DURAARK
DURABLE
ARCHITECTURAL
KNOWLEDGE

D5.5.1 Recognition of Architecturally Meaningful Structures and Shapes

Software prototype v1

DURAARK

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

Date: 2014-01-31
Version 1.0
Document id. : duraark/2014/D.5.5.1/v1.0

SEVENTH FRAMEWORK
PROGRAMME

Grant agreement number	:	600908
Project acronym	:	DURAARK
Project full title	:	Durable Architectural Knowledge
Project's website	:	www.duraark.eu
Partners	:	UBO – Rheinische Friedrich-Wilhelms-Universitaet Bonn [DE]
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/2014/D.5.5.1/v1.0
Title of document	:	Recognition of Architecturally Meaningful Structures and Shapes – Software prototype v1
Deliverable type	:	Software prototype
Contractual date of delivery	:	2014-01-31
Actual date of delivery	:	2014-01-31
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) Henrik Leander Evers `<heve@kadk.dk>` (CITA)
Responsible editor(s)	:	Sebastian Ochmann `<ochmann@cs.uni-bonn.de>` (UBO) Richard Vock `<vock@cs.uni-bonn.de>` (UBO)
Quality assessor(s)	:	Michelle Lindlar `<michelle.lindlar@tib.uni-hannover.de>` (LUH) Jakob Beetz `<j.beetz@tue.nl>` (TUE)
Approval of this deliverable	:	Stefan Dietze `<dietze@L3S.de>` (LUH) – Project Coordinator Marco Fisichella `<fisichella@L3S.de>` (LUH) – Project Manager
Distribution	:	Public
Keywords list	:	Curation, Building Semantics, Access Copies, Structuring, Geometric Enrichment

Executive Summary

This document presents deliverable D5.5.1, the first software prototype for the detection of meaningful structures in point cloud data. This deliverable is part of WP5, "Recognition of Architecturally Meaningful Structures and Shapes". The developed methods and tools provide means for segmenting a point cloud into point subsets that belong to individual rooms as well as for detecting connections between adjacent rooms. The derived semantic information supports several curation tasks in the long-term digital preservation framework. Furthermore, methods for the creation of lightweight access copies of datasets are evaluated.

Table of Contents

1 Introduction

Along with the ever increasing improvements in 3D acquisition hardware over the last two decades, semi-automated 3D-based methods for building documentation have begun to systematically replace manual site measuring. While at first sight the new technology comes with a lot of improvements including foremost short acquisition times and high measuring precision, it also introduces unwelcome implications. First of all, the produced data is large, i.e. depending on the overall size of the building, the size of the scans ranges from a few gigabytes up to a few terabytes. The ongoing trend towards improved acquisition hardware with increasing resolution is likely to further increase the data volume. Additionally, point cloud data is completely unstructured without any semantic or topological information attached.

While sheer size and unstructuredness render point clouds unhandy in general, the combination of these two aspects poses a particular challenge regarding long-term digital preservation. As argued in deliverable 4.4.1, the synchronization between 3D representations taken at different points in time and in possibly varying formats is a crucial ingredient for Long-Term Digital Preservation (LDP) when taking the requirements of the *Designated Community* into account. To this end, semi-automated curation tools are developed in the course of WP4. However, due to the sheer size of point cloud data, the coarse manual prealignment of two representations which is required prior to spatial synchronization can become a cumbersome task. Without exact information about where a recently generated scan fits to an existing Building Information Modeling (BIM) model or previously taken scan, the curator might become lost in a time-consuming task consisting of a series of trial-and-error operations. Additional problems arising from the unstructured nature occur after the spatial synchronization as soon as the difference between two representations is to be computed. Regarding the time horizon of LDP, the interest of the designated community is focused on sustainable long-term changes in the building's structure. However, 3D scans also contain measurements of furniture elements which can be exchanged quick and easy and thereby leading to massive but uninteresting differences between two distinct measurements taken at different points in time. Finally, size as well as lack of structure pose a challenge to several access- and storage-related issues. From a user's perspective, when browsing an archive, the original point clouds are not a very well suited representation to give the user a quick preview on the data due to loading times, which results in a strong need for smaller and handier access copies. Additionally, targeted retrieval and navigation is cumbersome, especially if no corresponding Industry Foundation Classes (IFC) representation exists which is often the case for older buildings, as the data is not segmented in a meaningful way and has no semantics attached.

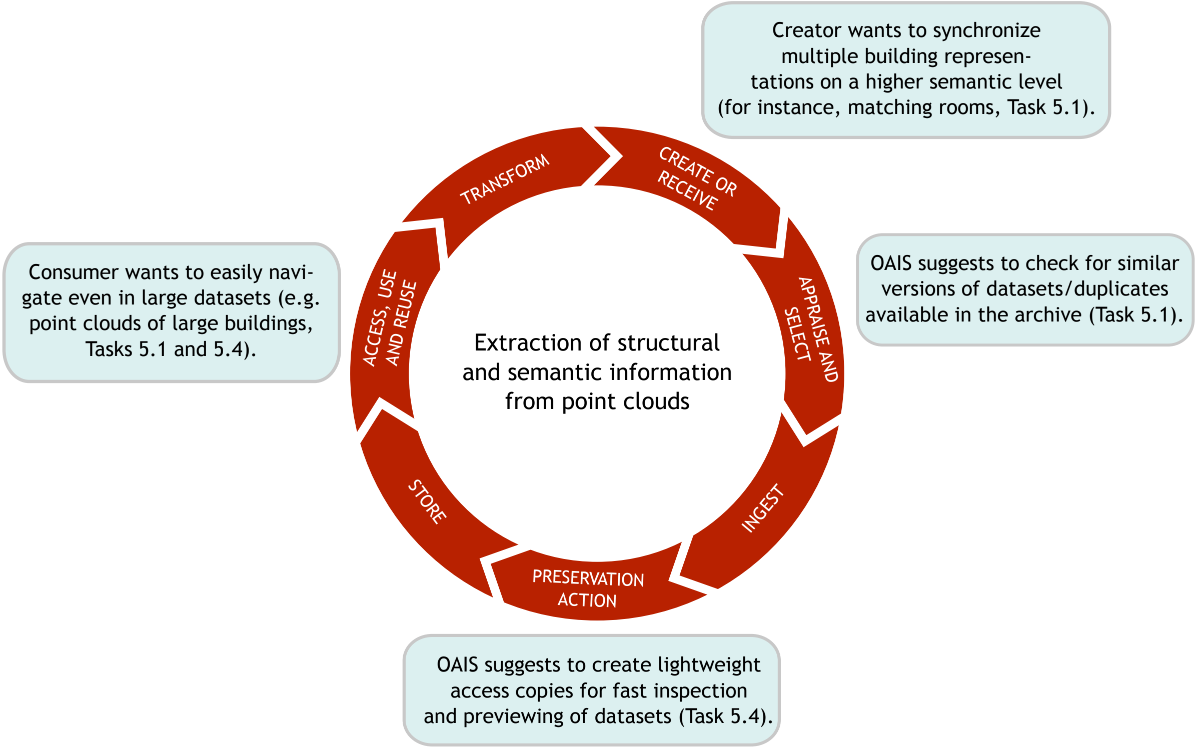


Figure 1: Usage scenarios tackled in WP5 associated with several *sequential actions* in the DCC Lifecycle Model. The Figure uses a part of the model described in [10].

The goal of WP5 is to develop tools that prepare 3D point clouds in a way such that LDP curation tasks can be performed on them as well as to guarantee smooth and easy access. By segmenting the data in an architecturally meaningful way and representing its topology in a compact graph structure (Task 5.1), the curation task of spatial synchronization (see deliverable 4.4.1) is simplified as the graphs allow to quickly generate intial coarse pre-alignment suggestions for the curator. Additionally, navigating the data becomes much easier, as the segmentation into rooms provides a basic semantic structure. Curation is further facilitated by the tools developed as part of Task 5.2 and 5.3 where the recognition of exchangable objects that do not belong to the building's structure are identified such that they can be omitted when comparing representations differing in acquisition time and/or format (Task 4.2). In particular, the structural information extracted using the tools of Task 5.1 may be used as a basis for more fine-grained segmentation and recognition tasks beyond the building structure as demonstrated in our approach[1] in which a hierarchical segmentation of point clouds into storeys, rooms, and objects is obtained. Finally, the development of handy access copies is the focus of Task 5.4.

Figure 1 shows the tasks of WP5 as part of the Digital Curation Centre (DCC) Lifecycle Model [10] which is a standard model for long-term preservation of digital assets. Figure 1 shows a graphical representation of the *sequential actions* described by this model as well as an association of usage scenarios related to WP5 with several of the actions. The scenarios as well as resulting requirements to the developed tools will be further discussed in Section 2.

