Indoor 3D Reconstruction of Buildings from Point Clouds

Shayan Nikoohemat

INDOOR 3D RECONSTRUCTION OF BUILDINGS FROM POINT CLOUDS

DISSERTATION

to obtain
the degree of doctor at the University of Twente,
on the authority of the rector magnificus,
prof.dr. T.T.M. Palstra,
on account of the decision of the Doctorate Board,
to be publicly defended
on Friday, March 27, 2020 at 12:45

by

Shayan Nikoohemat

born on May 22, 1983

in Tehran, Iran

This thesis has been approved by
Prof.dr.ir. M.G. Vosselman, supervisor

Dr.ir. S.J. Oude Elberink, co-supervisor

ITC dissertation number 381
ITC, P.O. Box 217, 7500 AE Enschede, The Netherlands

ISBN 978-90-365-4982-0
DOI 10.3990/1.9789036549820

Cover designed by Benno Masselink
Printed by ITC Printing Department

UNIVERSITY OF TWENTE. | ITC

Graduation committee:

Chairman/Secretary

Prof.dr.ir. A. Veldkamp	

Supervisor

Prof.dr.ir. M.G. Vosselman	University of Twente / ITC

Co-supervisor(s)

dr.ir. S.J. Oude Elberink	University of Twente / ITC

Members

Prof.dr.ir. A.M. Adriaanse	University of Twente / ET
dr.ir. D. Lutters	University of Twente / ET
Prof.dr. P. Alliez	Inria Sophia Antipolis, France
Prof.dr.-ing. U. Stilla	TU München, Germany

To my parents for their endless love …

Summary

The developments of 3D acquisition systems for indoor environments has increased in last year. Among them, the emerge of mobile laser scanners (MLS) and low-cost sensors for scanning interiors of large buildings and providing 3D scans (point clouds and RGBD images) enable architects, engineers, and managers to access affordable digital twins of the buildings in a short time. However, such improvements come at the cost of tackling a large amount of data in forms of point clouds and images. Users in the architecture, engineering, and construction (AEC) domain prefer a compact and light version digital representation of buildings instead of a large number of point clouds. Thus, the problem of designing (semi-) automatic methods for converting 3D scans to semantically rich 3D models raised in recent years. In the literature, this problem is addressed as scan-to-bim (Building Information Models) and as-is vs. as-built. However, this manuscript tries to go beyond providing just a BIM model by also studying the best practices to keep such 3D models up-to-date, and monitoring the changes during the building lifetime as well as investigating the compliance of the output with the standards and applications.

This thesis has three main parts: the first part, including chapters 1 and 2, explains the motivation of this PhD work, provides a review of current data acquisition devices and 3D indoor standards and the modeling methods in the related work, and summarizes the open challenges. The second part is presented in chapters 3 and 4, where the main pipeline for indoor 3D reconstruction from point clouds is further developed and discussed. The last part, including chapter 5 and 6, investigates the considerations need to be taken after the creation of a 3D model from scans. This contains consistency control and compliance of 3D models with indoor standards (IndoorGML and IFC). Furthermore, monitoring the changes of buildings without the need to scan the whole complex after each renovation and discovering the type of changes (temporary or structural changes) are described in this last part. The last chapter provides conclusions and recommendations.

The goal of this research is not only creating 3D models from point clouds but advancing the state-of-the-art and tackling the shortcomings of previous research. In this regard, addressing open challenges such as incomplete data because of cluttered environment, fictitious data because of reflective surfaces, modeling of non-Manhattan World structures and avoiding the assumptions of vertical walls and horizontal ceilings were main concerns of our work. Four objectives are proposed to engage in these open problems: semantic labeling, geometric modeling, watertight 3D model reconstruction, and consistency control of 3D models. The first objective contributes to the problem of the classification of indoor point clouds. The proposed solutions aim at discerning the permanent structures, including three classes of walls, floor and ceilings,

from the clutter (noise and furniture). Several heuristic methods with the support of creating an adjacency graph are developed which exploits the topology of manmade structures. The solutions prove that the trajectory of the mobile laser scanner is beneficial in understanding the indoor scenes. For example, the trajectory is used for separating floor levels of the building, detection of closed doors and openings, identifying fictitious caused by reflective surfaces and labeling points belonging to the stairs. In addition to the trajectory solution, 3D mathematical morphology is applied in voxel space for identifying navigable spaces and partitioning the space. The result of semantic labeling reaches an accuracy of an average 95% for permanent structures, tested in six different use cases with complex architectures and a high amount of glass surfaces and clutter.

After semantic labeling, the second and third objectives develop the process of constructing a watertight 3D model by creating volumetric walls and extracting the room polyhedra from enclosed spaces. This part of our research provides a semi-automatic method for modifying the geometry of planar segments before modeling permanent structures. Moreover, for disaster management applications, methods are developed for modeling stairs in multistory buildings, modeling furniture as obstacles, and adding doors. These are supported by a fine-grained space subdivision based on the enclosure of space, i.e. the connection of walls, floors and ceilings form a closed space. Space subdivisions are further divided into subspaces by including the furniture in the process. Finally, we demonstrate the robustness of our algorithms on four complex multistory buildings. The contributions of this part are modeling the interiors with and without the furniture for advanced navigation networks and modeling both volumetric walls (complying with BIM models) and volumetric spaces (complying with IndoorGML models). By comparing our models with handcrafted BIM models, we showed that our pipeline reaches an accuracy of 90% in modeling the rooms and doors and this includes detecting some of the closed doors. Unlike other related works that use 3D models only for BIM or for navigation purposes, our results demonstrate real-world examples from point clouds (no synthetic model) for both applications.

In addition to creating a 3D model, checking the compliance of the model with indoor standards and the suitability of the model for the application are sometimes neglected in the research. Therefore, in the last objective, we investigate, in the lack of ground truth, how the consistency of the model can be verified. The consistency envelopes the accuracy and correctness of the model semantically, geometrically, and topologically. In addition to the common expert knowledge which can be useful to verify the consistency of the model, experts provide standards such as IFC, IndoorGML, and ISO 19107 for the treatment of spatial information and indoor models up to three dimensions. However, as the 3D models created from scans can vary in terms of the level

of details and geometry, we tried no to provide a step-by-step instruction but design a grammar-based concept that is flexible to such a variation. The proposed solution is a conceptual framework that provides a formal approach in three phases to use standards and expert knowledge for consistency control of the 3D models. These three steps are: controlling and verifying the individual instances in the model (e.g., each wall object), verifying the interaction of instances (e.g., a door on a wall) and verifying the consistency of the model for a specific application (e.g., for navigation). To this end, indicating the inconsistency is the goal of the framework, not fixing the problem. Therefore, the output of such a formal grammar are valid or invalid components in the model which are rejected to previous steps (e.g., geometric modeling, semantic) for further investigations. Apart from reconstructing 3D models from point clouds, scan data can be useful for change detection in indoor environments. The changes can be monitored after renovation or redecoration of the interiors. We showed that point clouds could capture the changes below several centimeters and afterwards our 3D modeling algorithms can discern the permanent changes from changes in the furniture. As a use case, the application of changes is demonstrated in 3D cadaster of interiors.

As a conclusion, the methods developed in this research show that there is a great potential in the automation of scan-to-bim and creating as-is models even from complex architectures. The future work should be dedicated to adding level-of-details such as the type of furniture and function of the rooms. Another line of research can be applying deep learning methods for early-stage classification of the point clouds before the modeling step. Moreover, stitching indoor 3D models to the exterior model of buildings provides a seamless reconstruction of large-scale city 3D models.

Samenvatting

De mogelijkheden om de binnenkant van gebouwen in te meten in 3D zijn de laatste jaren enorm toegenomen. Eén voorbeeld daarvan is de opkomst van mobiele laserscanners (MLS) en goedkope sensoren voor het scannen van interieurs van grote gebouwen. Zij leveren hoge resolutie 3D-scans (point clouds en RGBD-afbeeldingen). Het stelt architecten, ingenieurs en managers in staat snel een 3D beeld te krijgen van de binnenkant van gebouwen. Maar met de mogelijkheden om snel heel veel data in te winnen komen ook problemen qua data verwerking. Gebruikers in het AEC-domein (architectuur, engineering en constructie) geven de voorkeur aan een compacte en lichte digitale versie van gebouwen in plaats van een groot aantal puntenwolken. Het probleem van het ontwerpen van (semi-) automatische methoden voor het converteren van 3D-scans naar semantisch rijke 3D-modellen is de afgelopen jaren aan de orde gekomen. In de literatuur wordt dit probleem aangemerkt als scan-to-bim (Building Information Models) en as-is versus as-built. Dit proefschrift probeert echter verder te gaan dan alleen het aanbieden van een BIM-model door ook de best practices te bestuderen om dergelijke 3D-modellen up-to-date te houden, de veranderingen tijdens de levensduur van het gebouw te volgen en te onderzoeken of de output voldoet aan de normen en toepassingen.

Dit proefschrift bestaat uit drie hoofdonderdelen: het eerste deel, bestaande uit hoofdstukken 1 en 2, legt de motivatie van dit promotieonderzoek uit, biedt een overzicht van de huidige laserscanningssystemen om data in te winnen en 3D- modelleringsmethoden, en benoemt de open uitdagingen. Het tweede deel wordt gepresenteerd in hoofdstukken 3 en 4, waar de hoofdlijn voor 3D-reconstructie binnenshuis uit puntwolken verder wordt ontwikkeld. Het laatste deel, hoofdstuk 5 en 6, onderzoekt de overwegingen die moeten worden genomen na het maken van een 3D-model van scans. Dit omvat consistentiecontrole en conformiteit van 3D-modellen met binnenstandaarden (IndoorGML en IFC). Verder worden in dit laatste deel het detecteren van de veranderingen van gebouwen beschreven zonder de noodzaak om het hele gebouw na elke renovatie te scannen en het soort veranderingen (tijdelijke of structurele veranderingen) te ontdekken. Ten slotte bevat het laatste hoofdstuk conclusies en aanbevelingen.

Het doel van dit onderzoek is niet alleen het creëren van 3D-modellen vanuit puntenwolken, maar ook het bevorderen van de nieuwste technieken en het aanpakken van de tekortkomingen van eerder onderzoek. We richten ons op het aanpakken van open uitdagingen zoals onvolledige datasets doordat er andere objecten voorstonden, of schijnwaarnemingen door reflecterende oppervlakken, modellering van niet-rechthoekige structuren. Er worden vier doelstellingen voorgesteld om deze open problemen aan te pakken:

semantische labeling, geometriemodellering, waterdichte 3D-modelreconstructie en consistentiecontrole van 3D-modellen. De eerste doelstelling draagt bij het verbeteren van de classificatie van puntenwolken die in gebouwen zijn ingewonnen. De voorgestelde oplossingen zijn gericht op het onderscheiden van de permanente structuren, waaronder drie klassen wanden, vloeren en plafonds, van de overige objecten zoals meubels. Er zijn verschillende heuristische methoden ontwikkeld die gebruik maken van de relaties tussen permanente structuren. Daarnaast blijkt dat het gebruik van de positie van de scanners tijdens de inwinning, een traject in het geval van mobiele inwinning, nuttig is om de binnenscènes te begrijpen. Het traject wordt bijvoorbeeld gebruikt voor het scheiden van vloerniveaus van het gebouw, het detecteren van gesloten en open deuren, het identificeren van schijnwaarnemingen door reflecterende oppervlakken. Naast het gebruik van het traject worden ook morfologische operaties gebruikt voor het herkennen van open ruimtes. Het resultaat van semantische labeling bereikt een nauwkeurigheid van gemiddeld 95% voor permanente structuren, getest in zes verschillende situaties met complexe architecturen en een grote hoeveelheid ramen en meubels.

Na de semantische labeling richten de tweede en derde doelstellingen zich op het construeren van een waterdicht 3D-model. Dit wordt bereikt door het creëren van volumetrische wanden. Dit deel van ons onderzoek biedt een semi-automatische methode voor het wijzigen van de geometrie van vlakke segmenten voordat permanente structuren worden gemodelleerd. Bovendien zijn voor rampenbeheertoepassingen methoden ontwikkeld voor het modelleren van trappen in gebouwen met meerdere verdiepingen, het modelleren van meubels als obstakels en het toevoegen van deuren. Deze worden ondersteund door een gedetailleerde onderverdeling van de binnenruimte op basis van de omsluiting van individuele ruimtes door muren, vloeren en plafonds. De onderverdelingen van de ruimte worden verder onderverdeeld in deelruimten door het meubilair in het proces op te nemen. Ten slotte tonen we de robuustheid van onze algoritmen op vier complexe gebouwen met meerdere verdiepingen. De bijdragen van dit onderdeel zijn het modelleren van het interieur met en zonder het meubilair voor geavanceerde navigatienetwerken en het modelleren van zowel volumetrische wanden (conform BIM-modellen) als volumetrische ruimtes (conform IndoorGML-modellen). Door onze modellen te vergelijken met handgemaakte BIM-modellen, tonen we aan dat onze methode een nauwkeurigheid van 90% bereikt bij het modelleren van de kamers en deuren en dit omvat het detecteren van enkele van de gesloten deuren. In tegenstelling tot andere gerelateerde onderzoeken die 3D-modellen alleen gebruiken voor ofwel voor BIM of voor navigatiedoeleinden, tonen onze resultaten voorbeelden uit de praktijk voor beide toepassingen.

Naast het maken van een 3D-model, blijft het controleren van het model met binnenstandaarden en de geschiktheid van het model voor de toepassing vaak onderbelicht. Daarom onderzoeken we bij het laatste doel, bij gebrek aan referentiemateriaal, hoe de consistentie van het model kan worden geverifieerd. De consistentie bevat de nauwkeurigheid en correctheid van het model op een semantische, geometrische en topologische manier. Naast de algemene kennis die nuttig kan zijn om de consistentie van het model te verifiëren, zijn er standaarden zoals IFC, IndoorGML en ISO 19107. Omdat de 3D-modellen die zijn gemaakt op basis van laserscans kunnen variëren qua details en geometrie, hebben we een op grammatica gebaseerd concept ontworpen dat flexibel is voor een dergelijke variatie. Dat is een conceptueel raamwerk dat bestaat uit drie stappen. Deze drie stappen zijn: de selectie van de afzonderlijke onderdelen in het model (bijvoorbeeld elk wandobject), het verifiëren van de combinatie van onderdelen (bijvoorbeeld een deur op een muur) en het controleren van de consistentie van het model voor een specifieke toepassing (bijvoorbeeld, voor navigatie). Afgezien van het reconstrueren van 3D-modellen uit puntenwolken, kunnen scangegevens nuttig zijn voor het detecteren van veranderingen in binnenomgevingen. De veranderingen kunnen een gevolg zijn van renovatie of herinrichting van het interieur. We tonen aan dat puntenwolken de veranderingen op enkele centimeters nauwkeurig kunnen vastleggen en daarna kunnen onze 3D-modelleringsalgoritmen de permanente veranderingen onderscheiden van tijdelijke veranderingen. We hebben dat laten zien aan de hand van een use-case op het gebied van 3D Kadaster informatie.

Concluderend laten de in dit onderzoek ontwikkelde methoden zien dat het mogelijk is om automatisch scan-to-bim en as-is modellen te maken, zelfs voor complexe architecturen. Toekomstig onderzoek kan zich richten op het toevoegen van meer detail, zoals het type meubels en het herkennen van de functie van een ruimte. Een andere onderzoekslijn is het toepassen van deep learning voor de classificatie van de puntenwolken vóór de modelleringsstap. Het combineren van 3D-modellen voor binnen met modellen van de buitenkant van gebouwen is een andere interessante onderzoekslijn die nog verder uitgewerkt kan worden.

Acknowledgements

This work is part of the TTW Maps4Society project Smart Indoor Models in 3D (SIMs3D) to support crisis management in large public buildings (13742) which is (partly) financed by the Netherlands Organization for Scientific Research (NWO).

When I was writing this acknowledgement, all the last four years flashed back to me, the efforts, deadlines, conferences, stress, laughs and parties. It was a trajectory not only of research but also of experiences, how to cope with challenging situations, and keeping the life-work balance! Traveling my PhD trajectory was not possible without many colleagues and friends whom I would like to thank here.

I would like to thank my supervisor, George who taught me how to be precise at my work and stay with what I plan, I tried to learn it from you George. He is the teacher who is meticulous both in administration and research. His knowledge of Lidar and photogrammetry was very valuable in my research. Thanks to my former co-supervisor, Michael Peter, he helped me like a friend in the early stage of my PhD and instilled me what to do when I was lost. Without his support and enthusiasm, I could not get the best paper award. Some ideas in this PhD work was initialized by Sander, thank you for your concise and effective input. Abdoulaye, you are the funniest and smartest buddy. Although most of our collaboration was from a far distance (Delft – Sydney – Enschede), we made it to the end of the project with success. Without the discussions, conference calls and brainstorming, this work was not possible (of course, it was ;)). Sisi, you are the kindest teacher for me and many others. Thank you for giving me the chance to work at your group at GRID, UNSW, Sydney.

Thanks to my friends and colleagues at ITC. To all of you, because of the small chats at the coffee machine, ITC lunchtime, international events and food festivals, which all helped me to speak out my thoughts and relieve my stress. I would like to thank especially two persons at ITC: Roelof because of his smile every morning, which gave me energy for the rest of the day, and to Teresa, our EOS office manager who carefully organized yearly events, staff presents and many more. To my office-mates Diogo and ZhenChao for all the laughter and talks about research and life. To my neighbor's office, Parya for the small chats and cookies, you saved me from starving. To Fashuai and Mila for their support and friendship, traveling in China was one of my best research travels with you two. Thanks to my Persian friends for the Persian parties and food. To all EOS Friends Group for the best moments which we spent together, it was four years full of memories. Thank you my paranymphs, Sebastian and Caroline for supporting me until the last moment of my defense.

This work was not possible without the input of the users and partners in SIMs3D project. Kourosh and Sisi for the initiation of the project, the user committee and Dutch Fire-brigade members, especially Huib Fransen and Gerke Spaling for providing scanning sites for use cases, Robert from CGI and Bart from Cyclomedia for their feedback, and Markus Gerke and his team from TU Braunschweig for scanning of several buildings. I would like to appreciate Liangliang and Jantien from TU Delft for taking the responsibility of leading the project.

Thanks to my friends all over The Netherlands, Europe and Iran, whom their positive energy and good vibes are with me. I hope your bottomless hopefulness stays with me forever. To my homeboy friends (G5): Nima, Mojtaba, Hamed, Ardalan and Makan. To my Munich buddies: Faraz, Hessam, Amin, Soheil, Zartosht, Mehrdad and Mahyar for making me laugh. To my Sydney babes: Milton, Selma, Mitko and others. To Shahin and Sepideh for encouraging me to come to The Netherlands.

Thanks to my climbing friends from the Cube, Climbing Beasts and A+ Team, for all the travels, climbs, camps and ascends, without you I could not recharge myself for this research.

Thanks for all the late gatherings and bottom-up bottles; sincerely, you are the best: Xavi, Caroline, Martin, Claudia, JuanRi and Laura. I wish for more travels and more bouldering with you guys.

My family has endless support all over my life no matter how far I am. I want to thank you, my lovely parents, for teaching me above all how to be a human, my kind sisters Shabnam and Shideh for always caring for me and my brother Shahryar for making me laugh.

Thank you Ieva, my dearest partner, for supporting me and always being there for me. Every piece of this work is touched by your kindness.

Last but not least, thanks to all the people who bear with me over the years.

Table of Contents

List of figures

List of tables

Chapter 1 - Introduction

1.1 Background and Motivation

Urban 3D models have been developed for cities and buildings in a variety of domains such as urban planning, real estate, tourism and computer graphics. In dealing with indoor environments, there is a demand for indoor 3D models in the mentioned domains as well as for indoor positioning and disaster management. Currently, for most of the buildings, the primary available sources are floor plans and CAD information. Modern buildings and recently renovated buildings may have a 3D model in the form of a building Information Model (BIM), which is widely used in Architecture, Engineering and Construction (AEC) industries. These sources of building representation are "as-designed" and they do not always address the current status of the building. The question is what is the most efficient solution to keep the existing floor plans or 3D models up-to-date or "as-is" during the lifetime of a building.

Manual creation of 3D indoor models from floor plans is a tedious process, apart from the often-outdated status of floor plans. In recent years, there has been impressive progress in indoor data collection technologies namely mobile laser scanners, Microsoft Kinect, Google Tango. Such mobile systems provide high-quality images, point clouds and depth information in a shorter time in comparison to terrestrial (static) laser scanners. However, the output of mobile laser scanners is a massive amount of raw geometry and images which are cumbersome for users to interact with and understand. Although manufacturers of mobile laser scanners provide software and virtual tours to explore the data it is not sufficient for more complex queries and operations, for example, to calculate the area of the glass surfaces in the building. Alternatively, in this research, we aim to provide computer algorithms that enable us to reconstruct and to update indoor 3D models through automatic methods with the minimum expert intervention. As an application, this research targets using the models for disaster management in complex buildings.

The generated models should follow the standards of current indoor models (IndoorGML, CityGML, and IFC) to provide a reliable platform for the evacuation of and safety management in large buildings. In addition to a 3D indoor model, the outcome of this research will be a set of algorithms and open-source software that will be applied by Dutch emergency services (BHV) and fire brigade for emergency responses.

1.2 Smart Indoor Models in 3D (SIMs3D) Project

This research is part of the project Smart Indoor Models in 3D (SIMs3D). The SIMs3D project is part of Maps4Society (M4S) program that aims to research on smart geo-information infrastructure and innovations in geo-information domain (www.maps4society.nl). The project contributes to the goals of

Maps4Society program by addressing the research area managing big data, and application areas crisis management, smart cities, human environment and management for buildings.

The Dutch Research Council (NWO) is the funding organization of the project who brings researchers, users and companies together. Following Dutch research council and partners contribute in SIMs3D project:

1. NWO as the Dutch research council
2. Academic partners:
    - University of Twente (UT), ITC Faculty
    - Delft University of Technology (TUD), OTB Department, GIS Technology
3. Companies:
    - Cyclomedia Technology B.V. as a data provider (www.cyclomedia.com)
    - CGI Nederland B.V. as a software advisor (www.cginederland.nl)
    - Leap3D as data provider (www.leap3d.eu)
4. End Users:
    - iNowit Brandweer Nederland (fire brigade) as an end user and advisor for user cases
    - Open Geospatial Consortium (OGC) as the international standards organization and IndoorGML developer.

The academic partners cooperate closely in two phases of the project: i. The researchers from University of Twente (UT) are responsible for 3D reconstruction (geometry, topology and semantic) of indoor models from point clouds; ii. the research team from TU Delft focuses on deriving indoor spaces (considering agents, activities and resources) from indoor models generated by the UT team for the evacuation goals.

Figure 1.1 shows an overview of the main stages in the project. Data is mainly acquired by mobile laser scanners, as they are faster in scanning large spaces. The output of MLS devices is already registered and our research does not focus on point cloud registration as a research problem. In the next chapter, an overview of mobile laser scanners is provided. The *model reconstruction* phase is the main focus of this Phd research. Two other phases including data management for generating GIS models and disaster management for end user interaction are in the work domain of the other partners (TU Delft and The Fire Brigade of The Netherlands).

