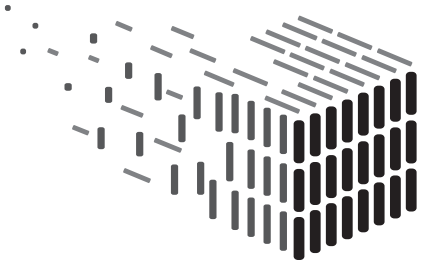

D5.5 Recognition of Architecturally Meaningful Structures and Shapes

Software prototype v3

DURAARK

FP7 – ICT – Digital Preservation
Grant agreement No.: 600908

Date: 2016-01-31
Version 1.0
Document id. : duraark/2016/D.5.5/v1.0

Grant agreement number	:	600908
Project acronym	:	DURAARK
Project full title	:	Durable Architectural Knowledge
Project's website	:	www.duraark.eu
Partners	:	LUH – Gottfried Wilhelm Leibniz Universitaet Hannover (Coordinator) [DE] UBO – Rheinische Friedrich-Wilhelms-Universitaet Bonn [DE] FhA – Fraunhofer Austria Research GmbH [AT] TUE – Technische Universiteit Eindhoven [NL] CITA – Kunstakademiets Arkitektskole [DK] LTU – Lulea Tekniska Universitet [SE] Catenda – Catenda AS [NO]
Project instrument	:	EU FP7 Collaborative Project
Project thematic priority	:	Information and Communication Technologies (ICT) Digital Preservation
Project start date	:	2013-02-01
Project duration	:	36 months
Document number	:	duraark/2016/D.5.5/v1.0
Title of document	:	Recognition of Architecturally Meaningful Structures and Shapes – Software prototype v3
Deliverable type	:	Software prototype
Contractual date of delivery	:	2016-01-31
Actual date of delivery	:	2016-01-31
Lead beneficiary	:	UBO
Author(s)	:	Sebastian Ochmann <ochmann@cs.uni-bonn.de> (UBO) Richard Vock <vock@cs.uni-bonn.de> (UBO) Raoul Wessel <wesselr@cs.uni-bonn.de> (UBO)
Responsible editor(s)	:	Sebastian Ochmann <ochmann@cs.uni-bonn.de> (UBO) Richard Vock <vock@cs.uni-bonn.de> (UBO)
Quality assessor(s)	:	Jakob Beetz <j.beetz@tue.nl> (TU/e) Ulrich Krispel <ulrich.krispel@vc.fraunhofer.at> (FhA)
Approval of this deliverable	:	Stefan Dietze <dietze@L3S.de> (LUH) – Project Coordinator
Distribution	:	Public
Keywords list	:	Curation, Building Semantics, Access Copies, Structuring, Geometric Enrichment, BIM, Point Cloud, E57, IFC

Executive Summary

This report presents the third version of the software prototype for the recognition of architecturally meaningful structures and shapes, as well as point cloud compression. It provides a thorough overview of the improvements and extensions incorporated since version 2 of the software prototype (see D5.3), gives details on their implementation, and describes the integration into the DURAARK Service Platform.

Table of Contents

1 Introduction

The ubiquity of high-resolution point cloud measurements as a cost- and time-efficient means to capture the as-built state of buildings makes automated methods for analyzing the measured data as well as efficient compression algorithms indispensable. In the domain of Long-term Digital Preservation (LDP), a suitable preprocessing and automated analysis of the data prior to ingest allows for the generation of light-weight, yet semantically rich representations of the buildings from the originally low-level point cloud data. The generated models can not only be used as an aid for more comprehensible visualization, but they also give insight into the stored data by providing high-level meta-information (e.g. number of rooms, room areas, room layout). This can be used for indexing of the data which would not be possible with only the raw geometric information provided by point cloud capturing devices.

Work package 5 deals with the detection and reconstruction of semantically meaningful structures and shapes in 3D indoor scans of buildings on different scales, i.e. primary structures (floors, ceilings, walls), architectural elements (doors, windows, columns) and detail elements (power sockets, power lines). The derived information is then used to generate detailed models in a highly automated manner. Furthermore, WP5 provides software tools to perform compression of large point cloud datasets in order to make e.g. transfer over low-bandwith connections for previewing purposes feasible.

In this report, the third version of the software prototypes implementing the detection of meaningful structures (Task 5.1), meaningful shapes (Task 5.2), and compression (Task 5.4) is described which were developed by UBO. The software prototype for the recongition of almost invisible structures (Task 5.3) developed by FhA is described in D5.6.

In the third and final project year, improvements and novel methods have been incorporated into each of the components in response to the evaluation and communication with stakeholders contributed by CITA (WP7). In addition, all developed software components are provided as self-contained Docker images for integration into the DURAARK Service Platform.

In the following we first give an overview of the integration of all components into the overall DURAARK Service Platform and their usage in the WorkbenchUI before describing the components in detail.

2 Integration and Software Manual

While the previous versions of the software prototypes were provided as stand-alone applications for usage on end-user systems, one of the main goals of the third and final prototypes was to provide a consistent, platform-independent deployment method for incorporating the tools as microservices into the DURAARK system. To this end, the Docker[1] containerization framework was chosen during the development of the DURAARK Service Platform in Work Package 2. Docker provides means to deploy software packages as self-contained images which allow the user to easily install and run applications on a variety of platforms. Consequently, the software prototypes developed in WP4 and WP5 are provided as Docker images which are available for installation through the publicly available Docker Hub[2]. Figure 1 shows an overview of the file formats, WP4/WP5 Docker images, and the data flow between them. A detailed description and usage guide for the images specific to this deliverable is given below.

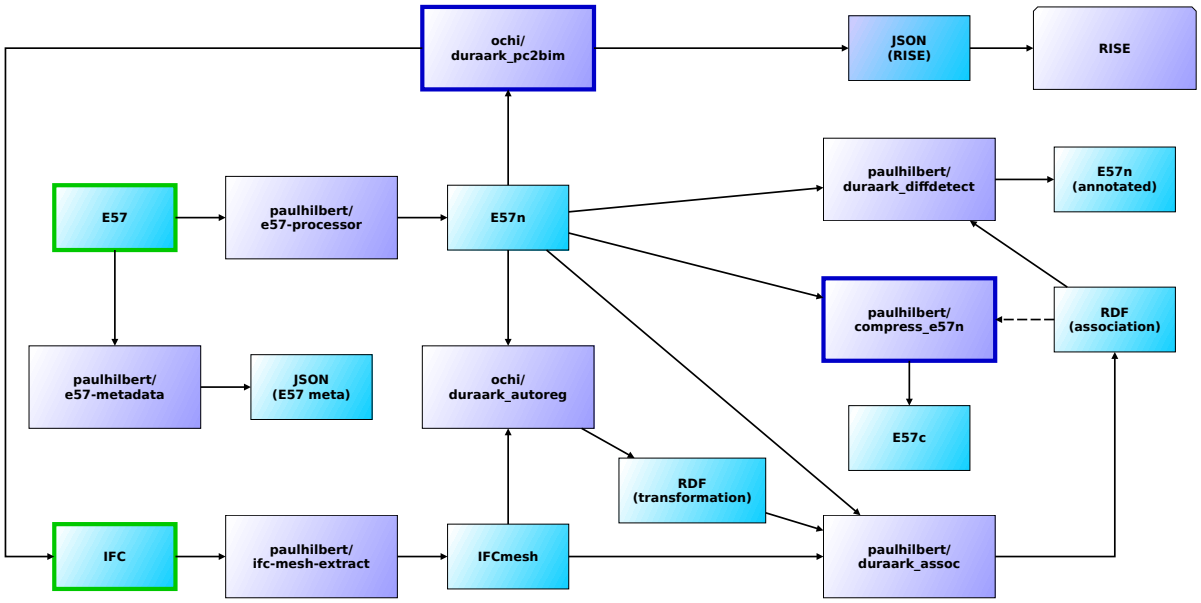

Figure 1: Overview of the Docker images provided by WP4 and WP5. The smaller, blue boxes represent files, purple boxes represent Docker images. The captions are file formats and names of the Docker images on Docker Hub, respectively. Processing starts with E57 and/or IFC files (boxes with green frames). The Docker images specific to WP5 are highlighted by blue frames.

[1] https://www.docker.com/
[2] https://hub.docker.com/

2.1 Component Installation and Images Shared Between WP4 And WP5

Please see deliverable D4.3 for a description of the component installation as well as a description of images which are shared between WP4 and WP5.