In this first WP5 deliverable, Tasks 5.1 and 5.4 are tackled. We present the first version of a software prototype[2] for semi-automated segmentation of point clouds into basic structures including storeys and rooms. Based on this segmentation, a compact graph-based representation is extracted that will enable easier synchronization and navigation. Figure 2 shows an overview of the results that have been achieved so far. Regarding the generation of handier access copies, two approaches are currently under examination. On the one hand, lossy compression algorithms that are tailored to 3D point clouds are investigated. Such access copies are useful if a very detailed but yet compact representation is needed. On the other hand, coarse approximations of the point cloud using shape primitives like planes and cylinders are developed. These representations should provide a very quick overview of the actual data when browsing the archive and enable fast streaming of previews even over low-bandwidth networks.

In the remainder of this deliverable, we will first investigate the involved challenges and difficulties more closely. After an overview of the state of the art on indoor point cloud structuring, we present our approach and the according software prototype along with our preliminary results. In the second part of this deliverable, we focus on access copy generation. To this end we will summarize the related work on point cloud compression and discuss a first evaluation of according algorithms on real-world data examples. Furthermore, we will introduce our method based on shape primitives for preview generation

---

[1] Ochmann et al.: Towards the Extraction of Hierarchical Building Descriptions from 3D Indoor Scans, submitted to The Seventh Eurographics Workshop on 3D Object Retrieval (3DOR 2014), in review at the time of this writing.

[2] **The software prototype may be obtained from the FTP site `ftp://ftp.cg.cs.uni-bonn.de` (anonymous login, file path `/pub/outgoing/duraark/Software_Prototype_v1_D441_D551.zip`) which has been tested on Windows 7 (64-bit). For downloading example datasets, user credentials are required which will be provided to reviewers separately.**

and present our first results.

Note that part of this deliverable's results have been presented as a paper [17] at the GRAPP 2014 conference.

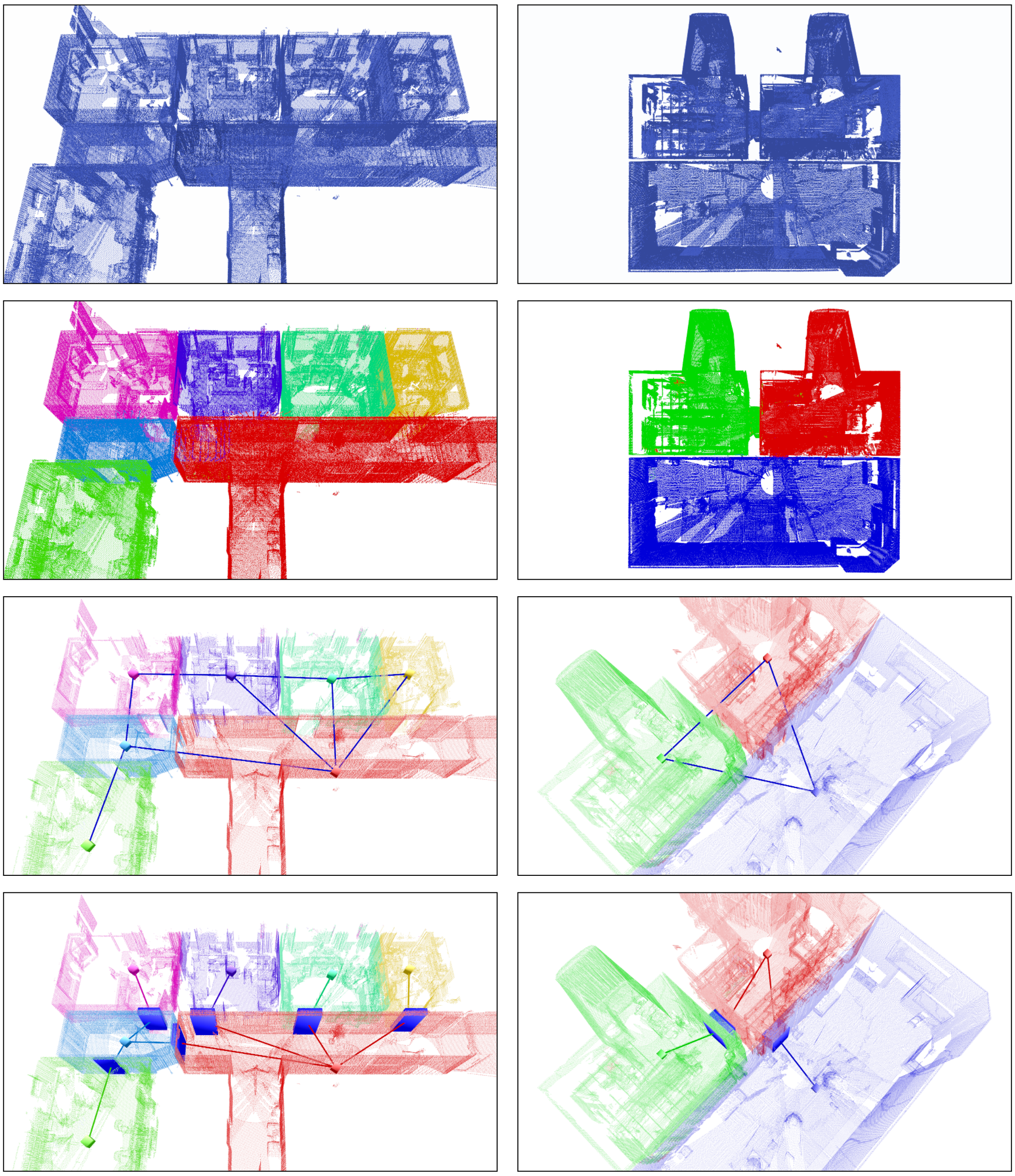


Figure 2: Example results of the developed point cloud segmentation and topology extraction methods. The left column shows a subset of scans of the *Risløkka Trafikkstasjon* (Norway), the right column shows a subset of scans of *Kronborg castle* (Denmark). First row: Original input point clouds. Second row: Room assignment after segmentation. Third row: Detected room neighbourhoods encoded in a graph structure. Fourth row: Detected doors between adjacent rooms.

2 Problem Description and Requirements

Long-term preservation of digital assets does not only incorporate the mere storage of datasets in an archival system but also includes methods and strategies for managing and working with the stored data. Archival of 3D building representations (BIM models, point cloud scans) poses new challenges concerning curation tasks in the scope of the long-term preservation framework. In particular, the unstructuredness of point cloud data and its data size makes handling them cumbersome.

2.1 Usage Scenarios

The developed software component allows the extraction of high-level building descriptions from point cloud scans. In addition to segmenting scans into point subsets belonging to individual rooms, a concise topological representation of the rooms and connections between them is obtained. The gained semantic information helps with performing several curation tasks (see also Figure 1).

- During the *Create or Receive* phase of the preservation lifecycle, a content creator may have multiple different representations of the same building available (for instance, a point cloud and a BIM model). The developed tools may be used to synchronize those representations on a higher semantic level, for instance matching rooms detected in the point cloud with rooms in the BIM model.

- In the *Appraise and Select* phase, the semantically rich building descriptions extracted using the developed tools may be used to search for similar versions of the datasets which may already be stored in the archive.

- The creation of lightweight access copies may be implemented as a *Preservation Action* in order to enable visual inspection of the stored three-dimensional data in an efficient manner. This also includes previewing of datasets even over low-bandwidth network connections.

- Easy navigation within datasets in the *Access, Use and Reuse* phase, especially in buildings which consist of many rooms and storeys. For instance, by using the extracted semantic structure of a building a subset of rooms may be hidden or highlighted in order to improve comprehensibility of the data.

This deliverable addresses several use cases as described in the Requirements Document D2.2.1:

- Use Case 2 – Search and retrieve archived objects,
- Use Case 4 – Detect differences between as-planned and as-built state,
- Use Case 5 – Monitor the evolution of a structure over time, and
- Use Case 7 – Plan, document and verify retrofitting/energy renovation of buildings.

2.2 Relation to the DURAARK System Architecture

The software components developed in the course of WP4 and WP5 are designed to be modular software tools that can be used on the client computer at any stage when enrichment, analysis or comparison of different geometric representations of building data (either acquired by the user or fetched from the LDP system) is requested. In this context the most efficient and generic way to interface with the rest of the DURAARK system architecture is by means of file input/output. For an overview of the system architecture and how it relates/interfaces to the software components developed in WP4/WP5, see Figure 3.

2.3 Requirements

This section gives an overview of the requirements of the developed software component from a usability and technical perspective. Section 4 describes how these requirements were met in the first software prototype.

2.3.1 Usability Requirements

The tasks pointed out in Section 2.1 pose several requirements to the developed structural analysis component. The user needs to be able to select and load the desired point cloud dataset for which a structural description shall be generated into an intuitive graphical user interface which is accessible also to inexperienced users. Navigation in 3D space is realized using user interface paradigms well-known by many users. Subsequently, the user shall be guided through the process of extraction of structural information from the data in which automated or semi-automated methods shall be provided where possible.

