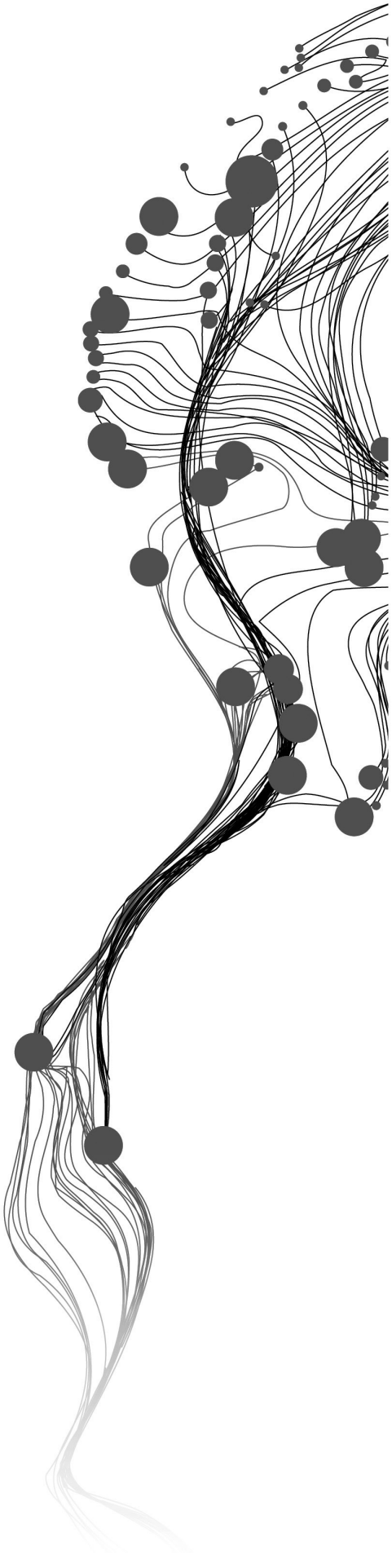
# SEMANTIC ENRICHMENT OF INDOOR MOBILE LASER SCANNER POINT CLOUDS AND TRAJECTORIES

AHMED MOSSAD IBRAHIM ELSEICY
February, 2018

SUPERVISORS:
Dr. ing. M.S. Peter
Dr. ir. S.J. Oude Elberink
S. Nikoohemat MSc (Advisor)



# SEMANTIC ENRICHMENT OF INDOOR MOBILE LASER SCANNER POINT CLOUDS AND TRAJECTORIES

AHMED MOSSAD IBRAHIM ELSEICY
Enschede, The Netherlands, February, 2018

Thesis submitted to the Faculty of Geo-information Science and Earth Observation of the University of Twente in partial fulfilment of the requirements for the degree of Master of Science in Geo-information Science and Earth Observation.
Specialization: Geoinformatics

SUPERVISORS:

Dr. ing. M.S. Peter

Dr. ir. S.J. Oude Elberink

S. Nikoohemat MSc (Advisor)

THESIS ASSESSMENT BOARD:

Prof. dr. ir. M.G. Vosselman (chair)

Prof. dr. ing. M. Gerke; Institute of Geodesy und Photogrammetry, Technische Universität Braunschweig

# ABSTRACT

State-of-the-art indoor mobile laser scanners allow the user to map challenging environments such as multi-story buildings and staircases while continuously walking through the environment. The trajectory of the laser scanner gives insights about indoor spaces and topological relations between them. Thus, analyzing the point cloud and the trajectory can automate the process of labeling them with information about the spaces such as floors, staircases, and rooms. In this research, the point cloud is analyzed with the help of the scanner trajectory to automate the labeling process. Analyzing the scanner trajectory as a standalone dataset is used to identify the staircases and to separate the floors. Also, it is used to infer the indoor spaces by analyzing the operator behavior during the scanning process. By processing the point cloud with the trajectory, the doors which are traversed by the trajectory are identified, and the spaces are labeled by transferring the labels from the annotated trajectory to the point cloud. The labels include floors, staircases, doorways, and rooms.

**Keywords**

*Indoor, Mobile laser scanning, Semantics, trajectory, Point cloud, Room segmentation*

# ACKNOWLEDGEMENTS

Writing this thesis has been a fun and rewarding experience. I would like to thank ITC for giving me the opportunity to be part of The ITC Excellence Scholarship Programme. It motivates me to keep a good academic record.

I would like to express my sincere gratitude to my first supervisor Michael for his continuous motivation and support. His passion towards indoor mapping persuaded me to go further in this research. I also gratefully acknowledge my second supervisor Sander for his critical feedback which is always to the point. Furthermore, I would like to thank my advisor Shayan for his support, critical feedback, and fruitful discussions.

Apart from academic life, I would like to thank Wan, Simon and all Run for Fun mates for acting as a renewable source of positive energy. Also, I thank all my friends and colleagues for the wonderful 18 months!

Last but not the least, I take this opportunity to thank my family for their love and unlimited support.

# TABLE OF CONTENTS

# LIST OF FIGURES

# LIST OF TABLES

# 1. Introduction

## 1.1 MOTIVATION AND PROBLEM STATEMENT

During the past few years, the demand for indoor modeling has increased. Up-to-date interior models are involved in many domains such as robotics, remote tourism, building information modeling (BIM), indoor mapping, gaming, and disaster management (Sirmacek et al., 2016).

Laser scanning sensors have many advantages and disadvantages which depend on the technology and sensor design as well as the purpose of use. For instance, the terrestrial laser scanner (TLS) can produce high-quality point clouds. The scanning system needs to be moved to multiple positions to minimize the occlusions, as it captures the data on its line of sight. Later, the tie points and the station data need to be post-processed to achieve optimal results. This approach is costly and time-consuming especially in complex environments with many occlusions (Jung et al., 2015; Thomson et al., 2013). On the other hand, state-of-the-art mobile 3D mapping systems are lightweight and portable enough that human can carry them (Bosse et al., 2012; Lauterbach et al., 2015; Lehtola et al., 2017; Nüchter et al., 2015; Wen et al., 2016; Chen et al., 2016). These mobile devices allow the user to map challenging environments such as multi-story buildings, tunnels, and caves while continuously walking through the environment (Moghadam et al., 2016; Zlot et al., 2015).

The indoor mobile laser scanners usually provide 3D point cloud beside the trajectory at which data were collected (Toschi et al., 2015). The point cloud represents different elements in the environment with accurate geometrical information. Although the visualization of 3D point cloud data enables humans to summarize the scene, further processing needs to be performed to extract information. Manual data processing is done by experienced users to assign semantic information. While this process makes the data meaningful and easier to interpret, this manual process is labor intensive and time-consuming for complex buildings (Moghadam et al., 2016). Therefore, there is a body of research which focuses on automating the information extraction process from the point clouds to overcome the problems of manual processing. Many approaches solved this problem by processing the data of the terrestrial laser scanners or RGB-Depth data either by assigning semantic information to the raw point cloud (Koppula et al., 2011; Lai et al., 2012; Moghadam et al., 2016; Armeni et al., 2016) or reconstructing the indoor scenes (Babacan et al., 2017; Mura et al., 2014; Ochmann et al., 2015).

Recently, system trajectory started to get more attention in joint analysis with the mobile laser scanning data either for indoor reconstruction (Díaz-Vilariño et al., 2017; Nikoohemat et al., 2017; Xie & Wang, 2017) or identifying the navigable spaces (Staats et al., 2017; Fichtner et al., 2018). On the other hand, human trajectory as a standalone dataset can contain a wealth of information. It has been used to get information about the indoor spaces such as constructing floor plans (Grzonka et al., 2012; Alzantot & Youssef, 2012) or enriching navigational maps (Guo et al., 2017; Prentow et al., 2015).

In this research, the information extraction from the point cloud is under the semantic enrichment category. This research aims to enrich the raw point cloud and the trajectory with semantic information about the indoor spaces such as floors, rooms, doors, and staircases. As will be shown, this can be done by analyzing the point cloud dataset and its corresponding trajectory. The tra-

jectory data can provide information for labeling different components in the point cloud. Movement patterns can be recognized by analyzing the system trajectory and subdividing the trajectory into sub-trajectories. Interesting patterns could be identified such as climbing stairs or identifying the turning points and directions along the trajectory (Grzonka et al., 2012; Prentow et al., 2015). This is utilized to label different parts of interior components such as rooms, hallways, and staircases. Also, it is used to detect secondary elements like doors.

## 1.2 RESEARCH IDENTIFICATION

The research will focus on the analysis of trajectory data in conjunction with the full point cloud to extract information about the indoor spaces. First, the trajectory of the mobile laser scanner is analyzed independently to understand its characteristics and to check how it can be used to extract semantic information about the indoor spaces. Then, it is used as a secondary dataset to enrich the point cloud and the trajectory with the semantic information by utilizing the key findings from the first step. The process include labeling both datasets with information such as floors, staircases, doorways, and rooms.

### 1.2.1 Research objectives

The main objective of this study is to apply semantic labeling of indoor point cloud data based on the shape of the system trajectory in conjunction with the point cloud dataset.

The specific objectives are:

1. To annotate the trajectory dataset with semantic information about the indoor spaces by analyzing the trajectory data only.
2. To classify the point cloud based on semantic information from both trajectory and point cloud.

### 1.2.2 Research questions

The research questions were reformulated during the implementation phase as part of the original questions focused on minor steps.

**Specific objective 1**

- What is the optimal method for trajectory simplification?
- What are the parameters that describe the critical points?
- What are the classes that could be identified from the trajectory data only?
- How can these classes be described based on the semantic rules?

**Specific objective 2**

- How can the point cloud be prepared for joint analysis process?
- How can the spaces be partitioned?
- How to detect the doors with the help of the trajectory?
- How to assign the trajectory classes to the corresponding point cloud?
- What are the methods for refining the labeling results?

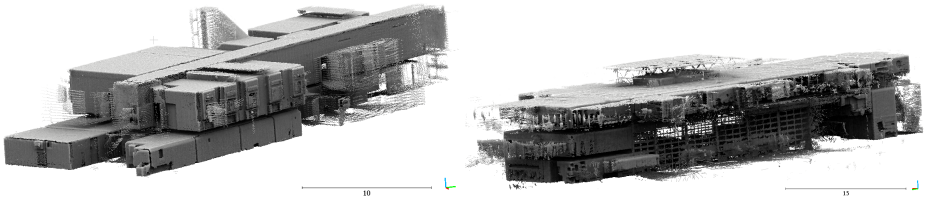
#### 1.2.3 Innovation aimed at

The trajectory of the indoor mobile laser scanners is not fully utilized in the literature. It was used to refine the data (Turner & Zakhor, 2015), or to classify single location (Mozos, 2010) of a mobile robot with range sensor. Recently[1], more methods started to make use of this dataset for indoor reconstruction (Díaz-Vilariño et al., 2017; Nikoohemat et al., 2017; Staats et al., 2017; Xie & Wang, 2017) and extracting navigable spaces (Staats et al., 2017; Fichtner et al., 2018). In this research, the classification of indoor point clouds into semantic spaces is introduced. Moreover, a detailed study of the trajectory datasets is applied to make use of the trajectory analysis techniques for the point cloud processing. The aim of this research is to make use of trajectory data as a primary input to classify the point cloud, as the trajectory is considered a valuable source of information about scanning system movement and the operator behavior. Joint analysis between point cloud and trajectory is performed to label both datasets with semantic information such as rooms, doors, and staircases.

### 1.3 PROJECT SETUP

#### 1.3.1 Datasets

Two datasets were used for the experiments as shown in Figure 1.1. First dataset (TUB2), is part of the ISPRS benchmark for indoor modeling (Khoshelham, Vilariño, Peter, Kang, & Acharya, 2017). It is a two-story building which consists of a basement and a ground floor connected by a staircase. It was acquired in a building in the Technische Universität Braunschweig, Germany, using ZEB-REVO sensor. The point cloud and the corresponding system trajectory were provided, and the level of clutter is low.
The second one was captured in Firebrigade office of Lansingerland, The Netherlands, and was introduced in related research by Sirmacek et al. (2016). It includes three-story building and it was captured using ZEB1 sensor. Both point cloud and the corresponding system trajectory were provided. The level of clutter is high, and it contains glass walls which make it a challenging environment for data acquisition with laser scanners and point cloud analysis.
We will refer to the first dataset as ZEB-REVO dataset and the second one as ZEB1 dataset through the analysis steps.



(a) (b)

Figure 1.1: Overview of the testing datasets. (a) Technische Universität Braunschweig building dataset (ZEB-REVO). (b) Fire bridge dataset (ZEB1).

#### 1.3.2 Project workflow

Figure 1.2 shows the process workflow. This research consists of two main blocks. The first one is concerned with analyzing the trajectory dataset to achieve the first objective. The trajec-

[1]The methods were published while working on the thesis.

tory dataset is processed based on the trajectory analysis techniques from the literature. The implemented methods aim to annotate the trajectory with semantic labels about the indoor spaces and to be reusable in the joint analysis process. In the second block, the raw point cloud dataset is analyzed in conjunction with the segmented trajectory from the first step. It started with separating each floor in the point cloud dataset based on the segmented trajectory. Then, for each floor, a door detection method is implemented to identify the transitions between the different spaces. After that, the trajectory dataset is annotated based on the analysis of the point cloud. Finally, the point cloud data is labeled with the semantic classes which represent the floors, the staircases, and the rooms. The accuracy assessment is done by visual interpretation on a digitized floor plan and explained during the discussion of the results.

### 1.3.3 Thesis structure

This thesis is organized into five chapters. The first chapter gives a brief introduction about the motivation for this research, research questions, and the project setup. Then, the theoretical background and the related work are introduced in the second chapter. In the third chapter, the full trajectory analysis process is discussed with its results, and the joint analysis process between the point cloud and the trajectory is discussed in the fourth chapter with the same structure of the third one. Finally, the conclusions and the recommendations are presented in the fifth chapter.

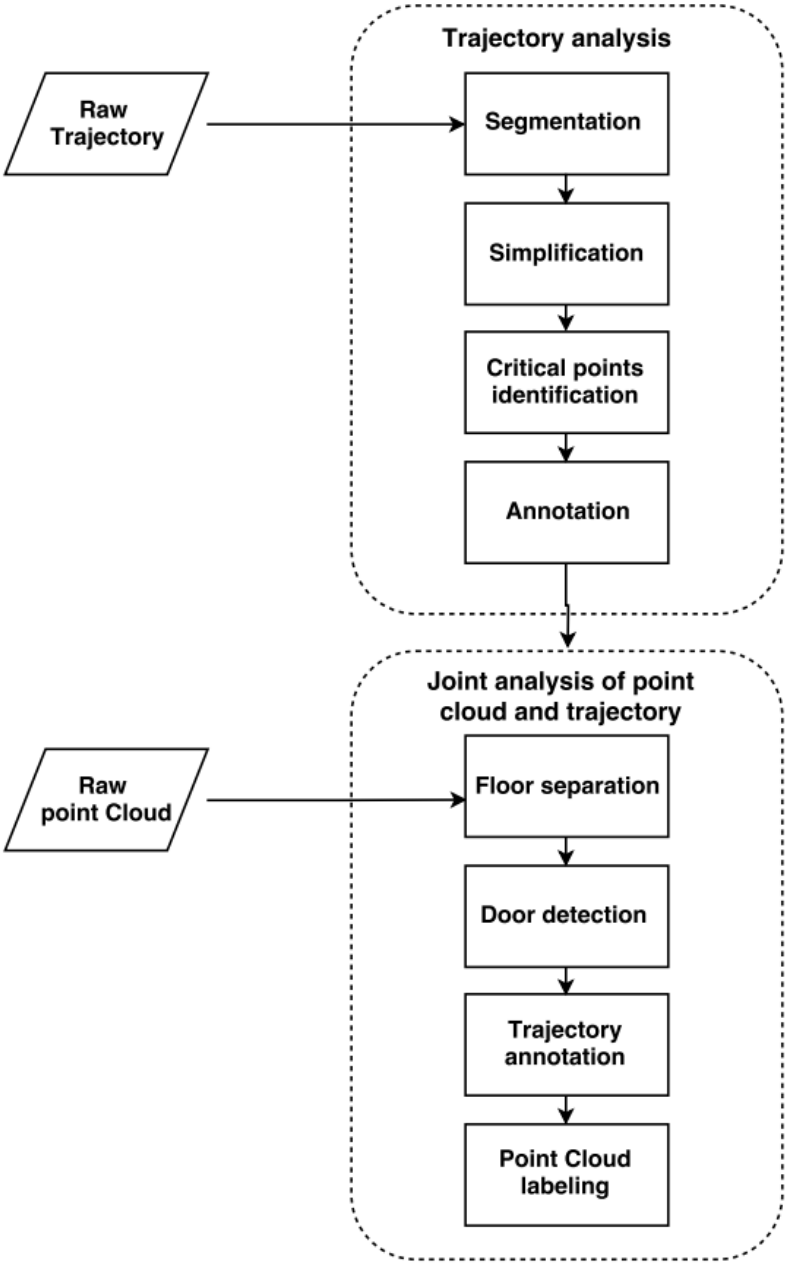



Figure 1.2: Process workflow.

# 2. Theoretical Background

The first part of this chapter will address related work which used trajectory for indoor reconstruction. That can be summarized into three main points: ($i$) reconstructing indoor environments based on human activities; ($ii$) categorizing indoor environments using mobile robots; ($iii$) modeling indoor environments using mobile laser scanning data. Also, there is a short overview for the laser scanning systems which were used to capture the test datasets.

## 2.1 RECONSTRUCTING INDOOR ENVIRONMENTS BASED ON HUMAN ACTIVITIES

Grzonka et al. (2012) proposed an approach to recover 3D human trajectory based on actions extracted from wearable motion capture suit. The actions represent activities such as walking, door opening, and stairs climbing. The constructed trajectory and human activities were used to derive approximate geometrical and topological maps for the environment as shown in figure 2.1.

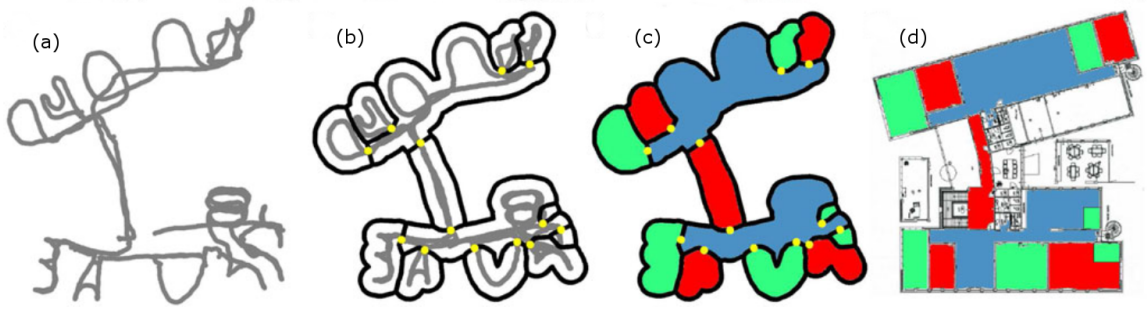



Figure 2.1: ($a$)Human trajectory. ($b$)Detected doors in yellow dots. ($c$)Approximated map with the distance between a wall and the nearest trajectory 1.5 m at maximum. ($d$)Actual floor plan. Image courtesy of Grzonka et al. (2012).

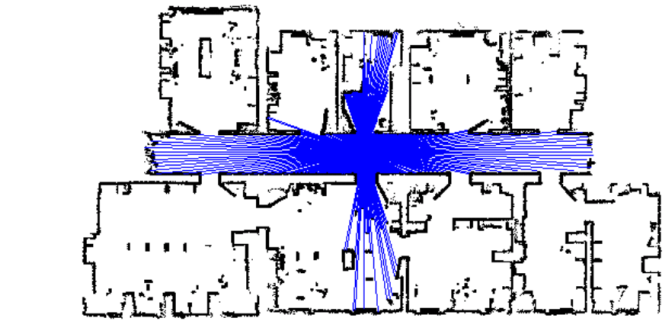
Human trajectory data by itself can provide valuable information which can be used to derive approximated floor plan. By analyzing the laser scanner trajectory, this can be used for detecting staircases and labeling the point cloud. Then, the labeled point cloud can be used to generate 2D floor plans and 3D models. Also, identifying point of interests like doors can be used as key element for constructing topological floor planes and navigational maps. The door acts as a connector between different spaces in the indoor environment.

## 2.2 CATEGORIZING INDOOR ENVIRONMENTS USING MOBILE ROBOTS

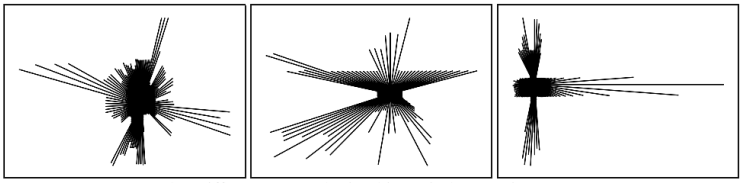
In the area of robotics, Mozos (2010, pp.15-34) classified indoor places into semantic categories based on a single observation of the range sensor along the robot's trajectory. A supervised classification approach using AdaBoost algorithm was implemented to learn a strong classifier. Then,

different places in the environment were classified based on a decision list of multiple binary classifiers. Laser range data for a single observation was used as each indoor class has a different structure as shown in Figure 2.2. Intermediate training data was simulated and trained along the trajectory based on each pose as shown in Figure 2.3.

Following research combined the laser range data with vision features of specific pose Mozos (2010, pp.57-69). Vision features were used to add more classes such as kitchen and office. Also, the combination improved the total accuracy of the classifier compared to using range data only. The training process for this approach done by manual data preparation and data from the simulated trajectory. Similar studies addressed the same problem using the Voronoi random field to classify the places (Friedman, Pasula, & Fox, 2007), and a comparison between different AdaBoost algorithms based on laser range data (Soares & Araújo, 2014).



(a) Scan obtained in a corridor



(b) Different scans obtained in an indoor environment

Figure 2.2: ($a$) Scan with 360° field of view for a corridor. ($b$) Different scans for indoor environment. Image courtesy of Mozos (2010, pp.16).

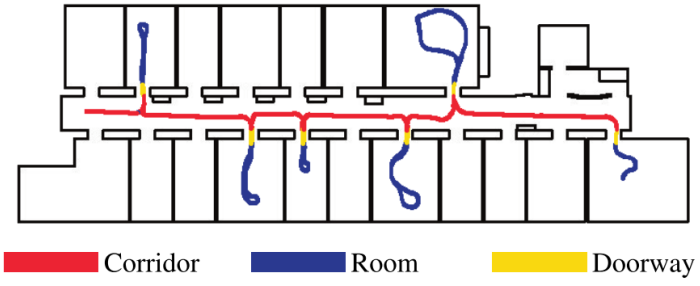



Figure 2.3: An example of training process along the trajectory of the robot. Image courtesy of Mozos (2010, pp.32).

Semantic information of the indoor environment can be used to label the point cloud or extract information such as rooms and hallways. The information is used to generate geometrical and topological floor plans and 3D models. Also, combining multiple sources of information such as

range data and vision features can improve the accuracy of the detected features and can increase the number of classes.

### 2.3 MODELING INDOOR ENVIRONMENTS USING MOBILE LASER SCANNING DATA

Xie and Wang (2017) proposed a method for reconstructing 3D indoor building model from mobile laser scanning data. The pipeline included generating 2D floor plan, detecting doors, and individual room segmentation. System trajectory was used in floor plan generation process and for door detection as well. For generating floor plan, first linear primitives were extracted using $RANSAC$ line fitting algorithm; then, it was used to define the outline for each room using a graph-cut technique. The trajectory was used to define the data term by utilizing the scanner position along the trajectory to construct virtual rays. These rays were used to identify the interior cells from exterior ones. Then, the cells were used to extract the inner and the outer walls to extract the floor plan. For detecting the doors, the projected system trajectory or the simulated rays were used to detect the candidate doors using the properties of Delaunay triangulation and alpha-shape. Finally, the generated floor plan was used to segment the rooms, and then they were extruded to generate a 3D model based on the heights from calculated elevation histograms. In the same context of indoor modeling, Díaz-Vilariño et al. (2017) proposed another scene reconstruction approach. System trajectory was used to detect the doors by profiling the ceiling along the scanning path. Then, the corresponding point cloud was partitioned into sub-spaces by using the timestamp from both datasets. After that, the sub-spaces were labeled with semantic information using ray tracing in 2D as energy minimization problem. After applying semantic labels to different spaces, the building model is reconstructed by analyzing the individual rooms first, then using the adjacency information to reconstruct the entire scene and the door locations.

Navigable spaces can be identified by utilizing the trajectory as it represents the scanning path. Staats et al. (2017) proposed a method to identify the navigable spaces in indoor environments. The walkable surfaces are identified through joint analysis between the point cloud and the trajectory of a mobile laser scanner. The trajectory was used to identify flat, sloped, and stairs surfaces based on the difference in slope angles between successive trajectory points. For identifying the surfaces, the classified trajectory was used as seed voxels for a region growing technique. Doors are detected by using proximity between trajectory voxels and the surrounding point cloud voxels. Then, it was used as one of the stopping criteria of the region growing process to identify the surface region per each room; in order to construct the final navigable space model. In the same context Fichtner et al. (2018) identified the navigable spaces inside the indoor environments by processing the points clouds within an octree-based data structure. Building elements such as floors, stairs, walls and obstacles were identified to extract the navigable space. A trajectory-focused research done by Nikoohemat et al. (2017) to extract semantic information from indoor scenes. They used the time attribute of the trajectory to detect the reflected points (ghost walls). The detection criterion was based on a lag parameter between the points of the constructed surface and the nearest trajectory points. Also, they proposed an approach to detect the openings, which included the door traversed during the scanning, by utilizing the trajectory dataset beside the point cloud and voxelated wall surfaces.

Finally, Turner and Zakhor (2015) proposed an approach to produce 2.5D extruded watertight models. Their approach used the shared boundaries between triangulated interiors to generate a 2d floor plan through a graph-cut based technique. The resulting room labels were used to produce 2.5D extruded watertight model. The approach classified the interior environment into rooms and proposed the openings for doorways. However, elements with a variety of levels such as split-levels and stairs are not handled by this method.

The geometric shape of the trajectory can be used to detect the door openings. Then, doorways can be utilized to identify different spaces in the indoor environment. Also, time attribute can be used related to the point cloud dataset, and extract information like detecting the reflections. The scanning position of the laser scanner on the trajectory can be used to construct line of sight, and it can be used for floor plan generation.
Working with IMLS trajectory as a spatiotemporal dataset can have more investigation by means of analyzing the behavior of the scanning system. This information can be integrated with the point cloud dataset to interpret the indoor scenes.

## 2.4 HANDHELD MOBILE LASER SCANNERS

GeoSLAM ZEB1 and ZEB-REVO 3D laser scanning systems were used to capture the test datasets. The systems have a 2D time of flight laser rangefinder, and they can record more than 40,000 measurement points/second. The scanning range is up to 30 meters indoor. The ZEB1 scanner was used to capture the Fire bridge office dataset. The scanner is mounted with the inertial measurement unit (IMU) on one or more spring. This makes the scanner head oscillates freely to get a 3D field of view. On the other hand, the ZEBREVO sensor was used to obtain the Technical University of Braunschweig building dataset. The scanner is coupled with the IMU and mounted on a motor. ZEB-REVO can be used as a used as a handheld scanner, or it can be mounted on mobile platforms such as UAV or a vehicle. Simultaneous localization and mapping (SLAM) algorithm are used to generate a 3D point clouds by combining the 2d laser data with the IMU data. It estimates the location of the deceive and a map for the environment. The main advantage of the SLAM algorithm is that the scanning system can operate through a complex indoor environment without the need for GPS (GeoSlam, 2016). Figure 2.5 shows the difference between the shape of the ZEB1 trajectory and ZEB-REVO trajectory due to the difference in the scanning mechanism. More detailed comparison between state-of-the-art scanning systems was provided by Lehtola et al. (2017); Maboudi, Bánhidi, and Gerke (2017).

