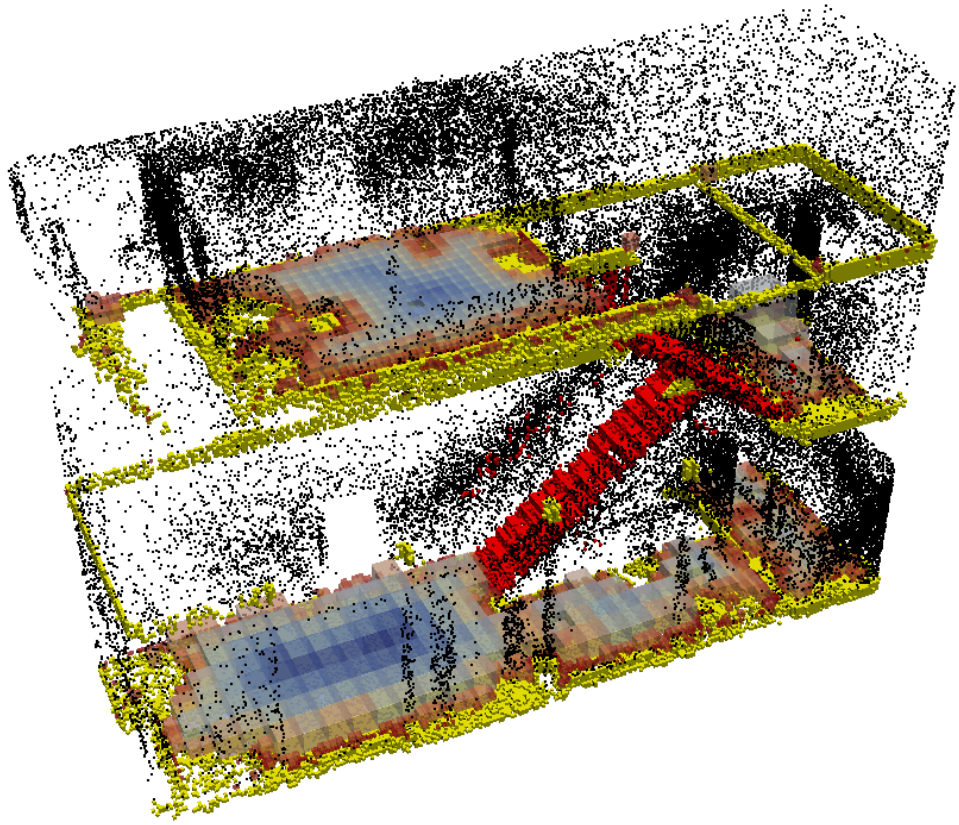
Master of Science Geomatics

Semantic Enrichment of a Point Cloud based on an Octree for Multi-Storey Pathfinding

Florian W. Fichtner

June 2016



TU Delft

CGI

SEMANTIC ENRICHMENT OF A POINT CLOUD BASED ON AN OCTREE FOR MULTI-STOREY PATHFINDING

A thesis submitted to the Delft University of Technology in
partial fulfilment
of the requirements for the degree of

Master of Science in Geomatics

by

Florian W. Fichtner

June 2016

Florian W. Fichtner: *Semantic enrichment of a point cloud based on an octree for multi-storey pathfinding* (2016)

The work in this thesis was made in:

3D Geo-information Group
Department of Urbanism
Faculty of Architecture & the Built Environment
Delft University of Technology

CGI Nederland

Supervisors:	Prof.Dr. Sisi Zlatanova
	Dr. Abdoulaye Diakité
	Robert Voûte (CGI Nederland)
Co-reader:	Edward Verbree

ABSTRACT

Acquiring point clouds of indoor environments became increasingly accessible in recent years. However, the resulting 3D point cloud data is unstructured, and does not contain enough information to be useful for complex tasks like pathfinding. Indoor models which are currently derived from point clouds do not include furniture and stairs. The necessary graph to enable multi-storey pathfinding is not available in the point cloud.

This thesis proposes a workflow to semantically enrich indoor point clouds using an octree data structure. Meaning is added to the point cloud scene that allows to act as a basis for a graph. This graph can then follow navigation constraints of humans through an indoor environment. The approach for semantic enrichment of this study is capable of separating storeys, detecting floors, walls, stairs and obstacles like furniture. Strict preconditions are used, like walls being perpendicular to each other and using noise free point clouds. The implementation works as a proof of concept and the octree proves to be a helpful data structure.

The contribution is a novel approach of using octrees for the semantic enrichment of indoor point clouds, including the detection of stairs. Combining and extending different works from various fields of research helped to develop the presented methodology. The semantic classification of floors and stairs in 3D point clouds allows to create a graph across multiple storeys. A big part of the methodology was implemented and tested on different types of buildings and scanned with dissimilar kinds of laser scanners. Mobile scanners were found to be advantageous, because they are less dependent on the line of sight. On top of that, they can provide a path of the scanner, which is precious information that can support several structuring improvements of the acquired data. The octree generalises the point cloud to leafs and adds a structure to the empty space. This does not only improve the calculation performance, but also provides more distinguishable results.

Concluding, the thesis proposes to extent the framework to cover a wider range of architectural structures. Furthermore, future research should deepen the knowledge for the stair detection of the presented conceptual framework. As the representation of stairs differ for every scanner, future research should focus on point clouds acquired by one kind of scanner first.

ACKNOWLEDGEMENTS

Thanks to everyone directly or indirectly involved throughout this research. Special thanks goes to my supervisors Dr. Abdoulaye Diakité, Robert Voûte and Prof. Dr. Sisi Zlatanova for their support, feedback and ideas. Special thanks to Abdou for his optimism and advice, Robert for our inspirational discussions and Sisi for pushing me in the right direction. My gratitude also goes to all other teachers I had during the programme, I really learned a lot in the last two years.

Thanks also to Edward Verbree who did not only support our research group during the Geomatics Synthesis Project, but also gave valuable feedback after P4. Furthermore I would like to thank the proof readers of my thesis Ivo, Tom and Merwin. My gratitudes also go to the colleagues at CGI for their inspiration and help.

Finally, I would like to thank my family and friends. Without them and their support a Master in Delft would not have been possible.

CONTENTS

ACRONYMS

BIM Building Information Model . . . 7
FFPH fast feature points histograms . . . 35
GIS Geographic Information System . . . 7
GNSS Global Navigation Satellite System . . . 35
GPS Global Positioning System . . . 2
ICP Iterative Closest Point . . . 35
IMU Inertial Measurement Unit . . . 35
LAS LASer File Format . . . 37
LiDAR Light Detection and Ranging . . . 2
MAVs micro aerial vehicles . . . 12
NHAPS National Human Activity Pattern Survey . . . 1
PCL Point Cloud Library . . . 9
RANSAC random sample consensus . . . 9
SLAM Simultaneous Localization and Mapping . . . 11
SVD Singular Value Decomposition . . . 27
UAV unmanned aerial vehicle . . . 12

1 INTRODUCTION

Indoor navigation consists out of two parts: The localisation inside a building and the route planning, which brings the user to a desired location. To successfully achieve those tasks an appropriate representation or model of the building is required [Liu and Zlatanova, 2012]. Many buildings, whether public or not, lack up to date 3D models or floor plans. Even emergency plans are often outdated or the model of the architect does not correspond with the building which was built in the end. Manually creating indoor models or floor plans is time consuming and expensive. In case of emergencies or for temporary exhibitions such models can be valuable for firemen for orientation purposes or forecasting fire and smoke developments. However, in such situations, there is no time to create a semantically rich vector model of the interior of a building. Point clouds, on the other hand, can be acquired rapidly and relatively cheaply, but they lack structure and semantic information which are necessary for pathfinding.

Consider a public building like a hospital or similar large public construction, where suddenly a fire breaks out. When the fire brigade arrives, they realise that their floor plan of the building is not up to date. In the meantime, several walls or even whole parts of the building were rebuilt or demolished. As it is a huge building only some parts are currently on fire. However, the fire will be hard to control and it cannot be guaranteed that other parts will not catch fire, too. While evacuation takes place some firemen also start with the acquisition of indoor data using laser scanners. This data will be helpful in a later stage to not only detect change, but also to help the firemen to navigate through the building and to tell them beforehand if a certain route is possible to take with large equipment. In such emergency situations every bit of information matters. The technology to acquire point clouds is already there, but the data is not very useful for pathfinding yet. The point clouds are unstructured and do not contain any semantic information.

This master's thesis describes a workflow which semantically enriches an indoor point cloud of a building with the use of an octree. The added semantics support the pathfinding between storeys and a path to follow the constraints of human movement can be derived. The results can help emergency responders to navigate through a building.

1.1 BACKGROUND

A study of the National Human Activity Pattern Survey (NHAPS) showed that Americans spend 87% of their life inside buildings. 11% of the time, these buildings are neither residential, nor a workplace or restaurant. This makes it likely for the person to be in relatively unknown or complex buildings, like shopping malls or airports. Only 5.5% of the time is spent in vehicles [Klepeis et al., 2001]. Nevertheless, pathfinding and navigation for outdoors have been studied for a long time already. Despite this, making

use of indoor navigation is still not very common. This is mainly due to the difficulties in positioning indoors as there is no Global Positioning System (GPS) or other signal globally available, which allows to navigate in all buildings. Furthermore, it is a complex challenge to make indoor models suitable for navigation as they need to be in 3D to optimally connect multi-storey buildings. Such pathfinding models should be automatically derived and follow the constraints of human movement.

The meaning of the terms *pathfinding* and *navigation* differs, even though used interchangeable many times. Navigation offers continuous real-time guidance during the movement, while pathfinding finds a route connecting two locations [Karimi, 2015]. Pathfinding is thus necessary for navigation. This thesis aims to support pathfinding, but the results can later also be used for navigation.

This thesis is part of the SIMs3D project, which aims to solve the problem of missing up to date 3D indoor models for many large public buildings. The safety management of public buildings like the fire brigade, are in need of such models. This project ranges from 3D indoor reconstruction from point clouds to 3D indoor models (geometry, semantics and topology), but also aims at 3D indoor navigation suitable for all kind of tasks and users. Finally, a rapid and low-cost 3D modelling approach which allows to identify spaces and networks, needed for a navigation applications, shall be created [SIMs3D (Smart Indoor Models in 3D), 2015].

During the 2015 Geomatics Synthesis Project at the Delft University of Technology I took part in a group that developed a workflow, which efficiently identifies the empty space in a point cloud, and structures both the empty space and the points in an octree. The identification of empty space, in other publications also called free space, instead of using boundaries (walls, obstacles, etc.), allows the focus on pathfinding instead of only collision avoidance. This makes sense because the space where the object or person actually moves in is structured [Broersen et al., 2016]. An octree is a three dimensional extension of a quadtree data structure. It consist out of a 3D volume and is recursively subdivided into eight octants until octants of a uniform colour (black or white so either filled with points or empty) are obtained, or a predetermined level of subdivision is reached [Samet, 1989]. Each octant not further split is called leaf.

Regarding localisation in indoor environments, van der Ham [2015] analysed the 2D performance of the Quuppa technology system [Quuppa Oy, 2015] in the Spark Center of CGI Nederland. He found that it can be used up to a sub-meter accuracy level for asset tracking. Quuppa provides a locating system which uses the Angle-of-Arrival processing of a Bluetooth Low Energy signal for its location data.

1.2 PROBLEM STATEMENT

Acquiring point clouds of the interior spaces in buildings became increasingly easy and cheap in the recent years. Technologies like the ZEB1 Light Detection and Ranging (LiDAR) laser scanner [3D Laser Mapping, 2016] or the Project Tango tablet [Google ATAP, 2015] allow to collect 3D point clouds in an efficient and mobile way. On top of that, there are other methodologies using for example sweep scanning LiDAR by Scanse or Microsoft Kinect.

Nevertheless, this data is unstructured and does not yet contain enough information to be useful for complex tasks like navigation or localisation. Furthermore, the majority of the current models and point clouds need to be free of any clutter like furniture and stairs. The point clouds are either cleaned or only acquired in completely empty buildings. In many models the geometry is preserved, however the semantics as well as the topological information are lost [Diakité et al., 2014]. Indoor environments change regularly. Therefore 3D indoor navigation modelling requires an accurate 3D topographic space of the interior of the building, which should be found in an automated and fast way [Jamali et al., 2015]. The representation of a whole room as one single object is not enough, especially if the room is big or the navigation system is used for autonomous objects [Krūminaitė and Zlatanova, 2014]. A further subdivision is therefore recommended. The octree structure subdivides the space, but only classifies in occupied and non-occupied (empty) space [Broersen et al., 2016]. Any additional semantic information is lost and there is no way to distinguish whether the space is open for navigation (for example above stairs) or not (for example under a table). Also the identification of staircases is essential to reach the ultimate goal of pathfinding on multi-storey indoor environments [Sinha et al., 2014] and to create a logical model seamlessly connecting all storeys.

In case of emergency or for temporary buildings, like for exhibitions or fairs, pathfinding models should be created as fast, cheap and autonomous as possible. A standard way is needed to obtain a pathfinding model from a point cloud connecting multiple storeys. This problem will be addressed in this thesis using point clouds and an octree structure. Semantics are added to establish relationships between different subspaces and allow to connect multiple storeys via the stairways. Therefore, a 3D (or at least 2.5D) model is required [Karimi, 2015]. The technique can also help to support user requirements, such as if a person or object cannot use stairs (for example with a wheelchair) and is not able to use a specific path because of that. Also if the actor is too wide to fit through certain environments 3D information is required.

The lack of structure and semantics in geometric data of indoor environments, whose acquisition becomes increasingly accessible, can be overcome when the point cloud and its empty space are structured in a semantically enriched octree. The structure of an octree enables to derive a graph or network model following the pathfinding constraints of the actor. The usual approach is the reconstruction of complex vector models, but in many cases this is not necessary for pathfinding applications. It usually makes more sense to show the user an intuitive set of cues about when to turn and where, than to make the user look at complex models or even pictures of the same hallway he is moving through [Halsted, 2014]. With the octree approach only the route can be shown. The data however is still complete with a detailed point cloud in the back end.

1.3 RESEARCH QUESTIONS

This MSc thesis has the aim to enhance the octree structure of a point cloud and its empty space with semantic information and to automatically derive a model that allows to find a path between multiple connected storeys. The research aims to answer the following question: *"To what extent can an octree support semantic enrichment of point clouds for the purpose of multi-storey*

pathfinding?" To answer this, the following sub research questions have to be answered as well:

- Can floors and walls be distinguished in the octree structure of a 3D indoor point cloud containing stairs and furniture?
- What methodology can be used to detect stairs in the octree structure derived from a 3D indoor point cloud?
- What is the influence of the three different scanners on the result of the semantic enrichment?
- Is the semantically enriched octree sufficient to derive a pathfinding model for humans?
- Which semantics improvements are necessary to establish a link between indoor and outdoor networks?

1.4 OBJECTIVES AND SCOPE

The aim of this thesis is to create a workflow that takes an indoor point cloud from a laser scanner as input. It then identifies and structures empty space and the points in an octree. Before it enhances the structure with automatically derived semantic information to subdivide the space and to create a model for multi-storey pathfinding. Thus, while Broersen et al. [2016] only found and structured the empty space in a point cloud, this work explores how to add constraints and semantics to the resulting model. The pathfinding model should relate to the movement of humans, so be restricted to floors and stairs only, as they are walkable and can be used for a path. Therefore, with the semantic enrichment the following features are found and classified:

- floor and storeys
- stairs
- walls
- obstacles (for example furniture)

After the classification all obstacles, walls and nearby empty space can be excluded from the membership of the pathfinding model. The resulting representation or graph should be layered to keep the connectivity (at the stairs) between different storeys in the building. The layers should be connected at the stairs.

Therefore, the location of floors and storeys as well as stairs will be identified. Furthermore walls and other obstacles will be classified in this workflow. The list could be extended with elevator, doors and windows for example. They lie, however, out of the scope of this research. Also a further definition of the obstacles is out of the scope. All not classified points will therefore be obstacles.

To be able to use doors for pathfinding it is assumed that they are open during the time of the laser scan to acquire the point cloud. Also cleaning of the data is out of the scope of this research as several tools, like CloudCompare [Girardeau-Montaut, 2016], are available to perform such tasks. Thus, it is assumed that the point clouds are clean and free of noise, from the beginning.

1.5 RESEARCH RELEVANCE

Point clouds of buildings can be obtained quickly nowadays. But there is no way to automatically derive a structured and semantically enriched model suitable for pathfinding, while at the same time enable and enhance the multi-storey connectivity via the stairs. This is essential however, as there is not always enough time to create a semantically rich building model that facilitates pathfinding. Also Liu and Zlatanova [2013a] write that there is only very few research about the creation of navigation models derived from 3D geometry. Current 3D indoor models are simple, not scalable and the data has no clutter, like furniture or stairs [Nikoohemat, 2016]. This thesis aims to provide a first step for combining and extending several methodologies, to not only semantically enrich and reconstruct a 3D indoor model, but also allow pathfinding across multiple storeys. On top of that, the research in not limited to pathfinding only and can be useful for reconstruction and the creation of vector models or floor plans which are automatically built from point clouds. Furthermore, the resulting graph can be a base model for tracking applications and research. Also in these cases semantic enrichment of the point cloud is necessary.

The research can have a value for the society as well. Especially in emergency cases automatically created pathfinding models of buildings can be useful, but also for temporal structure like company fairs or exhibitions. In many cases it can be too expensive or too time consuming to manually create a 3D indoor vector model. Having a model derived from a quickly obtainable point cloud is therefore advantageous.

1.6 READING GUIDE

The following chapter presents the related work (Chapter 2). Chapter 3 will present the conceptual framework and workflow developed and used to fulfil the objectives of this research. Subsequently the implementation will be explained in Chapter 4 before results and analysis follow in Chapter 5. Finally, conclusions will be drawn and the research questions answered in Chapter 6. Furthermore, recommendations for future projects are given.

2 RELATED WORK

Several areas of research are relevant for this thesis. The following chapter will provide an overview of related works. The first part (Section 2.1) will address the semantics and subdivision of space necessary for navigation or pathfinding, which was studied in other works. After that, the research to find an applicable and automatically derived navigation model for indoor applications will be explained (Section 2.2). Finally, research in indoor navigation in the field of robotics will be introduced as an example of collision avoidance, indoor navigation and as a possible future use case (Section 2.3).

2.1 SEMANTICS IN INDOOR POINT CLOUDS

The success of indoor navigation is mainly dependent on the model of the building available [Krūminaitė and Zlatanova, 2014]. Diakité et al. [2014] developed a framework to recover semantic information from Building Information Model (BIM) and 3D Geographic Information System (GIS) data. The framework was based on the combination of geometry and topology for automated semantic labelling. The initial situation of this MSc thesis however, is not based on vector models, but on point clouds and the empty space which are structured and subdivided into an octree. Therefore, to enhance subdivision and to create a logical model for indoor pathfinding across multiple-storeys, features have to be detected from those point clouds. 3D modelling of the building architecture from point cloud scans is a rapidly advancing field [Turner and Zakhor, 2013] and its results can be beneficial for indoor wayfinding. Broersen et al. [2016] efficiently generated a linear octree from a point cloud and derived the empty space, which is usable for pathfinding. However, there were no further semantics provided, besides empty and occupied space. The following sections will go deeper into detail about what has been researched in other works. This research will form the basis of this project.

2.1.1 Feature detection and 3D indoor reconstruction

For extracting information, or to build an application using point clouds, the recognition of object surfaces is often one of the first steps to perform [Vosselman et al., 2004]. Vo et al. [2015] were able to extract roads from point clouds using a supervised machine learning approach. Wang and Tseng (2004, 2005, 2011 and 2014) published extensive research about extraction of surface features from LiDAR data using an octree data structure. However, there is no focus on indoor environments. Yogeswaran and Payeur [2009] developed a technique to group features from a point cloud based on their proximity and similarity using an octree structure, but there was no further classification. Rusu et al. [2009a] present a method to identify objects in a point cloud of a kitchen environment. They find planar areas

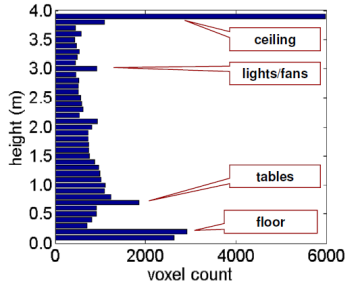


Figure 2.1: Height histogram, as projection of 3D points to the vertical axis. The larger maxima at top and bottom can be used to identify floor and ceiling heights. *image courtesy of* Okorn et al. [2010]

using the point's normals and then split them into smaller parts with a region growing method. Trained conditional random fields, purely based on geometrical reasoning, are used for classification. The result is a semantic object map, which can be of help to provide context for robots. Furthermore, the Triangulated Surface Map, which is also created, can support collision avoidance and path planning.

Khoshelham and Díaz-Vilariño [2014] present a method for automated 3D indoor modelling of floors, ceilings and walls with shape grammar and histograms in point clouds. This is suitable for the detection of floors and walls as well. Also Becker et al. [2015] use shape grammar to reconstruct 3D indoor models as BIM from point clouds.

Okorn et al. [2010] take a more planar based 3D indoor reconstruction method and use histograms to detect floor and ceiling data from 3D point clouds like shown in Figure 2.1. Histogram peaks represent many points, sharing the same height. The histogram approach is similar to the one in Khoshelham and Díaz-Vilariño [2014]. Walls are identified using histograms and a hough transformation approach for line detection. Oesau et al. [2014] further develop this methodology using a graph-cut formulation to solve the inside/outside labelling of a space partitioning. The approach of Ochmann et al. [2016] makes use of the piecewise linearity of wall segments to detect vertical planes as candidates for wall surfaces and project them to the horizontal plane. Furthermore, the structure gets enriched through door and window detection.

Turner et al. [2015] present a workflow to create 2D floor plans from point clouds making use of the scanner's position and voxel carving, as well as partitioning by height with histograms similar to the methodologies explained above. Such approaches can also be useful in this work to add semantics about room separation to the model or the histograms to distinguish different levels in the building. Turner [2015] go further into detail about their approach to derive 2D samplings of wall positions from histogram analysis on point clouds. It is also shown how to generate floor plans from octree structures and how to make use of the topology information to distinguish between walls and obstacles. The approach uses the assumption that walls are vertical and identifies large planar surfaces. The results of the wall samples are cleaner and more robust to clutter than the ones directly extracted from the point cloud.