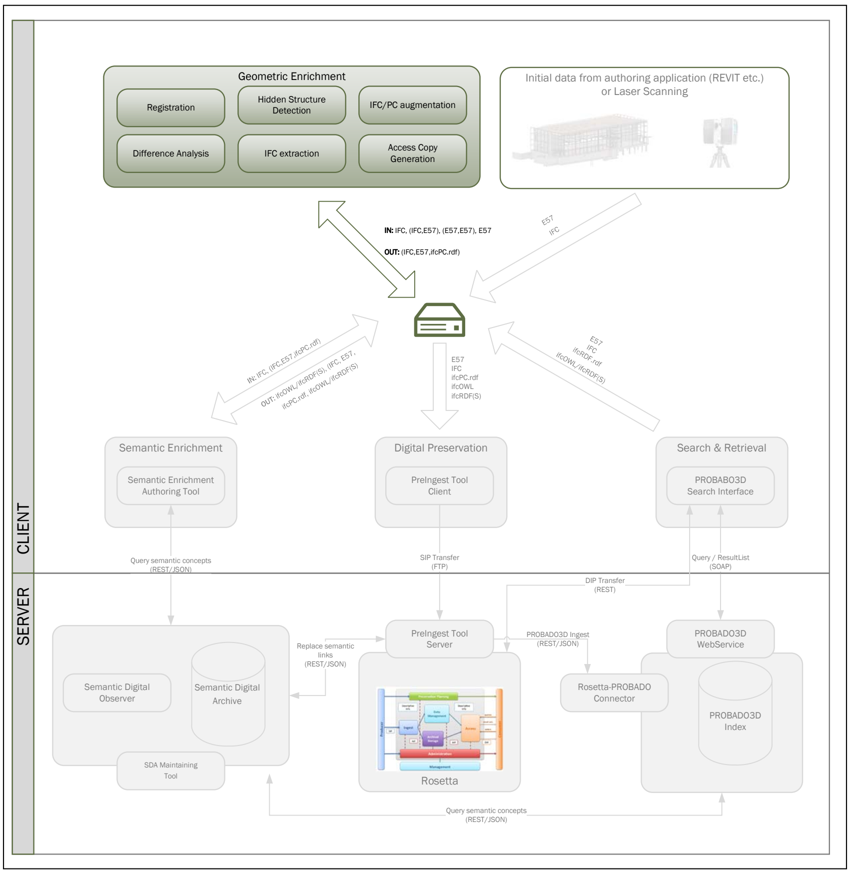


Figure 3: Overview of the current system architecture design for the DURAARK project. The Geometric Enrichment section in the upper left corner gives an overview of the WP4/WP5 software components and their interfaces/relations to other DURAARK framework components.

The user is supported by user interface elements grouped into sections which provide tools for each sub-task. In addition to automated processing tools, means to manually post process the results need to be included in order to tweak the generated results as necessary or desired. Finally, means to store the results in a suitable format must be provided. This is achieved by providing the user with conventional output file selection tools.

2.3.2 Technical Requirements

This software component deals with the processing of point cloud data for which the E57 file format [1] was selected for usage in the scope of the DURAARK project as it provides means to store multiple scans and associated metadata in a compact manner. The component needs to be able to load data stored in this format, including metadata such as scanning positions and subsets of points belonging to different scans. In addition, the software needs to be able to cope with the large amounts of data as is often encountered when working with point cloud data. This includes providing suitable methods for viewing the datasets as well as for processing them internally. Lastly, the resulting structural representation needs to be exported in a suitable format for storage in the LDP system alongside with the point cloud data.

3 Related Work

This Section summarizes the related work on structural and semantic analysis of digital representations of architectural content. The presented approaches may be categorized in two different classes. The first class of methods deals with reconstruction of 3D mesh representations from unstructured 3D point clouds. Such approaches are an important ingredient for transforming low-level architectural data to a higher semantic level (Task 5.1) which is a major prerequisite for enabling curation. They also allow a more compact representation of the data and therefore are important for the generation of previews of the archived data (Task 5.4). The second class of approaches focuses on the generation of higher-level structural and topological representations of buildings from various different representations (Task 5.1). While some of the presented methods use a different type of input data than 3D point clouds which are – apart from IFC models – the mainly targeted representation within the DURAARK project, they nevertheless provide insight to the underlying structuring task and give hints towards efficient methods for navigating in the data, which is a major prerequisite for curation.

Oesau et al. [19] present an approach for the generation of watertight meshes from point clouds representing multi-storey buildings. Building structures including walls, floors, and ceilings are assumed to be planar. Starting with an over-segmentation in polyhedral cells, each cell is classified as "inside" or "outside" by means of ray tracing operations. Subsequently, a plausible mesh representation of the building is determined using a Graph-Cut algorithm. A hybrid approach is presented by Xiao and Furukawa [26]. The authors use 3D point cloud data together with ground-level photographs to reconstruct a CSG representation based on fitted rectangle primitives. The method uses an "Inverse-Constructive-Solid-Geometry" approach with which the *seen empty* space within the building is determined. Note that this work rather focuses on indoor scene reconstruction than on the actual extraction of a semantic structuring. The work by Jenke et al. [11] proposes a method for reconstruction of mesh representations from unorganized point clouds by aggregation of cuboid primitives. The task of finding a constellation optimally explaining the given data is formulated as a maximum-a-posteriori problem taking into account matching of surfaces as well as connectivity between candidate cuboids. A solution is determined by means of a greedy algorithm. The method proposed by Budroni and Boehm [7] extracts polygonal floor plans from 3D point clouds. Floor and ceiling structures and – subsequently – wall structures are detected using translational and rotational sweeps. By projecting detected wall structures onto the horizontal plane, a segmentation of the plane in cells is obtained. After extraction of cells which belong to the "inside" of the building, the outer circumfence of the union of these cells is extracted which provide an approximation of the building's ground plan. Adan and Huber [4] present a method for reconstruction of floors, ceilings, walls as well as openings (doors, windows) even if the structures are partially occluded. After detection of planar structures, regions within these structures are labeled as "seen", "occluded" or "empty". A refinement of this labeling is obtained using a region growing approach. Detection of openings is performed by a Support Vector Machine (SVM) based learning approach. In the approach by Sanchez and Zakhor [20], buildings are reconstructed as a combination of planar structures and parameterized staircase models. Using region growing, planar structures are determined and (concave) polygonal approximations are generated using Alpha Shapes. In the set of remaining points (i.e. points which do not belong to extracted polygons), parameterized staircase models are fitted.

Ahmed et al. [5] use 2D floor plans to extract the structure and semantics of buildings. The input drawing is first segmented into graphical parts of differing line thickness used to extract the geometry of rooms by means of contour detection and symbol matching, and a textual part used for semantic enrichment by means of OCR (optical character recognition) and subsequent matching with predefined room labels. Wessel et al. [25] extract topological information from low-level 3D Computer-Aided Design (CAD) representations of buildings. They find floor planes in the input polygon soup and extract 2D plans by cutting each storey at different heights. These cuts are then analyzed in order to extract rooms and inconsistencies between different cut heights yield candidates for doors and windows. In the approach by Langenhan et al. [13], high-level BIM models are used for the extraction of a building's topology. They argue that explicit information about the topological relationship between rooms are not explicity contained within an IFC (Industry Foundation Classes) file. However, a derivation of this information is possible by analysis of certain entity constellations in the IFC model and can be used to perform sketch-based queries in a database.

Hichri et al. [9] give a good overview of current point cloud to BIM (as-built BIM) approaches as well as a classification of different aspects and requirements of such a process, taking into account the specific needs for the generation of semantically rich BIM models. The authors describe the characterization of shapes, relations and attributes as the basic requirements that need to taken care of in an as-built framework. Furthermore, they classify current approaches into the categories of heuristic, context-based, prior-knowledge-based and ontology-based approaches.

Mura et al. [15] extract a polyhedral model from cluttered indoor scans. In constrast to other approaches which focus on the segmentation of a scene into objects contained therein, the method aims to extract a building model which is segmented on a per-room level. During the generation of wall candidates, explicit reasoning about occluded parts of walls is taken into account in order to recover occluded regions. The model is subsequently extracted using a two-dimensional cell complex created from a projection of the wall candidates onto the x-y-plane. By modeling the affinity of neighboring line segments in this cell complex using diffusion maps in an embedding space, a segmentation into rooms is achieved by means of k-medoid clustering. Finally, the polyhedral model is constructed by intersecting the resulting walls with floor and ceiling planes.

Turner and Zakhor [24] present a method for creating 2D floor plans and 2.5D building models including a segmentation into rooms from data captured by a mobile mapping system or from 3D point clouds. One of their main goals is the creation of simplified models for usage in energy simulation tasks, but the resulting plans and room labeling may also be used in other architectural contexts. In a first step, a 2D point map is created in which each point represents a wall sample. Each sample point is assigned the scanner position from which it was captured. In a second step, a Delaunay triangulation of the sample point set is computed. Each triangle is subsequently labeled as "exterior" or "interior" space. Each triangle is checked for intersections with the line-of-sight between the sample points and the associated scanner positions and labeled as "interior" if a line-of-sight intersects the triangle. If the scans were acquired using a mobile scanning device and an ordering of the scans in the time domain is available, the line segments between subsequent scanning positions is also used for checking for triangle intersections. The third step assigns room labels to the triangles such that triangles belonging to the same room receive the same label. A Graph-Cut optimization is formulated on the dual graph of the Delaunay triangulation in which the lengths of shared sides of neighbouring triangles are used as edge weights. Together with a guess of seed triangles for each room, an initial segmentation into rooms is obtained which is usually an over-segmentation. A refinement step is performed in which redundant room labels (i.e. multiple labels belonging to the same room) are removed and the labeling process is repeated until convergence. After an optional simplification step, the resulting floor plan is extruded to an estimated height of the rooms which yields a labeled 2.5D model of the building.