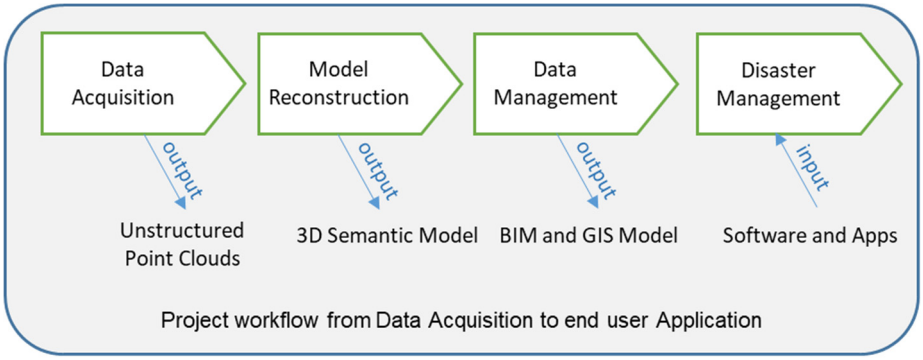


Figure 1.1. *The figure illustrates the overview of the main phases in the SIMs3D project. Data reconstruction phase is the main focus of current dissertation.*

1.3 Research Gap

The indoor environments have a high level of variety and complexity. Due to this complexity, reconstructing a faithful fully 3D model with a (semi-) automatic method is the main research problem. Most of the proposed approaches are dealing with simple structures or they are not scalable to the large multistory buildings. In the current research, we assume that the data (point cloud) is a registered point cloud and has a good quality in terms of point accuracy, because it is captured by accurate mobile laser scanners or terrestrial laser scanners. Therefore, registration problems and data acquisitions problems are not addressed as our research problem. However, dealing with the noise and incomplete data is considered as part of the open challenges. The current state-of-the-art for indoor 3D reconstruction does not address the below problems or they are in early stage of the research:

Reconstructing fully 3D models: Many of the current methods reconstruct models by assuming the same height for all ceilings, thus reconstruct a 2.5D model. A fully 3D method should be able to extract the correct angle and height of the ceilings. Besides, some of the buildings have intermediate floors or so called a mezzanine. The reconstruction methods should identify such floors and reconstruct them faithfully.

Dealing with the noise, incomplete data and glass surfaces: Unstructured point cloud always comes with artefacts and noise. The noise can be from the sensor, the registration of point sets or the SLAM algorithm (simultaneous localization and mapping). Furthermore, when using a mobile mapping system, the artefacts caused by reflective surfaces should be distinguished from the permanent structures (e.g., walls). Although MLS devices are mobile and can access a larger area than TLS devices, the presence of clutter (e.g., furniture) causes data occlusion and remains as a challenge.

Many current methods, assume a clutter free environment when modeling the interiors.

Dealing with arbitrary room layouts: Complex architectures could have very arbitrary wall arrangements. Therefore, assuming a grid layout (the Manhattan-World assumption) for the rooms means excluding many buildings from the modeling pipeline. Similarly, assuming a horizontal floor and ceiling is common in most of the current related work, while many buildings have ramps in the floor, slanted walls and sloped ceilings which need to be considered in the methodology. A state-of-the-art method should be robust to the variations in room layouts.

Scalability of the method: Proposing methods that can handle one-floor-building or several rooms is not a challenge anymore. Nowadays, MLS devices deliver a large number of images and point clouds in a short time. The proposed methodology should deal with large buildings which means more data. When it comes to the third dimension the complexity of calculation exponentially increases. Keeping such models up-to-date means to add the fourth dimension to the data (e.g., time) and it makes the scalability problem even harder. Therefore, any novel method for 3D reconstruction should be scalable to big data.

Consistency control of the models: The consistency of the reconstructed 3D models in terms of geometry, topology and semantics needs more attention in the research. Some of the methods can create a correct geometry but the topological correctness of the components is not assured. Regarding the applications, the generated 3D model should be tested against the demands for a specific application, for example if it is used for navigation, the detection of doors should be considered. Furthermore, the models should be tested against some of the current standards such as IndoorGML and IFC.

1.4 Research Objectives

This thesis focuses on 3D modeling of building interiors by dealing with the research problems discussed earlier. The main goal is to propose a pipeline for automatic reconstruction of 3D models using point clouds. As the human intervention for modeling more complex structures is inevitable, we try to minimize user interactions and to simplify it for non-expert users. The key objectives of this research are discussed in the following. More details to support these objectives are discussed in chapter 2 and the methods to satisfy these objectives will be deliberated in the methodology of following chapters.

1.4.1 Semantic labeling

Given a large point clouds from the interiors, understanding the role of each point and separating the noise from useful information are the goals of this objective. The pipeline starts with the classification of points in the classes: walls, floors, ceilings, doors, windows, furniture, and noise. Semantic labeling can be done on individual points, or segments. If segments are labeled then a semantic segmentation is the result of this objective. Semantic labeling is specifically more difficult when dealing with clutter and gaps in the data. Therefore, our objective is to suggest methods which are able to identify the semantics of a scene represented by sparse point clouds. For example, when using a laser scanner, the point clouds are missing glass surfaces. Thus, when the majority of a wall is made of glass (e.g., façade walls), only part of the wall is recognizable. The goal is to develop methods which can detect part of the object.

1.4.2 Geometric modeling

Geometric modeling is the process of describing mathematically the group of related points, segments and surfaces which form a specific shape such as a wall, door or piece of furniture. This objective develops the algorithms and methods that can deal with incomplete shapes caused by sparse data. For example, when part of a wall is identified in the point cloud as a segment, a method is required to estimate the correct extension, thickness and the normal vector of the wall. Computational geometry algorithms will be developed to reach the goal of this objective. One of the research problems which was mentioned earlier as dealing with non-Manhattan World cases should be tackled within this objective.

1.4.3 Watertight 3D model reconstruction

By water-tightness, we are referring to a model in which all the faces are connected and there are no disconnected and dangling surfaces. Similarly, neighboring spaces should be connected and no gap or sliver is justified. The method for 3D reconstruction should satisfy this objective in terms of the geometry and topology of the model. A watertight model can be safely used for indoor navigation purposes, evacuation simulation and so forth. Note that both a surface model and a solid model can be generated within this objective. Moreover, this objective aims at creating models of multistory buildings including stair cases while keeping the topology of the model consistent. Introducing a method for space subdivision, adding furniture as part of the space, and checking the usability of the model for indoor navigation are other topics contained in this objective.

1.4.4 Consistency and accuracy control of 3D models

The consistency of 3D models is neglected in related works. This objective is focusing on developing methods to control the consistency of the final models in terms of geometry, topology, and semantics when there is no ground truth. One source for consistency control is using the current standards such as IndoorGML, IFC, and ISO 19107. Furthermore, the common expert knowledge and the specific application of the model can imply more rules for checking the consistency of the model. Accuracy of the model is more related to the accuracy of the sensor and mathematical algorithms (e.g., fitting plane methods), thus can be controlled by comparison to the hand-crafted models.

1.5 Research Contribution

Recent research in the indoor 3D modeling domain has been done aiming at various research problems such as geometric reconstruction, opening detection, indoor navigation and routing. Our goal is to adapt the current state of the art and improve it for an automatic and scalable reconstruction of indoor 3D models which ensures semantically and topologically consistent models.

The main contributions of this research are:

- Reconstruction of full-3D models in challenging indoor environments with large glass surfaces, large amount of clutter (furniture) and moving objects (people).
- Modeling of interiors both for BIM and IndoorGML applications.
- Modeling of large and multistory buildings for disaster management applications. Therefore, detection of openings and clearance of emergency doors are investigated. Moreover, modeling furniture as obstacles, extracting navigable spaces in 3D and flexible navigation networks are focus of this research.
- Lifting assumptions such as Manhattan World, horizontal ceilings and floors, and vertical walls.
- Change detection in indoor environments between different epochs of scanning and separation of structural changes from temporary changes.
- Proposing a formal framework for consistency control of 3D models and their compliance with indoor standards (IFC and IndoorGML) and ISO.

1.6 Dissertation Overview

This research starts in chapter 2 with a thorough overview of the methods for 3D modeling, related standards, and indoor mapping systems. Then the objectives are addressed step by step in each chapter. The whole thesis from chapter 3 to the end can be read as a pipeline for 3D reconstruction from point clouds which starts with data collection and denoising to consistency control of

the created models. In total the data of five buildings is collected with various mobile mapping systems for use cases.

Chapter1: presents a background and motivation for the research, in addition to details about the research project which this dissertation is part of.

Chapter2: an overview of the indoor modeling methods, indoor standards and mobile mapping systems is given in this chapter. Existing indoor modeling methods and their state-of-the-art are categorized and compared. This chapter is a useful collection for other researchers who want to get familiar with indoor modeling and its research problems. Note that the material of this chapter is from the PhD proposal and it reviews the literature before 2015. The newer literature is reviewed in the related work of other chapters.

Chapter3: is proposing a novel method for identification of permanent structure (e.g., wall, floor and ceiling) in a point cloud by using the adjacency graph. As a major contribution, we show and exemplify the importance of MLS's trajectory for scene understanding in indoor environments, including detection of doors, windows, stairs and fictitious errors caused by reflective surfaces. Moreover, a mathematical morphology in 3D is applied for space subdivision. All the proposed algorithms are tested on four complex buildings which impose different challenges for 3D modeling.

Chapter4: this chapter is built on top of the findings of the previous chapter. As in chapter three a watertight mode is not yet generated, the pipeline is further developed in chapter four to generate a watertight model of multistory buildings. The third objective, watertight 3D model reconstruction, is covered in this chapter. The modeling of stairs and extraction of spaces as polyhedra are discussed in this chapter. Furthermore, a method for space subdivision, identifying and adding the furniture as a subspace and checking the consistency of the model against the indoor navigation graph are investigated.

Chapter5: is focusing on the fourth objective, consistency control of the 3D models. The chapter gives a review of the recent literature and the comparison of types of indoor models. This chapter is not just focusing on the models generated with our pipeline, but generally suggests a method to use standards and expert knowledge for evaluation of different types of 3D models. The proposed methodology does not aim to fix the corrupted models but assures whether a given model complies with the current standards in terms of topology, geometry, and semantics.

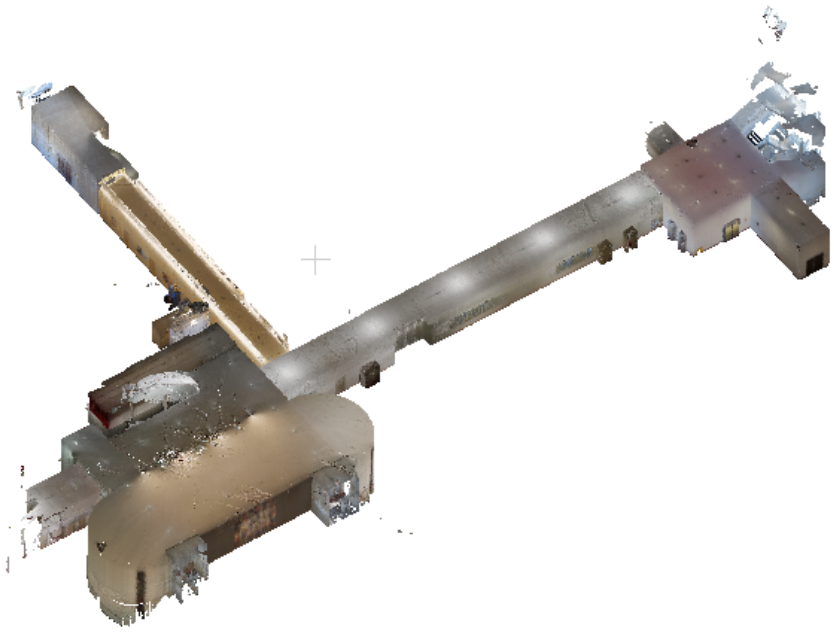
Chapter6: investigates the change detection from point clouds in indoor environments. A method is represented for separation of changes in the permanent structure from temporary changes (e.g., furniture). The data is

collected from two buildings in two epochs and changes are examined. Moreover, the application of change detection in indoor 3D cadaster is studied as the future line of research.

Chapter7: summarizes the finding of each chapter and answers the research questions by relating them to the objectives of this research. The directions and suggestions for future research are discussed in this chapter.

It should be noted that there is an overlap between chapter 3 and 4, as they are based on the journal papers.

Chapter 2 – A Literature Review of Current Indoor 3D Reconstruction Methods



2.1 Indoor Data Models and Standards

In this section we investigate current indoor models and their definition of indoor space. The justification behind this review is the ultimate application of our research output. As discussed in the introduction we generate a 3D reconstructed model that will be applied for emergency cases in evacuation of buildings. Therefore, we need to have a clear understanding of indoor model standards and specifications and try to generate a product compatible with current models such as IndoorGML, IndoorOSM and Industry Foundation Classes (IFC). There are various aspects of indoor space to study. From navigation aspects we have functional, operational and range space, from topological aspect we have primal space (topographic space) and dual space (adjacency, connectivity, accessibility) as well as semantic and spatial (topology, geometry) aspects.

2.1.1 Review of current indoor data models

With the growth of the interest in indoor mapping and indoor services in the last years the demand for standards and specifications raised. The goal of these standards (models) is to form the data collection, the maintenance and the software development for indoor environments. Currently there are four main indoor models: 1. IndoorGML, 2. CityGML LoD4, 3. IFC, 4. IndoorOSM.

IndoorGML: IndoorGML is a Geography Markup Language (GML) encoding standard defined by Open geospatial Consortium (OGC) for indoor navigation and representation. IndoorGML standard facilitates the modeling of interior space in terms of topology and semantic while avoids describing complex geometries. This avoidance is because of already existing indoor geometric models in other standards such as CityGML and IFC. The focus of IndoorGML is on two main functions of indoor environments: 1- Representing the properties of indoor space, and 2- Providing spatial reference of features in indoor space (Lee et al., 2014). Therefore, IndoorGML studies the indoor space from navigation aspects as well as the type of their connectivity to be applied for navigation and does not emphasize on building architectural components (roof, wall, …).

Definition of Indoor Space in IndoorGML: "Indoor space is defined as space within one or multiple buildings consisting of architectural components such as entrances, corridors, rooms, doors, and stairs" (Lee et al., 2014). IndoorGML's effort is to define rooms, corridors and stairs as indoor spaces, provide their spatial information and type of their connectivity in space, investigate navigation possibility regarding WiFi coverage, accessibility and functionality and does not provide information about walls, roof, ceiling, ventilation, installments and furniture. An important difference of the indoor space from an outdoor space is the indoor constraints such as corridors, rooms

and stairs. In IndoorGML, indoor constraints are considered from the following aspects;

- *Cellular space* which is defined as the smallest organizational or structural unit of indoor space, each cell has an ID, cells do not have overlap with any other cell but have common boundary, e.g., rooms. Cell position can be defined by its ID or (x, y, z) coordinates for more precise location.
- *Semantic representation* that means decompose indoor space to cells based on their semantic. The cell subdivision can represent the topography (e.g., room, door, window) of a building, available WiFi coverages, indicate security areas (e.g., check in area, crew area, boarding area), or public/office areas. In IndoorGML semantic has two purposes: 1. For classification and to define connectivity between cells. 2. For hierarchical structure and semantic interrelation (specialization and generalization)
- *Geometric representation* of 2D or 3D features in indoor space is not a major focus of IndoorGML, since they are clearly defined by ISO 19107 (ISO, 2003), CityGML, and IFC. However, there are three options to represent geometry in IndoorGML: 1. External reference to CityGML, 2. Geometry in IndoorGML as GM_Solid in 3D space and GM_Surface in 2D space, 3. No Geometry
- *Topological representation or Network representation,* the Node-Relation Graph (NRG) represents topological relationships, e.g., adjacency and connectivity among indoor objects. The NRG allows abstracting, simplifying, and representing topological relationships among 3D spaces in indoor environments, such as rooms within a building.
- *Multi-Layered Representation* is supported with IndoorGML for various cellular representation such as topology layer, sensor coverage space, topographic space in dual space or Euclidean space. This representation is useful to represent the interlayer relationships between two hierarchical levels.

CityGML LoD4: CityGML is an open data model and XML-based format for the storage and exchange of virtual 3D city models issued by Open Geospatial Consortium (OGC) (Gröger et al., 2012; Kolbe et al., 2005). CityGML supports five different Level of Details (LoD) that reflect independent data collection processes with various application requirements. LoD0 to LoD2 define standards for city modeling and building blocks. LoD3 uses a boundary representation (boundary surface) with simple geometry and texture to reflect external view of the building such as roof, entrance doors and windows. LoD4 is a supplementary model to complete building models from inside in presence of relevant data by adding interior details. For example, buildings in LoD4 are composed of rooms, interior doors and stairs (Gröger et al., 2012). The 3D City Modeling Standard CityGML and its LoD4 offers possibilities to represent interiors of buildings with their geometry, semantics, topology and appearance (OGC, 2008). Since CityGML and IndoorGML both are dealing with indoor space

they have many common standards. IndoorGML has a multilayered representation of the interior space and supports navigation networks. For example, the shortage of IndoorGML in visualization can be handled by CityGML LOD4 (Lee et al., 2014).

Industry Foundation Classes (IFC), BIM and BISDM: To improve communication within the industry, BIM users and vendors developed a data interchange standard format known as Industry Foundation Classes (IFCs) (ISO, 2018). BuildingSMART International (formerly the International Alliance for Interoperability, IAI) (buildingSMART International, 2013), has established IFC for representing building elements and their properties which are generated as object-oriented models in XML formats. A BIM is a digital representation of all the physical and functional characteristics of a building through its entire life cycle (Isikdag et al., 2007; NBIMS, 2006). However, it is a cumbersome process to keep BIM models as update as current situation of the buildings. Building Interior Space Data Model (BISDM, v3.0) is an object-oriented example of IFC models designed and supported by Esri for implementing GIS projects (ESRI, 2012).

Based on IFC definition: "a space represents an area or volume bounded actually or theoretically. Spaces are areas or volumes that provide for certain functions within a building" (buildingSMART International, 2013). A space in IFC standard can be a *space group* which is defined as COMPLEX (e.g., site, building) in the model, can be a *space* which is defined as ELEMENT (e.g., building story), and can be a *partial space* which is defined as PARTIAL (e.g., parking) in IFC model.

Since BIM models defined by IFC are for building maintenance and industry purposes, it is not trivial to adapt a BIM standard for indoor GIS targets such as indoor navigation. For instance IFC standards does not concern about the spatial relation among building interiors which is crucial for indoor navigation purposes, or interior spaces such as rooms and corridors defined by architecture components (e.g., wall, door, windows) not their functionality. (Isikdag et al., 2013) propose an approach to transform a standard BIM model to a BIM-Oriented Model (BO-IDM) for navigation and emergency cases in the buildings. Their method provides semantic information for indoor navigation goals and complex geometries interpretation.

IndoorOSM: IndoorOSM is an indoor extension of OpenStreetMap with the focus of collecting Volunteered Geographic Information (VGI). IndoorOSM is a 2D representation of indoor and does not support 3D modeling like previously described models. The information is provided by contributors and stored through tools developed by the OSM community (e.g., JOSM, Java tool for OSM). In OSM data there are three main concepts to represent map features:

1. *Nodes* that are very simple features and geo-referenced points such as trees, park bench. 2. *Ways* that are linear or polygonal geometries combined from nodes such as roads, building, region boundary and 3. *Relations* that are more complex features such as polygons with holes or complex relation between different OSM map features such as a construction site or a complex route, 4. *Tags* that are used for semantic information and it has a *key-value pair*, for instance "barrier" as a *key* could have many different *values* (curb, ditch and fence). In IndoorOSM nodes, relations, ways and closed ways represent the points of interest (POI), corridors and rooms in the indoor environments. IndoorOSM explicitly represents corridors as polygons, therefore obstacles and holes can be integrated in corridor polygons. Building floors can be presented and connected by relations and have attributes. In contrast to indoorGML that deals with indoor parts as modular cells and concerns about their functionality, IndoorOSM does not geometrically distinguish between the functionality of a building part (e.g., room, corridor, staircase etc.), therefore the mapping becomes much easier for the contributor (Goetz and Zipf, 2013). Unfortunately, in recent years there was not too much effort to complete and develop IndoorOSM standards and it has just studied in academic domain and not much practical projects have been carried out.

2.1.2 Transition from 3D reconstructed model to GIS/BIM Model for evacuation

In our research we mainly notice to transform our result to IFC or/and IndoorGML models because these two models are specifically designed and supported by respected communities for indoor applications. IFC similar to IndoorGML is an object-oriented data model for building components and they are related to each other in many aspects. It is important to understand the difference and relationship between them for better application. IndoorGML is representing the interior space by a cellular model and cells are smallest unit of the indoor, while IFC is modeling the interior by architectural components. IndoorGML can be applied for navigation purposes, while IFC is applied for building maintenance and contains components details. Therefore, we can indicate IndoorGML as our base model and enhance it by external references to IFC information. For example, when we need information about walls material and thickness IFC can easily provide such information (see Figure 2.1) (Lee et al., 2014).

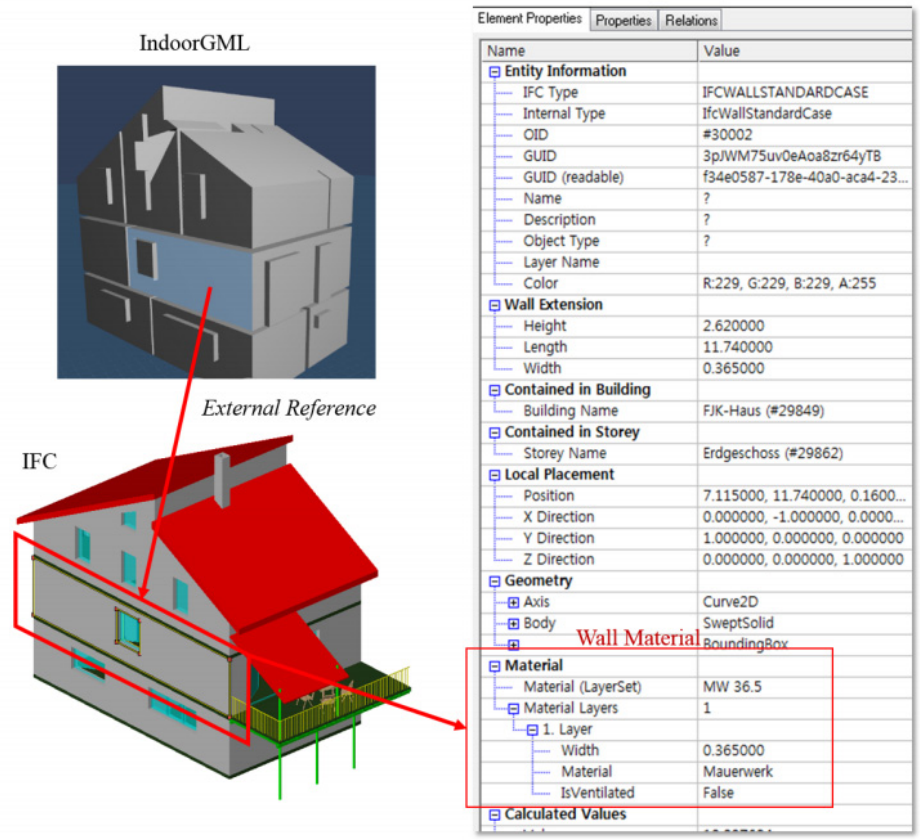


Figure 2.1. *The figure shows how IFC can act as an external reference to provide necessary information to fill IndoorGML gaps (e.g., wall parameters and material)* (Lee et al., 2014).