2.2 Images Specific to WP5

2.2.1 IFC Reconstruction

Docker image: `ochi/duraark_pc2bim`
Description: The point cloud to BIM reconstruction component. It takes a preprocessed (subsampled) E57n file and produces an Industry Foundation Classes (IFC) file of the reconstructed building model. Note that the E57n file is generated by the E57 processor as described in D4.3.
Input: E57n point cloud
Output: BIM model in IFC format and/or JSON for RISE component
Example:

```
docker run --rm -v /home/user/work:/work ochi/duraark_pc2bim --input
   /work/a.e57n --output /work/reconstruction.ifc --outputjson
   /work/reconstruction.json
```

2.2.2 Point Cloud Compression

Docker image: `paulhilbert/compress_e57n`
Description: The point cloud compression and decompression component. Note that the compression and decompression parts are contained in the same Docker image which can be called by using the respective arguments to the `docker run` command as shown in the examples below.
Input: E57n point cloud or compressed E57c point cloud
Output: E57c compressed point cloud (consisting of a JSON file and a binary file) or E57n point cloud

Example (compression):

```
docker run --rm -v /home/user/work:/work paulhilbert/compress_e57n
   duraark_compress --input-cloud /work/pointcloud.e57n --output-json
   /work/compressed.json --output /work/compressed.e57c
```

Example (decompression):

```
docker run --rm -v /home/user/work:/work paulhilbert/compress_e57n
   duraark_decompress --input-cloud /work/compressed.e57c --input-json
   /work/compressed.json --output /work/pointcloud.e57n
```

2.3 Usage in DURAARK WorkbenchUI

2.3.1 IFC Reconstruction

The IFC reconstruction component is integrated into the DURAARK WorkbenchUI as part of the "Geometric Tools" step during the ingest phase. The user is presented with a user interface similar to the screenshot shown in Figure 2. In case a point cloud file was added to the session beforehand, the user can now select the corresponding "Pointcloud" element on the left-hand side of the user interface to access geometric enrichment tools available for point cloud data.

Once a point cloud dataset has been selected, available geometric processing tools are shown on the right-hand side of the user interface. Amongst others, the "Reconstruct BIM Model" component allows to perform an IFC reconstruction for the selected point cloud dataset. The reconstruction process is started in the background by enabling the respective component as shown in Figure 3.

Once the reconstruction process has finished, the user can select the corresponding BIM model element on the left-hand side as shown in Figure 4. This loads a preview of the reconstructed IFC model on the right-hand side for inspection. The resulting IFC model is now included as part of data to be ingested into the archive.

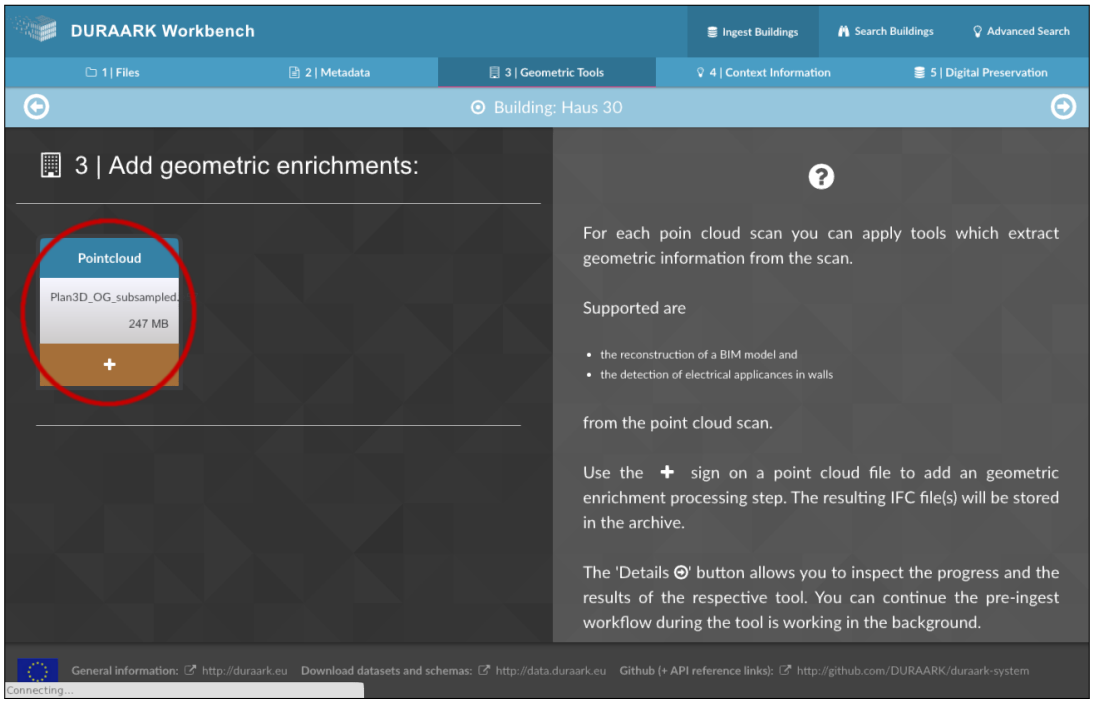

Figure 2: During an ingest session, the user may select available point cloud datasets on the left hand side to access geometric tools available for point cloud data.

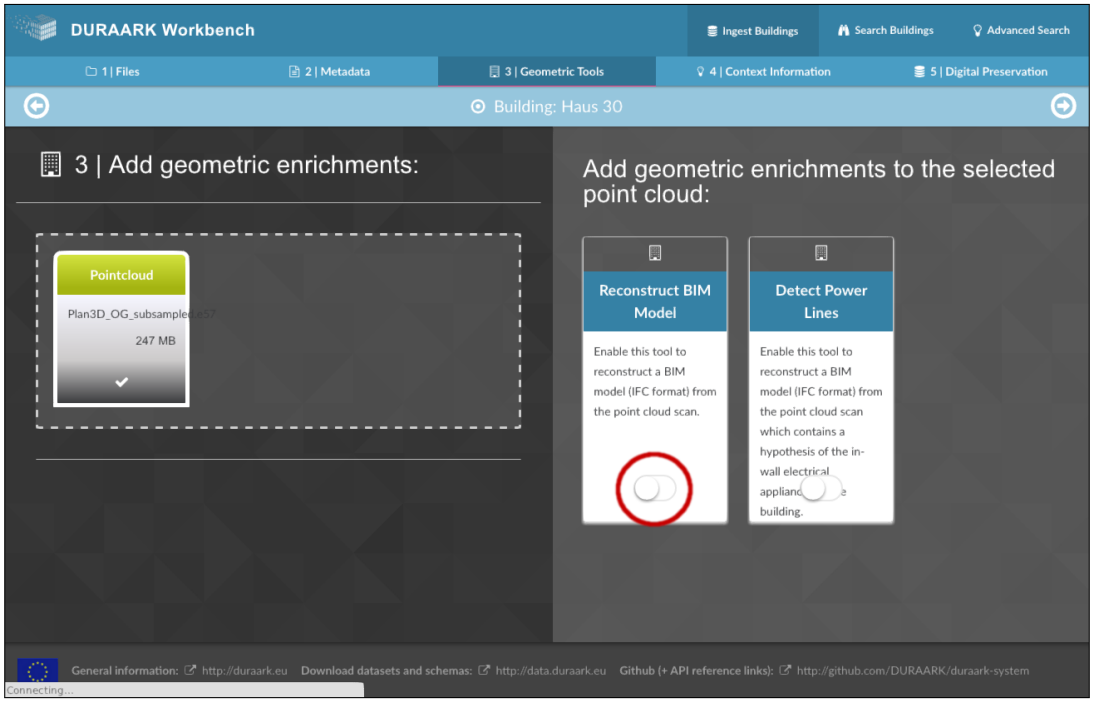

Figure 3: The IFC reconstruction is started in the background by enabling the respective component on the right hand side of the user interface.