(a) (b)

Figure 2.4: Left; ZEB-REVO laser scanner. Right; ZEB1 laser scanner.

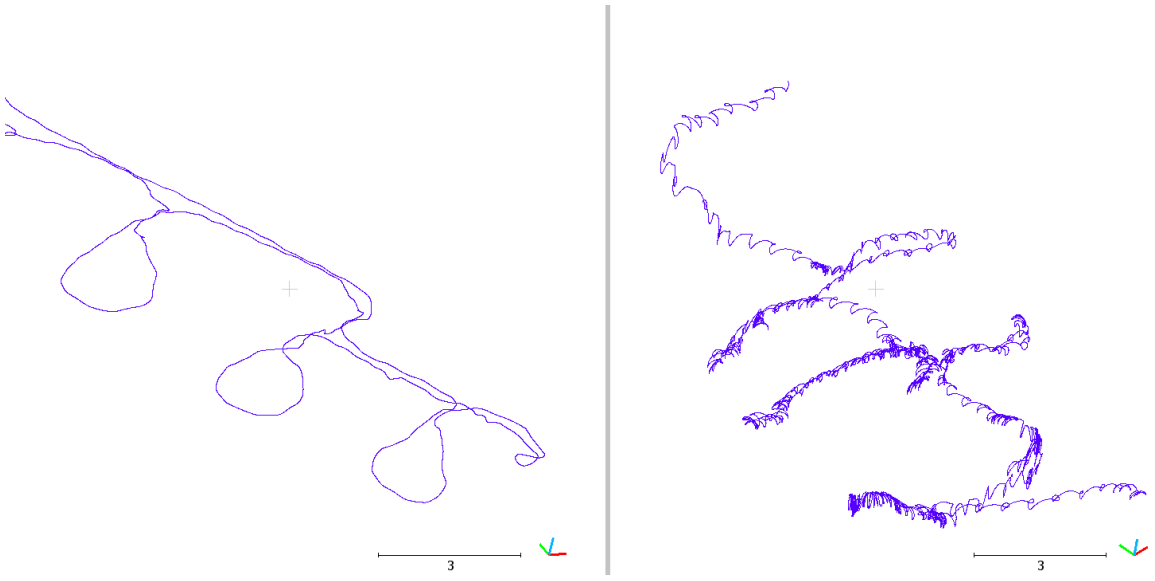



Figure 2.5: Left; a subset of ZEB-REVO trajectory. Right; a subset of ZEB1 trajectory.

## 2.5 LINE SIMPLIFICATION METHODS

The line simplification algorithms aims to eliminate the redundant points and maintain the shape information as the trajectory consists of points which are denser than necessary to perform the trajectory analysis. This simplification can help to deliver the trajectory of ZEB1 and ZEB-REVO to the same simplified geometric shape. Several methods were introduced in literature, but we focus on point-selection/rejection simplification methods which preserve the direction of the trajectory without loss of attributes or position of the original points. The choice of the suitable method is based on the experiments in section 3.3. The algorithms can be grouped into five categories defined by McMaster (1987) and described by Shi and Cheung (2006) and Regnauld and McMaster (2007) which include:

1. Independent point algorithms.

   Independent point algorithms do not consider the relationship between neighboring points within the simplification process. For example, the *Nth Point Elimination* removes a number of successive points N and only keeps one along the trajectory. Also, it can be extended to be an iterative process with a minimum number of points to keep.

2. Local processing routines.

   Unlike independent points algorithms, local processing routines consider the relationship between direct neighboring points. An example of these routines is the *Perpendicular Distance* algorithm which eliminates an intermediate point between two consecutive points if it is within a distance threshold from a line connecting these two points. Another example is the routine of distance between points or the *Radial Distance* algorithm. This algorithm walks through the line and eliminates the points near a key point if they are within a distance threshold, then, the first point after this threshold is picked up as a new key point. The local processing routines are influenced by points density more than the shape complexity. Figure 2.6 illustrates the concept of both routines.

3. Unconstrained extended local processing routines.

   The algorithms which belong to this category consider more than one neighbor during the process. An example is *Reumann-Witkam* algorithm (Reumann & P. M. Witkam, 1974). It is a perpendicular distance-based algorithm. It constructs a strip with threshold distance parallel to a line segment which is built from two neighboring points. Then, it eliminates the following points within the strip. After that, it constructs a new strip from the next points after the previous strip. It walks through the trajectory starting from the first point until the end.

4. Constrained extended local processing routines.

   Constrained extended local processing routines apply the same criteria like the previous category but it is constrained to a search region around the line segment. Example of these routines is *Opheim* algorithm (Opheim, 1981). It is similar to *Reumann-Witkam* algorithm, but it has a constraint over the search area using radial distance and perpendicular distance thresholds.

5. Global routines.

   The algorithms in previous categories walk through the line from the beginning till the end, but in global routines, the entire line geometry is considered during processing. Examples of this categories are *Douglas-Peucker* algorithm (Douglas & Peucker, 1973) and *Visvalingam-Whyatt* algorithm (Visvalingam & Whyatt, 1993).
   The *Douglas-Peucker* algorithm is commonly used for global simplification and for detecting the geometric critical points along a line. Also, it is influenced most by line complexity rather than points density.

The line simplification process is also known as trajectory compression in the literature which is concerned more about moving objects (Sun, Xia, Yuan, & Li, 2016). In this study, the trajectory compression is divided into two steps of trajectory simplification and critical points detection.

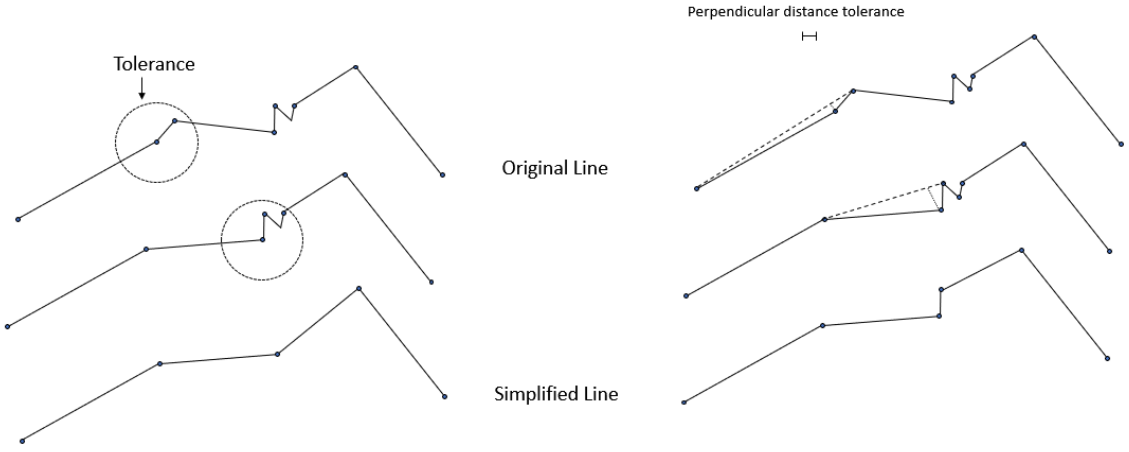



Figure 2.6: Radial distance routine (left) and perpendicular distance routine (right).

# 3. Trajectory Analysis

## 3.1 OVERVIEW

The main focus of this chapter was to add additional labels to the raw trajectory to enrich it with semantic information. The analysis was conducted based on the trajectory data only without the help of the point cloud dataset. A Trajectory $T$ of the indoor mobile laser scanner is composed of a set of points ($p_1, p_2 \ldots, p_n$). Each point $P$ along the trajectory holds information about system translation and rotation as a function of time relative to the outer world (Bosse et al., 2012). For the testing datasets, a point comes in the form ($time$, $x$, $y$, $z$,$q0$, $q1$, $q2$, $q3$), where time is the timestamp in millisecond, ($x$,$y$,$z$) are 3D coordinates and ($q0$, $q1$, $q2$, $q3$) are system quaternion. Considering the trajectory as a spatiotemporal dataset, valuable information has been revealed by segmenting the points into meaningful parts according to specific criteria. These criteria included the change in height, moving direction, speed, and system rotation or a combination of different attributes. Figure 3.1 shows the pipeline of the trajectory analysis process. In the first step, the raw trajectory was segmented into floors and staircases. The raw trajectory was used as an input as the implemented methods can work on any trajectory dataset regarding the scanning system. After that, each floor sub-trajectory was simplified to eliminate the redundant points without losing the trajectory trend and data. Also, the simplification process unifies the level of complexity of the geometric shape of the trajectory from the different scanning systems. The simplification process focus only on the geometric shape of the trajectory without considering the semantic meaning of the data. To detect the semantic patterns of the trajectory, the simplified dataset was processed to detect the critical points which can describe the operator behavior in terms of movement direction, change in speed, and scanning system attitude. Finally, the trajectory was annotated with classes of room, corridor, doorway, and staircase based on the critical points distribution and predefined semantic rules.

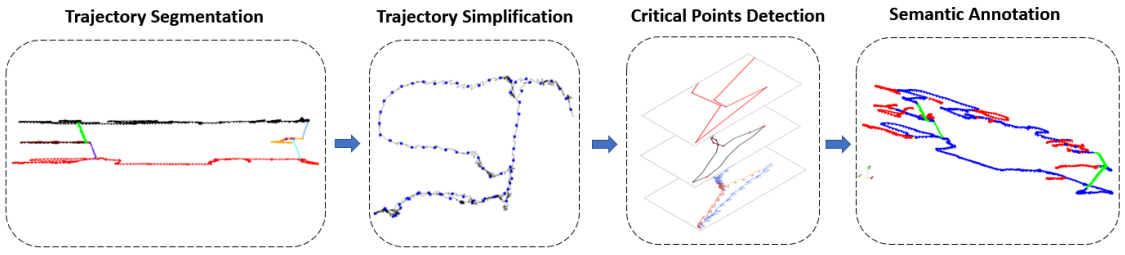



Figure 3.1: The pipeline of the trajectory analysis process: (1) trajectory segmentation; (2) trajectory simplification; (3) critical points detection; (4) semantic annotation of the trajectory dataset.

## 3.2 TRAJECTORY SEGMENTATION

In this step, the trajectory dataset was categorized into floors and staircases based on trajectory attributes. Such categories were a base for the following trajectory analysis steps and the joint analysis process. Three approaches were tested during the experiments to achieve the goal of this step. In the first approach, the dataset was segmented based on the height histogram. The second one was a combination of height histogram and Fisher's Natural Breaks Classification (Fisher, 1958). The final approach used a line growing technique based on height and time attributes. The raw trajectory was used during this process because adopted methods depend on the original points count in case of the histogram-based methods or the average points height within a sliding window in the line growing method. Also, these methods can work on any trajectory regarding the point density or the geometric shape complexity.

### 3.2.1 Segmentation based on height attribute

Height histogram can be used to split floors and ceilings from a point cloud dataset by detecting peaks in the histogram that represent dominant horizontal planes (Okorn, Xiong, Akinci, & Huber, 2010; Adan & Huber, 2011; Oesau, Lafarge, & Alliez, 2014). Similarly, height histogram was used to split the trajectory dataset as shown in Figure 3.2. A minimum bin size of 20 cm was used to detect any change in the floor level, and to separate floors from staircases. Bin size was entered manually as it depends on the trajectory points height variation. Bin size of 30, 40, 50 and 60 cm were also tested as the variation between z values along the trajectory in the same floor can be around 60 cm for ZEB1 system. The result of the histogram-based classification was

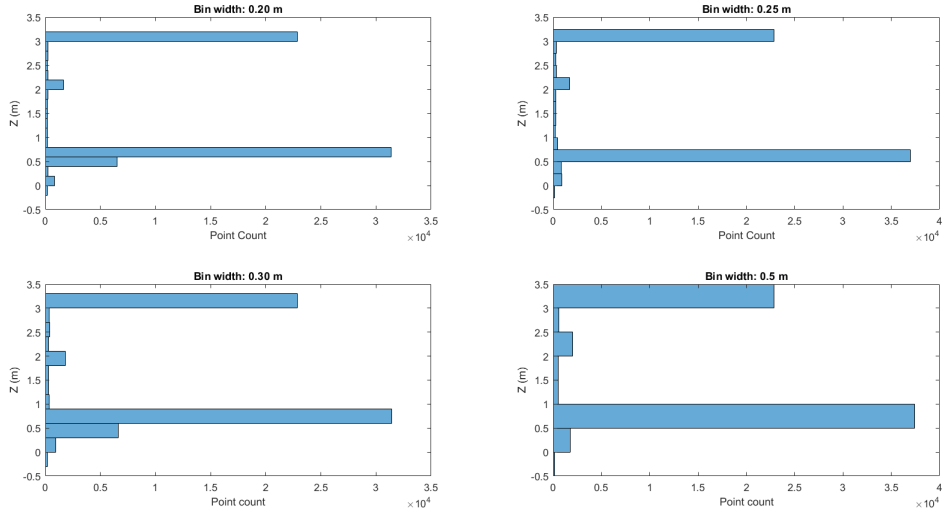



Figure 3.2: Height histogram for ZEB-REVO trajectory with different bin width.

influenced by choosing the proper bin size. Points on the same floor were misclassified if they were in a different interval as shown in Figure 3.3. A threshold of minimum point count per bin was specified to detect the histogram peaks and eliminate any local peaks. This threshold varied according to the scanning system and trajectory length. Also, the break value for each segment was not necessary the boundary of the larger peak. Thus, this method has been improved to detect the breaks between floor and staircase classes in an automated way.

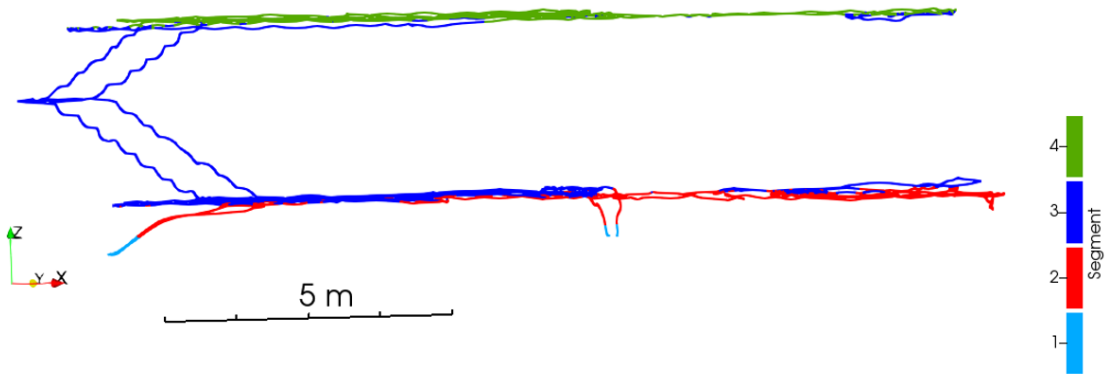



Figure 3.3: Segmented trajectory based on 0.25 m height histogram for ZEB-REVO trajectory, the problem appears when defining the classes with wrong values from the histogram.

An adopted method combines Fisher's Natural Breaks Classification (Fisher, 1958) with the generated height histogram with large bin width (e.g., 50 cm) was used. Fisher's Natural Breaks method aims to minimize the heights variance within each class and maximize it between different classes. This can be used to get breaking height values between floors and staircases. The implementation of the algorithm by Hilferink (2015) was modified to fit the trajectory dataset. First, height histogram was applied to detect the peaks. Then, distinct values of trajectory points height were sorted and processed to get the cluster breaks. Although this approach separated between floor and staircase classes, there was a clear problem in handling staircase class as there was a large variance between height values within the class as shown in Figure 3.4. Both approaches did not

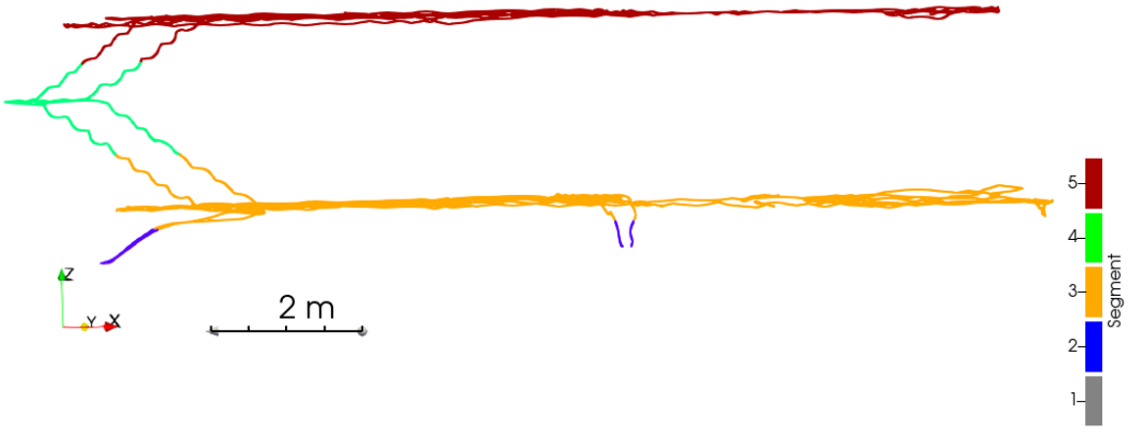



Figure 3.4: ZEB-REVO trajectory segmentation using Fisher's Jenks Natural Breaks.

consider the time attribute in the trajectory dataset. Further processing had to be applied to the results to differentiate activities based on time (e.g., going up the stairs has different time interval than going down, although both activities within same space).

### 3.2.2 Segmentation based on height and time attributes

In this approach, the spatiotemporal aspect was considered for global trajectory classification. **Definition 1 (Activity class)**. The sub-trajectory label represents specific scanning activity (climbing the stairs or walking on a floor) in certain time interval. It contains one or more sub-trajectory segments. Merging criteria is based on adjacency and the minimum number of points within the sub-trajectory.
**Definition 2 (Global Class)**. Final class labels which have one or more activity class. These activities have the same height, within a specified tolerance, but in different time intervals.
The algorithm used a line growing technique with three iterative steps.

1. A sliding window with fixed width (number of points or seconds) is walking through the trajectory dataset The trajectory points are ordered by the time attribute. The average point height is calculated for each window (sub-trajectory segment), then it is compared with the average point height of the two preceding neighboring segments. Second neighboring segment height threshold is used to detect any global change in heights. The height can change gradually and still below the first threshold. The trajectory segment stops growing when the difference between neighboring windows is exceeding a certain threshold and is given a certain identifier. The following segment starts to grow under same stopping criteria. The output of this step is candidate sub-trajectories with unique identifiers as shown in Figure 3.5.

2. Then, the adjacent sub-trajectories segments are merged if they have number of trajectory points less than minimum point count threshold. This threshold is used to identify horizontal surfaces, which have high point count, from the sloped ones. The merged segments are labeled with unique activity class identifier shown in Figure 3.6. This activity represents a continuous movement within a specific time frame for at a certain height level.

3. Finally, activity class segments with similar average height are labeled as one global class. A small tolerance (e.g., 10 cm) was used to group these segments. Each global class is indicated if it represents a floor or staircase based on the height range as shown in Figure 3.7.

The output of this process was stored in a $SQLite$ database. This made the process of attaching attributes to the trajectory points easier, and the intermediate data can be recalled and used without the need to reproduce the analysis. Also, all the processing was operated on a single file, which is the database, using $SQL$ statements and this ensured the data integrity and gave the ability to execute more complex queries for the following analysis steps.

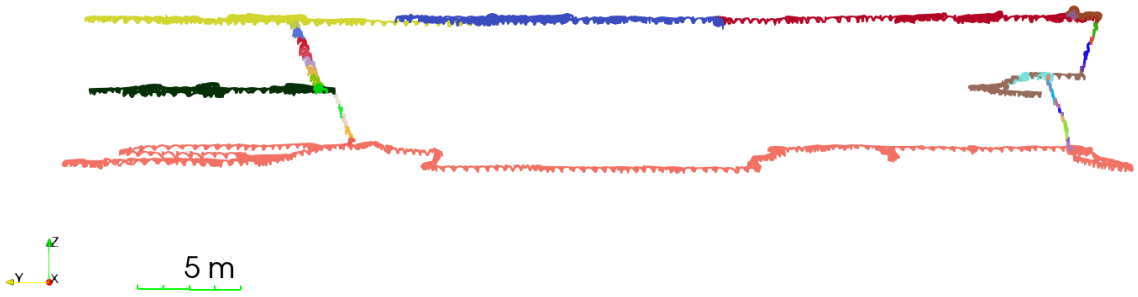



Figure 3.5: Growing sub-trajectory segments for ZEB1 data.

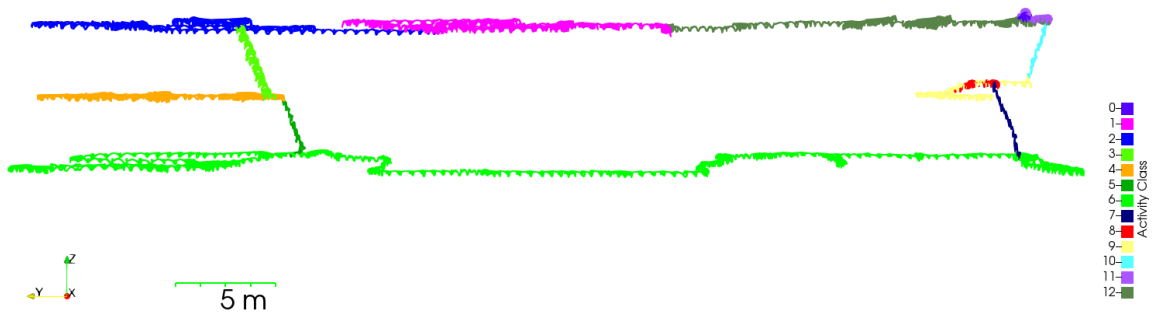



Figure 3.6: Activity labels for ZEB1 dataset. It represents a scanning activity based on a change in height for a specific time interval.

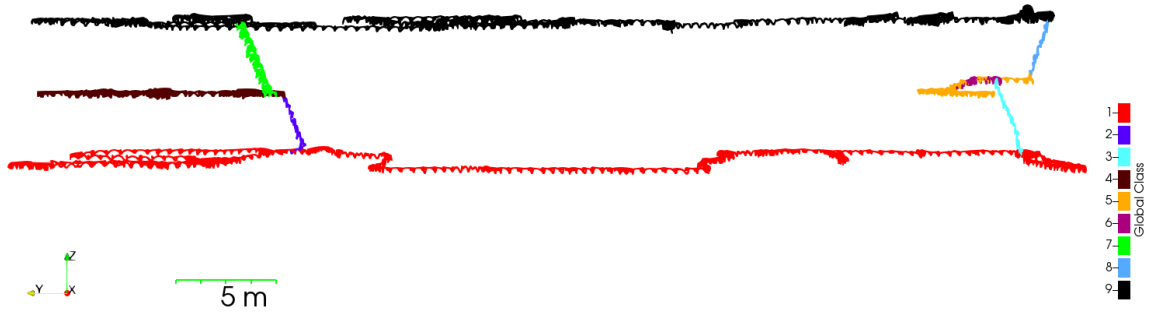



Figure 3.7: Global labels for ZEB1 dataset. Each global class represents a floor or staircase.

### 3.3 TRAJECTORY SIMPLIFICATION

The process aimed to simplify the trajectory shape for each floor sub-trajectory while preserving the directions. This process was applied on the segmented floors from step 3.2.2. Figure 2.5 shows a subset of a trajectory belonging to ZEB-REVO (left), and ZEB1 (right). It was noticeable that the representation of the trajectory varies from a scanning system to another. This process helped to make the computations with the trajectory simpler for further steps as the trajectories of different scanning systems would have the same shape complexity. Also, the trajectory consists of points which are denser than necessary to perform the trajectory analysis . Both datasets were processed in this step to remove the redundant points and to make them have the same level of geometric representation.
As the geometric aspect of the trajectory was considered in this process, the simplification methods which have been tested were selected based on the literature review. Each category from section 2.5 was represented by one or more algorithm in the tests. The implementation for these algorithms is adopted from geometric libraries (Boost, 2017; de Koning, 2011) and pseudo codes from SUNY Institute of Technology (2012) web page. It considered the temporal dimension of the data as trajectory points were ordered by time attribute, and these algorithms walk through the points from the starting point till the end. Also, a combination of simplification methods was tested to find the optimal simplification method.
The chosen simplification algorithm was "The routine of distance between points" or Radial distance routine described in (Long, Wong, & Jagadish, 2013). It belongs to the local processing

routines category which consider the consecutive neighboring points. Also, this routine is only influenced by the point density and it was not affected by the geometric shape of the trajectory. Thus, both trajectory datasets can have the same geometric shape criteria for following processing steps.

The algorithm starts from the first point as a key point. Then, it eliminates the following points within a specified tolerance. Then, the first point outside that tolerance becomes the new key point. The algorithm stops when reaching the last point.
Figure 3.8 shows a subset of the simplified trajectory of the ground floor of ZEB1 dataset compared to the original one. It preserves the general shape with fewer points. The trajectory had 60684 points and it was reduced to 175 points. Several tolerance values were tested, and tolerance of 0.80 meters was used for both datasets. The equal spacing of 0.80 meters was large enough to reduce the complex shape of ZEB1 dataset without affecting the general shape. Also, the line segments between each successive point could be compared along the trajectory. The key difference between using the simplification algorithms over the point cloud-like downsampling algorithms it that the simplification algorithms not only consider the spacing between the points, but also it respect the time order of the points.
The selection of suitable algorithm was based on visual interpretation and suitability for further analysis steps. Overview of the experiments applied on the trajectory using the simplification algorithms are illustrated in Figure 3.9.

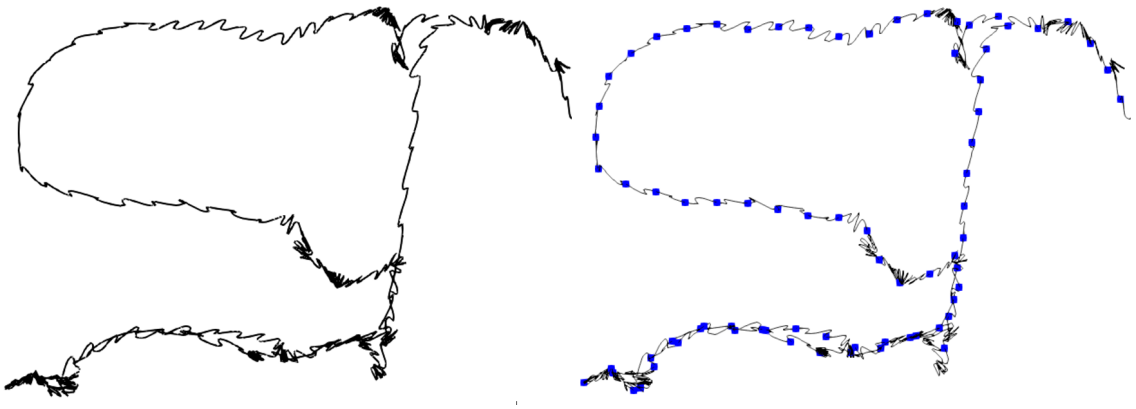


Figure 3.8: A subset of ZEB1 trajectory (left), the simplified trajectory using radial distance of 0.80 meters in blue dots overlaid on original trajectory (right).

### 3.4 CRITICAL POINTS IDENTIFICATION

The simplification process focused only on the geometric shape of the trajectory without considering the semantic meaning of it. The aim of the simplification process was to deliver the trajectories which have different geometric complexity to the same level of abstraction. After that, the simplified trajectory was analyzed to detect the critical points that can reveal the hidden information. Critical points are points along the trajectory that can divide it to homogeneous parts in some sense. They describe the important features and transition of the trajectory (Buchin, Driemel, Van Kreveld, & Sacristan, 2011; Sack & Urrutia, 2000; Jin, Cui, Wang, & Jensen, 2016). For example, change in movement direction can be described by fewer points, and the redundant points are removed. This can make the data processing simpler if it is based on direction change.