As a conclusion, indoorGML does not store geometry of features to avoid duplication by CityGML LOD4 and applies geometry as an external class. On the other hand, IFC does not concern about the spatial relationship among features and limits itself to the geometry and semantic information, therefore they can be used as supplement for each other. Currently there is no indoor standard that thoroughly covers all aspects of indoor environments in terms of topology, geography, semantics and to support various indoor data formats (CAD data, 2D floor plans, BIM models and point clouds). Additionally, it is not possible to provide all the necessary information to feed into a model. With this overview we enjoy an understanding of interior space and its subdivision in different indoor models that help us how to enrich and transfer our final 3D reconstructed model to an indoor model for further applications and specifically for evacuation goals.

2.2 Review of Existing Indoor Data Acquisition Systems

Indoor acquisition systems had a dramatic progress in recent years. In addition to stationary devices such as terrestrial laser scanners (TLS) there are a wide range of indoor mobile mapping systems (IMMS). While TLS systems (e.g., Leica, FARO, RIEGL) are more accurate (mm accuracy), IMMS systems (e.g., CSIRO Zeb1[1], Trimble TIMMS[2], NavVis[3] M3) are more flexible and faster for data acquisition. Another source of the data can be acquired by less accurate but low-cost mobile systems such as Google Tango and Microsoft Kinect that deliver RGB-Depth data.

In this project we intend to use IMMS systems (Zeb1, NavVis M3) because they are faster than TLS laser scanners and still deliver accurate data for our purpose (3-4 cm accuracy) and more accurate than RGBD sensors such as Kinect and GoogleTango. However, we test our algorithms on different data sources and compare the result.

ZEB1 (Zebedee) and ZebRevo: Zeb1 (Bosse et al., 2012) is a handheld 3D mapping platform constructed from a 2D laser scanner (Hokuyo UTM-30LX with 30 m range) and an inertial measurement unit (IMU) mounted on a spring platform (Figure 2.2). The system is based on simultaneous localization and mapping (SLAM) and delivers 3D point cloud from interior environment. The advantage of Zeb1 to other IMMS systems is that it has more movability and is faster for data acquisition. Other systems are not able of mapping on stairs and they need alignment of point clouds from different floors which leads to registration errors. However, for big buildings Zeb1 needs also (semi-automatic) alignment of different point cloud datasets.

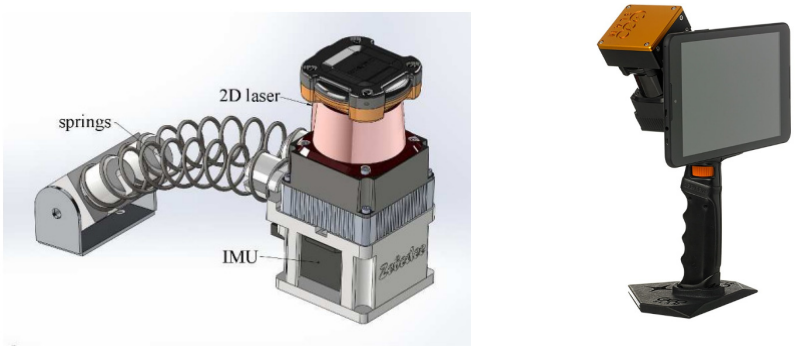


Figure 2.2. *Left: Zeb1 CAD model* (Bosse et al., 2012)*, Right: ZebRevo RT (www.geoslam.com)*

[1] www.geoslam.com
[2] www.trimble.com
[3] www.navvis.com

NavVis Trolley (M3): M3 Trolley is a mobile laser scanning platform constructed from 3 Hokuyo 2D laser scanners, 3 IMUs, 6 cameras, one battery supply and one tablet and CPU for real time processing (Figure 2.3). Out of three 2D laser sensors, two of them is mounted vertically for a 3D data acquisition on both side of laser scanner and one is mounted on head unit (on top of laser scanner and above operator height) for 2D localization and mapping. Cameras also are mounted on head unit to have panoramic coverage during data acquisition. Likewise, the system is capable of collecting WiFi data in case there is such signals in the environment. The system like Zeb1 applies SLAM method for localization and mapping and delivers 3D point cloud in addition to HD panoramic images from the environment. As mentioned before, the M3 Trolley is not able of mapping in stair cases and for acquiring high quality images operator needs to have a low speed. The height of the device is adjustable but during mapping it should be set above the height of the device operator to avoid any occlusion for the cameras.

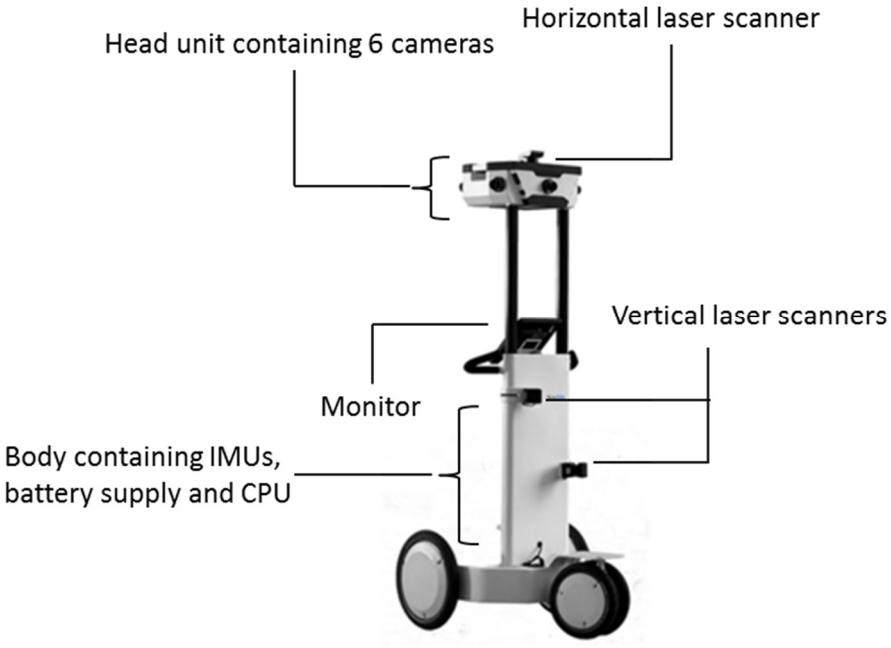


Figure 2.3. *NavVis M3 Trolley (image from www.navvis.com)*

Both systems use Hokuyo UTM-30LX laser scanner which has 30 m measuring range, 270-degree field of view and 50 mm accuracy in 10 m to 30 m range (Figure 2.4).

RGBD sensors and Microsoft Kinect: Microsoft Kinect2 is a range camera based on Time of Flight (TOF) method. Range imaging devices are low cost and most affordable 3D data acquisition systems. RGBD sensors constructed from an IR laser projector, an IR camera, an RGB camera and a 3-axis accelerometer

for device orientation. The depth resolution of the data at a distance of 5.5 m is more than 8 cm (Khosravani, 2016). In spite of their low cost, the application of them for 3D mapping depends on the level of accuracy and details we need to reach. They are vastly used for gaming and gesture tracking (e.g., Microsoft Xbox 360) as well as augmented reality and robotics, but they are not suitable for mapping in large buildings because the depth of sensors is not more than 8-10 m.

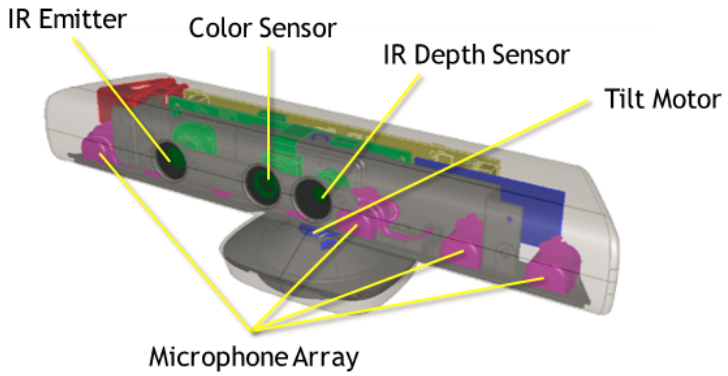


Figure 2.4. *Microsoft Kinect (source: msdn.microsoft.com)*

As mentioned before in our research we mainly use Zeb1 point cloud, for color point cloud NavVis Trolley and terrestrial laser scanners (Leica, FARO) are another source of data.

2.3 Review of Indoor 3D Reconstruction Methods

We sort indoor reconstruction methods (from point cloud) to four categories. This classification is regardless the data acquisition methods and principally discusses the methods for indoor reconstruction. 1. planar-based reconstruction, 2. Volumetric-based reconstruction, 3. Mesh-based reconstruction and 4. Indoor scene interpretation and semantic labeling. The last item is mainly dealing with level of details in 3D reconstruction not the reconstruction itself. Needless to say, that the selected literature could have overlap in the concept, for instance semantic labeling could be the result of either planar approach or volumetric approach.

2.3.1 Planar-based reconstruction

Planar based reconstruction methods require plane primitive detection by applying methods such as least square, region growing, RANSAC and alpha shapes. The results are polygons that reconstruct indoor space. Most of planar-based reconstruction methods rely on perpendicularity of walls and construct good results in Manhattan World cases (regular Cartesian structure also referred as Manhattan grid (Coughlan and Yuille, 1999)). Chen and Chen (2008) present an approach for planar regions detection in façade of the

buildings from sparse scanned range data and reconstructing polyhedron. The authors detect planes intersections. Boundary detection performed by projecting clustered points on the 2D plane and edge extraction algorithms. Although their sample data and proposed pipeline is performed for outdoor scenes, it is applicable for indoor scene and plane detection in indoor scenes. Sanchez and Zakhor (2012) propose an automatic system for planar 3D modeling of range scanner point cloud being inspired by Chen and Chen (2008) approach. The authors first classify the points based on their normal to floor, ceiling, walls and remaining. The normal vector angle threshold for ceiling and floor detection is less than 15 degree and for walls less than 45 degree. Walls are detected in X and Y direction. Assuming ceiling and floor are parallel to the x-y plane and walls are perpendicular to x-y plane. Therefore, this is one drawback of their method since they just detect walls in x, y orientation. Then they employ a RANSAC (Schnabel et al., 2007) method to detect planar primitives and their spatial extension which generates wall polygons and their orientation. Additionally, authors use a stair case model to determine stairs in indoor space. First, they fit an inclined plane to detect staircase ramp and then by fitting stair case model they detect steps and number of steps. The staircase detection method is a good approach for dealing with scalability challenge (detection of small-scale structure relative to the scene scale). The final model does not include doors and openings and is not tested against cluttered data (Figure 2.5).

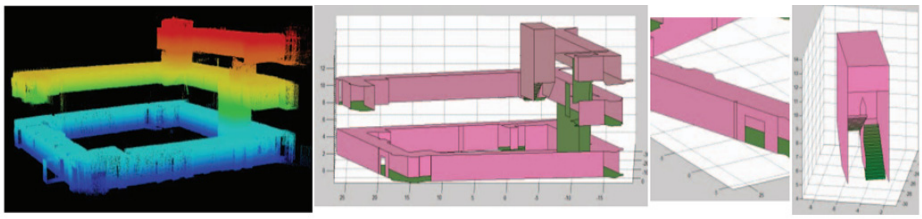


Figure 2.5. *Input data on left and reconstructed planar 3D models in middle and right. As the figure illustrates planes and staircases are detected* (Sanchez and Zakhor, 2012).

(Budroni, 2010) apply a sweeping plane method to detect vertical and horizontal segments in a Manhattan world case. The sweeping algorithm is a discrete method characterized by steps determined according to the input point cloud density. After detection of walls, floor and ceiling with the assumption of perpendicular walls, the authors employ a cell decomposition technique using detected segments and decompose the interior space to inside and outside based on the homogeneity criteria and density of point cloud. Horizontal plane is divided into cells using a half space modeling (also called half-space primitives) and straight lines as primitives. In the next step by a split-and-merge approach those cells that contain enough points are merged to shape the ground plane. This method also is not applicable to non-Manhattan world

and the resilience to missing data and cluttered data is not discussed. Moreover, the authors do not address the extraction of more interior details such as doors and staircases.

Another planar approach proposed by Okorn et al. (2010) extracts ceiling and floor by projecting onto the vertical (z) axis. The remaining points are projected on x-y plane to determine edges by Hough transform and label them as walls. A slicing plane in heights intervals (50 cm and 10 cm) and counting voxels generate histogram of clutters and permanent structure from point clouds which finally leads to classify the points to ceiling, floor, walls and clutter. The authors do not provide the structural relation between segments or classified points, openings are not detected and the final result could not be interpreted as an indoor 3D model but a floor plan.

Adan and Huber (2011) and Xiong et al. (2013) build on the previous approach by Okorn et al. (2010) and in the former work they focus on reconstructing walls under clutter and occlusion and in the latter they improve it by reconstructing a semantically rich 3D indoor model. Through a ray casting and learning approach they propose so called occupancy map that label wall surface as empty (e.g., windows) and occupied (e.g., wall surface) and occluded which is the result of clutter. Their approach fails in some cases such as arched shape windows and where windows and doors are surrounded by moldings or divided by frames and casings. Also, when doors are closed it could be labeled as occupied instead of empty. In section 2.3.3 we discuss detection of openings in details.

Mura et al. (2014) developed a cell decomposition method on cluttered point cloud from a non-Manhattan World environment. Their approach can be considered as a combination of planar and volumetric based methods. The author focused on detecting rooms by clustering point cloud from scans per room (each room at least has one scan position). Therefore, rooms that are not scanned from inside cannot be reconstructed properly. To reconstruct walls that are occluded by clutter they apply ray casting methods similar to previous work and their result is precise for cluttered and non-Manhattan World environment. For detection of wall candidates they extract vertical planes and build a 2d map by intersecting wall candidates and build a cell complex in 2D (similar to Oesau et al., 2014). Finally, the individual room polyhedra with the topology relation will be generated as 3D model (Figure 2.6). In their approach they do not detect openings or focus on semantic labeling. Moreover, low-level ceiling or tilted walls and non-horizontal ceilings cannot be reconstructed.

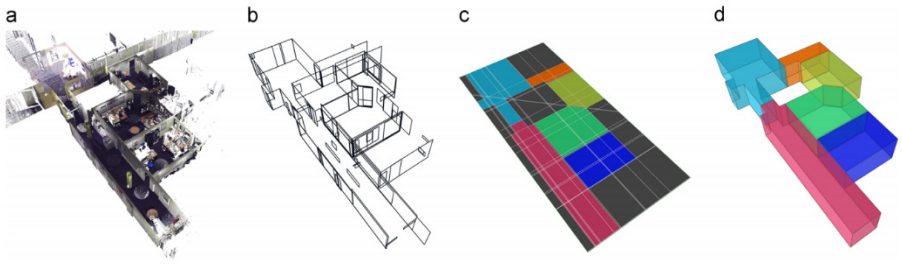


Figure 2.6. *The main phases of Mura et al. (2014) algorithm: (a) cluttered point cloud, (b) wall candidates (c) assignment of cells into individual rooms, (d) final individual room polyhedra*

In a work by Becker et al. (2015) a combined shape grammar is applied to reconstruct interior from point cloud. The authors define interior structure as two main subdivisions: 1. rooms, 2. hallways and per each structure they apply a specific grammar to reconstruct interior environment. The extraction of wall candidates for split grammar from point cloud is not described in their approach. The usage of a priori probability and context aware probability for frequency occurrence of a rule can track the neighborhood relationship between rooms and is useful for converting the model to a BIM model. The generated model is presented by planar surfaces (without openings) and for parts of the building that there is no data, the authors generate synthetic model based on regularity and shape grammar.

One shortage of planar-based algorithms is that it is problematic to convert them into BIM models that apply volumetric concept to provide indoor services. For instance, the connection of interior spaces such as adjacent rooms and topology relationship among interiors are not defined or cumbersome to determine because there is no detection for interior spaces (such as corridor, rooms and indoor divisions) and indoor spaces are encapsulated by surfaces (such as walls, doors, floor, etc.). On the other hand, planar-based approach is powerful enough for dealing with noisy and occluded data and reconstruct complex indoor environments, because as discussed it is possible to reconstruct and connect the surfaces together to generate a watertight model.

2.3.2 Volumetric-based reconstruction

Jenke et al. (2009) propose an algorithm for detection of cuboids in indoor environments from point clouds with the assumption that interior area is made of intersected cuboids. To fit a cuboid model into point clouds, first they need to detect planar primitives. Based on the efficient RANSAC algorithm by Schnabel et al., (2007) they detect planes in the data. They define unknown parametric cuboids with 9 parameters (3 for scale, 3 for translation and 3 for rotation) and for detection of cuboids they need to figure out how many cuboids are optimal to cover the interior point clouds and which data point is

represented by which cuboid. To answer these questions authors, reconstruct a graph which introduces the adjacent planes with distance threshold. To solve the 9 parameters for each cuboid the authors find the plane orientation from normal vector and plane extension. For each subgraph with perpendicular planes a cuboid will be candidate and the best fit with at least 5 planes and highest score of points on each face will be added to cuboids list (figure 2.7). Their method is robust to noisy and sparse data but is not applicable for non-Manhattan World cases. Another drawback is the assumption of presence of 5 planes for cuboid detection which is difficult to satisfy in cluttered environment as well as holes in the data like doors and windows will be completely filled by their algorithm.

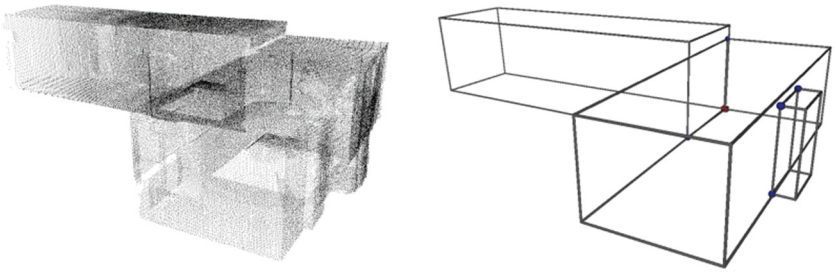


Figure 2.7. *Left side: Input data and right side: reconstructed model from cuboid primitives detection and shape graph* (Jenke et al., 2009)

Xiao and Furukawa (2014) propose iterative Constructive Solid Geometry (CSG) to generate a 2D model and with stacking 2D models on top of each other a 3D model will be generated. Their input data is a point cloud dataset acquired by a mobile laser scanner. As the 3D volumetric primitive, a cuboid is chosen and as 2D primitive rectangular candidates. Rectangles may be detected through double, triple and tuple line segments in the horizontal slice (see Figure 2.8). Dominant horizontal slices are chosen to split the 3D space into 2D slices for detection of rectangular primitives. Eventually texture from images and wall model (view dependent walls with front and back faces) will be added to the CSG model to reconstruct the texture. Since rectangles and cuboids are chosen as primitives their approach has a lack of efficiency for non-Manhattan-World cases. However, in their final result they present also non-Manhattan World interiors. The authors do not focus on detecting openings and doors and they just aim to reconstruct walls. Their approach is scalable and they compare their result with two other approaches. But the authors do not propose any method for quality control of their result and accuracy control is performed manually.

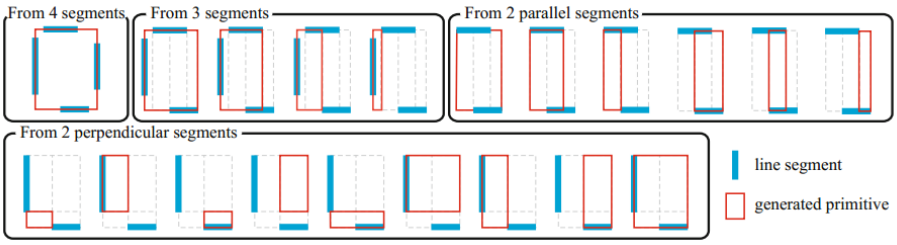


Figure 2.8. *The figure represents line segments that used to detect rectangle primitives. Rectangles can be composed of 2,3, or 4 line segments.* (Xiao and Furukawa, 2014).

In a recent work by Oesau et al. (2014) they apply space partitioning and primitive extraction from input point cloud to reconstruct indoor space. Similar to the previous approach they slice the space into horizontal slices and by Hough Transformation detect the lines to apply space partitioning. An empty/solid space labeling is formulated as an energy minimization and solved using graph cut. The final model is extracted as the union of all cells labeled empty (Figure 2.9). Line extraction includes three main steps: 1. Filtering to filter out clutter by projecting the points on a horizontal plane. 2. Line fitting to generate line hypothesis for each point representing the local wall direction. This method is useful to detect non-Manhattan World cases and arbitrary wall directions. 3. Clustering, the points are locally clustered to wall segments and wall segments are globally clustered into wall direction through Hough Transform. Authors apply ray casting method, to label empty or solid cells (Figure 2.10).

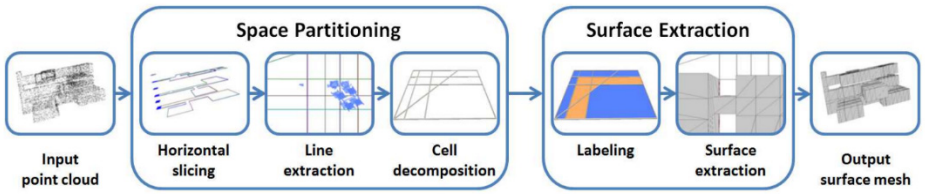


Figure 2.9. *Space partitioning from point clouds initializes by horizontal slicing and then cell decomposition. Through ray casting cells are labeled to empty and solid. Solid cells reconstruct permanent structure. Finally all horizontal slices stack on top of each other to form the 3D model* (Oesau et al., 2014).

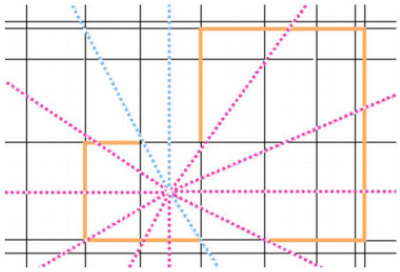


Figure 2.10. *Ray-casting method is applied to define solid and empty cells. Orange lines are edges (walls), black lines are cells, odd number of intersections (pink lines) represents that the point is in an empty cell and even number of intersections if the point is in a solid space (blue lines)* (Oesau et al., 2014).

They performed their method on real and synthetic data. Their approach is applicable for arbitrary wall direction (non-Manhattan World) and the only assumption is vertical wall and the result has more wall details than previous approach proposed by Xiao and Furukawa (2014). The proposed approach is robust against noisy and cluttered data but the authors propose no solution for detection of windows, doors and openings or stair cases. Their solution is tested on multilevel buildings. Input data is Kinect data and they apply CGAL library for part of the implementation. The main contribution of their work is detection of small details and niches in the walls. Circular shaped walls can be reconstructed with many small line segments. They check the accuracy of their result with a synthetic model using a Hausdorff distance algorithm and they reach the accuracy of 2.3 cm from the result to the ground truth.

In another different approach, Khoshelham and Diaz-Vilarino (2014) apply Palladian Grammar to reconstruct a Manhattan World environment. The grammar is defined by selecting a unit cuboid as starting shape and three rules including 1. place the cuboids, 2. connect cuboids and 3. merge cuboids. The first rule uses transformation parameters to define the size and location of the cuboids. The authors use a histogram to extract some of the cuboid parameters and define points-on-ceiling and points-on-wall indices to apply second and third rule. In the shape grammar section, we discuss in details their approach. Their approach is a good example for using the shape grammar to reconstruct the indoor environments as a volumetric method. However, it has two main drawbacks, first they just tested in Manhattan World cases and choosing cuboid as primitive limits the reconstruction to perpendicular structure. Second, they apply their method on a simple two-story dataset without any clutter which makes the selection and detection of cuboids less complex. In occluded and noisy data their approach has a shortcoming: some rooms and interior spaces will not be detected.