4 Workflow and Implementation Details

This section describes the developed software prototype from a user's perspective as well as implementation details. The current prototype guides the user through the creation of a segmentation of a point cloud into semantically meaningful parts (rooms) and the detection of connections (e.g. doors) between rooms. The resulting semantic building descriptions may be used to perform the tasks outlined in section 2. A visual outline of the usage workflow is illustrated in Figure 4. For a detailed step-by-step description of the software prototype usage see the manual section in Appendix A.

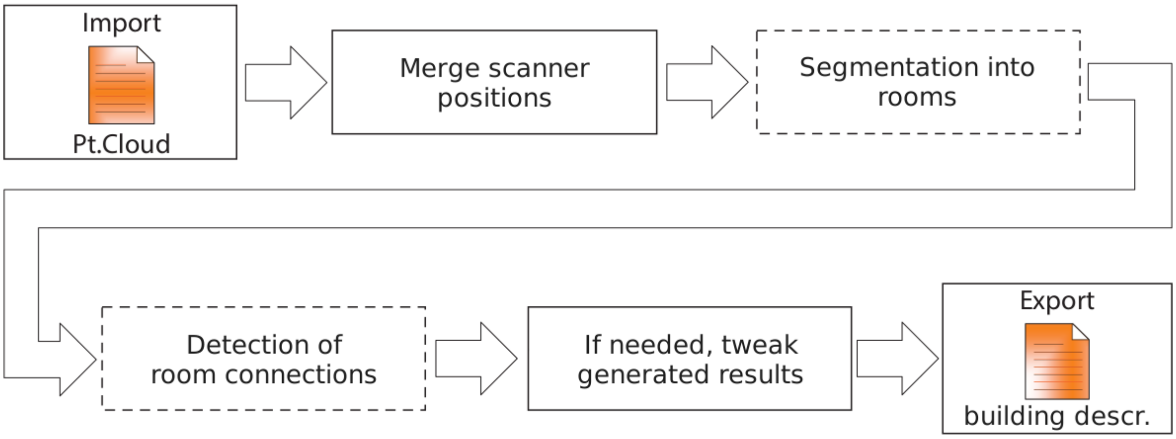


Figure 4: Flow diagram showing the steps of the structure analysis workflow.

4.1 Workflow

In a first step, the user loads a point cloud dataset into a graphical user interface which supports the user with the process of generating the desired structural building description in an intuitive manner. The loaded 3D point cloud is visualized in an interactive, three-dimensional viewport as shown in Figure 5.

As a starting point for determining the building's structure and topology, a semi-automated approach is made available in the software. The goal of this process is twofold. Firstly, overlaps between scans of different rooms shall be resolved such that each point is uniquely assigned to the room to which it belongs. This results in a segmentation of the point cloud into point subsets belonging to the individual rooms. Secondly, this segmentation is used to detect openings between adjacent rooms.

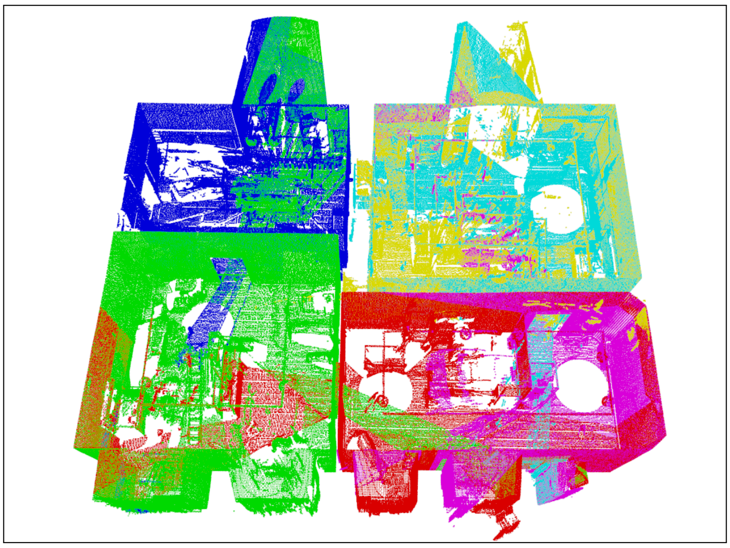


Figure 5: A point cloud loaded into the software prototype. The colors indicate the association of points to individual scans taken in the building. Note that overlaps between different scans result in an association that initially does not represent individual rooms of the building.

The implemented method exploits information about scanner positions and point subsets assigned to the individual scans in order to obtain an initial guess which points belong to which room. This initial guess is improved in a subsequent, automated process. A prerequisite is that multiple scans taken within the same room are merged together. Methods for automating this process are currently being evaluated, however this step needs to be performed manually in the current version of the software. For this step, the user tells the software which scanner positions shall be merged (i.e. are located within the same room) by selecting them interactively within the point cloud. Figure 6 shows an example for this process.

After the scanner positions have been merged, the automated segmentation and opening detection process may be initiated. After the computation is done, the user is presented with a visual representation of the results. Figure 7 shows the point-to-room association after the segmentation process as well as automatically detected connections between rooms. The user may inspect the resulting structures visually and make changes to the results if needed (e.g. add or remove connections which have been wrongly detected). More technical details of the approach are available in [17] and [16]. Figure 8 shows additional results on more real-world datasets.

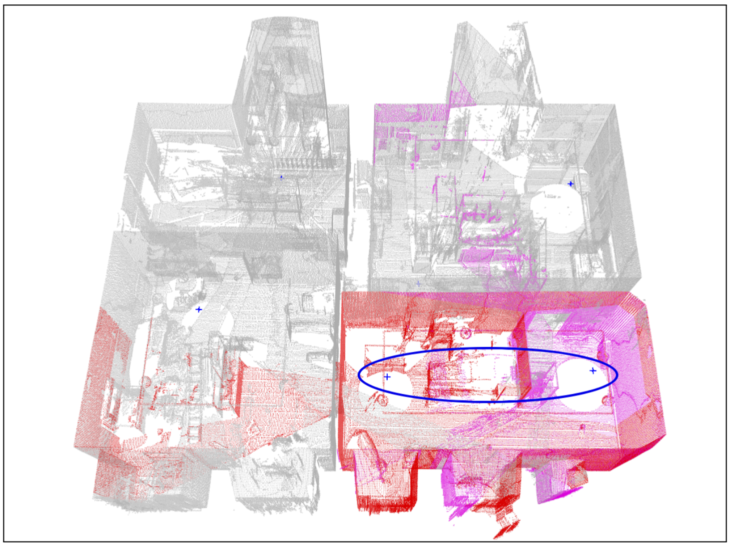


Figure 6: Merging of scanner positions. The two highlighted positions have been selected for merging because they belong to the same room.

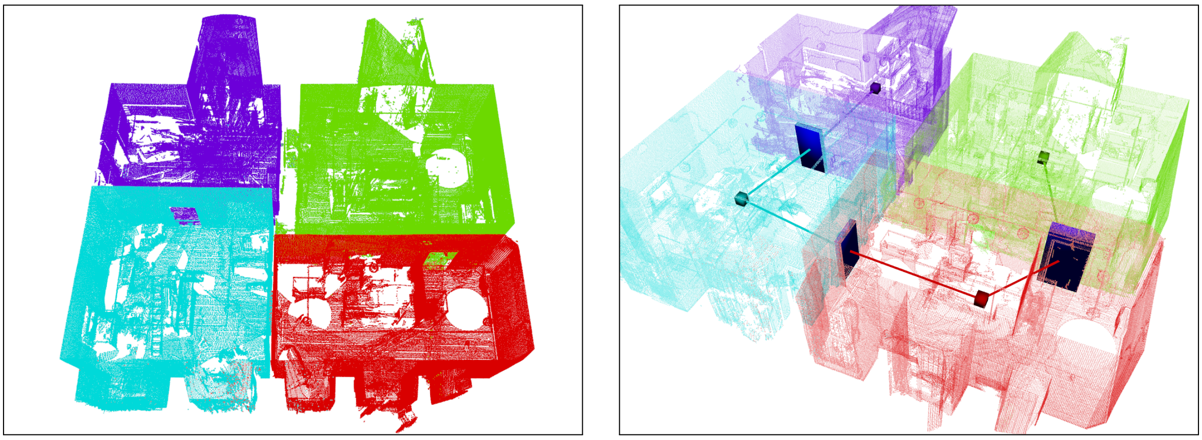


Figure 7: Left: Point-to-room association after the segmentation step. Right: Automatically detected doors between adjacent rooms.