Xiong et al. [2013] create a BIM from laser scanner data, where they reconstruct occlusions as well as are able to label walls, ceilings and doors. Volumetric primitives are reconstructed by Xiao and Furukawa [2014] from ground-level photographs and 3D laser points.

2.1.2 Identification of stairs

One of the main goals of feature detection and semantic enrichment in this research lays in the identification of stairs in a point cloud or octree. To enable indoor pathfinding, first an indoor model has to be created. The majority of such a model can be shaped with walls, ceilings, floors and stairways which the authors call the four structural elements [Sanchez and Zakhor, 2012]. They run a principal component analysis for every point and then they classify and segment the point cloud into these structural elements. Their system makes use of public C++ libraries of the Point Cloud Library (PCL).

Also Eilering et al. [2014] identify those surfaces, with a specific focus on segmenting stair structures. The classification of each point is dependent on local spatial features of the point cloud, but also on the classification of close points based on probability. With a trained dataset they achieve a success rate of approximately 75%. On the other hand, there was not yet a focus on segmentation and structural relationships between the objects, but they plan to do this in the future. Python and the PCL were used for the implementation.

Schnabel et al. [2007] use random sample consensus (RANSAC) techniques to identify structures in point clouds. Also Oßwald et al. [2011] argue that RANSAC tends to simplify complex planar structures so that small steps merge into a sloped plane. This however, might not be detrimental for this project.

Delmerico et al. [2013] model and localise stairways on a map from depth imagery where lines represent discontinuities and depth changes abruptly. Stairs are detected by a big and regular change of depth, however the viewpoint does matter. Even though this is a promising and well described method, Delmerico et al. [2013] use depth images for the detection of stairs instead of using the point clouds directly, like it is aimed at throughout this research. Another research shows an algorithm for grouping step-like obstacles from point clouds obtained by RGB-D sensor [Sinha et al., 2014]. It runs in real-time, divides stairs in subsections and the authors claim that it is suitable for any real-world application. The paper contains a well described method and pseudo-code, but only the point cloud and environment within the viewpoint around the robot is considered.

Bansal et al. [2010, 2011] stream a point cloud, store the points in a voxel grid and use image processing methods to detect doors and stairs. The usage of histograms for stairs detection, even though just a small part of the research, follows a similar approach as the histogram based method of Okorn et al. [2010] or Oesau et al. [2014] and can therefore potentially be connected. The authors use filters, see Figure 2.2, on top of 2D histograms built according to the plane direction through the points in each voxel, to identify the location of the stairs.

1. Bansal et al. [2010, 2011] first voxelise the point cloud, fit local planes and identify vertical and horizontal surfaces.

2. Then the voxels containing surfaces are projected to a 2D grid giving a vertical and a horizontal histogram representation respectively where the number of cells measure how many voxels above this cell contain vertical or horizontal surfaces.

3. Now the filters need to be set to detect the stairs, so that the matched filter for the vertical histogram is a rectangular block with alternating

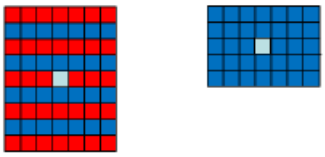


Figure 2.2: Matched filters: stair vertical filter (left) and horizontal filter (right), blue represents positive filter weights and red negative ones, *image courtesy of* Bansal et al. [2011] [*changed*]

rows of positive and negative weights. The horizontal histogram only shows a continuous horizontal structure. Moreover the number of steps should be at least 3 to gain robustness for the stair detection.

4. Finally, the filter responses for the vertical and horizontal histograms of the previous step need to be combined to increase robustness.

The approach works on a point cloud from a global perspective and does not rely on a viewing angle of the scanner. Also the initial voxel data structure has overlaps with the one which will be used for the research in this thesis, which was developed by Broersen et al. [2016].

Stairways can also be found using their slope, so normal directions found through a covariance matrix to segment a Kinect point cloud into regions [Wagner et al., 2015]. Thus again, the tilted position of the scanner has to be taken into account in order to use this methodology.

A majority of the studies are performed with the aim of improving robot navigation, more specific research about this can be found in Section 2.3.

2.2 INDOOR PATHFINDING MODELS

There already exists a wide range of different kind of digital models, which can be used for indoor pathfinding purposes. They range from geometric, to semantic and topological models or to a combination of them [Krūminaitė and Zlatanova, 2014]. Liu and Zlatanova [2013b] argue that there are different types of indoor paths that depend on attributes like the user size or a visibility graph. Zlatanova et al. [2014] present several 3D (and also 2D) approaches for indoor navigation models. The possibilities for creating a complete and partial subdivision for networks, or to use a 3D grid graph are presented. Zlatanova et al. [2014] identify four different steps necessary for a successful application for navigation through indoor spaces. The digital acquisition of available spaces (1), the structuring of acquired data (2), formalisation of the data to establish relationships between different subspaces (3) and lastly applying the user requirements on the formalised and structured data (4). Subspaces are formed by subdivision of indoor space into smaller parts. They can have semantic meaning. Moreover, they can be navigable, so be used to perform activities, or non navigable like for example walls. There are many applications in disaster management, facility management for indoor 3D models and they are necessary for indoor pathfinding. 2D plans can be a basis to derive 3D models, but they are often outdated, not available and tedious work is necessary to achieve good results [Nikoohemat, 2016].

Different logical models can be used for pathfinding, and the subdivision is dependent on the type of application [Liu and Zlatanova, 2013a]. There

are two different ways to subdivide, first semantically, which means to divide into meaningful subspaces (with information about the kind of room, etc.) and secondly geometrical, which divides based on geometry only (grid, Voronoi diagram etc.). Both methods have advantages and disadvantages, as the geometric approach can be fully automated, but the semantics remain unclear. For the semantic approach the reverse is the case. Liu and Zlatanova [2013a] point out, that there was only a limited amount of research to derive topological models from 3D geometrical models at the time of writing. With the aim to automatically derive a logical model, first navigable spaces and doors need to be identified in order to build a connectivity network. They also argue that to generate a multiple-storey navigation model from floor plans, semantics have to be added. However, there is no standard way of creating a network from floor plans.

Jamali et al. [2015] use a laser range finder for the acquisition of inaccurate geometry from buildings and propose a 3D modelling method for a topological navigation network. The model and connectivity is built up step by step, fully autonomous, but each space requires an ID, specified during the scanning process. Topology is mapped using dual edges for adjacent spaces.

Yang and Worboys [2015] use combinatorial maps and their duals to derive a navigation graph, while Yuan and Schneider [2010] introduce the so called LEGO model and then use merged blocks of maximum available widths and heights. From the connectors between blocks, which they define through neighbour finding, they build a graph where the nodes are the connectors and the edges the distance between them. Yuan and Schneider [2011] use their LEGO model and build a graph for navigation by representing blocks as nodes and connectors as edges. Because they first merge larger blocks, they create a structure quite similar to an octree and might therefore be useful for this research as well. Generally deriving topology using Poincaré duality is a common approach [Lee and Zlatanova, 2008; Becker et al., 2009; Boguslawski et al., 2011].

Narasimhan et al. [2006] extract a graph from a CAD model and store it in an octree. The octree enables them to determine the path to the closest destination.

The OGC IndoorGML standard for an open data model and XML schema for indoor spatial information is based on the requirements for pathfinding [Lee et al., 2014].

2.3 INDOOR NAVIGATION IN ROBOTICS

Even though this research's main goal is not aimed at robotics, experiences in this field can be useful for this study. In the area of robotics indoor navigation has been well researched for quite a long time already. Many approaches also use an octree data structure and Simultaneous Localization and Mapping (SLAM) technologies. Herman [1986] argues that an octree representation of obstacles enables fast wayfinding algorithms. Kitamura et al. [1995] represent everything in the environment by an octree structure and generate a potential field from the black leafs in the octree by successively adjoining regions with white leafs. This way they were able to create a collision free navigation method, which is not graph based. An approach that also Wu and Hori [2006] follow, who developed a collision free wayfinding method based on an octree model, which was faster than graph based so-

lutions. However, the research was limited on the movement of a robotic arm. Jessup et al. [2014] show that octree based occupancy grids are a suitable representation for multi-robot 3D mapping and that these grids are updatable.

Robotics need context and semantics, like the room structure, estimated from the environment [Olufs and Vincze, 2011]. Shen et al. [2011] describe a method for autonomous multi-floor indoor navigation of micro aerial vehicles (MAVs), while only using on-board sensors without prior knowledge of the environment. Therefore 2.5D environment models are used, which are formed by collections of vertical walls and horizontal floors or ceilings. Wang et al. [2014] were able to find the 3D coordinate of an unmanned aerial vehicle (UAV) using a 2-laser-setup and structured indoor features with preknown key corner coordinates.

However, the field of robotics does not seem to provide research about the shortest pathfinding in an octree representation of 3D spaces. Instead, sensors are added on moving objects to avoid collision when in motion. On top of that, sensors are also used for SLAM. SLAM stands for measuring an environment, building a map and finding itself in there with the help of feature point matching like in Bergeon et al. [2015]. They use point of interests in images obtained with a Kinect to localise the robot. The octree is often used as a supporting data structure.

Hornung et al. [2013] present the so called OctoMap, a volumetric representation of occupied, empty, and unknown space. It is built from a point cloud and allows sensor updates and raycasting. Hornung et al. [2012] argue that a majority of indoor navigation models use a projected 2D representation of the robot's footprint in a 2D projection of the world. Alternatively, they introduce a stack of 2D representations for the different layers, associated with the attributes of the robot's body at each height. Each of these layers serves as a collision map for each body part with obstacles at the same height. Also Payeur [2006] uses an octree to subdivide 3D space for robotics. The approach is comparable to the one of Broersen et al. [2016].

Vandapel et al. [2005] structure empty space in spheres where the radius of each bubble is determined by the distance to the next point in the cloud. Overlapping and connected bubbles form a network of tunnels, stored as graph. This provides an interesting alternative to the octree approach of Broersen et al. [2016]. A lattice graph, which guarantees that every connection represents a feasible path, like implemented in Brock et al. [2009] is out of the scope of this research.

2.4 OBSERVATIONS

There is a broad research about feature detection and 3D indoor reconstruction from point clouds, which can be reused in this thesis (2.1.1). Especially for stairs detection (2.1.2), but also for other purposes like indoor reconstruction and navigation the field of robotics (2.3) offers a broad research. Nevertheless, the focus of indoor navigation in robotics is mainly based on SLAM, while in this research the environment or at least the geometry is gathered beforehand using laser scanning methods. Also the viewpoint and depth plays a role in many methodologies found. This makes sense for robotics applications, which operate in real time, but it may not be suitable for this work, as the model has to be global and not just consider the environment around the current position.

The goal is to find and implement a workflow to semantically enrich indoor point clouds using an octree data structure and to create a logical model for pathfinding based on geometry, similar to Liu and Zlatanova [2013a] or also Jamali et al. [2015]. Those works did not use a point cloud as input data, though. Work focusing on point clouds on the other hand, are usually limited to modelling and reconstruction or to specific robotic applications or tasks. Also, there was no work found, which derives a pathfinding model in a way it is aimed throughout this research. The methods which will be reused and implemented, based on the findings in this Chapter, will be further explained in Chapter 3. The proposed workflow does first structure the point cloud according to Broersen et al. [2016] and then follows the histogram based extraction of storeys and floors, which was seen in Okorn et al. [2010], Oesau et al. [2014] or Khoshelham and Díaz-Vilariño [2014]. The last methodology is on top of that used to identify walls and vertical features in the point cloud scene. The approach of Bansal et al. [2010, 2011] will be employed to also find the location of stairs in the model and the methodology of Hornung et al. [2012] provides inspiration about how to derive a smart graph to perform pathfinding for humans.

3 CONCEPTUAL FRAMEWORK

The following chapter provides the conceptual framework and approaches used, developed and reused throughout this thesis project. It will emphasise the design of the workflow and describes each step in detail. First, the precondition of the data (Section 3.1) will be defined. Then, the methodology used for the semantic enrichment will be shown (Section 3.2). Following this, the actual approach will be explained in detail, starting with the separation of the storeys, which will be illustrated in Section 3.2.1. Section 3.2.2 describes how the walls are derived and located in the structure before Section 3.2.3 characterises how the stairs were found. Finally, Section 3.3 will construe the way on how to derive a graph network using the semantic enrichment achieved in the earlier sections.

The workflow of the concept can be found in Figure 3.1. The initial data is a 3D indoor point cloud of the building or the part of the building to be used. All points and the remaining empty space are structured and subdivided in an octree. Now, the black leafs in the octree, so the leafs containing points, are further subdivided and labelled according to their storey number and attributes (e.g. whether they are part of a floor, wall or stairs). Following this, a pathfinding model, thus a graph, needs to be derived from the empty space according to the semantically enriched black leafs of the structure. Finally, the indoor graph network should include the possibilities to be connected to outdoor networks.

Figure 3.2 shows the UML of the project logic diagram of this work and how the derivation of the graph relates to the semantic enrichment of the point cloud or octree data structure. First the indoor point cloud and the empty space are structured in an octree. Then the octree or the point cloud get subdivided so that floor, stairs, walls and other obstacles can be distinguished from one another. After that the pathfinding model or graph, which connects different storeys in the building, can be derived.

3.1 PRECONDITION OF THE INPUT DATA

To be able to create a stable framework and to reach the goals of this thesis (see Section 1.4), some preconditions of the input data are necessary to be defined. Below, the general criteria considered in this project are listed:

- The input data is a 3D point cloud of the indoors of a multi-storey building.
- The point cloud should be clean (free of noise).
- The rooms of the acquired point cloud can be furnished, so the point cloud can be cluttered.
- Colour information is not necessary.
- The path of the scanner is not necessary, but can improve the result.

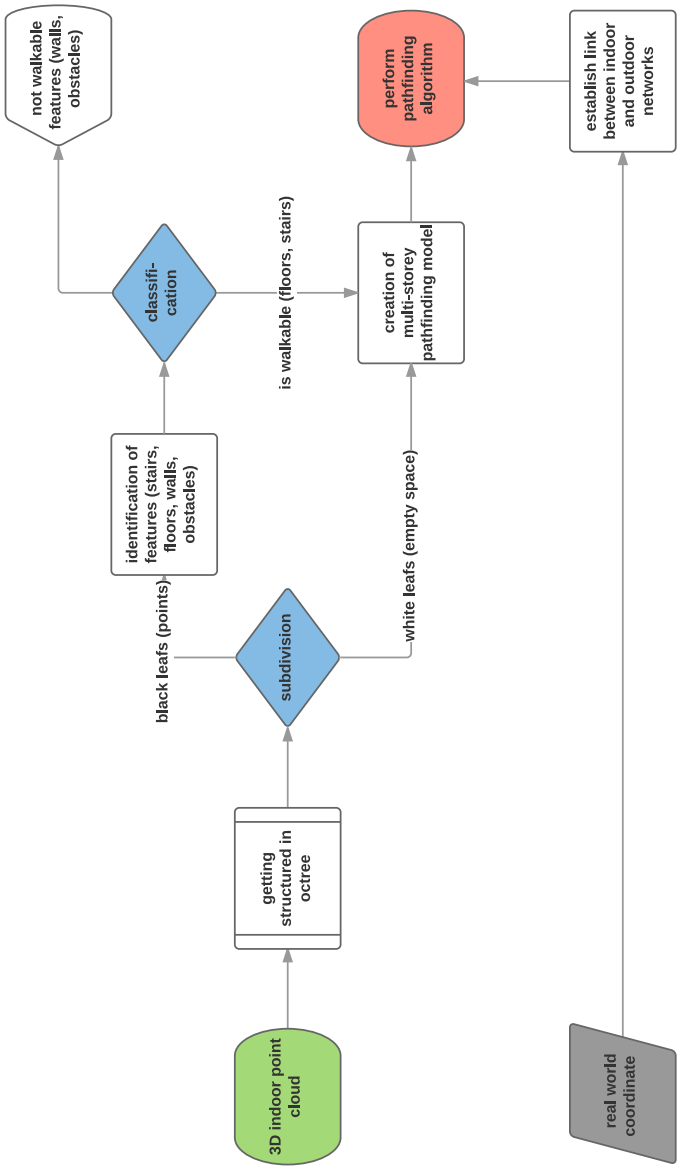


Figure 3.1: Workflow of the methodology for the semantic enrichment of a point cloud based on an octree for multi-storey pathfinding

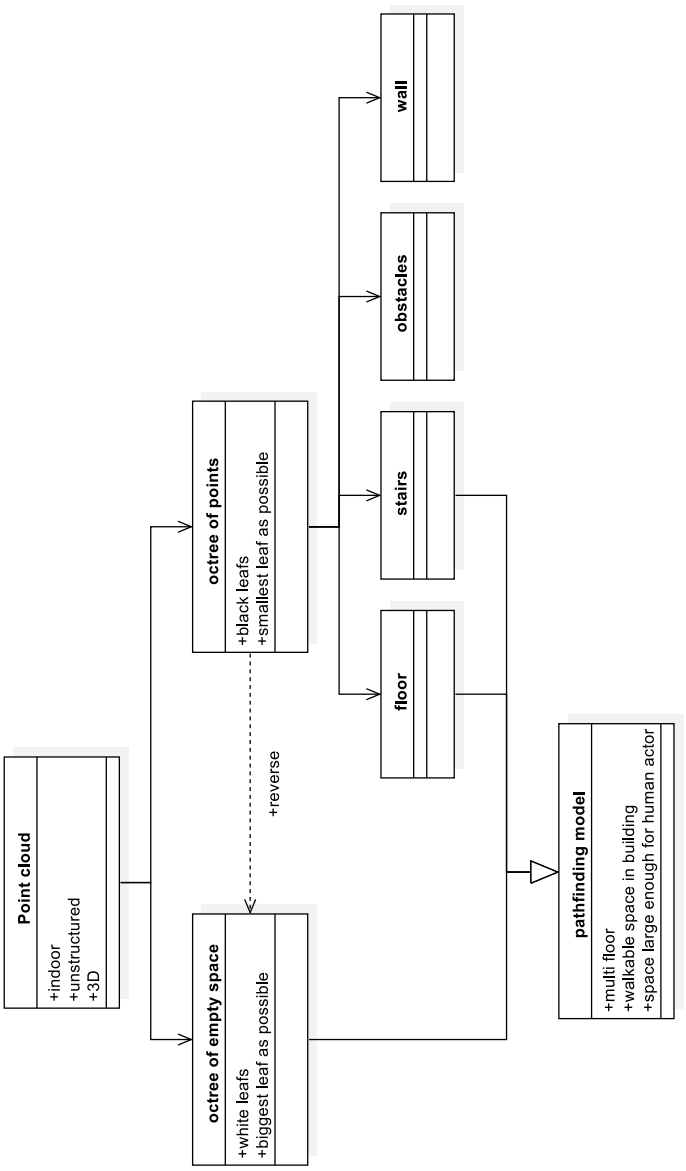


Figure 3.2: UML of the project logic diagram

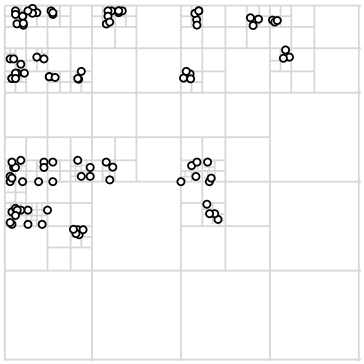


Figure 3.3: Current octree implementation simplified to quadtree structure

- The walls in the building should follow the Manhattan-World assumption.
- Floors should be horizontal and levelled.
- Stairs are perpendicular to walls and completely represented in the point cloud. For the best result, the riser is concrete.

The input point cloud should represent a building with multiple storeys (at least two) which are connected through a staircase. The building should follow the Manhattan-World assumption, this means that all walls are parallel or perpendicular to either the X- or the Y-axis and the floors are horizontal and levelled. Point clouds should be as free of noise as possible and can be cluttered, depending on the scanner used and the environment where they were acquired. Colour information is not needed for the workflow to be able to work, the same applies to the path or location of the scanner. However, having such information can increase the possibilities for a successful analysis and semantic enrichment of the data. The last point emphasises that all elements of the stairs' surface should be represented in the point cloud and not suppressed by either noise or the point of view of the scanner. Furthermore, also the staircases should follow the Manhattan-World assumption.

3.2 SEMANTIC ENRICHMENT

The conceptual framework used in this thesis is built on top of a point cloud structure, which uses an approach developed during the Geomatics Synthesis Project 2015. This approach structured the points and the empty space in an octree [Broersen et al., 2016]. Octrees for point clouds are based on the recursive and non-uniform subdivision of space and are a common approach to structure and segment 3D point clouds and like shown in Chapter 2 used for the partitioning of space. Octrees represent a hierarchical data structure, they allow addressing and efficient use. In the methodology of Broersen et al. [2016] the empty space, so all the free space which is not occupied by any points and therefore available for navigation, is structured in an octree as well. A similar approach was used for localisation by Payeur [2006].

In the current implementation of the octree structure [Broersen et al., 2016], all black leafs (leafs containing one or more points) are split until the deepest and smallest level in the tree. The octree of empty space (the reverse

with all white leafs) are as big and up in the tree as possible. Furthermore, both linear octrees supplement each other and the octree of empty space is the reverse of the full and occupied space. Figure 3.3 show the current octree implementation simplified to a 2D quadtree for visualisation purposes. It is distinguishable that the octree (or quadtree in this figure) is further split when there are points and the leafs are as big as possible when they are empty. This has the advantage that the structure has more detail close to obstacles and less when there is nothing around. Moreover, a method [Vörös, 2000] has been implemented which finds all the leaf's neighbors in the octree on-the-fly for a basic pathfinding calculation. The path, however, has neither size constraints nor is usable for humans and is not bound to the semantics of its environment. This means for example, that the route is not constrained to represent the possible path of a human, which is sticked to the floor or avoids obstacles like tables.