Ochmann et al. (2016) apply energy minimization with arbitrary wall detection to reconstruct volumetric walls and rooms from point clouds. Their method starts with clustering point clouds per room by merging the scans from the same room. Then vertical planes are detected and transferred in 2D plane and all wall candidates are intersected which is a drawback of their method because at the end some excess walls will be detected. A planar graph will be generated from intersected wall candidates whose edges are segments of possible walls and vertices are possible locations where walls are incident. Detection of rooms is based on assignment of all connected components with identical labels and edges between different rooms are labeled as walls. Therefore, each wall should have two faces and is reconstructed as a volumetric object with a centerline. In this manner it is possible to measure and define the thickness of walls and area of rooms. For opening detection authors use ray casting method from scanner position to the measured points. The author did not discuss how walls which are not sensed from both sides (because of occlusion or access problem during mapping) will be reconstructed. Additionally, their method is strongly based on assigning scans to rooms based on position of the scanner and is not applicable for mobile laser scanners (Figure 2.11). Also, for buildings with long corridors and rooms that scanned from outside (with big glass window and open doors) it is not straightforward to apply their method. However, presenting walls as volumetric objects is useful for BIM applications. Finally, the topological relation between spaces and rooms is reconstructed which is very important for a consistent model.

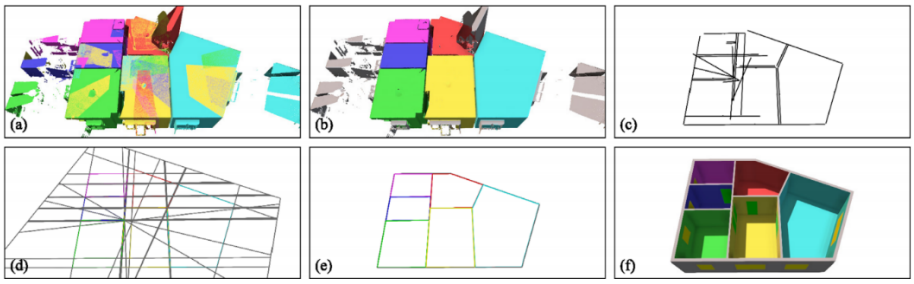


Figure 2.11. *(a) input point cloud, each color represents the assignment of points to scans; (b) segmentation result and refined points per scan for each room; (c) projection of vertical planes in 2D; (d) all lines are intersected and candidate walls are derived and represented in different colors. Space partitions are labeled as inside and outside. (e) edges that encapsulate rooms remain. (f) Final model with walls, doors (green) and windows (yellow)* (Ochmann et al., 2016).

2.3.3 Mesh-based reconstruction

Mesh-based methods are mainly applied for surface reconstruction, object recognition and 3D rendering. It has the advantage to be scaled from small features such as objects in a room (Izadi et al., 2011); (Newcombe and Davison, 2010) (Rusu et al., 2009) to larger features such as buildings (Frueh

et al., 2005). Rusu et al. (2009) reconstruct 3D model from point clouds in a kitchen environment. The authors apply a hybrid map including points, triangle meshes, 2d polygons and geometric shapes for different processes such as 3D collision detection and object classification. The result of their work is three classes of objects: vertical planes (walls) with 0.98 accuracy, horizontal planes (floor, ceiling, tables) with 0.97 and furniture candidates (cupboards, drawer, kitchen appliances) with 0.91 of accuracy. Frueh et al. (2005) develop a method for façade reconstruction from mobile laser scanners and digital images by generating meshes and depth images. The authors generate foreground as occlusion objects and background as building facades. Depth images are generated from the point cloud as are applied for processing and presenting data from 3D scanner and then the holes in the background will be filled by interpolation. To our knowledge mesh-based methods are not applied for a thoroughly indoor 3D reconstruction such as volumetric and planar based methods. The reason could be that meshes are applied for surface reconstruction, quick rendering and filling holes but they are not practical for semantic labeling and topology relationship in interior spaces. Additionally, because of complex structure of indoor (e.g., presence of small objects and large objects in the same scene) and clutter building meshes is not straight forward.

2.3.4 Grammar based reconstruction

Grammar and shape grammar as modeling and reconstruction techniques have been applied in architecture, computer graphic and engineering for many years. The term *grammar* is widely applied in the human language, programming languages, compilers, computer interpreters and text processing by creating a grammar for a language. A grammar is a set of rules that enumerates the sentences of a language. As an example Turing machines that are applied in cryptography use a form of grammars (Chomsky, 1959a). The term phrase structure grammar for analyzing and understanding languages was introduced in 1956 by the linguist Noam Chomsky (Chomsky, 1956). Shape Grammar uses the same concept as the phrase structure grammar. Stiny and Gips (1971) gives a formalism for shape generation in paintings as 2D shapes and sculptures as 3D shapes. Since painting and sculpture have location and scale on the painting canvas or sculpture frame, their specifications can be described by a grammar. The Palladian Grammar is introduced by Stiny et al. (1978) for Palladio's building architecture. Stiny sets rules to reconstruct different composition of Palladio building in an architectural language. He studies the geometry aspects of Palladio's art and proposes eight stages to generate the plan. The main conspicuous feature of Palladio's villa plans is their bilateral symmetry design and most of them are laid on a single axis in a two-dimensional Cartesian coordinate system. Plans are generated with respect to the north-east axis of this coordinate system and conventionally walls parallel to the x-axis has 'east-west' orientation and wall parallel to the

y-axis has the 'north-south' orientation. In other work by Khoshelham and Diaz-Vilarino (2014) use Palladian grammar concept to reconstruct a 3D volumetric model from point clouds for a two-story building. They use a cube as initial shape and apply transformation including translation and scale parameters. Since their method is applied for Manhattan-World structure similar to Palladian they do not apply rotation in their rules. Based on Palladian grammar for interior space wherever there is no wall between rectangles they use *connect* and *merge* rules to realign walls and reconstruct interior. Let's assume: S is starting cube and H is a transformation consists of a translation and scale. Ri represents rules and Ni denotes non-terminal shapes and T terminal shapes.

Instant Architecture and Split Grammar: Split Grammar is an extension of parametric set grammar for automatic reconstruction of building solids and facade in urban planning and architecture (Wonka et al., 2003). In *Instant Architecture* article the authors introduce this novel grammar to automate the process of urban design for a variety of building styles. The authors claim that the power of split grammar lies in the restriction of allowed rules that allows a control on derivation process while it keeps it simple. These grammar rules will be designed by introducing three main techniques:

1. Introducing *split grammar* for design and derivation of shapes.
2. Introducing *control grammar* as a simple context-free grammar to avoid random selection of rules and put them in an order based on the designer or architectural principles. In other words, control grammar as an external factor is used to calculate and distribute attributes to the shapes generated by the split grammar.
3. Introducing an *attribute matching system* for rule selection and control randomness of generated shapes based on the user specification.

Basic Shape: Basic shapes are simple building blocks of the grammar, e.g., cuboids, cylinders (made up of polygonal segments) or prisms. Basic shape is an attributed, parametrized labeled shape that manipulated by grammar. To formulate it (Wonka et al., 2003): "A basic shape *b* is given by $b = < s, P, V >$, where s is a simple shape centered on the origin, *P* is a set of three labeled points defined by the intersection of the positive coordinate axes with the (imaginary) faces of the shape, and *V* is a symbol from a vocabulary *N'* with arbitrary attributes attached to it, where a subset *T'* $\subseteq$ *N'* forms terminal symbols. A simple shape in this definition is a shape that contains all lines defined by the edges of a closed, convex 3D geometric object."

Split Definition: "The vocabulary of a split grammar is the set $B = \{ f(b) | b \text{ is a basic shape}, f \in F \}$ (note that $f(b) = f(s), f(P), V$). The set of allowable transformations F will be the affine transformations. A split is the

decomposition of a basic shape into shapes from the vocabulary B" (Wonka et al., 2003).
Split rule: $a \rightarrow b$ where *b* is generated from splitting *a* shape.

Conversion rule: $a \rightarrow b$ where *b* is a transformation shape of *a.*
"One notable difference between the split and the conversion rule is that in a split rule, the elements in the sets a and b fill the same volume, whereas in a conversion rule, this is not necessarily the case."

Two important problems that authors deal with them are:
Problem 1: How the texture and material can be handled by the system? How the appearance should look like?
Problem 2: How the appropriate matching rule can be selected among several matched rules?

The authors aim to select the best matched rules among the candidate rules. Since here the authors are using the attributed symbol selection of matched shape it is easier than problem in computer design approaches with shape grammar. Both solutions are associated with the attribute attached to the grammar symbols. Three ways for attribute propagation are proposed: 1 and 2 are straightforward because they apply settings manually, while the first applies settings on starting symbol shape, the second one propagates attributes from parent shapes to the children; 3. Control grammar is the 3rd way of assigning attributes in a mechanism to keep harmony and consistency in the building.

As a comparison with Palladian Grammar approach in *learning the language of indoor architecture* (Khoshelham and Diaz-Vilarino, 2014) the selection of geometric parameters for initial shape was done automatically with processing the point clouds and using histogram. But in Instant Architecture (Wonka et al., 2003) for a higher level design (e.g., style of the building) we have a symbol for which we can define attributes manually. However, this comparison is far away from the practice because the former approach is a bottom-up and data-driven method and the latter is top-down approach.

Selection of a rule among several matching rules should be done cautiously and should meet two criteria: 1. The derivation produces coherent and plausible result, 2. the results meet user goals. To achieve this, the attributes in the rule will be compared with the attributes in the current grammar symbol and the rule with the best match will be selected. Keep in mind that the attributes will be copied from parent shapes in a split grammar and will be checked through a control grammar. The authors (Wonka et al., 2003) also use below terms and formulas to implement their approach:
1. A deterministic matching function M_{DV}

2. A stochastic selection function
3. Maintaining a history of rule selection
4. Exclusion of the rule by interval or value

Generally, the definition of attributes for rules and symbols should consider the symmetry, building type, or façade style or the level of complexity in the façade. This helps the diversity of the design.

Here we give an example of using split grammar for a simple building façade (Figure 2.12): given a building solid with height h>12.5 and width w>17.5 and depth d>12, we design the façade for the ground floor and entrance. Notice that uppercase letters are non-terminal objects and lowercase letters are terminals. Symbols assigned to each shape are written inside them. We use Subdiv as subdivision function and Split as a split function to implement the grammar. The function *f(xyz, wdh)* applies the split rule on one of the xyz axis with the intervals of *wdh* stand for width, depth and height and the objects in the bracket { } can be terminal or non-terminal shapes.

R1: BUILDING → Subdiv (Z,0.3,3.5,3,3,3) {band|GROUNDFLOOR| FLOOR|FLOOR|FLOOR}
R2: GROUNDFLOOR → $Split_{hoz}$ (X, 4.5,4,4.5,4.5) {ROOM|ENTRANCE| ROOM | ROOM}
R3: ENTRANCE → $Split_{ver}$ (Z, 0.0, 2.4, 0.6) {∅ |extdoor| wall}
R4: ENTRANCE → $Split_{hoz}$ (X, 0.5, 3, 0.5) {wall|extdoor| wall}
R5: ROOM → $Split_{hoz}$ (X, 2.25, 2, 2.25) {wall|WINDOW|wall}
R6: WINDOW → $Split_{ver}$ (Z, 0.2, 2, 0.2) {frame|glass|frame}
R7: WINDOW → $Split_{hoz}$ (X, 0.2, 1.60, 0.2) {frame|glass|frame}

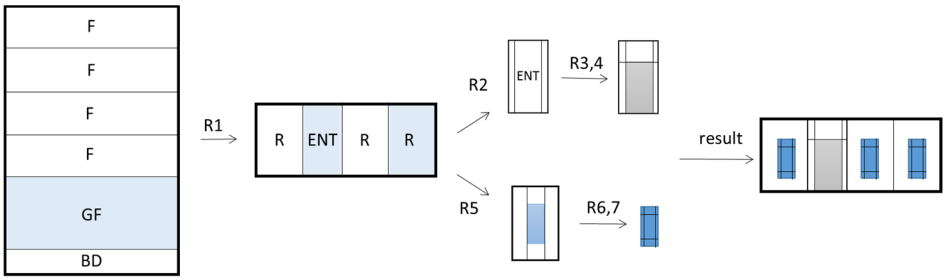


Figure 2.12. *Represents a split grammar and applying them on a building façade. The blue color areas show the matched object for applying the rule in each step, and upper-case letters such as GF, F, BD, ENT, R represent the symbols per shape. The result is a ground floor with an external door (gray color) and windows (dark blue).*

Applications of split grammar: Mueller et al. (Müller et al., 2006) in procedural modeling of buildings apply split grammar with CGA grammar to generate 3D detailed model for complex buildings and 3D city models. Authors

use *repeat* and *component* split. Repeat split tiles a specific element along with defined axis with the given intervals. Component split uses a parameter and type as input to split elements into their composite elements. For instance, *Comp ("faces") {A}* creates a shape with symbol A from original three-dimensional shape. Split grammar also is applied for façade reconstruction from images (Müller et al., 2007) and LIDAR data (Becker, 2009). Becker et al. (Becker et al., 2013) applies split grammar combined with other grammars such as L-System to reconstruct indoor of the buildings. The authors classify the indoor environments to corridor and non-corridor (or hallways and rooms) and using split grammar for non-corridor areas including rooms. The split grammar splits rooms based on 6 different kinds of rules by using the function $split^{space}(\alpha_i, d_i)$ where space means the split applies to non-terminal shapes and α_i is the orientation of split and di gives the distance value for the split. In Figure 2.13 you can see some of the split procedures.
1. SingleSplit, 2. RepeatSplit, 3. StringSplit, 4. MultiSplit, 5. Merge, 6. Instantiation

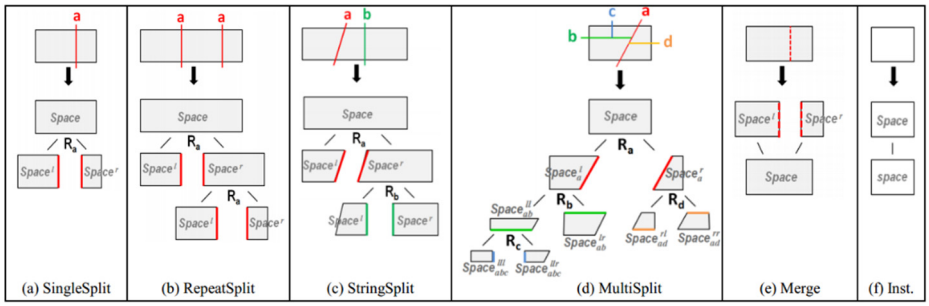


Figure 2.13. *Six split rules in* (Becker et al., 2013).

(Becker et al., 2015) define interior structure as two main subdivisions: 1. rooms, 2. hallways and per each structure they apply a specific grammar to reconstruct interior environment. For reconstructing hallways, a L-System grammar is applied and for room structure a split grammar. The main contribution of their work is extracting the grammar from the data and avoid setting grammar rules by an expert since this could be a complex and time-consuming task. However, to define grammar automatically from the data they should build their rules on many assumptions which makes their approach implausible for various building types. For example, for assigning L-System rules and instantiation authors suppose hallways are parallel to the main axis of the building and are connected, while it is not easy to extract the axis of the building especially for round or complex shapes. Additionally, the shell of building should be available to control the growth of L-System rules during production process. For instantiating split grammar first, the building should be partitioned to hallways and non-hallways parts which is error prone and not trivial. Then by a set of single and multi-split rules non-hallways can be divided

to rooms. Their approach is not tested on non-Manhattan World cases and the studied case has mainly perpendicular structure.

2.3.5 Indoor scene interpretation and labeling

Other indoor 3D reconstruction approaches are devoted to scene understanding and semantic labeling which have widely been studied in the domain of robotic and computer vision. For indoor 3D reconstruction we do not need to reconstruct small objects (kitchen appliances, objects on a table), however, we need to detect specific objects such as furniture, stair cases, pillars, tables, shelves, closets, doors, windows, etc. Therefore, it is important to have an overview on related work in this domain. Izadi et al. (2011) in a project called KinectFusion reconstruct features in a room acquired by a Kinect camera while the user may interact with the scene (in front of the sensor). The Kinect camera continuously has 6 degree-of-freedom (DOF) pose which is tracked by the system for SLAM and fuses live depth data from camera into a single global 3D model in real-time. Real-time reconstruction is implemented through parallel processing on GPU. When a user is interacting into the scene the camera tracking locks onto the background and ignores the foreground user for camera pose prediction (to avoid localization errors). Foreground data can be reconstructed independently from 6DOF. However, this method can be error prone for prolonged user interactions into the scene. Their method is scalable for bigger scenes like a room but they do not focus on labeling doors and windows in a room. In next section we discuss some of other approaches for semantic labeling and scene understanding in details.

2.3.6 Detection of openings

Detection of openings is a principal step in indoor reconstruction and is also necessary for our evacuation goals. Identifying doors and windows and extracting their geometry is the aim of opening detection. Even to enrich the model semantically respecting the openings we are able to define windows in ground floor as an escape route or in idealistic level defining which side doors can be opened. Status of the doors (open or closed) during data collection also is an important element for detection of the openings (Diaz-Vilarino et al., 2015). Adan and Huber (Adan and Huber, 2011) propose a method (we address in this literature as *occupancy map,* Figure *2.14*) for detecting openings from point cloud under occlusion and clutter. Here briefly we explain the steps:

1. Detecting the wall surface in the point cloud using a histogram analysis approach explained in (Okorn et al., 2010)
2. Each detected surface is modeled by a set of voxels bounded by rectangles that occupy the surface.
3. Occlusion labeling: each voxel assigned occupied (F for full), empty (E) and occluded (O) labeled (Figure 2.14 f). This is the main contribution of their

work. Labeling occupied or empty is easy based on whether or not there was a surface. For occlusion testing, the authors use a *ray tracing* method (cast a ray from position of the scanner to the measured point) to identify the empty voxel if is truly empty space or is because of occlusion.

4. Make a range image from the surface by projecting depth values on the surface (Figure 2.14 b).
5. Detecting of opening edges using a Canny's algorithm and Hough transform (Figure 2.14 c).
6. Modify occluded voxels by new empty voxels detected in previous step.

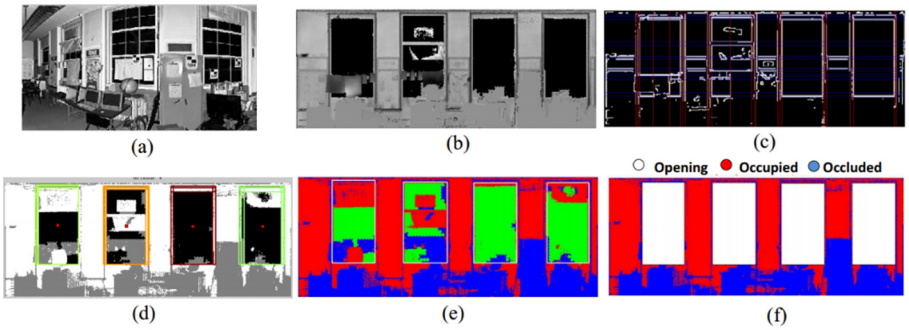


Figure 2.14. *Opening detection. (a) reflectance image of wall surface, (b) depth image, (c) edge detection by Hough Transform, (d) detecting of opening candidates by SVM detector, (e) prototype openings, (f) final labeling, openings are in white color, blue is occluded parts and red is solid surface* (Adan and Huber, 2011).

The accuracy of their approach is 2.5 cm for 36% of the opening's boundaries, and 64% have less accuracy around 5.39 cm with a standard deviation of 5.70 cm. Their algorithm correctly detects 93.3% of openings in their sample data (with 10 cm voxels). This approach delivers a good accuracy. However, it should be improved for non-rectangular openings. Additionally, it is applicable on the floor and ceiling to detect possible openings especially in the ceilings. More processing needs to be performed to label the openings as windows and doors (for instance doors bottom touches the ground). Also closed doors can be labeled as occupied and therefore will not be detected. The problem of reflection in data is not resolved in this approach. Ray casting method for opening detection have been used in many publications (Mura et al., 2014; Ochmann et al., 2016; Previtali et al., 2014; Xiong et al., 2013) in recent years. The input dataset in all cases had been captured by terrestrial laser scanners where the position of laser scanner is fixed during scanning, while in mobile data acquisition the position of laser scanner is changing and instead of several positions there is a trajectory. Therefore, we require a dataset that defines which points are measured from which poses of the scanner considering the timestamp. However, providing such data is possible but casting rays from a trajectory (instead of individual station) toward measured points can be subject to a research problem.

Díaz-Vilariño et al. (2015) apply imagery to detect closed doors in the images. Their approach to some extent solves the problem of closed doors in previous approach by using images. Main steps are as below:

1. Generate an orthoimage
2. Convert true color image to gray scale to detect edges by a Canny operator
3. Use a rectangular model shape with Generalized Hough Transform to find candidates in the image (the approach is invariant to scale changes)
4. Use point cloud to exclude false doors detection

The candidate doors are also tested against other parameters such as min and max width and height of the door and doors are assumed to be vertical. The results are 95% accurate for two case studies and 85% for two other cases also under clutter and occlusion. Failures are because of presence of similar structure and texture in the image.

2.3.7 Detection of obstacles

In indoor environments obstacles are interpreted as objects that are elevated from the floor plane such as chairs, desks, steps, columns, etc. Therefore, by identifying the floor plane, the detection of obstacles is not complicated. Detecting and labeling obstacles is important for evacuation purposes and is a subject of robotic domain (Espinace et al., 2013; Manduchi et al., 2005; Zhang et al., 1994). Simply by making a height histogram (projection of 3D data on vertical axis) we are able to identify objects which are above the floor plane and not touching the ceiling. But this is not always the case becasue a huge chandelier also can be assumed as an obstacle. Okorn et al. (2010) and Xiong et al. (2013) describe more details how to extract clutters which we interpret as obstacles in our research. In Figure 2.15, you can see ceiling and floor are separated from intermediate objects in the histogram by their height.

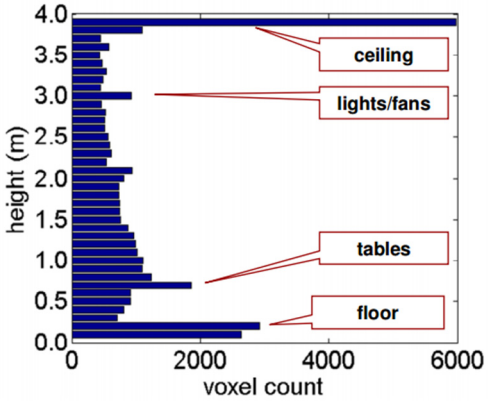


Figure 2.15. *The histogram presents z values and the count of voxels. The picks in the histogram are showing ceiling and floor and other bars show the clutter at different elevations* (Okorn et al., 2010).

In case of presence of imagery or Kinect-RGBD it is possible to extract objects from images and label them as obstacle. Additionally, we can get benefit from imagery for texturing objects. Koppula et al. (2011) and Anand et al. (2012) use Microsoft Kinect for two different indoor environments: 1. office, 2. home. In total 52 scenes from offices and homes are experimented and they reach an overall performance of 84% in the office including labeling wall, table (top, leg, drawer), chair, monitor, keyboard and CPU and 73 % at home including labeling bed, wall, floor, pillow. In their approach they consider three main properties to capture: 1. visual appearance. 2. local shape and geometry 3. geometrical context. In our research we may not require too detailed labeling of the features but as a subdivision of space (e.g., navigable and non-navigable area). Furthermore, identifying which features are dynamic obstacles (e.g., desk, chair, furniture) and which static (e.g., pillars, steps, stairs) is another critical point.

