An IFC schema extension and binary serialization format to efficiently integrate point cloud data into building models

Citation for published version (APA):

Krijnen, T. F., & Beetz, J. (2017). An IFC schema extension and binary serialization format to efficiently integrate point cloud data into building models. *Advanced Engineering Informatics*, 33, 473-490.
<https://doi.org/10.1016/j.aei.2017.03.008>

Document license:
CC BY-NC-ND

DOI:
[10.1016/j.aei.2017.03.008](https://doi.org/10.1016/j.aei.2017.03.008)

Document status and date:
Published: 01/08/2017

Document Version:
Accepted manuscript including changes made at the peer-review stage

Please check the document version of this publication:

- A submitted manuscript is the version of the article upon submission and before peer-review. There can be important differences between the submitted version and the official published version of record. People interested in the research are advised to contact the author for the final version of the publication, or visit the DOI to the publisher's website.
- The final author version and the galley proof are versions of the publication after peer review.
- The final published version features the final layout of the paper including the volume, issue and page numbers.

[Link to publication](#)

General rights

Copyright and moral rights for the publications made accessible in the public portal are retained by the authors and/or other copyright owners and it is a condition of accessing publications that users recognise and abide by the legal requirements associated with these rights.

- Users may download and print one copy of any publication from the public portal for the purpose of private study or research.
- You may not further distribute the material or use it for any profit-making activity or commercial gain
- You may freely distribute the URL identifying the publication in the public portal.

If the publication is distributed under the terms of Article 25fa of the Dutch Copyright Act, indicated by the "Taverne" license above, please follow below link for the End User Agreement:

www.tue.nl/taverne

Take down policy

If you believe that this document breaches copyright please contact us at:

openaccess@tue.nl

providing details and we will investigate your claim.

An IFC schema extension and binary serialization format to efficiently integrate point cloud data into building models

Thomas Krijnen¹ Jakob Beetz¹

Abstract

In this paper we suggest an extension to the Industry Foundation Classes (IFC) model to integrate point cloud datasets. The proposal includes a schema extension to the core model allowing the storage of points, either as Cartesian coordinates, points in parametric space of associated building element surfaces or as discrete height fields projected as grids onto building elements. To handle the considerable amounts of data generated in the process of scanning building structures, we present intelligent compression approaches combined with the Hierarchical Data Format (HDF) as an efficient serialization and an alternative to clear text encoded ISO 10303 part 21 files. Based on prototypical implementations we show results of various serialization options and their impacts on storage efficiency.

In this proposal the deepened semantic relationships have been favoured over compression ratios. Nevertheless, with various near-lossless layers of compression and binary serialization applied, a compression ratio of up to 67.7% is obtained for a building model with integrated point clouds, compared to the raw source data. The binary serialization is able to handle hundreds of millions of points, out of which specific spatial and semantic subsets can rapidly be extracted due to the containerized hierarchical storage model and association to building elements. The authors advocate the use of binary storage for sizeable point cloud scans, but also show how especially the grid discretization can result into usable points cloud segments embedded into text-based IFC models.

Keywords: Point clouds, Industry Foundation Classes, Building Information Modelling, HDF5, Binary storage, Serialization

1 Introduction

A point cloud model is a large set of unstructured 3D points that carries detailed geometric and appearance information of scanned objects in an implicit manner acquired in bulk in automated processes. Information in the Industry Foundation Classes (IFC) (Liebich, 2013) exchange model and file format on the other hand, represents detailed explicit knowledge and design intent and is most often modelled manually.

In IFC, a building is described as a semantically rich assembly of building elements, which can be represented by various forms of geometry, including 2D and 3D line drawings, Constructive Solid Geometry (CSG), Boundary Representations (BRep) and tessellated meshes. However, these three-dimensional representations are just one of the many aspects conveyed in an IFC model. In addition, attributes related to thermal or acoustic performance, costing or intended use of spaces etc. can be added.

In many common data formats for the storage of point cloud data, such as E57 (ASTM International, 2011) and PCD (Rusu and Cousins, 2011), metadata is attached to individual data sets. This metadata for example includes scanner

¹Eindhoven University of Technology (TU/e), The Netherlands

Figure 1: Colour-coded height map of the ceiling structure in the “Haus 30” dataset used throughout this paper. Deviations downwards are colour-coded in red. In the actual model, these deviations are stored as semantically precise length measures and not as a bitmap texture. Storing points as deviations from building elements renders the semantics of the points explicit and results in a favourable compression ratio. Note how the ceiling bulges downwards in some of the rooms. The slabs in this model cover the building storeys in their entirety, hence the aggregation of several rooms in a single ‘grid’.

positions or weather conditions that are perceived during the scan. From the acquisition process, the point data itself contains no grouping, decomposition or other information that relates the points to the semantic meaning of the real-world object that was scanned. In subsequent processing steps such labels are often added to the points. Several exchange formats, such as LAS, have options to store labels along with the points.

The magnitude of the data which is typically found in point cloud data sets and IFC model populations can be dramatically different for the two file types. A meaningful IFC file can have file sizes in the order of a few megabytes, if geometrical representations and property values are properly reused and especially when the file contains implicit, parametric, rather than tessellated geometry. Depending on the amount of detail and precision, point cloud scans can easily amount to gigabytes of data. Despite the larger size, due the uniform structure and explicit nature, point clouds can typically be more immediately explored than IFC building models, for which the boolean operations and implicit geometries need to be evaluated prior to visualisation.

The need for a unified and harmonized storage model of the two data types is observed in literature (Li et al., 2008; Yamaoka et al., 2013; Shih et al., 2004; Golparvar-Fard et al., 2011). Yet, the authors acknowledge that other use cases will exist in which a deep coupling between building models and point clouds is unnecessary or even undesirable. This paper presents an extension to the IFC schema by which an open and semantically rich standard arises. A number of advantages can be identified for the use and creation of such integrated datasets in a single and standardized format:

- (a) Being able to store a single file, as a harmonized storage model of semantically rich IFC data and detailed geometrical point cloud data, brings administrative

benefits by not having to implement, maintain and synchronise two distinct data carriers of the same construction project.

- (b) More importantly, the ability to document, structure and annotate the point cloud using the semantically rich constructs of the IFC schema allows the focused search, selection and use of point clouds. Subset clusters of points can be identified to describe the surface of specific building elements. This relation can be semi-automatically detected by the geometrical proximity of the points to the geometry associated to the respective element. This relation assigns meaning to the set of points, i.e. what kind of element class they describe, and by descending into the relationships of the IFC population model, users can relate additional information like material properties to these points.
- (c) In addition to providing additional semantics to the points, the subdivision structure and classifications present in an IFC file can be used to efficiently retrieve localized subset of the point cloud of interest.
- (d) Point cloud data enriches the information on the elements found in the original, explicitly modelled IFC data by providing insight into its exact placement, shape and detailed material irregularities and geometrical deformations. As such, they can constitute an effective mechanism to verify and compare what has been built to what was planned and designed.
- (e) By iteratively repeating point cloud scans of the elements, the behaviour of such elements over longer periods of time can be assessed and potential structural risks can be extrapolated. Especially if the format allows for multiple scan structures side-by-side, and the points are expressed as deviations from their associated building elements, the complexity of such computations is reduced and the likelihood of discovering anomalies might be increased. The coupling with building topology, load cases and detailed material semantics can be used to further quantify and analyse the measured deviations. In Figure 1, a graphical illustration is given of how the work in this paper can be used to assess the structural health of elements.
- (f) The combination of parametric IFC geometry and accurately measured point cloud segments can be used to precisely document the geometrical appearance of building elements. In particular for historical monuments with detailed ornaments, such hybrid modelling techniques are of interest.
- (g) IFC is expanding its domain to encompass infrastructural works, for example signified by the recently ratified *IfcAlignment* extension for linear infrastructural elements. For these domains, survey data, for example airborne point clouds, are a vital element in their engineering and design processes.
- (h) Lastly, the life span of construction and infrastructure works is significant. To guarantee interoperable use of the information over longer periods of time, the

use of open standards, such as IFC, can be beneficial.

At the same time, some obvious drawbacks of aggregating heterogeneous data into a single data format can be identified. These are provided below along with a short discussion.

- (a) The granular semantic interlinking and unified storage model negatively affects the ease with which data can be updated. The use of this aggregated data format is designed with a ‘*write once, read many*’ paradigm in mind. Appending new situations as the building evolves is explicitly enabled by allowing simultaneous side-by-side associations to multiple point cloud scans.
- (b) The fact that existing software for visualizing, analyzing or manipulating point cloud datasets cannot directly operate on the unified data model is a clear disadvantage, but on the other hand, by relying on an open standard for binary storage (HDF5) that is readily understood in scientific computing environments, such as *Python* and *MATLAB*, advanced analysis scenarios on the combined data format are enabled with relative ease.
- (c) The increase in file size, compared to the original IFC file, impacts the way such combined building models can be shared and transmitted. Note however that an HDF5 file is not an atomic blob. It is very much akin to a file system in itself. The standard library comes with utilities to extract and merge datasets from multiple sources and even allows to *mount* files into others, to access a set of files through a single hierarchy.

This paper proposes an extension to the IFC schema to harmonize point cloud data and semantic building models. By associating point cloud segments to building elements and expressing them in local coordinate systems of building element surfaces, the objectives above are met and a favourable compression ratio is obtained, especially when storing the augmented IFC model in the novel HDF5-based binary serialization format, out of which spatial and semantic subsets can easily and rapidly be extracted.

In the first section of this paper, an overview of the state of the art is provided, firstly an overview of applications of superimposing building models and point clouds in practice (Section 2.1), then limitations on point cloud storage and compressions (Section 2.2), lastly on the technical means of current schemas (Section 2.3.1), software solutions (Section 2.3.2) and serialization formats (Section 2.3.3).