So, after reusing the methodology of the Synthesis Project the empty space and the point cloud are subdivided into an octree structure. But, to find which parts of the model can be used for pathfinding, the floors and the stairs (together with the empty space they make the *walkable space*) need to be identified. In a building four different kinds of semantic classification are important for deriving a pathfinding model for humans:

- storeys
- floors as *walkable space*
- connections between storeys, in this case restricted to stairs as *walkable space*
- walls (static)
- obstacles (for example furniture, often not static)

The semantic attributes need to be derived from the current octree structure. Naturally the focus here lies on the full space rather then the empty space, as all the listed features are represented by full leafs. The pathfinding through the octree of empty space will be able to profit from the richer semantics of its surrounding and can use the better scene understanding to provide a more realistic path. The linear octree of black leaf nodes [Broersen et al., 2016] is the initial data structure and (sometimes along with the points itself) used to semantically enrich the formation. Figure 3.4 shows the point cloud and the empty space in it, structured in an octree. This also illustrates the connection between the empty space and the importance of semantic enrichment of the scene. On the left side of the figure you can see the bar (represented as point cloud instead of octree of full space for visualisation purposes), which is surrounded by empty leafs. The empty leafs are generally free for pathfinding. If they are on top of the bar, however, they should not be part of the graph used for pathfinding as humans usually go around the bar rather, than across it. Therefore, it is necessary to know which leafs can actually be used, so the general understanding of the scene is of importance.

Figure 3.5 shows the workflow of semantic enrichment of the black octree leafs. First the floors are identified and the storeys separated. The gained separation can, with the points be used to identify and verify the walls as well as to locate the stairs. This results in a semantically enriched octree with black leafs which can be put in combination with the white leaf nodes again to create a pathfinding network following human movement constraints. The derivation of the graph is further described in Section 3.3.

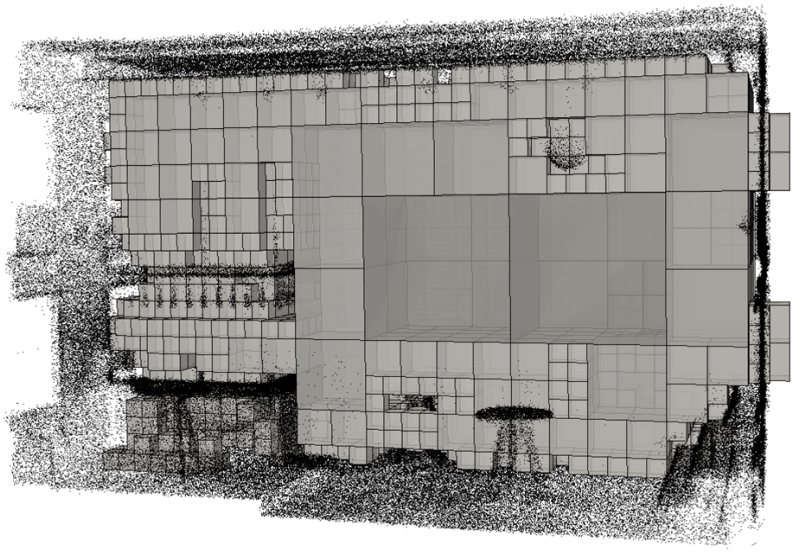


Figure 3.4: Slice through the point cloud of the Bouwpub of the Faculty of Architecture in Delft with the empty space (grey octants) structured as an octree. The bar is on the left of the figure, *image courtesy of* Broersen et al. [2016]

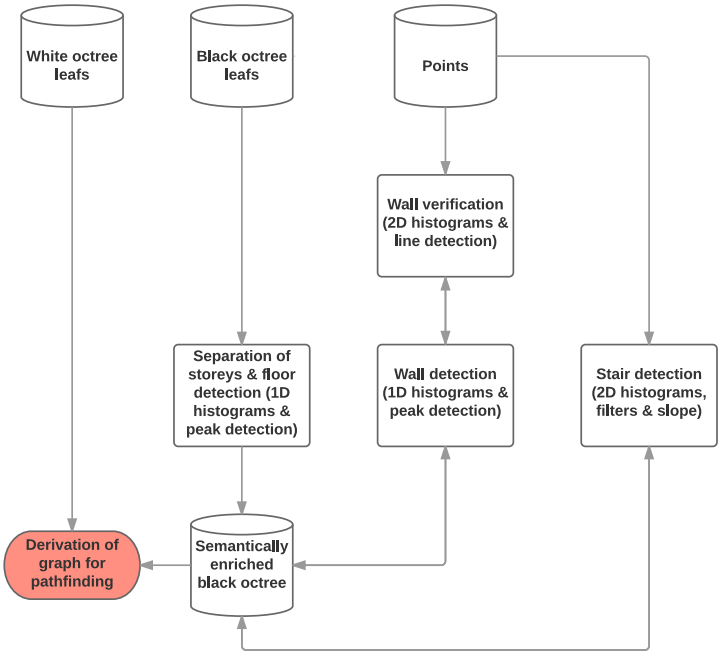


Figure 3.5: Workflow of the semantic enrichment of the octree and the connection to the empty space

3.2.1 Identification of floors and separating storeys

When all preconditions are fulfilled and the data preparation is completed the different storeys need to be distinguished and it has to be possible to separate them from each other. Every storey is defined as a horizontal part of the building, which can be used for walking enclosed by a floor and a ceiling. The typical and human perspective of storey enumeration only counts full storeys and not landings in between stair flights. In this case however, also a platform between different pitch lines or flights in a staircase will be called a separate storey, even though this might not add to the usual storey count used in architecture. The reason behind this is that the identification of storeys in the proposed methodology is equivalent to the identification of the floor. Furthermore, it is necessary for the following steps to look at each floor location separately. The approach will be explained in detail in the following.

The methodology to separate different storeys from each other and to identify the floor is based on Okorn et al. [2010]. A height histogram is created where the samples are projected to a histogram in Z-coordinate direction. Whereas Okorn et al. [2010] use voxels instead of points (like Khoshelham and Díaz-Vilariño [2014]), in this approach the black octree leafs can be used. Every time a black leaf has the same Z-value as another one it is stacked to the according histogram bin. The peaks at the highest increment of the histogram show the location of horizontal structures in the facility, and represent the locations of floors and ceilings. Using the octree leafs instead of the raw point cloud is advantageous because the result proved to be cleaner (see Figure 4.2). Furthermore, the bin size can follow the resolution of the octree and does not have to be manually specified. Also Okorn et al. [2010] state that using the raw point cloud would bias the result in direction density of the points rather than the actual floors.

The peaks in a histogram have the characteristic to be higher than their immediate neighbours [Duarte, 2015]. There are several methodologies to detect local maxima in histograms. The red line in Figure 4.2 shows the minimum peak height, which is necessary to suppress not significant peaks in the data, also Oesau et al. [2014] remove every local maximum which is not above a certain ratio to their neighbourhood. It is important to pay attention to the fact, that close peaks refer to ceiling and floor. Together they represent one concrete feature in the building if they are less far apart then the maximum wall thickness in the building.

Once the peaks are found, the storeys can be separated and the cloud be divided into subclouds according to their storey level. Nevertheless, there are still exceptions to be considered. One of them, which especially occurs in public facilities are dropped ceilings which result in two peaks close to each other, but further apart than the maximum wall thickness. A dropped ceiling is a secondary ceiling, which hangs below the main ceiling and can often been found in public buildings. Figure 3.6 shows such a case between the second and the third peak from the bottom. In many cases, such problems can be solved with a logic constraint by setting the minimum height of a storey to a minimum threshold to rule out such dropped ceilings. This can, however, lead to misleading results when there are for example many tables on a floor or a platform in a staircase. The tables cause similar results as dropped ceilings, because they are consisting out of a large planar area and can appear as another peak in the histogram.

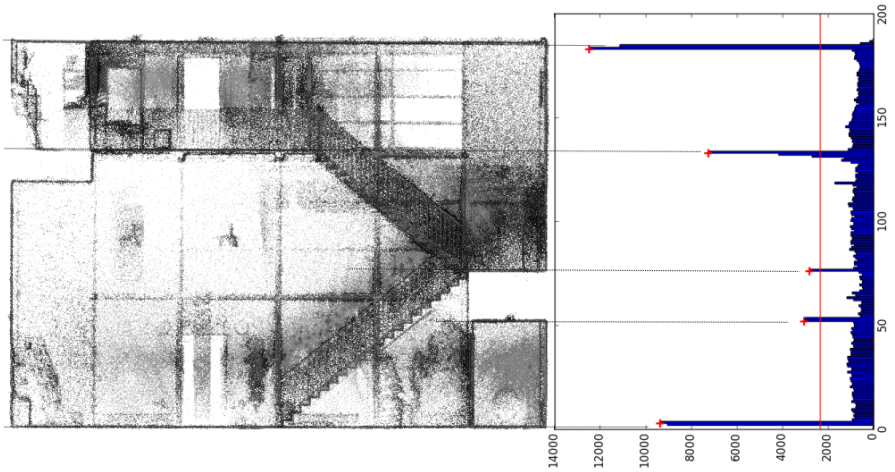


Figure 3.6: The point cloud (left) can be separated into its stories by using the peaks (marked with a red dot) above the minimum peak height (red line) of the histogram of the points in Z-direction (right)

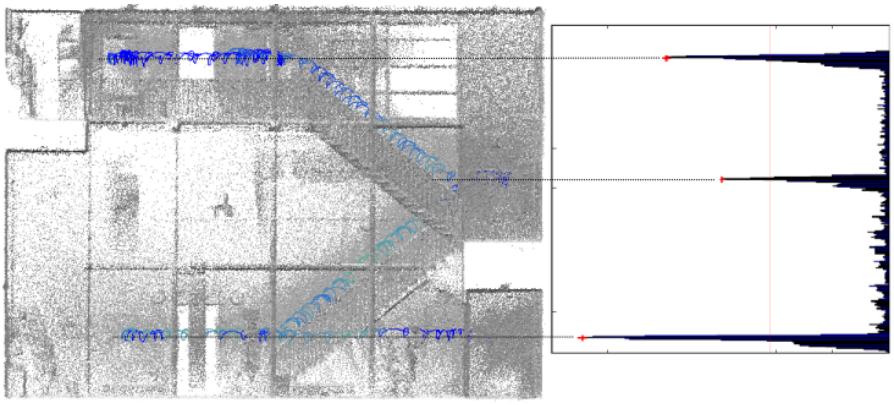


Figure 3.7: The stories in the point cloud (left, black) can be evaluated with the peaks of the histogram of the points of the path of the scanner (left, in colour) in Z-direction (right)

If available it is therefore recommendable to use the path or location of the scanner to solve this problem. For the ZEB1 scanner, for example, this data is usually available as a trajectory consisting out of another point cloud. Now, a similar approach as for the building's point cloud can be applied and a histogram can be created. As the scan is usually performed by walking around in the area which is meant to be captured, the trajectory is for the most part parallel to the slope of the floor, so horizontal. The distance between the height of the trajectory and the next lower peaks in the histogram of the building provides answers to the question for which peak in the building's histogram corresponds to a floor and which one is a dropped ceiling or a table. The distance should represent the height of the scan, which is performed within a certain height threshold above the ground.

Using these methodologies the storeys and the location of floors can be found and the attribute for storey height can be added to the octree. From now on, each storey can be treated and looked at independent from each other for the most of following steps. The octree allows a simple projection of attributes. Every black leaf node of the octree having the same Z-values

Figure 3.8: A locker covering a large part of the wall behind it. It might result in a higher peak in the histogram as it has a bigger area to reflect points to

as the location of the histogram peaks determined before (location of floors) can be assigned to the corresponding characteristic. The leafs with Z-values of in between the peaks can get the according enumeration of the storey. Generally this simplification due to the projection can also cause wrong attributions, for levels in the building or the attribute to be a floor. This, however, is not harmful for further processing as intended in this research. Furthermore, as already stated, the enumeration of levels might not correspond to a human perception of a storey as a landing, so the intermediate platform between flights of stairs, might get its own number. Such situation could be fixed by setting a minimum area for a floor to get a new, full number.

3.2.2 Identification of walls

Similar to the methodology explained in Section 3.2.1, histograms can be used to detect the walls in a building as well. As the Manhattan-World assumption applies (Section 3.1) this means that the peaks need to be found in X- or in Y-direction, instead of in Z-direction. Khoshelham and Díaz-Vilariño [2014] state that adjacent histogram peaks with a smaller distance than the maximum wall thickness correspond to the non-navigable space of a wall. This applies, with constraints for furniture, also to the research performed throughout this thesis. The maximum wall thickness has to be a manual parameter.

Walls and obstacles can both appear as a peak in the histogram of either direction. Therefore, to differentiate between the two proves to be one of the biggest challenges for the identification of walls. Especially large cupboards with a flat surface are hardly distinguishable from walls at all. Khoshelham and Díaz-Vilariño [2014] and many other works bypass such false assignment by restricting their methodology to uncluttered point clouds, thus data acquired in empty buildings only without any furniture.

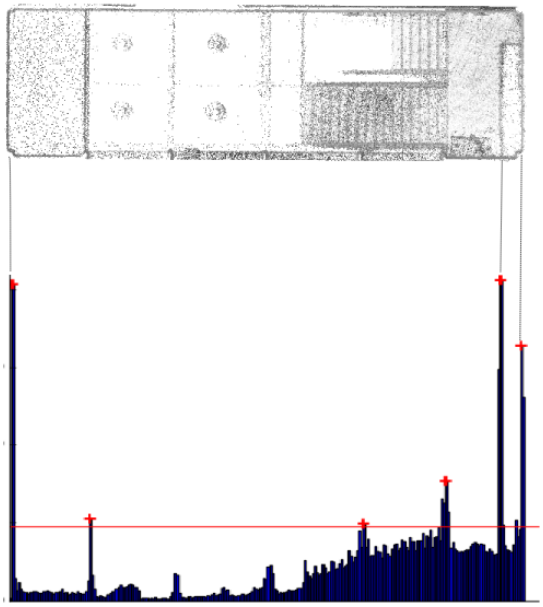


Figure 3.9: Histogram in X-direction showing peaks at the location of walls. Peaks on the right show the locker and the wall of Figure 3.8

The separation of walls and obstacles, however, can be important. Generally, they do both belong to non-navigable space, but obstacles are more likely to change their position than walls. Such information can be crucial in emergency situations, especially when the date of the last scan is long in the past. The height of a peak in the histogram cannot be used to distinguish between walls and obstacles and is misleading as a higher density or area of points scanned can also be on a cupboard rather than on a wall, like shown in Figure 3.8. Therefore following assumptions are made to distinguish between the two. It is assumed that main walls are load-bearing and if histogram peaks appearing in multiple storeys are with a higher certainty a wall and not an obstacle:

- The main walls are more likely to appear in multiple storeys, that is why it is important to identify them. Peaks above each other are more likely main walls as they represent load-bearing walls. In the majority of cases wall directions are shared across different levels of the building [Oesau et al., 2014].
- The first peak from either side of the histogram is a main wall. One of the general preconditions (Section 3.1) of the data was that the point cloud has to be of an indoor environment. As the acquisition therefore also has to be performed from the indoors, the first high peak on either side of the histogram represents the location of a main wall. In this case, this is a wall connecting to the exterior of the building.
- The peaks close to main walls and not in other storeys are more likely obstacle and not another wall, the same accounts for three close peaks, were at least one of them must be an obstacle.

Using these assumptions together with the histogram approach does provide a first estimation for the walls' locations. The model can be labelled accordingly and the attributes be projected to the corresponding X- and Y-values in the octree, similar to what has been explained for the floors

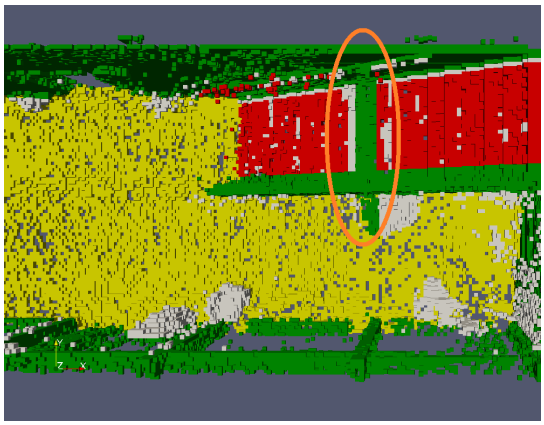


Figure 3.10: Wrong assignment of walls instead of stairway due to wall projection at histogram peak

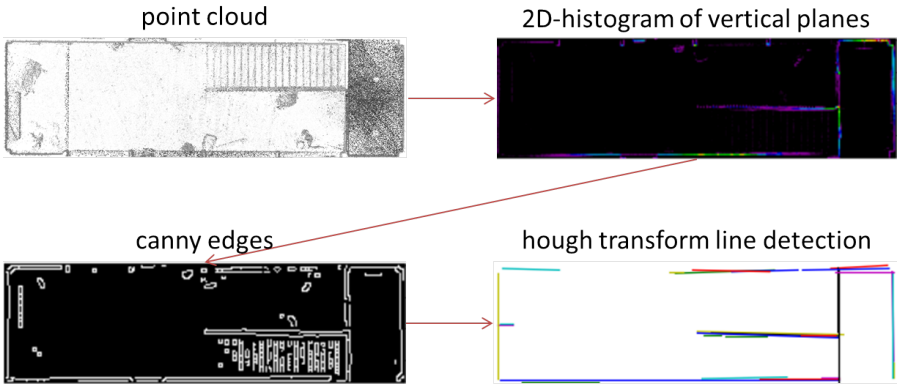


Figure 3.11: Workflow to detect vertical features like walls in point clouds. Top left: Ground level point cloud; Top right: vertical planes in octree leafs stacked to 2D histogram; Bottom left: Canny edge detection to reduce noise; Bottom right: line detection through hough transform, lines represent walls

and storeys in Section 3.2.1. In contrast to the methodology of Khoshelham and Díaz-Vilariño [2014], there is a validation of the walls necessary in this research. Khoshelham and Díaz-Vilariño [2014], besides using a shape grammar, exclude all furniture and staircases from the point clouds. Wrong projections are therefore not possible as every point in a certain height and direction can exclusively be a part of a wall only. The point clouds used here may have clutter and stairs, and therefore it needs to be determined whether all wall attributions do represent a large vertical plane.

Figure 3.10 shows a case where the histogram for the walls in Y-direction found a peak, which resulted in a wrong assignment of a wall where there is a stairway in reality. Such cases, that do not only happen with stairs, but also with other obstacles and clutter do make a validation of the walls necessary. It cannot simply solved by increasing the threshold for a peak to be found, because then relevant peaks might be lost as well.

A validation can be performed reusing the approaches of Okorn et al. [2010] or Oesau et al. [2014], who are using the hough transform line detection algorithm on 2D histograms. Figure 3.11 illustrates the workflow to detect walls in the point cloud reusing and extending the method of their research. For this workflow the 2D histogram can be built with leafs of the

octree that include vertical planes. In the octree structure a plane has to be fitted through all points in each leaf in the storey. To increase robustness also the points in each of the leafs' neighbours can be added. The direction of the normal can be detected using the singular value decomposition or the covariance matrices. When the normal corresponds to a vertical plane in the octree's leaf it is stacked to a 2D grid histogram of vertical leafs, where the grid size is equivalent to size of the leafs. All leafs containing vertical surfaces are projected on the grid so that each cell in the grid provides a measurement about how many leafs above this cell contain a vertical surface. This results in a 2D histogram with peaks at locations with many vertical surfaces. Figure 3.11 shows such a histogram in the top right corner, where the colours are brighter when the histogram bin is higher.

Peaks in the 2D grid histogram represent lines, which can be found using a Progressive Probabilistic Hough Transform [Galamhos et al., 1999]. This methodology assumes that a random subset of voting points can provide a good approximation of the actual results, where lines are extracted during the process by walking along connected components. Using a canny edge detector decreases noise and the number of operations necessary, which is proportional to the number of lines in the image, the edges need to be detected in van der Walt et al. [2014].

Each line must be within a threshold to the wall detected in the previous approach, else the previously detected wall is more likely to be an obstacle or corresponds to stairs like in Figure 3.10. As soon as the walls are exceeding the threshold to the distance to a line, the attribution can be removed.

There is a large potential to build up on this and to also extent such methodology to Non-Manhattan-World structures. For example Oesau et al. [2014] successfully use such approach for reconstruction. Furniture or other clutter will always be the cause of difficulties, though.

3.2.3 Identification of stairs

To be able to connect and find a path between different storeys, the stairs need to be identified. Like introduced in Section 2.1.2, there are many different approaches to do so, but they usually rely on the point of view of the scanner. Staircases and stairs are crucial to the derivation of a pathfinding model for humans from 3D indoor point clouds as they do not only form a connection between different storeys in the building, but also are part of the walkable space.

The approach for the identification of the stairs is initially based on reusing the method of Bansal et al. [2010, 2011]. Other methodologies described in 2.1.2 were often depending on the point of view of the scanner or had the focus on single steps and their attributes like the height of the riser (which are important for robotics) instead of on the whole stairway. The proposed methodology has the advantage of taking the point cloud as a whole into account, because it uses the point cloud on a global approach. It then analyses at which locations stairs can be identified.

Specific preconditions for stairs

Stairs come in various shapes and forms, the majority of the algorithms to detect them are limited to straight stairs and do not work for spiral stairs. Even though it could not be tested, it is not assumed that the methodology presented in this section does work properly for such cases. Therefore,

the constraint to follow the Manhattan-World assumption also applies to the stairs. Furthermore, the point cloud needs to be complete and should not contain gaps caused, for example, by the viewing angle of the scanner. Such gaps usually occur with stationary scanners and result from the static scanning. Therefore, mobile laser scanners are preferred in this scenario. Stationary laser scanner can cause the same stair facilities to be represented completely different in the point cloud, depending on the line of sight of the scanner.

Bansal et al. [2010, 2011] take voxels in point cloud chunks with points and compute a plane normal of the points inside each voxel or within a radius of the its centre. In this workflow the full octree leafs, located at the storey where stairs have to be found in and which are not classified as walls yet, can be taken into account instead.

Filter based approach

In the following, the steps for the algorithm to find stairs are described. If not explicitly explained, the methodology is close or similar to the one of Bansal et al. [2010, 2011] and also described in Section 2.1.2.

1. First, the normal of all points in each leaf and its face neighbours has to be found. This can be done using the Singular Value Decomposition (SVD), so a factorisation of the matrix. The last row of the unitary matrix of V represents the eigenvector of the smallest eigenvalue and therefore the normal. Also the covariance matrix can be used giving the same results. The neighbours can be found using the methodology described in Vörös [2000].

2. Using the normal of each leaf, it is checked whether the plane through the points corresponds to a vertical, horizontal or a differently oriented surface. The methodology is similar to the approach for building the 2D grid histograms for the hough transform to validate the walls, explained in Section 3.2.2.

3. Based on the smallest leafsize of the octree a 2D grid histogram is created. All leafs containing vertical surfaces are projected on the grid so that each cell in the grid provides a measurement about how many leafs above this cell contain a vertical surface. This results in a 2D histogram with peaks at locations with many vertical surfaces.