2.3.8 Using imagery and 2D data to enrich 3D model

Imagery besides point clouds can make great improvements in detection of objects and enhancing the indoor reconstructed model. For instance, in section 2.3.2 we explained the extraction of closed doors from images. Also, in section 2.3.3 we remarked that extracting features from images may improve significantly the obstacle detection approach. A profound complete example of using imagery and point cloud for indoor 3D reconstruction in huge environment is presented in *reconstructing the world's museum* (Xiao and Furukawa, 2014). They use a hand trolley to collect data including point cloud and imagery and deliver the data from different floors separately. Then they use a Constructive Solid Geometry (CSG) approach with a cuboid as volumetric primitive to reconstruct 2D and 3D model (we will explain this in details in section for indoor 3D reconstruction). Then the imagery is applied to texture the walls. The major contribution of their work is that their approach is applicable for huge number of images with a big dataset of point cloud. However, the registration between different datasets remains as a problem and is a source of inaccuracy in the texturing because of the presence of double walls at some segments. Hedau et al. (2009) and Wang et al. (2013) recover indoor scene spatial layout in cluttered rooms from single images. Authors first make a training dataset from many single images (308 images) and manually label clutters. Then they apply this prior knowledge to find walls (left, right, front side of the perspective image), floor and ceiling that are occluded by the furniture. Their process finds long lines in the image and finds three vanishing points (the point where sets of 3D parallel lines meet in 2D) to nominate several boxes in the image (Figure 2.16). Each candidate box can represent the edges of walls and floor and ceiling. The important task is identifying the most correct box based on its rank. To generate a candidate box, pair rays will be extended from vanishing point 1 and vanishing point 2 and the four intersection will be connected to the third vanishing point. The main difference

between the two approaches in Hedau et al. (2009) and Want et al. (2013) is that in the latter the user does not need to label clutters manually because this is a cumbersome task. Instead they introduce latent variables which present the clutter characteristic and are called latent because they are never observed in the image. Then Wang et al. (2013) apply energy minimization method consists of appearance energy function (e.g., color and texture) and clutter energy function whose parameters are learned from training data. They reach an error-rate of 20.1% without the need of user to label the clutters in the image. Their approach can be applied for visual concept reconstruction in the image such as occlusion and object detection.

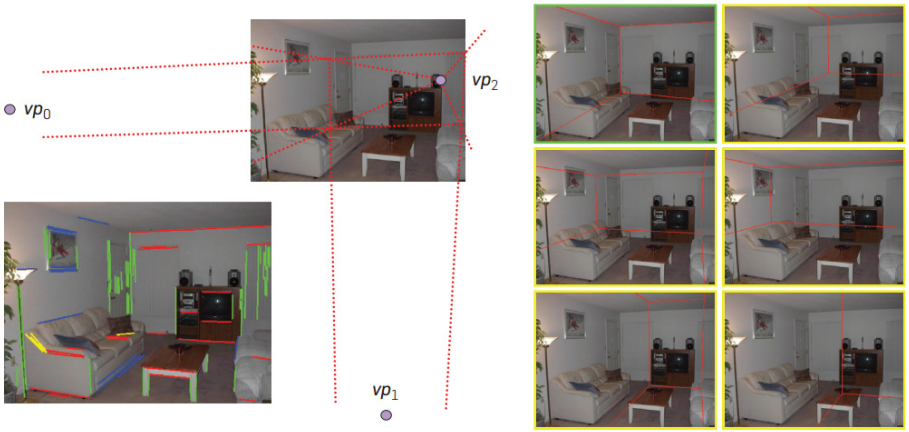


Figure 2.16. *Lower-left: three groups of lines (shown in R, G, B) corresponding to the three vanishing points. There are also "outlier" lines (shown in yellow). Upper-left: A candidate box layout is generated. Right: Different candidate box layouts (in yellow frames) are generated in the same way, and the hand-labeled true box layout (in green frame)* (Wang et al., 2013).

2.3.9 Semantic labeling

Semantic labeling of point clouds in indoor environments aims to specify that each segment containing points has what kind of identity such as kitchen appliances, furniture, objects on a table, wall, door, window, pillars, stairs and so forth. In the previous sections we explained some challenges in indoor environments namely opening and obstacle detection. When we are able to detect these features, we label them for further applications. However, the level of details that we semantically label depends on the data quality, type of the data (point cloud, RGBD and image) and application. For instance, for indoor modeling for evacuation we opt for more general feature detection (e.g., furniture, door, stair) not high level of details. Mattausch et al. (2014) perform an object detection and classification method on large cluttered point clouds from indoor spaces. First, they implement a patch growing algorithm where nearly-planar patches will be detected. Then a rectangle is fitted to each patch to define geometry characters of patches (area, width, length, height of

centroid). Fitting the rectangle primitive is too crucial in their algorithm because in the following step they apply these characteristics to cluster similar patches. Also, for similarity measure the authors take into account spatial consistency for adjacent patches to filter out dissimilar patches. A segmentation is also performed on the clusters to define each class of object in the entire point clouds. Their method does not use any training data and is an unsupervised segmentation approach. Their algorithm also generates a library of detected objects which can be used to replace similar objects that are occluded and incomplete with the object from the library. One drawback of their algorithm is the presence of 9 parameters that they fix from the data by trial and error. Additionally, because they apply planar primitive for patch detection their algorithm is not responsive to uneven features (keyboards, plants, lamps). Anand et al. (2012) based on the scene context (e.g., co-planarity, convexity, visual similarity, object co-occurrence, proximity) train a model for labeling features in indoor environments. The input data is captured by RGB-D and from that authors infer contextual relation (e.g., keyboards and monitors should be on the table), geometric relation (e.g., on-top-of, in-front-of) and local visual shape and appearance of the features. To reach this goal authors generate an undirected graph which represents segments adjacency and one dependency graph which will be exploited for object associative features. For instance, table leg, table top and table drawer are dependent features. Through a maximum-margin approach authors learn parameters from training examples to label segmented objects. As result authors detect features such as keyboard, table drawer, chair, monitor, wall, tabletop and printer. The accuracy of their result is between 93% to 96%.

2.3.10 Consistency Control

Since evacuation of buildings and emergency services in indoor environments are main objectives of this research, consistency of the reconstructed model is crucial. For instance, if the accessibility of rooms and interiors to an (emergency) exit is defected because of topology errors in the model then the indoor navigation for evacuation fails. Gröger and Plümer (2009a, 2009b) propose a thorough study for preserving geometric and topology consistency in 3D city models and specifically for indoor navigation purposes. The authors introduce a *"Constraints Store"* that prevents errors to contradict with consistency-reachability rules. For instance, a new split face should not traverse a door inserted with previous rule. These constraints are generated by rule applications and explicitly formalize the concepts of adjacency, reachability and semantics. The constraint store maintains three main rules: 1. Equality Constraints *Equals* (F_1, F_2) that means two boxes share the same face or wall. 2. Aggregation Constraints *Aggr*(F, F_1, F_2) means face F is the aggregation of face F1 and F2 where F is split with a face or wall. 3. *Inside* (F_1, F_2) that means face F1 is inside face F2, for instance a door inside a wall. These topology rules follow (Egenhofer and Franzosa, 1991). However, the proposed

approach by Gröger and Plümer has no example of real case study or a data-driven approach and applying their methods on point can be challenging.

2.4 Open Issues and Conclusion

We can summarize the current research problems regarding 3D indoor reconstruction as follow:

1. Extracting geometry and regularity: as mentioned in the literature (Oesau et al., 2014; Sanchez and Zakhor, 2012; Xiao and Furukawa, 2014) many approaches are based on primitive detection (plane, lines, cuboids, rectangles), therefore extracting geometry and topology of more complex features is limited to these primitives. Another reason that makes extraction of geometry characteristics difficult is the level of scale in indoor environments. Parameters that could be learned from the data for small-scale structure (e.g., stair, window, door, jagged walls) hardly can be applied for large scale structure (e.g., ceiling, big walls, floor). Regularities in the structure are mainly based on the assumption that floor and ceiling are horizontal and walls are vertical. To our knowledge no literature until now investigated reconstruction in architectures with inclined and curved structure while in airports, museums and concert halls there are diversity of such a constitution.

2. Improving 3D reconstruction methods in non-Manhattan World cases: the current researches produce promising result for Manhattan-world indoor structures because of the rectangular shape of the indoor space. Either using shape grammar (Stiny et al., 1978); (Khoshelham and Diaz-Vilarino, 2014) or other methods (Adan and Huber, 2011; Budroni, 2010; Tang et al., 2010) the authors rely on the perpendicular shape of the indoor structure. Some of the researchers tackled this problem by detecting direction of walls by means of cell decomposition and primitive line extraction (Ochmann et al., 2016; Oesau et al., 2014) but the data is simple structure or not cluttered environment. To our knowledge until now there is no entire work done on complex interior buildings. Most of the researches performed reconstruction on simple interior structure including regular rectangular shapes. Applying current methods such as CGA grammar, split grammar, L-system may solve the problem to some extent but currently the available software such as CityEngine accepts 2D floor plans or shapefiles as input not volumetric data such as point cloud.

3. Dealing with clutter and occlusion in the data: the footprint of clutter and occluded data can be noticed in all problems. Because always part of the data can be occluded in indoor environments extraction of spatial characteristics are strongly dependent on the consistency and

completeness of the input. Most of the current literature for indoor environments tackle this problem through occupancy map solution (Adan and Huber, 2011) but few of them deal with reconstruction of occluded data. Moreover, none of the literature have tested their approach on mobile laser scanner data using the benefits of the trajectory.

4. Adapting rules from the data structure in the grammar: this is one of our fundamental tasks in applying shape grammar for indoor reconstruction. Based on the fact that most of the current literature define rules by an expert through a manual process (Wonka et al., 2003); (Müller et al., 2006); (Schwarz and Müller, 2015), therefore learning these rules from the input data is a challenging task. Until now there is few literature that investigates this problem (Müller et al., 2007) (Becker et al., 2013); (Khoshelham and Diaz-Vilarino, 2014); (Adao et al., 2014). While detecting canonical geometry relations such as parallel, coplanar, collinear, orthogonal and co-angular is crucial for rule grammars and reinforcing semantic labeling, it is intensely dependent on the quality of the data and absence of clutter.

5. Selection of appropriate rule and attributes for the selected shape: by solving the problem of shape extraction from the data, we run into a new problem which is assigning correct parameters and attributes to detected shapes in the data. That means set of rules can be applied to the same shape and lead to different derivations. Different products for the same data are not something that we expect in a real environment. Maybe it is not a problem in virtual modeling but for evacuation goals we rely on an accurate model derived from the data. Additionally, rules that are inferred from the data cannot be applied freely on any shape but require sophisticated process to avoid generation of defective structure. Wonka et al., (2003) and Becker et al., (2013) suggest probability and constraint solutions for this problem. Therefore, the output result can be improved by refining current solutions.

6. Extracting semantic information from indoor laser scanner point cloud: semantic information could have a wide range in our project from extracting and labeling features in indoor space (e.g., furniture, door, window, wall, stair, …) to identifying subdivisions of space (navigable space, functional space, agent-based space). Regarding the input data, extracting these semantics has limitations, for instance the functionality of rooms from point cloud. Furthermore, because of the presence of clutter and complex indoor structure identifying features could be challenging. Since most of the buildings which are subject to this research have complex structure (airports, university, museum, hospital, concert halls) it raises

the need of smart and flexible algorithms that are compatible for different type of buildings.

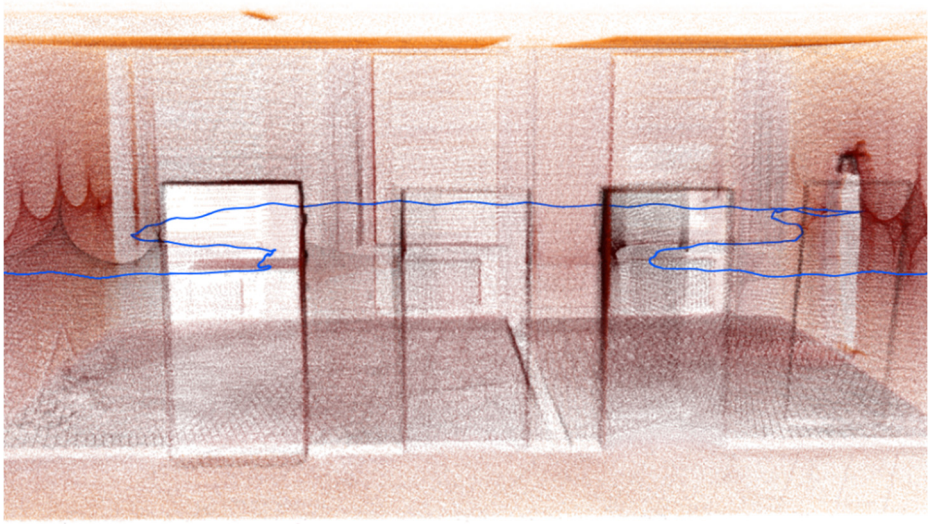
7. Consistency control of the model concerning topology and semantics

8. Consistency control of the generated result has not been addressed in many literature. Gröger and Plümer (2009a, 2010) investigate this issue with shape grammar. The logical assumption for using shape grammar or other planar or volumetric approach for 3D reconstruction is that the generated model topologically and geometrically should be correct because of the rule-based approaches. There seems to be no compelling reason to argue the consistency of the result especially because of the lack of ground truth models in the most cases. For example, since we are using the rule-based approach, we can claim that two rooms are adjacent by means of a wall and there is no gap/sliver in their geometry but we do not prove or control if all rooms have connectivity to the main exit. This issue is possible to be checked by designing more rules but it has not been investigated closely in recent literature.

9. Automatic accuracy control of the model against the ground truth: in the current literature for indoor 3D reconstruction automatic accuracy control of the result always has been an issue, either because of the incomplete ground truth data or the inefficiency of approaches. Even with the assumption of presence of a complete and up-to-date 3D model in other standards (indoor GML, IFC) until now no automatic thorough approaches have been proposed to control a 3D indoor model generated from point clouds against current status of the building. In this project we are focusing on automatic methods for accuracy control, however, manual checks should be considered for cases that automatic control is not possible. More accuracy control will be performed by TU Delft team during designing evacuation models.

It should be mentioned that all problems should be tackled in an automatic process and should be scalable to fulfil research objectives. In next chapters, we discuss these problems with respect to the project objectives.

Chapter 3 - Semantic Interpretation of Mobile Laser Scanner Point Clouds in Indoor Scenes Using Trajectories *



* This chapter is based on:

Nikoohemat, S., Peter, M. S., Oude Elberink, S. J., & Vosselman, G. (2018). Semantic Interpretation of Mobile Laser Scanner Point Clouds in Indoor Scenes Using Trajectories. *Remote sensing*, *10*(11), 1-23. [1754]. https://doi.org/10.3390/rs10111754

Abstract

The data acquisition with indoor Mobile Laser Scanners (MLS) is quick, low-cost and accurate for indoor 3D modeling. Besides a point cloud, an IMLS also provides the trajectory of the mobile scanner. We analyze this trajectory jointly with the point cloud to support the labeling of noisy, highly reflected and cluttered points in indoor scenes. An adjacency-graph-based method is presented for detecting and labeling of permanent structures, such as walls, floors, ceilings, and stairs. Through occlusion reasoning and the use of the trajectory as a set of scanner positions, gaps are discriminated from real openings in the data. Furthermore, a voxel-based method is applied for labeling of navigable space and separating them from obstacles. The results show that 80% of the doors and 85% of the rooms are correctly detected, and most of the walls and openings are reconstructed. The experimental outcomes indicate that the trajectory of MLS systems plays an essential role in the understanding of indoor scenes.

3.1 Introduction

Due to recent improvements, mobile laser scanners (MLS) became an effective means of data collection in urban and indoor scenes. Indoor mobile laser scanners (IMLS) are capable of quick data collection at a lower cost than terrestrial laser scanners (TLS). Three types of common IMLS devices can be distinguished: Handheld devices (e.g., Zeb-Revo), push-cart systems (e.g., NavVis Trolley) and backpack sytems (e.g., Leica Pegasus). Thanks to the MLS mobility, these devices can achieve a more complete coverage of cluttered scenes in a shorter time.

In addition to generating point clouds, IMLS systems generate a trajectory of the sensor positions, which is a valuable source for the scene understanding. The trajectory can be linked to the point clouds through the time stamp. In robotics, some researchers have exploited the robot's trajectory to classify indoor places from both the trajectory and point clouds (Mozos, 2010; Mozos et al., 2005). However, the trajectory can be more useful in understanding indoor scenes. In our research, the trajectory is used for the detection of openings, separating building floors and the detection of stairs. For example, the trajectory as a set of scanner positions is used for occlusion reasoning to discriminate between openings and occlusions. Furthermore, wall planes that are intersected by the trajectory can be used to detect doors. Points that belong to stairs can be extracted by using the trajectory of the stairs. Obviously, detecting stairs by trajectory analysis is only applicable for laser scanners that are operable on stairs, i.e. for backpack and handheld systems.

In addition to using the trajectories, our research introduces a method for detecting the permanent structure, such as walls, floors, ceilings, and stairs from point clouds. Most current indoor reconstruction methods are limited by assuming vertical walls and a Manhattan World (Becker et al., 2015; Budroni and Boehm, 2010; Ikehata et al., 2015) to reduce the complexity of 3D space. Few works deal with arbitrary wall layouts (Mura et al., 2014; Ochmann et al., 2016; Turner and Zakhor, 2014), but they are restricted to vertical walls and horizontal ceilings. Our method detects slanted walls and sloped ceilings exploiting the adjacency of permanent structures, based on the assumption that there is less clutter near the ceiling in indoor environments. Additionally, the arbitrary arrangements of walls (non-Manhattan-World) will be handled in this work. Our pipeline for semantic labeling of permanent structure uses detection of planar primitives labeled as wall, floor and ceiling, and their topological relations.

Room segmentation is another research problem in large-scale indoor modeling. In the literature, different approaches, such as Voronoi graphs, cell decomposition, binary space partitioning and morphology operators (Bormann

et al., 2016) are suggested for 2D and 3D room segmentation. Some of these methods have limitations, such as Manhattan-World constraints and vertical walls. Most of the room segmentation methods rely on the viewpoint (Mura et al., 2014, 2016) and require scanning with a TLS in each room (Ochmann et al., 2016). However, as opposed to one scanning location per room, mobile laser scanning systems produce a continuous trajectory and assigning points per room based on the scan location is not possible. Similar to our method for trajectory analysis, (Elseicy, 2018; Zheng et al., 2018) exploit the trajectory for space subdivision. Although their focus is only on space subdivision and simple structure, their results support our motivation of using the trajectory for interpretation of point clouds.

In our pipeline, a novel method is suggested for partitioning interior spaces based on voxels and exploiting unoccupied space. Besides knowing the room layout, information about the doors, walkable space and stairs supports navigation planning. Therefore, voxels are used to identify the walkable space and the trajectory to identify the stairs and doors.

Reflective surfaces, such as glass, complicate the analysis of indoor point clouds. Such surfaces cause the appearance of "ghost walls" in the data that do not exist in the real building. Ghost walls may incorrectly be detected as part of the room layout and sometimes result in an incorrect room segmentation. The problem of transparent and specular surfaces is addressed in robotics applications (Foster et al., 2013; Koch et al., 2017). We tackle this problem by comparing the time stamps of points with the time stamp of the nearest trajectory parts before starting the wall detection process. Using our method, some of the noise caused by the reflective surfaces can be corrected.

The contribution of this work is introducing methods for using the sensor trajectory as a valuable source for semantic labeling of IMLS points clouds. The result is not a watertight model, although it extracts a coarse 3D model from heavily cluttered data with the presence of noise. Some of the methods presented in this work (e.g., door detection) are limited to mobile laser scanner data because of use of the trajectory. Most of our methods are applicable to TLS point clouds as well. For example, methods for the wall, floor, and ceiling detection can be implemented on both RGBD data and TLS point clouds. The proposed methods are tested on three types of mobile laser scanner data: Backpack systems, trolley systems (push-cart), and handheld devices. The rest of the chapter explains the related work, and data collection, followed by the methodology for permanent structure detection, space partitioning and door detection in sections 3.4, 3.5, and 3.6, respectively. The results, evaluation and conclusion are described in sections 3.7 and 3.8.

3.2 Related Work

In this work, several known problems are addressed in the domain of indoor modeling, such as detection of permanent structures, room segmentation, opening detection and dealing with noise and reflective surfaces. For each of the cases, the state of the art is reviewed in the following subsections.

Data acquisition: The first step in any indoor modeling pipeline from real data is collecting data and pre-processing to clean up the data. The main sources of the data for indoor modeling in large scale are point clouds from LiDAR Systems or RGBD Systems. LiDAR systems could be TLS devices, such as RIEGEL VZ, FARO FOCUS, or MLS devices, such as the Google Cartographer backpack, Leica Pegasus backpack, NavVis M3 Trolley, VIAMETRIS iMS3D and Zeb-Revo and Zeb-1. RGBD cameras, such as Matterport and Google Tango, are another source of data for indoor modeling. However, RGBD cameras have less accuracy in comparison with TLS or MLS. (Lehtola et al., 2017) present a thorough review of various indoor mobile laser scanners based on Simultaneous Localization and Mapping (SLAM). According to their study, TLS systems have the highest accuracy, but less flexibility, than MLS for indoor data acquisition. Backpack and handheld systems have the most mobility, but at the cost of a lower accuracy than trolley and TLS devices. The trolley devices are constrained to near-flat surfaces; they cannot be used on staircases and steep slopes. RGBD cameras are accurate enough for indoor 3D modeling purposes and scene understanding, but not surveying goals. In our research, we only use the point clouds from laser scanner systems, such as the data from NavVis M3 Trolley, handheld Zeb-1, Zeb-Revo and a prototype backpack system (ITC Backpack) based on the proof of concept of 6DOF SLAM (Vosselman, 2014).

Reflective Surfaces: The first step after data acquisition is dealing with noise and artefacts. Often these artefacts come from transparent and specular surfaces. Koch et al. (Koch et al., 2017) investigate this problem to identify specular and transparent surfaces during scanning with a SLAM robot. Their goal is to identify and purge the corrupted points from the data on the fly or by post-processing. The intensity of the reflected laser pulse and the material of the surface (e.g., aluminum surfaces, glass, and mirror) often have unique distribution for discrimination of the transparent and reflective surfaces. However, the detection of transparent surfaces is more challenging because of the characteristic of the material. In another study by Foster et al. (Foster et al., 2013) the authors employ both the geometry and the angle of incidence between the laser and the surface during scanning. They suggest that in a particular angle of incidence, specular and glass surfaces are visible to LiDAR and glass can be detected.

Approaches to indoor reconstruction either from LiDAR point clouds or RGBD images can be categorized to three following categories:

Indoor Volumetric Reconstruction: These approaches involve volumetric primitive detection (e.g., cuboid) and are often computationally more expensive than grammar-based and Binary Space Partitioning (BSP) methods. However, volumetric methods have a better representation of non-Manhattan-World structures, slanted and rounded walls and sloped ceilings. Xiao et al. (2014) employ inverse constructive solid geometry (Inverse CSG) to build the 3D model. A 3D CSG is generated by iteratively stacking 2D CSG models. Each 2D CSG model is produced with many line segments that form various rectangle primitives. Their approach cannot model rounded walls because their hypothesis is based on extracting rectangles. Mura et al. (2016) apply the piecewise-planar detection and encode the adjacency of planar segments into a graph that represents the scene.