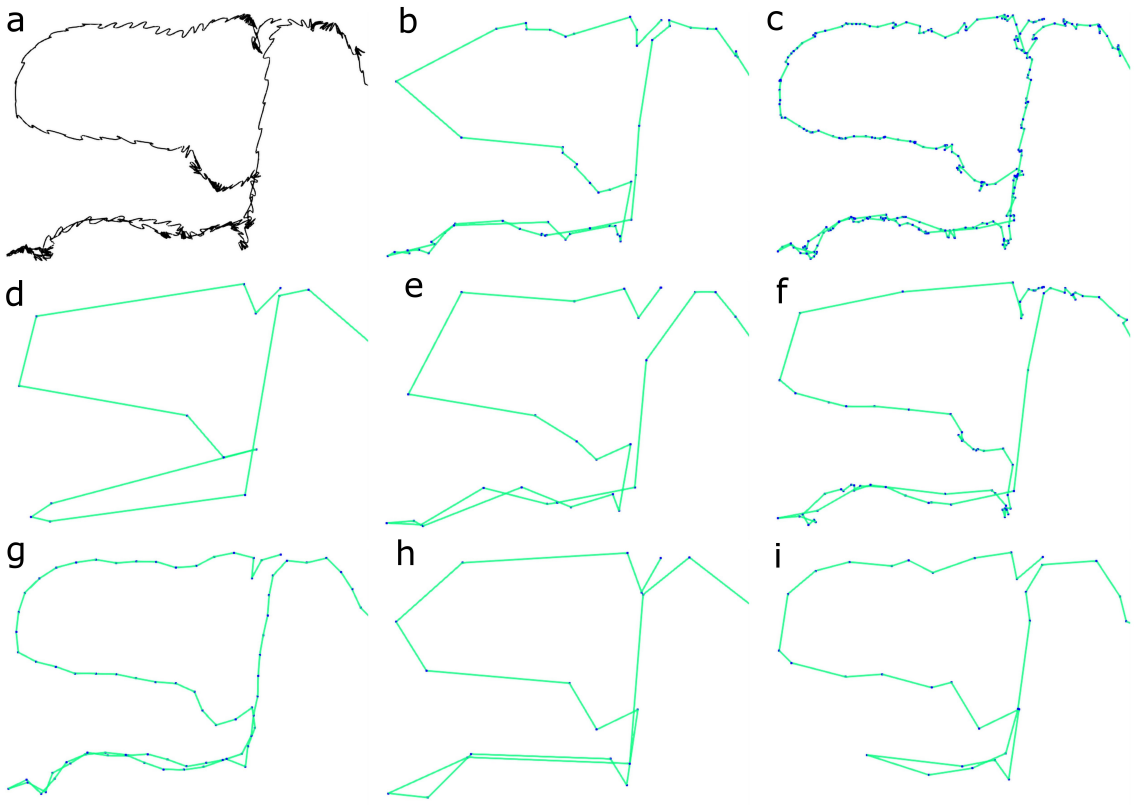



Figure 3.9: $(a)$A subset of ZEB1 trajectory, the simplified trajectory in blue dots overlaid on original trajectory $(b)$Nth point elimination, $(c)$Opheim, $(d)$Perpendicular distance, $(e)$Nth point stepping, $(f)$Reumann-Witkam, $(g)$Radial distance, $(h)$Douglas-Peucker, $(i)$Visvalingam-Whyatt.

Each point which belongs to the trajectory dataset holds information about translation and rotation of the scanner as a function in time. Following this fact, the critical points were based on the geometry, time, and scanner attitude.

### 3.4.1 Geometry-based critical points

The geometric shape of the trajectory was characterized by the change in movement direction or the important points that can describe the general shape without affecting the trend. The simplified layer from step 3.3 was used as an input for this step. Two approaches were applied. In the first approach, the Douglas-Peucker algorithm was applied over the simplified trajectory as shown in Figure 3.10. This algorithm is widely used to detect the critical points as it preserves the direction trends using a perpendicular distance-based threshold (Meratnia & de By, 2004). The choice of this algorithm was based on the experiments while combining multiple simplification methods at the previous step.
The second approach was based on the change in movement direction between two simplified segments. A segment is defined as the line connecting between two consecutive points along the trajectory. The angle of one segment is calculated from the positive x-axis counter-clockwise as the calculations are in the x-y plane. The change in angles was calculated for each two trajectory line segments and compared to a threshold. If the angle difference exceeded that threshold, the connection point flagged as a critical point. A threshold of 25 degrees was applied as shown in

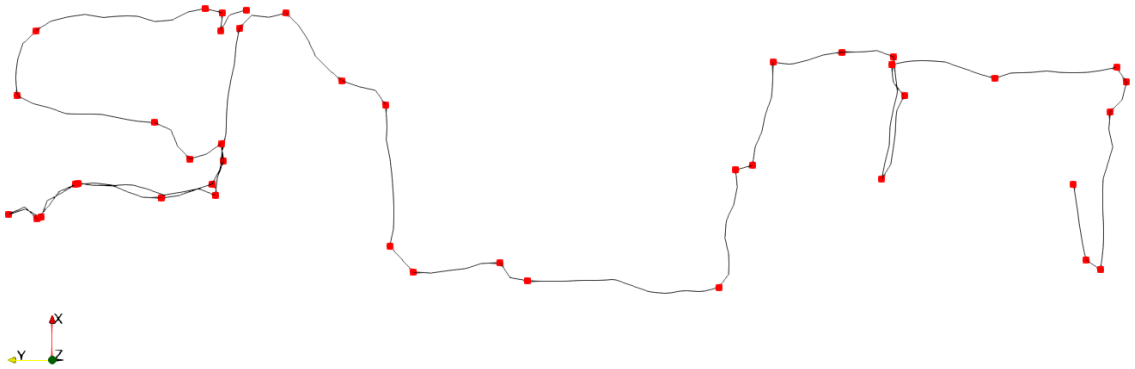


Figure 3.10: Geometric critical points using Douglas-Peucker with 0.5 m threshold for the simplified ZEB1 floor.

Figure 3.11.

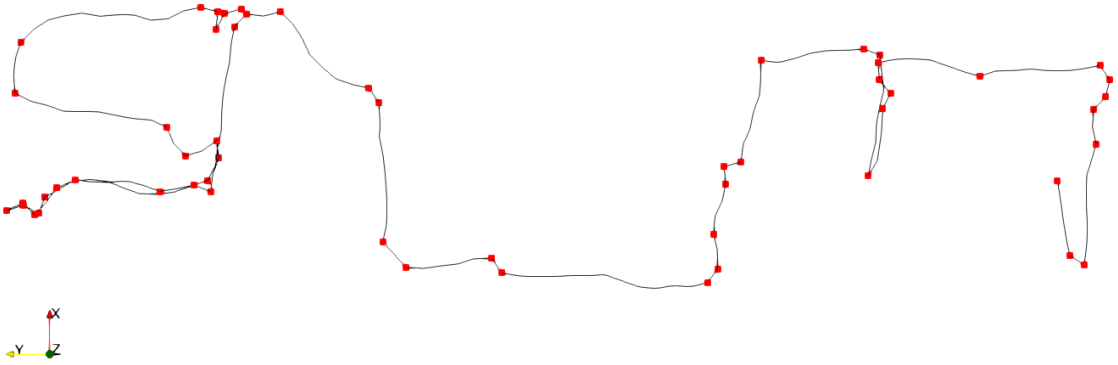


Figure 3.11: Geometric-based critical points using direction change with 25 degrees threshold for the simplified ZEB1 floor.

Several thresholds were tested for both approaches. Douglas-Peuker based approach showed better results in detecting the critical points as it considers the entire line during simplification process. The direction-based approach detects the local critical points which were not important to describe the global trend, and it had some failure cases. Moreover, Douglas-Peuker showed better results when using higher threshold values.

### 3.4.2 Time-based critical points

Based on the user guide of ZEB-REVO scanner(GeoSlam, 2016), the transition through doorways or around tight bends should be scanned slowly to make sure that the scanner can detect all features properly. As trajectory data is a spatiotemporal dataset, the change in moving speed inside the environment can be utilized to detect doorways. The distance between every two points can be considered equal as the simplified layer of the trajectory was based on radial distance. Thus, the time difference between each neighboring point was calculated and compared to a certain threshold.
For points $P_n$ and $P_{n+1}$ the simplified trajectory T', Time difference = $P_{n+1}(time)$-$P_n(time)$ where time is the time stamp of each point in milliseconds.

If the time difference exceeded a threshold in N seconds, the two points were marked as time critical points as shown in Figure 3.12. The threshold of the number of seconds represents the operator speed during the scanning process as the length of the segment which connects the two points was equal to the radial distance simplification threshold. The choice of the time threshold value is influenced by the simplification threshold.

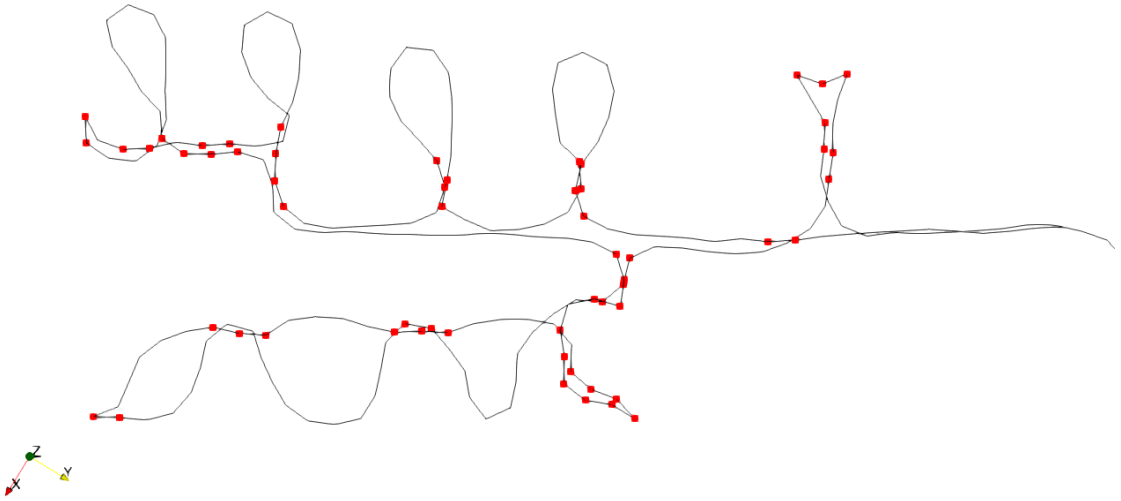



Figure 3.12: Time-based critical points with 2.5 seconds threshold for simplified ZEB-REVO floor.

### 3.4.3 Scanner attitude-based critical points

System quaternions were converted to Euler angles (roll, pitch, and yaw) in order to make the visualization process more sensible. The trajectory points was plotted over time to show the rotations of the scanning system. By analyzing the rotation around the vertical axis (yaw angle), patterns of the change in the direction of scanning were detected along with the movement direction. All figures related to the plotting over time is listed in section A.3.1.

## 3.5 SEMANTIC ANNOTATION

In the last step in trajectory analysis process, initial annotation classes were assigned to the raw trajectory dataset as labels. These classes were defined based on the identified critical points and semantic roles of indoor spaces. The classes included rooms, corridors, doorways, and staircases.

Adopted from trajectory based clustering approaches (Rocha, Times, Oliveira, Alvares, & Bogorny, 2010; Xiu-Li & Wei-Xiang, 2009), intersecting patterns can be detected by joining the change in direction with trajectory points clusters as well as detecting the scanning system behavior. Basically, the previous analysis step had three layers of trajectory as shown in Figure 3.13. The first layer was the raw trajectory which revealed the scanner behavior in terms of changing in yaw angle. The second layer was based on the time-critical simplified layer. It was used to detect the stops or the slow down along the trajectory. The top trajectory layer was the geometric abstraction of the trajectory. This layer was generated by applying the Douglas-Peuker simplification algorithm on the simplified layer. It was used to analyze the change in the movement direction along the trajectory to detect the indoor classes.

For the raw trajectory layer, it showed that both scanning systems behave in a different way and it depended on the operator behavior. Critical points from the scanner system attitude were not used in further analysis steps. The geometric-based and time-based critical points were used as an input for the next step.

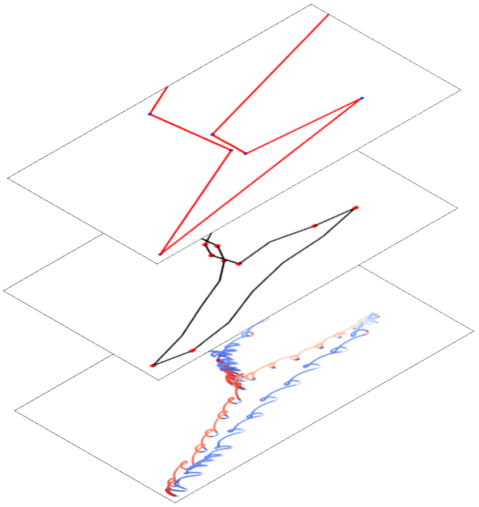


Figure 3.13: A conceptual figure for the three layers of annotation process, geometric-critical points (top), time-critical points (middle), and raw trajectory with scanner attitude (bottom).

The annotation process followed the following steps for each floor:

1. Input layer

   Geometric-based critical points was merged with time-critical points and ordered by time attribute as shown in Figure 3.14. Each critical point was flagged either to be time-critical or geometric critical, and it contained its trajectory attributes. Every two successive points form a line segment. This segment holds information like distance and direction. Then, the procedure started to iterate over the segments to check first for rooms, then corridors, and finally the doorways.

2. Room identification

   Room was identified by searching for the self-intersected trajectory to form a closed loop or to detect semi-closed one as the operator walks through the door to enter and leave the room. The check starts when finding a time critical point or detecting a change in the direction of trajectory segments. At this trajectory point (key point), nearest neighbor search within a distance threshold was applied to the merged critical points. Then, following rules were applied between the key point and its nearest neighbor in an iterative process through all the neighbors:

   - Check for scanning time between the two points and it should not exceed maximum scanning time threshold. The scanning time is identified as the difference in time attribute between the two points. The scanning time indicates how long the operator was in the space which can indicate the size of it.
   - Related check to the scanning time is that the maximum number of critical points (time-based and geometry-based) between the two points should not exceed a threshold. This can indicate the trajectory length between the two points without the need

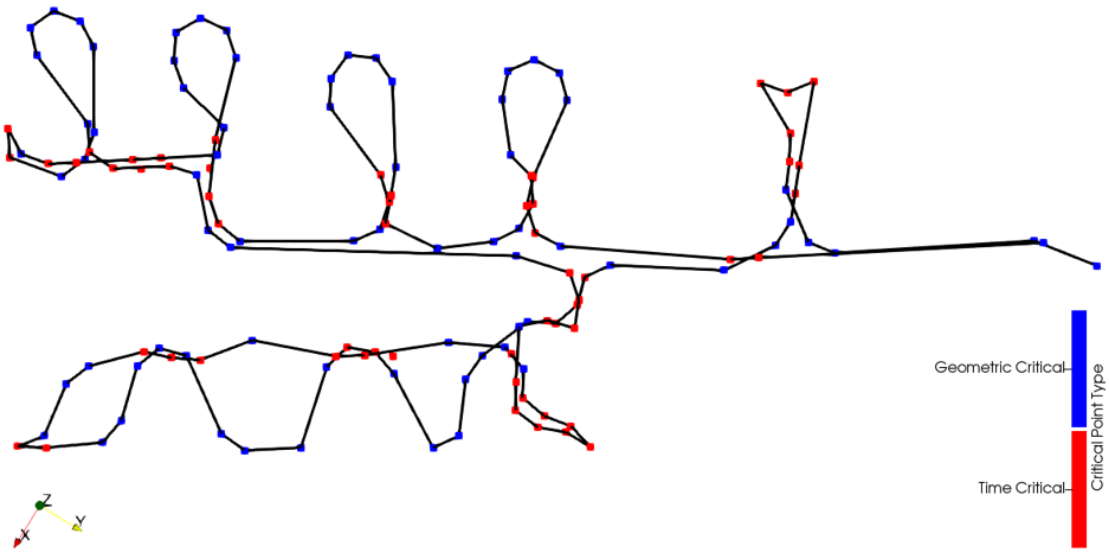



Figure 3.14: Geometric-critical points merged with time critical-points for ZEB-REVO floor, trajectory segments in black line are connecting the points.

to calculate the actual length. The spacing between any points will not be less than trajectory simplification threshold as the critical points were identified based on the simplified trajectory. The maximum distance can be calculated by multiplying the maximum number of critical points by the simplification distance.

- If the neighboring point met the previous checks, the next rule is to search for self-intersections or semi-closed loops. For self-intersections, a line constructed from the key point and the neighboring point including all the intermediate critical points and the self-intersection was checked by Boost geometry library. For the semi-closed loop, the minimum distance between the trajectory segment containing the key point and the segment containing the neighboring point is calculated and compared to the minimum neighboring distance threshold.
- The final check for the point is to check for minimum trajectory length to form a room, to avoid small bends or self-intersections.

If one of the point from nearest neighbor search satisfies these rules, the trajectory segment connecting the points between that point and the key point is annotated as a room. If not, the trajectory segment containing the key point will be checked for the other classes. The case of a room which has more than one door was not handled by this algorithm to avoid over fitting.

3. Corridor identification

   A corridor trajectory segment was identified by considering it to be parallel to the main building direction or perpendicular to it with some tolerance (e.g., 10 to 15 degrees) . To detect the main building direction, the minimum bounding box of floor trajectory was calculated. Then, the main direction was defined to be parallel to the long side of the box.
   If the line segment did not satisfy the corridor rule, it was annotated as "$undefined$" as shown in Figure 3.15.

After iterating over all the critical points and annotating the trajectory segments, the "$undefined$" labels were updated to be corridors.

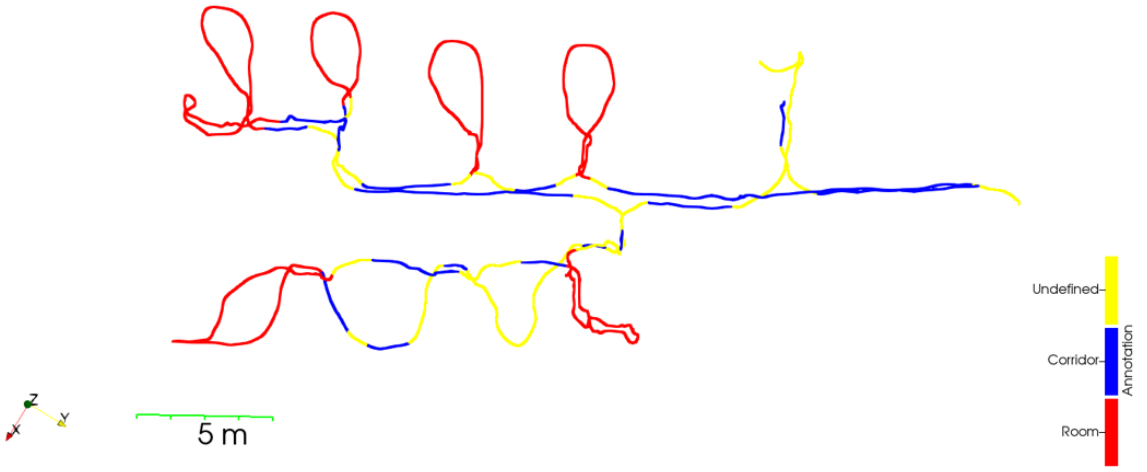



Figure 3.15: An intermediate sample of annotation before final processing for ZEB-REVO floor.

4. Doorways

   Finally, door positions were identified to be the critical point between the annotated sub-trajectories representing different spaces (end point of one sub-trajectory and starting point for the other as they are connected) as shown in Figure 3.16.

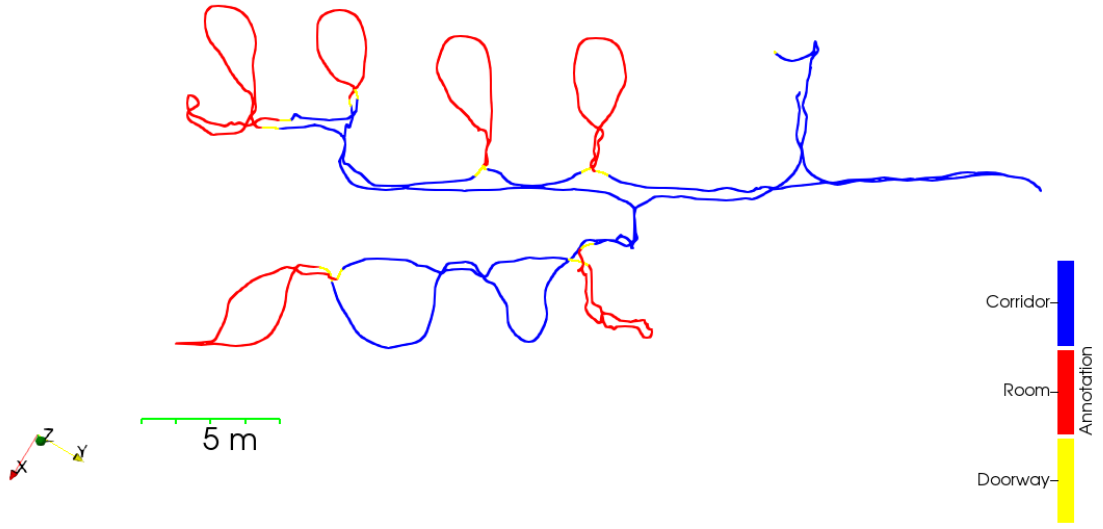



Figure 3.16: Example of annotated ZEB-REVO floor.

After annotating the trajectory segments, the annotations were updated in the raw trajectory table in the $SQLite$ database based on the time interval of each sub-trajectory. Door positions were updated with a margin of 0.5 seconds around the time attribute of the point. Staircases were annotated from step 3.2 of the trajectory segmentation as shown in Figure 3.17.


Figure 3.17: Example of annotated ZEB-REVO dataset.

## 3.6 EXPERIMENTAL RESULTS

In this section, the key results of the trajectory analysis process are illustrated. The two datasets described in section 1.3.1 were used for all experiments for trajectory analysis.

### 3.6.1 Trajectory segmentation

Trajectory dataset was segmented using the line growing technique described in section 3.2.2. Table 3.1 lists the parameters used to segment ZEB1 dataset shown in Figure 3.7 and ZEB-REVO dataset shown in Figure 3.18. Tables A.1 and A.2 show the statistics of different trajectory segmentation stages for ZEB1 dataset and ZEB-REVO dataset respectively.


Figure 3.18: Global labels for ZEB-REVO dataset. Each global class represents a floor or staircase.

Table 3.1: Line growing segmentation parameters for ZEB1 and ZEB-REVO datasets

| Parameter Name | Parameter Value |
|---|---|
| Trajectory point step | 400 points (equal to 4 seconds for 100 Hz scanning rate) |
| First neighbor height difference threshold | 0.2 meter |
| Second neighbor height difference threshold | 0.3 meter |
| Minimum points per global class | 1000 points (equal to 10 seconds for 100 Hz scanning rate) |
| Merging height threshold | 0.1 meter |

### 3.6.2 Critical points identification

Table 3.2 provides the parameters used to simplify the trajectory and to identify the critical points for the next process. The simplification was based on radial distance algorithm, and the identified critical points were analyzed on top of the simplified trajectory.
Figure 3.19 shows the critical points for ZEB-REVO basement. The total number of points was 35,910, and it was reduced to 124 critical points. For ZEB1 dataset, Figure 3.20 shows the merged critical points for the top floor. The floor had a total of 94,416 trajectory points, and It was reduced to 143 critical points. Also, only 54 points were identified as geometric critical and time critical at the same time as shown in Figure 3.21. This points can indicate the possible doors position as the operator tends to slow down near the transitions between the spaces.
During the experiments, several combinations of thresholds were tested. Figure A.4(b) shows different parameters for time critical points for ZEB1 floor. Time threshold of 6.0 seconds was used, and Figure A.4(b) shows the critical points which are geometric-based and time-based at the same time.
The parameters were minimized to get denser critical points for the annotation process.

Table 3.2: Critical points detection parameters.

| Dataset | Simplification tolerance | Time threshold | Douglas-Peuker tolerance |
|---|---|---|---|
| ZEB-REVO | 0.80 meters | 2.5 seconds | 0.20 meters |
| ZEB1 | 0.80 meters | 3.0 seconds | 0.50 meters |

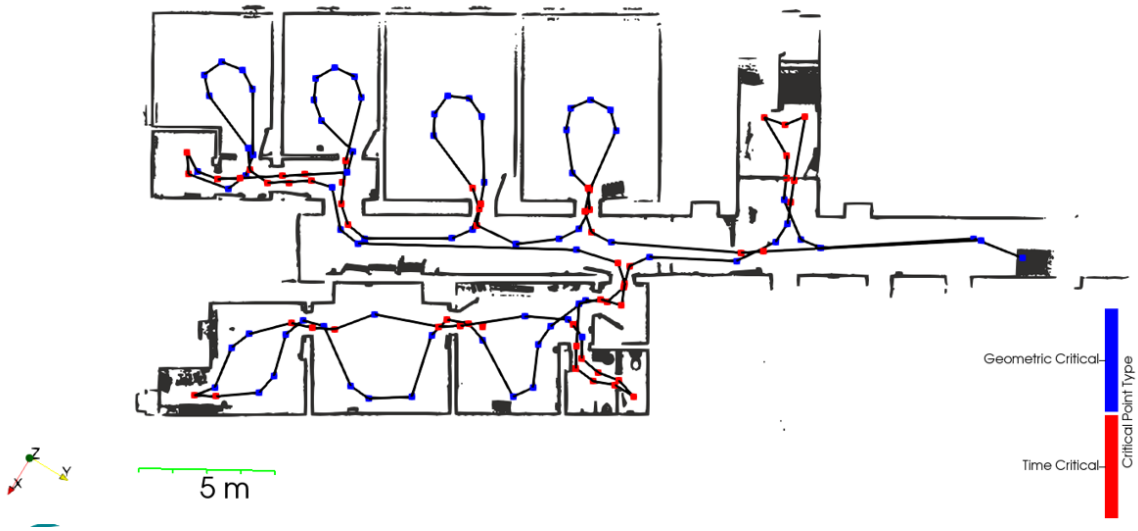



Figure 3.19: Critical points for the basement of ZEB-REVO dataset. Total of 124 critical points compared to 35,910 original points.

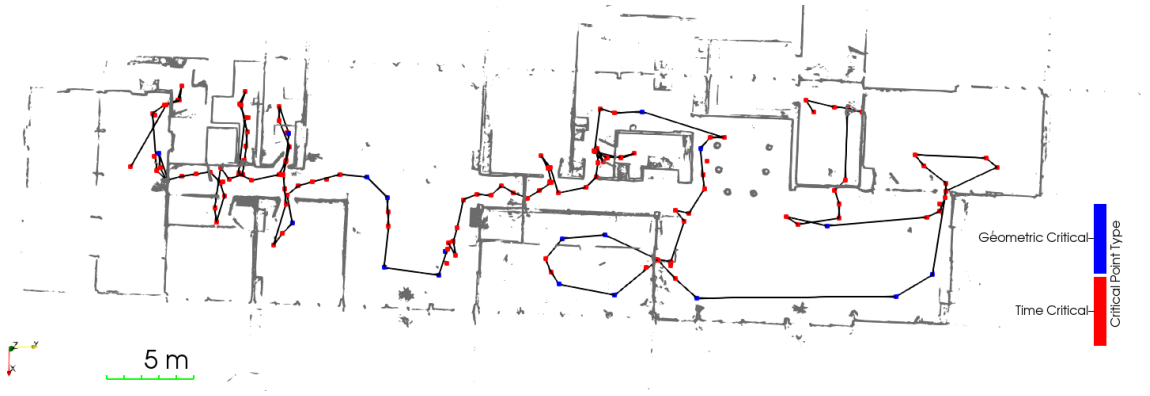



Figure 3.20: Merged critical points for the ZEB1 dataset top floor. Total of 143 critical points compared to 94,416 original points.

