-Heinz Häfele, and Veit Hagenmeyer. Thermal energy simulation of buildings based on the city-gml energy application domain extension. In *BauSIM2018–7. Deutsch-Österreichische IBPSA-Konferenz:*

Tagungsband. Hrsg.: P. Von Both, pages 295–302, 2018.

Jingwei Huang, Yichao Zhou, and Leonidas Guibas. Manifoldplus: A robust and scalable watertight manifold surface generation method for triangle soups. *arXiv preprint arXiv:2005.11621*, 2020.

IBPSA. project1-wp-2-2-bim. https://github.com/ibpsa/project1-wp-2-2-bim/find/master, 2021.

KIT. Kit ifc examples. https://www.ifcwiki.org/index.php?title=KIT_IFC_Examples, n.d.

Thomas Krijnen, Francesca Noardo, Ken Arroyo Ohori, Hugo Ledoux, and Jantien Stoter. Validation and inference of geometrical relationships in ifc. In *Proceedings of the 37th International Conference of CIB W*, volume 78, pages 98–111, 2020.

David Luebke, Martin Reddy, Jonathan D Cohen, Amitabh Varshney, Benjamin Watson, and Robert Huebner. *Level of detail for 3D graphics*. Morgan Kaufmann, 2003.

DT Mulder. Automatic repair of geometrically invalid 3d city building models using a voxel-based repair method. Master's thesis, Delft University of Technology, 2015.

Francesca Noardo, Teng Wu, Ken Arroyo Ohori, Thomas Krijnen, and Jantien Stoter. IFC models for semi-automating common planning checks for building permits. *Automation in Construction*, 134: 104097, feb 2022. ISSN: 0926-5805.

Open Cascade. Open cascade technology collaborative development portal. https://dev.opencascade.org/, n.d. Accessed: 20-05-2023.

Giuseppe Peronato, Stéphane Bonjour, Jérémie Stoeckli, Emmanuel Rey, and Marilyne Andersen. Sensitivity of calculated solar irradiation to the level of detail: insights from the simulation of four sample buildings in urban areas. In *Proceedings of PLEA 2016, 32th international Conference on Passive and Low Energy Architecture*, number CONF, 2016.

RWTH E3D. Digitalhub. https://github.com/RWTH-E3D/DigitalHub, 2022.

Jantien Stoter, Ravi Peters, Tom Commandeur, Balázs Dukai, Kavisha Kumar, and Hugo Ledoux. Automated reconstruction of 3d input data for noise simulation. *Computers, Environment and Urban Systems*, 80:101424, 2020.

TU Delft and 3DGI. 3d bag viewer. https://3dbag.nl/en/viewer, n.d.

Jasper van der Vaart. Automatic building feature detection and reconstruction in ifc models. Master's thesis, TU Delft, 2022.

Jasper van der Vaart. Cjt. https://github.com/jaspervdv/CJT, 2023a. Accessed: 20-05-2023.

Jasper van der Vaart. Ifc_buildingenvextractor. https://github.com/jaspervdv/IFC_BuildingEnvExtractor, 2023b. Accessed: 20-05-2023.

6 Appendix

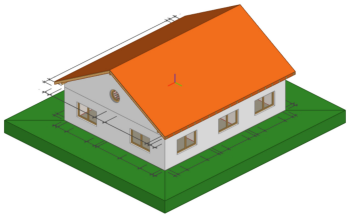


Figure 9: Visual isometric representation of the AC20-FZK-Haus model

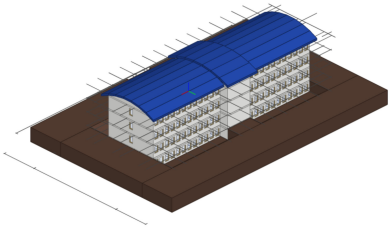


Figure 10: Visual isometric representation of the AC20-Institute-Var-2 model

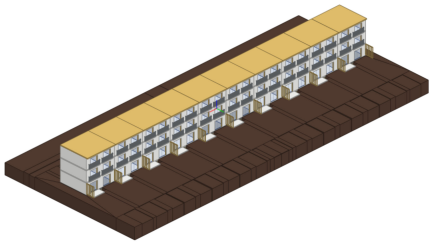


Figure 11: Visual isometric representation of the AC-20-Smiley-West-10-Bldg model

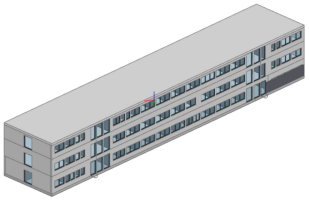


Figure 12: Visual isometric representation of the RE16_E3D_Building_2x3_Testversion model

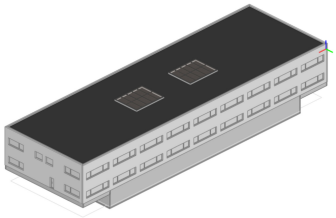


Figure 13: Visual isometric representation of the FM_ARC_DigitalHub model

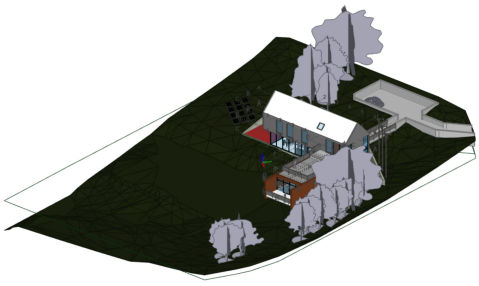


Figure 14: Visual isometric representation of the RAC_basic_sample_project model