4. The previous step is repeated with leafs containing horizontal surfaces. The 2D histograms are not illustrated here because they are hard to visualise, Figure 3.13 shows how they look like after the filters, which are explained in the following step, were applied.

5. The filters for the vertical and the horizontal histogram are created and their responses to the histograms computed. The filter for the vertical histogram is a rectangular block having alternating rows of positive and negative weights. The horizontal matched filter is a box-car filter to be able to detect continuous horizontal structures. Both filters combined correspond to the direction changing structure of the surfaces in stairs (see Figure 3.12).

6. The filters' responses are combined using the geometric mean.

The following steps extent and change the methodology of Bansal et al. [2010, 2011], who perform a non-maximal suppression to suppress spurious

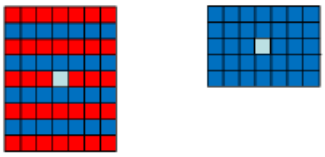


Figure 3.12: Matched filters: stair vertical filter (left) and horizontal filter (right), blue represents positive filter weights and red negative ones, *image courtesy of* Bansal et al. [2011] [*changed*]

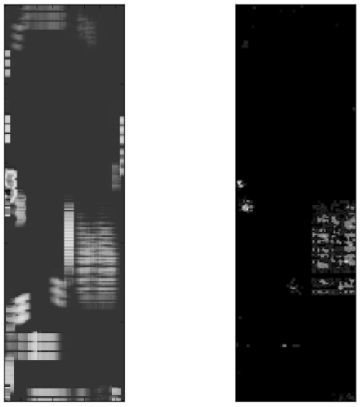


Figure 3.13: Response of vertical filter on the left, response of horizontal filter on the right. The figures can be seen as a top view on the scene of the storey, where positive responses are in brighter colours than negative ones. Walls and other already labelled leafs in the octree are not used for the 2D histogram the filter is applied on

responses instead. Changes needed to be made as the stairs could not be found due to the functionality of the vertical filter. Figure 3.13 shows how many, even smaller regions where found and identified as stairs. Many of them correspond to flat surfaces, which were not identified to be floor, but are rather caused by obstacles. The weaknesses of the method and especially the filters will be elaborated on in Section 5.2.3.

To overcome these limitations following extensions of the methodology are necessary:

7. Regions get grown on the filter responses using a Gaussian filter to treat possible stairs as one area (see 3.14). Otherwise all leafs are treated separately, which limits the possibilities for further analyses. On top of that, it can be used to filter out too small areas that can impossibly be stairs and do more likely represent obstacles. It was also considered to use the shape of the resulting regions found, but as there could have been an obstacle placed on the stairs it has been excluded from the methodology.

8. The heights of different pixel rows per region are compared and outliers are removed. The height of each pixel is its Z-value, which is important for the calculation of the slope of the area. Outliers can be identified with least square adjustment, linear regression or other methods, like the absolute scaled distance to the median of all values. The assumption is, that stairs are shaped regular and outliers might be artefact from the point cloud or of the handrail. Therefore, they can influence the slope estimation.

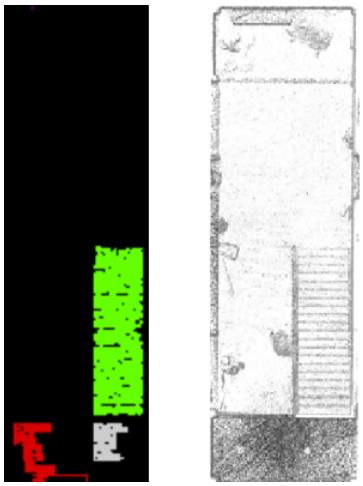


Figure 3.14: Areas in different colours (left), grown of combined positive filter responses. The cut through the point cloud on the right is for better orientation

9. Also, the Manhattan-World assumption of the stairs can be used. Each row and column of the grid which belong to the bounding box of a region can be looked at separately. If there are too many pixels per row or column, which do not have a positive value from the filter response they are removed because it can be assumed that stairs have a close to rectangular shape. The red region in Figure 3.14 shows such an artefact on the right side, where a line exceeds the area.

10. When all outliers are removed the slope of a region is calculated. Is the slope in between the dimensions of stairs, the region will be labelled a staircase. Interpreting the numbers of Table 3.1 concludes to a maximum slope of the pitch line for stairs in the Netherlands should have an angle to the floor of about 43.5° for residential buildings and about 37.9° for all others buildings. In the other direction, the slope should be close to 0 accordingly, in other words: stairs should not be tilted. Furthermore, the slope direction can be determined to find the orientation of the stairs. Figures 3.15 and 3.16 show the average Z-value of the data or octree leafs in the region. The red line in Figure 3.16 is the aggression line to estimate the slope of the region in each direction. When the region and slope are badly conditioned it will not be identified as stairs.

11. Flat areas with not enough slope or regions with a low linearity, which can be detected using the Pearson's coefficient, are removed and will not be detected as stairs.

Figure 3.17 shows the most important steps in the workflow of the approach to discover the stairs in the point cloud scene.

3.3 DERIVATION OF THE GRAPH

Once the octree structure is semantically enriched, the new information can be used to get a better and smarter graph than before. In Broersen et al. [2016] the shortest path would just fly through the building, independent

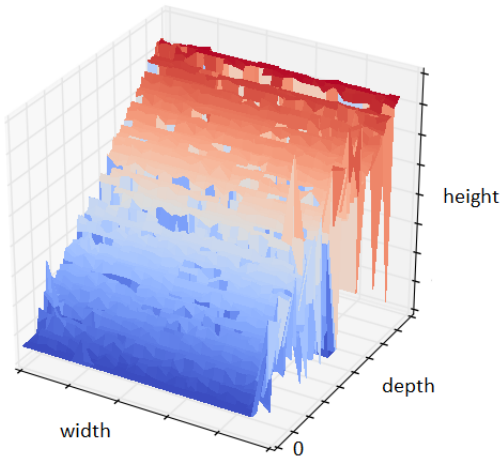


Figure 3.15: 3D raster of the data from the green area in Figure 3.14

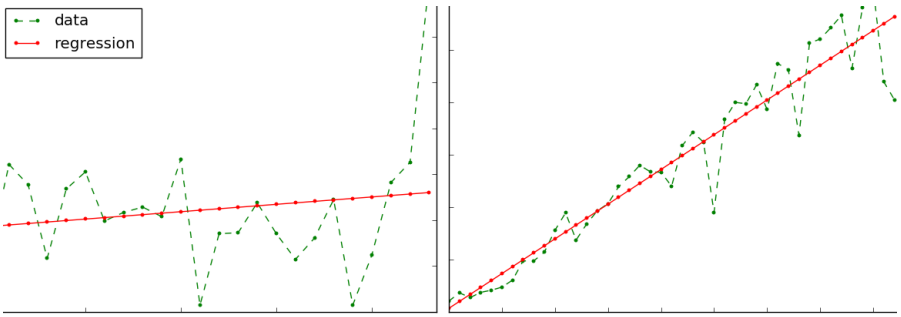


Figure 3.16: Linear regression in X- and Y-direction of the same data as in Figure 3.15

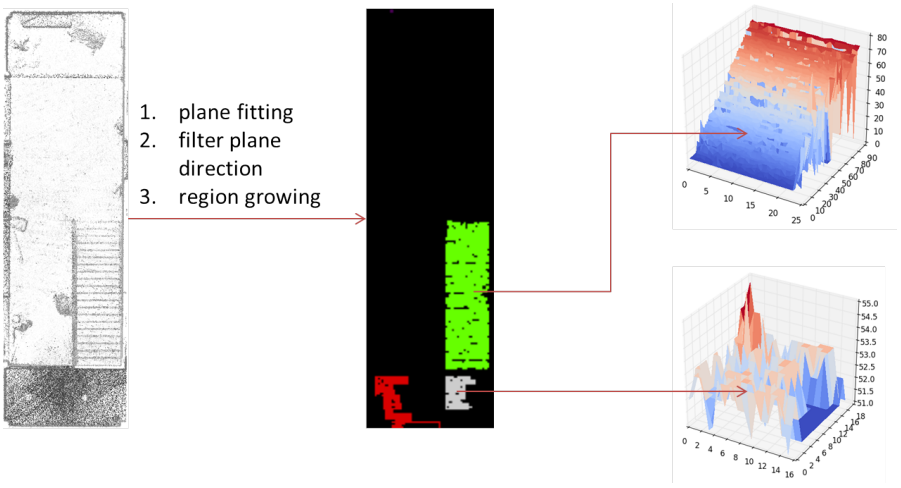


Figure 3.17: Workflow to detect stairs in point clouds. Left: Ground level point cloud; Centre: regions in 2D histogram after plane fitting, filter application and region growing; Top right: 3D raster of region complying to slope and tilt for stairs; Bottom right: 3D raster not complying to the requirements

Table 3.1: Staircase dimensions in the Netherlands, Bouwbesluit 2012 [Minister van Binnenlandse Zaken en Koninkrijksrelaties, 2015] [*changed*]

	residential	other
Minimum width of the stairs	0,8 m	0,8 m
Minimum free height above the stairs	2,3 m	2,1 m
Minimum tread depth per step	0,22 m	0,185 m
Maximum rise height per step	0,188 m	0,21 m

from what is underneath or if a human actor could actually use this path. There was no understanding of the scene and the surroundings of the graph. The goal of the semantic enrichment was not only to reconstruct the building, but aimed specifically at providing a possibility to derive a smarter graph. The graph should be restricted to only walk on top of floor and stairs, thereby sticked to the ground and to go around obstacles.

3.3.1 Derivation of the graph on floors

To derive the graph on floor level it is first necessary to project the 3D empty space of the octree to 2D. The graph can still be layered for the different storeys, but needs to be 2D locally as human tracks are also 2D, horizontal and sticked to the floor. It is thus necessary, to create the graph on floor level. The derivation of the graph at the stairs follows another approach as it cannot be horizontal and will be described in a separate Section 3.3.2. A straightforward solution to find such 2D graph would be to derive the 2D equivalent of an octree from the current structure, a quadtree. This would also allow an implementation of the methodology for a distance transform of Samet [1982]. A distance transform is necessary to create a clearance map or graph that makes sure that at all places there is enough space for the actor to move through. The distance transform is the value of the shortest distance of each empty space entity to an obstacle or generally non-movable leaf. However, deriving a quadtree would not make use of the current data structure and give up a lot of information in vertical direction. That is why a cut of the octree can provide a more complete picture of the set and is also cheaper to calculate with the current set up. A cut of an octree is not equivalent to a clean quadtree structure. It is a challenge though, to find an appropriate cutting height from which a good network can be derived using the principle of the Poincaré duality (see 2.2) and at the same time keeps all necessary information.

Octrees are regularly structured, therefore a cutting height can be found using the characteristics that the octants of empty space are as big as possible and the floor horizontal. This guarantees, that the first time the Z-value of the octree's leafs, higher than the floor, can be divided by 8 is an advantageous cutting height. The attributes of the octree and its regularity make sure that this is a height where the leafs are completely covering the empty space in a relatively small grid like shown in Figure 3.18. The first time the Z-value can be divided by 4 is also the first time a new, complete grid of new leafs start. As there is floor underneath or the graph should only exist when this is the case (because the empty space should only be usable with floor underneath) it is also assured that there is no bigger leafs starting in a lower Z-value. Larger leafs having their origin lower are therefore not part of the grid. However, dividing by 8 leads to better results as this makes the

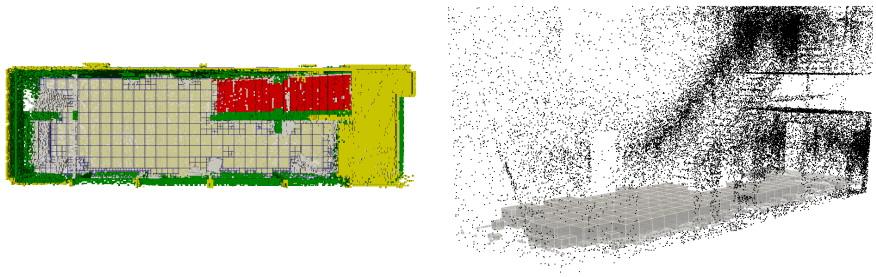


Figure 3.18: Left: Cut of empty space (transparent) in semantically enriched octree of first storey level of scene in top-view, where floor is yellow, stairs are red, walls green and obstacles grey. Right: Cut of empty space (grey) in thinned point cloud at first Z-height divisible by 8 above the floor height

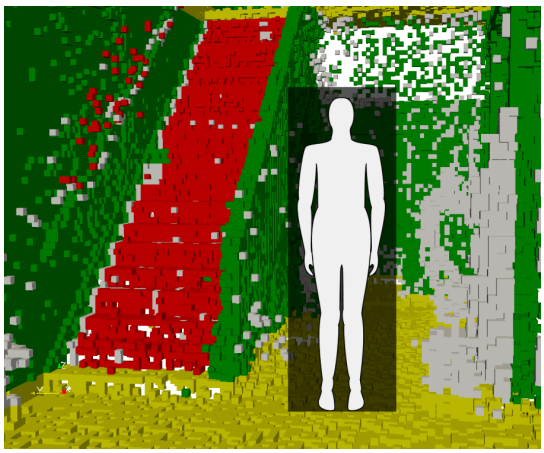


Figure 3.19: Human actor with bounding box in semantically enriched octree scene, empty space is not shown

grid less susceptible to noise and a guarantees a larger resolution. On top of that, dividing by 4 or 8 might even lead to the same height in some cases.

Once the cut is performed it needs to be tested whether all the leafs are really free for navigation. Therefore, it needs to be known whether there is floor (or stairs, see Section 3.3.2) underneath and no obstacle above, up to the actor's maximum height. Figure 3.19 shows a human actor standing in the scene displayed earlier in Figure 3.18. The dark bounding box around him illustrates the space that should be empty or free for him to comfortable navigate through. The following sections will demonstrate how it is made sure that a network graph will only deploy free leafs.

Checking space vertically

Moreover, should be tested, whether there is enough vertical space for a person to move. First of all it needs to be checked, whether there is enough floor underneath. It is recommendable to set a threshold so that the floor only has to cover some parts of the empty leaf's extent. This is because point clouds tend to have holes and gaps especially at the floor, which do not exist in reality. All leafs, found to fulfil this criteria can be added to a 2D map, which represents navigable space and from now on only these are further considered and checked. The map also still holds the locational codes of the octree structure to address each octant.

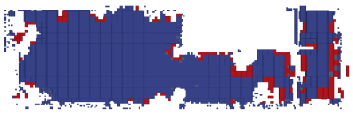


Figure 3.20: Cut of empty space found to be free. Red leafs are further subdivided, blue ones are still in their original size

Furthermore, it needs to be determined whether there is enough space above each leaf before it can be declared navigable and therefore become a part of the graph. Also here the octree structure can be used as there can only be an obstacle to collide with when the leafs above are smaller than the size of the leaf of the cut. If there is no obstacle, the leafs are either bigger or of the same size. The general methodology is comparable with Hornung et al. [2012], but takes the entire height of the actor as constant layer. If for all z_i above the leaf to be checked (x, y, z_i) is free of any obstacles or other occupied leaf, the leaf to be checked (x, y) is marked as free (blue in Figure 3.20). Is (x, y, z_i) occupied and will be further subdivided to test whether a smaller instance of $(x_{smaller}, y_{smaller})$ is free (red in Figure 3.20). When a maximum and predefined level is reached and occupied $(x_{smaller}, y_{smaller})$ are considered as occupied. The resulting 2D map holds only free leafs that can be used for navigation, they guarantee that there is not collision in 3D space above it. The further subdivision, once a leaf is found to be not free, is necessary to prevent marking large areas as occupied, even though only a marginal part of them actually is (see Figure 3.20).

Distance Transform (horizontal)

To make sure that the object or person fits through the width of the space a distance transform has to be performed. The distance transform calculates the closest vertical distance of the centre of each leaf, which was declared free for movement before, to a cell which does not apply to that. Is the distance less than half of the width of the object to be moved, the leaf cannot be a part of the navigation graph because it would not be able to fit through. It is tested, whether there is enough space to horizontally. For this task only the resulting 2D map of Section 3.3.1 is taken into consideration.

To be able to calculate the distance to the boundary of the next cell which cannot be used for navigation the non-movable leafs, sharing the same height, first have to be created. Projecting the walls and obstacles found earlier to 2D does not work, as they can be on different heights. A good example for such situation is an open door. If a door gets projected to 2D it seems like a wall. It is, however, possible to move through and therefore there should not be a boundary. Also a cut of the data between the appropriated heights does not work in all cases, for example where there is no obstacle, but also no floor underneath. Also the empty space, structured reusing the methodology of Broersen et al. [2016], is not sufficient because it is not limited to the necessary height only.

Thus, the reverse of the 2D map created in Section 3.3.1 and shown in Figure 3.20 needs to be found. It is possible to find all locational codes at a certain height in the octree making use of its Z-order and bitwise interleaving. As soon as a number in the locational code is above 3 this means that the leaf is in one of the upper octants in the current level (see 3.21).

Algorithm 3.1 shows how the height can be determined by following these attributes of the octree's locational codes. The returned list of Booleans can

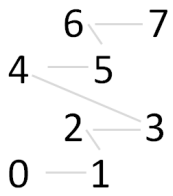


Figure 3.21: Octants enumeration in Z-order

then be used to find all remaining leafs at the height of the navigable leafs as they are complementing each other. This approach resembles the finding of the empty space by taking the reverse of the full octree in Broersen et al. [2016], the Z-order of the octree leafs can be found in Figure 3.21.

Algorithm 3.1: Height to Boolean

Input: locational code of deepest leaf usable for navigation
Output: locational codes of deepest level in booleans according to their height in the octree

```
1 foreach digit i of locational code do  if i < 4 then
2 |  append False
3 else
4 |  append True
5 ;
```

Having both, a slice of the navigable space and a slice of the non-navigable space the distance transform can be calculated. There are several methodologies to do so. The approach for distance transform on quadtrees by Samet [1982] does not work in this situation as data structure is represented by a sliced octree, which does not correspond to a correct quadtree data structure. The computation of the distance between the centre of a navigable leaf to every non-navigable leaf, which keeps the value of the shortest distance for the clearance map is one possible solution. Due to its simplicity it can be implemented quickly, but lacks performance. Another alternative would be to find the neighbour of each leaf, check whether one of them is non-navigable and if this is the case calculate the distance. Is there only navigable neighbours, then their neighbours need to be found. This is challenging though, as the grid is irregular and the first non-navigable neighbour is not necessarily the one with the closest distance. Furthermore, using filters like suggested in Borgefors [1986] are an efficient possibility to achieve the same result for a distance transform.

3.3.2 Derivation of the graph on stairs

The graph network on the stairs can be derived using a similar methodology as the one on floor level. Furthermore, it needs to be connected to adjacent floor networks to enable multi-storey pathfinding. There are, however, some differences between the methodology to derive the graph on floor levels. For example the height of the cut has to be floating above the stairs instead of above the floor. Due to the slope and irregularity of the octree it is not possible to simply cut the structure as it is done for the graph above floor and described in Section 3.3.1. Instead of using the octree, each edge in the

graph will be equal to the size of the smallest octree leaf and the graph on the stairs will be similar to a graph in a voxel based approach. This means, that the X- and the Y-coordinate of the origin of each leaf are represented by the full coordinates, which are above stairs. The Z-coordinate, which is necessary to get the locational code, can be a value close above the average step height in the octree.

Furthermore, the graph at the stairs need to be connected to adjacent floor on each level, so at the start and the end of the stairs' flight. Therefore adjacent leafs of the cut on floor level have to be found and the neighbours, as well as their distances, added to the network. Optimally building the distance or clearance map is performed afterwards on each storey, to keep the connection intact. The distance transform works in the same way as described in Section 3.3.1 and 3.3.1.

3.4 CONNECTING TO OUTDOOR REFERENCE SYSTEMS

It is important for emergency responders to also keep a connection to the environment of the building. A completeness of the scene including the outdoors is necessary to provide a full understanding of the situation. Also a path is not limited to the indoors and often the closest and safest route includes leaving the building. van der Marel [2016], assistant professor of Faculty of Civil Engineering and Geosciences at the Technical University of Delft, who's research focuses on continuously operating Global Navigation Satellite System (GNSS) networks, states that real world coordinates can as well be used indoors. However, the majority of all point clouds uses its own reference and coordinate system. The dimensions are in tact, so it is possible to calculate distances, but the coordinates and orientation are lost in relationship to the outside world. Therefore it is necessary to create control points in order to be able to align the indoor point cloud with a global coordinate system. The real world coordinates of the control points have to be known and the points should be able to be registered in the point cloud of indoor space. Automated point cloud registration and alignment to known reference point clouds can be performed using methodologies, like Iterative Closest Point (ICP) [Tamas and Goron, 2012] or fast feature points histograms (FFPH) [Rusu et al., 2009b].

Alternatively, if a scanner has GPS on board the reference points can be the locations of the scanner, provided a GPS connection can be achieved with a high enough accuracy. If it is a mobile laser scanner, the loop for the SLAM can be closed at a point where there is a GPS accessibility, like in front of the building. Inertial Measurement Unit (IMU) can provide additional information for when the scanner is used.

4 IMPLEMENTATION AND EXPERIMENTS

This chapter will describe the implementation details of the workflow described earlier in Chapter 3. It will be more specific about what has been implemented and which techniques have been used throughout this thesis project. The first two sections introduce the tools and data to be used for the proof of concept, then the implementation of the workflow will be explained. Finally, in the last sections the performance will be shown.

4.1 TOOLS AND LIBRARIES

For the implementation the high-level programming language Python and the object-relational database management system PostgreSQL were used. The most important (non-standard) Python packages or algorithms used for the implementation of the workflow are listed in the following:

- Psycopg2 2.6.2 [Varrazzo, 2015]
- NumPy 1.10 [van der Walt et al., 2011]
- Matplotlib 1.5.1 [Hunter, 2007]
- Scikit-image 0.11.3 [van der Walt et al., 2014]
- `detect_peaks` [Duarte, 2015]

Moreover programs like CloudCompare 2.6.2 [Girardeau-Montaut, 2016], FME Workbench 2016 and ParaView 5.0.0 [Ahrens et al., 2005] were used, but mainly for the visualisation of the results. Some processing, like for example the rotation of the point cloud (Section 4.2.2) were also done using CloudCompare.