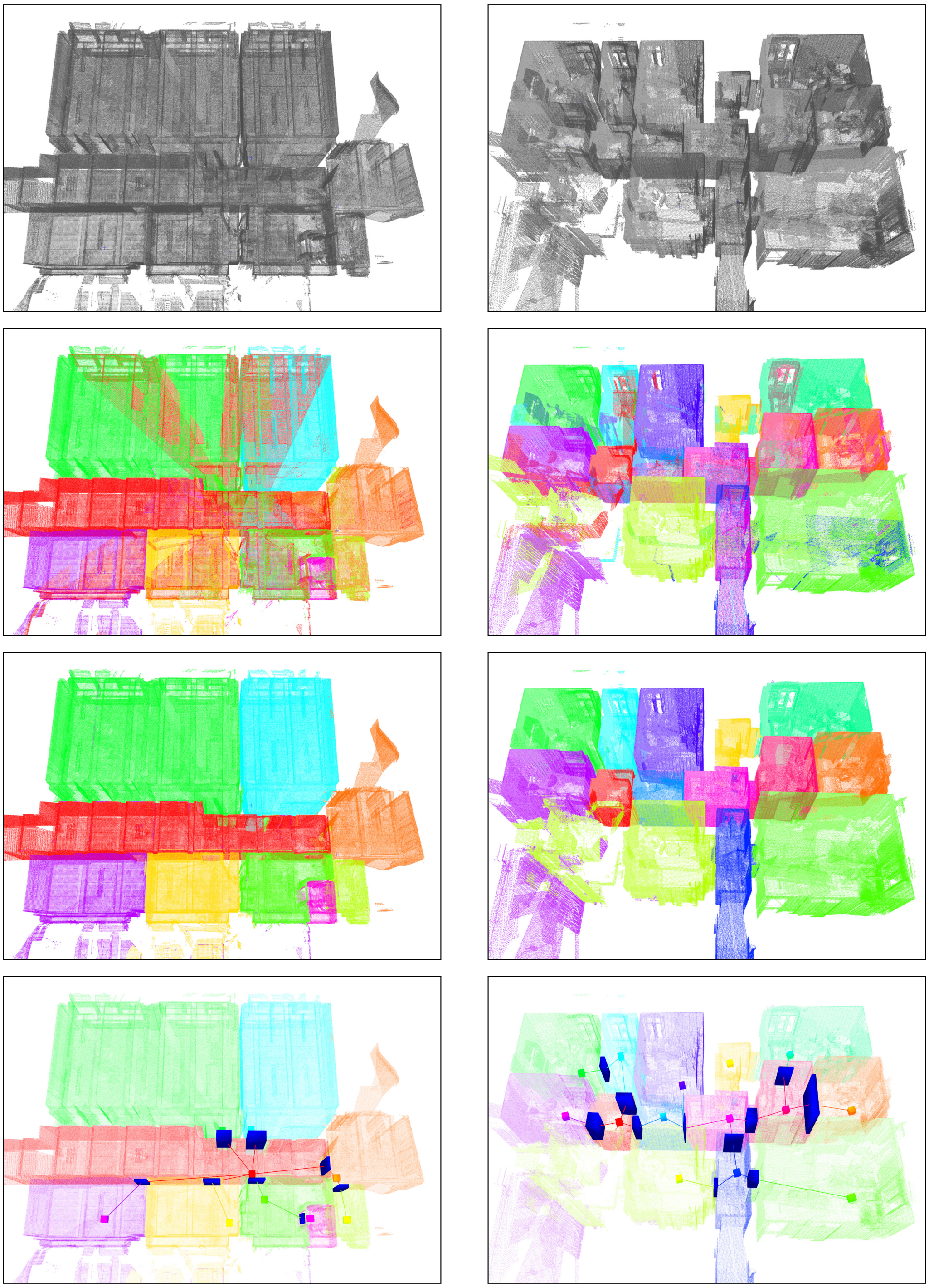


Figure 8: Example results on two real-world datasets. From top to bottom: Input point cloud, room labels after the merging step, room labels after the relabeling process, detected room connections.

4.2 Implementation Details

For loading E57 point cloud files, the open-source libE57 library [2] has been chosen. In addition to providing programming interfaces for accessing the data part of the stored point cloud (e.g. point coordinates), it also allows access to associated metadata such as scanner positions and subsets of points belonging to individual scans. This information is an important ingredient for the implemented automated segmentation and opening detection. For working with point cloud data internally once it has been loaded into memory, the Point Cloud Library (PCL) [3] is used as it provides efficient data structures and algorithms for processing point clouds.

The output format of the computed structural representations is based on the *Resource Description Framework* (RDF) family of specifications. This standard provides us with universal and versatile means for interlinking stored datasets or parts therein. See document D2.2.3 for further information as well as a concrete example.

5 Access Copy Generation

Current scanning devices available on the market produce very dense, high quality measurements of the scene captured. In the context of architectural scenes, a rather large amount of outdoor, as well as indoor scans are necessary in order to concisely and completely capture the state of a building. Therefore the size of the captured point data in its uncompressed state is not practical for delivery over e.g. network channels.

Observing the current development of scanning devices and methods, the amount of data per scan (e.g. resulting from high-quality laser scans) as well as the amount of scans per object (e.g. resulting from stereoscopic video data analysis) are expected to rapidly grow in the near future.

The scope of a research project in the field of long-term preservation (like DURAARK) should therefore include fast and efficient compression methods in order to reduce data size while maintaining a sufficient level of accuracy.

In order for a compression algorithm to be suitable for long-term preservation, the loss (i.e. error) introduced by the algorithm should be bounded by the uncertainty of the scanning process itself. For example when considering the depth of measured points an algorithm should not introduce any error in terms of depth uncertainty that exceeds the scanning depth accuracy of the device used. This is particularly important for performing tasks like taking measurements in the data.

Additionally any compressed output should be as robust to bit level corruption as possible and contain redundancy where necessary in order to avoid loss by bit flip influence.

Lastly, any considered algorithms have to be analyzed for necessary assumptions on input data. For example the necessity to have organized or regular data might render an algorithm unsuitable for any practical application.

5.1 State-of-the-Art Analysis

Several methods for compressing point cloud data have been proposed. Almost all of them rely on the fact, that scanned points can be considered as discrete samples of an underlying surface. Approximating this surface and representing points in relation to this surface yields efficient ways of reducing the storage complexity of point data without substantial loss of accuracy.

A few methods are based on the *moving least squares* (MLS) surface definition developed by Fleishman et al. [8] which locally describes the surface around a point based on the

point's neighborhood. Alexa et al. [6] for example propose reducing the input to a base point set used for triangulation and then use well studied mesh compression algorithms. Where necessary additional points are then inserted into the base point set, projected onto the MLS surface and consequently only the differences are compressed and stored.
Another approach relying on MLS surface approximation is proposed by Ochotta et al. [18]. They resample the MLS surface using planar height fields obtained beforehand by partitioning the point cloud and compress the resulting resampled data using image based techniques.

Both of these methods are lossy compression algorithms and suffer from the fact, that approximating the surface by means of MLS smoothes out sharp features. Since architecture (and to some degree most man-made objects) primarily consists of flat surfaces with sharp edges, this drawback renders methods based on MLS surface approximations unsuitable for long-term preservation of scanned building data.

Another class of methods use spatial organization of input data by regularly subdividing the space and using the resulting cells to compress the contained data. For example Krüger et al. [12] propose a lossy compression of point and normal data by means of *closest sphere packing grids* (CSP grid) which are a specialized spatial data structure derived from the concepts in *closest sphere packing theory* [23]. Depending of whether a cell in the CSP grid contains at least one point, the cell is considered full, otherwise it is considered empty. The algorithm then encodes runs of full cells for each slice of the grid.
Schnabel et al. [21] propose a compression scheme based on the encoding of an *octree* data structure. At first, the coordinates of the input cloud are inserted into the octree structure, and then every octree leaf cell center is considered a point in the compressed point cloud. Consequently only the structure of the octree needs to be encoded, compressed and saved. By altering the encoding scheme and the maximum octree depth, the loss of the outcome can be controlled, up to the point of a virtually lossless compression.

Finally, there are compression schemes that exploit certain assumptions that may (depending on the acquisition context) be made about the scan data. For example Merry et al. [14] propose a method based on *spanning trees* that yields very high compression rates with low error margins when the point cloud is dense and regular (i.e. with uniform point density). However none of these assumptions can be made in the context of the DURAARK project.

From the methods mentioned (and related state-of-the-art methods) only the work done by Schnabel et al. currently fits the requirements for a long-term preservation context (as it is given in the DURAARK project). This conclusion stems from the observation, that it is (optionally) *lossless*, given the assumptions valid for the input data yields the *highest compression ratios* and is potentially extensible to the point where other *related data* (as for example point color values and normals) may also be compressed along with the point data.