The fundamental concepts behind the proposed IFC schema extension are outlined in Section 3, which are materialized in a listing of the cardinal entities of the schema in Section 4. To counter the inherent limitations of the prevalent text-based serialization forms of IFC, discussed in Section 2.3.3, an efficient binary serialization format for IFC and its implications on storing large point clouds is

presented in Section 5. A prototypical implementation to generate and test datasets that conform to the extended schema is explained in Section 6. The paper closes with a conclusion and pointers for future research.

2 State of the art

2.1 Superimposing IFC and point clouds

Despite the fundamental differences in point clouds and IFC building models, there is a growing interest in the use of point cloud data in the architecture engineering and construction (AEC) industry. Point clouds are a viable means to precisely document a structure as it is physically built, rather than as it has been designed, and to detect differences between these 'as-planned' and 'as-built' states is interesting for many use cases. Superimposing, reconstructing, comparing and integrating 'as-planned' and 'as-built' models of buildings and other structures has been the subject of many research and development efforts (Bhatla et al., 2012; Pu and Vosselman, 2009). Practical use cases include the monitoring of construction processes (Bügler et al., 2014; Yang et al., 2015; Teizer, 2015; Braun et al., 2015) or indoor navigation (Khoshelham and Elberink, 2012). The increasing availability of affordable acquisition technology such as aerial and terrestrial laser scans (Pu and Vosselman, 2009; Tang et al., 2010), image based approaches (Bhatla et al., 2012) or combinations (Brilakis et al., 2010) as well as the necessary processing and storage resources have contributed to this spike in interest amongst practitioners and researchers alike (Braun et al., 2015; Bügler et al., 2014; Teizer, 2015; Yang et al., 2015). Especially in the domain of digital preservation there is a confluence in the desire to document the design intent and intended use along with the physical edifice as it is delivered and changes over time (Dietze et al., 2013).

A typical use case for such representations are the often intricate details of historical buildings including detailed plaster works, for which a definition using parametric surfaces is beyond what is typically the scope of the modelling effort in IFC. Digitally remodelling such ornaments is a painstaking process and common BIM authoring tools are not equipped for this process. Capturing such geometrical detail as a local point cloud can therefore be a good compromise between geometric and semantic precision. The success of hybrid modelling techniques for documenting historical artefacts has been shown in the cultural heritage domain, for example in Alshawabkeh and Haala (2004), but to date such workflows mostly operate on photogrammetric models or meshes and not on detailed engineering models, such as IFC. Also in the field of structural health monitoring, the analysis of point clouds is employed to find local deformations beyond critical values that impact the structural integrity of the edifice (González-Aguilera et al., 2008; Gordon and Lichti, 2007; Teza et al., 2009).

2.2 Point cloud visualization and compression

Efficient visualization and analysis of point clouds require the usage of spatial indexing structures like Octrees (Meagher, 1982) or kd-trees (Bentley, 1975). Such acceleration structures divide the point clouds into hierarchical cells that contain subsets of the overall point cloud. During visualization only cells visible to the observer have to be rendered. As point clouds are inherently unstructured, acceleration trees are not necessarily tailored semantically to what the points represent: The cell borders rather arbitrarily cut through semantic structures such as architectural spaces. Conversely, explicitly modelled objects can contain detailed semantic decomposition structures. For example, in the case of the IFC standard, there is a subdivision structure by which a building is decomposed into individual building storeys and spaces. To the authors' best knowledge, no research has been conducted on using such explicitly modelled decomposition structures from existing external sources to establish the location of the splitting planes in point indexing structures, and hence to form boundaries tailored to the scene at hand.

The introduction of various Levels Of Detail (LOD) is an additional approach to address visualisation of large datasets. For point cloud segments that are further away, or otherwise not of interest to the observer, a coarser representation can be loaded and visualised to conserve resources. The aim is that, when the rendering function is terminated for a frame, a representative and visually relevant subset of points has been drawn. Methods to construct such levels are for example given in Goswami et al. (2013). Crucial aspects are forming balanced tree representations with evenly-sized nodes to be able to swap nodes in memory. Based on a prior point labelling step visual integrity can be better maintained (Richter et al., 2015). In Wimmer and Scheiblauer (2006) a visualisation approach is presented without the need for expensive pre-processing steps. In order to maintain a responsive application, typically a rendering budget is defined that controls how much time can be spent on the rendering of points.

Driven by the increasing resolutions of 3D acquisition devices, researchers have investigated possibilities to efficiently compress point clouds. Universal entropy-based methods like Ziv and Lempel (1978) that perform well on data like text are less suited for point cloud compression as they cannot detect and exploit the hidden underlying geometric structures in the point cloud. Therefore a large branch of research is directed at shape-proxy-based compression methods, for example as outlined in Digne et al. (2014); Golla et al. (2014); Schnabel et al. (2008). In such approaches, the key to an efficient compression is to find shape proxies that can be related to a large number of measured points. By describing the position of an individual point in relation to the surface of a local proxy rather than within a local or global coordinate system, points do not need to be stored as triplets of $\{x, y, z\}$ coordinates, but instead as a height field projected onto the shape proxies. This reduces the dimensionality of a point to only a single z offset component that

specifies deviation from the surface at a specific known grid point. The other two components are implicitly known from the intersection points of the grid axes. The points will need to be ordered according to the way the axes are defined.

For airborne laser scans, **LAS** is the de facto exchange format and is maintained by the ASPRS organization². It features a compressed variant called **LAZ**. By combining knowledge of the **LAS** file format structure and entropy-based coding, compression ratios of 75-93% can be established. As opposed to generic compression mechanisms, the **LAZ** file format enables partial decompression and overlaying spatial indexing structures (Isenburg, 2013).

The conducted experiments documented in this paper show that a semantic coupling of point clouds and IFC models can be beneficial for efficiently visualizing and intelligently compressing point clouds. The IFC file format is not a likely candidate for a general purpose point cloud storage format. This paper focuses on the use cases, benefits and implications of the combined data model and not solely point cloud compression. Therefore, the rationale of harnessing the IFC decomposition structure and building element surfaces is outlined in this section, but for these reasons the results of the conducted experiments are not compared to general purpose state of the art point cloud visualization and compression techniques. Similarly, using the IFC building model surfaces as input for existing shape-proxy-based compression methods (Digne et al., 2014; Golla et al., 2014; Schnabel et al., 2008) and compare their resulting compression ratios to surfaces fitted through the data, is something not investigated in this paper.

2.3 Technical means for overlaying point clouds on BIM models

Effort on the intersection of BIM and point clouds centres for a large part around constructing geometric solid models from point clouds to capture as-built models, as discussed in Tang et al. (2010); Ochmann et al. (2016). As such, a comprehension of an as-built model can be succinctly expressed. However, these approaches provide an approximation of the measured points. The unaltered and unbiased original reference that the points constituted is lost. Furthermore, when overlaying as-planned and as-built models, comparing distances between points and a surface can be easier than comparing two surfaces (Shih et al., 2004).

It is noted that an integrated representation of as-planned and as-built models is attractive for subjects such as construction progress monitoring and quality control (Golparvar-Fard et al., 2011), but, as of now, reconciling data into a shared format typically entails finding a file format that is a common denominator in terms of the semantic explicit BIM geometry and large point cloud datasets. This can result in a format that offers a limited number of features, such as VRML in a case study by Gordon et al. (2003), that has neither means of retaining the BIM semantics nor is it tailored to efficiently capturing and retrieving large point clouds.

²<https://www.asprs.org/committee-general/laser-las-file-format-exchange-activities.html>

The remainder of this section presents limitations on existing storage formats, software and serialization formats.

2.3.1 Existing entities in the IFC schema

In version 4 of the IFC specification (Liebich, 2013) the `IfcCartesianPointList3D` entity has been added to the schema. In principle this allows the representation of point clouds such as acquired from laser scans directly into a model population as an unbounded, two-dimensional sequence of rational numbers, typed as `IfcLengthMeasure (LIST [1:?] OF LIST [3:3])`. However, this approach has several limitations:

- (a) there is no possibility to register additional per-point data channels that are prevalent in other point cloud formats, such as colours and normal vectors
- (b) there is no mechanism to store different levels of detail in parallel to allow an efficient handling during processing and visualization (e.g. high resolution details for selected parts of building components where defects or intricate structures exist)
- (c) the lack of decomposition and association to building elements impedes efficiently querying for missing or deviating point cloud coverage
- (d) metadata pertaining to the capturing process such as humidity, temperature, capturing date etc. cannot be stored in a standardized fashion for whole scans or clusters

To capture point cloud data as height maps (See Section 2.2) in the current IFC standard, another approach would be the use of bitmap structures already present in the schema. These are the `IfcPixelTexture`, `IfcImageTexture`, `IfcBlobTexture` entity definitions, representing pixel information as hexadecimal `BINARY` fields, external references or compressed blobs respectively. In addition to the limitations listed above for `IfcCartesianPointList3D`, the obscure semantics of using textures for alternative purposes is unappealing, as, for example, it becomes hard to separate textures representing point clouds from textures representing conventional surface colours and there would not be an explicit semantic construct to map pixel colours to a location in Cartesian space. To date, the support for textures in IFC implementations is limited. The IFC2x3 version of the schema contains a notable flaw, which is addressed in version IFC4. The main attribute `IfcBlobTexture.RasterCode` had been wrongfully declared to be of type `boolean`³, rather than `binary`, rendering it unable to comprehend any meaningful bitmap data. Embedding large blobs of binary data into IFC files can deteriorate the ease of exchanging files. Similarly, linking to external data and foreign

³<http://www.buildingsmart-tech.org/ifc/IFC4/final/html/schema/ifcpresentationappearanceresource/lexical/ifcblobtexture.htm>

image formats in order to represent e.g. height fields for image compression would complicate the exchange of such data sets.