4.2 DATA

The data to be used in this research consists out of 3D indoor point clouds acquired with different kinds of scanners. The resulting point clouds must be available in LASer File Format (LAS) to be used in this workflow. If the file is not available in this format, it can be converted with CloudCompare.

4.2.1 Choice of scanner

The used point clouds was acquired using three different kinds of laser scanners. Besides the entrance area of the fire brigade in Berkel en Rodenrijs, which was scanned with all scanners, also point clouds of other areas in the building or the Bouwpub of the Technical University of Delft were available. Two of the used scanners for acquiring point clouds were mobile, the ZEB1

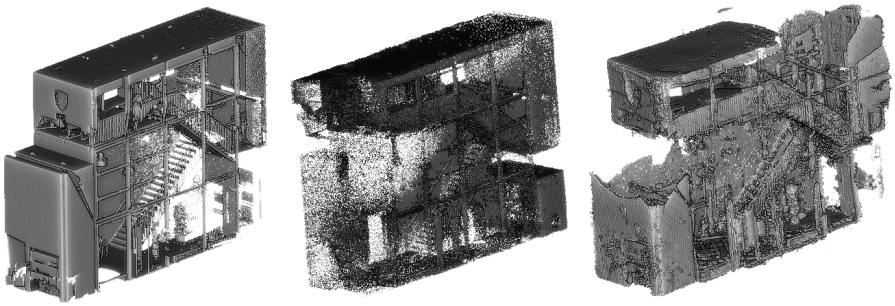


Figure 4.1: Three point clouds of the same scene with different laser scanners after application of an Eye-Dome Lighting shader for better visualisation: Stationary Leica C10 laser scanner (left), the ZEB1 hand-held mobile laser (middle) and the Project Tango tablet (right)

laser scanner and the Project Tango tablet. Moreover, a stationary Leica C10 laser scanner was used to acquire LiDAR scans. The resulting point clouds have different attributes in terms of quality, precision and completeness. Furthermore, another part of the building, scanned with the ZEB1 was also used.

4.2.2 Rotation

Besides the already mentioned octree structure the data needs to fulfil some other attributes to be applicable for the following methodology. As described in Section 3.1 the data needs to be aligned to follow the Manhattan-World assumption. Besides only having perpendicular and straight walls, they also need to be axis aligned. However, this is rarely the case for point clouds, even though they usually use their own coordinate reference system. It is therefore necessary to rotate the point cloud so that the main walls are parallel to either the x- and the y-axis. Main walls are important, load-bearing and large walls, which occur in many storeys. There are several methodologies in which way this can be achieved and can be done manually, semi-automated and completely automated. Examples for axis alignment can be found in Khoshelham and Díaz-Vilariño [2014] or Bansal et al. [2011]. To detect planes in the point cloud the RANSAC implementation of Schnabel et al. [2007] for example in CloudCompare can be used. The main plane's normal can then be found and the angle to one of the axes calculated. The resulting angle is the angle the point cloud needs to be rotated to fulfil the Manhattan-World requirements, under the condition that all walls in the scene are either perpendicular or parallel to each other. If the requirements are fulfilled and one of the main walls is either parallel or perpendicular, then it should also be fulfilled for all other walls.

The rotation parameters can also be of importance for the connection of the pathfinding networks in the indoors and outdoors, which were further explained in Section 3.4. They can be necessary for the transformation of one coordinate system to the other.

For the rotation itself, some manual steps are necessary with the current implementation. All manual steps can be carried out in CloudCompare. First, RANSAC needs to be performed to find planes through the walls of the building. Second, a plane representing a wall and thus, should be parallel to one of the axes needs to be manually selected and exported to an *.xyz-file. This file can be opened in CloudCompare again and the parameters of

the normal can be found under properties. Now the rotation parameters, so the angle between the plane's normal and one of the axes can be calculated using the Python SymPy- and Math-packages. The resulting angle can be used as a rotation parameter in CloudCompare to rotate the point cloud. Now the point cloud is axis aligned and follows the Manhattan-World assumption. As most buildings have levelled floors the assumption is made, that the point clouds are horizontally aligned. In other words, it is assumed that the floor is parallel to the X,Y-plane. This was also the case for all point clouds tested during the implementation.

4.2.3 Octree

Using the methodology of Broersen et al. [2016] a linear octree of the point cloud and the empty space is created. Their script was slightly changed, to fulfil the needs of this research. The most important adjustment is that it now calculates and returns the size of the smallest leaf in the linear octree and the scale. This information is necessary to create some parameters of other functions which are needed later (like the wall thickness). The scale can be used to translate the point cloud back to its original extent. Furthermore, the point cloud can be de-noised by activating a simple query in the script. The query adds a threshold for the point count of each leaf that must be reached for a leaf to be part of the full octree. In other words there must be $x > t$ points in a leaf to not be considered as white and thus empty space.

It was found that an octree of 8 levels gave the best results for buildings with a diameter between 15 and 20 meters. However, the level is less important than the size of the smallest leaf of the octree. This size determines the bins of the histograms and how big around each leaf is used to fit planes through the points. In all tested point clouds the best working leaf size was slightly above 5 centimetres. When the smallest leafs were bigger, the results were much worse and especially the stairs could not be found.

4.3 SEMANTIC ENRICHMENT

In the following section the implementation of the concept for semantic enrichment for the purpose of multi-storey pathfinding (see Section 3.2) will be described. When the same implementation details are used for different purposes (like the 1D histograms for floors and walls), they will be described the first time they occur in the workflow.

4.3.1 Identification of floors and separating storeys

The first and most important step in the semantic enrichment is the identification of the floors and the separation of storeys. It is necessary for all following steps because for the identification of walls and stairs the storeys need to be handled individually.

Creation of histograms

The projection of point clouds to a 1D histogram is an approach, which is frequently used for point cloud reconstruction in recent research [Okorn et al., 2010; Oesau et al., 2014; Khoshelham and Díaz-Vilariño, 2014]. The histograms are built on top of the octree structure of Section 4.2.3. Directly

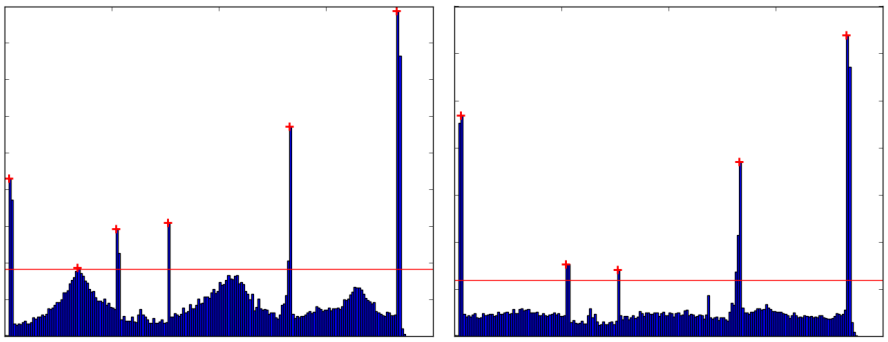


Figure 4.2: The Z-histogram projected directly from the points (left), and projected from the black octree leafs (right). The result of the octree is less effected by the density of the point cloud. The red line is the minimum peak height

using the LAS containing the LiDAR point data records was less sophisticated (see also Section 3.2.1). Instead the method was implemented using the black leafs of the linear octree. Experiments throughout this research have shown that this was not only faster, but also gave better and especially cleaner results like shown in Figure 4.2. The size of the smallest leaf of the octree, which is usually between 5 and 10 centimetres has proven to be a good bin size as well. The histograms can be visualised using the Matplotlib package [Hunter, 2007].

There are many different possibilities to detect peaks in 1D histograms. The repository of Tournade [2015] provides an overview of several algorithms for peak detection, which are all written in Python. The first possibility listed there is the `scipy.signal.find_peaks_cwt` approach. This is an obvious choice when working with SciPy [Jones et al., 2001] (the NumPy and Matplotlib packages are a part of SciPy and used in this thesis project). Even though it is working fine for the majority of the cases, the settings were not sufficient as it does not include a minimum peak height. Furthermore, the wavelet convolution approach which is used seems to be very complex. Also other methods like the `findpeaks` by Slavic [2015] provided only limited settings.

Finally, the algorithm `detect_peaks` by Duarte [2015] was chosen for the peak detection as it provided everything necessary for the task, including a minimum peak height and various other settings. This method can also be used for the identification of walls (Section 4.3.1). However, there was one exceptional case none of the approaches took into consideration. Mathematically seen an extreme at the edges of the histograms is not a peak, because it does not go down at either side of the edge. Such situation is likely to occur in this application because lowest floor, the upper ceiling and outer walls should be at the border of the point cloud's bounding box. To overcome this limitation a temporary bin with a height of 0 is added to either side of the histograms. Now, the peak will be detected and the temporary bins need to get removed again, as well as the bin count adapted. The function to find the peaks takes several parameters. Especially the minimum peak height influences the result as it can suppress spurious responses from noise or obstacles. A threshold, which makes sure that a peak has to be at least above 95 percent of all other bins was found to be working well in most cases.

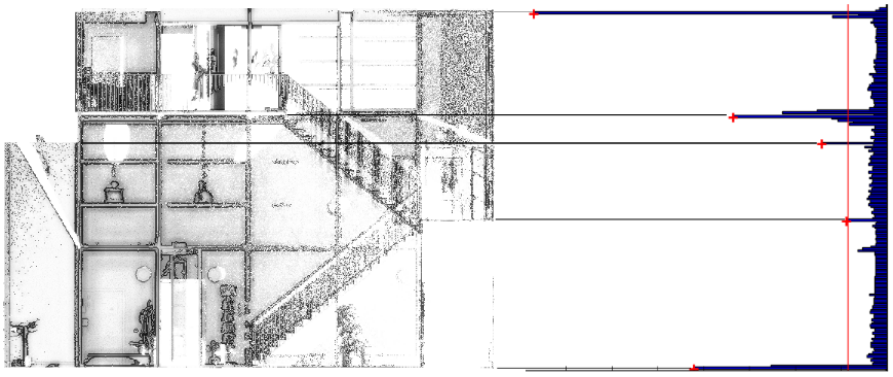


Figure 4.3: Point cloud of Leica laser scanner (left) and Z-histogram with detected peaks. The third peak represents a dropped ceiling

While the location of a peak can be a good indicator of either a floor or also a wall, the height of peak is not (see Figure 3.9). The height strongly depends on the length of the object and on the point cloud density. This however, can cause difficulties when distinguishing between obstacles. A ceiling or floor (to separate from furniture like tables) or a wall (to separate from furniture) can have a smaller peak size, as for example a large cupboard covering a wall. In such case the LiDAR response are of the cupboard can be bigger than the one of the wall behind. Nevertheless, small obstacles or clutter can be removed from the peak detection by using a minimum height parameter.

Identification and verification

The peaks of the histogram in Z-direction are used to separate the storeys in the building from each other and to identify the floors (see Figure 3.6 and Section 4.3.1). This method was also used and explained in Okorn et al. [2010]; Oesau et al. [2014]; Khoshelham and Díaz-Vilariño [2014].

However, the results need to be verified because of peaks resulting from either dropped ceilings or table surfaces that might lead to wrong assignments. Figure 4.3 shows a case of a peak, which does not represent a new storey and which is further apart from another one than the maximum wall thickness. This peak is caused by a dropped ceiling.

One indicator which can help to verify a storey is the minimum storey height and thus the distance of the peaks in the Z-histogram. The minimum storey height needs to be set as a parameter, and peaks closer than this value can be ignored. However, if there are both, table-like structures covering large surfaces and dropped ceilings, it can still not clearly be distinguished as it is hard to decide which peak represents the floor, the ceiling or the table. The path of the scanner can overcome this difficulty. The implementation of the usage of the path of the scanner to separate the storeys is explained in the following.

If the path of the scanner is available, for example when using a ZEB1 laser scanner, it can be used to validate a storey. Furthermore, it can be used to verify, that it is really possible to walk on this structure, thus whether the peak in the histogram Z-direction represents floor. The implementation is similar to finding the Z-value of floors in the point cloud described earlier (compare Figure 3.7). Considering that the average shoulder height is of about 1 meter 40 centimetres [Adler, 1999] the height of the path scanned

with the ZEB1 should vary in between 1400 and 1700 millimetres above the floor ground. Experience showed that the scanner is usually carried at such heights. To use the path of the scanner, the file needs to be placed in the same folder of the point cloud and named $"filename" + "\_traj.las"$.

Projection to octree

Once the information about the storeys is found the data can be projected to the octree. Every histogram bin represents one row or column of leafs in the deepest level of the octree. The semantic information, which can be written to the database is about the object classification the leaf represents. So far this can be the floor or the storey level. From now on, the data for each storey can be retrieved separately from the PostgreSQL database. The analysis can therefore happen independent from other storeys.

4.3.2 Identification of walls

The detection of walls is restricted to those, which follow a Manhattan-World assumption. The peaks in the 1D histogram in X- and Y-direction represent a wall in the model. This approach is very similar to the Z-histogram method used for the storey separation and the identification of floor (Section 4.3.1).

Close Peaks and average wall thickness

The average wall thickness has to be set as a manual parameter as it differs widely for different kinds of buildings and is even depending on the quality of the point cloud. If the point cloud is of a lower quality, holds more clutter or noise, the maximum wall width has to be set to a higher number in order to classify them successfully. Histogram peaks close to each other and within the average wall thickness can usually be interpreted as either side of a wall. The empty space in between represents thus the volume of the wall.

The parameter for the maximum wall thickness exceeds the importance it has in Khoshelham and Díaz-Vilariño [2014]. In that research the parameter only provides valuable information to decide whether an area is empty space or enclosed by a wall. This thesis project however has a stronger focus on pathfinding, rather than about reconstruction. Volumes enclosed by a wall would lack connectivity to any start and end of the route and can be ignored accordingly. The wall thickness has therefore importance for the distinction of walls from furniture, like large lockers. However, in many buildings the wall thickness differs. The wall to the outside is usually thicker than interior walls, which makes finding one parameter for the whole building a difficult task.

On top of that, when the precision is low, like with the Project Tango tablet, the wall thickness has to be set a lot larger than it is in reality or with a more precise laser scanner like the Leica C10 to detect the walls successfully. The reason behind this is the SLAM distortion and the noise of the Tango. This can lead into walls not exactly following the Manhattan-World assumption, even though they do in reality. Also the hough line detection (Section 4.3.2) needs different and less strict parameter settings for the Tango tablet.

Furniture and other obstacles can also be recognised when they show up as peaks in the histogram, using the location and distance of the peaks. In

order to tell the difference between walls and obstacles some assumptions are necessary and implemented which were explained Section 3.2.2.

Lines of vertical planes

The second approach to find the walls was based on the approached of Okorn et al. [2010]; Oesau et al. [2014]. In the current implementation it is only used to verify the outcomes of the previous way. First of all, using NumPy [van der Walt et al., 2011] a 2D histogram is made. The 2D histogram is represented as a grid or an image which can be used for further analyses. For the construction of the histogram SVD [van der Walt et al., 2011] is used. SVD enables to fit a normal through the points and to find the local plane direction. The histogram has then peaks where the stack of vertical planes through all points within the neighbourhood of each octree leaf is the highest. Secondly, image processing techniques of Scikit-image [van der Walt et al., 2014] are used to detect lines of peaks in the resulting 2D histogram representation. If there is no line detected within a proximity of the previously identified part of a wall, the classification will be removed.

The implementation of this part of the workflow is the most expensive part of the whole framework because planes have to be fit through all points in each of the building's storey. It is, nevertheless, necessary in many cases to avoid situations like shown in Figure 3.10.

4.3.3 Identification of stairs

When the storeys are separated and the walls are found, the stairs need to be located in the octree structure. As explained in Section 3.2.3, the stair detection methodology of Bansal et al. [2010, 2011] was not sufficient or could not be implemented properly to meet the requirements of this thesis project. Therefore, the approach was extended. In the following the implementation will be described.

The implementation of the 2D histograms to locate the different plane directions in the model is similar to the one used to create histograms for the wall verification explained in Section 4.3.2. However, the plane fitting is limited to not yet labelled octree leafs. Furthermore, two kinds of histograms are created, including one for leafs having horizontal planes. In the future the implementation of the wall verification and stair detection could go hand in hand to avoid searching for planes in the same octree leafs twice.

The resulting 2D histograms alter according to the stair's characteristics or whether the scanner is mobile or not. If the scanner is stationary the attributes of the point cloud differ like it is shown in Figure 4.4. The figure exemplifies how stairs, which follow the same architectural structure in reality have different appearances in the point cloud depending on the line of sight of the scanner.

As a next step the filters were implemented and applied on the histograms using NumPy. However, the implementation of the matched filter for the vertical histogram, which is supposed to be constructed "as a rectangular block with alternating rows containing positive and negative weights" [Bansal et al., 2011, p. 1809] did not give the expected results. The description of the filters [Bansal et al., 2010, 2011] was not entirely clear and left open questions. Thus, the implementation might not meet exactly what was intended by the authors of the paper. Generally the filter design cannot work throughout all stair designs since not all stairs have a vertical com-

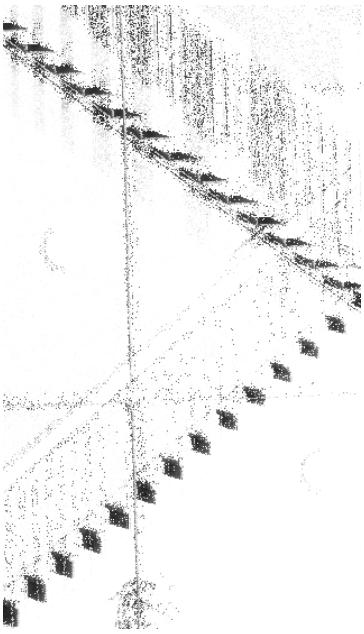


Figure 4.4: Staircase in the Leica point cloud. Stairs lack the vertical (upper part) or horizontal (lower part) component depending on the position of the scanner, which makes both completely different even though they are the same in reality (just the orientation changes)

ponent (riser) as can be seen in Figure 4.5. Additionally the step distance should match the filter's grid size or generally complement to the stair's architectural structure. On top of that, the orientation of the filter is unclear. Finally, experimental results have shown that not applying the vertical filter at all had only minimal effects on the results of the stair detection.

The filter responses are combined with the general mean and regions are distinguished from each other through region growing like explained earlier in Chapter3. To identify the stairs, the slope of each region is calculated. This is implemented by computing the linear regression using SciPy. Disregarding the preconditions for point clouds to be free of noise (Section 3.1), also noisy data was used to test the implementation. Therefore, as the surroundings of stairs in point clouds is sensitive to noise, outliers need to be removed. The removal is important as extreme values have a negative influence on the resulting slope. Ordinary least squares adjustment can be

Figure 4.5: Staircase at the Faculty of Architecture in Delft, with no concrete vertical riser

carried out to do so, but the implementation proved to be not robust enough with rows having too few data values. The absolute scaled distance to the median of all values on the other hand works just fine.

Furthermore, since stairs are restricted to follow the Manhattan-World assumption, the regions in the 2D histogram should have a close to rectangular shape accordingly. Pixel rows with very few data values are therefore excluded. Moreover, each region should have a minimum size to be taken into consideration for a further selection process.

The slope threshold parameters have to be tolerant as well. So does the tilt of the region. The tilt, which should be around 0 in reality, has to be flexible and should be adapted to the quality of the data and filter responses. Figure 3.15 shows why this is necessary. There is a rise of the data at the location of a handrail, which often also fall into the stairs' regions. The requirements for the stairs' regions to have a linear correlation should be set leniently as well. A high Pearson's coefficient denotes a linear correlation and can have values between 0 and 1.

4.4 DERIVATION OF THE GRAPH

To prove the concept and to show the usefulness of the previous work also the workflow to derive a graph from the semantically enriched octree structure has been implemented (see Section 3.3). The graph should be restricted to only use walkable spaces, where a human fits through. The necessary space should be selected according to the space a human needs to navigate comfortably. It can be represented as a bounding box. The implemented extend has half a meter of width and more than 1.8 meter of height. 1.8 meter is usually enough (even though many people are taller), since the lowest cut and base of the graph is floating above the floor and is found according to the method described in Section 3.3.1. In some cases it can also be 20 centimetres above ground, which is not a problem, as humans can step easily over slightly uneven floors. To find the neighbours a modified implementation of the neighbour-finding algorithm of Broersen et al. [2016], which is based on Vörös [2000], is used. In order to find the locational code or to retrieve the coordinates from locational codes the respective algorithms of Broersen et al. [2016] can be reused as well.

PgRouting which was planned to be used for the implementation to build the graph and perform the pathfinding only works in 2D. As the extension does not rely purely on topology, but also on geometry the Z-coordinate is lost. Therefore, this implementation leads to wrong results with the graph jumping up and down between different storey levels. Instead, the topology and graph is stored in a Python dictionary, which can be serialised into a byte stream using the pickle-module. Due to a different height of the leafs at the stairs, the leafs were not always directly adjacent. Therefore, to connect the graph above stairs with the graph above floors the closest, instead of adjacent leaf as proposed in Section 3.3.2 was taken. The distance transform at the stairs was not implemented yet due to the limited time of this research. Moreover, for the current implementation it is assumed that the stairs run in X-direction only. The shortest pathfinding is finally performed using a simple Dijkstra algorithm [Chiu, 2013].

4.5 EXECUTION PERFORMANCE

The workflow was implemented and tested on an Intel(R) Core(TM) i7-4800MQ CPU @ 2.70GHz and 16.0 GB RAM on a 64-bit Windows 7 machine. To give an impression about runtime the workflow was tested with the point clouds shown in Figure 4.1. The following measurements only show the time needed for the semantic enrichment, including building the octree, but without the pathfinding and creation of the graph (see Section 4.4).

- Google Tango point cloud with 750.000 points, takes around 330 seconds
- ZEB1 point cloud with 2.200.000 points, takes around 350 seconds
- Leica C10 point cloud with 4.800.000 points, takes around 530 seconds

The, in comparison, fast execution of the Leica C10 point cloud is attributable to the high density and good condition of the points, which causes more points to be in one octree leaf. Furthermore, the path of the scanner was only included for the ZEB1 laser scanner. This causes another file to be opened, more database queries to be necessary and finally to a longer execution time.

5 RESULTS AND ANALYSIS

This chapter will present the results and analysis of the earlier explained implementation (Chapter 4) of the workflow (Figure 3.1). Figure 5.1 shows what can be achieved by applying the methodology explained in Chapter 3 to 3D indoor point clouds which did not have any semantic information. The colours depict the classification categories of the octree leafs.