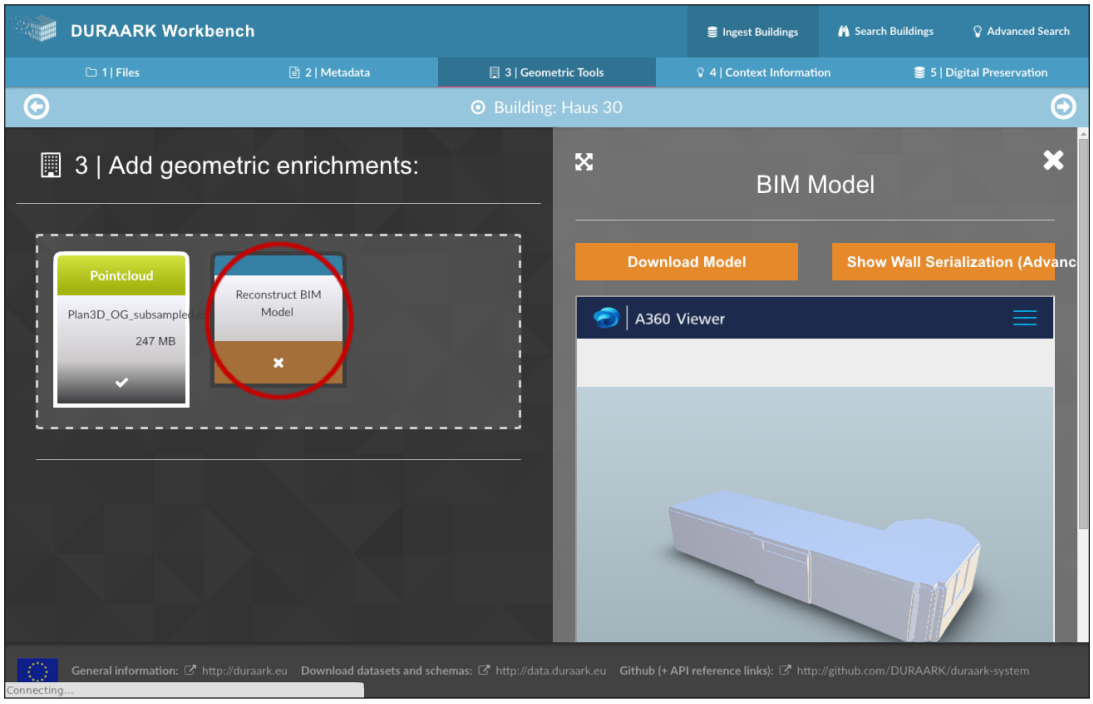

Figure 4: After the reconstruction process is finished the user may select the resulting BIM model element which opens a preview of the generated model.

2.3.2 Point Cloud Compression

During the ingest phase in the DURAARK Workbench UI, point cloud files can be compressed and downloaded for further processing as part of the "Geometric Tools" step.

The process starts with the selection of a point cloud file via the + sign. This opens up the available geometric enrichment tools for point clouds on the right. From the list switch on *Point Cloud Compression*. Next to the selected file a new tile appears, which allows you to select a compression ratio (between 0 and 1) for the output file. After the ratio was selected click the *Compress* button which starts the compression component in the background. See Figure 5 for a screenshot of the described step.
When the processing is finished the tile's header is switching from *Processing ...* to *Download File*. Click on *Download File* to download the compressed point cloud file (see Figure 6 for a screenshot).

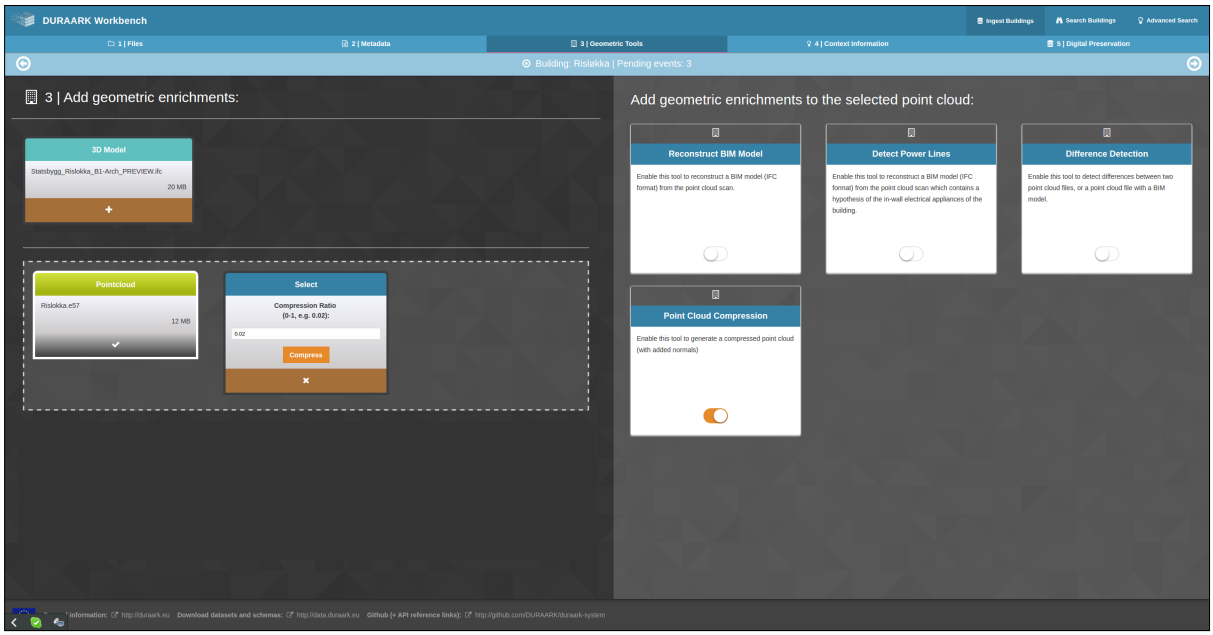

Figure 5: Compression is initiated by selecting a dataset on the left and then switching on the Compression component on the right.

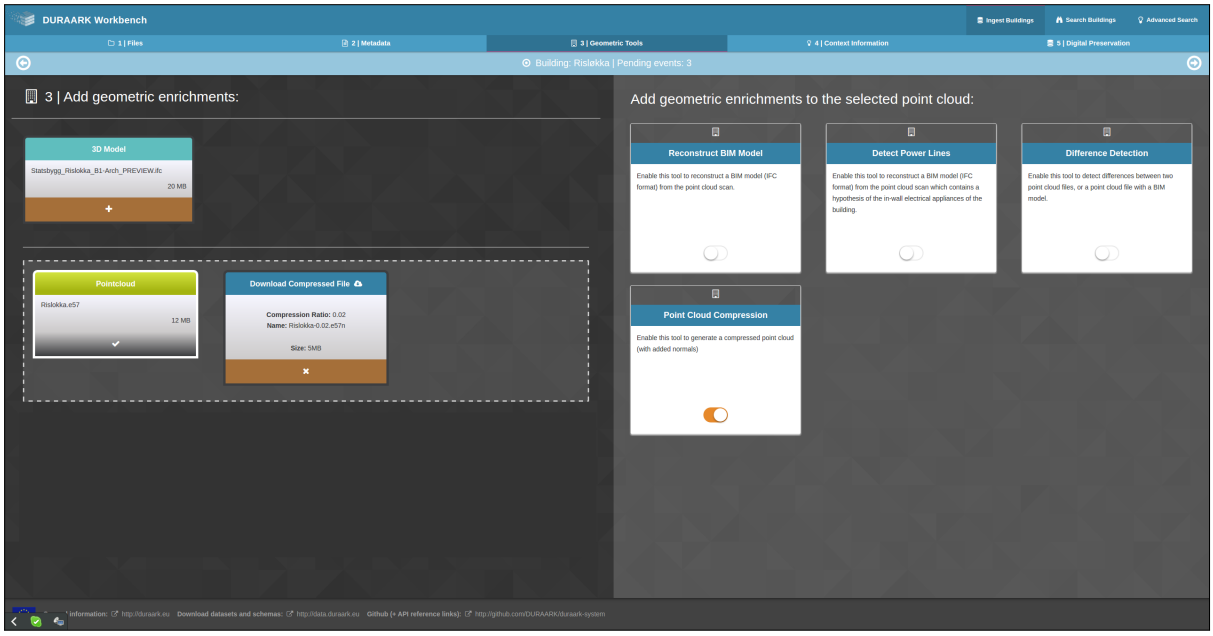

Figure 6: After processing is finished, the comprest result file can be downloaded for further processing.

3 Related Work

In this Section, we give a brief overview of related methods which have been published since the previous version of this deliverable (D5.3).