2.3.2 Proprietary and open source software

Software exists for the registration and difference detection of point clouds and (typically mesh based) geometric structures. Examples include the open source *CloudCompare* (Girardeau-Montaut, 2011) and proprietary *Autodesk Navisworks*. To date, none of these applications use open standards to document the linkage between point cloud subsets and the building model, however for the purpose of deviation analysis, CloudCompare does offer serializing meshes and point clouds together in a single file format along with manually aligned transformation matrices and mesh-point distances (Figure 2). Other than transformation matrices, no association is stored, furthermore, in order to be opened in CloudCompare, a BIM model needs to be converted into a geometrical mesh format, such as Wavefront OBJ, which has no options to retain semantics from the BIM model.

Figure 2: The process of overlaying point clouds and meshes in *CloudCompare*. Mesh-point distances can be visualized in a histogram. The bins and frequencies can be exported for further analysis, the actual per-point data channels are only available in the native file format.

2.3.3 Existing Serialization formats

To date, the most common serialization format of IFC instance models is the ISO 10303 part 21 (Step Physical File Format, SPF). It has been designed to maximize interoperability, not storage efficiency. Its clear-text encoding uses 8-bit (1 byte) values corresponding to the $\{0x21 - 0x7e\}$ range of ASCII characters. As a consequence, for example for the number "123.4567" 8 bytes are required when encoded as a character string. In binary form the same number could have been encoded in 4 bytes (32 bits) using an IEEE 754 single precision floating point representation.

On top of that, sequences of text-encoded numbers need to be separated by comma delimiters, whereas boundaries of binary number representations are known by definition of the width of their data-types. The amount of time required to parse binary data is significantly lower than clear-text counterparts as the binary equivalents map directly to registers implemented in hardware. The ISO 10303 part 28 serialization which uses XML as an underlying encoding suffers from the same issues as the part 21 and brings additional overhead and verbosity by its underlying XML syntax, which will yield even larger files.

Apart from inefficiencies in encoding scalar values, structurally, SPF also lacks capabilities such as built-in indexing for randomly accessing part of a file. An example of such random access would be to read only a certain Level Of Detail (LOD) of a point cloud from a particular room on a certain building story without having to parse the complete file in advance.

Recently, in Krijnen and Beetz (2016), the authors have proposed a binary serialization format for IFC, which is an adaptation of ISO 10303 part 26. The essential details of this serialization format are reiterated in Section 5.

2.3.4 Other means of interlinking heterogeneous data

Within the buildingSMART IFC standardization activities, considerable progress has been made on a Linked Data representation of the schema (Pauwels and Terkaj, 2016). IFC populations can be uplifted to this format, technically called RDF or *Resource Description Framework*. In principle this allows arbitrary relations from IFC instances to other elements that have a stable dereferencable location, called a URI. However, point clouds are not typically encoded in this format. One of the reasons is the fundamental issue in encoding ordered aggregates in RDF that directly stems from the nature of RDF to encode every fragment of information as a triple of *subject - predicate - object*. As a consequence, this convolutes the storage of lists as either recursive binary trees or a notation using slots with explicit numeric indices (Pauwels et al., 2015).

The IFC schema also offers means for external references to URIs by means of its `IfcDocumentReference`. This enables one to establish a link from a building element in IFC, pointing to a document elsewhere identified by a string value. The granularity of this approach is limited to integral elements in the model. The identification of individual surfaces is not possible in this approach as the connection to document references is made on the level of `IfcObjectDefinition`, a construct to which geometrical entities are not related.

Irrespective of the mechanism to create relationships, in general, referring to subsets of an external point cloud is not something trivial. The subset can either be defined as ranges of points belonging to a certain entity instance. These ranges are not expected to be contiguous, making the syntax for this elaborate. Alternatively a bitmask can be supplied which would induce a large structure for every

association to be made. On top of that, not all point cloud formats would allow for easily and rapidly extracting such slices.

3 Proposed data structure

To recapitulate, a variety of methods exists for point cloud storage, point cloud compression, serializing IFC building models and approaches of interlinking heterogeneous data formats. None of these preexisting methods explicitly facilitate the rich semantic interlinking by which (a) hybrid representations of point clouds and parametric geometries in IFC building models are feasible (b) localized and semantic subsets of point clouds can be efficiently extracted based on their correspondence to particular building elements (c) the deviation between both representations can be efficiently queried in accordance with other semantic constructs, such as spatial regions or material characteristics encoded in the building model.

To bridge the disparate approaches, in IFC and point cloud processing, different technical solutions are introduced and discussed in the following sections. On a schema level, the solution approach suggested here consists of three components, which can be layered to obtain a near optimal storage of point clouds: (a) points can be recorded either in Cartesian coordinate space or in surface-parametric space (b) the points within a bounded surface- parametric space can be discretized and interpolated into a (albeit lossy) grid-based height field superimposed on bounding surfaces of building elements such as walls, floors or ceilings (c) floating point numbers within a bounded range regarding surface $\{u, v\}$ values or offsets from the association surface can be recorded in a scaled integer notation, with a single denominator for an entire set of points.

In addition to the data structures introduced into the IFC schema itself, a second mechanism is suggested to allow the efficient processing of realistic data set sizes of several Gigabytes as they are expected in typical work-flows involving point cloud measurements and processing. In this paper we suggest the use of the Hierarchical Data Format (HDF5) (Folk et al., 2011) to allow the efficient handling of large quantities of data. This binary form of serialization is discussed in Section 5.

The proposed schema extension, fully published in an open repository⁴, follows the conventions in the existing IFC schema such as the separation of product level instances from their representations. Our proposed approach incorporates state of the art point cloud compression techniques while allowing variable precision ratios and levels of detail for different purposes or elements. The suggested approach is modelled with the prevalent serialization formats in mind, but allows a transition to a more efficient serialization in the HDF5 format.

⁴<https://github.com/DURAARK/IFCPointCloud>

3.1 Decomposition and association

Figure 3: Decomposition relationships in hybrid data sets showing the bidirectional relationships between point cloud clusters and explicit geometric shape representations of building components

The schema extension allows a semantically rich coupling of IFC and point cloud data. For this purpose, the point cloud is decomposed into a structure analogous to building elements. Relevant subsets of point clouds describing such elements are explicitly linked to those building elements. This is conceptually depicted in Figure 3.

Building elements in the IFC schema descend from `IfcProducts` and their geometrical nature consists of a pair of *placement* (location and orientation in 3D space) and a set of *representations* with individual items (descending from `IfcRepresentationItem`), typically solid volumes.

The general approach suggested in this paper follows this convention. A new entity type `IfcPointCloud` is introduced as a subtype of the abstract `IfcRepresentationItem` base definition. This allows the coherent extension of well-established modelling approaches by allowing the assignment to arbitrary populations of `IfcProducts` and its subclasses. By providing a label to indicate the Level Of Detail, different resolutions of point clouds can be stored in parallel to allow the incremental access of rough approximations to fine-grained detail when needed by end-users. Colours and surface normal vectors can optionally be stored alongside the positions of points and are captured in attributes of the `IfcPointCloud` structure. Such attributes can model surface characteristics and irregularities that are not apparent in the geometric form of the element, but their application remains somewhat speculative within this paper. The point coordinates themselves have been elaborated more in the use cases discussed. The definition of this entity in the EXPRESS modelling language is provided in Listing 2.

In addition to the geometric storage structure itself, an `IfcPointCloudElement` entity is introduced to which several `IfcPointCloud` geometries can be assigned. To achieve the objective of fine-grained decomposition the `IfcPointCloudElement` can take part in decomposition relationships. Similarly to the `IfcProxy` entity, its

Figure 4: Instantiation examples of two means of associating point cloud segments to building element data according to the proposed schema. An EXPRESS-G-like notation style is followed.

main use is that of a placeholder and grouping element. It can also be used as a virtual element that carries all point cloud structures for which no other explicit association could be identified. This includes superfluous data from furniture or drapes as well as scanner reflections in glass panes or targets outside the building such as foliage. These are the points coloured grey in Figure 6. By means of the association to an IFC building model, a labeling of the points is obtained automatically by virtue of the building element geometries. On a per case basis can be decided whether to include unassociated points. As can be seen in Listing 3 the **IfcPointCloud** entity does not declare any new explicit attributes, but by means of preexisting linking mechanisms, the metadata defined in Section 3.4 can be assigned to this entity.

To summarize, as depicted in Figure 4, depending on the use case at hand, the schema allows for two ways to reconcile IFC and point cloud data: Figure 4(a) **IfcPointCloud** representation items can be embedded as actual geometric forms of building elements alongside conventional CAD geometries if present Figure 4(b) Alternatively **IfcPointCloudElements** can be used to further qualify point cloud segments by means of additional metadata, see Section 3.4. These product level entity instances are then associated to the building elements by means of the preexisting **IfcRelAssignsToProduct** entity. This approach connects to the existing modelling paradigm in IFC of using objectified relationships (entities subtyped of **IfcRelationship**), for example to express containment, decomposition and connectivity.