5.2 Evaluation of Compression Results

Two approaches for the creation of compressed access copies are currently being evaluated. The first approach uses the octree-based compression method proposed by Schnabel et al. [21]. Compression results have been evaluated for different octree levels (i.e. maximum depths) which is the parameter that controls the trade-off between precision and file size of the compressed point cloud. A higher value of the level parameter yields a higher spatial precision of the resulting compressed point cloud at the cost of larger file size. Table 9 shows an overview of compression results expressed as the average number of bits per point that are required to represent the data for different level parameters. Figure 10 shows a visual comparison of the resulting point clouds compressed with different parameters.

Level	Bits per point
Original	96
9	1.9685
7	0.13236

Figure 9: Compression results.

The second method for creating access copies replaces large planar structures within the point clouds by bitmap representations using a RANSAC implementation by Schnabel et al. [22]. The rationale behind this approach is that large planar structures, as are often encountered in buildings, bear little detail while at the same time a large number of points is required for representing them in a point cloud. Replacing these structures by a more light-weight representation reduces the file size while preserving the general structure of the building. An example result is shown in Figure 11. Large planar structures with few details have been automatically replaced by lightweight bitmap approximations while details (marked red in the Figure) have been retained.

5.3 Framework Interface Requirements

It is one goal of the compression component to solely rely on intrinsic properties of the input data given. Therefore in terms of the DURAARK system framework the compression component may be regarded as a closed ("black box") component, whose input is the uncompressed point cloud data as well as parameters, and output is the compressed point cloud. The same (with reversed input/output roles) is true for the decompression component. Hence, the only framework interface requirement for these components is a fixed definition of input/output data formats as well as data transmission channels.

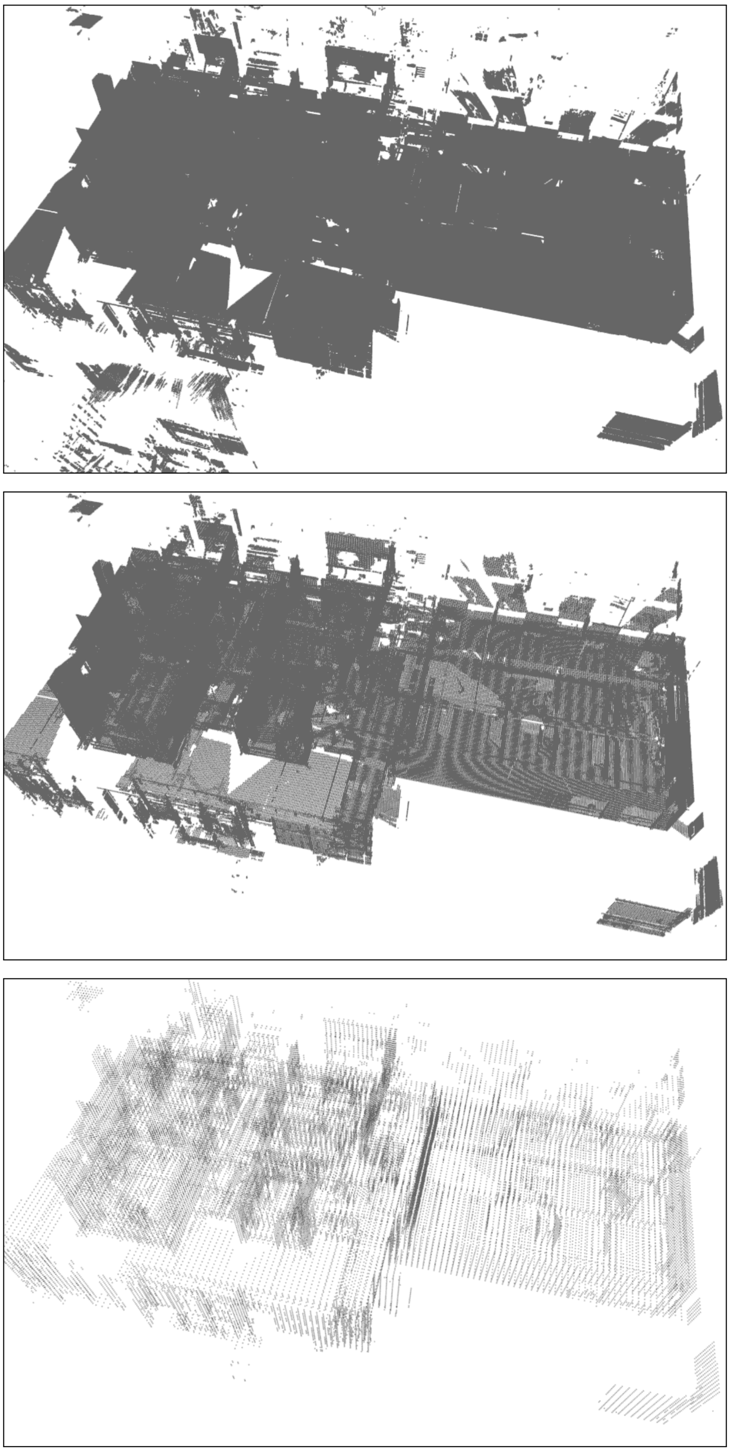


Figure 10: Comparison of different levels of compression.

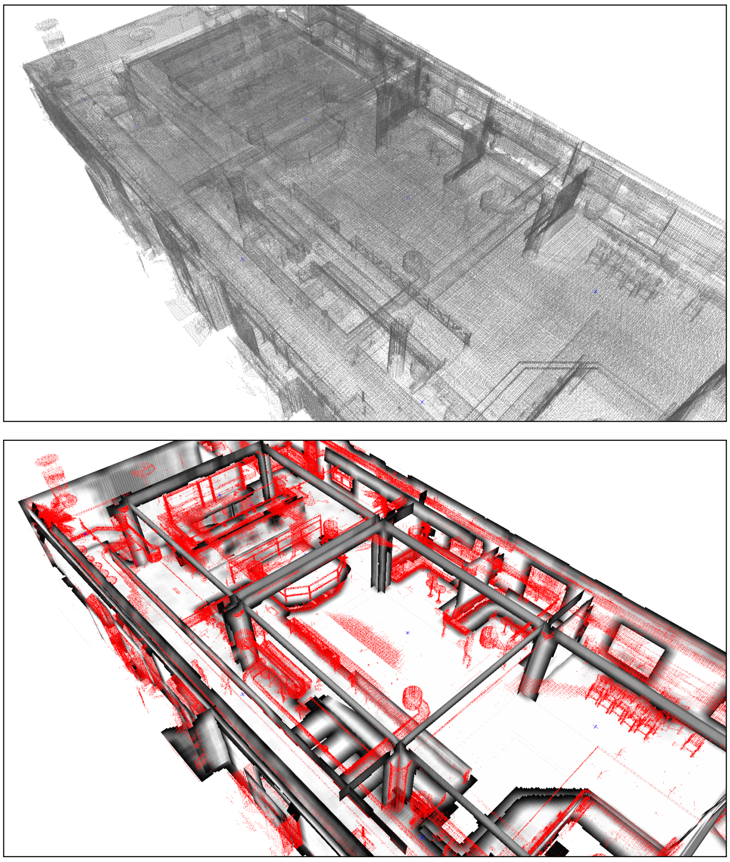


Figure 11: An example for a simplification of a dataset for providing a more lightweight access copy. Left: An unmodified point cloud representation of a building's interior. Right: Large planar structures have been replaced by bitmap representations. Note that bitmaps belonging to ceiling structures have been removed for visualization. In this example, the bitmaps replaced about 72% of the points. Details which were not replaced by bitmaps are marked red.

6 Summary

The software prototype developed in the course of this deliverable provides means to generate structural and topological representations of buildings from 3D point cloud scans in a semi-automated manner. The user is guided through the process by an intuitive and interactive graphical user interface and is supported by a novel, automated approach for segmenting a point cloud into rooms and detecting openings. The generated results may be post processed by the user if necessary or desired. Together with further shape detection methods which will be evaluated and developed in the course of WP5 (Tasks 5.2, 5.3), the resulting structural building descriptions are an important prerequisite for performing curation tasks in the scope of the long-term archival framework. Furthermore, the developed methods may be used to complement the approaches for registration and difference detection developed in WP4 by providing for example automatic alignment suggestions.

The evaluation of compression methods provides an insight into approaches which will be further developed in the course of WP5 (Task 5.4). Compression is a crucial ingredient for providing access copies of large datasets which may be displayed and transferred in an efficient manner.

References

[1] *E57 standard.* `http://www.astm.org/Standards/E2807.htm`.

[2] *libE57.* `http://www.libe57.org/`.

[3] *Point Cloud Library (PCL).* `http://www.pointclouds.org/`.

[4] A. Adan and D. Huber. 3D reconstruction of interior wall surfaces under occlusion and clutter. In *3D Imaging, Modeling, Processing, Visualization and Transmission (3DIMPVT), 2011 International Conference on*, page 275–281, 2011.