3.1 Structure Recognition and Reconstruction

Monszpart et al. [10] propose a method for extracting planar structures in point clouds which follow regularity constraints. Their optimization approach balances data fitting and simplicity of the resulting arrangement of planes. Ikehata et al. [7] present a 3D modeling framework for reconstructing indoor scenes from panorama RGBD (i.e. color and depth) images. The scene geometry is represented as a graph in which nodes correspond to structural elements such as rooms, walls, and objects. A grammar defining how a scene may be manipulated drives a reconstruction algorithm in which grammar rules are sequentially applied to recover a structured model. Thomson and Boehm [21] present a literature review and point cloud processing framework for the automated generation of BIM models from point cloud data, as well as the classification of point clouds given a corresponding BIM model. The BIM reconstruction process mainly consists of the detection of planar walls and a subsequent generation of suitable IFC entities (slabs, walls), incorporating several geometric reasoning rules. The method by Turner and Zakhor [22] generates high-detail watertight indoor models from point cloud data, including a segmentation of the coarse building structure and detail objects. To this end, they use volumetric representations of interior space which is built using carving and set (union, difference) operations. From this data structure, simplified mesh models are generated. Macher et al. [8] propose a multi-scale segmentation approach of point clouds into floors, rooms, and planes as a step towards generating as-built BIM models. For the detection of floors, a histogram-based approach analyzing clusters of altitudes of measured points is used. The room segmentation step is based on region growing in 2D projections of measured points. Finally, planar structures are detected using a RANSAC approach. Hong et al. [6] present a semi-automatic approach which aids with the generation of as-built BIM models from interior point cloud datasets. The method consists of multiple steps including a reduction of data size, determination of story heights, and floor boundary modeling. The resulting, extruded 3D wireframe model is used as a basis for BIM modeling in architecture software.

3.2 Shape Recognition

Mattausch et al. [9] propose an approach for automatic segmentation of indoor scenes by detecting repeated objects. To this end, planar patches are detected in the input point cloud data and related to each other by geometric transformations. Clustering patches according to a patch similarity measure based on shape descriptors and their spatial configuration yields a segmentation of the input data. The approach by Shi et al. [20] consists of scene segmentation, object classification by means of machine learning, and a graph matching step for extracting objects and object groups from indoor point clouds. The result is a labeled segmentation of the input scene on an object level. Díaz-Vilariño et al. [3] propose a method for the detection of columns in point clouds acquired by terrestrial laser scanners. They perform a rasterization of the input data onto the horizontal plane and implement a model-driven recognition approach based on Hough Transform.

3.3 Compression

The approaches by Ruhnke et al. in [16, 17], and Digne et al. [2] use oriented local surface patches which are fitted to the measured points. The data is represented by means of linear combinations of elements in an overcomplete codebook. Morell et al. [11] compress point clouds by replacing segmented subsets of planar points (detected using a Random Sample Consensus (RANSAC) approach) by delaunay triangulations where triangles are attributed with color information as well as their point distribution. In [12], the authors extend this approach by a second compression method, where a dimension reduction is achieved using self-organized neural networks. Coatsworth et al. [1] propose a method for real time compression of RGBD data by separating color from depth data and compressing both images separately (lossy for RGB and lossless for depth data). They then propose a general streaming architecture for fast (de-)compression and transmission from unmanned vehicles.

4 Recognition of Meaningful Structures

4.1 Recap of Developments in Year 1 and Year 2

We first give a short recap of the developments in the first two project years regarding the recognition of semantically meaningful structures in indoor point cloud scans before highlighting new developments during the third year.

In the first project year, a segmentation of the point cloud into separate, semantically meaningful parts was developed (Figure 7, left), including a detection of openings such as doors and windows between rooms. This segmentation allows for improved navigation and visualization of the point cloud by enabling targeted highlighting or selection of separate rooms in large point clouds. Furthermore, it provides information about room connectivity in the form of a light-weight graph representation in which rooms are nodes, and connections between rooms (e.g. doors) are edges. The results of this development were published in [15] and presented at the GRAPP 2014 conference in Lisbon, Portugal.

Building upon these results, UBO developed an approach for the reconstruction of high-level Building Information Modeling (BIM) models during the second project year (Figure 7, right). The resulting building representations go beyond a mere segmentation of the point cloud by automatically determining plausible, volumetric wall, floor, and ceiling elements as well as their connectivity. This directly follows the modeling paradigms used in modern BIM software and formats which allows the export of the resulting reconstructions in industry-standard IFC format for re-use and further processing in software like e.g. Autodesk Revit. In addition to detecting openings in a similar manner like the first software prototype, these openings are classified as doors and windows which are asso-

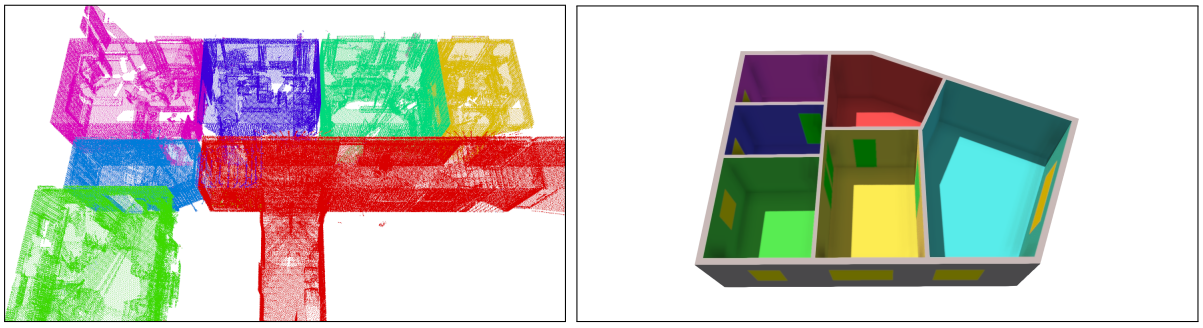

Figure 7: Example results from the previous reconstruction prototypes. Left: The first prototype performs a segmentation of the point cloud into separate rooms. Right: The second prototype reconstructs BIM models from indoor point clouds.

ciated with the respective, reconstructed walls. The results of the year 2 developments were published as a full paper in the Computers and Graphics Journal [14] and presented at the CAD/Graphics 2015 conference in Xi'an, China.

4.2 Problem Definition and Challenges

Developments in the third project year focused on improvements of the second version of the reconstruction software prototype and integration into the DURAARK System Platform. We first give an overview of the improvements that have been incorporated into the third software prototype and provide further details in Section 4.3.

Avoidance of short wall segments During evaluation of the Y2 reconstruction software prototype by CITA it became evident that e.g. clutter or partially occluded walls in the input point clouds sometimes led to overly segmented walls in the reconstructed model. While it is possible to fix these reconstruction errors manually in a post-processing step, we modified the reconstruction process such that an additional, iterative regularization step is performed which tries to avoid implausibly short wall segments (Figure 8). It should be noted that a simple removal of short wall segments is not a feasible solution since the optimization step of the reconstruction algorithm performs the reconstruction under certain constraints, e.g. that walls are always connected to other walls at their endpoints. A simple removal of wall elements would result in holes in walls and destroy the clear separation of neighboring rooms which may not be desirable.

Improved handling of multiple scans per room An assumption of the second reconstruction prototype is that each room of the building was scanned from one scan position, or only few scan positions. Due to the way the reconstruction approach works, multiple scans within a particular room lead to a segmentation of the room into multiple regions which are separated by implausible walls. An automatic post-processing step attempts to detect and remove these additional walls such that the segmented room is merged into a single region. While this approach often works well for few scan positions, alternative capturing methods which become increasingly relevant (e.g. using mobile devices) may yield a large number of scans per room, or even continuous trajectories of scan positions. In order to prepare for such scenarios, additional means to merge scans during the reconstruction process have been devised (Figure 9) as detailed in Section 4.3.