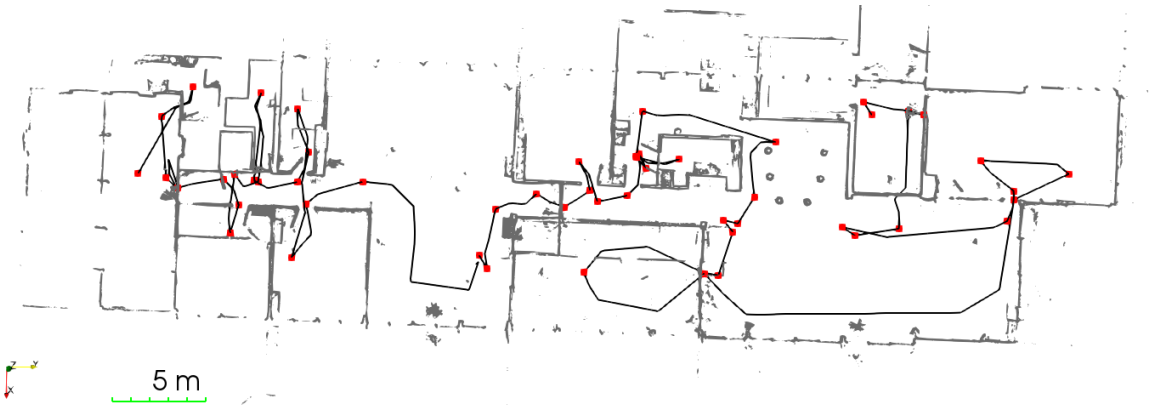



Figure 3.21: Critical points for ZEB1 dataset top floor. Red points represent critical points that are time-based and geometric-based at the same time.

### 3.6.3 Semantic annotation

Based on the results from the previous section, the trajectory was annotated according to the rules defined in Section 3.5. Table 3.3 lists the parameters values used to annotate both datasets. The only difference is the *maximum scanning time for space* parameter which was 45 seconds in case of ZEB-REVO trajectory and 120 seconds for ZEB1 trajectory. Figure 3.22 shows semantic annotations for the basement of ZEB-REVO trajectory, the rooms with two doors were misclassified. For ZEB1 trajectory, Figure 3.23 shows the annotated trajectory of the top floor. One closed loop was misclassified as a room. For the full annotated trajectory Figure A.7 shows the results for ZEB1 system while Figure 3.17 shows results for ZEB-REVO system.

Table 3.3: Parameters used for annotating the trajectory datasets.

| Parameter Name | Parameter Value |
|---|---|
| Nearest neighbor distance for closed loop | 1.0 meter |
| Maximum scanning time for space | 45 seconds (ZEB-REVO)/120 seconds (ZEB1) |
| Maximum critical points inside space | 20 points |
| Minimum trajectory length for a room | 3.20 meters |

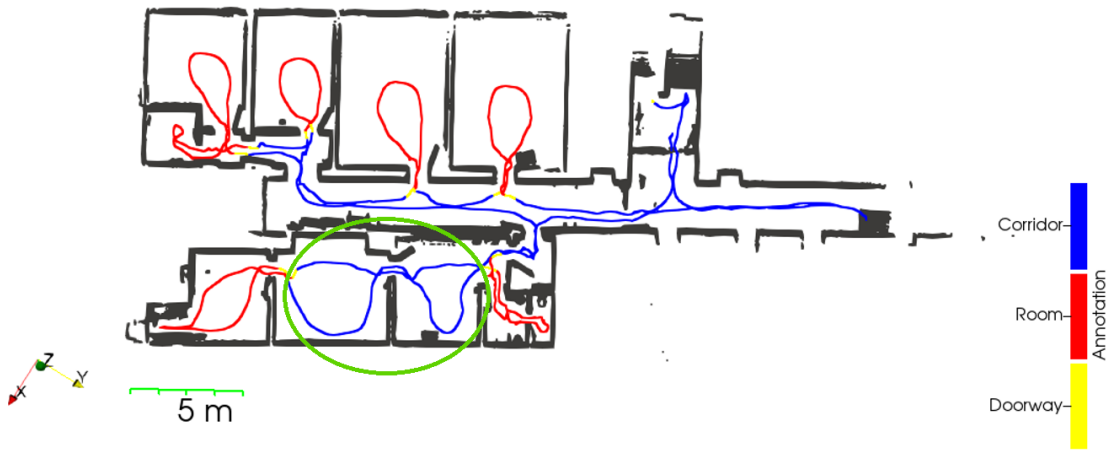



Figure 3.22: Semantic annotation for the basement of ZEB-REVO trajectory. Rooms with two doors were misclassified (green circle).

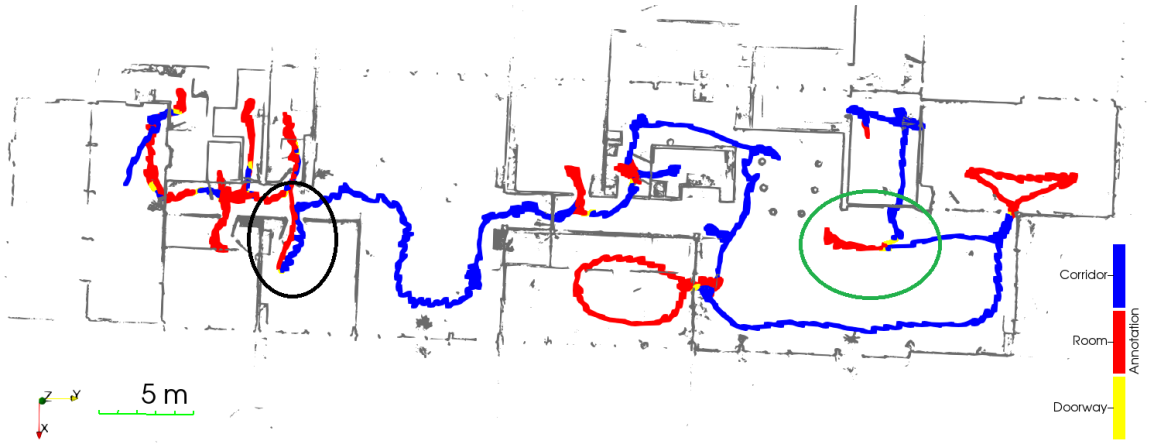



Figure 3.23: Semantic annotation for the top floor of ZEB1 trajectory. Closed loops do not necessary represent a room (green circle). The distribution of the critical points can affect the annotation (black circle).

## 3.7 DISCUSSION

The aim of trajectory analysis process was to enrich the trajectory of a mobile laser scanner with semantic information about indoor spaces such as rooms, doorways, corridors, and stair-

cases. This analysis was conducted based on analyzing the trajectory dataset only without looking at point cloud dataset. The spatiotemporal aspect of the trajectory was considered during the data processing which helped to detect the patterns of movement during the scanning process. These patterns included a change in the movement direction, stops/moves along the trajectory and change of scanning attitude during the scanning process. By combining such patterns, a rule-based model has been defined to characterize different indoor spaces based on the semantic information.

A line growing segmentation technique was developed to segment the trajectory dataset into sub-trajectories represent floor and stair classes. Unlike the histogram-based approaches, this technique considered the time aspect of the trajectory. The activity classes which were segmented in the second step of this approach can be used to split floors and staircases as it represents different time clusters (activities) within one floor. The "*merging height threshold*" illustrated in Table 3.1 is the key criterion to group such activities. It can be adjusted to fit the required process. For example, a lower threshold can reveal any change in the ground level within the floor, while higher threshold can get the whole floor for later processing. Finally, using $SQLite$ database to store the trajectory data gave an advantage for checking the intermediate data and processing all the data in a single file during the whole process. Also, summary statistics can be generated using simple $SQL$ queries.

In the next step, a radial distance simplification algorithm was applied to each floor to bring trajectories from different scanning systems into a similar representation without affecting the trajectory trend. Also, it made the data processing easier. This helped to eliminate the irregular shape of ZEB1 trajectory dataset without affecting its trend. The simplification tolerance can be adjusted according to the processing stage, and the new attributes of the simplified trajectory (if any) can be transferred to the original trajectory using the time attribute. Finally, the other simplification algorithms were used on top of the radial distance method to achieve more trajectory compression and to detect patterns such as geometric-critical points.
In the critical points detection step, different key points were identified based on criteria from the geometric shape, time attributes, and scanner attitude. The Douglas-Peucker algorithm was applied to the simplified trajectory to detect the global change in the movement direction along the trajectory. These key points were used as the main element to identify different rooms. On the other hand, the time difference between each consecutive point along the trajectory was used to detect the change in the operator movement speed along the trajectory. The simplified trajectory helped to make this criterion comparable as the simplified points had an equal distance in between. The time-critical points introduced candidate positions for doorways and tight bends. Based on the results in Figures 3.19 and 3.21, doors could be identified near the time-critical points, but this depends on the operator behavior and the complexity of the indoor environment. If the operator stopped or slowed down near the doors, they can be identified by using suitable time threshold. This can be used to partition the spaces as the doors are the key separator between different spaces. By combing the results from geometric and time based critical points, intersecting patterns about the indoor spaces start to emerge. After defining rules for characterizing the rooms, corridors, and doorways from the key points, the trajectory was annotated. While rooms were identified by closed loops, it was not always a valid case. The indoor environment can be more complex. The rooms can have more than one door as shown in Figure 3.22, and not all the closed loops represent a room as shown in Figure 3.23. The operator behavior besides the scanning system type can influence the annotation process. For example, the scanning time inside a room in the ZEB1 dataset was greater than the one in ZEB-REVO dataset. Moreover, the different working mechanism for both sensors appeared in the trajectory shape which affected the scanner attitude as well. In conclusion, analyzing trajectory data alone was not sufficient to classify the indoor spaces correctly. It was used to separate different floors and staircases. Also, it could help to detect

doorways candidate or information about the space, but joint analysis with the primary dataset of point cloud is required. Therefore, the joint analysis between point cloud and its trajectory is provided in the next chapter.

# 4. Joint Analysis between Point Cloud and Trajectory

## 4.1 OVERVIEW

In this chapter, the point cloud dataset was processed with the help of its corresponding trajectory. The main objective was to label the different spaces inside the environment with semantic information. The first step was to separate the floors. Then, for each floor, the doorways were detected. After that, the trajectory was annotated with the semantic labels. Finally, the trajectory annotations were transferred to the point cloud dataset. The input for this process is the raw point cloud and the segmented trajectory from step 3.2.2. Figure 4.1 shows the pipeline of the joint analysis process. First, the floors were separated based on the segmented trajectory. Then, the doors which are traversed by the trajectory were detected. After that, the trajectory was annotated with the room labels based on the trajectory segments between the detected doors. Finally, the trajectory annotations were transfered to the point cloud by using the timestamp in both datasets. The labeled point cloud was post-processed to refine the labeling results.

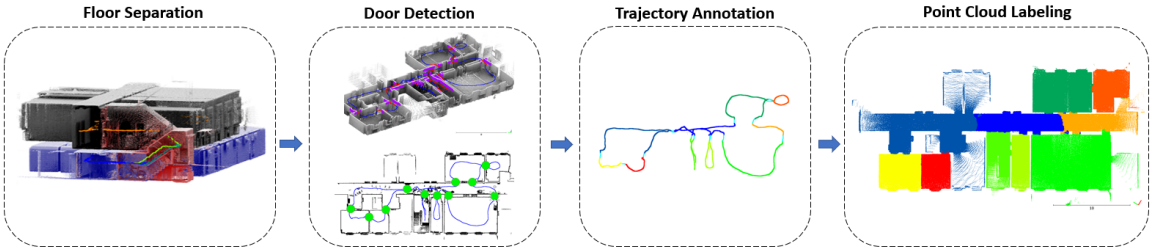



Figure 4.1: The pipeline of the joint analysis between the point cloud and the trajectory: (1) floors separation; (2) door detection; (3) trajectory annotation; (4) point cloud labeling.

## 4.2 FLOOR SEPARATION

The segmented trajectory from step 3.2.2 was used to separate different floors. The point cloud dataset was filtered by time attribute based on the time bounds from the segmented trajectory, as both point cloud dataset and its corresponding trajectory have the timestamp attribute. Each segmented sub-trajectory (global class) had one or more activity class which have different time intervals. These activity classes were used to filter the point cloud into sub-clouds based on the time attribute; then, the sub-clouds were merged to represent floor or staircase and saved for further processing. Figure 4.2 shows a cross-section of separated ZEB-REVO dataset. The trajectory was colored by the global class.

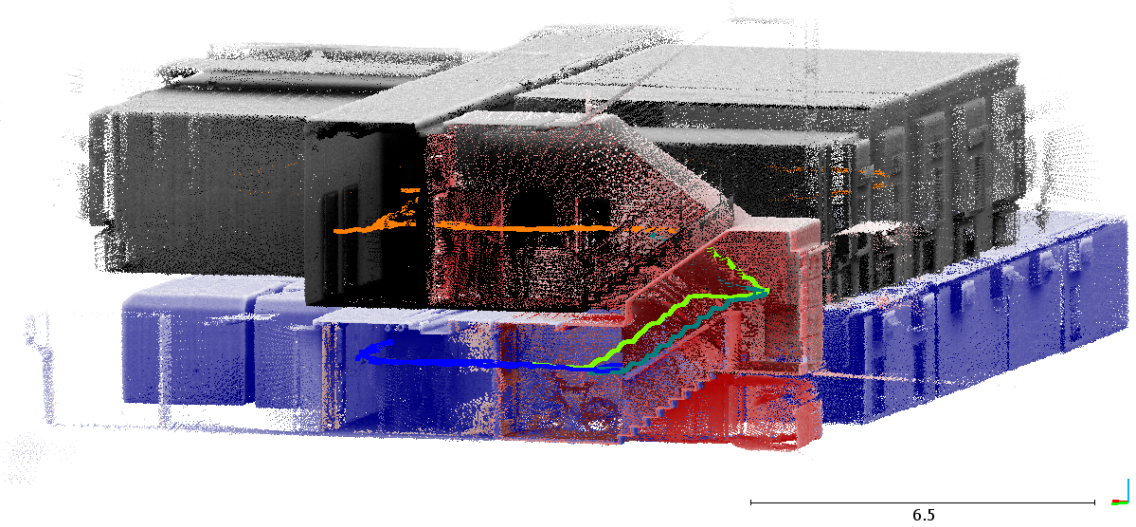



Figure 4.2: Cross section for the separated floors of ZEB-REVO dataset. Each floor has unique color while the segmented trajectory is passing through the environment.

## 4.3 DOOR DETECTION

The first step was to detect the doorways which represent possible doors. Doors are the key elements for partitioning the indoor spaces as they are the transition between different place. Many approaches were developed to detect the doors based on the colored point clouds (Quintana, Prieto, Adán, & Bosché, 2018), but only few approaches utilized the trajectory. As examples of using trajectory to detect the doors, Nikoohemat et al. (2017) utilized the trajectory to detect the door centers based on proximity rules within voxel space. This approach considered opened and closed doors. Also, Díaz-Vilariño et al. (2017) detected the doors by profiling the ceiling and the floor along the trajectory from the point cloud to get the net height. Then, the doors were identified at the trajectory points that have a local minimum of vertical distance. Both approaches detected only the doors traversed during the data acquisition. They may be not efficient in case of low ceiling such as basements where the ceiling drop is minimum as few occupied voxels are required in the first approach, and the vertical profile depend on the variation of the floor height along the trajectory in the second approach.

The proposed method targeted the doorways which represent a possible single leaf door in the case of opened, semi-opened and closed states. The targeted dimensions of the door width and height could be parameterized according to the environment. The process started with identifying the candidate positions for possible doors along the trajectory. This has been done by applying proximity analysis between trajectory and a slice of point cloud parallel to the x-y plane. Then, the point cloud was segmented around each candidate door position to detect the planar patches which represent the walls. After that, the trajectory points were checked against the detected walls to confirm the door width. Finally, the candidate trajectory points were validated against door heights and then clustered to represent the door position. The detailed method is described in the following steps:

a) **Input data preprocessing**
**Trajectory dataset:** The floor trajectory was simplified with a small tolerance of 0.2 meters

to reduce the computation cost.
**Point cloud:** A horizontal slice of height between 0.50 to 0.60 meters around the average height of the trajectory was extracted from the floor point cloud. The slice is represented in purple color in Figure 4.4. The reason behind this was to avoid the clutter which can affect the processing in the following steps as it is expected to have less clutter near this level. Also, the height of the slice was sufficient to segment the point cloud and to decrease the computation time. The slice should be below the door tops to have the door boundary represented by segments which are to the left and right of the trajectory.
An optional step of outliers removal was applied to the highly cluttered floors in case of ZEB1 dataset. Point density of the surface varies according to the surface type. Problems appeared during the proximity and segmentation steps because of sparse points around the surfaces. Statistical outliers removal (SOR) described by (Rusu et al., 2008) was applied to ZEB1 ground floor solve the problem. This method calculates the mean and standard deviation for the distance between each point and its neighbors; then it eliminates the points that have mean distances falling outside a threshold. This threshold is calculated based on the global distances mean and standard deviation. Figure 4.3 shows example of applying SOR on a door sub-cloud in ZEB1 ground floor. The sub-cloud represents a room with glass walls and a glass door.

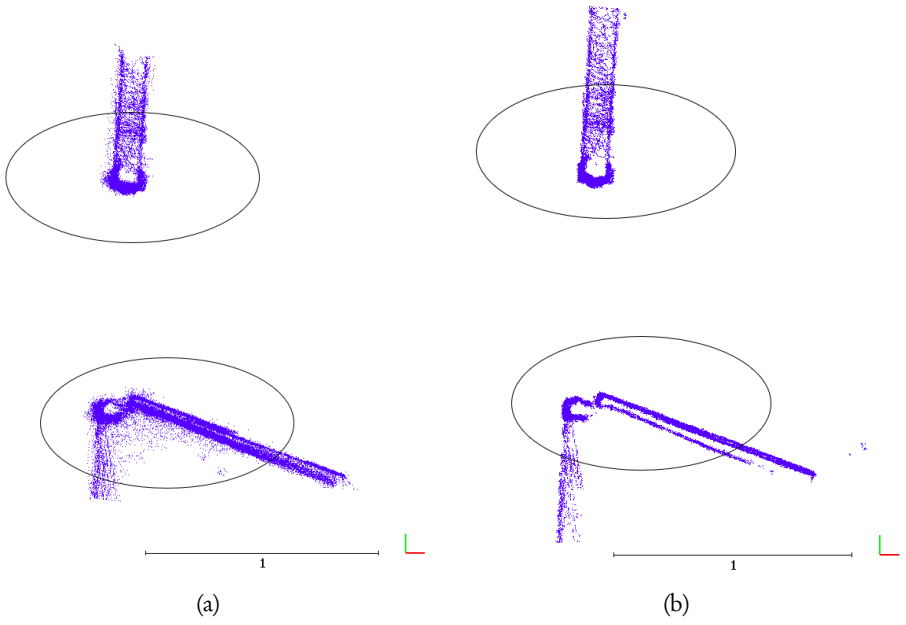



Figure 4.3: Example for using the statistical outliers removal (SOR) on ZEB1 ground floor top view of a sub-cloud. The difference is indicated by the black ellipses; (a) before applying SOR, (b)after applying SOR

b) **Identifying the door spots**
After preparing the input data, nearest neighbor search between each trajectory point and the laser points within point cloud slice was applied to detect the objects that were nearby the trajectory. This include door openings that the trajectory passed through during the scanning. The search process was based on KD-tree data structure, and search radius was parametrized and varied from 0.80 meters to 1.20 meters according to the environment.
Two constraints were applied during the search process to extract the neighbors of each trajectory point:

i) Minimum count of nearby laser points threshold was applied to ensure that the surface was dense enough to represent a wall and to eliminate any temporary objects. This threshold varied according to the floor, and the minimum was 500 points.

ii) Time lag parameter adopted from Nikoohemat et al. (2017) method was used to eliminate any reflected surfaces and moving objects. The laser point was considered a reflected point if the laser point had timestamp more than the timestamp of the trajectory point within a threshold (e.g., 45 seconds). After eliminating the nearby laser points that did not satisfy the threshold, the minimum count of nearby laser points check is reapplied again.

If a trajectory point met the previous criteria, it was flagged as a possible candidate, and the corresponding point indices were saved.
After identifying all the trajectory candidates, the trajectory points were clustered based on the Euclidean distance between points. Each clustered trajectory points represents one or more door candidates which located nearby. Then, the flagged points were transferred to the full trajectory dataset using their time attribute. The corresponding points were sorted, then the duplicated indices were removed, and the sub-cloud surrounding each possible door was retrieved, and saved beside its trajectory. Figure 4.4 shows an overview of the process on the top floor of the ZEB-REVO dataset. Only the trajectory points shown in red and the corresponding point clouds with purple colors are saved for further processing. Each sub-trajectory with its sub-cloud is processed individually in the next process.

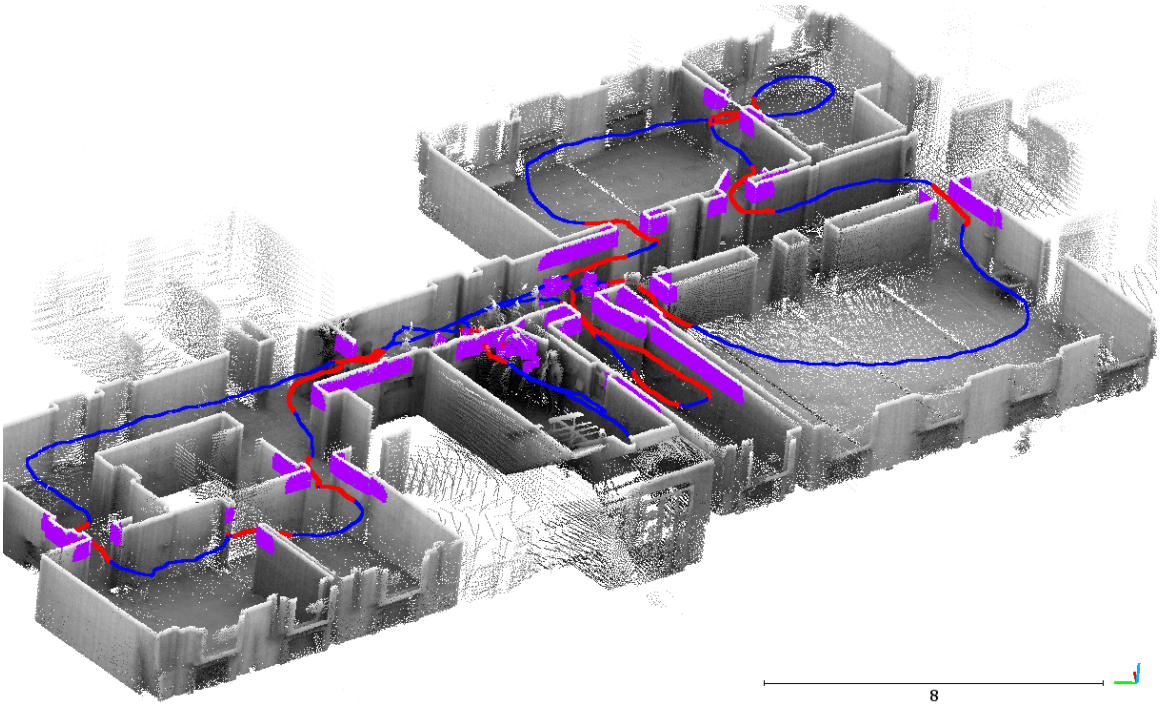



Figure 4.4: The top floor of ZEB-REVO dataset. Candidate trajectory points (red) over the full trajectory (blue). The nearby point cloud points within the slice (purple). The ceiling of the shaded point cloud was removed for visualization purposes.

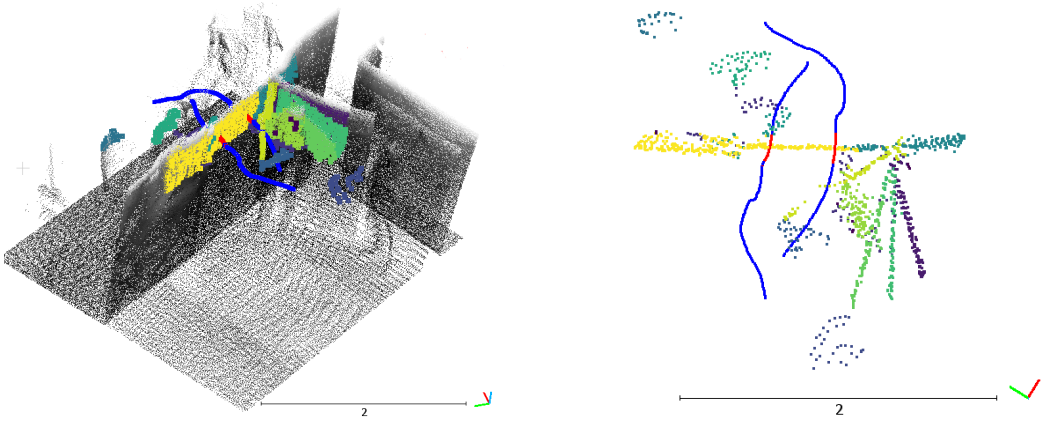
c) **Sub-cloud segmentation**
Each sub-cloud was processed to identify the walls that surround the door and to eliminate

the clutter. Also, this process helped to distinguish the semi-opened door leaves from the surrounding walls by analyzing the segments within the sub-cloud. The door plane was assumed to be a part of a wall surface in case of a closed door, and a gap in an existing wall segment in case of opened and semi-opened doors.
First, surface growing described by Vosselman, Gorte, Sithole, and Rabbani (2004) was applied on the sub-cloud to segment the surfaces. During this step, some issued showed up during selecting the segmentation parameters because the surface was not smooth enough in most cases. The sub-cloud was downsampled to 0.05 meters voxels to optimize the segmentation process. After the segmentation step, the walls were identified according to the following constraints:

i) The surface should be parallel to the vertical direction with some tolerance (e.g., 10 degrees).
ii) The surface should be parallel or perpendicular to the local dominant wall direction. The dominant wall direction was estimated for each sub-cloud as it is not necessary that all the doors be related to the main building direction. The dominant wall direction was assumed to be the direction of the largest vertical plane which has another plane parallel or perpendicular to it within a tolerance of 10 degrees. The normal vector of each plane was estimated, and the dot product was calculated to get the angle between different planes. The output of this step was the vertical walls. The semi-opened door leaves were eliminated besides the small surfaces and the non-vertical surfaces.