Indoor grammar-based Reconstruction: One popular modeling approach, especially in regular environments, is adopting a (shape) grammar (Gips and Stiny, 1980; Stiny, 1982; Tran et al., 2017), Lindenmayer Systems (L-systems) (Peter, 2017) or (inverse) procedural modeling (Bokeloh et al., 2010; Martinovic and Van Gool, 2013; Müller et al., 2006; Wonka et al., 2003) approaches for interiors. Becker et al. (Becker et al., 2015) use a combination of split grammar and L-system to reconstruct a 3D model for as-built BIM (Building Information Model). Their approach has a different view of the indoor space, since it divides the building into two main partitions as corridors and rooms. In another innovative approach, Ikehata et al. (2015) introduce an indoor structure grammar consisting of eight rules. Their approach is limited to Manhattan-World structures and 2.5D space. In (Gröger and Plümer, 2010; Khoshelham and Diaz-Vilarino, 2014; Tran et al., 2017) authors apply simple examples of shape grammar to reconstruct indoor models that are clutter free.

Binary Space Partitioning (BSP) or cell decomposition: In the domain of indoor reconstruction, many researchers use BSP to tackle the problem of room segmentation. In indoor space partitioning, BSP is a piecewise-planar approach that subdivides the space in 2D cells and as an output generates a 2.5D model (Budroni and Boehm, 2010; Ochmann et al., 2016; Oesau et al., 2014; Previtali et al., 2014). In using BSP, 2D approaches have the assumption of both vertical walls and horizontal ceilings, which is a shortcoming of the 2D-BSP. If BSP is implemented in 3D, it results in a 3D reconstructed model (Boulch et al., 2014; Chauve et al., 2010; Mura et al., 2016), where the limitations of vertical walls and horizontal ceilings can be lifted. Additionally, BSP methods are able to assign the 2D or 3D cells of space partitions to the rooms based on the viewpoint and ray-casting. However, it requires scan positions per room with enough overlap to make the room labeling process possible. The main

problems of BSP approaches are the restriction of viewpoints, the emergence of ghost primitives and the computation cost for labeling the cells as inside and outside.

Opening Detection: Among the work for the indoor reconstruction of points clouds, some of them (Adan and Huber, 2011; Diaz-Vilarino et al., 2015; Díaz-Vilariño et al., 2017; Ikehata et al., 2015; Ochmann et al., 2016; Quintana et al., 2018; Rusu, 2013a; Xiong et al., 2013) consider the problem of opening detection (doors and windows) and in their final model reconstruct the openings. Doors are essential elements for route planning and space subdivision. In our definition openings are not just limited to doors, but any opening in the wall that could be passed by individuals and connect two spaces. However, in cluttered environments and because of the presence of the furniture and obstacles, many walls could have data gaps that can be falsely considered as openings. Adan and Huber (2011) propose an occlusion test to detect windows in the walls. Ikehata et al. (2015) use a grammar rule to add a door in the wall between two separate rooms such that the walls are connected through a doorway. Therefore, in their pipeline, the addition of the doors is after reconstruction of the room. In a recent work Diaz-Vilarino et al. (2017) use the trajectory for door detection followed by an energy minimization to separate rooms with the known location of the doors. However, their example is a simple and clutter-free dataset. Another approach for door detection especially in the robotic domain is using images besides point clouds for detection of semi-open doors and closed doors. Quintana et al. (2018) and Diaz-Vilarino et al. (2015) present such techniques for detecting closed doors from images and point clouds.

Similar to our approach, Elseicy et al. (2018, 2018a) use the trajectory for semantic enrichment of indoor spaces. The authors exploit the fact that doors are the connecting elements of two spaces. By detecting the doors using the trajectory, it is possible to partition the trajectory and the space. This approach is only suitable for interiors with low level of transparent surfaces. Similarly, Zheng et al. (2018) analyze the scanlines to find local geometric regularities and to detect openings. By using extracted information, such as doors from scan lines, it is possible to segment the trajectory to associated spaces and subdivide the space. Both approaches may have poor results in environment with a large number of transparent surfaces or when the operator of the laser scanner has inconsistent behavior.

There is a large body of literature regarding scene understanding in small-scale indoor spaces, such as the detection of objects in a kitchen (Rusu, 2013b; Rusu et al., 2009) for robot operation or in a bedroom (Silberman and Fergus, 2011; Wolf et al., 2015). In large-scale there are works by Armeni et al. (2016) for scene parsing, Mattausch et al. (2014) using a similarity matrix in cluttered

environment and Qi et al. (2016) using deep learning for object classification. Some other works in the domain of indoor 3D reconstruction from point clouds use semi-automatic approaches to generate BIM models (Barazzetti, 2016; Jung et al., 2014; Macher et al., 2017) or stochastic methods to make a hypothesis on generating floor plans (Loch-Dehbi et al., 2017).

Our work is innovative in terms of dealing with glass reflection problems using mobile laser scanners and exploiting the potential of trajectories as a supplementary data produced by MLS systems. This work can be further improved to reconstruct a complete 3D indoor model from complex structures. Furthermore, the generated navigable space can be used for route planning in 2D (e.g., pedestrians, wheelchair and robots) and 3D space (drones).

3.3 Data Collection and Pre-processing

The data for this research is captured with three different mobile laser scanner systems. Each system has advantages and disadvantages in terms of mobility and accuracy. The data is collected by means of NavVis Trolley, Zeb-1 (Bosse et al., 2012), ZebRevo, and ITC Backpack, a backpack system that is developed in our department and is in the stage of proof of concepts (Vosselman, 2014), see Figure 3.1. All three systems use Hokuyo UTM-30LX as the laser rangefinder sensor.

According to the Hokuyo UTM 30LX specification[1], the accuracy of the sensor in indoor environments for the range between 0.1 to 10m is ±30mm, and in the range of 10 to 30m is ±50mm. Backpack and handheld systems have more mobility than push-cart systems (trolley) and are able to scan stairs, while push-cart systems deliver a better quality of point clouds in comparison to handheld systems (Lehtola et al., 2017).

In Section 3.1, the data and the trajectory from various MLS devices used in this research are presented. In Sections 3.2 and 3.3, the process of identifying corrupted points caused by reflective surfaces and then the segmentation process are explained.

[1] www.hokuyo-aut.jp

Figure 3.1. *From left to right: Our prototype backpack system (ITC backpack), NavVis Trolley, Zeb-1 and Zeb-Revo.*

3.3.1 Point Clouds and the Trajectory

One advantage of MLS systems over TLS devices is that in addition to the point clouds, they provide the laser scanner trajectory. The trajectory is a dataset containing a discrete suite of the device's location during data acquisition and is synchronized with the point cloud. Therefore, by means of time stamps stored in the trajectory and point clouds, it is possible to know which points are collected from which location in the trajectory. In our experiment, a 0.01 second time resolution is used to group points from each scanner position. Figure 3.2 shows the trajectories of various MLS devices. The z-value of the points in the trajectory varies depending on both the scanning system and the height of the operator for a backpack or a handled device. Because mobile devices are moving in the environment, there would be less occlusion, but more artefacts caused by glass surfaces. The next section explains how to deal with such corrupted points in the data.

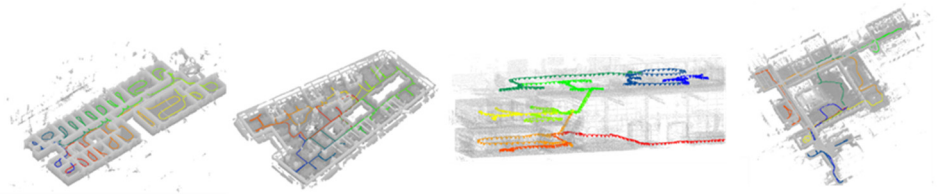


Figure 3.2*. The trajectory of various mobile laser scanners that are colored by the time. From left to right: ITC Backpack, NavVis Trolley, Zeb-1 and Zeb-Revo.*

3.3.2 Identifying the Artefacts from Reflective Surfaces

In addition to the noise introduced by SLAM, another source of the noise is reflective and transparent surfaces, such as glass and specular metals. The MLS devices that are used in our experiments do not use a multi-echo sensor similar to the one is used in Koch et al (Koch et al., 2017). In our process, the trajectory and ray casting are exploited to detect and remove these artefacts.

According to (Foster et al., 2013), when a laser beam strikes a glass surface three cases will happen: (i) Most of the light (almost 92%) is transmitted through the glass , (ii) some light is reflected back under a specular angle, and (iii) a small percentage of the light is scattered. If part of the glass surface appears in the point cloud it is because the incidence angle of the beam is near the perpendicular angle to the surface. Therefore, in the presence of a lot of glass surfaces in environments, three types of objects would be present in the data:

1. Objects behind the glass if the laser beam is transmitted. Since almost 92% of the light is transmitted through the glass, a lot of objects behind a glass surface are measured through the glass. However, these points are less reliable than a directly measured object.

2. Objects in the front of a glass surface which are reflected in the glass. In this case, the glass is acting like a mirror or a specular surface. Therefore, in the point clouds a *mirrored object* will appear exactly at the same distance from the glass and with the same size as the real object. We call these virtual objects "*ghost walls*". They are problematic because it could happen that the whole room is mirrored to the other side of the specular surface. This artefact occurs when the laser scanner is moving in a specific angle toward the glass surface, naturally the same angle that objects could be seen in the glass.

3. Objects that represent the glass surface itself. If the laser beam is almost perpendicular or there is dust and other features on the glass, then part of the glass surface will be present in the point cloud.

Knowing above facts, it is possible to analyze the behavior of LiDAR systems in interaction with glass surfaces. Ghost walls could happen outside the building layout, where the façades are made of glass and the laser scanner is moving alongside a corridor. In this case, some of the indoor spaces are mirrored outside the building. Highly problematic ghost walls are those that occur inside the main structure. In such cases, detecting and removing them is challenging, but also important.

In our pipeline, ghost walls are detected and purged based on segments. Our method for semantic interpretation is a planar segmentation approach. Therefore, the point clouds are segmented with a surface growing algorithm (Vosselman et al., 2004). To detect ghost walls, the time stamps of the points are compared with the time of the closest trajectory point. Logically, because ghost walls are mirrored, they often have a time stamp, which differs from the time stamps of their neighboring points (which were not mirrored), as well as from the time stamp of the nearest trajectory point. Each point in the data is labeled as reflected point for which the time T point is more than Δt before or

after the time T traj of the nearest trajectory location. Δt is the time lag between the points in a ghost wall surface and the closest trajectory time. Δt is obtained empirically, and is obtained by checking such artefacts in the data. After labeling the points, the segments of which the majority of the points are labeled as reflected, are selected as ghost walls (Figure 3.3). In the next step, these ghost segments are projected back to their correct location. This is a relatively simple process, because they are in the same distance of the glass surface that the real object is located. But first, the glass surface should be detected. The glass surface is located between the real wall and the ghost wall. To detect the glass surface, a ray is reconstructed from a point on the ghost wall to the corresponding trajectory (see the purple line in the Figure 3.3c). This ray intersects a segment which almost has an equal distance to the real wall and ghost wall. The intersected segment is the glass surface. After detecting the glass surface, the points on the ghost wall are mirrored back relative to the glass surface to the other side (white points in the Figure 3.3d). Finally, after correcting the data from the ghost walls, it is ready to be applied for further processing.

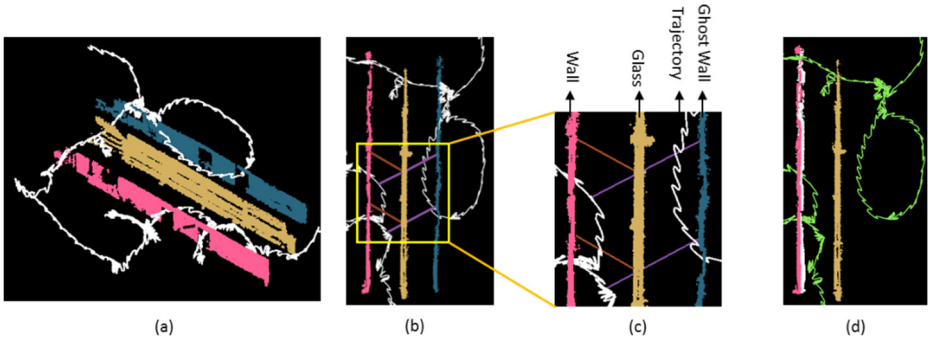


Figure 3.3. *(a) The perspective view and (b) the top view of the reflection situation. (c) The purple line is the incident line from the sensor to the glass and then to the reflected point on the other side of the glass surface. The brown line shows the secularly reflected line from the glass surface to the exact position of the object. (d) Shows the correct situation after the back projection of the ghost wall. The white points are corrected wall.*

3.3.3 Segmentation and Generalization

Since most indoor environments are composed of planar structures, extracting and labeling of planar faces is faster and more reliable than processing individual points. Because of the clutter and noise in the data the result of a segmentation cannot directly be used for semantic labeling and reconstruction. To generate planar patches that represent permanent structures, such as walls, floors and ceilings, a generalization method will be applied to the segments. For this purpose, we build on a method described by Kada, (2007) for generalization of 3D building models. Our adopted generalization method aims at merging segments based on their co-planarity, angle between normal vectors and their distance. First, all the segments are sorted by their size in

terms of the number of points. Starting with the largest segment three criteria are considered to merge a candidate segment into the current segment: (i) A generalization distance (= *D*) should be satisfied to accept or reject the candidate segment for merging, (ii) the parallelism of two segments by comparing their plane normal vectors, (iii) bounding boxes of two segments should be within a certain distance (= *d*). The proximity is checked alongside two segments planes. For example, two coplanar segments alongside a corridor should be within a threshold *d*. We refer to the result of generalization as "*surface patches (S)*" and for each surface patch a plane is fitted to its point cloud using a least squares method. The generalization method decreases the number of segments to be analyzed significantly. Additionally, small segments will not disturb the process of semantic interpretation. For detecting permanent structures, described in the next section, surface patches will be used instead of segments.

3.4 Permanent Structure Detection

For the detection of walls, floors and ceilings, the surface patches that are generated in the previous step are further processed. An adjacency graph is constructed from the patches and is further analyzed to induce the correct class of each patch (Section 3.4.2). For the detection of openings, an occlusion reasoning method is applied to discriminate between real openings and gaps that are caused by occlusion (Section 3.4.3). The occlusion test is also used to remove points that are outside the building layout and could be disturbing the reconstruction process. To start with detecting the permanent structure, the building levels are separated and then each level is processed separately (Section 3.4.1).

3.4.1 Separation of Building Levels and Stairs

The typical solution in the literature (Mura et al., 2016; Oesau et al., 2014; Turner et al., 2015) for separating building levels in indoor point clouds is using a height histogram of points. A level in a building is a horizontal section that extends over the floor space. Using the histogram is straightforward and gives an initial separation of the building levels. However, it is not applicable to buildings where a building level is extended vertically in the space to other levels (see Figure 3.4a) or a building with sub-levels. To overcome this problem in complex architectures, first the trajectory is separated to several levels and staircases. If the trajectory belongs to a handheld or a backpack system, the separation should be done where the operator enters the stairs. Therefore, the flat trajectory can be split from a sloped trajectory on the staircase. If the trajectory belongs to a push-cart scanner, then the trajectory of the levels is already separated, because the device does not move up or down the stairs.

To separate the levels, the process starts with the segmentation of the trajectory to the horizontal and sloped segments. A surface growing segmentation is used and points on the same horizontal or sloped plane are segmented together. Figure 3.4b shows that the trajectory points in the upper level (blue segment) belong to the same level and points on the staircases are segmented together. However, this segmentation needs a modification to make sure staircases are separated correctly. For example, if in the same level of the trajectory, there are several segments with a height difference of fewer than two meters (see Figure 3.4c, the orange and purple segments in the first floor) they will be merged. This is done because trajectories belonging to different levels typically have a height difference more than the ceiling height (at least two meters). After separating the trajectory to meaningful building levels, for each segment in the trajectory, the associated points from the point clouds will be selected using the timestamp.

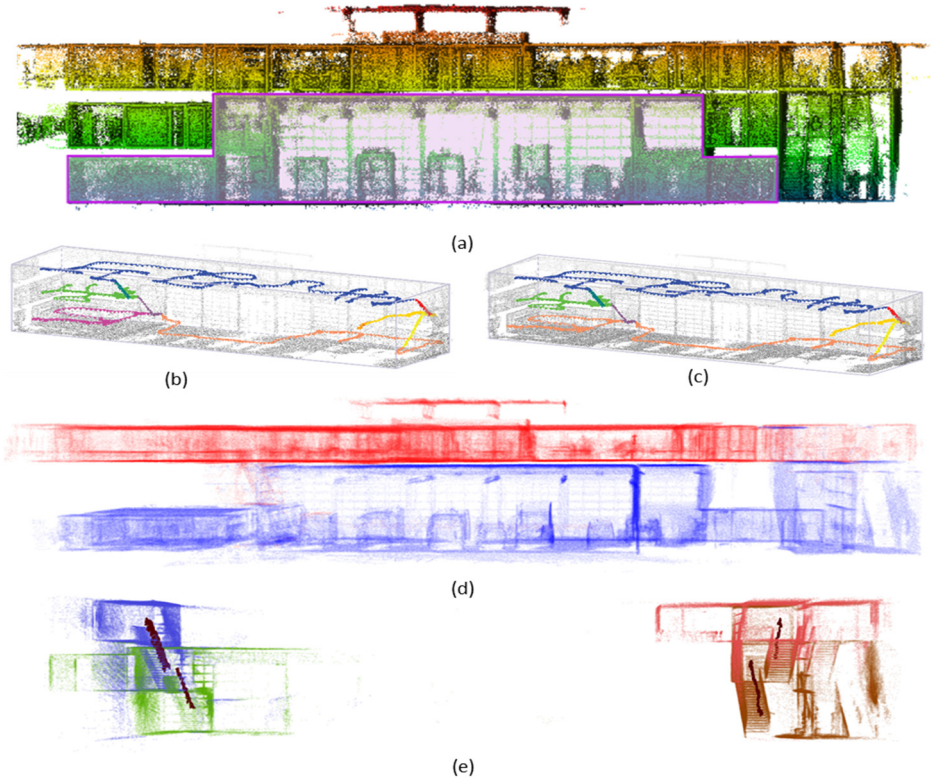


Figure 3.4. *(a) In complex buildings, part of one building level can be extended vertically to other levels. To separate levels, a height histogram approach is not working on this type of buildings. (b) Segmentation of the trajectory to horizontal and sloped segments. (c) After correction of segmented trajectory, for example, the purple and orange segments in the first floor are merged into one segment. (d) The separation of first (blue) and third levels (red) using the trajectory. The intermediate floor is removed for better visualization. (e) The stairs are extracted using the trajectory on stairs. Each color belongs to a segment of stair's trajectory.*

Near the staircases, the laser scanner measures points from other levels; to modify the level of these points to their correct level, the two dominant horizontal planes are detected as floor and ceiling of the current level and the label of the points is changed to the corresponding levels. Figure 3.4d shows the first and third level of the building. After separation of levels, each level will be processed individually for detection of walls, floors, and ceilings.

The point clouds of the stairs are extracted using the trajectory segments of stairs and the associated timestamp. Figure 3.4e shows four different stairs datasets colored based on four segments of the trajectory. Because a large portion of other levels may be seen from stairs, it is sometimes inevitable to have an overlap between point clouds of the stairs and the floors. For example, in Figure 3.4e part of the floors are also scanned from the stairs.

3.4.2 Wall Detection

The wall detection process includes detecting the permanent structures, such as walls, floors and ceilings. This process starts by making an adjacency graph (*G*) from surface patches (*S*). An adjacency graph is presented by *G = (V, E)* where nodes (*V*) are surface patches and edges (*E*) are connecting two adjacent nodes. Each node is associated with the point clouds of a surface patch *S*. When a label (*l*) is assigned to a surface patch, all the associated points obtain that label. The label shows the class of the surface, such as wall, floor, ceiling, door, and window.

Two nodes (*V*) are adjacent if their corresponding surface patches are within a specific distance from each other. This distance is set to d_{adj} = 0.1 meter in all of our experiments. Note that the coplanar or parallel segments are already merged. Therefore, two adjacent surface patches could meet under any arbitrary angle, which means our method is not limited to Manhattan-World. To deal with slanted walls and non-horizontal ceilings an angle threshold (α) should be specified to separate the candidate walls and ceilings before proceeding with the analysis of the graph. Each node in the graph is labeled as *almost-vertical* or *almost-horizontal* based on a threshold α. By default, this threshold is set to α = *45 degrees* to make a primary separation between candidate ceilings and walls. Considering this threshold, the node *V* in the graph *G* will be categorized to *Vh* and *Vv* for *almost- horizontal* and *almost-vertical*. By comparing a pair of surface patches out of nodes *V(v1, v2),* three principal labels will be assigned to each edge $e \in E$ of adjacent nodes *v1, v2*:

> *E* obtains the label *wall-wall iff v1* and *v2* are both almost-vertical *and* adjacent.
>
> *E* obtains the label *wall-ceiling iff v1* and *v2* are almost-vertical and almost-horizontal respectively *and* the center of *v2* is higher than the center of *v1*.

E obtains the label *wall-floor iff v1* and *v2* are almost-vertical and almost-horizontal respectively *and* the center of *v2* is lower than the center of *v1*.

After labeling the edges, each node in the graph will be analyzed based on the connected edges and the respective labels. Three main rules are applied to each node $v \epsilon V$ to decide for the label:

Rule 1. *V* obtains the label *wall iff* the count of *wall-ceiling* edges is equal or more than one *and V* is *almost-vertical*. This means every wall should be at least once connected to the ceiling.

Rule 2. *V* obtains the label *ceiling iff* the count of *wall-ceiling* edges is more than two *and* the count of *wall-wall* is equal to zero. This means an almost-horizontal surface with wall-ceiling edges should be connected more than two times to the walls to get the ceiling label.

Rule 3. *V* obtains the label *floor iff* the count of *wall-floor* edges is more than two *and* the count of *wall-wall* is equal to zero. This means an almost-horizontal surface with wall-floor edges should be connected more than two times to the walls to get the floor label.

Note that in Rule1, the connection of the wall candidates to the floor is not checked because of possibly heavy occlusions near the floor.

During the processing of the rules, further considerations as soft rules need to be applied. For example, during applying second and third rule on the ceilings and floors, each almost-horizontal surface cannot be a floor or a ceiling candidate. This happens especially in the case of horizontal surfaces of shelves and tables. Therefore, the average z-value of a horizontal patch is compared with an estimation of the floor and ceiling height to decide if it is near the floor or ceiling. In this way, horizontal surfaces of objects, such as tables and boxes, could be discarded. However, some of the horizontal surfaces that are near the floor and ceiling disturb the correct semantic labeling. For example, the top of shelves and cabinets that are near the ceiling could be labeled as the ceiling (see Figure 3.5b). As a drawback, the attached vertical surfaces that are connected to them may be also mislabeled as walls. To avoid this problem, the overlap of projection of almost-horizontal surfaces in the xy-plane is checked before starting with the rules. If the 2D projection of two horizontal surfaces has overlap (considering a small buffer), the upper surface is preserved as a ceiling candidate and then the process with the rules will follow. Since, the topological relations of the surfaces are exploited in our method, it is not limited to regular manmade structures or Manhattan-World.