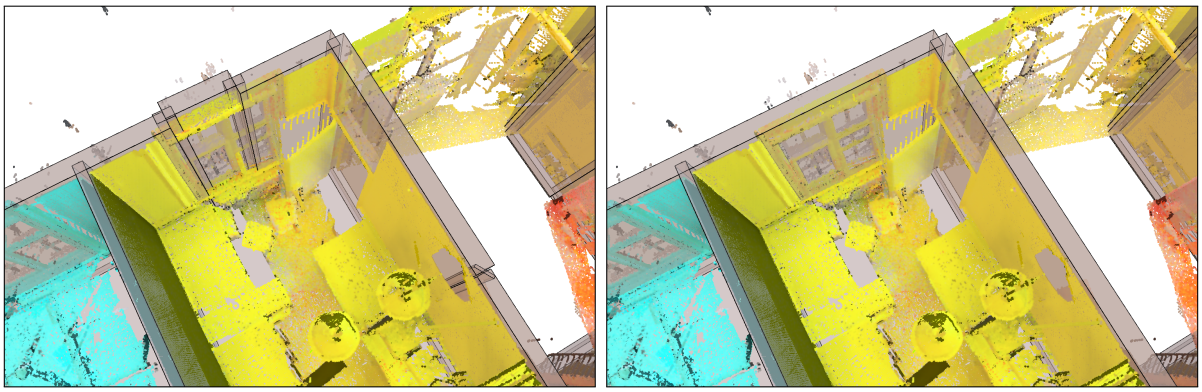

Figure 8: Reduction of implausibly short wall segments. Left: Result without short wall removal. Some walls are overly segmented. Right: Result after performing the iterative short wall removal.

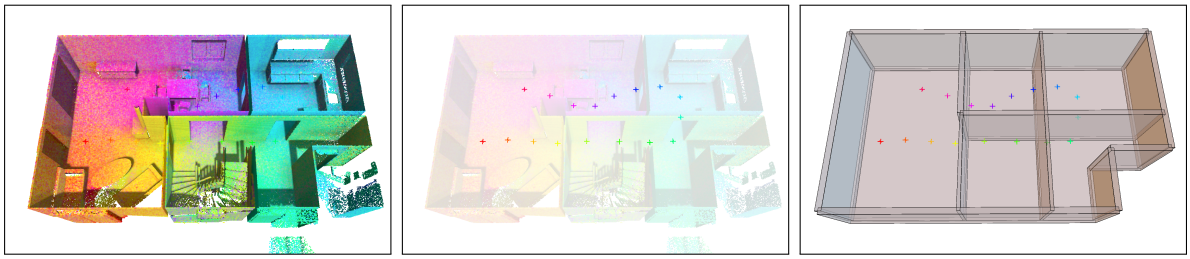

Figure 9: Handling of many scan positions in a synthetic example. Left: Input point cloud, separate scans are color-coded. Middle: The small crosshairs show the scan positions. Right: Even though there are many scan positions in each room, the algorithm tries to reconstruct larger, plausible rooms without over-segmenting the room layout.

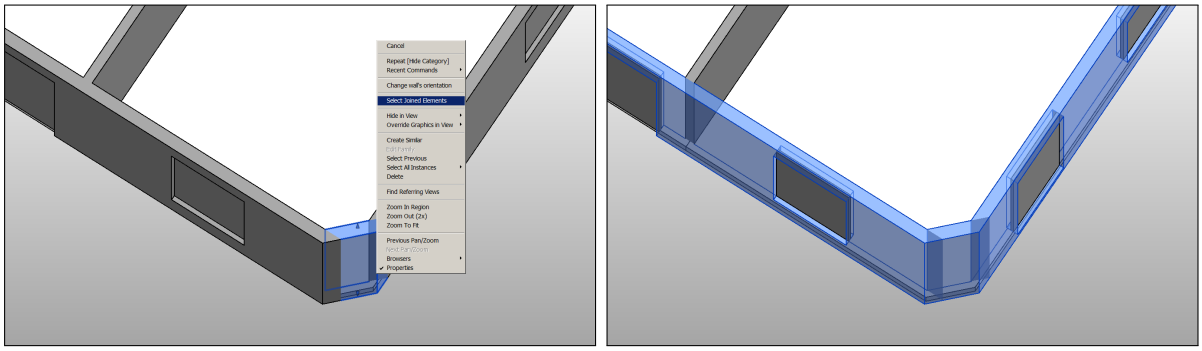

Figure 10: Exporting relations such as wall connections in the IFC file can help external software to relate elements to each other. Left: A wall is selected and the "Select Joined Elements" command is used in Autodesk Revit. Right: Wall elements incident to the selected wall are automatically selected.

Improved IFC export, JSON export for RISE The third reconstruction prototype supports more types of IFC entities and relations. In particular, detected doors and windows are now included in the exported IFC file and annotated with their respective type. For junctions between incident walls, additional relations are included in the IFC file that describe how walls are connected (e.g. by "L"- or "T"-shaped connections). This helps e.g. external software to correctly relate elements to each other (Figure 10). In addition to the IFC file format for exporting the resulting reconstruction, the prototype now also supports a JSON-based export format for usage in the Reveal Almost Invisible Structures (RISE) component (D5.6).

4.3 Implementation and Workflow

In this Section we provide details on how the aforementioned improvements have been implemented in the third version of the reconstruction prototype.

Avoidance of short wall segments During the BIM model reconstruction process, an intermediate data structure representing all possible constellations of walls is used, i.e. a planar graph in the horizontal plane in which edges are wall segments which may be used in the final reconstructed model. Using an optimization approach, areal regions (i.e. faces in the aforementioned planar graph) are labeled such that connected components of identically labeled regions are the final rooms (or outside area), and edges between differently labeled regions are walls (see D5.3 for a detailed description).
In some situations (especially in cluttered or partially occluded regions), the optimization resulted in room boundaries which are overly segmented and contained small, implausible wall segments. In order to avoid such segments, we exploit the cost minimization process which is applied for solving the optimization problem used for determining the region labeling: After the room labeling has been performed, the length of each resulting wall segment is evaluated and walls with a length below a user specified threshold are marked. The costs for using marked edges as walls in the final reconstruction are set to a high value and the optimization process is repeated. This forces the optimization algorithm to avoid the marked edges and search for alternative room boundaries. This process is performed iteratively until no more wall segments with a length below the set threshold are contained in the resulting reconstruction.

Improved handling of multiple scans per room The basis for estimating costs for assigning room labels to regions in the horizontal plane is a coarse segmentation of the point cloud into rooms. This segmentation is initialized by considering each separate scan as a different room label. The underlying, idealized assumption is that each room has been scanned from one position and thus the number of scans equals the number of rooms in the building. Naturally, larger, non-convex rooms (long hallways in particular) need to be scanned from more than one scan position which violates the aforementioned assumption.

In the previous reconstruction prototype, we enforced that one room was reconstructed for each scan position, leading to an over-segmentation of the building into rooms where multiple scans were taken. We alleviated this issue in the second prototype by performing a heuristic merging of adjacent reconstructed rooms if openings were detected between them that were too large to be plausible doors or windows. While this often works well with only few scans within a room, the problem becomes more pronounced when a larger number of scans is located within a room.

To account for many scan locations within a room, two changes have been incorporated in the third prototype: First, the prior segmentation of the point cloud into rooms does no longer yield a hard assignment of each scan point to only one of the scans, but the resulting segmentation is a *soft* assignment in which each point may be assigned to multiple scan positions with certain probabilities. As an example, measured points within a certain room that was scanned from multiple locations will have similar assignment probabilities for each scan position within that room. Second, we no longer force the reconstruction of one room per scan location. Instead, the costs for assigning a certain scan label to a region of the building is defined proportionally to the aforementioned assignment probability. As a consequence, the optimization is allowed to assign the label of *any* of the scans within a room. Due to the smoothness property of the underlying graph-cut-based optimization (i.e. the algorithm prefers large, uniformly labeled regions if the penalty is not too high), this results in a local merging of labels with similar assignment costs. Possible remaining, implausible walls after this initial merging may then be detected and removed in the same manner as in the previous prototype.

Improved IFC export, JSON export for RISE The IFC export functionality has been extended by a more detailed annotation of IFC entities. Openings are annotated with their respective type (door, window) and their opening direction. Furthermore,

walls are related to incident wall entities at their endpoints using suitable relation entities specified in the IFC standard.
In order to provide data input for usage by the RISE component developed by FhA, a JSON-based format for concisely encoding reconstructed rooms and walls as well as their relations (neighborhood, connectivity) has been defined in close collaboration with FhA. A corresponding export module has been developed by UBO and integrated into the reconstruction prototype.