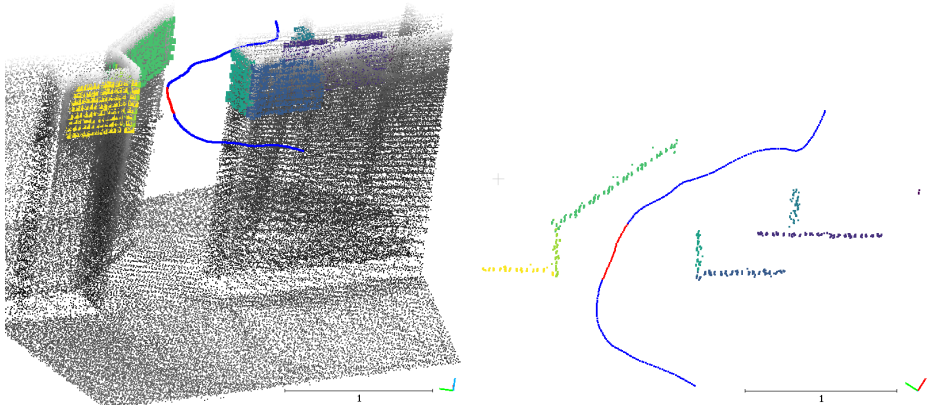
d) **Door width validation**
At this stage, we had the walls or any vertical planes and its nearby trajectory. In this step, the door width was validated using proximity rules between trajectory and the detected walls. First, the trajectory was simplified with 0.01 meters tolerance to decrease the computation cost. Then, nearest neighbor search between the trajectory and the walls which are within a specified distance (e.g., 0.80 meters) were queried. The candidate door trajectory points were refined as the following:
**Closed doors:** The point was flagged as a door if the distance between the trajectory point and the wall is less than a certain threshold (e.g., 0.10 meters). This threshold indicated that the trajectory passed through the door opening and it was closed before or after passing through it. Figure 4.5 shows a closed door sub-cloud. The door candidates in red color were correctly being identified as the segmentation process rules helped to eliminate the small segments and the semi-opened door leaves.
**Opened doors:** The main idea was to check the width of the gap between each pair of walls with the help of the trajectory point.
The nearest point from each wall to the trajectory point was queried. Then, each pair of walls were compared with each other. The trajectory point should be collinear with the two points represent the walls with some tolerance (e.g., 10 degrees). In addition, the trajectory point should be in the middle and the line constructed between the two wall points is within the door width threshold.
If the trajectory point satisfied these conditions, it was checked for the door height.
Figure 4.6 shows a case of a semi-opened door. The segmentation process was applied to detect the vertical surfaces, and to eliminate the semi-opened door leaf by checking each segment against the local dominant wall direction. Also, the door width and height check were applied to get the final door candidates.

e) **Door height validation**
At this step door height for each trajectory points were validated. This helped to eliminate




(a) (b)

Figure 4.5: Example for a case of closed door; sub-trajectory is shown in blue while door points in red. (a)Represents the 3D perspective, and (b)is a top view.

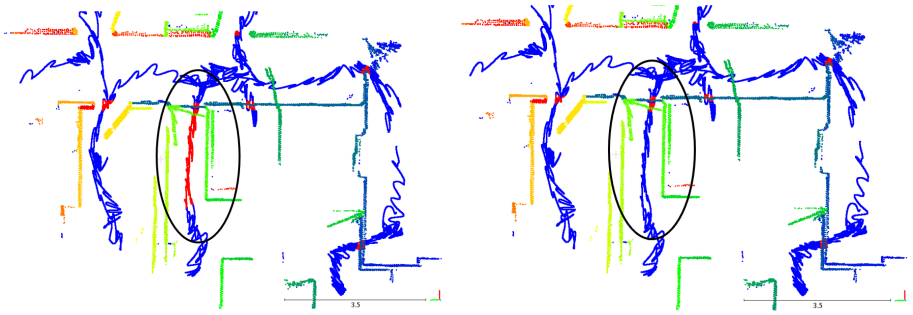



(a) (b)

Figure 4.6: Example for a case of semi-opened door; sub-trajectory is shown in blue while door points in red. a)represents the 3D perspective, and (b)represents the top view.

the candidate door points that were passing through narrow pathways or between high furniture objects such as bookshelves. In this check, the voxel space of the full point cloud was used. A voxel size of 0.05 meters was used. Occupied voxels above and below each trajectory candidate were queried. For the voxels below the trajectory, the last voxel was assumed to be the floor level. Then, net height between the floor level and the voxels above the trajectory point was calculated. The trajectory point was identified as final door candidate if there are occupied voxels at a height within a threshold (e.g., between 1.80 meters and 2.20 meters). Figure 4.7 shows a subset of ZEB1 door cluster. Trajectory points were marked wrongly as door candidates as they were passing through a narrow path. The correct door candidates were marked after checking the door height.

f) **Clustering doors**
The final trajectory candidates were clustered based Euclidean distance to get the trajectory points for each door. The Euclidean distance clustering was used as the door can have trajec-



(a) (b)

Figure 4.7: Top view of using the door height check on ZEB1 dataset a)door candidates based on door width b)door candidates after applying door height check. Trajectory points in blue while door candidates in red. The black ellipse shows the results (a)only with door width validation, (b)after applying door height validation.

tory points represent the path of entering and leaving a space. These points are ordered by time but they can have points of another door in between. Grouping the nearby trajectory points within a distance threshold (e.g., 0.5 meters) helped to identify the points belonging of each door. Also, a threshold for minimum points inside one cluster was defined to eliminate small clusters (e.g., contain less that 5 trajectory points). Then, the full trajectory points belonging to the door were labeled with a unique identifier in the $SQLite$ database.

## 4.4 TRAJECTORY ANNOTATION

This process aimed to annotate the trajectory segments with the semantic information after getting the door positions. Doors were considered the main element to split the different spaces. The full trajectory was queried from the $SQLite$ trajectory table without the door parts. Then, the nearby trajectory points were segmented together based on the Euclidean distance within a small threshold (e.g., 0.2 meters). The Euclidean distance clustering was used to group the sub-trajectories within a space like a corridor in one class. Figure 4.8 and Figure 4.9 show the annotated trajectories of ZEB-REVO and ZEB1 datasets respectively.

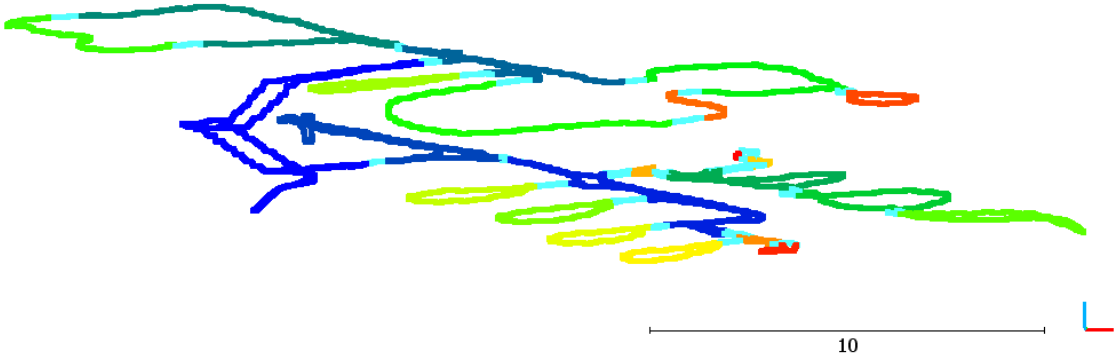



Figure 4.8: The annotated trajectory of ZEB-REVO dataset. Each sub-space has unique color while the detected doorways are colored in cyan.

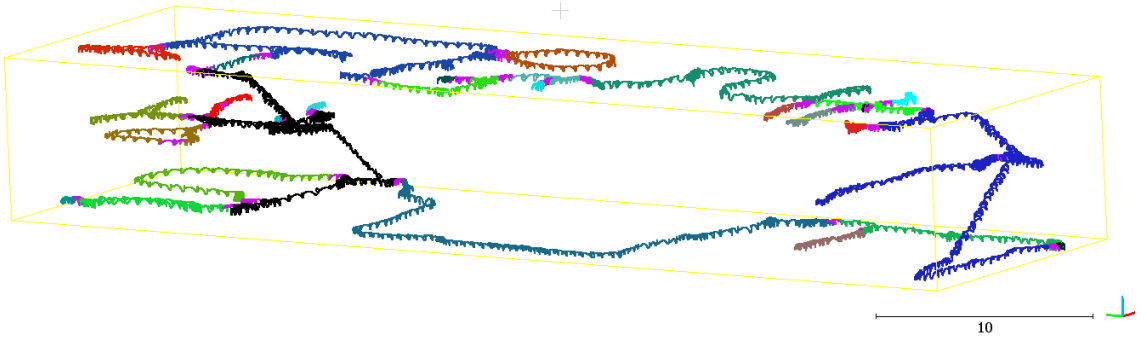



Figure 4.9: The annotated trajectory of ZEB1 dataset. Each sub-space has unique color while the doorways are colored in purple.

There was an expansion tolerance for each door sub-trajectory to increase its length while saving it to the database. The length of the trajectory representing a door could be as small as 0.05 meters. This small segments affected the clustering of the spaces sub-trajectories in the trajectory annotation step as there was not enough separation distance between them. This tolerance helped to have a clear separation for the trajectory belonging to different spaces. This expansion tolerance was applied by adding 0.5 seconds at the beginning and the end of each door trajectory in case of ZEB-REVO dataset and 1.0 seconds in case of ZEB1 dataset.

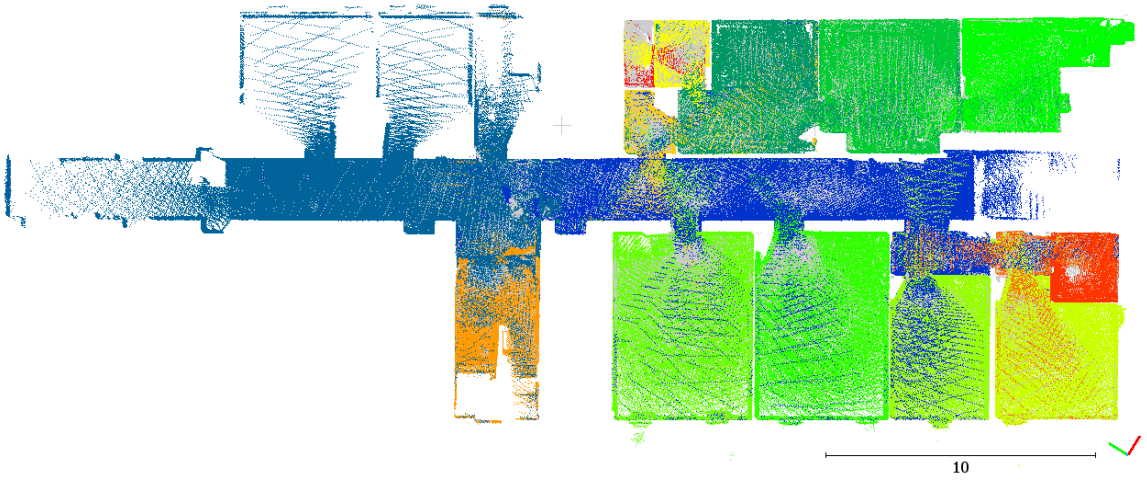
## 4.5 POINT CLOUD LABELING

In the previous step, the trajectory was annotated with the semantic classes. Time attribute was used to transfer the labels from the trajectory to the point cloud. However, there was an overlap between the classes inside each room as shown in Figure 4.10(a). That is because some parts of the space were scanned from another space through the opened doors or the glass walls. Also, there were unlabeled points which were scanned while passing the doorways.

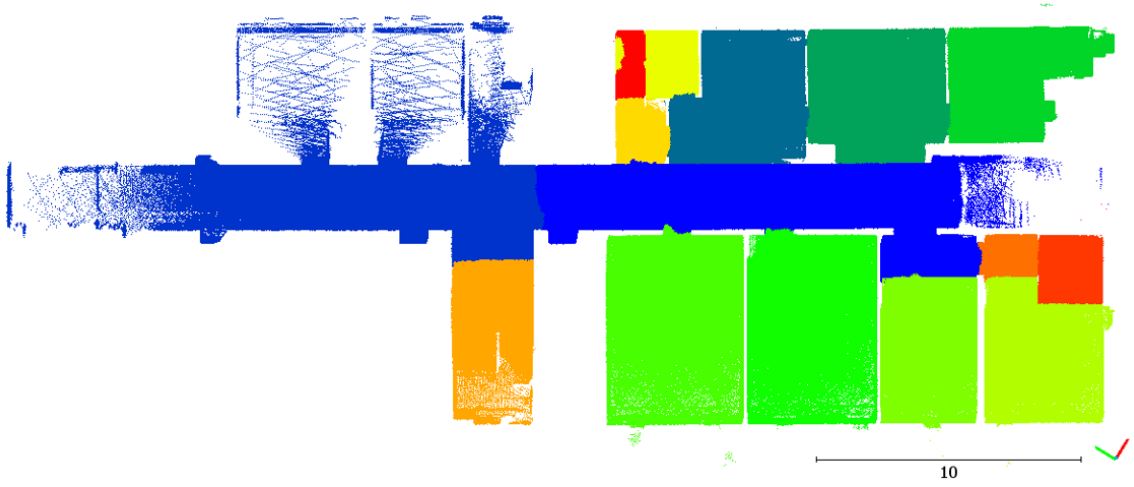
Majority filter was applied to the labeled point cloud of each floor to refine the classification results. All points inside one space should have one class; therefore, the problem was simplified using a 2D approach. The process was applied as the following:

- All the points within the 3D point cloud were projected into the x-y plane.
- A quadtree was constructed over the projected points with 0.05 meters cell size.
- The occupied cells were labeled with the class of the majority of the laser points inside.
- The occupied cells that were within a distance threshold (e.g., 0.3 meters) from a trajectory point was assigned to the same class as the trajectory point.
- A post-processing check was applied for each cell to consider its direct neighboring cells. The class of the majority of neighbors was assigned to the cell.
- Finally, the labels were transferred back to the original point cloud.

The post-processing step was applied because there were sparse cells that have a different label that its neighboring cell. Only one iteration for neighboring cells was applied to the occupied cells. The kernel size of 8 neighbors and 24 neighbors were tested in this step. An example of the majority filter process as shown in figure 4.10(b).




(a)




(b)

Figure 4.10: Top view for a floor of ZEB-REVO dataset. (a)Time-based labeling. The gray spots represent the unclassified points. The ceiling was removed for visualization purpose. (b) The labeled point cloud after applying the majority filter. Each space has unique color based on the corresponding sub-trajectory clusters.

## 4.6 EXPERIMENTAL RESULTS

In this section, the key results of the joint analysis process are illustrated. The primary objective was to label the point cloud dataset with semantic information about the indoor spaces. The process started with separating the floors; then, detecting the doors on each floor. After that, the trajectory was annotated to represent the possible spaces. Finally, each floor was labeled with the semantic information. The two datasets described in section 1.3.1 were used for all experiments. The floor separation results are illustrated in section 4.6.1. Door detection results are shown in section 4.6.2, while the point cloud labeling results are illustrated in section 4.6.3.

### 4.6.1 Floor separation

This section illustrates the results for floor separation using the segmented trajectory. Figure 4.11 shows an overview of the separated ZEB-REVO dataset. Figures 4.12 and 4.13 show one of

the floors and the staircases respectively. There were points scanned from another separated space as the separation was based on the timestamp. On the other hand, Figure 4.14 shows the top and ground floor of the ZEB1 dataset without the intermediate floor and the stairs. The staircases and one of the intermediate floors are shown separately in Figures 4.15 and 4.16 respectively. There was an overlap between the separated spaces as the walls in between were made of transparent material.

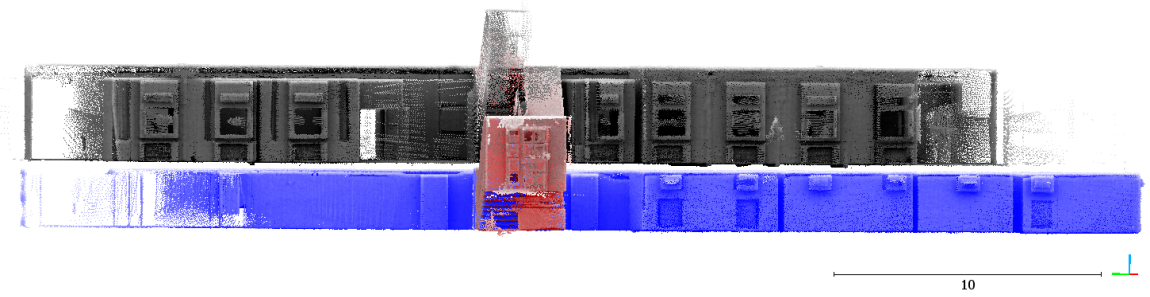



Figure 4.11: Separated floors of ZEB-REVO dataset shaded in different colors.

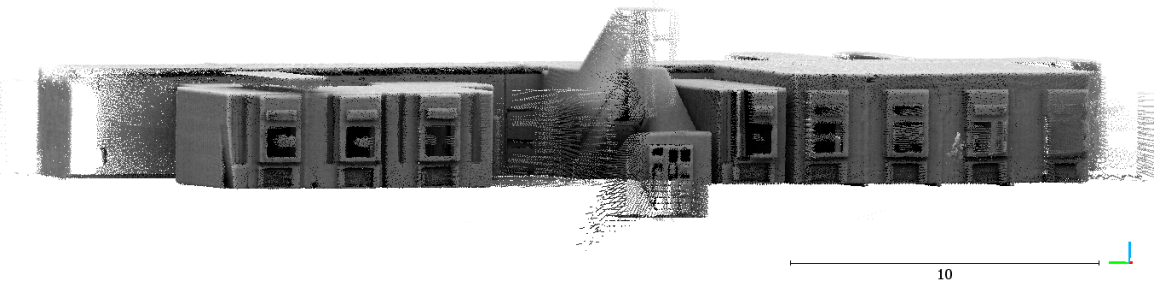



Figure 4.12: Separated top floor of ZEB-REVO dataset

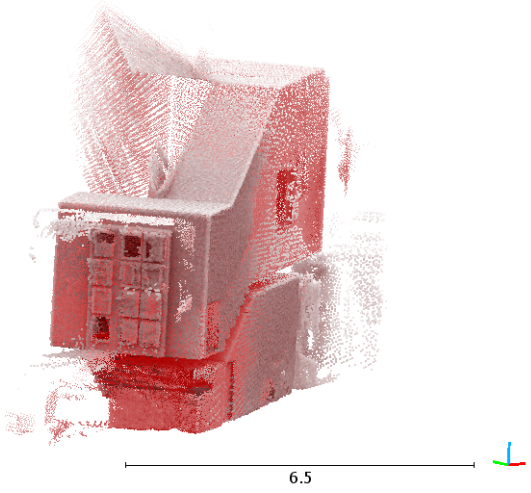



Figure 4.13: Separated staircase of ZEB-REVO dataset

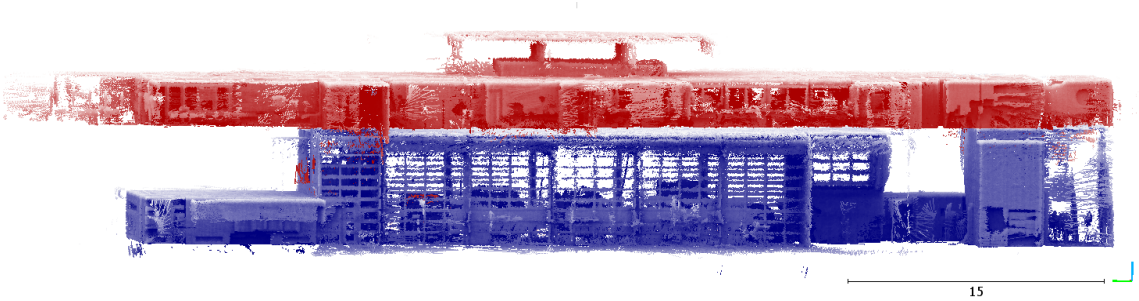



Figure 4.14: Ground and top floor of ZEB1 dataset shaded in different colors. The intermediate floor and the staircases were removed to show the separation between the floors.

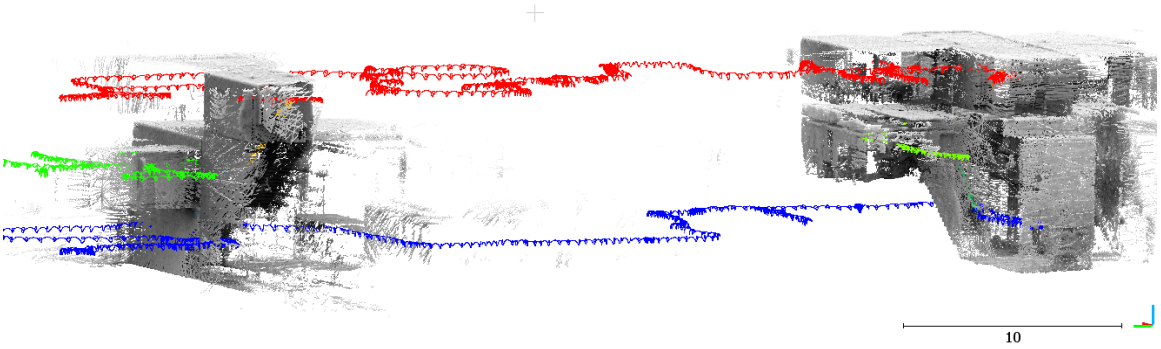



Figure 4.15: Staircases sub-cloud with the trajectory colored with the global class.

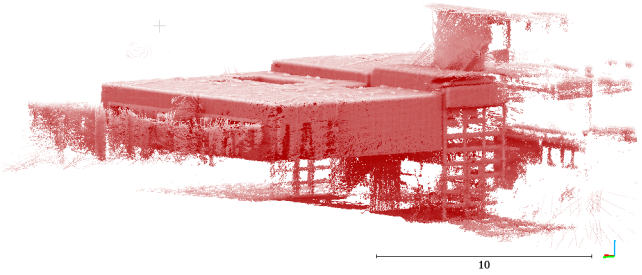



Figure 4.16: The left intermediate floor of ZEB1 dataset shaded in different colors. There is overlap between the staircase and the floor as there are glass walls.

### 4.6.2 Door detection

In this section, the door detection results are illustrated for ZEB1 and ZEB-REVO floors.

#### ZEB-REVO dataset

Figure 4.17 shows the detected doors for the the first floor.All the 11 doors were detected. Figure 4.18 shows the detected doors of the basement. There were 14 doors crossed by the operator during the scanning process. All the doors were detected. Only door $D6$ was represented with

more trajectory points. This was due to a problem in verifying the door width and height in the small spaces.

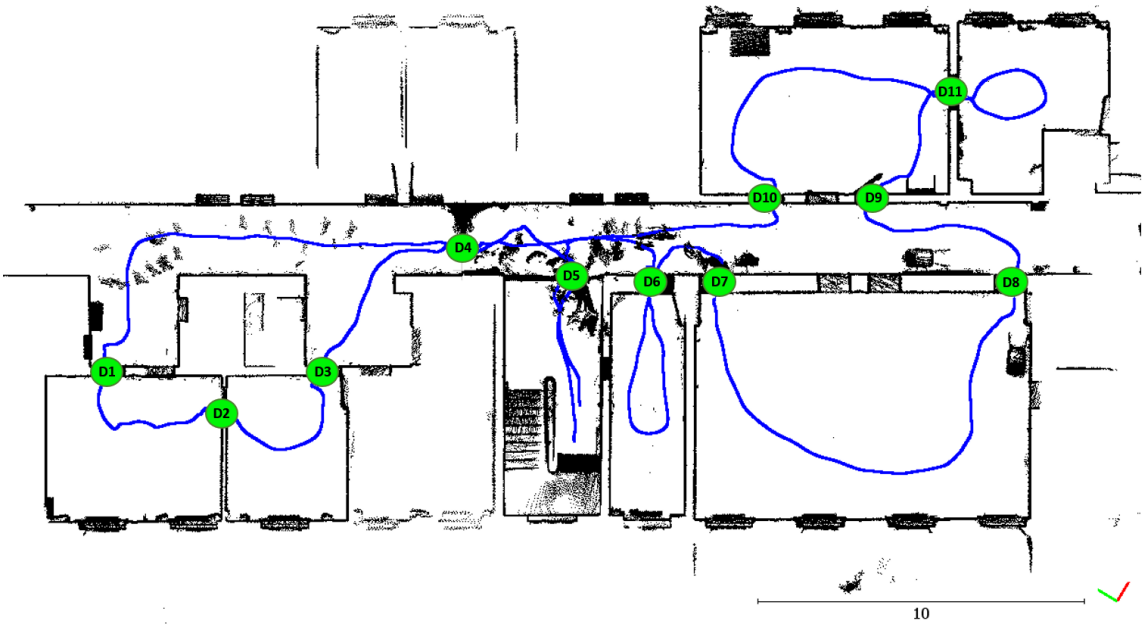



Figure 4.17: Correctly detected doors of the ZEB-REVO top floor shown in green circles.

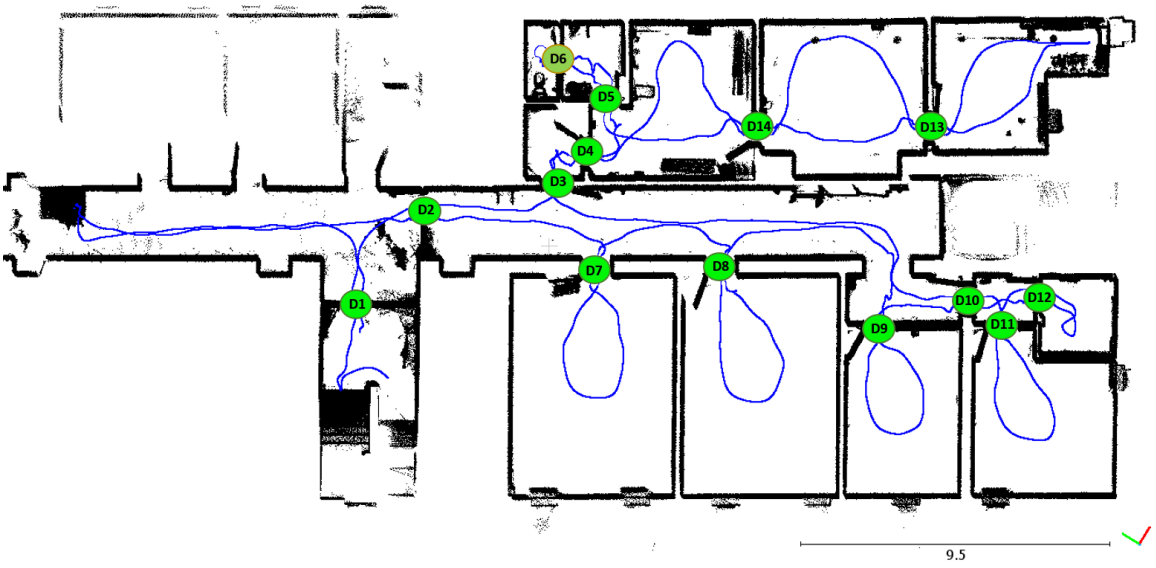



Figure 4.18: Detected doors of ZEB-REVO basement. All traversed doors were detected correctly, but D6 shown in olive had long trajectory segment.

#### ZEB1 dataset

Figure 4.19 shows the detected doors for the top floor. In total, there were 17 doors traversed by the operator during the scanning process. Door *D*4 was not detected due to incorrect estimation

for the floor level. The method was identifying the last voxel under the trajectory as the floor level; but in this case, there was part stairs boundary below the floor. Besides the two undetected doors, door $D1$ was correctly detected but not according to the correct rule. This door was a sliding door, and the boundary was not detected, but the trajectory was near one of the sides with distance less than 0.10 meters. The door was identified as a closed door which was not correct. Figure 4.20 shows the detected doors for the ground floor. All the 8 doors were detected successfully. Finally, Figure 4.21 shows the detected doors for the intermediate floors of ZEB1 dataset. There were two false doors shown in figure 4.21(b) due to the closed doors issue which was previously mentioned. The trajectory points were near a wall with less than 0.10 meters, and they were identified as closed doors, but those false doors did not affect the labeling process as the trajectory was clustered as one space in step 4.4.


Figure 4.19: The top floor of ZEB1 dataset, the correctly detected doors are shown in green circles. Red circles represent the undetected doors.