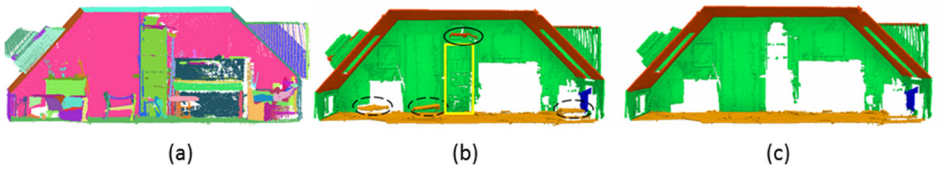


Figure 3.5. *(a) The segments of surfaces patches, (b) permanent structures, the wall in green, the ceiling in red and the floor is in orange color. The solid black circle shows the top part of the book shelf that is mislabeled as the ceiling. Hence, the bookshelf (yellow rectangle) is mislabeled as wall. Likewise, near the floor some horizontal segments are mislabeled (circles with dashed line). (c) After checking the intersection of vertical projection for each pair of surfaces and correction, the result is shown as the wall (green), the ceiling (red) and the floor (orange). The blue object is a clutter. Angle threshold is α= 50 degrees. Notice that the dormer and attached walls are labeled correctly in our method. The data is obtained from Reference* (Mura et al., 2016).

In the permanent structure detection method, a ceiling or floor will be distinguished from a wall by the angle threshold which is by default *α= 45* degrees. By applying rules 1, 2, and 3, a *slanted surface* could be labeled to a wall or ceiling (floor) depending on its normal angle. In our method, a slanted surface is distinguished by this angle threshold defined by the user. Figure 3.6 shows two different cases when *α* is set to 40 and 50 degrees. However, there is a special case where the slanted surface is distinguished as a wall and is supported by another vertical wall that is connected to the floor (see Figure 3.6b). Such a case happens when a slanted wall and a vertical wall are not segmented in the same surface patch since they have different normal angles during the generalization. Therefore, an extra check is required to see if the almost-vertical surface that is not connected to the ceiling is a wall or not. This check could be done by means of support and adjacency relation between a slanted surface and a vertical surface.

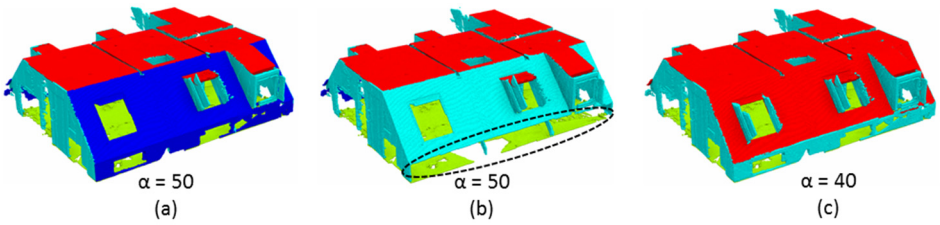


Figure 3.6. *(a) shows the permanent structure, ceiling (red), wall (cyan), blue (slanted walls) and green (floor). The angle threshold is 50 degrees. (b) Shows the permanent structure, with the same angle threshold (α =50), but the slanted walls algorithm is off. Consequently, supporting walls are not detected (dashed circle). Only walls (cyan color) that are connected to the ceiling are correctly detected. (c) The angle threshold is set to 40 degrees, and slanted walls are labeled as the ceiling.*

Let *v1* and *v2* represent the two almost-vertical surfaces and one of them is not connected to the ceiling, then the lower one (with a lower center) is called *supporter* (*v1*) and the upper one is called the *supported (v2)*. Furthermore,

the condition *max-z(v1)* < *min-z(v2)* including a buffer should be satisfied. Notice that checking the support relation is necessary, otherwise objects attached to the wall could be labeled as a slanted wall. Respecting this explanation, the corresponding edge (*E*) of two adjacent wall candidates (*v1, v2*) could obtain the following label:*E* obtains the label *wall-slantedwall iff v1* and *v2* are both almost-vertical *and* the intersection line is almost-horizontal *and* one surface is supporting the other one. The following rule is applied to define the label of a node *V*:

Rule 4. *V* obtains the label *slantedwall iff* the count of *wall-slantedwall* edges is more than zero *and* the count of *wall-wall* edges is more than zero *and V* is almost-vertical.

Since a real dataset with slanted walls from a MLS system was not available, our algorithm is tested on a part of the penthouse dataset from Mura et al., (2016). We assumed the slanted surfaces once as the non-horizontal ceiling (α = 40) and once as slanted walls (α = 50). Figure 3.6 demonstrates the results on a part of the penthouse building. This experiment shows the robustness of the algorithm in case of non-horizontal ceiling or slanted walls. In the next section, a method is presented for detecting the openings by using the trajectory and applying occlusion-test.

3.4.3 Opening Detection Using the MLS Trajectory

After detecting the walls, floor and ceilings, the point clouds are enriched with more semantics, such as openings (doors and windows). Reasonably, it is expected that doors and windows are located on the walls. Furthermore, openings are represented as holes or gaps in the data because where there is an open door or a window the laser rays go through the wall surface. The same gaps happen in the data, if part of the scene is not captured by the laser scanner, e.g., because of occlusion. Therefore, one problem of opening detection is to discriminate between data gaps and real openings in the data. We exploit the fact that a laser beam, crossing a wall surface with the opening, hits the objects behind the surface. Hence, from each location on the trajectory a ray is reconstructed to the measured laser point. Note that here the time attribute of the points plays an important role. Because from every point on the trajectory only the measured points at that specific time are evaluated for the ray casting. This process is named occlusion-test and is implemented as the following (see Figure 3.7): First, each surface patch Si with the wall label would be enveloped by a 3D voxel grid (grid size of 10 cm). Second, a ray is constructed from *t1* on the trajectory to the corresponding point *p1* in the point cloud. If the ray intersects a surface $s1 \in Si$, the intersection point of the ray and the surface corresponds to one of the voxels of the *s1*. The incident voxel obtains one of the four labels: Occupied, occluded, open or unknown. The

incident voxel is occupied if the measured point *p1* belongs to the *s1*, occluded if *p1* is in front of the *s1*, opened if *p1* is behind the *s1* and is unknown otherwise. If the ray does not intersect the surface the labels remain unchanged.

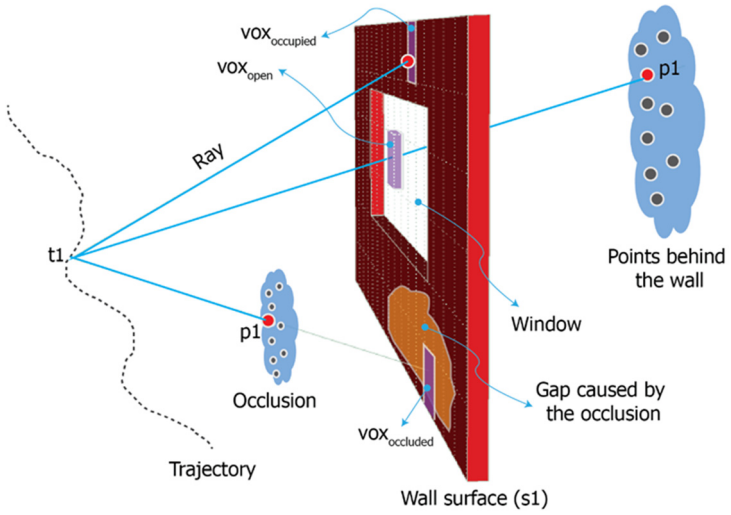


Figure 3.7. *An incident voxel on the wall surface will be assigned the label occupied, occluded or open if the measured point p1 is in the front, on the surface or behind the wall surface respectively.*

After the occlusion-test process, the results need to be further inspected to identify false openings. False openings happen where a clutter is connected to the ceiling and is extended to the neighboring walls. Therefore, during the occlusion test it is considered as a surface with opening (Figure 3.8b). Such false openings are identified and removed if more than a percentage (e.g., 80%) of voxels in the wall surface are labeled as openings (Figure 3.8c). With this simple check most of the false openings and erroneous walls are removed.

Furthermore, it is possible to separate the openings into openings that intersect the floor (doors), and those that are above the floor (windows). However, the clear frame of the opening could not be inferred because of the noise and occlusion.

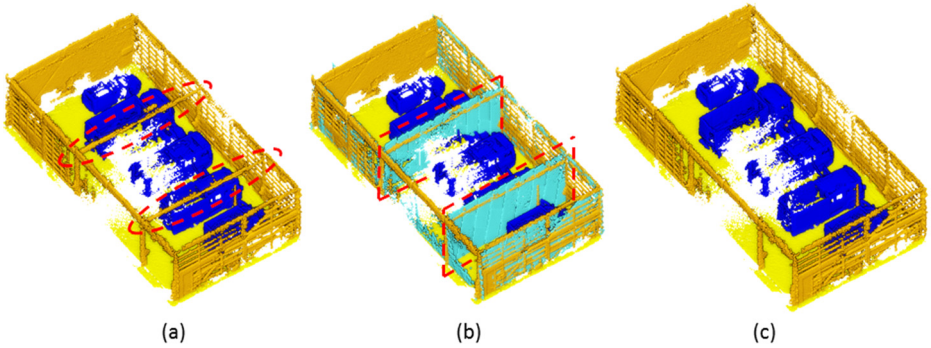


Figure 3.8. *(a) The classification of walls (orange), opening (light blue) and clutter (blue) in the fire truck hall of Fire Brigade building. The misclassified walls (red dotted area) cause the occlusion test algorithm to add the excess glass walls (light blue in (b)) in the middle of space that unnecessarily divides the space to several partitions. Figure (c) shows the correct classification of walls after identifying and removing false openings.*

The occlusion-test provides additional information about the points behind the wall surface. During the occlusion-test, points that are behind each surface are flagged for further inspection. Each point p_1 that is behind the surface s_1 and is measured from t_1 on the trajectory, can be a reflected point or a point that is sensed through a transparent surface. In Section 3.2, it was explained how to identify points that are caused by the reflection. Otherwise, the point is labeled as a *point-behind-surface artefact* and will be removed from the collection. Here, the assumption is that the objects behind an opening are scanned properly from the belonging space. A point behind a surface is less reliable because it is possibly measured through a glass surface. For example, in one of the datasets (Fire Brigade building, level 2) some of the rooms are partially mirrored to the outside of the building, because of a lot of glass surfaces in the façade. Consequently, in detecting the permanent structures they are mislabeled as walls, floors and ceilings. By removing points behind a surface, artefacts that are outside the building layout and could not be identified as reflection will be removed.

3.5 Space partitioning

Space partitioning is the process of separating space into more meaningful partitions that could be differentiated by permanent structures. Every space represents a room or a corridor. Unlike other methods that use a 2D projection of walls into xy-plane and applies cell decomposition, our method relies on volumetric space partitioning (Section 5.1). Therefore, slanted walls and non-horizontal ceilings do not constrain our method. For this purpose, a voxel space with the voxel-size of 0.10 m is exploited. In Section 5.2, the navigable and non-navigable spaces are extracted from the voxels.

3.5.1 Volumetric Space Partitioning

A voxel space is generated from the point clouds for space partitioning. Voxels are labeled with the permanent structure semantics. The occupied, opening and occlusion labels (Section 4.3) are transferred to the voxels as occupied label. Rest of the voxels are labeled as empty (unoccupied). Including the label of openings and gaps is important for space partitioning, because spaces can be connected through openings (e.g., a window) or gaps (e.g., an occlusion). Therefore, the dataset that is used to label voxels contains openings, occluded areas, walls, floors and ceilings.

After labeling the voxel space to occupied and empty, three main steps generate the spaces: (i) A morphological erosion method is applied on the empty voxels. Therefore, the area covered by occupied voxels will grow and empty voxels with weak connections will be separated. (ii) A connected component analysis is applied on selected empty voxels from the previous step to make separate clusters of empty connected voxels. Each cluster at this stage represents a space partition. (iii) Then a morphological dilation is applied on empty voxels, while this time empty voxels have a cluster number. Consequently, the area covered by empty voxels grow while occupied voxels area is shrinking. Finally, each cluster of empty voxels represents a space partition. This approach has two advantages, it is volumetric and it is independent of Manhattan-World constraints. However, the empty voxels that are present outside the building layout will generate some invalid spaces that need further attention. In the following, we explain how to modify these invalid spaces.

Validating Space Partitions Using the Trajectory: In case the building layout is known, for example from a ground plan, it is possible to detect and remove invalid spaces generated outside the building structure. However, our pipeline is just relying on the geometry of the point clouds. Therefore, by using the trajectory, spaces that are not traversed during the data collection will be discarded. In other words, space partitions (e.g., rooms, corridors) are representing empty spaces in the environment that have intersection with the trajectory. A kd-tree search algorithm is used to check a partition's intersection with the trajectory. Furthermore, the space partitioning process is retained as a volumetric solution and projecting spaces to xy-plane is avoided (because of possible slanted walls). For each partition, the nearby trajectory is found and if the distance is less than the voxel size

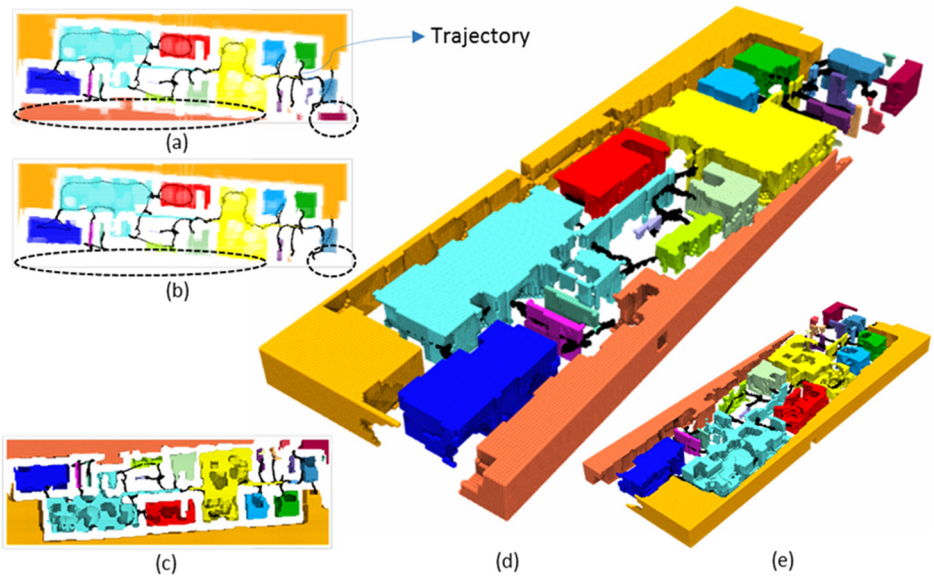


Figure 3.9. *(a, b) show the top view of the partitions in various colors and the trajectory in black. The white places between the spaces are occupied places (e.g., furniture and walls). The dotted circles show the invalid partitions that are removed, because there is no intersection with the trajectory. The orange large partition is also an invalid space but is not removed, because it has connection with the interior space and with the trajectory. (d) The perspective view of the spaces and the trajectory. (c, e) Show the bottom view of the spaces. The carvings of furniture and occupied places are visible inside the partitions.*

it indicates the intersection, hence, a valid partition. This can be done in 3D and it enables us to discard outside partitions that are not navigated by the trajectory. This approach is favored over methods of calculating the alpha shape of a partition in 3D or the minimum enveloping polygon in 2D to check the intersection with the trajectory, because an alpha shape or a minimum enveloping polygon cannot precisely represent the complex shape of a space partition. Figure 3.9 shows the spaces and the trajectory from different views.

During the space partitioning process, each space partition represents only the empty space if the furniture is included in the process. This fact is exploited to generate the 3D navigable space. However, including furniture can cause over-segmentation of the space because some of the furniture can divide the space in the same room. Next section elaborates on the details of navigable space.

3.5.2 Extracting the Navigable Space

Having discussed how to generate space partitions, in the following, it is explained how to extract the navigable and walkable area out of the empty space. Each space partition represents the empty space (after including the

furniture space). The navigable space can be generated in different heights above the floor and below the ceiling, which is suitable for flying objects to navigate in the space. Again, it is important that the gaps on the walls caused by the occlusion or opening are labeled as occupied in the voxels to avoid misinterpretation as walkable space. Doorways that are considered as openings are labeled as navigable in the final navigable space. Empty voxels just above the floor are extracted as walkable space. In Figure 3.9c and 3.9e, the spaces are illustrated from the bottom to show the carvings of the occupied spaces in the empty spaces. If some of the openings connect the spaces and they are not recognized during the opening detection, as a drawback few partitions cannot be split and remain as one space (e.g., the orange space in Figure 3.9). The void between the space partitions is caused by furniture and permanent structure.

3.6 Door Detection Using the Trajectory

In this stage, doors that are intersected by the trajectory during the data acquisition can be detected. Note that in Section 3.4.3, some doors were already detected as openings by occlusion tests, while here it is possible to detect closed doors as well. For detecting the doors using the trajectory, the voxels and the trajectory are the input data of the process. Voxels are used for this step, because the algorithm tries to find the center of each door that is crossed by the trajectory (see Figure 3.10).

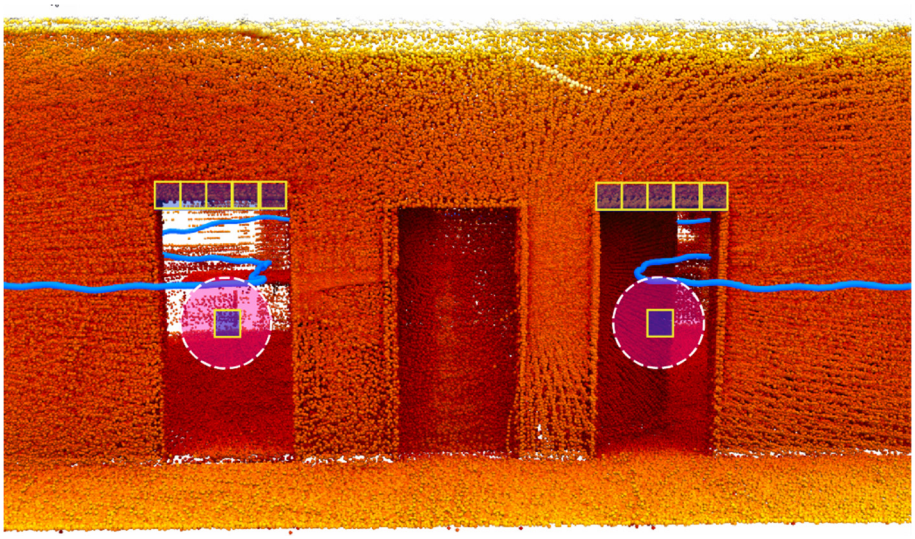


Figure 3.10. *A Zeb-Revo trajectory (blue) crosses an open door in the left and a semi-open door in the right. The middle door, that is closed, is not traversed by the trajectory thus cannot be detected by our algorithm. The yellow boxes show the door center candidates and top of the door voxels. The circles show the search radius from the door center candidate to the trajectory.*

The door center is represented by an empty voxel in case of an open door and an occupied voxel in case of a closed door. Each voxel in the voxel space is checked whether it can be a center of a door candidate (a *door center*). A voxel is a door center candidate if: (i) Nearby the voxel there is a trajectory. (ii) Above the voxel occupied voxels exist that represent top part of a door frame. (iii) The neighborhood of the voxel should be empty for an open door. These three criteria enforce three main search radius parameters: (1) A search range to look for a nearby trajectory ($r_{traj} < \sqrt{3} * voxel\text{-}size$); (2) a search radius to look for voxels on top of the door frame relative to the floor ($1.80\ m < r_{top} < 2.10\ m$); and (3) a neighborhood search radius ($r_{void} < n * voxel\text{-}size$) to make sure around the candidate voxel is empty, where the search radius is a factor of voxel size. The r_{void} threshold should always be smaller than the door width to exclude the door frame in the search for empty neighborhood. Empirically, if the percentage of empty voxels around a door center within the search radius (r_{void}) exceeds 70% of the total neighbor voxels, then the third criteria for an open door is fulfilled. Furthermore, to speed up the calculation process, only voxels are explored to be a door center that are located in the height between 0.8 to 1.10 m relative to the floor, as the door center is expected to be in this height.

Closed Doors: Closed doors appear in the point cloud as part of the wall (Figure 3.10, the middle door). When the trajectory crosses the door and the door is closed before or after the scanning, it appears in the data as if the trajectory went through the wall. To detect closed doors crossed by the trajectory, the same three criteria as open doors are applied, but with the difference that for the third rule the neighborhood of the voxel candidate as the door center should be occupied instead of empty. Notice that simply intersection of wall planes with the trajectory is not sufficient to detect closed doors. Because in cluttered rooms the trajectory goes between the congested furniture or false detected walls that can be identified as false doors. Therefore, checking the three criteria is also required for closed doors. The door detection algorithm, using the trajectory, can only be used for spotting the location of the door (also double doors), For identifying the door frame or the door leaf, further inspection is required.

3.7 Results and Evaluation

Our approach is tested on four datasets collected with four different mobile laser scanners. The details of the datasets and the scanners are given in Table 3.1. The results of all datasets and the ground truth are shown in Figure 3.11. The results show that 80% of the doors and more than 85% of the rooms are correctly detected. Our methods are tested on buildings that violate the 2.5D and Manhattan World assumptions. The space partitioning results (Figure 3.11, 3rd column) show our constraint-free approach in arbitrary room layouts with

different ceiling heights. Both datasets from Fire Brigade building level 1 and the TU Delft Architecture building have large halls with a high ceiling and different ceiling heights in other rooms. Figure 3.11 (3rd column, first and last row) illustrates the extracted spaces for these two datasets. The permanent structures in Figure 3.11 (2nd column) indicates that our pipeline is capable of detecting most of the walls and openings in the heavy cluttered environments with many reflective surfaces.

Each dataset is subsampled to ease the visualization and to decrease the processing time. For subsampling, every k'th point of a kNN is used, where a reduction factor between 4 and 6 is applied to decrease the size of the original dataset while keeping the structure of the point clouds. The subsampling keeps the average point distance less than 0.05 m.

The other important influential factors are noise, the level of clutter and the number of glass surfaces in the data. The level of noise depends on the sensor precision and the SLAM algorithm. For details of each MLS device accuracy and noise, the readers are referred to the specification of each product and the review by Lehtola et al. (2017). In terms of high clutter and high number of glass surfaces, Fire Brigade dataset poses a lot of challenge because of the very large glass walls. Such glass walls, as well as heavy clutter are present on the first floor (Figure 3.12), where the fire trucks are located.