The chapter will follow a similar structure as the previous ones and first discuss the results with different kind of point clouds in Section 5.1. Then the floor and wall detection as well as the stair identification will be presented and analysed. Finally, the results for the derivation of the graph will be presented. In the majority of the cases in which the workflow did not work as intended, the point clouds or scenes did not meet the preconditions. To illustrate the limitations of the workflow and to underline the preconditions set in Section 3.1 they will take a prominent part in this chapter.

5.1 DIFFERENT KINDS OF SCANNERS

The quality of the reconstruction and semantic enrichment strongly depends on the device the point cloud was acquired with. A visualisation of the resulting point clouds representing the same building scene from different laser scanner can be seen in Figure 4.1. The following sections will elaborate on the advantages and disadvantages of each scanner and their point clouds used and their effects on the result.

Independent from the scanner every scene is different, so parameters have to be adapted. In total 6 point clouds were used for testing purposes. There is a point cloud of the same architectural scene of all laser scanners and additional ones of the ZEB1 of other parts in the same building. In total 4 of the point clouds were taken with the ZEB1 laser scanner, 3 of them were acquired in the building of the fire brigade in Berkel and Rodenrijs and one

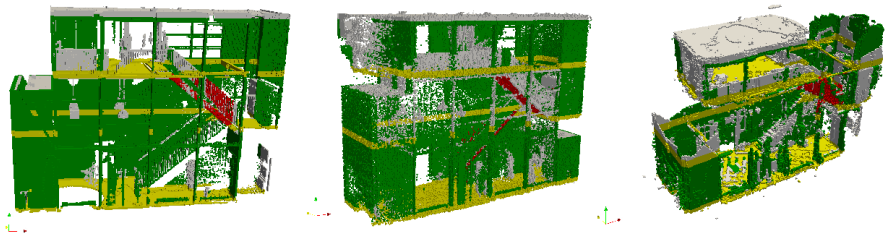


Figure 5.1: Semantically enriched octree of the point clouds of the same scene with different laser scanners: Stationary Leica C10 Laser scanner (left), the ZEB1 hand-held mobile laser (middle) and the Google Tango tablet (right) (to compare with the unprocessed point clouds see Figure 4.1). Green: wall; yellow: floor; stairs: red; obstacles: grey

Figure 5.2: The figure shows a top view of the point cloud acquired with the Project Tango tablet. The offset of the second storey is disadvantageous for the approach using 1D histograms

in the Bouwpub of the faculty of Architecture and the Built Environment at Delft University of Technology. The advantage of using the same building or architectural scene acquired by different scanners is that this makes the results comparable, even using the same parameters.

5.1.1 Project Tango Tablet

The Project Tango is a Google tablet which combines 3D motion tracking and depth sensing. The depth data has the form of a point cloud. The point cloud can be acquired quickly with the tablet and the device is much cheaper than the other scanners. The cost is around 500$ [Google ATAP, 2015]. It produces a 3D mesh from which a point cloud can be derived. The SLAM seems to be losing accuracy over time and distance though. Consequently the walls have an offset in the different storeys of the building, which does not appear in other point clouds. Figure 5.2 shows the offset at the highest storey of the building compared to the other storeys which are parallel in reality. Therefore, the threshold for wall width or thickness has to be set rather wide in order to still follow the Manhattan-World assumption. This makes it almost impossible to differentiate walls from furniture, when they are positioned close to walls.

5.1.2 Leica ScanStation C10

The Leica ScanStation C10 is an All-in-One device that includes a wide range of capabilities [Leica Geosystems, 2016]. The laser scanner differentiates strongly from the other scanners used, because it is stationary. Consequently the point clouds have a high accuracy and density. The floor histogram peaks are rather low as the surface where the scanner is placed on is not scanned. Unless you make multiple scans of the same room, there is a round hole in the point cloud around the scanner's location. The density in the point cloud is inconsistent and related to the distance of the reflection point to the scanner. Additionally there are a lot fewer viewing angles in a point cloud acquired by stationary laser scanners and the results of the detection of certain features is strongly depending on the line of sight. The line of sight needs to be clear and especially for stairs this can cause problems as only one side of them can be fully scanned. Furthermore, the scanner provides colour information, however, this did not play any role in the proposed methodology.

5.1.3 ZEB1 hand-held indoor mapping

The main focus of the workflow and implementation laid on the ZEB1 hand-held indoor laser scanner. The ZEB1 laser scanner is designed to be hand-

carried and the data is captured by simply walking around [3D Laser Mapping, 2016]. The point cloud which was used for the majority of the implementation of the workflow was scanned with a ZEB1. The data was acquired in the entrance area of the fire brigade in Berkel en Rodenrijs. The device holds several advantages over the other scanners: It does, for example, have a more precise SLAM technology on board than the Project Tango tablet and the offset of walls in different storey levels is therefore negligible. On top of that, the point cloud is complete and handles different viewing angles. The scanner also provides the path of the scan, which proved to be a powerful addition to the raw point cloud and can result in an improved semantic enrichment.

5.1.4 Comparison

The results and experiences with the scanners used should be comparable to other devices providing a similar point cloud quality and dataset. The point cloud of the ZEB1 works best with the presented workflow, as can be seen in Figure 5.1. The semantic enrichment is more complete than with the other scanners and all stairs can be identified. Other advantages are a fast acquisition time and the provision of a path of the scanner.

5.2 SEMANTIC ENRICHMENT

In the following, the results of the implementation (see Section 4.3) of the conceptual framework for semantic enrichment of the point cloud, presented in Section 3.2, will be shown and analysed. It puts a focus on the strengths of the proposed framework, but also points out the open challenges and limitations.

5.2.1 Identification of floors and separating storeys

In all point clouds the majority of the floors can be found in the octree structure, as shown in Figure 5.1. Following this, also the storeys in the building can be enumerated and handled separately, so walls can be distinguished. Dropped ceilings and tables represent the biggest challenge in separating the storeys, but if available, this can be solved using the path of the scanner. Also landings in between stairs flights are treated as separate storeys. This is necessary for the subdivision and further semantic enrichment, but can cause a different enumeration as in usual building plans.

5.2.2 Identification of walls

Also the walls can be identified, but the first intention to follow the 1D histogram approach of Khoshelham and Díaz-Vilariño [2014] proved to be insufficient for buildings containing obstacles or other structures like stairs. Using only this approach, the walls could not simply be projected to the octree as this could conduct to false assignments as shown in Figure 3.10. The 2D histogram and line detection approach (Figure 3.11, [Okorn et al., 2010; Oesau et al., 2014]) solved this problem in many cases. It also holds promising results that might lead to a replacement of the 1D histogram approach in the future. On the other hand, using both methodologies increases ro-

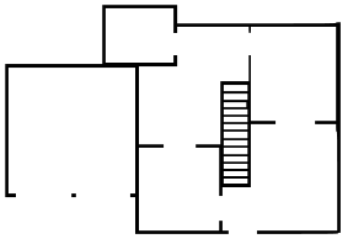


Figure 5.3: Common house floor plan of a building following the Manhattan-World assumption, *image courtesy of* Wikimedia Commons [2012]

bustness of the wall detection. The following sections will elaborate on the findings.

Better classification in regular buildings

Following the Manhattan-World assumption, which limits the building to only have straight, parallel walls and perpendicular corners, the majority of the storeys or walls could be identified correctly. In this thesis a regular building is defined as a construction, where walls appear in all storey, are long and have only a few gaps. Moreover, they do follow the Manhattan-World assumption strictly. This is favourable for the peak detection and for the comparison of the histograms across different storeys. Figure 5.3 shows a floor plan of a building following the Manhattan-world assumption. Nevertheless some parts of the building are irregular and many walls are a lot shorter than others, which can cause problems when determining the minimum peak height of the 1D histograms (see Section 3.2.2). Due to the small surface of such walls, the peaks might be too small to be detected, when using solely this approach. Especially in large buildings small walls are likely to get lost. This is another reason to also use the combination of 2D histograms and line detection to find the walls. Longer walls are still more likely to be detected using this approach, but the overall size of the building is less influential.

The more regular the building structure, the easier is it to find the right parameters, like the maximum wall width. This criterion is difficult to set as walls in the interior of the building tend to be more narrow, especially in respect to outside walls. This can, for example, cause problems for the separation between lockers and walls. For a correct labelling of walls the wall diameter should be comparable in the whole building. Such inconsistency causes a false classification in the point cloud of the Bouwpub, shown in Figure 5.4. The wide walls in the lower left corner of the figure gets wrongly classified as obstacle. The reason for this is the inconsistency of the wall width in the building. The wall is a lot wider in this case than elsewhere in the building and gets therefore partly labelled as obstacle.

A changing ceiling width in the building can cause false classification, too. The ceiling width changes for example due to dropped ceilings, like it is the case in the building of the fire brigade. Nonetheless, the previous Chapter 4 has shown that if a path of the scanner is available, such problems can be solved. The lower right corner of the model in Figure 5.4 shows another case of inconsistency that causes many false classification. The floor in the sitting area in the top right corner of the figure is some steps higher than the rest of the floor in this storey and is therefore not detected as such. Aforementioned situation could be overcome when this part of the floor

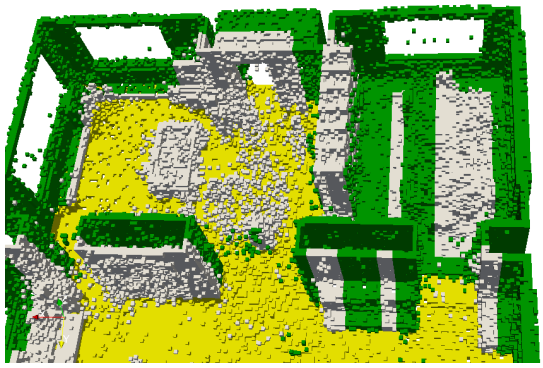


Figure 5.4: Octree leafs of ZEB1 point cloud of Bouwpub, labelled as floor (yellow), wall (green) and obstacle (grey)

would be treated as a separate storey. But this would have a negative effect in many other places because many parameters would need to change. One of the preconditions (see Section 3.1) was to have horizontal and levelled floors only. Due to noise, the hough transform finds many lines at this location and is not able to correct wrong identifications which are caused by the projection of the 1D histogram's peaks to the octree. The Bouwpub would therefore not be an optimal point cloud to start with and the higher area should be classified as obstacle. The following section looks at such situations in more detail.

Distinguish walls from other features

The histogram approach of Khoshelham and Díaz-Vilariño [2014], which leads to their shape grammar methodology, was only tested on completely empty buildings. In this research it was aimed to overcome this limitation and to include buildings with furniture. Even though the majority of the furniture gets detected as an obstacle using the logic constraints explained in Chapter 3.2.2 there are still some challenges left. Figure 3.10 shows a case where the projection to the octree causes a wrong classification of a wall at the place of another feature (here stairs). Such cases could be widely eliminated with the introduction of the methodology of Okorn et al. [2010] and Oesau et al. [2014]. Nevertheless, it remains challenging to distinguish between walls, furniture or other obstacles. All features that cannot be identified as either walls, floors or stairs are classified as obstacle.

Regularly shaped furniture with large surfaces can cause problems and might be wrongly detected as walls. An example where this could happen is a large cupboard, like a locker standing close to a wall. When such cupboard covers the majority of the wall's surface, the wall itself might not occur as a peak in the histogram. However, assuming that buildings follow a regular structure, the peak's location can be compared with the ones in other storeys. Additionally, the peaks at either side of the histogram represent a wall to the exterior. These kind of walls are called main walls. The proposed approach assumes, that peaks close to main walls (but further apart than the maximum wall width) represent obstacles. Figure 5.5 showcases this situation.

In case of a low precision in the acquisition data, like in point clouds acquired with the Project Tango tablet, the wall thickness or width has to be set to a higher value to make up for the inconsistency causing offsets. This causes obstacles close to walls to be labelled as walls, too.

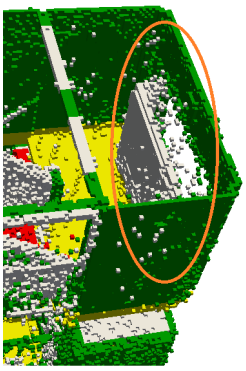


Figure 5.5: Grey obstacle in front of wall (locker in Figure 3.8), which could be distinguished due to regularity of the building. Figure 3.9 shows a 1D histogram of this scene

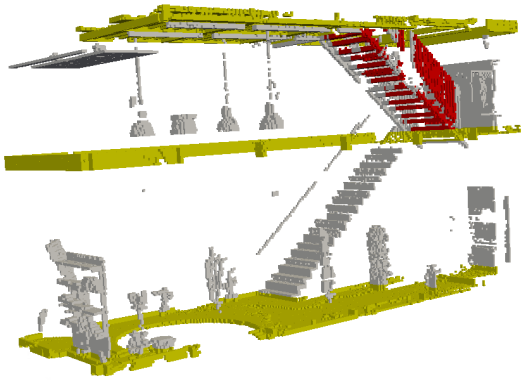


Figure 5.6: Octree leafs labelled as staircase (red), obstacle (grey) and floor (yellow) with Leica point cloud as source file. The point cloud represents the same scene as the left one in Figure 5.7

Nevertheless, using precise point clouds the approach has good results. Figure 5.6 shows that obstacles or furniture can be distinguished in the point cloud. The grey leafs represent lamps, lockers or cupboards.

Outcomes of wall detection

In emergency cases it can be of high importance whether the way is blocked by an obstacle or by a wall. Walls are usually immobile and made out of different materials than obstacles. Especially in fire situations this can be valuable information, considering the impermeability of walls for smoke. In addition, it is a lot more likely that an obstacle has moved or is movable to free a path which was blocked before, than the same is the case for walls. The Figures 5.1 and 5.8 show that the detection works in almost all cases, but still has room for improvement.

Relying only on 1D histograms has many down sides and is not recommended. On the one hand, it can give a first impression and is likely to work in empty buildings without staircases or furniture. On the other hand, the approach can cause false classifications when there are other features in the building or when the scene is large or irregular. The larger a building is, the bigger is the likelihood of small walls which are unique in their direction. These walls might get lost due to the 1D histogram's peak approach. Also,

the projection to the octree has disadvantages and only works if the scene is small. When there are stairs or obstacles in the same line as walls, the stairs or obstacles in this directions will also be classified as walls. The usage of 2D histograms of vertical leafs and the line detection with the hough transform is more promising to be scalable, especially referring to the building size. Moreover, the approach can be extended to non-perpendicular walls. Completely round walls might still not be identified using this kind of technique though. The features to be detected need to be straight to some extent at least to be identified as lines. Although, similar techniques could also be used to detect round shapes.

There are, however, many open questions, which could not be solved. First of all, glass windows are not represented in point clouds (this can be observed in Figure 5.13, where the windows are not represented in the point cloud), which might lead to a false path or impression of the architectural scene. But, this is a general disadvantage of laser point clouds. Secondly, pillars should be treated like walls as well. In the current approach they are likely to be labelled as obstacles. Regularity in the building can also help to distinguish them from furniture, as the pillars' locations are usually shared with walls or pillars across different storey levels.

5.2.3 Identification of stairs

As stairs come in various shapes and forms it seems to be almost impossible to find a unique algorithm that works with all of them. Even stairs which are architecturally identical can be represented differently in the same point cloud, like shown in Figure 4.4. Despite this, in the majority of the analysed cases, the identification of stairs works and the leafs in the octree can be labelled as such. Figure 5.7 shows the identified leafs in red colour in two different ZEB1 point clouds. Notwithstanding the good results with the point clouds of the ZEB1, Figure 5.6 shows that the approach does not always work and is strongly depending on the point cloud. The figure shows the same scene as Figure 5.7 on the left, but scanned with a stationary laser scanner. In the ground level the stairs are not found because they are oriented differently to the scanner's position. The filters do not give any positive response because the horizontal structure is missing in the point cloud. In the following the findings of the proposed methodology will be discussed and the limitations analysed.

Open challenges

After the implementation of the proof of concept it is visible that many laser scanners, and so does the ZEB 1, have difficulties in producing a good quality point cloud for staircases. Therefore, the methodology to find them must be robust to noise and artefacts. This is a big challenge, although it should be noted that one of the preconditions was to have a noise free point cloud. The implementation however, was tested also with point clouds which did not fulfil all of them. Furthermore, there are many different kinds of stairs, which makes it challenging to find an approach that works on all indoor point clouds at the same time. An example of this difficulty is shown in Figure 5.8, where the stairs are not labelled as such. In that case the filters do not give a positive response at the stair's location, because the point cloud in this area is too badly conditioned (see Figure 5.9) and noisy. The plane detection algorithm cannot find horizontal features at the stair's

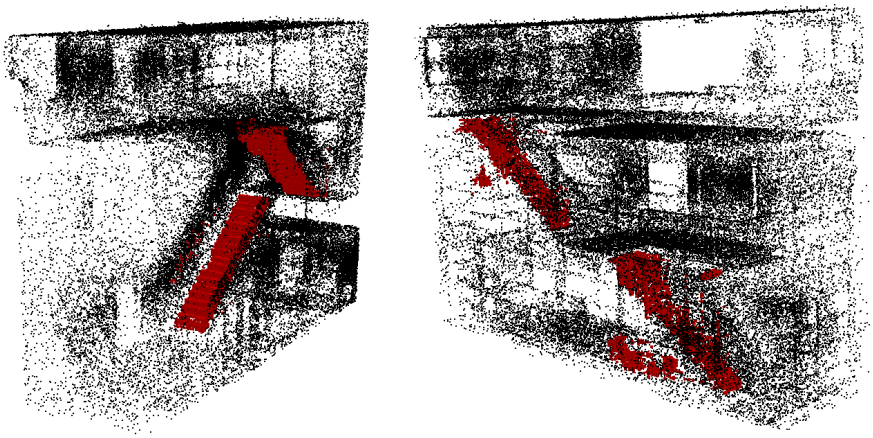


Figure 5.7: Octree leafs labelled as staircase (red) in point clouds of ZEB1 (black)

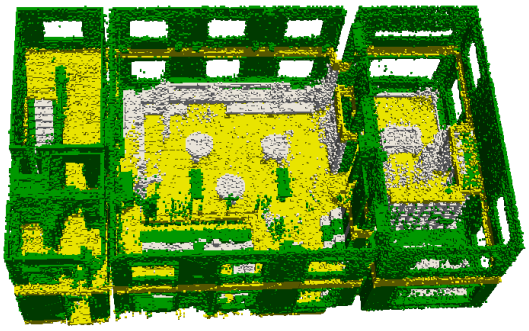


Figure 5.8: Octree leafs of ZEB1 point cloud of Bouwpub in a bigger extent and different orientation as in Figure 5.4, labelled as floor (yellow), wall (green) and obstacle (grey). The stairs in the upper left corner could not be identified

location. Besides the noise and inaccuracy, there are a lot of things, like furniture or beer crates underneath the stairs. These objects are becoming problematic for the projection to the 2D histograms and the resulting filter responses, too.

Besides the challenge of a large variation of stairs, also the methodology itself has room for improvement. Especially the part using filters to detect the typical change of plane directions in the point cloud did not provide the expected results. The vertical filter, whose implementation was unclear (like explained in Section 4.3.3), has only a negligible influence on the current result. Even when it was not applied, the stairs could still be found in the ZEB1 reference point cloud used for large parts of the implementation. The description of the filter in Bansal et al. [2010, 2011] does not go into much detail so that many decisions remain unclear. It is unknown how to deal with different directions of the stairs as well as the step width and height. The unknown direction can be overcome by applying the filter twice, once in each direction. However, there are also many cases where stairs do not have a vertical riser at all, like for example the stairs at the Faculty of Architecture in Delft, shown in Figure 4.5. The vertical filter would not give a positive response in such situations.

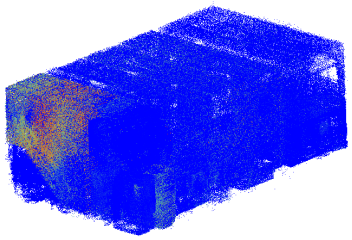


Figure 5.9: The Leica C10 point cloud of the Bouwpub, coloured according to the condition of the point cloud. Good condition (blue) → bad condition (red)

Outcomes of stair detection

In total, in five out of eight cases the stairs could be identified using the proposed methodology. The reason, why the stairs could not be found in all point clouds is because not all of them meet the preconditions set in Section 3.1. The methodology is limited to rectangular stairs, which are placed in parallel direction of the walls in Manhattan-World structured buildings. Furthermore, they should be completely represented in the point cloud and for the best result have a complete riser. In point clouds fulfilling these and the other conditions, the stairs can be found.

The stairs' regions contain small gaps and can include adjacent features. This lays in the nature of the filter responses which might lead to distortions of the regions. Such observations can also be made in the research of Bansal et al. [2010]. However, pathfinding in this thesis requires only a fuzzy estimation of the semantics. So the estimated stair location is enough. If the ultimate goal would be a vector model another methodology is recommended.

Clutter above and under stairs, for example bicycles on the right of Figure 5.7, are wrongly assigned to be stairs due to the 2D projection to the octree. Nevertheless, in the majority of the tested point clouds the stairs were found and the area is mostly correctly assigned. The method works on global point clouds, which means the viewing angle of the scanner does not play a role and neither does depth or colour. This is an advantage as it can be reused for many kinds of point clouds and applications. Yet, many methodologies to detect stairs in point clouds rely on such parameters as discussed in Section 2.1.2. The attributes of the octree allow quick neighbour finding and access to relevant points for plane fitting. The additional usage of linearity, slope and tilt provides a good estimate for stair identification. A positive response of the horizontal filter can help to distinguish stair features from ramps, which do not have horizontal surfaces.

Generally, the approach has promising aspects, but can be improved. A focus on horizontal planes in addition to a rise in height can give a good indication. The projection to 2D, however, can still lead to false results when there are obstacles or other features underneath the stairs. Therefore, the path of the scanner should be taken into consideration as well, as it has a typical gradient at the stair's location and can be a good indicator. The preconditions, however, would needed to be changed to make such data a necessity.