The aptness of the two approaches is a function of the objectives of the association and the granularity in which IFC elements are modelled. This is best illustrated by several examples. For elements that are not typically modelled in

IFC, such as furniture, the first approach in Figure 4a is most suitable. It provides a means to “tag” point cloud segments with type labels and semantically rich relationships from IFC. The geometrical nature of such instances would then solely encompass point cloud representations. As a consequence of the relational nature of IFC, with proper classification algorithms, this would enable queries very relevant to Facility Management, e.g how many instances of a certain type of chair are located in each room. On the other hand, for elements that are adequately modelled in IFC, the approach with objectified association relationships in Figure 4b is most suitable, as the provenance information can be accurately maintained. As such it is possible to track the geometric deformation of elements over time by annotating the decomposed point clouds with time stamps. Lastly, consider the use-case of hybrid modelling of historical ornaments. A simplified definition could be provided using parametric IFC geometry, to which additional representations can be added that document the element as measured with point cloud scans. In this last case, the first approach in Figure 4a would be preferred as well, but mixing parametric and point cloud representations.

Coordinates in the `IfcPointCloud` elements themselves make up the bulk of the average expected data sets. We suggest one of three possible types of representations described in the following sections. End-users and implementations are free to mix different coordinate storage mechanisms within a single file to come to a most optimal solution. In particular the authors envision how the various formats can be combine to form a progressive notion of Level Of Detail, in which several grids (Figure 5c) and (Figure 5b) describe an ordered collection of point cloud subsets over the same association surface, in which GPU rendering budget (see Section 2.2) or user-defined queries can select among the segments to be rendered. The label on `IfcPointCloud` that signifies to what LOD a segment belongs is intentionally flexible to support such strategies.

3.2 Point cloud coordinate storage

Figure 5: Schematic representations of three ways of storing point cloud coordinates

3.2.1 3D Cartesian coordinates

Re-using the existing two dimensional array, `IfcCartesianPointList3D` elements can be used to capture the coordinates of `IfcPointClouds` using the established

mechanism of nested local coordinate systems relative to the site or storey of a building. This allows capturing the positions at arbitrary precision rates. By means of the `IfcPointCloud` attributes vector, colours, normals and other metadata can be stored alongside the coordinates of the points. On a geometrical level, no relationship is established between the points and the building elements (Figure 5a).

3.2.2 Parametric surfaces

To increase the efficiency of storage and deepen the semantic connection of a point cloud segment and the building element it describes, the schema extension incorporates the storage of points in the form of parametric coordinates $\{u, v\}$ on an association surface $\mathcal{S}$, with an offset w along the surface normal of $\mathcal{S}$ evaluated at $\{u, v\}$. The entity is named `IfcParameterValueList` (Listing 4). When paired with other serialization techniques, described in Section 3.3, parametric coordinates will require less storage space, as opposed to coordinates in Cartesian space. For this serialization technique, values need to conform to a fixed domain of values. For this purpose `IfcRectangularTrimmedSurfaces` are used to indicate the area of an unbounded surface in which the points are located. Therefore the coordinates can be normalized over this range and represented as “scaled integers” with an explicit precision. Note that here, “rectangular trimmed” refers to a rectangular area in the parametric $\{u, v\}$ space and hence is applicable to cylindrical, spherical or b-spline surfaces as well.

There are two options to instantiate this entity from point cloud data. Preferably, existing building element geometry is used to parametrize the points. In Figure 5b is illustrated how points are defined relative to a building element surface. In itself this does not entail a loss in precision, the points are solely represented in another coordinate space. As discussed in Section 2.2 the building element will provide meaningful and efficient best-guesses of surfaces to explain points. Semantically, this entails in addition that the distance from points to this surface is explicitly stored in the w component. Hence, structural deviations between the building as designed in IFC and as measured in the point cloud can be retrieved from the model later-on without computation. The procedure of associating points in software is described in Section 6.2.1.

Alternatively new surfaces can be instantiated to parametrize points in case no nearby building elements are suitable. These surfaces can be fit through the point clouds using conventional surface fitting techniques, such as RANSAC (Fischler and Bolles, 1981), as implemented in software (Section 6.2.2).

3.2.3 Height fields

As an extension to the approach of parametrization described in the preceding Section 3.2.2, a third option is the usage of height fields which store points as offsets

Figure 6: Visualization of associated and unassociated points. Associated points are coloured according to their parametrization, unassociated points, describing abutting constructions, reflections and furniture, coloured in grey

in discretized grid cells, illustrated in Figure 5c. As such, the $\{u, v\}$ components in parametric space are defined at the intersection of grid axes and per point only the w component needs to be stored in an entity named `IfcGridOffsetList` (Listing 5). Hence, the dimensionality required to represent a single point is reduced from three, for either $\{x, y, z\}$ or $\{u, v, w\}$, to one, $\{w\}$, which is the offset from the surface along the surface normal.

Two ways of constructing such grids are proposed. Firstly, the existing `IfcGrid` entity is leveraged and allows arbitrary grids, including radial and irregular grids, to be constructed from a list of arbitrary curves. These axes should be defined as `IfcPCurves` to maintain the invariant that these curves lie on the association surface. **p-curves**, short for parametric curves, are curves defined in the $\{u, v\}$ space of an underlying surface. Secondly, to reduce the overhead of `IfcGrid` definitions for a simple rectangular grid a new entity `IfcSurfaceGrid` is introduced that models a rectangular grid on an association surface by means of a regular spacing interval and number of axes in both the u and v direction.

Finding efficient stratifications for the points to conform to a grid is up to the implementation that instantiates the data. This is a typical operation in point cloud compression techniques, for example as described by Schnabel et al. (2008). Several grids can be superimposed by means of aggregating multiple `IfcPointCloud` instances as representation items.

In case of discontinuities in the density of point clouds the `CoverageMask` is to be used to model the absence of points in a certain region. An alternative would be to employ an `EXPRESS ARRAY` for the `Offsets`, since `ARRAYs` of `OPTIONAL` elements allow for `NULL` (“\$” in IFC-SPF) elements. The separate attribute has been chosen instead, because an `ARRAY` attribute would result in a less optimal serialization in HDF5 (see Section 5). In HDF5 missing or null elements are not available. Therefore, such `ARRAY` elements are represented as a pair of (a) a flag that signifies whether the element is set or unset (b) the valuation of the

Figure 7: Coarse visualization of a collection of `IfcGridOffsetLists` in 3D and their underlying parametrization surfaces

element (or default value if it is unset). This compound value would hamper the application of specific-purpose imaging compression methods that work best on a contiguous sequence of similar values.

A visual depiction of such a height map with colour-coded distance values is given in Figure 1.

3.3 Floating point discretization

Two approaches have been presented in the preceding sections 3.2.2 and 3.2.3 that model point cloud coordinates in bounded parametric spaces. The space is bounded in the $\{u, v\}$ direction by means of an `IfcRectangularTrimmedSurface`. The w direction is bounded by two additional attributes that specify the minimum and maximum deviation from the surface for a subset of point cloud coordinates (`IfcGridOffsetList.MinOrthogonalOffset` and `-MaxOrthogonalOffset` and `IfcParameterValueList.WMinOffset` and `-WMaxOffset` respectively).

With the range of values bounded by a lower and upper bound, the range can be discretized using a fixed number of steps and expressed as an integer numerator with known denominator.

A floating point number is represented as $(-s) \times d \times \beta^e$, with, in case of typical IEEE 754 single precision, 23 bits for the significand d , 8 bits for the exponent e , 1 sign bit s and base $\beta 2$. If the range of values is limited to $[0.0, 1.0]$ a number with similar, but uniform, precision can be formalized as $d/(2^n - 1)$, with n being the number of bits for the significand and hence can be represented using an integer number. If the parametric range $[u1, u2]$ corresponds to $1m$ and n is either 8 bits or 16 bits, the equivalent of an **unsigned byte** or **unsigned short** in standard computer architectures, the obtained precision is respectively $1m/255 \approx 4mm$ and $1m/65535 \approx 0.02mm$. For on-site measurement with terres-

trial laser scanners this usually exceeds the accuracy of common hardware and satisfies tolerances for potential use cases such as e.g. precast concrete elements (Kim et al., 2015) or steel. Also note that parametric ranges can be arbitrarily subdivided by instantiating multiple `IfcRectangularTrimmedSurfaces`, over the building surface geometries, facilitating arbitrary precision even for integer numeral representations. With this in mind a discrete and a continuous variant for each of the abstract base classes `IfcParameterValueList` and `IfcGridOffsetList` is created, either supplying $\{u, v, w\}$ and $\{w\}$ values as floating point numbers or as integer numerals. An indication of the advantage of such an approach, even in SPF serialization can be found in Listing 1. It exemplifies how less characters are necessary to approximate a bounded floating point number with a ratio of integer numbers ($12/255 \approx 0.04918746$, and so forth). Such representations of rational numbers are common in audio visual signal processing due to storage efficiency and financial applications due to precise rounding on decimal cents Cowlishaw (2003). In the E57 standard a similar data type for point cloud coordinates is available as an `E57_SCALED_INTEGER` and LAS uses signed integers for its coordinate structure (Isenburg, 2013).

Listing 1: Comparison of clear text encoding for floating point numerals

```

1  ((0.04918746,0.1925387,0.8992467))
2  ((12,49,229),255)

```

Apart from bounding the values and hence enabling the scaled integer notation. These bounds also provide interesting semantic information on point cloud subsets that can be retrieved efficiently and models a measure of the degree of conformance of the point cloud subset to the building element.

3.4 Scanner metadata Property Sets

It is of importance to record metadata pertaining to the scanning process as part of the point cloud model (Maravelakis et al., 2013) as the atmospheric characteristics during the scanning process influence the scanner precision. Furthermore metadata pertaining to the time of capturing is necessary to document the changing nature of the edifice. The E57 standard ASTM International (2011) outlines a comprehensive set of metadata elements that capture relevant information pertaining to a point cloud scan. These elements are fully described in Lindlar et al. (2015) and are listed here for reference.