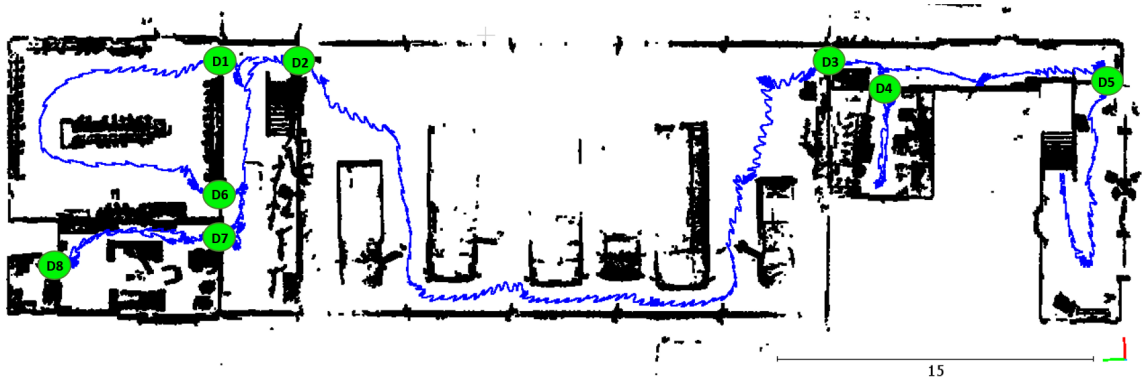
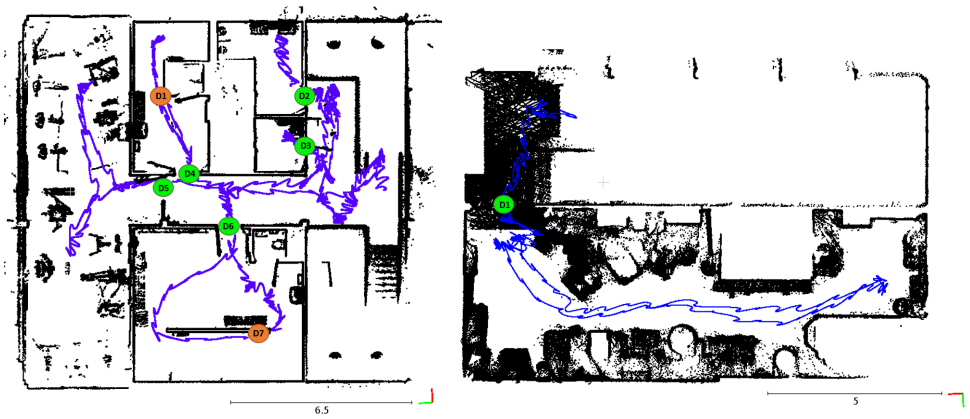



Figure 4.20: The ground floor of ZEB1 dataset, all traversed doors were detected correctly.




(a) (b)

Figure 4.21: Detected doors of the ZEB1 intermediate floor; (a) the correctly detected doors are shown in green circles. Orange circles represent the false detected doors. (b) All the traversed doors were detected correctly.

Table 4.1 lists the common parameters values used in the door detection process. These parameters were modified for some floors to adapt the environment as the level of clutter varied from floor to another and doorways width had variations. The parameter of *neighbors search radius* for the ZEB1 top floor was 1.20 meters instead of 0.80 meters to get more point cloud points to enhance the wall segmentation process. Also, *minimum neighboring points* for the ZEB1 dataset was increased to 5000 points instead of 500 points as the dataset had high level of clutter. Increasing the minimum number of neighboring points threshold helped to eliminate some reflected surfaces. Finally, *maximum door width* threshold for the ZEB-REVO top floor was increased to be 1.2 meters to detect all the doorways.

Table 4.1: Common parameter values used in the door detection process.

| Algorithm | Parameter | Value |
|---|---|---|
| **Identifying the door spots** | | |
| | Trajectory simplification tolerance | 0.20 meters |
| | Neighbors search radius | 0.80 meters |
| | Minimum neighboring points | 500 points |
| | Time lag for reflected surfaces | 45 seconds |
| **Door detection** | | |
| | Search radius between candidate trajectory points and a wall surface | 0.80 meters |
| | Closed door distance threshold | 0.1 meters |
| | Maximum Door width | 1.1 meters |
| | Minimum door width | 0.50 meters |
| | Door height threshold | 1.70 to 2.20 meters |
| | Door cluster tolerance | 0.50 meters |
| | Minimum trajectory points per door | 3 points |

Table 4.2: Common parameter values for segmentation process

| Algorithm | Parameter | Value |
|---|---|---|
| **Point cloud segmentation** | | |
| | Point cloud downsampling voxel size | 0.05 meters |
| | Trajectory simplification tolerance | 0.01 meters |
| **Surface growing** | | |
| | Seed neighborhood radius | 0.30 meters |
| | Growing search radius | 0.15 meters |
| | Maximum distance of point to seed | 0.10 meters |
| | Maximum distance of point to surface | 0.08 meters |
| | Minimum distance to recompute surface | 0.05 meters |

#### 4.6.3 Point cloud labeling

In this section, the results of the point cloud labeling is illustrated. These results represent the final output of this chapter and the main objective of this research.

##### ZEB-REVO dataset

Figure 4.22 shows the top view of the labeled basement of the ZEB-REVO dataset. There was a mixed class in a small space due to an incorrect trajectory shape for the detected door. On the other hand, the top floor which is shown in Figure 4.23 has a clean separation between the spaces with a small overlap at the doorways. The corridor was partitioned into two spaces with a door in between. However, one of the corridor spaces was labeled into two classes as it had two different trajectory clusters. This oversegmenataion was influenced by the operator behavior as the space was not fully covered by the trajectory which was used to identify the spaces.

##### ZEB1 dataset

This dataset has high level of clutter and reflected surfaces which affected the results. Figure 4.24 shows a top view of the labeled point cloud of the ground floor. There was a problem of mixed labels around the doors. Also, the glass walls between different spaces had poor separation of labels. The top floor which is shown in Figure 4.25 had different labeling problems. The rooms which were surrounded by one or many glass walls had mixed labels, as there were reflecting surfaces within the space and some parts were scanned from the surrounding spaces. Also, there was one space which was not labeled separately due to the undetected door. This space is surrounded in the figure by the black ellipse. Also, there were mixed labels near the doorways which was a common problem between all the floors. Finally, the elongated spaces were oversegmented as they were not fully covered by trajectory points. The intermediate floor results are illustrated in Figure 4.26.

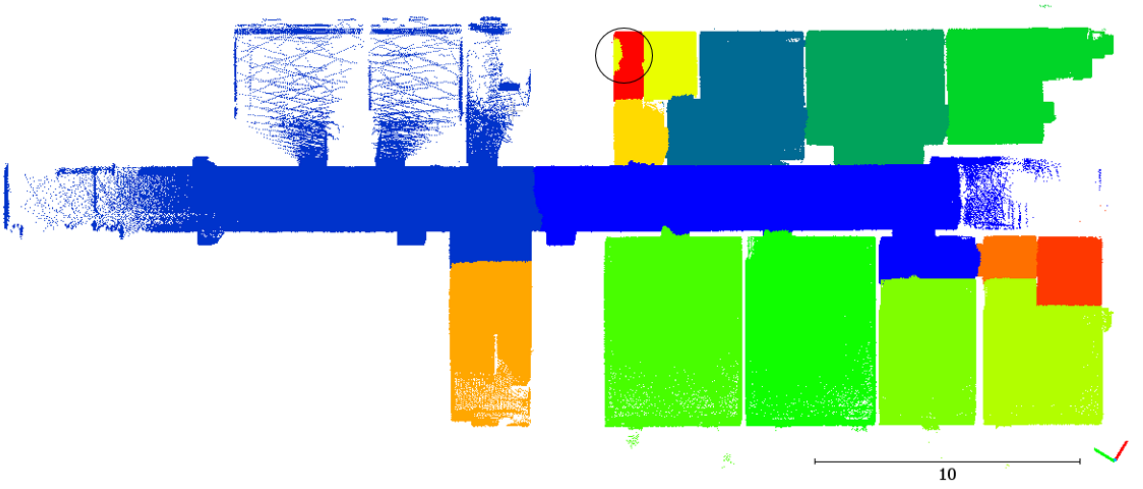



Figure 4.22: Top view for the basement ZEB-REVO dataset. The black circle shows mixed labels in one of the spaces due to door detection problem.

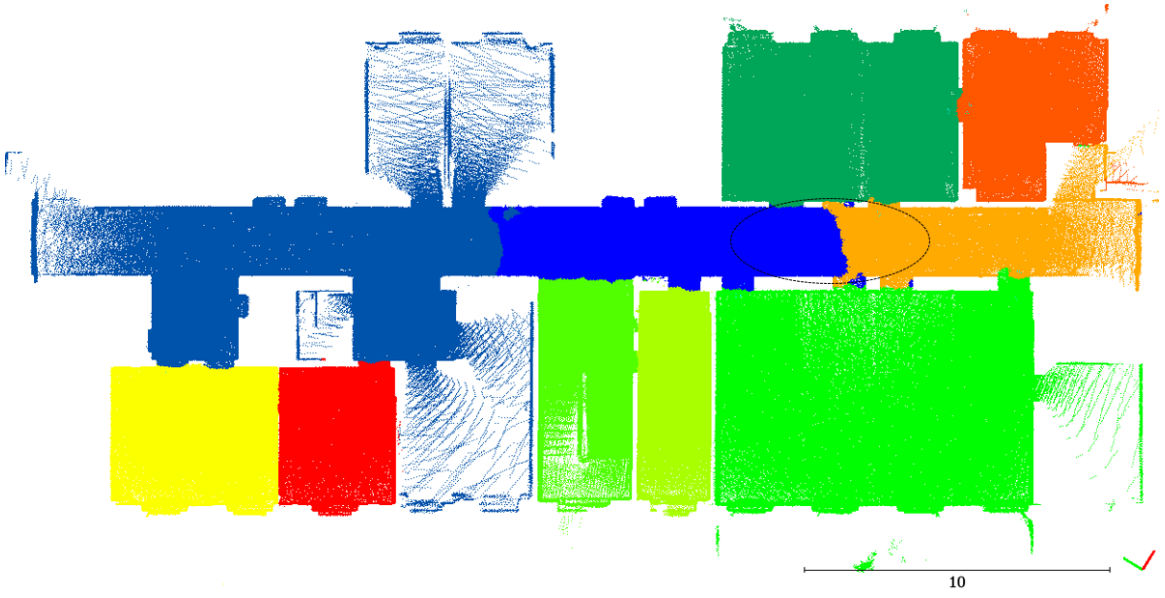



Figure 4.23: Top view for the top floor of ZEB-REVO dataset. The black ellipse shows oversegmentation in one of the spaces as it was not fully covered by trajectory points.

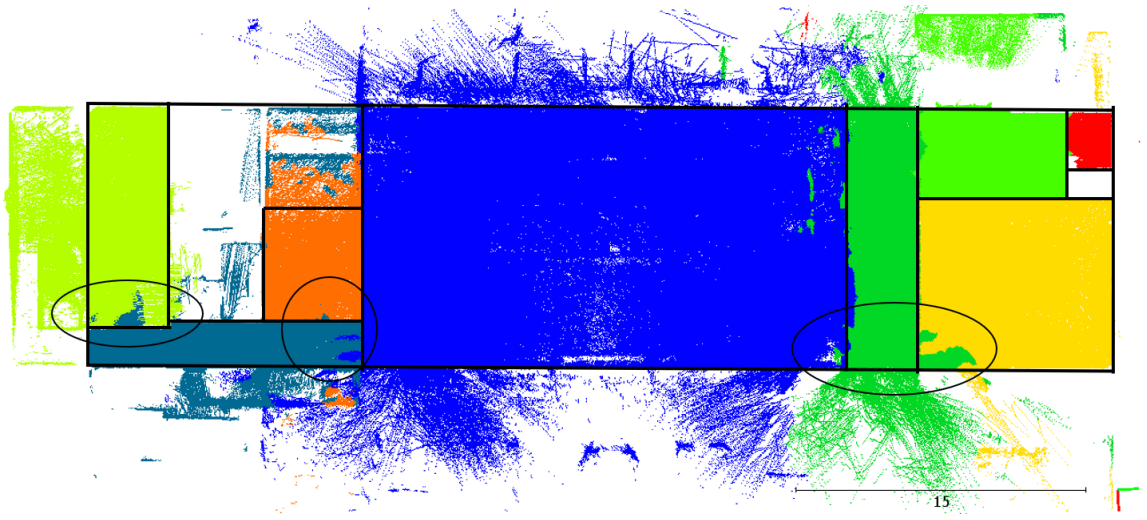



Figure 4.24: Top view for the ground floor of ZEB1 dataset. The black ellipses show mixed labels near the doorways.

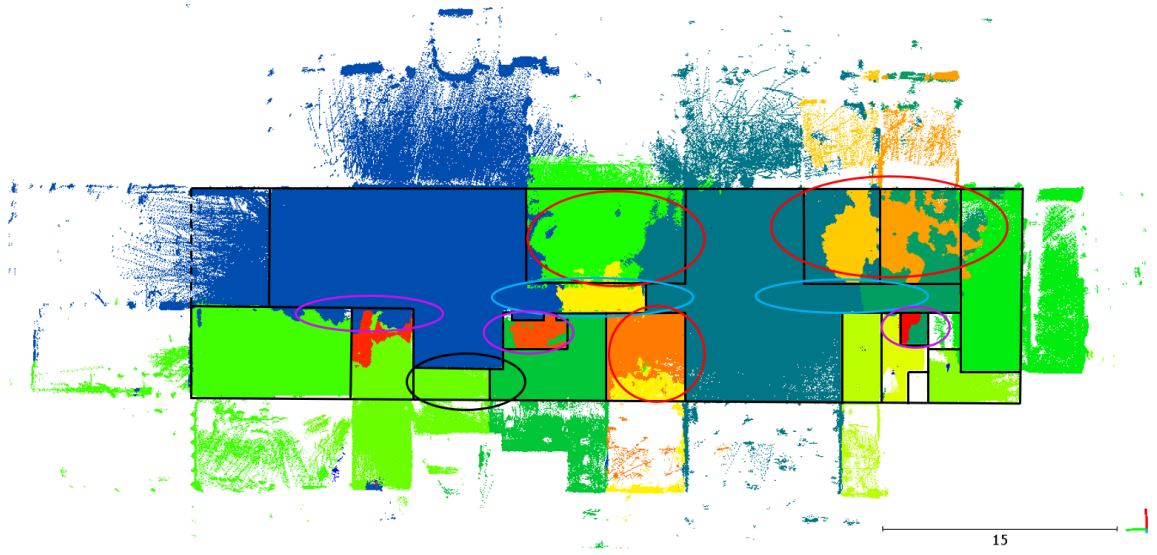
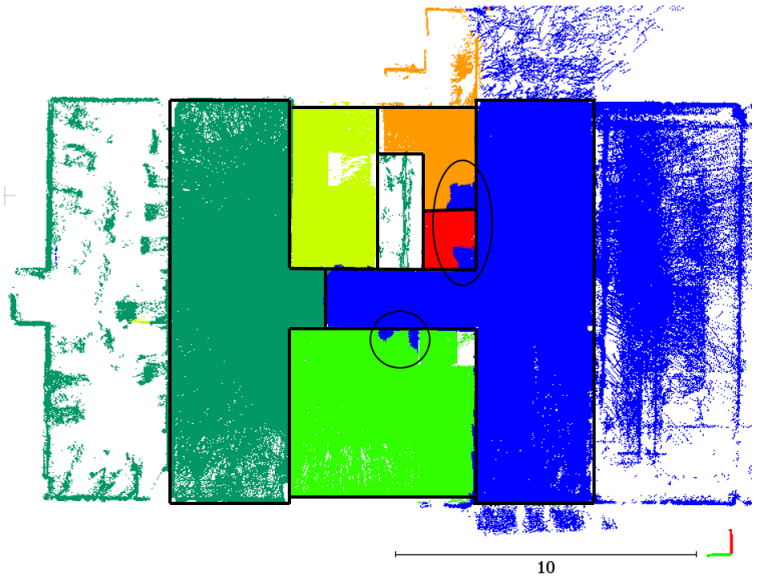
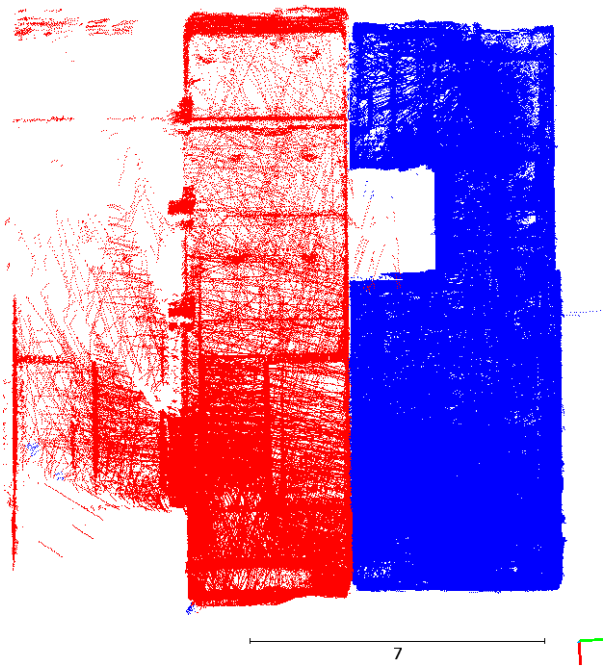



Figure 4.25: Top view for the top floor of ZEB1 dataset. The black ellipses show the misclassified spaces due to undetected doors. The cyan ellipses show the oversegmented spaces. The red ellipses show mixed labels inside the spaces due to the glass walls. The purple ellipses show the mixed some overlap between spaces around the doorways.




(a)




(b)

Figure 4.26: Top view for the labeled intermediate floor of ZEB1 dataset. a) Black circles show mixed labels around the doors. b) Two spaces have only one door in between.

## 4.7 DISCUSSION

The main aim of the joint analysis process is to enrich the point cloud and its corresponding trajectory with semantic labels about the indoor spaces such as floors, staircases, doorways, and rooms. The segmented trajectory is used to split the floors of the point cloud dataset based on the time attribute. This approach worked well in the environments which have different ceiling

heights, and when relying on the height histogram-based method of the point cloud is not enough. Nevertheless, post-processing may be needed after separating the floors, as splitting based on time attribute can show overlap between another floor as shown in Figure 4.12 and Figure 4.15.

In the following step, the doors are detected as an intermediate step towards labeling the point cloud and the trajectory. Using the average height of trajectory to slice the point cloud increases the processing speed as we do not need to calculate the floor level. Also, it helps to avoid the clutter by having the slice of the point cloud at height just below the beams level. However, this approach will not work properly if there is a big variation in the floor level or the trajectory height is relatively low in the case of trolley-based scanning systems. The solution in case of the difference in the floor levels is to utilize activity classes from step 3.2.2 to have the slice for each average height. On the other hand, the slice height can be parametrized by adding a fixed value to the trajectory average height to overcome the low height of the trajectory in some scanning systems. For identifying the door spots, the choice of the threshold value for the minimum neighboring laser points for each trajectory point can affect the results in terms of clutter removal. For example, this value is 500 points in case of ZEB-REVO dataset, and it is increased to 5000 points in case of the ZEB1 top floor which has a high level of clutter and reflected surfaces. The choice of this parameter is based on running experiments as it takes only one minute to process a floor with around 33.5 million points. The developed approach detected all the closed doors but relying on the intersection between the trajectory and the walls is not sufficient. Two false doors are detected as illustrated in Figure 4.21(a), and this is also due to the complex shape of the ZEB1 trajectory which can be near the surfaces with distance less than 0.10 meters. To avoid the false doors problem, the trajectory should be checked to be passing through the surface and not only the intersection. Another limitation for the door detection method appeared while applying the rules for the door width and height inside narrow spaces where their dimensions can match a door width and height constraints as illustrated in Figure 4.22 and Figure B.1. To avoid this problem, the algorithm needs to be optimized to check the door height locally instead of using a door height range. The last possible failure case for the door detection algorithm will be while checking the semi-opened doors. All the semi-opened doors are detected by identifying the dominant wall direction locally for each door cluster. The check will fail if the opened door leaf is perpendicular to one of the walls and there is an intersection between the trajectory and the door. The operator behavior can help to avoid all the door detection problem if all the doors are opened before starting the scanning process. Also, the operator can avoid getting very close to the walls. Finally, following the scanning guidelines to scan the doors slowly can increase the point density and enhance the point cloud segmentation results.

The trajectory candidates from step 4.3 can be considered as a new concept of critical points which is based on the proximity between the trajectory data and the point cloud data. It is more reliable than the combination of geometric and time-critical points which are explained in the previous chapter regarding identifying the separation between the different spaces. The doors are detected first then, the different rooms are labeled, but in the first approach which involved the trajectory only, the rooms are detected; then, the door positions are inferred. Also, in the first approach, the operator behavior have more influence on the trajectory annotation results

The annotated trajectory is used to label the point cloud by transferring the trajectory labels to the point cloud using the time attribute. There are mixed labels inside each space, and this was very clear in case of the glass walls as shown in Figure B.2. The concept of the majority filtering is used to refine the labels within each space. However, the labeling method showed poor performance in case of the glass walls as illustrated in Figure 4.25 relying only on the timestamp attribute is not sufficient in this case even after applying the majority filter. Constructing the boundary for each space can be applied to confirm the labels inside each space, but this step was out of the scope of this study. Also, there is a problem of the over-segmented spaces shown in Figure 4.23 as the

trajectory did not fully cover them. This problem can be solved in the further post-processing steps as mentioned by Díaz-Vilariño et al. (2017) and Mura et al. (2014).One advantage of using the trajectory is the ability to change to operator behavior to walk in multiple paths during the scanning process to cover the elongated and big spaces.
In conclusion, the proposed algorithm had better results for ZEB-REVO dataset than the ZEB1 dataset. The results varied due to the low level of clutter for the first dataset, the shape complexity of the trajectory of the ZEB1 system, the operator behavior during the scanning process, and type of the surface material inside the environment.

# 5. Conclusions and Recommendations

In this research, a method for enriching the mobile laser scanning point clouds and trajectories with information about the indoor spaces is introduced. The trajectory dataset is used as a source of information to optimize the point cloud processing. By processing the trajectory first, the trajectory gives the ability to analyze the point cloud dataset as small parts without the need to downsample the full point cloud which prevents the loss of information. Also, annotating the trajectory with information such as the floors and the staircases helps to process the point cloud for a specific purpose such as detecting the doors or labeling the point cloud without the need to perform the full analysis every time. The trajectory is used for separating the floors, detecting the staircases, and detecting the doors which are traversed during the scanning process. Also, the point cloud is labeled with the rooms and staircases information with the help of the trajectory. The experiments on the ZEB-REVO and ZEB1 show that the proposed methods can work for different environments with minimum constraints as both datasets have different levels of clutter and trajectory shape. The limitation of this method appears in the labeling process within an environment that has glass walls. Also, the operator behavior can influence the results of the labeling process.

## 5.1 RESEARCH QUESTIONS: ANSWERED

**Specific objective 1:** To annotate the trajectory dataset with semantic information about the indoor spaces by analyzing the trajectory data only.

- **What is the optimal method for trajectory simplification?**The radial distance algorithm is used to simplify the trajectory for both ZEB1 and ZEB-REVO datasets. The trajectory datasets are represented by dense points as the scanning systems have a scanning rate of 100Hz. Given that both trajectories have different geometric complexity, the radial distance algorithm is a good choice as it is affected more by point density. It helps to eliminate the redundant points and simplify the geometric shape without losing the information. Moreover, by simplifying the trajectory by equal distance criterion, the simplified segments can be compared with each other using the time attribute and also it is used as a base for detecting the geometric-based critical points.

- **What are the parameters that describe the critical points?** There are three parameters which are used to identify the critical points. The first parameter is the radial distance simplification tolerance which is used to get the simplified trajectory dataset. The choice of the values of the other two parameters depends on the selected simplification tolerance. The second parameter is the simplification tolerance of the Douglas-Peucker algorithm. The algorithm is applied to the simplified dataset to get the geometrically critical points, which globally preserve the trend and the geometric shape of the trajectory. The third parameter is the difference in the timestamp between two consecutive simplified points. It is used to identify the time-based critical which are used to detect the change in operator speed along the trajectory which can reveal the bends and door positions in the environment if the operator followed the scanning guidelines.

- **What are the classes that could be identified from the trajectory data only?** Floors and staircases can be separated using the trajectory segmentation algorithm. For each floor, candidates of doorways can be detected based on critical points and comparing the neighboring classes. Also, corridors and rooms can be identified, but the results showed that there are misclassified segments. Joint analysis with point cloud is necessary for the complex indoor environments.

- **How can these classes be described based on the semantic rules**? Floor segments are identified as a group of sub-trajectories that are connected and have the same average height within a small threshold. The staircases class is identified to be the sub-trajectories that connect the floors and have a large variation of height. Within each floor, corridors are assumed to be parallel and perpendicular to the main building direction. Rooms are detected by closed loops within a time threshold. Finally, doorways are the intermediate link between these classes.

**Specific objective 2:**To label the point cloud based on semantic information from both trajectory and point cloud.

- **How can the point cloud be prepared for joint analysis process?** The segmented trajectory is used to split the point cloud dataset into floors and staircases. Then, each floor is analyzed individually with its corresponding trajectory. The separation process helps to reduce the computation cost and complexity as each floor can have different analysis parameters during the processing. These parameters depend on the level of clutter on the floor and the trajectory shape.

- **How can the spaces be partitioned?** Identifying the doorways is the key for partitioning the spaces as the doorway is considered the transition element between different spaces. Based on the detected doors, the trajectory dataset is annotated with labels about different spaces and these labels are transferred to the point cloud dataset.

- **How to detect the doors with the help of the trajectory?** The trajectory is utilized to detect the doors that were traversed during the data acquisition. A constrained nearest neighbor search between the simplified trajectory and a slice of the point cloud is applied to detect the candidate door locations. Then, each sub-cloud is segmented to get the candidate walls. After that, the candidate trajectory points are analyzed in conjunction with the candidate walls to check the width of the possible door. Finally, the trajectory points that satisfied the door width condition are analyzed with the full point cloud to check the door height. This approach can detect the doors which are opened, semi-opened and closed.

- **How to assign the trajectory classes to the corresponding point cloud?** The time attribute of the trajectory is used to transfer the labels to the point cloud as both datasets are related together with the timestamp. For each set of sub-trajectories represent one space, the laser points which have the same timestamp as the trajectory points are labeled with the same class. Nevertheless, some laser points are within a space and have a different label. The mixed laser points are scanned from another space like the case of opened doors or glass walls, and the labeling results need to be refined.

- **What are the methods for refining the labeling results?** A majority filter is applied to each floor of the point cloud dataset. The filter is applied on a 2D representation of the point cloud as a quadtree with an assumption of the inner walls are vertical within each floor. The label that represents the majority of the points within each cell is assigned to the cell

label. Also, the geometry of the trajectory is used during the process as all the cells within a certain distance of the trajectory should have the same trajectory label. The geometry of the trajectory helps to refine the labels that are near the doorways as they are more likely to be scanned from different spaces. After that, another iteration of the majority filter is applied to every cell in the quadtree by considering the neighboring cells and take the majority of the labels. Finally, the labels were transferred back to the original points in the point cloud dataset.

## 5.2 RECOMMENDATIONS

The following suggestions are recommended for the operator behavior during the scanning process:

- Following the scanning guidelines by slowing down at the transitions between spaces, e.g., doorways and tight bends can help to detect the doorways without the need to analyze the full point clouds.
- The operator should walk through the elongated spaces like the corridors in multiple paths to have more trajectory point. The well covered space by trajectory can enhance the labeling process.
- Opening all the doors before starting the scanning process can enhance the door detection process and detecting the door boundary.
- Reflective surfaces such as mirrors should be covered if possible to reduce the reflected surfaces in the point cloud dataset.

The following suggestions are recommended for the future work:

- The detected critical points from trajectory analysis step can be used to optimize the operator behavior during the scanning, and to plan for the scanning process in further steps.
- The door detection method can be enhanced in case of the closed doors. Also, the door height can be analyzed locally for each door instead of using a fixed range for the door height.
- The analysis of the single scan line along the trajectory can be utilized to detect the opened doors which are not traversed during the scanning process.
- Eliminating the points outside the building boundary by detecting the walls can be a step towards generating a floor plan and resolving the oversegmented rooms issue.
- Testing the workflow on Non-Manhattan World datasets and with trajectories from different laser scanners can enhance the method performance.