[5] S. Ahmed, M. Liwicki, M. Weber, and A. Dengel. Automatic Room Detection and Room Labeling from Architectural Floor Plans. In *Document Analysis Systems*, pages 339–343, 2012.

[6] M. Alexa, J. Behr, D. Cohen-Or, S. Fleishman, D. Levin, and C. T. Silva. Computing and rendering point set surfaces. *Visualization and Computer Graphics, IEEE Transactions on*, 9(1):3–15, 2003.

[7] A. Budroni and J. Boehm. Automated 3D reconstruction of interiors from point clouds. *International Journal of Architectural Computing*, 8(1):55–73, 2010.

[8] S. Fleishman, D. Cohen-Or, M. Alexa, and C. T. Silva. Progressive point set surfaces. *ACM Transactions on Graphics (TOG)*, 22(4):997–1011, 2003.

[9] N. Hichri, C. Stefani, L. De Luca, P. Veron, and G. Hamon. From Point Cloud To BIM: A Survey of existing Approaches. *ISPRS - International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, XL-5/W2:343–348, 2013.

[10] S. Higgins. The DCC Curation Lifecycle Model. *International Journal of Digital Curation*, 3(1):134–140, 2008.

[11] P. Jenke, B. Huhle, and W. Straßer. Statistical reconstruction of indoor scenes. In *Proc. WSCG*, volume 9, 2009.

[12] J. Kruger, J. Schneider, and R. Westermann. Duodecim-a structure for point scan compression and rendering. In *Point-Based Graphics, 2005. Eurographics/IEEE VGTC Symposium Proceedings*, pages 99–146. IEEE, 2005.

[13] C. Langenhan, M. Weber, M. Liwicki, F. Petzold, and A. Dengel. Graph-based retrieval of building information models for supporting the early design stages. *Advanced Engineering Informatics*, May 2013.

[14] B. Merry, P. Marais, and J. Gain. Compression of dense and regular point clouds. *Computer Graphics Forum*, 25(4):709–716, 2006.

[15] C. Mura, O. Mattausch, A. Jaspe Villanueva, E. Gobbetti, and R. Pajarola. Robust reconstruction of interior building structures with multiple rooms under clutter and occlusions. In *Proceedings of the 13th International Conference on Computer-Aided Design and Computer Graphics*, November 2013.

[16] S. Ochmann. Extraktion semantischer Repräsentationen von Gebäuden aus 3D Innenraum-Laserscans. Diplomarbeit, Universität Bonn, November 2013.

[17] S. Ochmann, R. Vock, R. Wessel, M. Tamke, and R. Klein. Automatic generation of structural building descriptions from 3d point cloud scans. In *GRAPP 2014 - International Conference on Computer Graphics Theory and Applications.* SCITEPRESS, Jan. 2014.

[18] T. Ochotta and D. Saupe. Compression of point-based 3d models by shape-adaptive wavelet coding of multi-height fields. In *Proceedings of the First Eurographics conference on Point-Based Graphics*, pages 103–112. Eurographics Association, 2004.

[19] S. Oesau, F. Lafarge, and P. Alliez. Indoor Scene Reconstruction using Primitive-driven Space Partitioning and Graph-Cut. In *Eurographics Workshop on Urban Data Modelling and Visualisation*, 2013.

[20] V. Sanchez and A. Zakhor. Planar 3d modeling of building interiors from point cloud data. In *Image Processing (ICIP), 2012 19th IEEE International Conference on*, page 1777–1780, 2012.

[21] 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.

[22] R. Schnabel, R. Wahl, and R. Klein. Efficient ransac for point-cloud shape detection. *Computer Graphics Forum*, 26(2):214–226, June 2007.

[23] N. J. Sloane, J. Conway, et al. *Sphere packings, lattices and groups*, volume 290. Springer, 1999.

[24] E. Turner and A. Zakhor. Floor plan generation and room labeling of indoor environments from laser range data. In *Proceedings of the 9th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications (GRAPP 2014)*, January 2014.

[25] R. Wessel, I. Blümel, and R. Klein. The room connectivity graph: Shape retrieval in the architectural domain. In *The 16-th International Conference in Central Europe on Computer Graphics, Visualization and Computer Vision*, 2008.

[26] J. Xiao and Y. Furukawa. Reconstructing the world's museums. In *Computer Vision–ECCV 2012*, page 668–681. Springer, 2012.

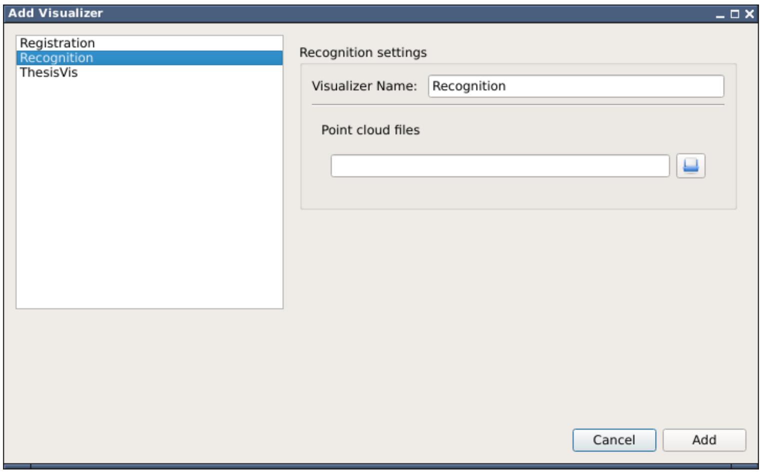
A Prototype Software Usage Manual

A.1 Prerequisites and Installation

Before using the prototype software it is recommended to update the graphics card driver(s) to the newest stable version. Apart from this there are no steps necessary to use the software - it is sufficient to simply copy the software folder to a local folder and execute **Recognition.bat** with user privileges. This automatically starts the modular visualization software in a setting tailored to the task of structural analysis of indoor point clouds.

A.2 First Steps and Camera Navigation

After starting the software for room segmentation and door detection in point clouds, the user is greeted with a dialog as depicted in Figure 12. Here the user may choose



Recognition settings

Visualizer Name: Recognition

Point cloud files

Figure 12: The user is greeted with a dialog for selecting point cloud files.

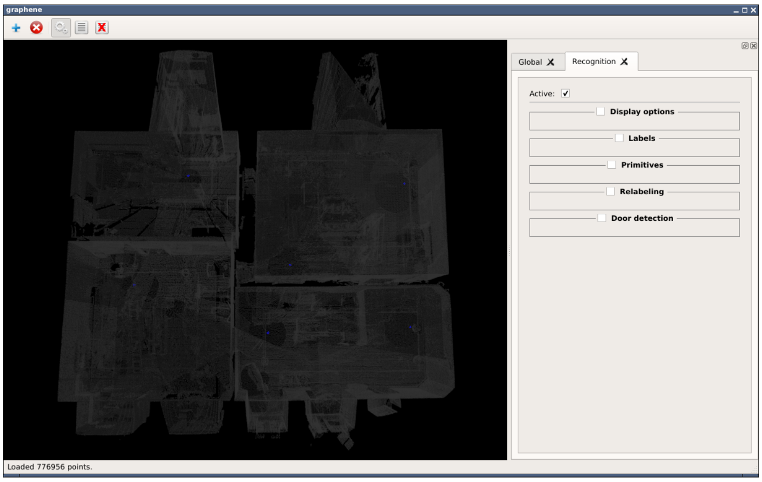
multiple (registered) point cloud scans of a building for which the computations shall be performed. In the current version, *.pcd* point cloud files are supported[3]; example files are included in the software prototype package. In the file chooser dialog, *Ctrl+click* may be used to select multiple point cloud files.

After the point cloud files have been selected, they are loaded into the software and displayed in the 3D viewport similar to Figure 13. In addition to the point cloud itself, the scanner positions are shown as blue crosshairs. These scanner handles allow the selection of separate scans for merging scanner positions which belong to the same room (see below). In order to navigate the data, the software supports common ways of navigating the virtual camera. *Dragging* with the *Right Mouse Button rotates* the camera around the center of the scene. *Right Dragging while holding Shift* allows to *zoom* in and out of the scene and *Right Dragging while holding Ctrl* pans the camera up/down or left/right relative to the viewing angle.

On the right hand side of the main window, multiple groups of options and commands are available. Figure 14 shows the first three sections which control how and which elements are displayed in the 3D viewport. The *Navigation* group contains controls for switching between two different camera movement modes. In *Orbit* mode, the camera rotates around a pivot point in front of the camera while in *Fly* mode, the camera can be rotated in-place. The *Rendering* section contains interface elements to set the background color

---

[3] For this deliverable the prototype has been ported from a *GNU/Linux* version to a *Windows* version - the former can also import *.ifc* and *.e57* files, however due to time constraints the latter (and therefore the delivered prototype) does not yet support IFC/E57.



Active: ☑

Loaded 776956 points.

Figure 13: The initial state of the software after loading the point clouds.

of the viewport as well as for switching between perspective and orthographic projection modes. In the *Display options* group, the point size used for rendering the point cloud may be adjusted, and display of detected planes (see below), point normals, and scanner positions may be toggled.