Property sets are a mechanism in IFC to embed data in a model that is not standardized in the schema. In essence they are key-value pairs associated to one or more elements. Commonly used sets of such properties have been standardized, such as `PSET_WallCommon` to describe common properties of a wall, in addition, applications are free to emit custom property sets. Analogous to the other stan-

standardized IFC property sets, a `PSET_PointCloudCommon` is introduced as part of this proposal.

Name	Property Type	Data Type
SensorVendor	IfcPropertySingleValue	IfcLabel
SensorModel	IfcPropertySingleValue	IfcLabel
SensorSerialNumber	IfcPropertySingleValue	IfcLabel
SensorHardwareVersion	IfcPropertySingleValue	IfcLabel
SensorSoftwareVersion	IfcPropertySingleValue	IfcLabel
SensorFirmwareVersion	IfcPropertySingleValue	IfcLabel
Temperature	IfcPropertySingleValue	IfcThermodynamicTemperatureMeasure / THERMODYNAMICTEMPERATUREUNIT
RelativeHumidity	IfcPropertySingleValue	IfcRatioMeasure
AtmosphericPressure	IfcPropertySingleValue	IfcPressureMeasure / PRESSUREUNIT
AcquisitionStart	IfcPropertyReferenceValue	IfcDateAndTime
AcquisitionEnd	IfcPropertyReferenceValue	IfcDateAndTime

Table 1: `PSET_PointCloudCommon` property listing

The e57 standard in addition defines entries for `Name`, `Guid`, `Description`. These are covered by respective `IfcPointCloudElement` explicit attributes inherited from `IfcRoot` and therefore do not need a property definition. The `Pose` metadata, which models a location and orientation shall be modelled by means of `IfcPointCloudElement.ObjectPlacement`.

4 Definition of the schema extension

Provided below is a definition of the schema extension. Certain parts are omitted for brevity, such as the inclusion of the IFC4 schema and certain elementary building blocks such as the `IfcDirectionList` and attribute definitions for colours and normals. These omitted parts can be found on-line in an open source repository⁵.

Listing 2: Definition of the `IfcPointCloud` entity

```

1 ENTITY IfcPointCloud
2   SUBTYPE OF (IfcGeometricRepresentationItem);
3     LevelOfDetail : OPTIONAL IfcLabel;
4     Coordinates    : IfcPointCloudCoordinateSelect;
5     Attributes     : LIST [0:?] OF IfcPointCloudAttribute;
6   WHERE
7     WR1 : SIZEOF(QUERY(Attr <* SELF\IfcPointCloud.Attributes |
8                     SELF\Coordinates.Dim <> SIZEOF(Attr.Values))) == 0;
9 END_ENTITY;

```

⁵<https://github.com/DURAARK/IFCPointCloud>

All concrete types of `IfcPointCloudCoordinateSelect` have a derived attribute `Dim` that indicates the amount of points in the set. Using the `QUERY` construct in `WR1` this amount is checked to match the extent of every attribute vector. This establishes that for every point in the `Coordinates` definition a matching attribute element exist.

Listing 3: Definition of the `IfcPointCloudElement` entity

```

10 ENTITY IfcPointCloudElement
11   SUBTYPE OF (IfcElement);
12 END_ENTITY;

```

As mentioned in Section 3.1, a subtype of `IfcElement` is introduced to act as decomposition node to group several scans into a single organizational element. The metadata defined in Section 3.4 can be associated to such elements by means of the conventional `construct`.

Listing 4: Definition of the `IfcParameterValueList` entity

```

13 ENTITY IfcParameterValueList
14   ABSTRACT SUPERTYPE OF (ONEOF(IfcContinuousParameterValueList ,
15     IfcDiscreteParameterValueList))
16   SUBTYPE OF (IfcGeometricRepresentationItem);
17     WMinOffset : IfcLengthMeasure;
18     WMaxOffset : IfcLengthMeasure;
19     Surface    : IfcRectangularTrimmedSurface;
20 END_ENTITY;
21
22 ENTITY IfcDiscreteParameterValueList
23   SUBTYPE OF (IfcParameterValueList);
24     Values : LIST [1:?] OF LIST [3:3] OF INTEGER;
25     USampleWidth, VSampleWidth, WSampleWidth : INTEGER;
26   DERIVE
27     Dim : INTEGER := SIZEOF(Values);
28   WHERE
29     WR1 : SIZEOF(QUERY(UVW <* SELF\IfcDiscreteParameterValueList.Values | ((
30       UVW[0] < 0) OR (UVW[1] < 0) OR (UVW[2] < 0) OR (UVW[0] > USampleWidth
31       ) OR (UVW[1] >= VSampleWidth) OR (UVW[2] >= WSampleWidth)))) == 0;
32 END_ENTITY;

```

The `IfcDiscreteParameterValueList` models point cloud coordinates in the parametric space of an association surface, as described in Section 3.2.2. Parameter values are represented as scaled integers, as discussed in Section 3.3. The continuous variant of this construct that models such coordinates as conventional floating point numbers is omitted for brevity.

Listing 5: Definition of the `IfcGridOffsetList` entity

```

30 ENTITY IfcGridOffsetList
31   ABSTRACT SUPERTYPE OF (ONEOF(IfcContiousGridOffsetList,
    IfcDiscreteGridOffsetList));
32   Grid                : IfcGridSelect;
33   MinOrthogonalOffset : IfcLengthMeasure;
34   MaxOrthogonalOffset : IfcLengthMeasure;
35   CoverageMask        : OPTIONAL LIST [1:?] OF LIST [1:?] OF BOOLEAN;
36 END_ENTITY;
37
38 ENTITY IfcContinuousGridOffsetList
39   SUBTYPE OF (IfcGridOffsetList);
40   Offsets            : LIST [1:?] OF LIST [1:?] OF IfcParameterValue;
41   DERIVE
42     Dim              : INTEGER := SIZEOF(Offsets) * SIZEOF(Offsets[0])
    ;
43   WHERE
44     -- Same dimensions as CoverageMask
45     WR1 : NOT(EXISTS(CoverageMask)) OR (QUERY(Temp1 <* Offsets | SIZEOF(
    Temp1)) ==
46       QUERY(Temp2 <* CoverageMask | SIZEOF(Temp2)));
47     -- The offsets form a rectangular grid:
48     WR2 : SIZEOF(QUERY(Row <* Offsets | SIZEOF(Row) == SIZEOF(Offsets[0])
    )) == 0;
49     -- An offset exists for every grid intersection
50     WR3 : ((NOT('IFC4.IFCSURFACEGRID' IN TYPEOF(Grid)) OR (
51       (Grid.NumUAxes == SIZEOF(Offsets[0])) AND
52       (Grid.NumVAxes == SIZEOF(Offsets)))
53     ) AND (NOT('IFC4.IFCGRID' IN TYPEOF(Grid)) OR (
54       (SIZEOF(Grid.UAxes) == SIZEOF(Offsets[0])) AND
55       (SIZEOF(Grid.VAxes) == SIZEOF(Offsets)) AND
56       (SIZEOF(Grid.WAxes) == 0)))
57     );
58 END_ENTITY;

```

5 Proposed binary serialization format using HDF

Using the schema extension described above, point clouds can be serialized as SPF, the prevalent clear-text IFC serialization format. As shown in Listing 1 the floating point discretization technique (Section 3.3) is effective also in a clear-text encoding. However, we recommend the use of the Hierarchical Data Format (HDF) (Folk et al., 2011) for an efficient handling of large datasets. HDF has been used for more than 20 years in different engineering and scientific communities to cope with large amounts of data including e.g. physics, astronomy and medical datasets. It is an open standard with mature portable implementations in many languages that allows efficient Input and Output (I/O). Subsets of HDF5-serialized models can be retrieved in near constant time (Krijnen and Beetz, 2016). It provides flexible mechanisms for data modeling in a transparent, self-documenting way that makes

it an accepted candidate for long-term preservation (Bertin-Mahieux et al., 2011; Gray et al., 2005) and interoperability (Choi and Folk, 2007; Dougherty et al., 2009) use cases. With the ISO 10303 part 26, a standard exists that specifies mappings of STEP schemas from EXPRESS into HDF5 as well as the serialization of instance model populations. The rich hierarchical data model of HDF5 and the preexisting mapping defined in ISO 10303 part 26 are compelling reasons to favor HDF5 over, for example, *netCDF* (Rew and Davis, 1990) and *FITS* (Wells et al., 1981), which also offer support for large tabular data structures, have generally simpler interfaces, but less support for the complex data types encountered in STEP and IFC.

Apart from the storage space advantages that comes with a binary format, HDF also adds a number of additional advantages over part 21 SPF workflows. Firstly, the underlying data schema and metadata is transported along with the instance files. Secondly, the hierarchical nature of HDF and its indexing mechanisms, allow fast random access and slices through large datasets. The fact that parts of the model can be parsed independently allows for incrementally loading the fragments of interest to the end-user. This is of particular interest for point cloud data workflows where often only partial datasets (‘progressive’ LODs, spatial compartments, element clusters) need to be processed.

The binary HDF5 serialization of conventional IFC models is governed by and described in Krijnen and Beetz (2016). For this research an adaptation has been made to operate on the extended schema with point clouds. In (Krijnen and Beetz, 2016) the performance implications of variable length aggregates are described. Most notably these aggregates necessitate storage on a separate object heap that is unaffected by (compression) filters. ISO 10303 part 26 also defines an option to store large aggregate values in separate datasets and to refer to them from the instantiating instance by means of the object referencing methods offered by HDF5. This schema extension introduces potentially lengthy aggregates for all geometric forms. Hence, this serialization option is investigated particularly as it also allows such attributes to have chunking applied to them, which further facilitates reading subsets of the data.