# Appendices

# A. Trajectory Analysis

## A.1 TRAJECTORY SEGMENTATION

Figure A.1 shows height histogram for the ZEB1 trajectory dataset with different bin width. It was used during the experiments of segmenting the trajectory. However, this method did not provide a good segmentation for the trajectory. An optimized method was developed by combining the histogram with Fisher's Natural breaks as shown in Figure A.2 but this method need post processing for the data and it did not consider the time attribute.

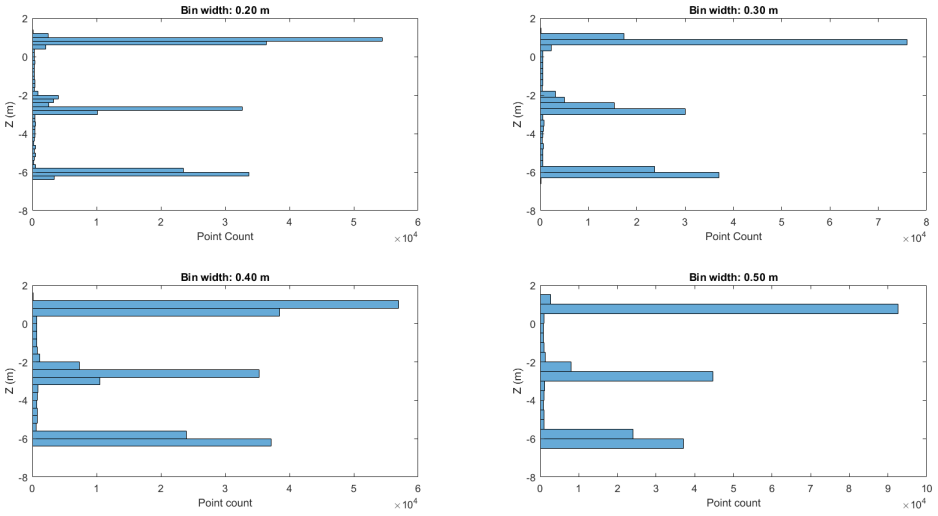



Figure A.1: Height histogram for ZEB1 trajectory with different bin width.

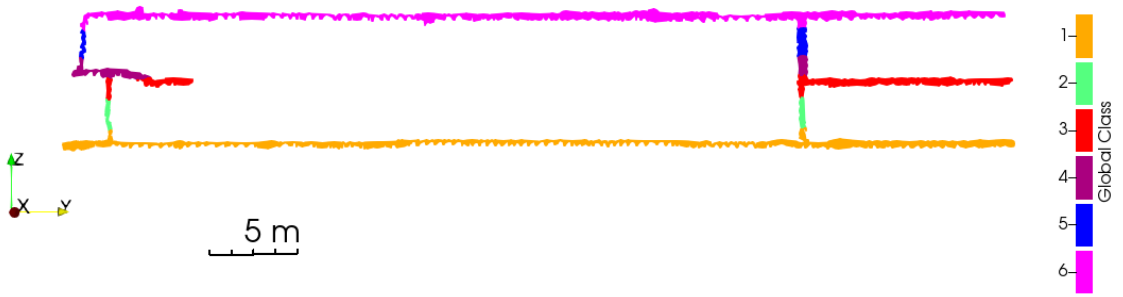



Figure A.2: ZEB1 trajectory segmentation using Fisher's Jenks Natural Breaks.

Table A.1: Trajectory segmentation summary for ZEB1 dataset from $SQLite$ database describing the different Global classes, time columns are in Unix timestamp.

| Time From | Time To | Point Count | Average Height (m) | Activity Class | Global Class |
|---|---|---|---|---|---|
| 1444730711 | 1444731317 | 60648 | -6 | 6 | 1 |
| 1444730691 | 1444730711 | 1995 | -4.6 | 5 | 2 |
| 1444731317 | 1444731361 | 4389 | -3.9 | 7 | 3 |
| 1444730332 | 1444730691 | 35910 | -2.8 | 4 | 4 |
| 1444731397 | 1444731521 | 12369 | -2.5 | 9 | 5 |
| 1444731361 | 1444731397 | 3591 | -2.2 | 8 | 6 |
| 1444730296 | 1444730332 | 3591 | -0.9 | 3 | 7 |
| 1444731521 | 1444731544 | 2394 | -0.4 | 10 | 8 |
| 1444731588 | 1444731592 | 399 | 0.9 | 0 | 9 |
| 1444729717 | 1444730068 | 35112 | 0.8 | 1 | 9 |
| 1444730068 | 1444730296 | 22743 | 0.8 | 2 | 9 |
| 1444731544 | 1444731588 | 4389 | 0.9 | 11 | 9 |
| 1444731592 | 1444731910 | 31773 | 0.9 | 12 | 9 |

Table A.2: Trajectory segmentation summary for ZEB-REVO dataset from $SQLite$ database describing the different Global classes, time columns are in Unix timestamp.

| Time From | Time To | Point Count | Average Height (m) | Activity Class | Global Class |
|---|---|---|---|---|---|
| 1490287691 | 1490287696 | 453 | 0 | 7 | 1 |
| 1490287687 | 1490287691 | 399 | 0.4 | 0 | 2 |
| 1490287037 | 1490287045 | 798 | 0.4 | 1 | 2 |
| 1490287045 | 1490287384 | 33915 | 0.7 | 2 | 3 |
| 1490287667 | 1490287687 | 1995 | 0.6 | 6 | 3 |
| 1490287384 | 1490287420 | 3591 | 1.5 | 3 | 4 |
| 1490287647 | 1490287667 | 1995 | 1.7 | 5 | 5 |
| 1490287420 | 1490287647 | 22743 | 3.1 | 4 | 6 |

## A.2 TRAJECTORY SIMPLIFICATION

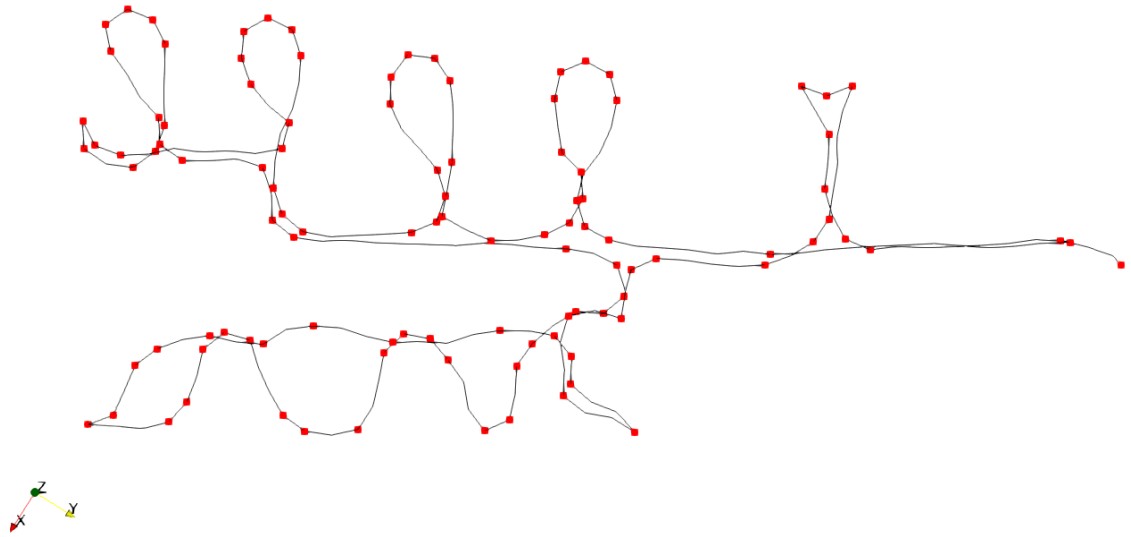
Table A.3 shows the optimal parameters used to compare the trajectory simplification algorithms which were shown in Figure 3.9.

Table A.3: Optimal parameters used during the trajectory simplification algorithms.

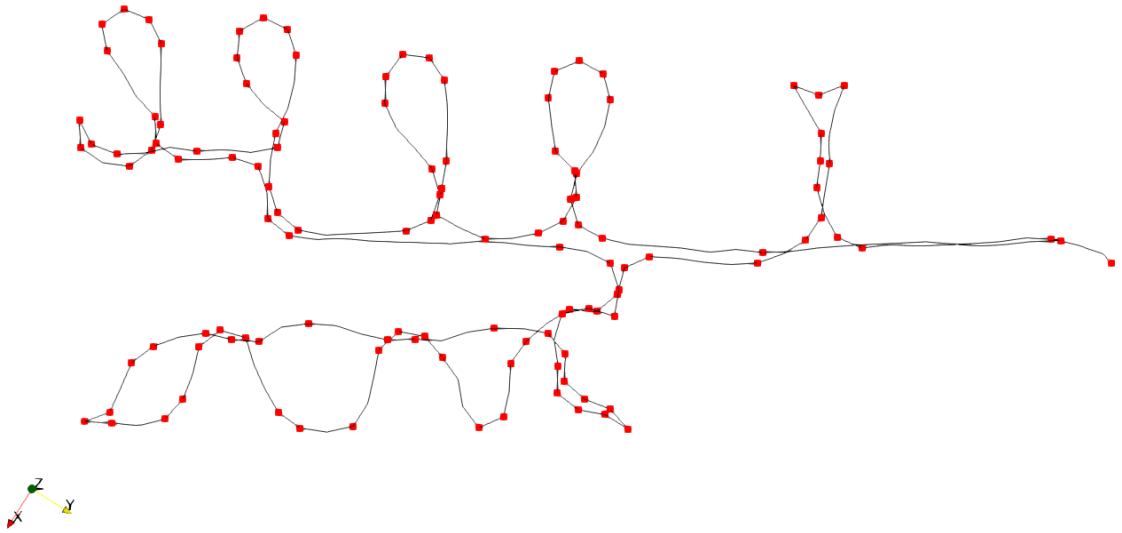
| Algorithm | Parameter | Value |
|---|---|---|
| **Douglas-Peucker** | | |
| | Maximum distance | 0.80 meter |
| **Visvalingam-Whyatt** | | |
| | Points count to keep | 80 points |
| **Radial Distance** | | |
| | Tolerance | 0.80 meter |
| **Perpendicular Distance** | | |
| | Maximum distance | 0.40 meter |
| **Reumann-Witkam** | | |
| | Maximum distance | 1.20 meter |
| **Opheim** | | |
| | Perpendicular distance | 1.30 meter |
| | Radial distance | 0.80 meter |
| **The Nth point stepping** | | |
| | Keep point every N points | 400 points |
| **The Nth Point Elimination with minimum number of points to keep** | | |
| | Point step | 10 points |
| | Points count to keep | 150 points |

## A.3 CRITICAL POINTS DETECTION

Figure A.3 shows a comparison between the two methods adopted get identify the geometric-based critical points for one of the ZEB-REVO floors. Although the choice of the parameters can show similar results, the Douglas-Peucker algorithm gives better results in case of using higher threshold for more generalization.
Figure A.4 shows the distribution of the critical points for the trajectory of ZEB1 top floor. Combining the turns in the trajectory with the stops can indicate the possible doors positions as shown in Figure A.4(b).




(a)




(b)

Figure A.3: Geometric-based critical points in red dots for simplified ZEB-REVO floor. (a) Douglas-Peucker with 0.2 m threshold. (b)Direction change with 25 degrees threshold.

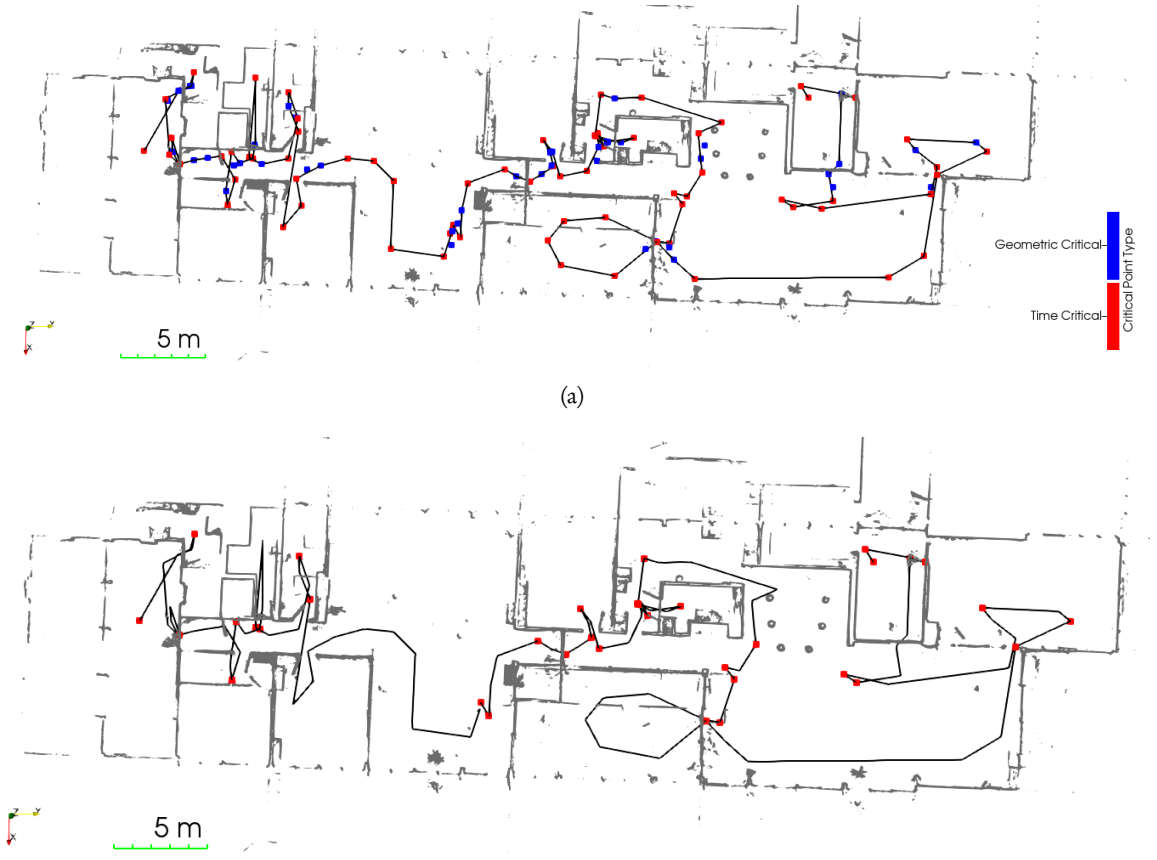



Figure A.4: (a)Critical points for ZEB1 dataset top floor. (b)Only critical points which are geometric-based and time-based at the same time are retained in red. Critical time threshold is 6.0 seconds.

### A.3.1 Scanner attitude-based critical points

Figure A.5 the three rotation angles of the ZEB-REVO scanner along the trajectory. We can notice the repeated pattern in Figure A.5(a) as the scanner head rotates around a horizontal axis.Also, the scanner has a limited change in pitch angle with the operator movement as shown in Figure A.5(b). On the other hand, the rotation around the vertical axis represented by the yaw angle shows a change in the pose with respect to the operator movement direction.
For ZEB1 trajectory, Figure A.6 shows the rotation angles change overtime along the trajectory. The device behaves in different way than the ZEB-REVO system as the scanner head mounted on a spring. As a result, there is a repeat of a wider rang of values for each rotation angle per unit distance. In case of yaw angle, this behavior can give totally different value for each trajectory point.

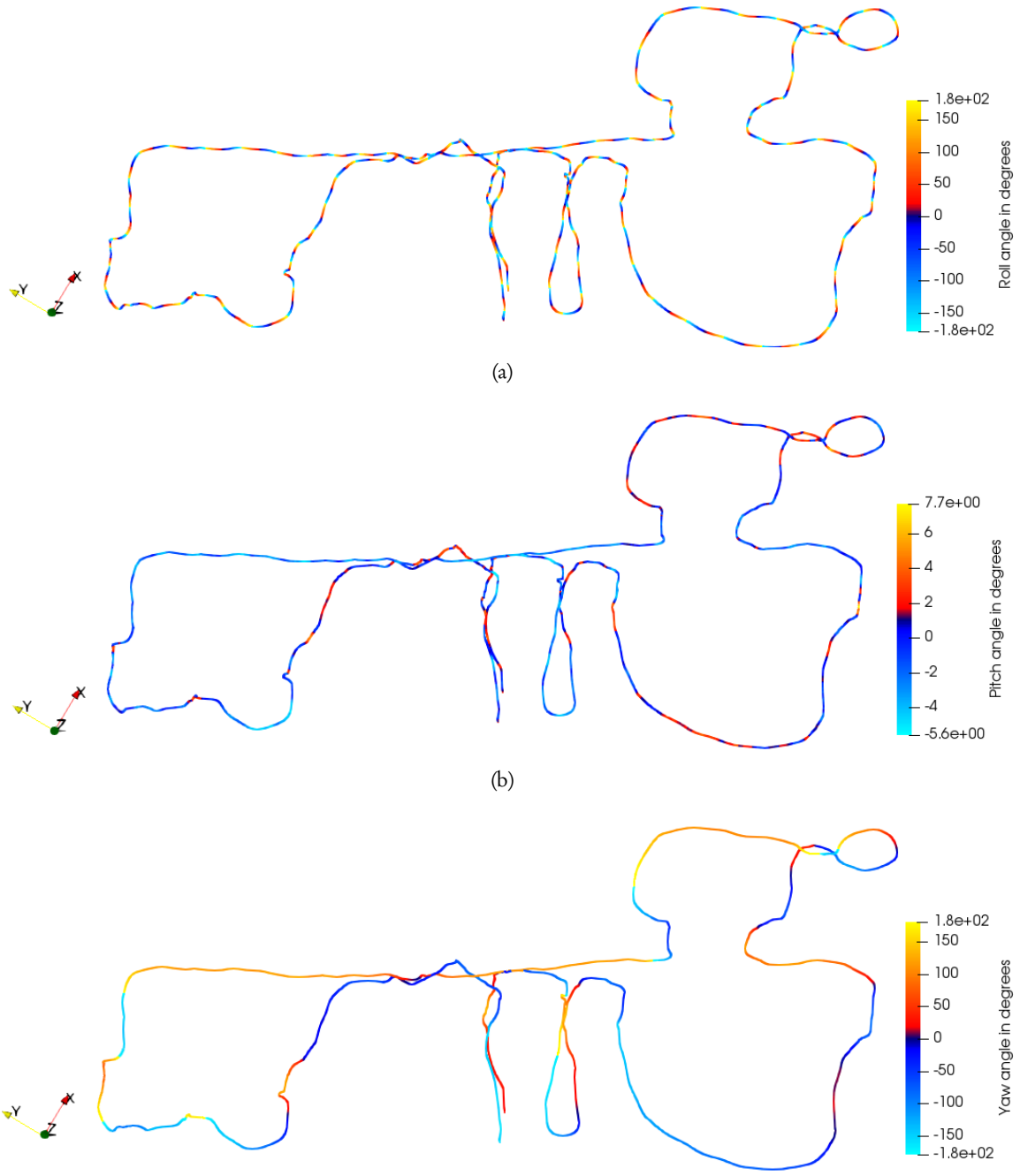



Figure A.5: The ZEB-REVO scanning system rotation angles in degrees along the trajectory (a)Roll angle. (b)Pitch angle with stretched values. (c)Yaw angle

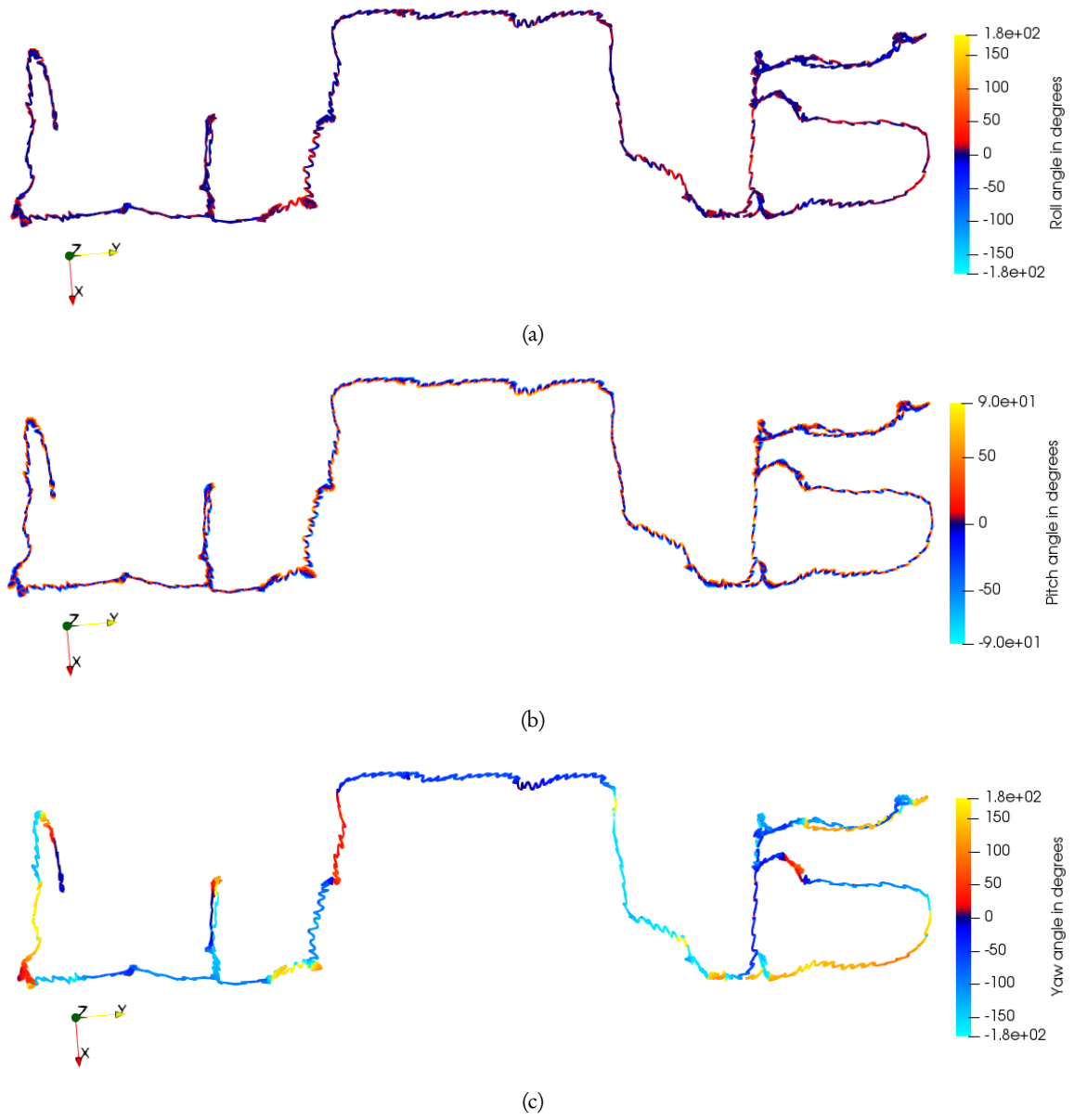



Figure A.6: The ZEB1 scanning system rotation angles in degrees along the trajectory (a)Roll angle. (b)Pitch angle. (c)Yaw angle

## A.4 SEMANTIC ANNOTATION

Figure A.7 shows the annotated ZEB1 trajectory after applying the semantic rules to identify the rooms, corridors, and staircases using the trajectory dataset only.

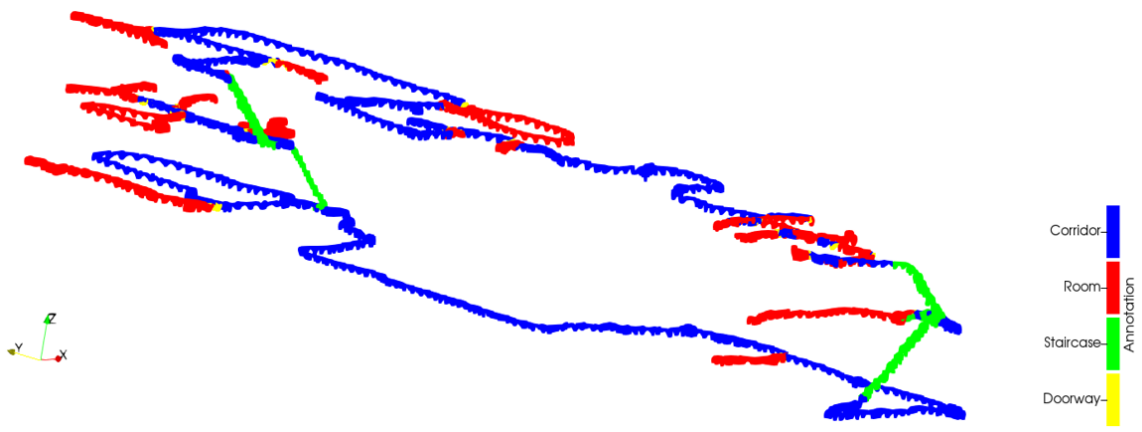



Figure A.7: The full annotation of ZEB1 trajectory.

# B. Joint Analysis between Point Cloud and Trajectory

## B.1 DOOR DETECTION

Figure B.1 shows the lines which are drawn to connect the extracted door boundaries. The blue circle shows the problem appeared during the door width check in the narrow areas. These spots have a width less than the door width threshold. In most cases, this issue is avoided by applying the door height check, but the ceiling level is low it causes a problem during the point cloud labeling process.

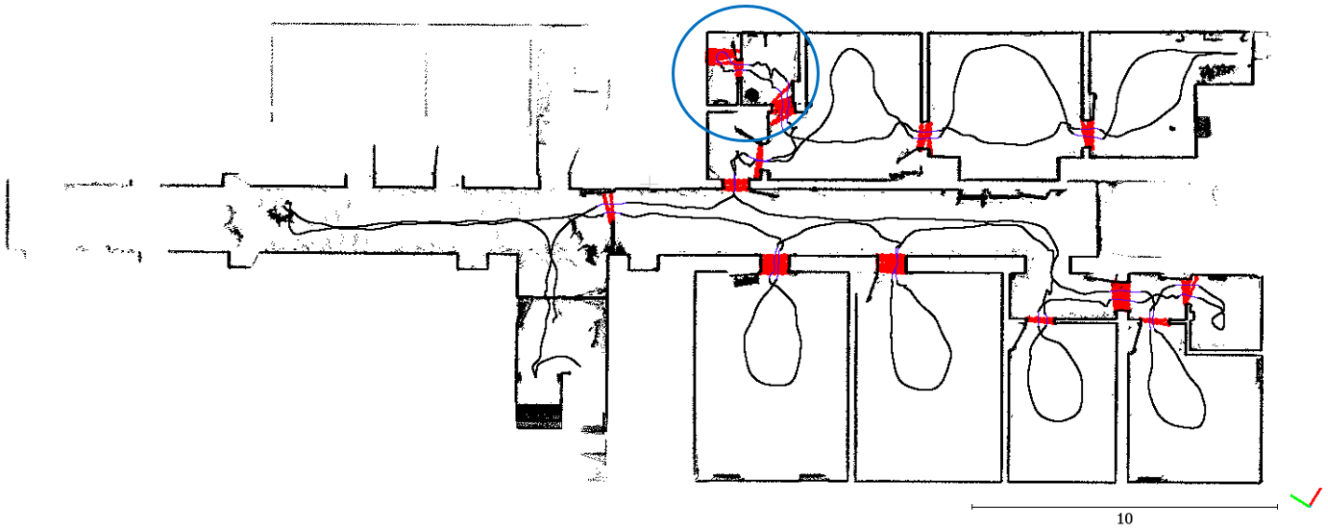



Figure B.1: The intermediate data for detecting the doors for the ZEB-REVO basement.The red points represent boundary region which affected the door width check. The blue circle shows the problems in the narrow areas which affect the candidate trajectory points.