At the bottom of the main window, a status bar informs the user about the current state of the program. For instance, when a computation is running, a message will be displayed here.

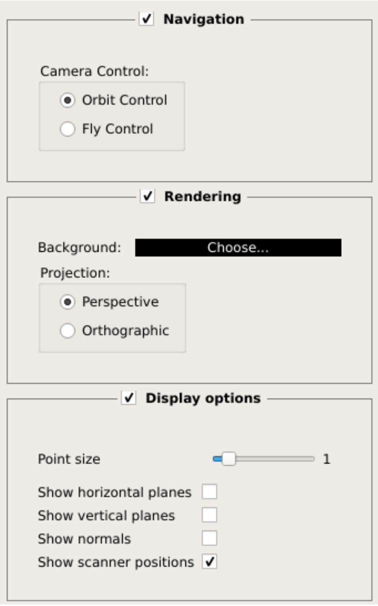
In a first step of the structural analysis process, scanner positions which are located within the same room need to be merged. The scanner positions which shall be merged may be selected in the 3D viewport by *left-clicking* near the scanner position crosshairs; multiple scans may be selected by clicking while holding down the *shift* key. Once the desired scans have been selected, they may be merged using the *Connect selected scanners* button in the *Labels* section. Figure 15 shows an example where two scanner positions have been selected for merging.

The next step is the detection of planes within the point cloud; this step is performed using the *Primitives* section on the right hand side. The minimum number of points as well as the minimum (approximate) area of planes may be set before the plane detection is started. After plane detection, the *Show horizontal/vertical planes* checkboxes in the *Display options* section may be used for visual inspection of the detected planes. Figure 16 shows an example in the *Kronborg* example dataset.

At this point, the automatic relabeling process for updating the point-to-room assign-

Function	**Shortcut**
Rotate Camera	Right Mouse Drag
Zoom Camera	Shift + Right Mouse Drag
Pan Camera	Ctrl + Right Mouse Drag

Table 1: Available shortcuts for camera navigation.



☑ **Navigation**

Camera Control:
● Orbit Control
○ Fly Control

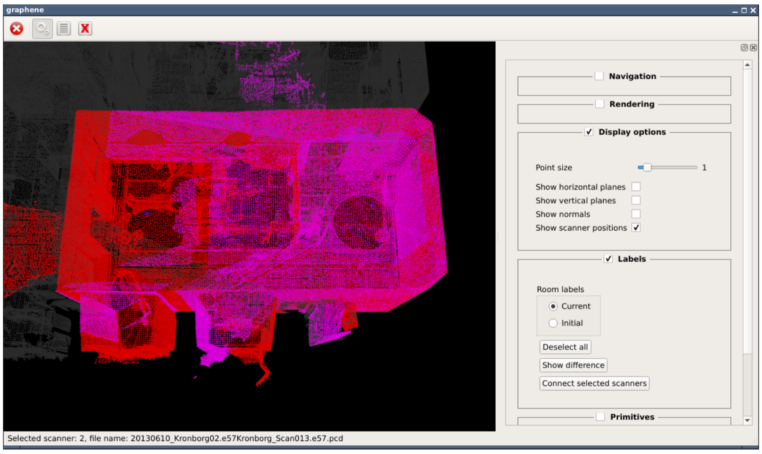
☑ **Rendering**

Background: Choose...
Projection:
● Perspective
○ Orthographic

☑ **Display options**

Point size 1
Show horizontal planes ☐
Show vertical planes ☐
Show normals ☐
Show scanner positions ☑

Figure 14: Display option sections.



☐ **Navigation**

☐ **Rendering**

☑ **Display options**

Point size 1
Show horizontal planes ☐
Show vertical planes ☐
Show normals ☐
Show scanner positions ☑

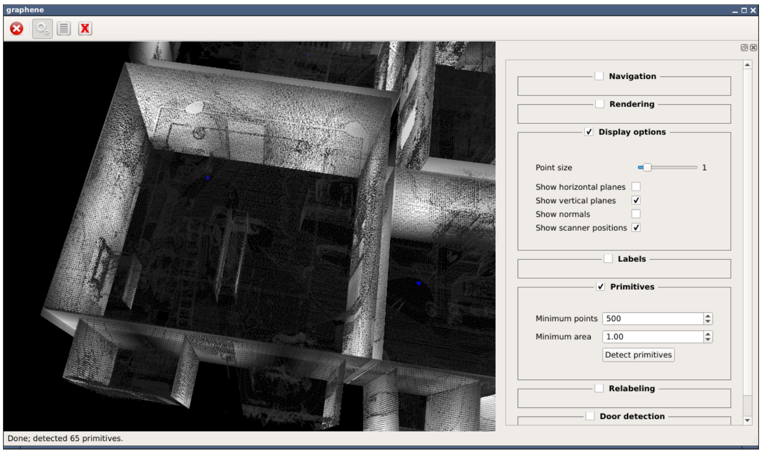
☑ **Labels**

Room labels
● Current
○ Initial

☐ **Primitives**

Selected scanner: 2, file name: 20130610_Kronborg02.e57Kronborg_Scan013.e57.pcd

Figure 15: Two scanner positions to be merged.



graphene

☐ Navigation

☐ Rendering

☑ Display options

Point size 1

Show horizontal planes ☐

Show vertical planes ☑

Show normals ☐

Show scanner positions ☑

☐ Labels

☑ Primitives

Minimum points 500

Minimum area 1.00

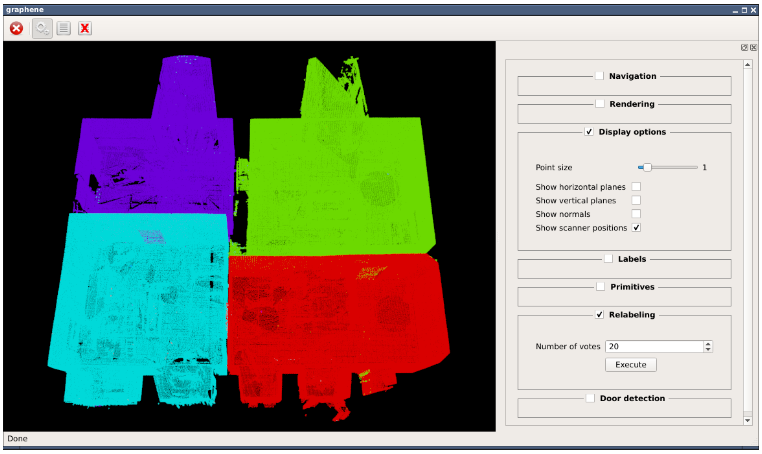
Detect primitives

☐ Relabeling

☐ Door detection

Done; detected 65 primitives.

Figure 16: Detected (vertical) planes in the point cloud.



graphene

☐ Navigation

☐ Rendering

☑ Display options

Point size 1

Show horizontal planes ☐

Show vertical planes ☐

Show normals ☐

Show scanner positions ☑

☐ Labels

☐ Primitives

☑ Relabeling

Number of votes 20

Execute

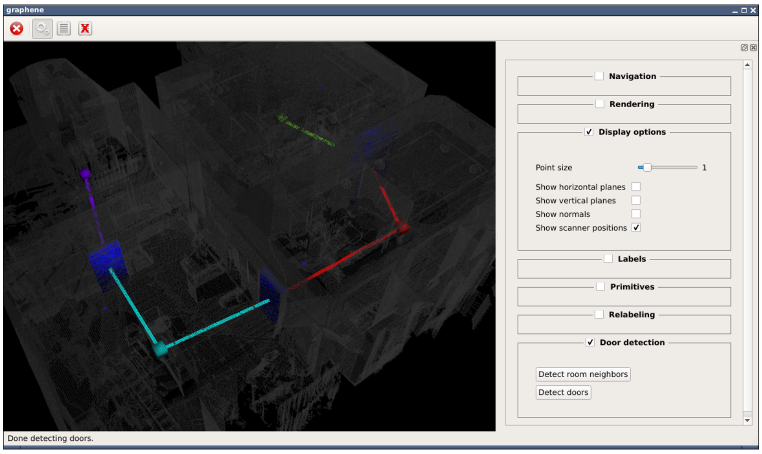
☐ Door detection

Done

Figure 17: Room labeling after the relabeling process.

ments and resolving scan overlaps may be started using the *Execute* button in the *Relabeling* section. The process may be repeated multiple times in order to improve the room segmentation result. Figure 17 shows the room labeling on the *Kronborg* example datasets after a few iterations.

Once the room segmentation is computed, room neighbors and room connections may be determined using the respective buttons in the *Door detection* section. Note that room neighbors need to be determined first before the door detection can be performed. Figure 18 shows the doors extracted in the *Kronborg* example dataset.



Point size 1
Show horizontal planes ☐
Show vertical planes ☐
Show normals ☐
Show scanner positions ☑

Done detecting doors.

Figure 18: Doors detected in one of the example datasets.