6 Prototype implementation and test results

In this section we provide an overview of the prototypical implementation that we have developed to validate the feasibility for the suggested approach.

6.1 Test data sets

The test data sets used for the experiments described in this paper are based on the publicly available “Haus 30” and “Byg 72” ⁶. Both datasets contains a

⁶<http://data.duraark.eu>

Figure 8: A general purpose HDF viewer, HDFView (freely and openly distributed by the HDF Group), showing an IFC instance model population with point clouds. The outline on the left shows the hierarchical nature of HDF, in which the mapping approach results in a separate dataset for every entity type. In addition, large aggregate-valued attributes can be stored in an individual dataset as well, using HDF’s dataset referencing mechanism.

professionally modelled IFC, paired with a point cloud from a terrestrial laser scanner (a *FARO Focus 3D S 120*). The “Haus 30” dataset comprises a historic building near Berlin, Germany with five stories. The point cloud consists of a scan of a single storey within this building, comprising of 40 internal scanner positions, sampled down to approx. 13 million individual points. These have been registered with the IFC representation and when stored as an E57 require about 201 MB of storage space. The original IFC file is about 19 MB in size and contains 1115 building elements. The building plot covers 37×18 meters. The “Byg 72” point cloud dataset has not been downsampled, the E57 point cloud file is 12.8 GB and the IFC file a mere 409 KB, describing a single building storey with 168 elements covering an area of 30×19 meters.

6.2 Software Prototype

A number of conversion tools have been implemented, that extract the raw point clouds from the source files, register point cloud clusters to building elements, structure them according to different options of the proposed schema, serialize them as IFC-SPF and, lastly, convert the SPF model into an IFC-HDF5 model. For visual control, a graphical 3D viewer prototype has been built by the authors that allows additional analytical visualizations of e.g. point-to-surface deviations. A screen-shot of this viewer opening a low-resolution grid discretized point cloud is given in Figure 7. Only open source libraries, such as IfcOpenShell, h5py, Point Cloud Library (PCL), e57lib, NumPy, Scipy and OpenCascade have been used during the implementation and the prototypes themselves are available in the open source software repository. Permutations of different combinations have been generated to compare the resulting files, which form the basis of Section 6.3.

(a) Complete point cloud (100%)

(b) Points associated to elements (74.5%)

(c) Points unassociated to elements (25.5%)

(d) Points with planar segmentation (84.1% of 74.5%)

(e) Points without planar segmentation (15.9% of 25.5%)

Figure 9: Graphical illustration of the association process of the “Haus 30” dataset.

6.2.1 Point association

As can be seen in Figure 9b, the input point cloud is associated to building element surfaces that are extracted from the IFC file. IfcOpenShell is used to convert the implicit building element geometry into explicit boundary representation surfaces. The Point Cloud Library (PCL) is used to match points to building element surfaces based on geometrical proximity. If deemed desirable, additional criteria could be taken into account such as an agreement between point normal vectors, computed from neighbouring points, and the IFC surface normal. For this prototype geometrical proximity was deemed sufficient. This approach works for planar as well as arbitrarily curved surfaces, but is more efficiently computed for the former, as the parametrization on planar surfaces is essentially an affine transformation that can be represented by a matrix multiplication. Figure 10 shows for various building element types the distances to the associated points for the “Haus 30” model, based on this figure a global association threshold of 0.15 meters has been chosen. With this threshold 74.5% of the nearly 13 million points was associated to a building element surface. For the “Byg 72” dataset this is 70.0% of the 636 million points. Because this model contains more furniture, the association percentage is slightly lower, but the same 0.15m association distance is applicable.

Figure 10: Kernel density estimation (bandwidth=0.01) of point association distances for various building element types in the “Haus 30” model.

6.2.2 Planar segmentation

Not all points can be associated to building elements. Additional points are introduced by window reflections, points on outside foliage or indoor elements that have not been modelled explicitly, such as furniture or opened doors. In Figure 9b it can be observed that for the “Haus 30” model the unassociated points are mostly due to opened doors, external points and lightning fixtures that have not been modelled in the IFC file. PCL is used to perform an iterative planar seg-

Algorithm 1 Planar segmentation of unassociated points

```
1: function SEGMENT( $pts$ )
2:    $P_{min} \leftarrow \langle \min_{p \in pts} p_x, \min_{p \in pts} p_y, \min_{p \in pts} p_z \rangle$   $\triangleright$  lower corner of bounding box
3:    $P_{max} \leftarrow \langle \max_{p \in pts} p_x, \max_{p \in pts} p_y, \max_{p \in pts} p_z \rangle$   $\triangleright$  upper corner of bounding box
4:    $\langle N_i, N_j, N_k \rangle \leftarrow \text{CEIL}((P_{max} - P_{min} + \epsilon) / \text{VOXEL\_SIZE})$   $\triangleright$  number of voxels in three dimensions
5:    $voxels \leftarrow \emptyset$ 
6:   for all  $p$  in  $pts$  do
7:      $\langle i, j, k \rangle \leftarrow \text{FLOOR}((p - P_{min}) / \text{VOXEL\_SIZE})$ 
8:      $v_{i,j,k} \leftarrow v_{i,j,k} \cup p$ 
9:      $voxels \leftarrow voxels \cup v_{i,j,k}$   $\triangleright$  add point to relevant voxel
10:   $visited \leftarrow \emptyset$ 
11:  while  $visited \neq voxels$  do
12:     $v_{i,j,k} \leftarrow v \mid v \in voxels, v \notin visited$   $\triangleright$  for every voxel not visited
13:     $visited \leftarrow visited \cup \{v_{i,j,k}\}$ 
14:     $queue \leftarrow \{v_{i,j,k}\}$   $\triangleright$  add voxel to queue
15:     $component \leftarrow v_{i,j,k}$   $\triangleright$  add points in voxel to current component
16:    while  $queue \neq \emptyset$  do
17:       $v_{i,j,k} \leftarrow v \mid v \in queue$   $\triangleright$  pop element from queue
18:       $queue \leftarrow queue - \{v_{i,j,k}\}$ 
19:      for all  $\langle \Delta i, \Delta j, \Delta k \rangle$  in  $\{-1, 0, 1\} \times \{-1, 0, 1\} \times \{-1, 0, 1\}$  do
20:         $\triangleright$  consider the eight neighbouring voxels
21:        if  $v_{i+\Delta i, j+\Delta j, k+\Delta k} \neq \emptyset$  and  $v_{i+\Delta i, j+\Delta j, k+\Delta k} \notin visited$  then
22:           $\triangleright$  if not empty and not priorly visited
23:           $visited \leftarrow visited \cup \{v_{i+\Delta i, j+\Delta j, k+\Delta k}\}$   $\triangleright$  mark as visited
24:           $component \leftarrow component \cup v_{i+\Delta i, j+\Delta j, k+\Delta k}$ 
25:           $\triangleright$  add points to current component
26:           $queue \leftarrow queue \cup \{v_{i+\Delta i, j+\Delta j, k+\Delta k}\}$   $\triangleright$  add to queue to process recursively
27:    while  $|component| \geq \text{THRESHOLD}$  do
28:       $inliers \leftarrow \text{RANSAC}(component)$   $\triangleright$  apply ‘RANSAC’ to every component
29:      if  $|inliers| < \text{THRESHOLD}$  then  $\triangleright$  ... until the number of inliers is small
30:        break
31:       $component \leftarrow component - inliers$   $\triangleright$  ... or the remaining number of points
32:       $\triangleright$  in the component is small
33:   $\text{WRITE}(inliers)$ 
```

mentation of the point cloud using the RANSAC algorithm. In this prototypical implementation the RANSAC algorithm operates on local subsets that are found using a voxel-based approach, in which subsets span the connected components of neighbouring three-dimensional non-empty cells of points. An overview of this procedure is given in Algorithm 1. Using this approach another 2.8 out of the 3.3 million points (84.1%) were stored within bounded planar surfaces. Hence, these segments can be compressed using the scaled integer notation outlined in Section 3.3.

6.2.3 IFC and HDF5 serialization

The input IFC model is augmented with point cloud entities using IfcOpenShell based on several input parameters. These include settings such as grid sizes and levels of association. An IFC-SPF file is serialized first, which is converted into an HDF5 file using Krijnen and Beetz (2016).

(a) Points associated to elements (70.0%)

(b) Points unassociated to elements (30.0%)

Figure 11: Some excerpts of the association process of the “Byg 72” dataset.

6.3 Resulting model sizes

Figure 12: Overview of resulting file sizes for various serialization options. With: (a) **fmt**: the file format being used, either the e57 input file an or IFC-SPF or IFC-HDF serialization of the model (b) **fpd**: the usage of floating point discretization (c) **pln**: the usage of addition plane segmentation for unassociated points (d) **grd**: the usage of grid discretization (e) **cpn**: usage of compression, zip in case of IFC-SPF and gzip in case IFC-HDF.

The performance indicators used during the test are mainly focused on the impact of different data modelling approaches on file sizes, an overview of access times on IFC-HDF models can be found in Krijnen and Beetz (2016). The complexity of writing the models is not assessed, as the methodologies conform to

conventional point cloud compression techniques.

From the comparison of file size for the “Haus 30” model, presented in Figure 12, several observations can be made. Firstly, note that the e57 file (column 0), which has been the source document for the point clouds is half the size of the naive uncompressed IFC-HDF serialization in (3). This is due to the fact that e57 uses 32bit floating point numbers, whereas the IFC-HDF serialized model uses 64bit floating point numbers, this in order to maintain the integrity of the building element geometry. While in theory nothing prevents an implementation to choose different floating point numeral widths for the instantiation of distinct entity types, this has not been attempted.