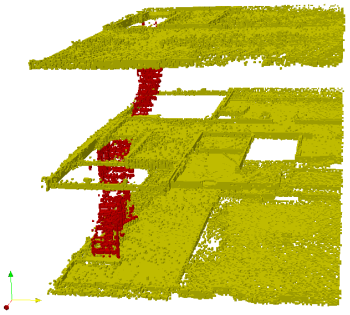


Figure 5.10: Walkable underground leafs where the floor has yellow octree leafs and the stairs are red. The point cloud represents the same staircase, but with a bigger extent as the right one in Figure 5.7

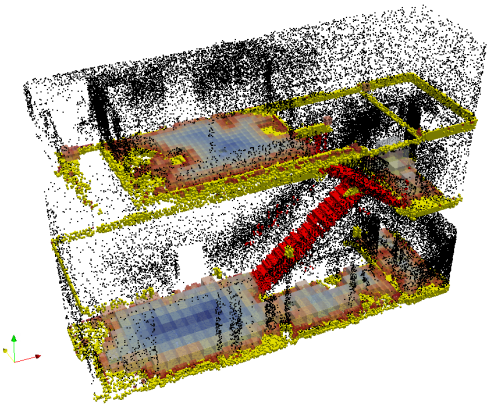


Figure 5.11: ZEB1 point cloud with octree showing underground which can be used for pathfinding (floor and stairs). On the floor is a clearance maps, showing the distance transform (the bluer, the more likely the object fits)

5.3 DERIVATION OF THE GRAPH

The semantic enrichment and subdivision of space aims to create a smarter graph to enable pathfinding for humans through a point cloud. A graph following the constraints of human movement can be derived, using the semantically enriched octree. To do so, the walkable surface needs to be found first. Walkable surface is made up of floor and stairs. Figure 5.10 shows the underground identified on a point cloud, acquired with the ZEB1 laser scanner, which can be used for walking. Furthermore, it has to be determined, that the human fits through the space as shown in Figure 3.19. Therefore, a clearance map, holding only nodes which are free up to the height of the person has to be created. The colours of the clearance map in Figure 5.11 show the distance transform on floor level to either walls or obstacles in each direction for all voxels on walkable space in a point cloud. Depending on the size of the object to be moved (in this case the size or width of a human) only leafs with a distance value smaller than half the object's diameter can be a part of the resulting graph.

Figure 5.12 shows how the route does follow the shortest path possible according to the size constrains of a human. Instead of taking a direct and straight way, the path pays attention to the clearance map and the distance

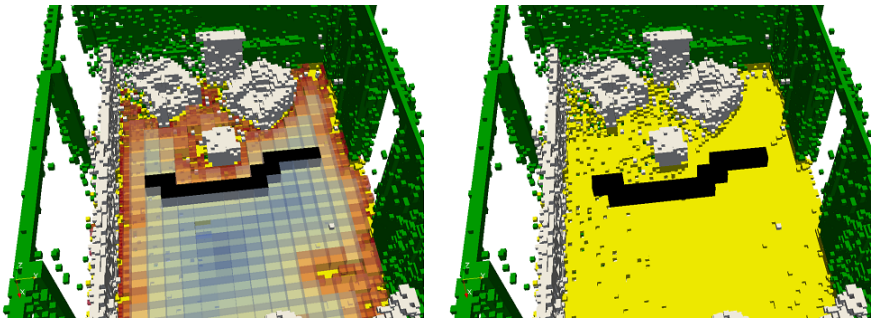


Figure 5.12: The path (black) takes a small detour around an obstacle (on the left including the clearance map)

transform to only use leafs which are far enough from the next obstacle or wall to be used by a human. This proves the concept of semantic enrichment necessary for a better and more realistic path for humans. The implementation for the graph at floor level works fine, but the graph-network at the stairs level and also the connection to different storeys has room for improvement. Also, to perform routing, the locational codes of the leafs the path should connect, have to be known.

5.4 ANALYSIS OF RESULTS

The workflow aimed to achieve a rough estimate for storey separation, the location of floors, walls, stairs and obstacles. For point clouds following the preconditions, this could be achieved. For routing and to optimise the pathfinding for humans a rough estimation of the semantic classification is enough scene understanding. Figure 5.13 shows the octree and photographs from the same viewing angle and visualises the success of the semantic enrichment, furthermore it provides a path through the scene. The workflow was also tested with point clouds which do not fulfil all requirements. This has shown that a big part of the created workflow does not only work for different types of buildings, but also for different kinds of laser scanners. Nevertheless, the result and its characteristics strongly depends on the quality of the scan. Furthermore, to what extent the scene meets the preconditions is of a large importance. Therefore, for the current framework these constraints have to be carefully set. Figure 5.1 shows how the results differentiate form each other for different kinds of laser scanners.

Also a shortest path can be derived which can connect different storeys (Figure 5.14). The path avoids obstacles and follows the logic of a human actor, who is only able to use floors and stairs for walking. Figure 5.15 shows another example of a shortest path through the fire brigade building which uses the stairs. The region of the stairs is, even though detected, not clean. This makes the implementation of the distance transform challenging.

The workflow to go from a point cloud to such navigable structure is, besides some small manual actions like the rotation of the point cloud or the choice of parameters, automated. The choice of parameters, for example the maximum wall width, cannot be automated because they can be very different depending on the building and can also not be derived without determining which parts of the buildings are the walls, first. Other parameters like the number of levels in the octree or the minimum peak height

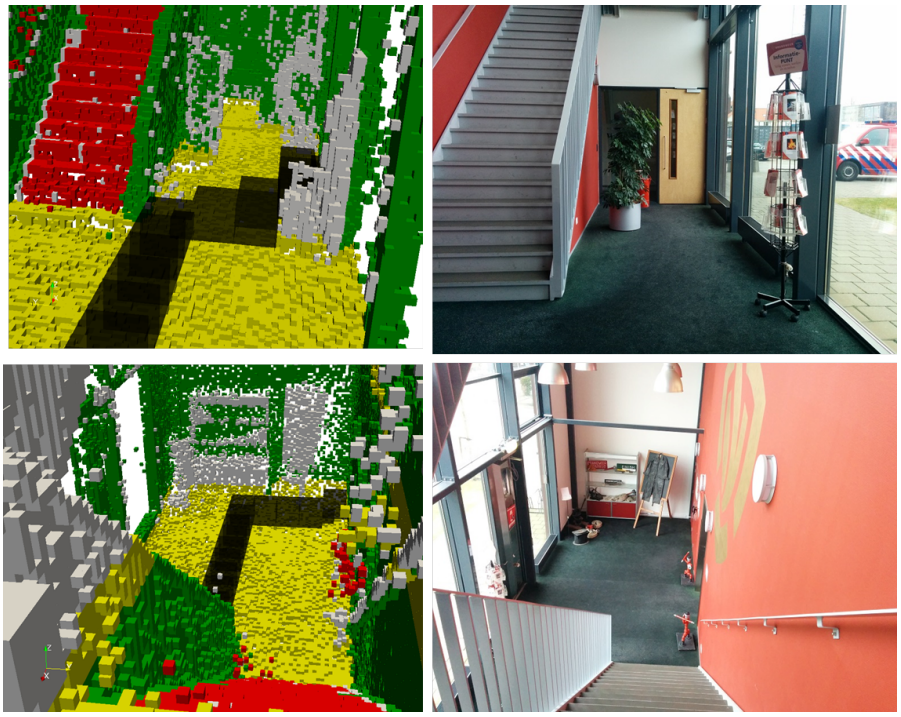


Figure 5.13: Left: Semantically enriched octree leafs labelled as floor (yellow), wall (green), staircase (red), obstacle (grey) and a random path through the scene (black); Right: Photo of original scene at fire brigade in Berkel en Rodenrijs

are set to a predefined value, which worked good in all point clouds used. This was further described in the Sections 4.3.1 and 4.2.3. Furthermore, it is necessary to include the path of the scanner to become less dependent on such parameters. Otherwise there can be a false floor detection and storey separation when the predefined values are kept.

Especially the choice to use 1D histograms for walls limits the workflow to Manhattan-worlds. Furthermore, the projection of the found semantics back to the octree can cause false classification. That is why it is important to also use the verification steps explained in Section 3.2.2. A filter methodology enabled stair detection without having the point of view of the scanner. To increase the quality of the results also the slope had to be introduced.

To enable pathfinding it is important to know where floors are located and storeys can be separated in the octree. The location of the stairs are relevant because they connect the different storeys. Furthermore, the distinction between walls and obstacles is valuable because it adds a lot of scene understanding and some obstacles are not permanent. Once all these objects are classified a graph network can be derived. The combination with the octree of empty space can be used to derive a pathfinding network following the constrains of human movement. Especially at this step the octree structure is beneficial due to its hierarchy.

All things considered, it can be concluded that the semantic enrichment works for many cases. The chosen methodologies proved to be working and the experience gathered during the implementation process and analyses of the results led to conclusions on how to improve or replace them. The derivation of the path is able to prove the concept.

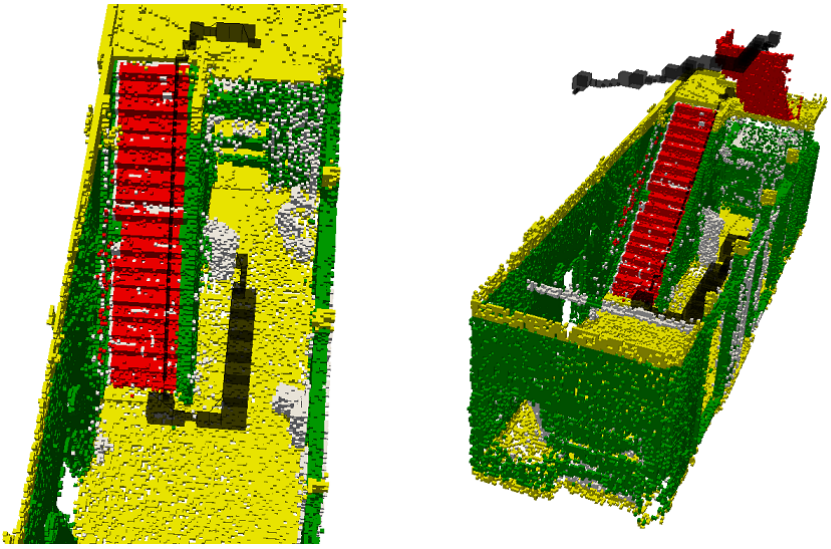


Figure 5.14: Path as octree leafs (black) across two (left) and three levels (right), for a better visualisation some walls, floor and obstacles are not shown

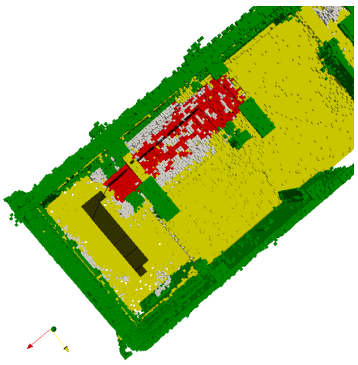


Figure 5.15: Path as octree leafs of another point cloud scene in the fire brigade building

6 CONCLUSION AND RECOMMENDATIONS

This thesis aimed to design a workflow that takes a 3D indoor point cloud and identifies objects, which are given semantic meaning, like the stairs and floors. Next, a graph can be derived which connects different storeys in the building. Due to the semantic enrichment, the graph is able to adapt to the constrains and size of a moving human actor. Before the conclusions can be drawn, the research questions defined in Section 1.3 need to be answered and the choice of methodologies will be discussed (Section 6.2). Following this, the future work and recommendations will be presented.

6.1 RESEARCH QUESTIONS

In this section the research questions posed in Chapter 1.3 are addressed. First the sub research questions will be answered, before finally the main research question can be answered as well.

- *Can floors and walls be distinguished in the octree structure of a 3D indoor point cloud containing stairs and furniture?*
  As shown in Section 5.2 floors and walls can be distinguished in the octree structure of a 3D point cloud, also when there are stairs and furniture in the scene. Nevertheless, the building should follow the preconditions set in Section 3.1, like the Manhattan-World assumption. The main challenges are irregularities like dropped ceilings or furniture. Irregular or unusual building structures can cause confusion for the detection of storeys and floors. If available, the path of the scanner can provide powerful additional information to differentiate between floors, dropped ceilings and tables. Even though they can all appear as a peak in the 1D histogram which is used in the proposed approach.

  Similar to the distinction between dropped ceilings and floors, the biggest challenge in identifying walls is the differentiation from obstacles and furniture like cupboards or lockers. The location of the 1D histogram peak can be an indicator. Additionally, line detection with hough transform on 2D histograms of vertical planes to detect and distinguish walls in the model can verify the classification. The combination of algorithms and methodologies proposed in Chapter 3 create a powerful set to detect the large majority of the walls. Nevertheless, finding the right parameter for the maximum wall width is challenging, especially as the width of walls can differ a lot within the same building. Along with wall shaped obstacles this is the main reason for false classifications.

  Altogether, walls and especially floors can be found in the octree of the 3D point cloud, but the building structure has to follow the preconditions. Like shown in Section 5.2 the results are better when the building follows a regular architectural structure.

- *What methodology can be used to detect stairs in the octree structure derived from a 3D indoor point cloud?*
  Stairs come in many different shapes and can differentiate within the same point cloud, even if they follow the same architectural design in reality. Point clouds of different kinds of laser scanners differ strongly in the region of stairs. Not having a viewing angle or depth information in the point cloud complicates matters and excludes many methodologies introduced in Section 2.1.2.

  Yet, the suggested methodology is still able to find the stairs' region in all tested cases which met the preconditions. Provided that floor and walls are identified, all not yet labelled octree leafs are taken into consideration. First of all, it is checked if there is either a vertical or horizontal plane found in the leafs and its neighbours. These leafs are taken into account for a 2D histograms showing the respective orientation. Secondly, matched filters are applied and the positive filter response is segmented using region growing. Thirdly, the regions are tested on slope, tilt and evenness, which has to meet the requirements typical for stairs to be able to tell whether a region represents stairs or not.

  The introduction of slope as indicator for stairs was necessary, as the suggested filter approach [Bansal et al., 2010, 2011] found not to be sufficient in this case. Notwithstanding the limitations of the methodology to only work for a limited types of stair structures, it does work where the point cloud and scene meets the preconditions.

- *What is the influence of the three different scanners on the result of the semantic enrichment?*
  There are many differences in point clouds of the same architectural scene which are acquired with a different kind of laser scanner. First of all, there is a distinction between stationary and mobile laser scanners. Throughout this research the point clouds of a stationary (Leica C10) and two mobile (ZEB1 and Project Tango Tablet) laser scanners were tested.

  For a stationary laser scanner, like the Leica C10, the point of view and line of sight can cause a big discrepancy of point representations for the same feature. The point of view of the scanner to the stairs can cause the point cloud representation to have different characteristics even though they are architecturally identical. Mobile laser scanners can give a more regular representation of the world. It is less likely that important attributes are hidden from the line of sight of the scanner, because it moves around. Also the density of the points can be more evenly spread if a smart route through the building is chosen. A point cloud of a stationary scanner typically has holes where the line of sight is blocked. This can even be the floor around the scanner's location and causes the histogram peaks to be less distinctive when not multiple scans of the same room were made. Such situations do not happen with mobile laser scanners. Furthermore, the ZEB1 can provide the valuable path of the scanner and not just a coordinate like a stationary one. In terms of accuracy and cleanliness, the stationary scanner gives much better results especially compared to the point cloud of the Project Tango tablet. The Tango's point cloud has an offset across different storeys which can even lead to a violation of the Manhattan-World assumption. Furthermore, the offset can make it

hard to use the assumption that wall directions are often shared across multiple storeys and to distinguish walls from obstacles.

Generally a high density of points is not necessary, because they are getting generalised in the same octree leaf. In fact, the majority of the algorithm was performed on the octree structure only (disregarding the actual points) and thinned point clouds were used to speed up calculation time. Even though, comparing results with different laser scanners provides interesting insights for mobile scanning of indoor scenes, future research about reconstruction should focus on one kind of scanner. Once this workflow is optimised it can be extended to other scanning methodologies.

- *Is the semantically enriched octree sufficient to derive a pathfinding model for humans?*
  The semantically enriched octree structure is able to provide a better scene understanding, which enables to derive a path for humans from point clouds. Therefore it is important, to differentiate between floors, stairs, walls and obstacles. Floors and stairs make up the walkable part, while walls and obstacles should be excluded from the graph. The distinction between obstacles and walls is not always necessary just to get a path for humans. But, it can provide valuable information to orientate yourself or whether an object is permanent or not. The derived graph network is constrained to follow the floor and the stairs and keeps a clearance to all obstacles or walls. Moreover, it considers also hanging obstacles, when they are not high enough to walk underneath. On top of that, the subdivision does not provide only one node per room, but is further subdivided in the whole area/ This allows the path to freely use all available space. On the one hand, if there is a large open space only, the octree structure assures, that the path does not get too complex. On the other hand, close to obstacles or at the location of stairs, the graph network and the derived shortest path is very precise.

  The implementation of the graph derivation above stairs needs to be improved to follow the same standards, like the distance transform, as the graph above floors. However, it works as a proof of concept and shows how this additional information can provide a realistic path for humans in multi-storey buildings. The path avoids obstacles and only uses walkable space.

- *Which semantics improvements are necessary to establish a link between indoor and outdoor networks?*
  According to van der Marel [2016] there is no obstacle in using real world coordinates inside buildings. However, there is a problem determining those coordinates as GPS does not work inside buildings. Traditional survey techniques using a tachometer work inside and so do mobile laser scanners equipped with IMU. Control points with known coordinates and bundle adjustment software are necessary to georeference the point clouds. It can be estimated, that it is sufficient to provide three control points, especially since the distances within the point clouds, as well as in real world coordinate systems like the RD New, are metric. However, it is always good to have more and evenly spread control points.

  Once the transformation parameters are known, the conversion can be performed for every coordinate in the point cloud. The workflow and

implementation in this work makes use of such a transformations and the point cloud get converted into a local reference system to increase performance [Broersen et al., 2016]. Theoretically the whole workflow proposed in Chapter 3 also works using real world coordinates only. The transformation is also similar, just the parameters would needed to be adjusted accordingly. Adding a georeference to a point cloud makes the data semantically richer.

In brief, a number of control points whose coordinates are known in both reference systems are necessary to establish a link between indoor and outdoor networks. However, the actual implementation is a matter of future work (Section 6.4).

In conclusion to the responses of the sub research questions the main research question can be answered as well:

"To what extent can an octree support semantic enrichment of point clouds for the purpose of multi-storey pathfinding?"

Point clouds are unstructured and besides geometry they do not contain any semantic information. An octree can help to not only give structure to the points, but also to the empty space. The ultimate goal of this thesis is to support multi-storey pathfinding and therefore it is necessary to identify useful features, like the empty space, storeys, floors, stairs walls and obstacles. Instead of directly classifying the points, the octree leafs are classified. For the pathfinding network it is not only necessary to determine whether there is empty space, but also what the surrounding features represent. The result is a path through the empty space that follows the constrains of human movement.

Using the octree enables the workflow to be less dependent on the laser scanner, as for the majority of calculations the octree leafs are used instead of the points. There is still a variation in quality of the results, which are dependent on the accuracy of the scan, rather than the density of the points. The floor and wall identification uses 1D histograms, which are built using the octree data structure only. The octree generalises the point cloud to leafs and adds a structure to the empty space. This does not only improve the performance, but also provides more distinguishable results. Furthermore, with an octree data structure, the path is less detailed when it is further away from obstacles or walls. In free and open space less information is necessary to navigate collision free than in proximity to obstacles. Therefore the resulting graph is more precise when necessary. If there is a lot of free space, the octree and thus the path do not go that much into detail.

However, for walls and stairs, relying purely on the octree data structure shows limitations, because for plane fitting the points are necessary. On the other hand, the octree facilitates fast access to the points and allows neighbour finding or addressing other points in the proximity. Also the stairs can be found which enables multi-storey pathfinding. The characteristics of the octree data structure, like quick neighbour finding, addressing, etc. proves to be beneficial for the implementation as well. Furthermore, the octree allows to find a good cutting height on where to apply the Poincaré duality.

Generally, the success of the implementation is depending on custom parameters. The depth of the octree is dependent on the size of the building and the smallest leaf. The size of the leaf after the last subdivision proved to give the best results for the tested point clouds when the length of the smallest leafs was around five centimetres. Furthermore, the building needs to be regular and to follow the preconditions set earlier. However, also point clouds not fulfilling all requirements, were used for testing. They indicated

challenges for data exceeding the preconditions, which should be solved in future research. For example non-Manhattan-World structures do not only have a negative influence on the octree as there are more smaller leafs, but also large parts of the workflow are not applicable to them. Due to the large variety of architectural structures, like different kinds of stairs, not all can be detected.

Altogether, the workflow results in a path through a semantically enriched indoor point cloud scene which is following navigation constraints of humans. The octree enables the path to be more detailed close to the points. Furthermore, the octree makes the approach less dependent on the point cloud attributes, adds speed and is used in almost the entire workflow. The graph network derived from the octree is not limited to a single storey. Additionally, the graph could even be connected to the outside to facilitate transition free pathfinding in different environments.

6.2 DISCUSSION

In this section the choice of methods used in this research will be discussed. A linear octree was chosen because of the experiences of Broersen et al. [2016]. Especially the hierarchical structure of empty space is beneficial in terms of for example storage space compared to other approaches like a complete voxelisation. The octree structure proved to be supportive in many steps of the workflow. In the part of the octree with empty space, where uniform leafs are as big as possible, the workflow makes use of the merging of uniform empty space. It does not only enable the resulting graph to be more detailed in proximity of obstacles, but also simplifies the implementation and adds speed. The non empty leafs have an advantage as it can be beneficial to work with them directly in many cases, instead of with the points.

Storeys and floors are identified with histogram peaks. The approach was based on Okorn et al. [2010], Oesau et al. [2014] or Khoshelham and Díaz-Vilariño [2014]. Such approach is not only widely used in literature, but also proved to work fast and reliable. A path of the scanner can verify the results, especially when there are for example hanging ceiling or many tables this is necessary.

The detection of walls was also based on this approach. This has the advantage that it was easy to implement, because only the histogram direction had to be changed, but also clutter like furniture could be excluded from being detected as walls. However, due to the projection of the semantic classification to the octree structure and the regularity constraints explained in Section 5.2 the approach can not be recommended (see also Figure 3.10). In this thesis a method of plane fitting, 2D histograms and line detection which was based on the methods of Okorn et al. [2010] and Oesau et al. [2014] was added. This approach is more error prone to clutter, however, the combination of both approaches provided the best results.

The approach for the detection of stairs is based on Bansal et al. [2010, 2011]. The method was chosen due to its implementation overlaps with the previous steps and the independency from the point of view of the scanner. However, it had to be extended to meet the requirements of this thesis, like it was explained earlier. For future research, it might be better to put more focus at the path of the scanner to find the stairs' locations. The derivation of

the graph works good for its purpose. Nevertheless, a full implementation of the graph derivation on the stairs is complex.