5 Recognition of Meaningful Shapes

5.1 Recap of Developments in Year 1 and Year 2

The first version of the reconstruction prototype provided a segmentation of the point cloud on the level of rooms, and a light-weight graph representation of rooms and their connections (Figure 11, left). Building upon this representation, a more fine-grained object-level segmentation and classification was implemented. Objects detected within rooms (such as furniture) extend the room-level building representation such that a hierarchical decomposition into building storeys, rooms, their connections, and objects within rooms is obtained. Objects within rooms were classified as e.g. chairs and tables and the corresponding nodes in the resulting graph were attributed to allow for automatic searches for given room and object constellations within the building. These results were presented at the Eurographics Workshop on 3D Object Retrieval (3DOR 2014) in Strasbourg, France [13].
In addition to the first reconstruction software prototype, the second version included a classification of detected openings into doors and windows by means of a supervised machine learning approach (Figure 11, right).

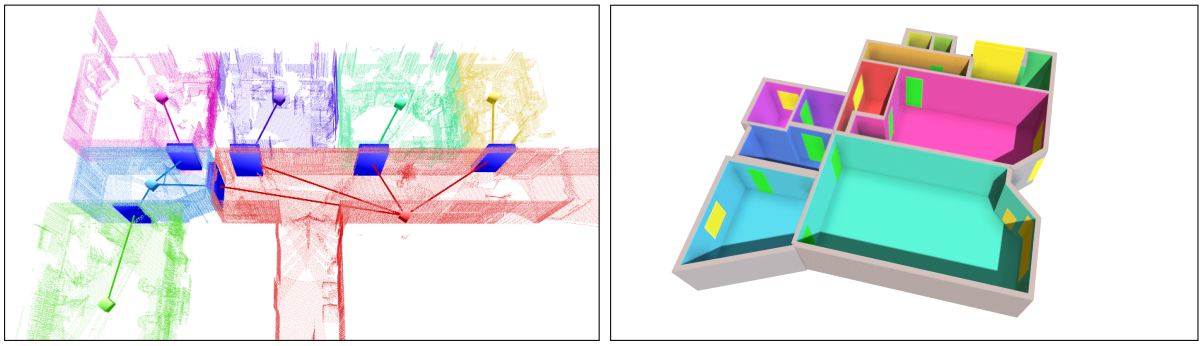

Figure 11: Example results of the previous shape recognition prototypes. Left: The first prototype is able to detect openings between adjacent rooms. Right: The second prototype classifies openings as doors and windows, and is able to detect openings between rooms and the outside area.

5.2 Problem Definition and Challenges

To further enrich the reconstructed BIM models, the following improvements over the second prototype have been developed and incorporated into the third version:

Door opening direction estimation An important property associated with openings (in particular doors) in a BIM model is their type together with additional information about e.g. their pivot points and opening directions. Since rotating doors are ubiquitous in the point cloud datasets gathered within the DURAARK project, we have incorporated an estimation for the opening direction of detected doors in order to further enrich the reconstructed model with this information (Figure 12).

Column detection Architectural elements such as columns and beams are important aspects of a building's primary structure which were not captured by our previous versions of our reconstruction prototype. As a first step towards extending the resulting models by such entities, we have incorporated a method for detecting vertical, cylindrical columns in the point cloud scans, including a faithful estimation of their dimensions (Figure 13).

5.3 Implementation and Workflow

Door opening direction estimation The estimation of door opening directions is performed as part of the BIM model reconstruction after walls have been generated and opening locations and geometries have been detected. It should be noted that the opening detection approach assumes that doors were opened during the scanning process. Consequently, the door geometry itself is usually scanned in its opened position. We exploit this circumstance to estimate the opening direction of each detected door in the reconstructed model.

For each detected door opening, a small point cloud subset around the door's position is extracted for further analysis. The size of this region depends on the width of the detected door. Our assumption is that the scanned, opened door geometry lies within this extracted point cloud subset such that we can restrict the search to this subset and thus reduce computational costs. Subsequently, a plane detection within the extracted point cloud is performed and all approximately vertical planes are considered as candidates for the door geometry. In order to determine the best matching candidate, we select the plane whose surface area is most similar to the detected door in the BIM model and whose left or right, vertical edge is close enough to either vertical edge of the reconstructed door opening. The latter criterion also determines the side on which the detected door geometry is hinged to the detected door opening. After the best candidate (with respect to the aforementioned criterions) has been determined, the opening direction is easily

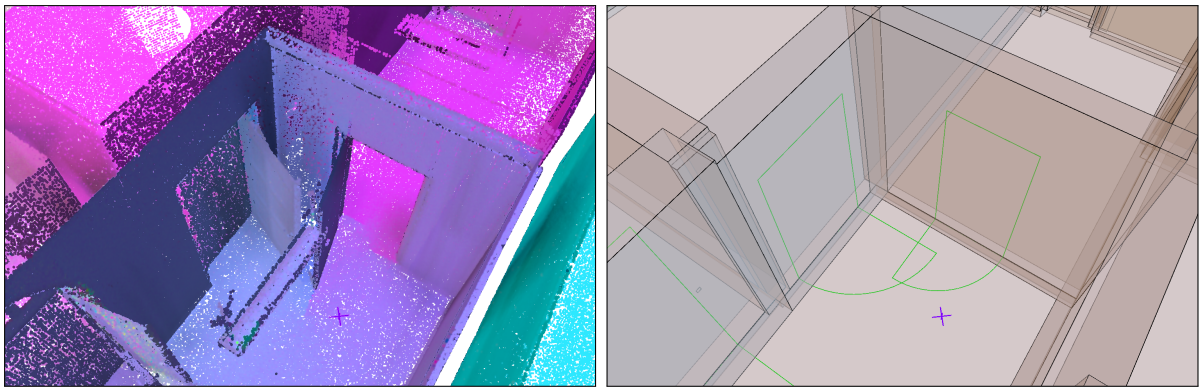

Figure 12: Door opening directions are now estimated from geometry detected in the point cloud data. Left: Input point cloud. The geometry of the opened doors are visible near the door openings. Right: Visualization of the detected openings and door opening directions (thin green lines).

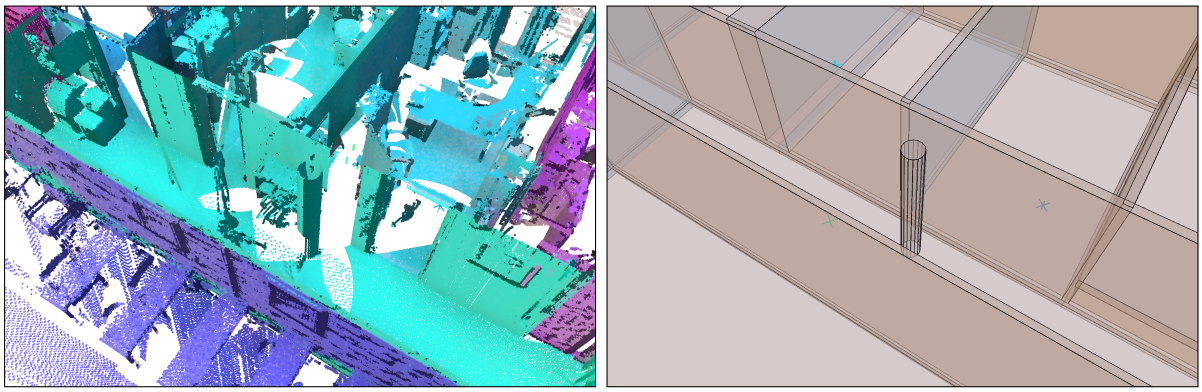

Figure 13: The third software prototype performs a detection of vertical columns in the point cloud datasets. Left: Input point cloud. A column is located in the hallway. Right: Resulting detected column.

determined by testing on which side of the corresponding wall the detected door geometry is located.

Column detection After reconstruction of floor, ceiling, wall, and opening geometries, an additional detection of cylindrical, vertical columns is performed. To this end, we exploit that the RANSAC-based primitive shape detection [19] implementation – which we use for the detection of planes – is also able to detect different shapes such as cylinders. A detection of differently shaped columns (e.g. columns with a rectangular profile) is in principle possible by implementing other kinds of shape definitions for the RANSAC shape detection, however we restricted ourselves to cylindrical columns in the scope of this prototype. Given user-specified thresholds for minimum and maximum column radii, heights, and a maximum angular deviation from the ideal "up" direction, invalid column candidates are filtered out from the set of all detected cylindrical shapes. For each of the remaining cylinders, a column is constructed and its shape is fitted to match the detected cylinder geometry. Since floor and ceiling heights are estimated beforehand by the reconstruction algorithm, the lower and upper bounds of the column are set to the floor and ceiling height at the estimated column position, respectively.