Also note that the compressed SPF serialization (2) in byte count, does not deviate tremendously from the binary HDF file (4). In fact, SPF is not at all inefficient with regard to byte count. For example, an SPF entity instance reference like ‘#123’ can take more space in HDF as it is described as a compound structure of two unsigned integers, according to ISO 10303 part 26 §6.10.4. In addition **SELECT** and **OPTIONAL EXPRESS** attributes in HDF need to be initialized for all possible concrete instantiations, no matter the actual attribute value, as HDF does not allow missing data, as described in ISO 10303 part 26 §6.9.3.4 and Krijnen and Beetz (2016). On the other hand, the zip compression quite effectively minimizes the otherwise massive overhead of repeated IFC entity names throughout the SPF file, such as the thousands of occurrences of `IfcCartesianPoint`. However, the compressed SPF file provides an undesirable solution as the file needs to be decompressed in its entirety and offers no mechanisms to retrieve subsets. This is exactly a feature built-into HDF by means of its hierarchical nature and block-level compression and decompression.

The introduction of associations between point clouds segments and building elements comes with a minimal cost, as association entities need to be populated, as is visible when comparing cases (1) and (5). However, as discussed in Section 3.3, the parametrization onto bounded surfaces allows the use of an integer discretized notation for the parametrized components. Combined with a precision loss under typical scanner’s resolution, this yields a tremendous file size reduction in both SPF (8) and HDF (9-10). At this point the HDF file (9-10) is smaller than the original e57 file, carrying the same number of points, enriched with the complete IFC building model instantiation. The planar segmentation yields another additional compression step in (11-13).

The compression steps applied up until now are *near-lossless*: in order to describe points in $\{u, v, w\}$ coordinates of building element surfaces (Section 3.2.2) the points undergo a transformation that is invertible, but due to the limited precision of IEEE 754 floating point arithmetic there will be rounding errors. The scaled integer notation introduces a known precision loss, that will typically be within the range of the scanner hardware precision. The error is easily quantifiable at the moment the discretization is applied. For the “Haus 30” dataset a

Figure 13: Density plot of the error (in *millimetres*) due to scaled integer notation in the “Haus 30” dataset, using a range of $[0, 2^{16} - 1]$ to fit into a 2-byte unsigned integer. Note how especially the error in the w -direction is tiny, which is the distance from the point along the surface normal.

distribution of this error is given in Figure 13. It shows how this error depends on the extents of the bounded association surfaces, for the “Haus 30” model, these can be large, especially for the floor and ceiling slabs, as indicated in Figure 1. The error can be further reduced by decomposing such association surfaces into smaller bounded regions.

The introduction of grid discretization as described in section 3.4 introduces a *lossy* compression layer in the sense that points will be discarded and stratified over grid cells. A grid size of 50mm and 100mm is shown in (14-15) and (16-17) respectively. When using the discretized notation for the floating point numbers, even in text-based SPF, a low resolution associated point cloud can be embedded in the IFC file of comparable file size to the building data itself, as can be seen in the red and green parts of (16). Such a low resolution (50mm) point cloud totalling to over 1.1 million points is probably not useful to assess exact surface properties of the building elements, but might be sufficient for facility management purposes to assess the exact location, rotation and existence of building elements. Due to the stratification over grid cells, the error obtained by this approach is most prominent in directions along the building element surface.

The introduction of grid discretization is also where the difference in file size between SPF and HDF serialized models becomes the most apparent. This is due to the option described in Section 3.2.3 to choose to separate the coverage information in a separate **CoverageMask** attribute, rather than encoding this into the **Offsets** attribute directly, making use of the fact that EXPRESS **ARRAY** values can be set to NULL (‘\$’ in IFC-SPF). The four bytes needed for encoding a boolean value in IFC-SPF including separation character (.T.,...) is significantly larger than the single byte in IFC-HDF, which is compressed efficiently typical compression methods employed in **gzip** such as run length encoding as the areas

covered by points will mostly be contiguous. By applying rectangular decomposition onto the projected point cloud segments, large voids and the presence of such coverage masks can be partly avoided, see Figure 1 at the beginning of this paper.

Figure 14: Overview of resulting file sizes for various serialization options. With: (a) **fmt**: the file format being used, either the e57 input file an or IFC-SPF or IFC-HDF serialization of the model (b) **fpd**: the usage of floating point discretization (c) **pln**: the usage of addition plane segmentation for unassociated points (d) **grd**: the usage of grid discretization (e) **cpn**: usage of compression, zip in case of IFC-SPF and gzip in case IFC-HDF.

The same encoding options are given for the “Byg 72” dataset in Figure 14. Overall the resulting file sizes follow a similar pattern. The main differences are in the relative size of the IFC model, which is utterly insignificant in relation to the size of the point cloud (409 KB versus 12.8 GB without compression). Secondly, since this point cloud has not been subsampled prior to association, the spatially non-uniform distribution of points, results in a larger loss of points when the grid discretization is applied. This is a consequence of the fact that for this software prototype only a regular rectangular grid has been implemented. More points will be retained when a radial grid is implemented that matches the scanner profile.

The aim of this proposal is to create a semantically rich coupling between building elements and point cloud segments. As such, subsets of the point clouds, relevant to a certain information need, either a spatial subset or pertaining to a certain category of elements can easily and rapidly be extracted, especially in the HDF5-based binary serialization format due to its hierarchical storage paradigm. By expressing point coordinates based on building element surfaces a favourable

	IFC-SPF model size*	IFC-HDF model size*	Number of points
Haus 30	19 758 kb	18 100 kb	12 944 802
Byg 72	409 kb	1 139 kb	636 250 789
	Size in IEEE 754 32bit [†]	IFC-HDF combined size [‡]	Realized compression ratio
Haus 30	151 696 kb	79 711 kb	53.1%
Byg 72	7 456 063 kb	2 411 217 kb	67.7%

Table 2: Comparative listing of file sizes. * This is the original IFC model without point cloud data. Due to overheads in dataset allocation, for small files, the HDF5 serialized model can be larger in size [†] This has been chosen as a reference rather than E57 [‡] Serialized according to option (13): near-lossless compression with planar segmentation, floating point discretization, and HDF5 compression, but no grid discretization. Note that 64bit floating point values are used for points that are neither segmented nor associated.

amount of compression has been obtained, which is listed in Table 2. This is not the main objective and the exact compression ratio is not quantifiable as the resulting output is a combination of building model and annotated point cloud. The compression ratio is higher for larger files, in particular due to the fact that point cloud segments are populated in individual HDF5 datasets that have a non-negligible file size overhead.

7 Conclusion

In this paper a novel extension is introduced that provides a harmonized storage of point cloud models and IFC files, using an extension to the IFC schema. The method presented follows state of the art point cloud compression techniques. In addition, their application in this context deepens the semantic relation between building elements and point clouds, as points are parametrized onto the building surface they describe. Compression is introduced by means of discretized notation of floating point numerals and surface discretization into rectangular and irregular free-form grids. A prototypical implementation has shown that using this schema, coarsely subsampled point clouds can be embedded in the prevalent clear text serialization form of IFC. In order to facilitate the storage of industry standard large point cloud datasets, a novel serialization form using the HDF file format is proposed, which follows existing ISO standards to serialize EXPRESS instance models.

8 Future work

One of the core advantages of the HDF5 format is the usage of transparent block-level compression. HDF5 allows several compression schemes, including user-defined compression methods. These would allow much higher compression ratios by exploiting structural knowledge of the point cloud or by introducing additional lossiness in the compression methods. In the prototypical implementation only

gzip compression is used. Especially the point clouds segments stored as height maps projected on parametric surfaces (Figure 1) might be suitable for specific-purpose compression methods, such as jpeg or png, which can exploit and filter imperceivable differences.

A more rigorous study of the precision implications of the various constructs introduced in this paper will result in best practices for future implementers of the proposed standard extension. In addition, it might yield a more successful compression level when this knowledge is used to apply rectangular decomposition methods on the parametrized surface in order further constrain the parametric range based on localized point cloud densities and in order to decompose faces with internal boundaries, for example from walls with opening elements.

When more metadata about scanner positions and models is known, it could be worthwhile to assess whether the projection of points onto parametric surfaces conforms to a re-usable pattern. In this case a precise IfcGrid can be derived, reused for multiple point cloud associations, minimizing the precision loss by point cloud interpolation onto grids and maximizing the compression ratio.

Lastly, future research will indicate how the associated point cloud structure presented in this paper can be paired with other spatial indexing structures to further advance the localized extraction of point cloud segments and spatial querying techniques. Further experiments will be conducted to harness and reuse the general purpose decomposition and aggregation relationships of the IFC to implement octrees and kd-trees to further enhance the structure and accessibility of the data.

9 Acknowledgements

This work has been partially funded by the EU FP7 project "DURAARK", grant agreement 600908.

10 References

References

- T. Liebich, Ifc4 release summary., 2013. URL: <http://www.buildingsmart-tech.org/specifications/ifc-releases/ifc4-release>.
- ASTM International, E2807 - 11: Standard specification for 3d imaging data exchange, version 1.0, 2011. URL: <http://www.astm.org/Standards/E2807.htm>. doi:10.1520/e2807-11.
- R. B. Rusu, S. Cousins, 3d is here: Point cloud library (pcl), in: Robotics and Automation (ICRA), 2011 IEEE International Conference on, IEEE, 2011, pp. 1-4.