## B.2 POINT CLOUD LABELING

Figure B.2 shows the top view for the labeled point cloud of ZEB1 to floor before applying the majority filter.

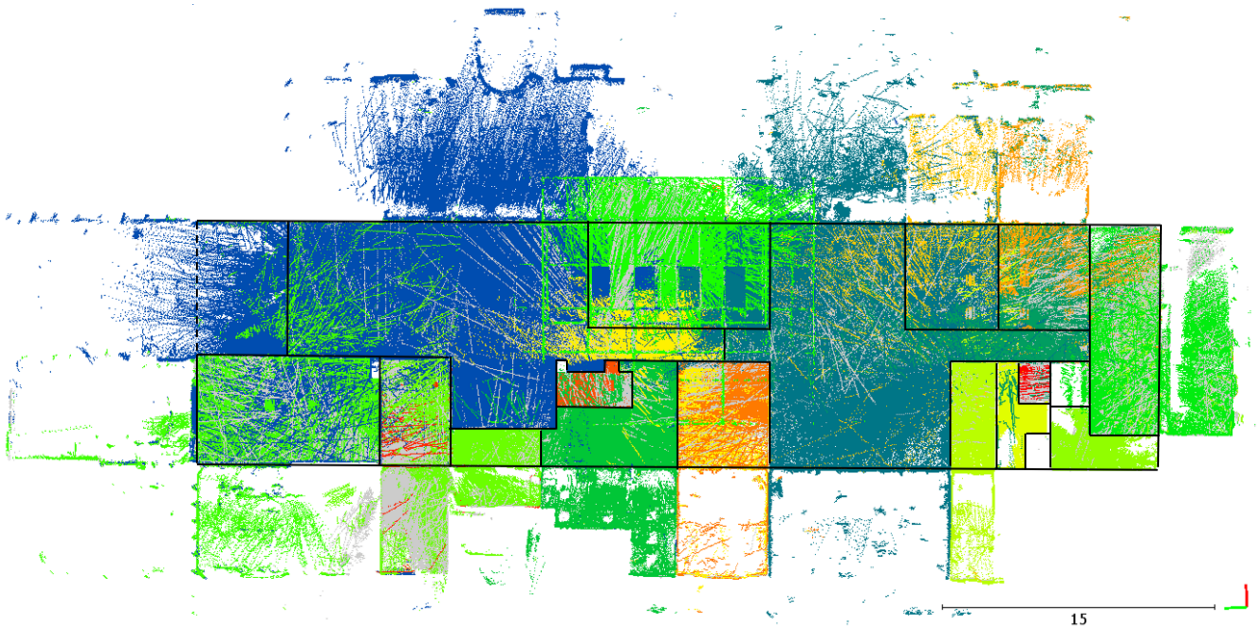



Figure B.2: The ZEB1 top floor with semantic labels transfered based on the time attribute.

## B.3 POST-PROCESSING TOWARD A FLOOR PLAN

The candidate walls were extracted as the initial step to solve the over-segmented rooms issue. The wall extraction method was adapted from (Turner & Zakhor, 2015). The idea was to take a slice with height H meters for the floor. Then, octree structure with voxel size n was used to get the occupied voxels. A wall candidate was defined as the number of occupied voxels along the vertical direction should equal to H/n. The aim of the wall candidates was to generate a floor plane based on Turner and Zakhor (2015) approach and to solve the over-segmentation issue at the same time. But the datasets had different characteristics and more advanced approach need to be developed. Figure B.3 and Figure B.4 show the extracted walls of the ZEB-REVO floors with the information of the door positions. In case of ZEB1 dataset, the extracted walls were representing the frame of the curtain walls as shown in Figure B.5 and Figure B.6. The octree voxel size was 0.05 meters and the point cloud slice height was 1.10 meters.

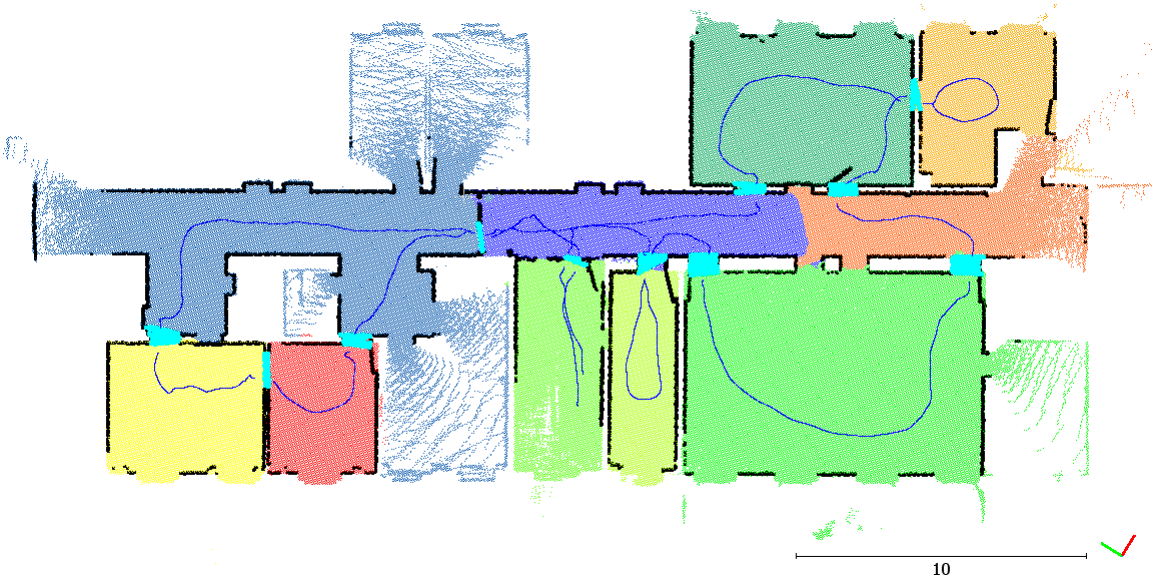



Figure B.3: The extracted walls of the ZEB-REVO top floor. The door positions are shown in cyan. The extracted walls are colored in black.

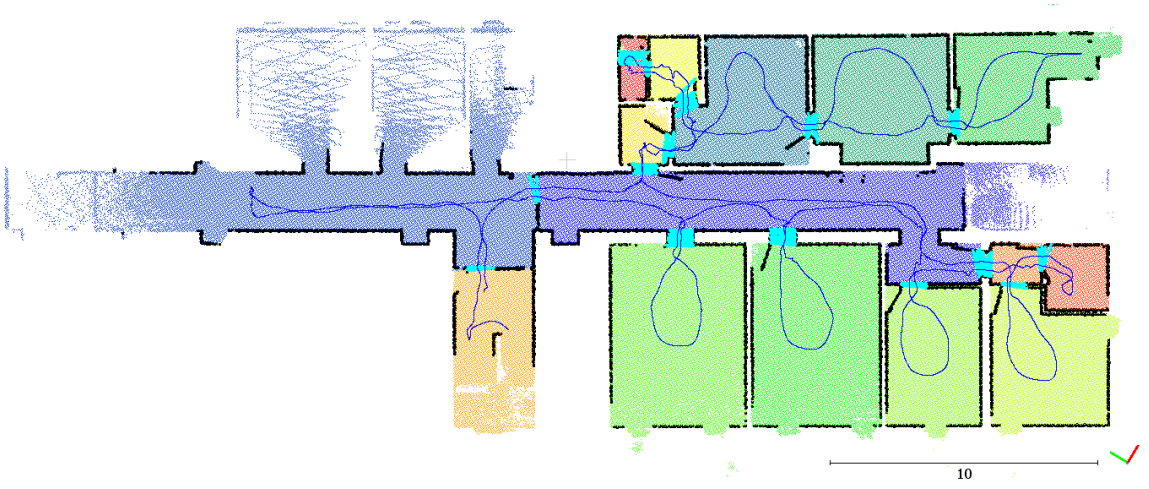



Figure B.4: The extracted walls of the ZEB-REVO basement. The door positions are shown in cyan. The extracted walls are colored in black.

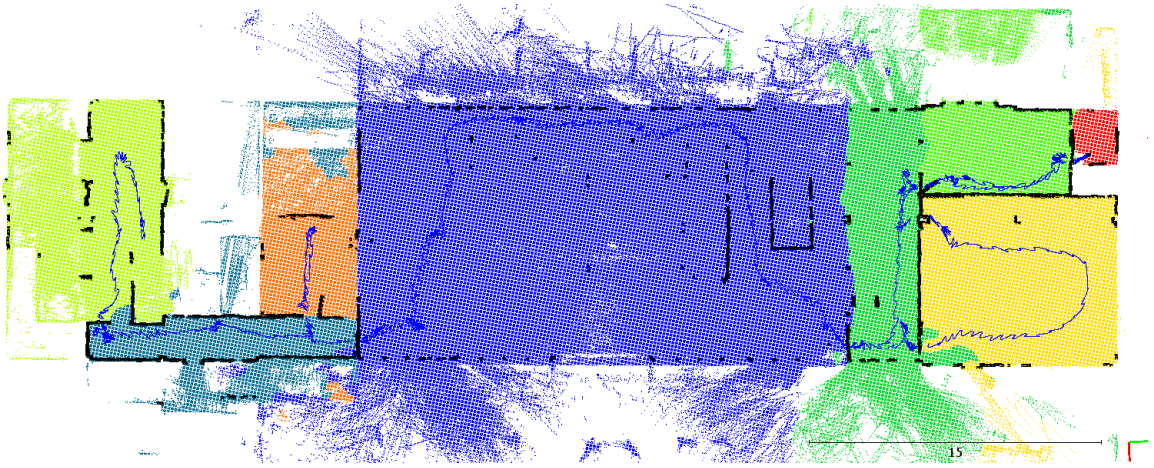



Figure B.5: The extracted walls of the ZEB1 ground floor. The extracted walls are colored in black.

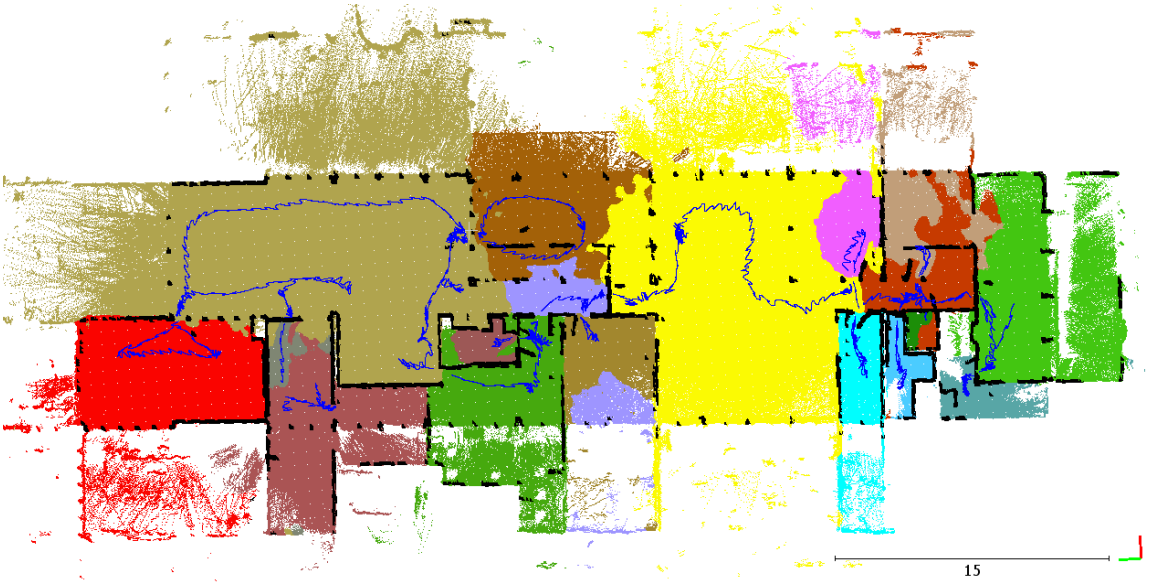



Figure B.6: The extracted walls of the ZEB1 top floor. The extracted walls are colored in black.

# References

Adan, A., & Huber, D. (2011, 5). 3D Reconstruction of Interior Wall Surfaces Under Occlusion and Clutter. In *Proceedings - 2011 international conference on 3d imaging, modeling, processing, visualization and transmission, 3dimpvt 2011* (pp. 275–281). IEEE. doi: 10.1109/3DIMPVT.2011.42

Alzantot, M., & Youssef, M. (2012). CrowdInside: Automatic Construction of Indoor Floorplans. In *Proceedings of the 20th international conference on advances in geographic information systems - sigspatial '12* (p. 99). New York, New York, USA: ACM Press. doi: 10.1145/2424321.2424335

Armeni, I., Sener, O., Zamir, A. R., Jiang, H., Brilakis, I., Fischer, M., & Savarese, S. (2016). 3D Semantic Parsing of Large-Scale Indoor Spaces. In *2016 ieee conference on computer vision and pattern recognition (cvpr)* (pp. 1534–1543). doi: 10.1109/CVPR.2016.170

Babacan, K., Chen, L., & Sohn, G. (2017, 11). Semantic Segmentation of Indoor Point Clouds Using Convolutional Neural Network. *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, *4*(4W4), 101–108. doi: 10.5194/isprs-annals-IV-4-W4-101-2017

Boost. (2017). *Boost C++ Libraries*. Retrieved from `http://www.boost.org/`

Bosse, M., Zlot, R., & Flick, P. (2012, 10). Zebedee: Design of a Spring-Mounted 3-D Range Sensor with Application to Mobile Mapping. *IEEE Transactions on Robotics*, *28*(5), 1104–1119. doi: 10.1109/TRO.2012.2200990

Buchin, M., Driemel, A., Van Kreveld, M., & Sacristan, V. (2011, 12). Segmenting Trajectories: A Framework and Algorithms Using Spatiotemporal Criteria. *Journal of Spatial Information Science*(3). doi: 10.5311/JOSIS.2011.3.66

Chen, C., Yang, B. S., & Song, S. (2016, 6). Low Cost and Efficient 3D Indoor Mapping Using Multiple Consumer RGB-D Cameras. *ISPRS - International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, *XLI-B1*, 169–174. doi: 10.5194/isprs-archives-XLI-B1-169-2016

de Koning, E. (2011). *psimpl - a generic n-dimensional polyline simplification library*. Retrieved from `http://psimpl.sourceforge.net/`

Díaz-Vilariño, L., Verbree, E., Zlatanova, S., & Diakité, A. (2017, 9). Indoor Modelling from Slam-Based Laser Scanner: Door Detection to Envelope Reconstruction. *ISPRS - International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, *XLII-2/W7*, 345–352. doi: 10.5194/isprs-archives-XLII-2-W7-345-2017

Douglas, D. H., & Peucker, T. K. (1973, 12). Algorithms for The Reduction of The Number of Points Required to Represent A Digitized Line or Its Caricature. *Cartographica: The International Journal for Geographic Information and Geovisualization*, *10*(2), 112–122. doi: 10.3138/FM57-6770-U75U-7727

Fichtner, F. W., Diakité, A. A., Zlatanova, S., & Voûte, R. (2018, 1). Semantic Enrichment of Octree Structured Point Clouds for Multi-story 3D Pathfinding. *Transactions in GIS*. doi: 10.1111/tgis.12308

Fisher, W. D. (1958). On Grouping for Maximum Homogeneity. *Journal of the American Statistical Association*, *53*(284), 789–798. doi: 10.1080/01621459.1958.10501479

Friedman, S., Pasula, H., & Fox, D. (2007). Voronoi Random Fields: Extracting the Topological Structure of Indoor Environments via Place Labeling. In *Proceedings of the 20th international joint conference on artifical intelligence* (pp. 2109–2114). San Francisco, CA, USA: Morgan Kaufmann Publishers Inc. Retrieved from `http://dl.acm.org/citation.cfm?id=1625275.1625616`

GeoSlam. (2016). *ZEB-REVO User Guide*. Retrieved from `https://solutions.seilerinst.com/Portals/1/2017/ZEB-REVOUserGuideV1-0-2.pdf`

Grzonka, S., Karwath, A., Dijoux, F., & Burgard, W. (2012, 2). Activity-based Estimation of Human Trajectories. *IEEE Transactions on Robotics*, *28*(1), 234–245. doi: 10.1109/TRO.2011.2165372

Guo, S., Xiong, H., Zheng, X., & Zhou, Y. (2017, 3). Activity Recognition and Semantic Description for Indoor Mobile Localization. *Sensors (Switzerland)*, *17*(3), 649. doi: 10.3390/s17030649

Hilferink, M. (2015). *Object Vision*. Retrieved from `http://wiki.objectvision.nl/index.php/CalcNaturalBreaksCode`

Jin, P., Cui, T., Wang, Q., & Jensen, C. S. (2016). Effective Similarity Search on Indoor Moving-Object Trajectories. In (pp. 181–197). Springer, Cham. doi: 10.1007/978-3-319-32049-6{_}12

Jung, J., Yoon, S., Ju, S., & Heo, J. (2015, 10). Development of Kinematic 3D Laser Scanning System for Indoor Mapping and As-Built BIM Using Constrained SLAM. *Sensors*, *15*(10), 26430–26456. doi: 10.3390/s151026430

Khoshelham, K., Vilariño, L. D., Peter, M., Kang, Z., & Acharya, D. (2017, 9). The ISPRS Benchmark on Indoor Modelling. *International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences - ISPRS Archives*, *42*(2W7), 367–372. doi: 10.5194/isprs-archives-XLII-2-W7-367-2017

Koppula, H. S., Anand, A., Joachims, T., & Saxena, A. (2011). Semantic Labeling of 3D Point Clouds for Indoor Scenes. *Neural Information Processing Systems*, 1–9. doi: 10.1109/CVPR.2012.6247992

Lai, K., Bo, L., Ren, X., & Fox, D. (2012, 5). Detection-based Object Labeling in 3D Scenes. In *2012 ieee international conference on robotics and automation* (pp. 1330–1337). IEEE. doi: 10.1109/ICRA.2012.6225316

Lauterbach, H., Borrmann, D., Heß, R., Eck, D., Schilling, K., & Nüchter, A. (2015, 10). Evaluation of a Backpack-Mounted 3D Mobile Scanning System. *Remote Sensing*, *7*(10), 13753–13781. doi: 10.3390/rs71013753

Lehtola, V., Kaartinen, H., Nüchter, A., Kaijaluoto, R., Kukko, A., Litkey, P., … Hyyppä, J. (2017, 8). Comparison of the Selected State-Of-The-Art 3D Indoor Scanning and Point Cloud Generation Methods. *Remote Sensing*, *9*(8), 796. doi: 10.3390/rs9080796

Long, C., Wong, R. C.-W., & Jagadish, H. V. (2013, 8). Direction-preserving Trajectory Simplification. *Proceedings of the VLDB Endowment*, *6*(10), 949–960. doi: 10.14778/2536206.2536221

Maboudi, M., Bánhidi, D., & Gerke, M. (2017). *Evaluation of Indoor Mobile Mapping Systems* (No. December).

McMaster, R. B. (1987, 9). The Geometric Properties of Numerical Generalization. *Geographical Analysis*, *19*(4), 330–346. doi: 10.1111/j.1538-4632.1987.tb00134.x

Meratnia, N., & de By, R. A. (2004). Spatiotemporal Compression Techniques for Moving Point Objects. In (pp. 765–782). Springer, Berlin, Heidelberg. doi: 10.1007/978-3-540-24741-8{_}44

Moghadam, P., Evans, B., & Duff, E. (2016). SAGE: Semantic Annotation of Georeferenced Environments. *Journal of Intelligent and Robotic Systems: Theory and Applications*, *83*(3-4),

635–648. doi: 10.1007/s10846-015-0302-3

Mozos, Ã. M. (2010). *Semantic Labeling of Places with Mobile Robots* (Vol. 61). Berlin, Heidelberg: Springer Berlin Heidelberg. doi: 10.1007/978-3-642-11210-2

Mura, C., Mattausch, O., Jaspe Villanueva, A., Gobbetti, E., & Pajarola, R. (2014, 11). Automatic Room Detection and Reconstruction in Cluttered Indoor Environments with Complex Room Layouts. *Computers and Graphics (Pergamon)*, *44*(1), 20–32. doi: 10.1016/j.cag.2014.07.005

Nikoohemat, S., Peter, M., Oude Elberink, S., & Vosselman, G. (2017, 9). Exploiting Indoor Mobile Laser Scanner Trajectories for Semantic Interpretation of Point Clouds. *ISPRS Ann. Photogramm. Remote Sens. Spatial Inf. Sci.*, *IV-2/W4*, 355–362. doi: 10.5194/isprs-annals-IV-2-W4-355-2017

Nüchter, A., Borrmann, D., Koch, P., Kühn, M., & May, S. (2015, 8). A Man-Portable Imu-Free Mobile Mapping System. *ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information Sciences*, *II-3/W5*, 17–23. doi: 10.5194/isprsannals-II-3-W5-17-2015

Ochmann, S., Vock, R., Wessel, R., & Klein, R. (2015). Automatic reconstruction of parametric building models from indoor point clouds. *Computers & Graphics*, *54*, 94–103. doi: 10.1016/j.cag.2015.07.008

Oesau, S., Lafarge, F., & Alliez, P. (2014, 4). Indoor Scene Reconstruction Using Feature Sensitive Primitive Extraction and Graph-Cut. *ISPRS Journal of Photogrammetry and Remote Sensing*, *90*, 68–82. doi: 10.1016/j.isprsjprs.2014.02.004

Okorn, B., Xiong, X., Akinci, B., & Huber, D. (2010). Toward Automated Modeling of Floor Plans. In *Proceedings of the symposium on 3d data processing, visualization and transmission* (Vol. 2). Retrieved from `http://swing.adm.ri.cmu.edu/pub_files/2010/5/20093DPVTplanviewmodelingv13(resubmitted).pdf`

Opheim, H. (1981). Smoothing a Digitized Curve by Data Reduction Methods. In J. L. Encarnacao (Ed.), *Eurographics conference proceedings.* The Eurographics Association.

Prentow, T., Thom, A., Blunck, H., & Vahrenhold, J. (2015, 6). Making Sense of Trajectory Data in Indoor Spaces. *2015 16th IEEE International Conference on Mobile Data Management*, *1*, 116–121. doi: 10.1109/MDM.2015.44

Quintana, B., Prieto, S. A., Adán, A., & Bosché, F. (2018, 1). Door Detection in 3D Coloured Point Clouds of Indoor Environments. *Automation in Construction*, *85*, 146–166. doi: 10.1016/j.autcon.2017.10.016

Regnauld, N., & McMaster, R. B. (2007). A synoptic View of Generalisation Operators. In *Generalisation of geographic information* (pp. 37–66). Elsevier. doi: 10.1016/B978-008045374-3/50005-3

Reumann, K., & P. M. Witkam, A. (1974). Optimizing Curve Segmentation in Computer Graphics. *Proceedings of International Computing Symposium*, *7*.

Rocha, J. A. M. R., Times, V. C., Oliveira, G., Alvares, L. O., & Bogorny, V. (2010, 7). DB-SMoT: A Direction-Based Spatio-Temporal Clustering Method. In *2010 5th ieee international conference intelligent systems* (pp. 114–119). IEEE. doi: 10.1109/IS.2010.5548396

Rusu, R. B., Marton, Z. C., Blodow, N., Dolha, M., & Beetz, M. (2008, 11). Towards 3D Point cloud based object maps for household environments. *Robotics and Autonomous Systems*, *56*(11), 927–941. doi: 10.1016/j.robot.2008.08.005

Sack, J.-R. J.-R., & Urrutia, J. J. (2000). *Handbook of computational geometry*. Elsevier. Retrieved from `http://www.sciencedirect.com/science/book/9780444825377`

Shi, W., & Cheung, C. (2006, 3). Performance Evaluation of Line Simplification Algorithms for Vector Generalization. *The Cartographic Journal*, *43*(1), 27–44. doi: 10.1179/000870406X93490

Sirmacek, B., Shen, Y., Lindenbergh, R., Zlatanova, S., & Diakite, A. (2016, 6). Comparison of Zeb1 and Leica C10 Indoor Laser Scanning Point Clouds. *ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information Sciences*, *III-1*, 143–149. doi: 10.5194/isprsannals-III-1-143-2016

Soares, S. G., & Araújo, R. (2014). Semantic Place Labeling Using a Probabilistic Decision List of AdaBoost Classifiers. *International Journal of Computer Information Systems and Industrial Management Applications*, *6*, 2150–7988. Retrieved from `http://www.mirlabs.net/ijcisim/regular_papers_2014/IJCISIM_51.pdf`

Staats, B. R., Diakité, A. A., Voûte, R. L., & Zlatanova, S. (2017, 9). Automatic Generation of Indoor Navigable Space Using a Point Cloud and Its Scanner Trajectory. *ISPRS Ann. Photogramm. Remote Sens. Spatial Inf. Sci.*, *IV-2/W4*, 393–400. doi: 10.5194/isprs-annals-IV-2-W4-393-2017

Sun, P., Xia, S., Yuan, G., & Li, D. (2016, 5). An Overview of Moving Object Trajectory Compression Algorithms. *Mathematical Problems in Engineering*, *2016*, 1–13. doi: 10.1155/2016/6587309

SUNY Institute of Technology. (2012). *Curve Simplification Algorithms*. Retrieved from `https://web.cs.sunyit.edu/~poissad/projects/Curve/about_algorithms/`

Thomson, C., Apostolopoulos, G., Backes, D., & Boehm, J. (2013, 10). Mobile Laser Scanning for Indoor Modelling. *ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information Sciences*, *II-5/W2*, 289–293. doi: 10.5194/isprsannals-II-5-W2-289-2013

Toschi, I., Rodríguez-Gonzálvez, P., Remondino, F., Minto, S., Orlandini, S., & Fuller, A. (2015, 2). Accuracy Evaluation of a Mobile Mapping System with Advanced Statistical Methods. *ISPRS - International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, *40*(5W4), 245–253. doi: 10.5194/isprsarchives-XL-5-W4-245-2015

Turner, E., & Zakhor, A. (2015). Multistory Floor Plan Generation and Room Labeling of Building Interiors from Laser Range Data. In (pp. 29–44). Springer, Cham. doi: 10.1007/978-3-319-25117-2

Visvalingam, M., & Whyatt, J. D. (1993, 6). Line Generalisation by Repeated Elimination of Points. *The Cartographic Journal*, *30*(1), 46–51. doi: 10.1179/000870493786962263

Vosselman, G., Gorte, B. G. H., Sithole, G., & Rabbani, T. (2004). Recognising Structure in Laser Scanning Point Clouds. In M. Thies, B. Koch, H. Spiecker, & H. Weinacher (Eds.), *Isprs 2004 : proceedings of the isprs working group viii/2 : laser scanning for forest and landscape assessment, freiburg, october 3-6, 2004. / ed. by m. thies, b. koch, h. spiecker and h. weinacher. freiburg; university of freiburg, 2004. pp. 33-38* (pp. 33–38). University of Freiburg.

Wen, C., Pan, S., Wang, C., & Li, J. (2016, 7). An Indoor Backpack System for 2-D and 3-D Mapping of Building Interiors. *IEEE Geoscience and Remote Sensing Letters*, *13*(7), 992–996. doi: 10.1109/LGRS.2016.2558486

Xie, L., & Wang, R. (2017, 9). Automatic Indoor Building Reconstruction from Mobile Laser Scanning Data. *Int. Arch. Photogramm. Remote Sens. Spatial Inf. Sci.*, *XLII-2/W7*, 417–422. doi: 10.5194/isprs-archives-XLII-2-W7-417-2017

Xiu-Li, Z., & Wei-Xiang, X. (2009). A Clustering-Based Approach for Discovering Interesting Places in a Single Trajectory. In *2009 second international conference on intelligent computation technology and automation* (pp. 429–432). IEEE. doi: 10.1109/ICICTA.2009.569

Zlot, R., Borges, P., & Bell, G. (2015). *Royal Exhibition Building: Zebedee 3D Data Collection* (Tech. Rep.). CSIRO.