The usual approach to enable indoor pathfinding is to create complex vector models of the building. Such complexity is not necessary though, if the octree method of this work is used. The data can be structured, semantically enriched and a graph can be derived. For visual aspects, end user experience or manual corrections a vector model should still be the ultimate goal. The semantically enriched octree structure can be a good base for a semantically enriched vector model as well.

6.3 CONCLUSION

As shown above, using the octree and the workflow storeys and walls can be found in a 3D point cloud of an indoor scene. Furthermore, obstacles can be distinguished, and stairs can be identified in almost all cases. Also a graph can be derived that enables multi-storey pathfinding, following constraints for humans. On top of that, the network can potentially be connected to the environment.

Nevertheless, indoor environments differ strongly and it is a challenge to find a workflow for semantic enrichment that does not only work for point clouds acquired by different kinds of laser scanners, but also for a wide range of architectural scenes. Even the same scene can be represented completely different depending on the scanner used to acquire the data. To achieve the best results all preconditions for scenes and point clouds should be met. The most important requirements are a Manhattan-World assumption for walls, floors and stairs and that the point cloud include furniture, but should be free of noise. Combining different works helped to apply the workflow to more scenes accordingly. Furthermore, the methodology to identify the stairs by Bansal et al. [2010, 2011] was found to be not sufficient for the tested cases and had to be extended like described in Section 3.2.3.

For many parts of the workflow projecting the data to 2D simplifies the problem. On the other hand, information might get lost as not everything in the scene is aligned. Regular buildings do not have this problem. The proposed methodology and workflow add structure and semantics to 3D indoor point clouds. These semantics in combination with the octree enable multi-storey pathfinding following the restrictions for humans.

All things considered the fire brigade can use such a concept as a first step to provide additional information for people in an emergency situation. After the fire brigade arrives at the site, they are able to scan a large part of the building before other parts of it also catch fire. In many cases the fire brigade might even be prepared and have a scan from only a couple of months ago. These are still far more up to date than the majority of old floor and emergency plans, which were not updated for decades. From the point cloud they can now quickly derive a model which allows them to calculate the shortest path to any scanned location in the building, across multiple storeys. The path can be provided in a way, which shows the firemen whether a route is free to use also with large equipment. Having knowledge about the shortest route can be a decision making factor, especially in large and complex constructions. If there is a pre-installed localisation hardware available they are even able to navigate themselves through the building in real time.

6.4 RECOMMENDATIONS

A set of new research questions, insights and ideas arose throughout the work on this thesis and will be introduced in the following. Generally it is recommended for future research to put the focus on one scanner and one part of the workflow only and optimise the methodology. Once the results are stable and reusable the workflow can be extended to other kinds of scanners or scenes.

- One of the main recommendations is to *extend the current workflow to non-Manhattan-Worlds*. Fitting planes through the octree leafs and projecting them to 2D grid histograms is a promising approach. Oesau et al. [2014] show among others how the hough transform can be used to detect lines which represent walls in the building. In this thesis such approach was limited to a verification of identified walls, but should be extended further in future research. Such method is not limited to Manhattan-Worlds only. The majority of the buildings do not follow Manhattan-World structures and new constructions become more and more complex.

- Furthermore, also *the identification of stairs has to be extended to more cases*. The ZEB1 mobile mapping system includes the path of the scanner, which could offer valuable information for the detection of stairs as well. It is recommended to conduct a research, which has the focus solely on identification of stairs. Also, using high density point clouds, and not thinned ones as throughout this research can enable better plane fitting. Additionally, the octree itself, which has a distinctive accumulation and distribution of leafs at the stairs' regions can provide further information. As point clouds from different kinds of scanners represent stairs in an unlike manner, it is recommended to focus on on scanner only.

- Moreover, *the semantic enrichment could follow a more human understandable classification*. Rooms could get a label, so empty leafs enclosed by the same walls would have a common attribute. In a like manner, the storey count needs to be adapted. Landings between the stairs' flights are counted as a normal storey in the current workflow. This might lead to misunderstandings and false interpretation by the users. The semantic enrichment can also be extended to more features like doors. Doorways can only be used for pathfinding at this moment, if they were open during the scan. Also elevators were not identified, but they are not recommended for usage in emergency cases.

- *The graph needs to be improved and the distance transform on stairs implemented*. The connection between the floors is important to provide a fluent transition in between storeys. The implementation of the graph derivation has limitations at the moment. For example, depending on the orientation of the stairs, a separate algorithm is necessary and the distance transform is not available at stairs. This needs to be implemented in the future because the time of this research was too limited.

- Additionally *the network could be connected to outdoor worlds*, which has not been done as there were no control points set when the scans were made. The methodologies proposed in Section 3.4 can be used for this implementation.

- Lastly, *the results of this research could be combined with development of other works of the SIMs3D project*, like for example of Rodenberg [2016]. In the future the proposed model might not only be improved in terms of accuracy, but also made updatable in real time.

BIBLIOGRAPHY

3D Laser Mapping (2016). Geo Slam ZEB1. http://www.3dlasermapping.com/zeb1/. Accessed 16.02.16.

Adler, D. (1999). *Metric Handbook: Planning and Design Data*. Architectural Press.

Ahrens, J., Geveci, B., and Law, C. (2005). 36 - paraview: An end-user tool for large-data visualization. In Johnson, C. D. H. R., editor, *Visualization Handbook*, pages 717 – LXXII. Butterworth-Heinemann, Burlington.

Bansal, M., Matei, B., Southall, B., Eledath, J., and Sawhney, H. (2011). A LIDAR streaming architecture for mobile robotics with application to 3D structure characterization. In *2011 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1803–1810.

Bansal, M., Southall, B., Matei, B., Eledath, J., and Sawhney, H. (2010). LIDAR-based Door and Stair Detection from a Mobile Robot. In *SPIE, the International Society for Optical Engineering. Proceedings*. Society of Photo-Optical Instrumentation Engineers.

Becker, S., Peter, M., and Fritsch, D. (2015). Grammer-supported 3D indoor reconstruction from point clouds for "as-built" BIM. *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, 2(3):17.

Becker, T., Nagel, C., and Kolbe, T. (2009). Supporting contexts for indoor navigation using a multilayered space model. In *Tenth International Conference on Mobile Data Management: Systems, Services and Middleware, 2009. MDM '09.*, pages 680–685.

Bergeon, Y., Hadda, I., Krivanek, V., Motsch, J., and Stefek, A. (2015). Low cost 3d mapping for indoor navigation. In *2015 International Conference on Military Technologies (ICMT)*, pages 1–5.

Boguslawski, P., Gold, C. M., and Ledoux, H. (2011). Modelling and analysing 3D buildings with a primal/dual data structure. *ISPRS Journal of Photogrammetry and Remote Sensing*, 66(2):188–197.

Borgefors, G. (1986). Distance transformations in digital images. *Computer vision, graphics, and image processing*, 34(3):344–371.

Brock, O., Trinkle, J., and Ramos, F. (2009). Planning long dynamically-feasible maneuvers for autonomous vehicles. In *Robotics:Science and Systems IV*, pages 214–221. MIT Press.

Broersen, T., Fichtner, F. W., Heeres, E. J., de Liefde, I., Rodenberg, O. B. P. M., Verbree, E., and Voûte, R. (2016). Project Pointless. Identifying, visualising and pathfinding through empty space in interior point clouds using an octree approach. In *AGILE 2016; 19th AGILE Conference on Geographic Information Science, 14-17 June, 2016; Authors version*.

Chiu, W. (2013). Python dijkstra algorithm. http://stackoverflow.com/a/16117378. Accessed 26.04.2016.

Delmerico, J., Baran, D., David, P., Ryde, J., and Corso, J. (2013). Ascending stairway modeling from dense depth imagery for traversability analysis. In *2013 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2283–2290.

Diakité, A. A., Damiand, G., and Gesquière, G. (2014). Automatic Semantic Labelling of 3D Buildings Based on Geometric and Topological Information. In *9th International 3DGeoInfo Conference (3DGeoInfo). Proceedings.*, 3DGeoInfo conference proceedings series, pages 49–63, Dubai, United Arab Emirates. Karlsruhe Institute of Technology.

Duarte, M. (2015). Notes on scientific computing for biomechanics and motor control. https://github.com/demotu/BMC. Accessed 10.05.2016.

Eilering, A., Yap, V., Johnson, J., and Hauser, K. (2014). Identifying support surfaces of climbable structures from 3D point clouds. In *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pages 6226–6231.

Galamhos, C., Matas, J., and Kittler, J. (1999). Progressive probabilistic hough transform for line detection. In *Computer Vision and Pattern Recognition, 1999. IEEE Computer Society Conference on.*, volume 1, page 560 Vol. 1.

Girardeau-Montaut, D. (2016). Cloudcompare (version 2.6.2) [gpl software]. http://www.cloudcompare.org/. Accessed 23.02.2016.

Google ATAP (2015). Project tango. https://www.google.com/atap/project-tango/. Accessed 09.12.2015.

Halsted, G. (2014). Connexient Blog: Google's Project Tango - and Why a 3D Model is Not an Indoor Map. http://www.connexient.com/blog/entry/google-s-project-tango-and-why-a-3d-model-is-not-an-indoor-map. Accessed 18.12.15.

Herman, M. (1986). Fast, three-dimensional, collision-free motion planning. In *1986 IEEE International Conference on Robotics and Automation. Proceedings.*, volume 3, pages 1056–1063.

Hornung, A., Phillips, M., Jones, E. G., Bennewitz, M., Likhachev, M., and Chitta, S. (2012). Navigation in three-dimensional cluttered environments for mobile manipulation. In *IEEE International Conference on Robotics and Automation (ICRA). Proceedings.*, St. Paul, MN, USA.

Hornung, A., Wurm, K. M., Bennewitz, M., Stachniss, C., and Burgard, W. (2013). OctoMap: An efficient probabilistic 3D mapping framework based on octrees. *Autonomous Robots*, 34:189–206. Software available at http://octomap.github.com.

Hunter, J. (2007). Matplotlib: A 2d graphics environment. *Computing in Science Engineering*, 9(3):90–95.

Jamali, A., Rahman, A. A., Boguslawski, P., Kumar, P., and Gold, C. M. (2015). An automated 3D modeling of topological indoor navigation network. *GeoJournal*, pages 1–14.

Jessup, J., Givigi, S., and Beaulieu, A. (2014). Robust and efficient multi-robot 3D mapping with octree based occupancy grids. In *2014 IEEE International Conference on Systems, Man and Cybernetics (SMC)*, pages 3996–4001.

Jones, E., Oliphant, T., Peterson, P., et al. (2001). SciPy: Open source scientific tools for Python. http://www.scipy.org/. Accessed 23.02.2016.

Karimi, H. (2015). *Indoor Wayfinding and Navigation*. CRC Press, New York City.

Khoshelham, K. and Díaz-Vilariño, L. (2014). 3D Modelling of Interior Spaces: Learning the Language of Indoor Architecture. *ISPRS - International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, pages 321–326.

Kitamura, Y., Tanaka, T., Kishino, F., and Yachida, M. (1995). 3D path planning in a dynamic environment using an octree and an artificial potential field. In *1995 IEEE/RSJ International Conference on Intelligent Robots and Systems 95. 'Human Robot Interaction and Cooperative Robots', Proceedings.*, volume 2, pages 474–481 vol.2.

Klepeis, N. E., Nelson, W. C., Ott, W. R., Robinson, J. P., Tsang, A. M., Switzer, P., Behar, J. V., Hern, S. C., and Engelmann, W. H. (2001). The national human activity pattern survey (nhaps): A resource for assessing exposure to environmental pollutants. *Journal of Exposure Analysis and Environmental Epidemiology*, 11(3):231–252.

Krūminaitė, M. and Zlatanova, S. (2014). Indoor space subdivision for indoor navigation. In *Sixth ACM SIGSPATIAL International Workshop on Indoor Spatial Awareness. Proceedings.*, ISA '14, pages 25–31, New York, NY, USA. ACM.

Lee, J., Li, K.-J., Zlatanova, S., Kolbe, T. H., Nagel, C., and Becker, T. (2014). OGC® IndoorGML. http://docs.opengeospatial.org/is/14-005r3/14-005r3.html. Accessed 20.06.2016.

Lee, J. and Zlatanova, S. (2008). A 3d data model and topological analyses for emergency response in urban areas. *Geospatial information technology for emergency response*, 143:C168.

Leica Geosystems (2016). Leica ScanStation C10. http://hds.leica-geosystems.com/en/Leica-ScanStation-C10_79411.htm. Accessed 16.02.16.

Liu, L. and Zlatanova, S. (2012). A semantic data model for indoor navigation. In *Fourth ACM SIGSPATIAL International Workshop on Indoor Spatial Awareness. Proceedings*, ISA '12, pages 1–8, New York, NY, USA. ACM.

Liu, L. and Zlatanova, S. (2013a). Generating navigation models from existing building data. In *Acquisition and Modelling of Indoor and Enclosed Environments 2013, Cape Town, South Africa, 11-13 December 2013, ISPRS Archives Volume XL-4/W4, 2013*. ISPRS.

Liu, L. and Zlatanova, S. (2013b). A two-level path-finding strategy for indoor navigation. In *Intelligent Systems for Crisis Management*, pages 31–42. Springer.

Minister van Binnenlandse Zaken en Koninkrijksrelaties (2015). (bouwbesluit 2012), stb. 2011, 416. http://vrom.bouwbesluit.com/Inhoud/docs/wet/bb2012/hfd2/afd2-5. Accessed 26.02.2016.

Narasimhan, S., Mundani, R.-P., and Bungartz, H.-J. (2006). An octree-and a graph-based approach to support location aware navigation services. In *PSC*, pages 24–30.

Nikoohemat, S. (2016). Indoor 3d model reconstruction to support disaster management in large buildings. PhD proposal, SIMs3D (Smart Indoor Models in 3D).

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

Oesau, S., Lafarge, F., and Alliez, P. (2014). Indoor scene reconstruction using feature sensitive primitive extraction and graph-cut. *ISPRS Journal of Photogrammetry and Remote Sensing*, 90:68–82.

Okorn, B., Xiong, X., Akinci, B., and Huber, D. (2010). Toward automated modeling of floor plans. In *Symposium on 3D Data Processing, Visualization and Transmission. Proceedings.*, volume 2.

Olufs, S. and Vincze, M. (2011). Towards efficient semantic real time mapping of man-made environments using microsoft's kinect. In *2011 IEEE International Conference on Robotics and Biomimetics (ROBIO)*, pages 130–137.

Oßwald, S., Gutmann, J.-S., Hornung, A., and Bennewitz, M. (2011). From 3D point clouds to climbing stairs: A comparison of plane segmentation approaches for humanoids. In *2011 11th IEEE-RAS International Conference on Humanoid Robots (Humanoids)*, pages 93–98. IEEE.

Payeur, P. (2006). A computational technique for free space localization in 3-d multiresolution probabilistic environment models. *IEEE Transactions on Instrumentation and Measurement*, 55(5):1734–1746.

Quuppa Oy (2015). Technology. http://quuppa.com/technology/. Accessed 10.12.15.

Rodenberg, O. (2016). The relation between the computational effort and path length of A* pathfinding in an octree representation of an indoor point cloud. Master's thesis, TU Delft.

Rusu, R., Marton, Z., Blodow, N., Holzbach, A., and Beetz, M. (2009a). Model-based and learned semantic object labeling in 3D point cloud maps of kitchen environments. In *IEEE/RSJ International Conference on Intelligent Robots and Systems, 2009. IROS 2009.*, pages 3601–3608.

Rusu, R. B., Blodow, N., and Beetz, M. (2009b). Fast Point Feature Histograms (FPFH) for 3D registration. In *IEEE International Conference on Robotics and Automation, 2009. ICRA '09.*, pages 3212–3217.

Samet, H. (1982). Distance transform for images represented by quadtrees. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 4(3):298–303.

Samet, H. (1989). Neighbor finding in images represented by octrees. *Computer Vision, Graphics, and Image Processing*, 46(3):367–386.

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

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

Shen, S., Michael, N., and Kumar, V. (2011). Autonomous multi-floor indoor navigation with a computationally constrained mav. In *2011 IEEE International Conference on Robotics and Automation (ICRA)*, pages 20–25.

SIMs3D (Smart Indoor Models in 3D) (2015). Smart 3D indoor models to support crisis management in large public buildings. http://www.sims3d.net/. Accessed 09.12.15.

Sinha, A., Papadakis, P., and Elara, M. (2014). A staircase detection method for 3D point clouds. In *2014 13th International Conference on Control Automation Robotics Vision (ICARCV)*, pages 652–656.

Slavic, J. (2015). Py-tools. https://github.com/jankoslavic/py-tools. Accessed 10.05.2016.

Tamas, L. and Goron, L. C. (2012). 3D map building with mobile robots. In *2012 20th Mediterranean Conference on Control Automation (MED)*, pages 134–139.

Tournade, Y. (2015). Overview of the peaks dectection algorithms available in python. https://github.com/MonsieurV/py-findpeaks. Accessed 25.02.2016.

Tseng, Y.-H. and Wang, M. (2005). Automatic plane extraction from LIDAR data based on octree splitting and merging segmentation. In *2005 IEEE International Geoscience and Remote Sensing Symposium, 2005. IGARSS '05. Proceedings.*, volume 5, pages 3281–3284.

Turner, E., Cheng, P., and Zakhor, A. (2015). Fast, Automated, Scalable Generation of Textured 3D Models of Indoor Environments. *IEEE Journal of Selected Topics in Signal Processing*, 9(3):409–421.

Turner, E. and Zakhor, A. (2013). Watertight planar surface meshing of indoor point-clouds with voxel carving. In *2013 International Conference on 3D Vision - 3DV 2013*, pages 41–48.

Turner, E. L. (2015). 3D Modeling of Interior Building Environments and Objects from Noisy Sensor Suites. Technical report, DTIC Document.

van der Ham, M. (2015). Real Time Localization of Assets in Hospitals using Quuppa Indoor Positioning Technology. Master's thesis, TU Delft.

van der Marel, H. (2016). Personal communication: Real world coordinates.

van der Walt, S., Colbert, S., and Varoquaux, G. (2011). The numpy array: A structure for efficient numerical computation. *Computing in Science Engineering*, 13(2):22–30.

van der Walt, S., Schönberger, J. L., Nunez-Iglesias, J., Boulogne, F., Warner, J. D., Yager, N., Gouillart, E., Yu, T., and the scikit-image contributors (2014). scikit-image: image processing in Python. *PeerJ*, 2:e453.

Vandapel, N., Kuffner, J., and Amidi, O. (2005). Planning 3-d path networks in unstructured environments. In *2005 IEEE International Conference on Robotics and Automation, 2005. ICRA 2005. Proceedings.*, pages 4624–4629. IEEE.

Varrazzo, D. (2015). Psycopg2: Postgresql adapter for the python programming language. http://initd.org/psycopg/. Accessed 2016-02-23.

Vo, A.-V., Truong-Hong, L., and Laefer, D. (2015). Aerial laser scanning and imagery data fusion for road detection in city scale. In *2015 IEEE International Geoscience and Remote Sensing Symposium (IGARSS)*, pages 4177–4180.

Vosselman, G., Gorte, B. G., Sithole, G., and Rabbani, T. (2004). Recognising structure in laser scanner point clouds. *International archives of photogrammetry, remote sensing and spatial information sciences*, 46(8):33–38.

Vörös, J. (2000). A strategy for repetitive neighbor finding in octree representations. *Image and Vision Computing*, 18(14):1085 – 1091.

Wagner, D., Kalischewski, K., Velten, J., and Kummert, A. (2015). Detection of ascending and descending stairways by surface normal vectors. In *Multidimensional (nD) Systems (nDS), 2015 IEEE 9th International Workshop on*, pages 1–5. IEEE.

Wang, F., Wang, K., Lai, S., Phang, S. K., Chen, B., and Lee, T. (2014). An efficient uav navigation solution for confined but partially known indoor environments. In *11th IEEE International Conference on Control Automation (ICCA)*, pages 1351–1356.

Wang, M. and Tseng, Y.-H. (2004). Lidar data segmentation and classification based on octree structure. *parameters*, 1:5.

Wang, M. and Tseng, Y.-H. (2011). Incremental segmentation of lidar point clouds with an octree-structured voxel space. *The Photogrammetric Record*, 26(133):32–57.

Wang, M. and Tseng, Y.-H. (2014). Extraction of surface features from lidar point clouds using incremental segmentation strategy. *Journal of Photogrammetry and Remote Sensing*, 19(1).

Wikimedia Commons (2012). Typical house floor plan [gpl]. https://commons.wikimedia.org/w/index.php?curid=18377401. Accessed 07.06.2016.

Wu, L. and Hori, Y. (2006). Real-time collision-free path planning for robot manipulator based on octree model. In *9th IEEE International Workshop on Advanced Motion Control, 2006*, pages 284–288.

Xiao, J. and Furukawa, Y. (2014). Reconstructing the world's museums. *International Journal of Computer Vision*, 110(3):243–258.

Xiong, X., Adan, A., Akinci, B., and Huber, D. (2013). Automatic creation of semantically rich 3d building models from laser scanner data. *Automation in Construction*, 31:325–337.

Yang, L. and Worboys, M. (2015). Generation of navigation graphs for indoor space. *International Journal of Geographical Information Science*, 26:1–20.

Yogeswaran, A. and Payeur, P. (2009). Features extraction from point clouds for automated detection of deformations on automotive body parts. In *IEEE International Workshop on Robotic and Sensors Environments, 2009. ROSE 2009.*, pages 122–127.

Yuan, W. and Schneider, M. (2010). Supporting 3d route planning in indoor space based on the lego representation. In *2Nd ACM SIGSPATIAL International Workshop on Indoor Spatial Awareness. Proceedings.*, ISA '10, pages 16–23, New York, NY, USA. ACM.

Yuan, W. and Schneider, M. (2011). 3d indoor route planning for arbitrary-shape objects. In *Database Systems for Adanced Applications*, pages 120–131. Springer.

Zlatanova, S., Liu, L., Sithole, G., Zhao, J., and Mortari, F. (2014). Space subdivision for indoor applications. Technical report, Delft University of Technology, OTB Research Institute for the Built Environment.

COLOPHON

This document was typeset using LaTeX. The document layout was generated using the `arsclassica` package by Lorenzo Pantieri, which is an adaption of the original `classicthesis` package from André Miede.