- S. Li, J. Isele, G. Bretthauer, Proposed methodology for generation of building information model with laserscanning, *Tsinghua Science and Technology* 13, Supplement 1 (2008) 138 – 144.
- S. Yamaoka, S. Kanai, H. Date, Data modeling and geometry registration for the integrated management of the bim and laser-scanned point clouds, in: *Proc. International Conference on Civil and Building Engineering Informatics*, 2013, pp. 55 – 63.
- N.-J. Shih, M.-C. Wu, J. Kunz, The inspections of as-built construction records by 3d point clouds, *Center for Integrated facility engineering (CIFE)* (2004).
- M. Golparvar-Fard, F. Peña-Mora, S. Savarese, Integrated sequential as-built and as-planned representation with d 4 ar tools in support of decision-making tasks in the aec/fm industry, *Journal of Construction Engineering and Management* 137 (2011) 1099–1116.
- A. Bhatla, S. Y. Choe, O. Fierro, F. Leite, Evaluation of accuracy of as-built 3d modeling from photos taken by handheld digital cameras, *Automation in construction* 28 (2012) 116–127.
- S. Pu, G. Vosselman, Knowledge based reconstruction of building models from terrestrial laser scanning data, *ISPRS Journal of Photogrammetry and Remote Sensing* 64 (2009) 575–584.
- M. Bügler, G. Ogunmakin, J. Teizer, P. A. Vela, A. Borrmann, A comprehensive methodology for vision-based progress and activity estimation of excavation processes for productivity assessment, in: *Proceedings of the 21st International Workshop: Intelligent Computing in Engineering (EG-ICE)*, Cardiff, Wales, 2014. doi:10.13140/RG.2.1.4630.2561.
- J. Yang, M.-W. Park, P. A. Vela, M. Golparvar-Fard, Construction performance monitoring via still images, time-lapse photos, and video streams: Now, tomorrow, and the future, *Advanced Engineering Informatics* 29 (2015) 211–224.
- J. Teizer, Status quo and open challenges in vision-based sensing and tracking of temporary resources on infrastructure construction sites, *Advanced Engineering Informatics* 29 (2015) 225–238.
- A. Braun, S. Tuttas, A. Borrmann, U. Stilla, A concept for automated construction progress monitoring using bim-based geometric constraints and photogrammetric point clouds, *ITcon* 20 (2015) 68–79.
- K. Khoshelham, S. O. Elberink, Accuracy and resolution of kinect depth data for indoor mapping applications, *Sensors* 12 (2012) 1437–1454.
- P. Tang, D. Huber, B. Akinci, R. Lipman, A. Lytle, Automatic reconstruction of as-built building information models from laser-scanned point clouds: A review of related techniques, *Automation in construction* 19 (2010) 829–843.

- I. Brilakis, M. Lourakis, R. Sacks, S. Savarese, S. Christodoulou, J. Teizer, A. Makhmalbaf, Toward automated generation of parametric bims based on hybrid video and laser scanning data, *Advanced Engineering Informatics* 24 (2010) 456–465.
- S. Dietze, J. Beetz, U. Gadiraju, G. Katsimpras, R. Wessel, R. Berndt, Towards preservation of semantically enriched architectural knowledge., in: *SDA*, 2013, pp. 4–15.
- Y. Alshawabkeh, N. Haala, Integration of digital photogrammetry and laser scanning for heritage documentation, *International Archives of Photogrammetry and Remote Sensing* 35 (2004) B5.
- D. González-Aguilera, J. Gómez-Lahoz, J. Sánchez, A new approach for structural monitoring of large dams with a three-dimensional laser scanner, *Sensors* 8 (2008) 5866–5883.
- S. J. Gordon, D. D. Lichti, Modeling terrestrial laser scanner data for precise structural deformation measurement, *Journal of Surveying Engineering* 133 (2007) 72–80.
- G. Teza, A. Galgaro, F. Moro, Contactless recognition of concrete surface damage from laser scanning and curvature computation, *NDT & E International* 42 (2009) 240–249.
- D. Meagher, Geometric modeling using octree encoding, *Computer graphics and image processing* 19 (1982) 129–147.
- J. L. Bentley, Multidimensional binary search trees used for associative searching, *Communications of the ACM* 18 (1975) 509–517.
- P. Goswami, F. Erol, R. Mukhi, R. Pajarola, E. Gobbetti, An efficient multi-resolution framework for high quality interactive rendering of massive point clouds using multi-way kd-trees, *The Visual Computer* 29 (2013) 69–83.
- R. Richter, S. Discher, J. Döllner, Out-of-core visualization of classified 3d point clouds, in: *3D Geoinformation Science*, Springer, 2015, pp. 227–242.
- M. Wimmer, C. Scheiblauer, Instant points: Fast rendering of unprocessed point clouds., in: *SPBG*, 2006, pp. 129–136.
- J. Ziv, A. Lempel, Compression of individual sequences via variable-rate coding, *Information Theory, IEEE Transactions on* 24 (1978) 530–536.
- J. Digne, R. Chaine, S. Valette, Self-similarity for accurate compression of point sampled surfaces, in: *Computer Graphics Forum*, volume 33, Wiley Online Library, 2014, pp. 155–164.

- T. Golla, C. Schwartz, R. Klein, Towards efficient online compression of incrementally acquired point clouds (2014).
- R. Schnabel, S. Möser, R. Klein, Fast vector quantization for efficient rendering of compressed point-clouds, *Computers & Graphics* 32 (2008) 246–259.
- M. Isenburg, Laszip, *Photogrammetric Engineering & Remote Sensing* 79 (2013) 209–217.
- S. Ochmann, R. Vock, R. Wessel, R. Klein, Automatic reconstruction of parametric building models from indoor point clouds, *Computers & Graphics* 54 (2016) 94 – 103. Special Issue on CAD/Graphics 2015.
- C. Gordon, F. Boukamp, D. Huber, E. Latimer, K. Park, B. Akinci, Combining reality capture technologies for construction defect detection: a case study, in: *EIA9: E-Activities and Intelligent Support in Design and the Built Environment*, 9th EuropIA International Conference, 2003, pp. 99–108.
- D. Girardeau-Montaut, Cloudcompare - open source project, 2011. URL: <http://www.danielgm.net/cc/>.
- T. Krijnen, J. Beetz, Efficient binary serialization of ifc models using hdf5, in: *Proceedings of the 16th International Conference on Computing in Civil and Building Engineering (ICCCBE2016)* Osaka, Japan, 2016.
- P. Pauwels, W. Terkaj, {EXPRESS} to {OWL} for construction industry: Towards a recommendable and usable ifcowl ontology, *Automation in Construction* 63 (2016) 100 – 133.
- P. Pauwels, W. Terkaj, T. Krijnen, J. Beetz, Coping with lists in the ifcowl ontology, in: *Proceedings of the 22nd EG-ICE Workshop 2015*, Eindhoven, The Netherlands, 2015, pp. 111 – 120.
- M. Folk, G. Heber, Q. Koziol, E. Pourmal, D. Robinson, An overview of the hdf5 technology suite and its applications, in: *Proceedings of the EDBT/ICDT 2011 Workshop on Array Databases*, ACM, 2011, pp. 36–47.
- M. A. Fischler, R. C. Bolles, Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography, *Communications of the ACM* 24 (1981) 381–395.
- M.-K. Kim, J. C. Cheng, H. Sohn, C.-C. Chang, A framework for dimensional and surface quality assessment of precast concrete elements using bim and 3d laser scanning, *Automation in Construction* 49 (2015) 225–238.
- M. F. Cowlishaw, Decimal floating-point: algorism for computers, in: *Computer Arithmetic*, 2003. *Proceedings. 16th IEEE Symposium on*, IEEE, 2003, pp. 104–111.

- E. Maravelakis, A. Konstantaras, A. Kritsotaki, D. Angelakis, M. Xinogalos, Analysing user needs for a unified 3d metadata recording and exploitation of cultural heritage monuments system, in: G. Bebis, R. Boyle, B. Parvin, D. Koracin, B. Li, F. Porikli, V. Zordan, J. Klosowski, S. Coquillart, X. Luo, M. Chen, D. Gotz (Eds.), *Advances in Visual Computing*, volume 8034 of *Lecture Notes in Computer Science*, Springer Berlin Heidelberg, 2013, pp. 138–147. doi:10.1007/978-3-642-41939-3_14.
- M. Lindlar, M. Panitz, U. Gadiraju, D6.2: Ingest and storage of 3d objects in a digital preservation system, 2015. Retrieved from http://duraark.eu/wp-content/uploads/2015/02/DURAARK_D6.2.pdf April 2016.
- T. Bertin-Mahieux, D. P. Ellis, B. Whitman, P. Lamere, The million song dataset, in: *ISMIR 2011: Proceedings of the 12th International Society for Music Information Retrieval Conference*, October 24-28, 2011, Miami, Florida, University of Miami, 2011, pp. 591–596.
- J. Gray, D. T. Liu, M. Nieto-Santisteban, A. Szalay, D. J. DeWitt, G. Heber, Scientific data management in the coming decade, *ACM SIGMOD Record* 34 (2005) 34–41.
- V. Choi, M. Folk, Investigations into using hdf5 as an alternative to step for finite element modeling data, in: *Proceedings of the 9th NASA-ESA Workshop on Product Data Exchange*, National Aeronautics and Space Administration, Santa Barbara, CA, 2007.
- M. T. Dougherty, M. J. Folk, E. Zadok, H. J. Bernstein, F. C. Bernstein, K. W. Eliceiri, W. Benger, C. Best, Unifying biological image formats with hdf5, *Communications of the ACM* 52 (2009) 42–47.
- R. Rew, G. Davis, Netcdf: an interface for scientific data access, *IEEE computer graphics and applications* 10 (1990) 76–82.
- D. C. Wells, E. W. Greisen, R. Harten, Fits-a flexible image transport system, *Astronomy and Astrophysics Supplement Series* 44 (1981) 363.