6 Point Cloud Compression and IFC Storage

6.1 Recap of Developments in Year 1 and Year 2

In the first project year, UBO evaluated two compression methods for generating lightweight access copies of point cloud datasets: An octree-based approach [18] that subsamples the point cloud in order to generate more coarse representations that are still suitable for previewing the stored data, and a newly developed method based on a detection of planes which replaced large planar structures by more coarse image-based representations. A compression technique based on a decomposition of the point cloud into small planar patches was developed during project year 2. The resulting method [5] was presented at the International Workshop on Vision, Modeling and Visualization (VMV 2014) in Darmstadt, Germany.

6.2 Problem Description and Challenges

While the patch-based compression technique developed in year 2 yielded good compression results and is well suited for architectural scenes with many planar surfaces, it has some drawbacks which we overcome in the third prototype.

Local decompression of point cloud subsets In certain usage scenarios such as the visualization of only a region of interest (e.g. a single room), the decompression of only this particular part of the point cloud is desirable. However the existing approach required decompression of the entire point cloud. In the third prototype, we thus modified the storage of compressed point cloud parts in a way which allows for targeted, local retrieval and decompression of only parts of interest.

Semantically meaningful structuring The second compression prototype uses an octree partitioning of the point cloud and a subsequent estimation of planes within octree cells. This is a very general approach which does not make strong assumptions about the underlying data. In architectural scenes however, we can make use of domain-specific knowledge and heuristics to improve compression and storage efficiency. In particular, the availability of both a point cloud and a corresponding BIM model enables usage of the surfaces given by the model as accurate and semantically meaningful hints where large planar structures are located. Using these hints for decomposing the point cloud into

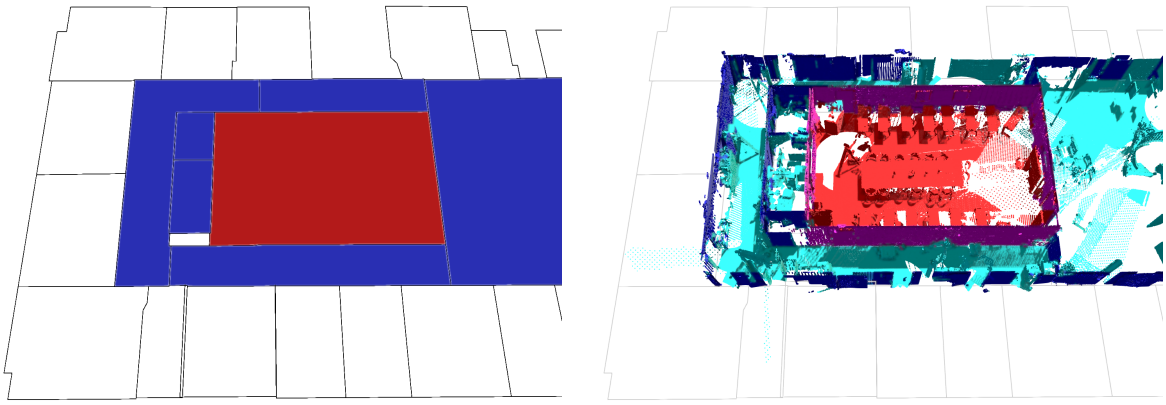

Figure 14: A semantically meaningful decomposition of a point cloud can be achieved based on a floor plan automatically generated by our BIM reconstruction component. Left: Floor plan in which a room has been selected (red). Neighboring rooms (blue) are automatically determined from the relations between rooms and walls. Right: This allows for targeted retrieval of only the relevant point cloud subsets.

smaller parts thus yields not only a more efficient compression, but also enables targeted retrieval and decompression of e.g. point cloud parts that correspond to a certain room or wall. Figure 14 shows an example for such a semantically meaningful segmentation based on a floor plan automatically generated by the BIM reconstruction component.

6.3 Implementation and Workflow

Local decompression of point cloud subsets The previous patch-based compression method used a global encoder for compressing part of the patch metadata such as their positions, orientations and sizes. This circumstance hinders local decompression of only part of the point cloud data which is currently relevant to the user, e.g. a single room, or a room and its neighbors. In contrast to this, the third version of our compression prototype provides an efficient indexing structure which allows for targeted selection of relevant point cloud parts with respect to e.g. the current camera position and orientation of a viewer. This pre-selection of relevant parts then allows for retrieval and decompression of only those relevant parts, and also for performing a suitable, prior ordering of the data subsets which need to be retrieved. This enables efficient retrieval of data in e.g. streaming scenarios.

Semantically meaningful structuring The aforementioned indexing structure may be based on different kinds of structurings of the point cloud data. A simple example is the detection of planes in the point cloud and a subsequent compression of point sets which correspond to separate detected planes. However, it is also possible to incorporate higher-level semantic information for structuring the point cloud and thus the stored, compressed representation. Given either an already existing BIM model, or a model automatically generated using our reconstruction prototype, the point cloud can be segmented into meaningful subsets such as points corresponding to separate stories, rooms, or building elements such as floors, ceilings, and walls. This structuring allows to perform selection of relevant point cloud subsets on a high semantic level, e.g. all points belonging to the same room as the camera of the viewer is currently located in, or for all rooms adjacent to this room (for targeted pre-loading of possibly relevant data).

7 Collaboration with Other Projects

UBO has collaborated with the *Mapping on Demand*[3,4] and *Harvest4D*[5,6] projects, especially during the second and third year of the DURAARK project. The goal of this collaboration was the exchange of expertise regarding point cloud compression techniques and the development of novel ideas in this field. Two papers [5, 4] have been published in collaboration with the respective partners from UBO during the timeframe of the DURAARK project which present improved compression schemes on architectural point cloud data. A variant of the method presented in [4] has been implemented in the third software prototype for point cloud compression.
Furthermore, the versatile and fast WebGL-based *Potree*[7] point cloud viewer which is being developed under the Harvest4D project has been adopted by the DURAARK project for visualization of point cloud data in the WorkbenchUI.

[3] `http://cg.cs.uni-bonn.de/en/projects/mapping-on-demand/`
[4] Funding: DFG Research Unit 1505
[5] `http://cg.cs.uni-bonn.de/en/projects/harvest4d/`
[6] Funding: European Commission, 7th Framework Programme
[7] `http://potree.org/`

8 Decisions & Risks

The technical decisions and risks discussed in this Section are an amendment to the risk assessment in the previous deliverable D5.3.

8.1 Technical Decisions

Component deployment as Docker images During development of the DURAARK Service Platform, it was decided to use the Docker containerization platform to allow for platform-independent deployment of the DURAARK microservices (see D2.5). Consequently, the software components in WP4 and WP5 are deployed as Docker images which are publicly available through Docker Hub.

Storage of Docker images in Docker Hub For easy installation of all software components, the Docker images are stored in the publicly available Docker Hub. In addition to storing the built images, Docker Hub also provides means to have images built automatically from e.g. GitHub repos containing the necessary Docker image description as a "Dockerfile".

8.2 Risk Assessment

Obsolescence of the Docker platform

- **Risk Description:** The Docker platform becomes obsolete and is superseded by alternative containerization solutions.
- **Risk Assessment:**
  - **Impact:** Medium
  - **Probability:** Low
  - **Description:** The Docker platform is relatively new and there exist competing containerization frameworks such as "rkt" or "LXD". It thus cannot be ruled out that Docker will be superseded by other containerization/virtualization solutions.

- **Contingency Solution:** Each of the Docker images fundamentally is a self-contained Linux-based operating system including the respective software component. As such, porting the components to similar frameworks should be relatively easy to do. Alternatively, the software could be run in full virtualization environments such as VirtualBox.

Abandonment or unavailability of Docker Hub

- **Risk Description:** The Docker Hub (but not Docker) is being abandoned or otherwise unavailable.
- **Risk Assessment:**
  - **Impact:** Low
  - **Probability:** Low
  - **Description:** Since the Docker Hub repositories are separate from the core development of the Docker framework, it is possible that the repository infrastructure is abandoned while the Docker platform remains available. Also, Docker Hub could be temporarily unavailable for various reasons.
- **Contingency Solution:** Usage of the Docker Hub for storing and building images is mostly a convenience. Docker images can also be built manually provided the respective "Dockerfile" which contains the information how a Docker image is built. In our case, all necessary Dockerfiles are publicly available on GitHub and thus the necessary images can be built with relatively little effort without using Docker Hub.

9 Licenses

The IPR type "software" is implied for all IPs listed below.

IP used / generated	**Software Name**	**License**	**Information**
used	libE57	libE57 license (similar to Boost Software License)	http://www.libe57.org/license.html
used	Xerces	Apache License 2.0	http://www.apache.org/licenses/
used	ICU	ICU License	http://source.icu-project.org/repos/icu/icu/trunk/license.html
used	IfcOpenShell	LGPL v3	http://ifcopenshell.org/
used	Open CASCADE community edition	LGPL v2.1 with additional exception	http://www.opencascade.org/getocc/license
used	Point Cloud Library	3-clause BSD	http://pointclouds.org/
used	Eigen	Mozilla Public License 2.0 (except for few parts that are under LGPL)	http://eigen.tuxfamily.org/
used	Boost	Boost Software License	http://www.boost.org/users/license.html
used	Flann	2-clause BSD	http://www.cs.ubc.ca/research/flann/
used	OpenMesh	LGPL v3 (with exception clause that "you may use any file of this software library without restriction")	http://www.openmesh.org/license/
used	zlib	zlib/libpng License	http://opensource.org/licenses/zlib-license.php

used	Primitive Shapes	Custom, see below.	—
used	NVIDIA OptiX	Inclusion in free applications allowed; commercial use requires Commercial License	https://developer.nvidia.com/optix
used	CGAL	LGPL/GPL, or commerical licensing	http://geometryfactory.com/products/licenses/
used	mpfr	LGPL v3	http://www.mpfr.org/
used	OpenCV	3-clause BSD	http://opencv.org/license.html
used	Leptonica	2-clause BSD	http://www.leptonica.com/about-the-license.html
used	JBig2Enc	Apache License Version 2.0	https://github.com/agl/jbig2enc/blob/master/COPYING
used	JBig2Dec	GPL v2 or later	https://github.com/rillian/jbig2dec/blob/master/LICENSE
used	OpenJPEG	2-clause BSD	https://github.com/uclouvain/openjpeg/blob/master/LICENSE
used	GCoptimization	Custom, see below.	—
generated	E57 processor	2-clause BSD	http://opensource.org/licenses/BSD-2-Clause
generated	IFC mesh extract	2-clause BSD	http://opensource.org/licenses/BSD-2-Clause
generated	E57 metadata	2-clause BSD	http://opensource.org/licenses/BSD-2-Clause
generated	Reconstruction shared library & command line tool	GPLv2	http://www.gnu.org/licenses/gpl-2.0.html
generated	Compression shared library & command line tool	GPLv2	http://www.gnu.org/licenses/gpl-2.0.html

License Implications

It shall be noted that some of the used software components restrict usage of the respective software to research purposes (e.g. Primitive Shapes, GCoptimization), or usage in free applications unless a commercial license is obtained (e.g. NVIDIA OptiX). This implies that commercial usage of the software developed by us requires e.g. the re-implementation of some of the functionality provided by the respective libraries and/or the purchase of a commercial license where applicable.

10 Conclusion, Impact and Outlook

We have presented the third version of the software prototype for the recognition of meaningful structures, meaningful shapes, and compression of point cloud datasets. The latest versions of the software prototypes include improvements over the previous versions which account for evaluation results obtained by CITA and through feedback by stakeholders, as well as new features such as the detection of opening directions, column detection, improved quality of reconstructions, improved file export, and more efficient and versatile compression.

Especially during the third and final project year, the growing interest of stakeholders in the industrial and scientific domains regarding the overall aim of the project as well as regaring individual developments of the software prototypes became evident. UBO is currently in contact with industry partners as well as researchers from departments outside of the DURAARK consortium to share research results, and to plan further steps going beyond the end of this project.

References

[1] M. Coatsworth, J. Tran, and A. Ferworn. A hybrid lossless and lossy compression scheme for streaming rgb-d data in real time. In *Safety, Security, and Rescue Robotics (SSRR), 2014 IEEE International Symposium on*, pages 1–6. IEEE, 2014.

[2] J. Digne, R. Chaine, and S. Valette. Self-similarity for accurate compression of point sampled surfaces. In *Computer Graphics Forum*, volume 33, pages 155–164. Wiley Online Library, 2014.

[3] L. Díaz-Vilariño, B. Conde, S. Lagüela, and H. Lorenzo. Automatic Detection and Segmentation of Columns in As-Built Buildings from Point Clouds. *Remote Sensing*, 7(11):15651–15667, Nov. 2015.

[4] T. Golla and R. Klein. Real-time point cloud compression. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2015.

[5] T. Golla, C. Schwartz, and R. Klein. Towards efficient online compression of incrementally acquired point clouds. In *Vision, Modeling & Visualization*. The Eurographics Association, Oct. 2014.

[6] S. Hong, J. Jung, S. Kim, H. Cho, J. Lee, and J. Heo. Semi-automated approach to indoor mapping for 3d as-built building information modeling. *Computers, Environment and Urban Systems*, 51:34–46, 2015.

[7] S. Ikehata, H. Yan, and Y. Furukawa. Structured indoor modeling. *ICCV*, 2015.

[8] H. Macher, T. Landes, and P. Grussenmeyer. Point clouds segmentation as base for as-built bim creation. *ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information Sciences*, 1:191–197, 2015.

[9] O. Mattausch, D. Panozzo, C. Mura, O. Sorkine-Hornung, and R. Pajarola. Object detection and classification from large-scale cluttered indoor scans. In *Computer Graphics Forum*, volume 33, pages 11–21. Wiley Online Library, 2014.

[10] A. Monszpart, N. Mellado, G. Brostow, and N. Mitra. RAPter: Rebuilding man-made scenes with regular arrangements of planes. *ACM SIGGRAPH*, 2015.

[11] V. Morell, S. Orts, M. Cazorla, and J. Garcia-Rodriguez. Geometric 3d point cloud compression. *Pattern Recognition Letters*, 50:55–62, 2014.

[12] V. Morell-Giménez, M. Á. C. Quevedo, S. Orts, and J. García-Rodríguez. 3d data compression for rgb-d data. In *XV Workshop of physical agents: book of proceedings, WAF 2014, June 12th and 13th, 2014 León, Spain*, pages 7–17, 2014.

[13] S. Ochmann, R. Vock, R. Wessel, and R. Klein. Towards the extraction of hierarchical building descriptions from 3d indoor scans. In *EUROGRAPHICS Workshop on 3D Object Retrieval (3DOR 2014)*, Apr. 2014.

[14] S. Ochmann, R. Vock, R. Wessel, and R. Klein. Automatic reconstruction of parametric building models from indoor point clouds. *Computers & Graphics*, 54:94–103, Feb. 2016.

[15] S. Ochmann, R. Vock, R. Wessel, M. Tamke, and R. Klein. Automatic generation of structural building descriptions from 3D point cloud scans. *GRAPP*, 2014.

[16] M. Ruhnke, L. Bo, D. Fox, and W. Burgard. Compact rgbd surface models based on sparse coding. In *Twenty-Seventh AAAI Conference on Artificial Intelligence*, 2013.

[17] M. Ruhnke, L. Bo, D. Fox, and W. Burgard. Hierarchical sparse coded surface models. In *Robotics and Automation (ICRA), 2014 IEEE International Conference on*, pages 6238–6243. IEEE, 2014.

[18] R. Schnabel and R. Klein. Octree-based point-cloud compression. In *Proceedings of Symposium on Point-Based Graphics (SPBG 2006)*, pages 111–120, July 2006.

[19] R. Schnabel, R. Wahl, and R. Klein. Efficient RANSAC for point-cloud shape detection. In *Computer graphics forum*, volume 26, pages 214–226. Wiley Online Library, 2007.

[20] Y. Shi, P. Long, K. Xu, H. Huang, and Y. Xiong. Data-driven contextual modeling for 3d scene understanding. *Computers & Graphics*, Nov. 2015.

[21] C. Thomson and J. Boehm. Automatic geometry generation from point clouds for bim. *Remote Sensing*, 7(9):11753–11775, 2015.

[22] E. Turner and A. Zakhor. Automatic indoor 3d surface reconstruction with segmented building and object elements. *3DV*, 2015.

Glossary

BIM Building Information Modeling

IFC Industry Foundation Classes

LDP Long-term Digital Preservation

RANSAC Random Sample Consensus

RISE Reveal Almost Invisible Structures