Table 3.1. *Details of the datasets and capturing device. The number of correctly detected rooms and doors is mentioned in the fourth and fifth columns.*

Dataset	# Points	MLS Device	#Rooms/ #detected	#Doors/ #detected	Clutter and Glass
Fire Brigade level 1	2.9 M	Zeb1	9/8	8/7	High
Fire Brigade level 2	3.6 M	Zeb1	16/14	17/12	High
Cadaster Building	4.1 M	NavVis Trolley	10/9	7/5	High
TUBraunschweig	1.7 M	ITC backpack	30/27	30/29	Low
TUDelft Architecture	3.2 M	ZebRevo	18/13	25/18	High

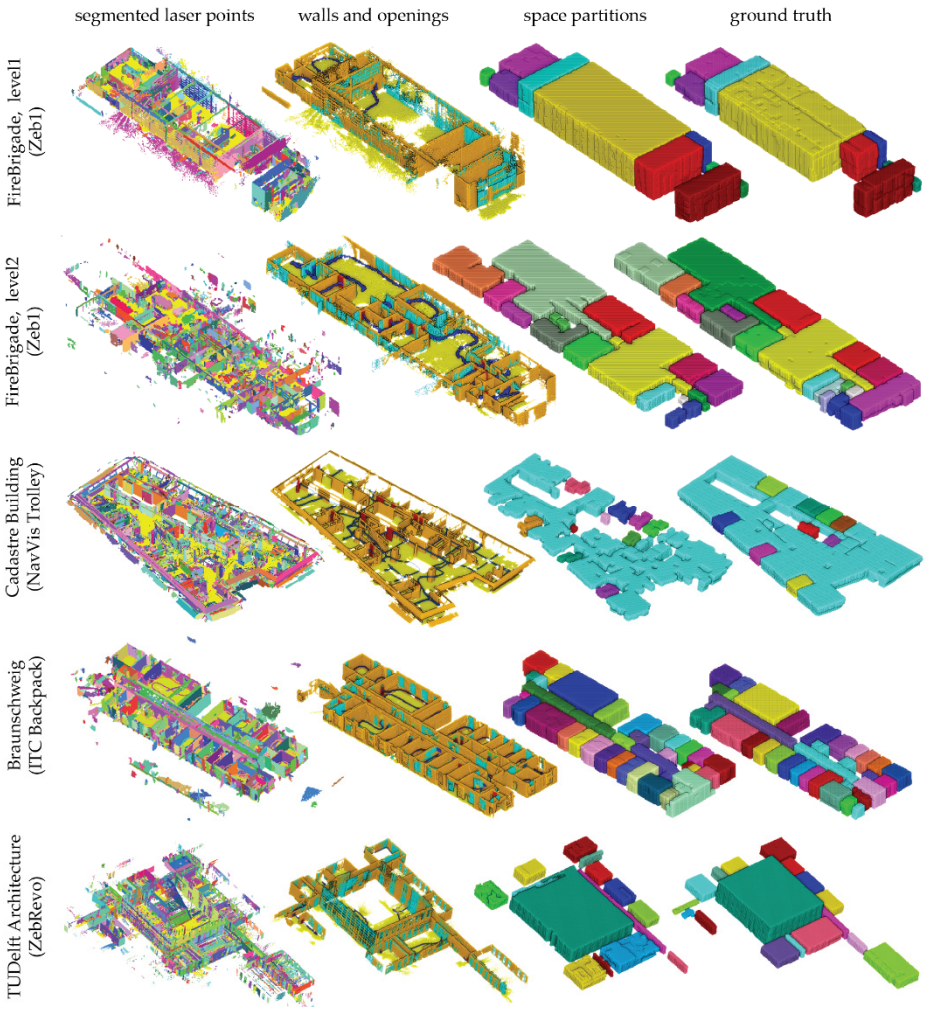


Figure 3.11. *Results of datasets of Table 3.1. From top to bottom: Fire Brigade building level 2, level 1, TU Braunschweig, Cadaster building and TU Delft Architecture building. In the second column, detected walls (orange), floor (yellow), doors (red) and openings (blue) are shown.*

Figure 3.12. *First level of Fire Brigade building. The amount of clutter and very large glass walls makes the process of wall detection challenging. The ceiling has two different heights and there is a lot of clutter below the ceiling. The colors represent the segments.*

Implementation: All algorithms are written in C++ and tested on a Lenovo ThinkPad workstation with an Intel core i7 (2.5 GHz), 16 GB RAM. The main computational cost is devoted to occlusion test process, because of the ray casting algorithm where the peak of RAM usage is 16 GB and for large datasets it takes up to an hour. Another expensive process is space partitioning, because of the 3D morphological process on a large number of voxels. For an area with almost 15 million voxels, it takes 10 minutes with a voxel size of 10 cm, and 3 minutes with a voxel size of 20 cm. Other algorithms including permanent structure detection, door detection, reflection removal, level partitioning and surface growing take seconds up to 5 minutes depending on the size of the dataset. The computation time for space partitioning is more dependent on the volume of the building and height of the ceiling than the size of the point clouds. For example, for the TU Delft dataset the number of voxels exceeds 100 million, since the orange hall has high ceilings (almost 13 meters). Therefore, the space partitioning method is implemented with 20 cm voxel size for this dataset to speed up the process.

Parameter Selection: Our pipeline starts with the surface growing segmentation followed by a surface patch generalization algorithm. For the surface growing segmentation, the most important parameter is the smoothness threshold. The optimal value depends on the amount of sensor noise and the noise caused by the SLAM algorithm. The sensor noise is less than 5 cm for the MLS devices in this work. However, there is more noise in the data created by SLAM algorithm and artefacts of the glass reflections. Therefore, we experienced a value between 10 to 15 cm for datasets from Zeb1 as a good threshold for planar segmentation and between 5 to 10 cm for other datasets (ZebRevo, ITC Backpack and NavVis Trolley). For the surface patch generalization, nearby surfaces are considered parallel if their normal vector angle tolerance is less than $\theta < 10°$, and their proximity (d) alongside the plane is less than 60 cm. The time lag Δt is the important parameter for

detecting and pruning of ghost walls. Empirically, we choose 150 seconds time lag for a point to be considered as a reflected point, and if more than 70% of the points in a segment have this time difference with their neighbor trajectory that segments are defined as a ghost wall.

For reconstructing the adjacency graph, the distance for adjacency of two surface patches is less than d_{adj} < 10 cm and the minimum length of an intersection line is 20 cm. We experienced that a minimum length of 20 cm in most datasets is reasonable. There is just one special case that the threshold is increased to 25 cm, when the frames of doors are extended to the ceiling (e.g., glass rooms in Figure 3.13). To avoid door leaves to be misclassified as wall, a minimum length of 25 cm for intersection line is considered.

The default threshold of 45° is considered for separating the surfaces to almost-vertical and almost-horizontal. In case of different sloped ceilings, the angle threshold could be changed to recognize the ceilings from slanted walls. The minimum number of supporting points for each surface to be included in the adjacency graph is 500 points. A voxel size of 10 cm is preferred for algorithms operating on voxels, such as the occlusion test, space partitioning and door detection. For a point spacing of 2 to 5 cm, the voxel size of 10 cm offers a good balance between the computational time and the number of preserved details of permanent structures. The window size of the morphological operator for the space partitioning should be larger than a doorway to ensure the separation of spaces at the locations of open doors. Therefore, a window size between 1.0 to 1.3 m is suggested. Other soft parameters, such as kNN search, proximity search and connected components do not have significant influence on the whole pipeline.

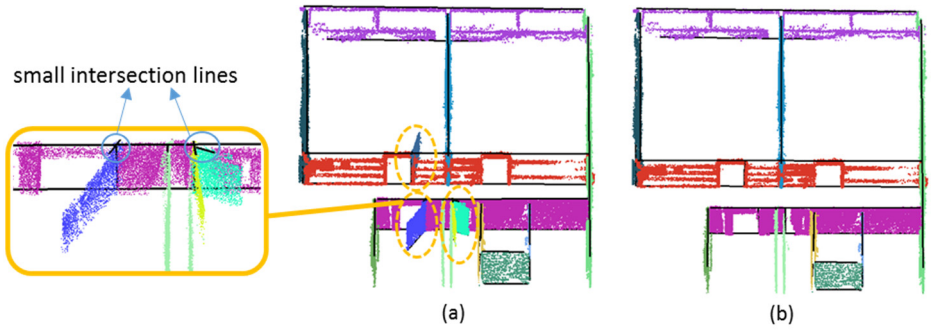


Figure 3.13. *Result of wall detection, using the adjacency of segments. The effect of minimum length parameter for intersection lines between adjacent segments (door leaves and the ceiling) on the result of wall detection is shown. In (b) the minimum length is 25 cm, so small intersections are discarded and consequently door leaves are not misclassified as wall.*

Robustness: The robustness of our algorithms is evaluated by testing on various datasets collected by four different mobile laser scanning devices. Additionally, our pipeline is tested on a multistory building (Fire Brigade dataset), a building with slanted walls (Figure 3.6) and different ceiling heights (Fire Brigade building, level 1 and the TU Delft Architecture building), and a building with high amount of clutter and glass surfaces (Cadaster building and Fire Brigade building, level 2). Among those, buildings with large glass walls pose the largest challenge to our pipeline (for wall detection and space partitioning), because the connection of glass surfaces near the ceiling is not guaranteed in the segmented data and in some cases these glass surfaces are missing entirely in the data (the TU Delft Architecture building the hall with orange stairs, Figure 3.14). However, the wall detection is capable of detecting glass walls even with loose connections to the ceiling.

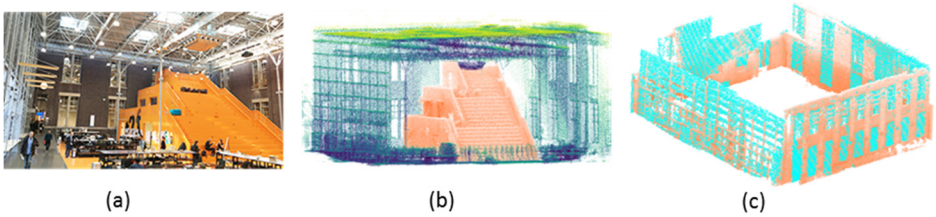


Figure 3.14. *The robustness of our algorithms for the buildings with many glass surfaces. (a) The orange hall in the TU Delft Architecture Building, (b) the point clouds and (c) the classified walls and glass surfaces.*

All datasets are processed with d_{adj} 10 cm for reconstructing the adjacency graph. We experienced, in most cases that increasing this threshold to 20 cm or higher results in losing some of the walls; and decreasing the threshold to less than 10 cm results in misclassification of some clutter surfaces to wall surface. The d_{adj} parameter depends on the noise in the data. For datasets with a higher level of noise the threshold could be increased to 20 cm.

Limitations: The permanent structure detection using the adjacency graph is susceptible to errors when there is a clutter at the ceiling close to the walls (Figure 3.15). This kind of clutter could be misclassified as wall if the size of the clutter is large. Hence, during the occlusion test it may be misclassified as a glass surface. Consequently, the space would be partitioned incorrectly. The reason behind this limitation is that the rules in the adjacency graph deliberately do not check if a wall candidate is connected to the floor, because in most cases walls are occluded near the floor. Hence, a structure in the ceiling connected to the neighboring walls could cause this error.

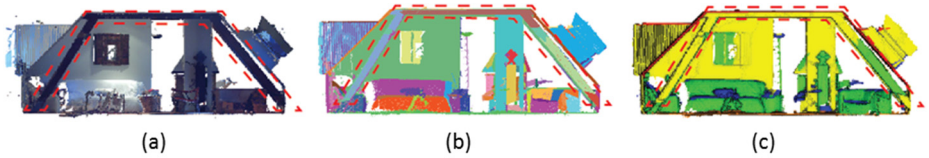


Figure 3.15. *(a) Color point clouds and (b) segmented point clouds. Our adjacency graph algorithm is limited when a clutter in the ceiling is connected to the walls and constructs a vertical surface attached to the ceiling and neighboring walls (red area in the images). In (c) walls are yellow, and the red area is misclassified as wall. The dataset belongs to Reference* (Mura et al., 2016)*.*

During the door detection algorithm, the algorithm fails in case of low ceilings spaces, such as basements or tall doors reaching until the ceiling. This is because the algorithm searches for the points on top of the door center, and when the ceiling is low it could be considered as the top of the door that results in detecting false positive doors (see Figure 3.16). Detection of doors may be difficult if they are semi-open, because the condition that checks if a door center is in a void neighborhood for an open door cannot be true if a door-leaf is occupying part of the doorway. Double doors, could be spotted with our algorithm, but the exact door frame could not be extracted.

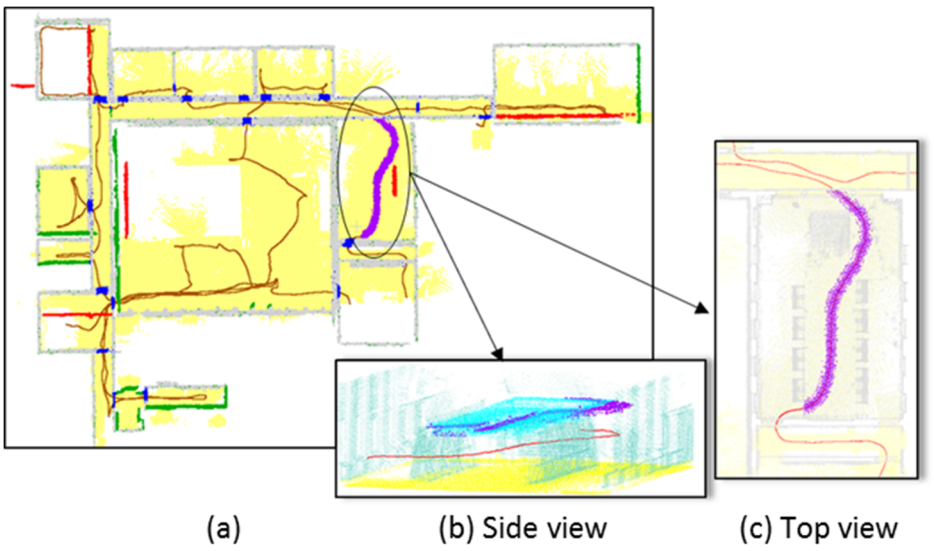


Figure 3.16. *Door detection method in an area with a low ceiling. (a) Shows the detected walls (grey), false walls (red), missed walls (green) and detected doors (blue). Most of the doors crossed by the trajectory are detected. (b) The side view shows the trajectory and low ceiling (light blue). The purple dots are points above the trajectory that are wrongly detected as a door. (c) is the top view of b.*

Using the trajectory to separate the levels can be error-prone in buildings with a lot of glass surfaces, because objects could be seen from other levels, especially in the stair cases. However, using the trajectory for the Fire Brigade

building with a huge space in the first level that spans to the other level is the reasonable option.

In the cadaster building dataset (Figure 3.11, 3rd row), the surface growing segmentation results in flawed segments because of slanted glass surfaces and artefacts outside the façade (Figure 3.17). Consequently, the supporting walls that are connected to the slanted glasses could not be extracted. The opening detection for cadaster dataset is not performed, since the time stamp of the point clouds were not available. All the point clouds including the furniture are used for the space partitioning of the cadaster dataset. Otherwise the interior space will be connected to the outside through the missing walls.

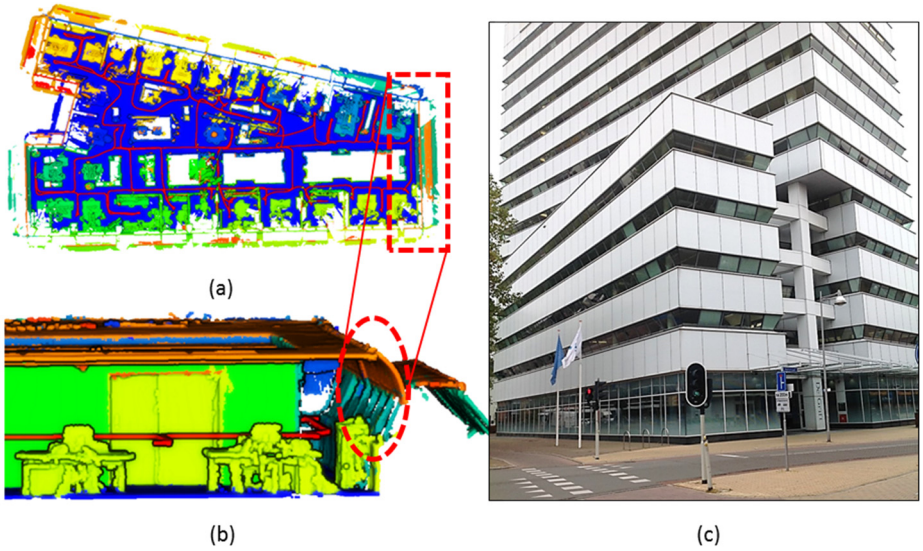


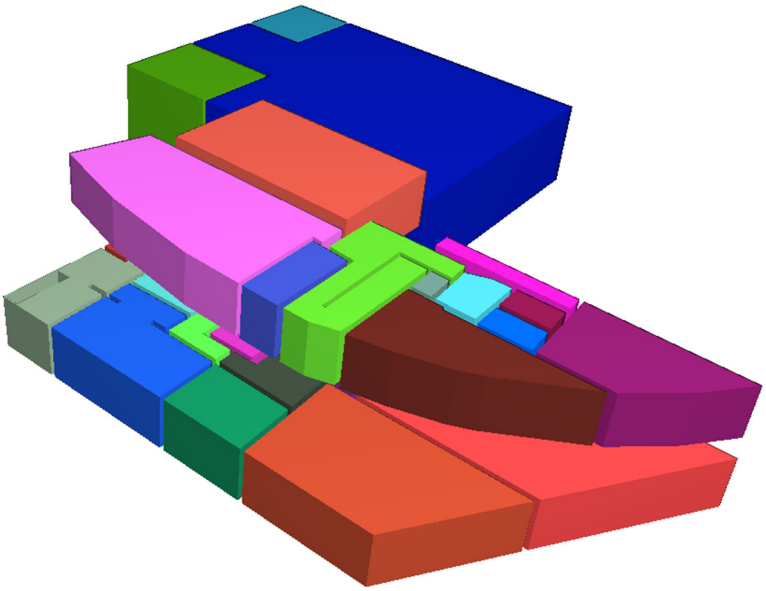
Figure 3.17. *The cadaster building. (a) The top and (b) side view of the point cloud of one of the floors. The glass façade has slanted surface and artefacts that pose a problem for detecting them by surface growing. The supporting walls connected to the floor are not detected by our algorithm. (c) The front view of the cadaster building. Slanted glass surfaces are visible in the façade.*

3.8 Conclusions and Future Work

Several algorithms are presented for the interpretation of complex indoor scenes captured by a mobile laser scanner. Our work proposes a complete pipeline for classification of MLS indoor point clouds captured by four different systems. The methods show robustness in dealing with cluttered scenes and glass surfaces. Arbitrary wall layouts, slanted walls, and non-horizontal ceilings can be correctly identified in most cases. We presented how to deal with artefacts caused by reflective surfaces. The usefulness of the scanner trajectory is proved in several algorithms, such as detecting closed and open

doors, removing invalid spaces outside the building layout, separating complex building levels and detecting ghost walls. Although our approach is not limited to Manhattan-World and 2.5D assumptions. Still there is a need for improvements to reconstruct water-tight models.

Chapter 4 - Indoor 3D reconstruction from point clouds for optimal routing in complex buildings to support disaster management *



* This chapter is based on:

1. Nikoohemat, S., Diakité, A., Zlatanova, S., and Vosselman, G.: Indoor 3D Modeling and Flexible Space Subdivision from Point Clouds, ISPRS Ann. Photogrammetry Remote Sensing Spatial Inf. Sci., IV-2/W5, 285-292, https://doi.org/10.5194/isprs-annals-IV-2-W5-285-2019, 2019.

2. Nikoohemat, S., Diakité, A.A., Zlatanova, S., Vosselman, G., 2020. Indoor 3D reconstruction from point clouds for optimal routing in complex buildings to support disaster management. Automation in Construction 113, 103109. doi.org/10.1016/j.autcon.2020.103109

Notes: Section 4.5 on Flexible Space Subdivision is written and implemented by Diakité, A.

Abstract

During an emergency inside large buildings such as hospitals and shopping malls, the availability of up-to-date information is critical. One common source of information is the 2D layout of buildings and emergency exits. For most buildings, this information is represented as tangled floor plans, which in most cases are outdated. One solution to update the data of buildings after each reconstruction is to recreate 3D models of buildings in a quick and automatic approach and to make these models available for first responders to be used in emergency cases. Thanks to advances in remote sensing, laser scanners can be used to generate an accurate spatial representation of buildings quickly. However, such devices provide point clouds, which are unstructured data. In this chapter, we introduce a complete workflow that allows to generate 3D models from point clouds of buildings and extract fine-grained indoor navigation networks from those models, to support advanced path planning for disaster management and navigation of different types of agents. The process extracts structural elements of buildings such as walls, slabs, ceiling and openings, and reconstruct their volumetric shapes. Additionally, the furnishing elements in the input point clouds are identified and reconstructed as the obstacles. Stairs are also reconstructed to allow multistory navigation path planning. Our algorithm is fully 3D and can handle vertical and slanted structures. We test it on several real datasets, compared it to state-of-the-art approaches and provide a process to check the consistency of the reconstruction, which allows in return to further improve its result.

4.1 Introduction

Monitoring the changes of buildings after each renovation and auditing the compliance of the changes according to the safety standards is a known problem. For most new buildings, Building Information Models (BIM) are available at the start of the construction, but these models are no longer current after building renovation. Keeping the data of the building up-to-date and checking it against the safety regulations is problematic, although it is critical to support rapid intervention in situation of emergency. Indeed, information about building layouts and indoor objects occupancy is a game-changing support to efficient and safer emergency response in disaster management. Mobile Laser Scanning (MLS) systems provide such a possibility when their data is post-processed using a smart and fast workflow.

During the last years, significant improvements have been made in Indoor Mobile Laser Scanning systems (IMLS) (Lehtola et al., 2017). Using a mobile mapping system, it is possible to scan a large multi-story building up to 100 rooms in one day. However, processing the large data produced by an MLS system to reconstruct a 3D model is not a trivial task and needs sophisticated software and expert knowledge. In this chapter, we present an automatic approach which enables us to produce a coarse 3D model of buildings in a short time. Such models can be further improved and enriched by user interactions to keep the data of large buildings up to date. Safety authorities can use these models to see if the renovated buildings comply with the safety regulations. Moreover, the produced models can be used to support advanced processes such as fine-grained indoor path planning to facilitate efficient emergency response.

The problem of creating a 3D model from cluttered point clouds attracted many researchers in different domains such as robotics, architecture, engineering and construction. The fact that buildings have varying and complex structures makes the problem challenging. Training computers to learn all types of building structures looks like an unfeasible task. During the last years, researchers developed algorithms that work for less complex buildings or buildings with a regular layout or that only create a 2.5D model (Becker et al., 2015; Ikehata et al., 2015; Khoshelham and Diaz-Vilarino, 2014; Mura et al., 2014). Most of the 3D reconstructed models in the literature reflect simple room segmentation (space subdivision) and a clutter-free environment (Becker et al., 2015; Khoshelham and Diaz-Vilarino, 2014) or they rely on having the scanner positions per room (Mura et al., 2016; Ochmann et al., 2016). In contrast, when a mobile laser scanner is used for data acquisition, the perception of the space is continuous and there is no separate scanning per room. Moreover, the presence of furniture causes occlusion problems, which makes the process of model reconstruction and room separation more

challenging. Additionally, slanted walls, ramps and non-horizontal ceiling are other challenges that in most reconstructed indoor models are not addressed.

In our approach, we first detect the permanent structures such as walls, floors and ceilings adapting the method presented in (Nikoohemat et al., 2018a). With further automatic processing such as undershoot correction, the connection of walls to each other and the ceiling and floor is guaranteed. Volumetric walls are reconstructed by detecting the parallel faces of a wall and merging them into one wall object which is represented by a parametric minimum rectangle. The room segmentation method is based on the correct reconstruction of walls and enclosure of the space in an early stage. We proceed to regularized Boolean operations (Tilove and Requicha, 1980) on the reconstructed permanent structures to reconstruct the room volumes.

Openings are crucial items during a disaster as well as for safety regulations. Doors which are crossed by the trajectory of the mobile laser scanner are detected and added to the model. The combination of all the reconstructed features in addition to their semantic information allows us to apply the Flexible Space Subdivision (FSS) (Diakité and Zlatanova, 2018), which is a 3D navigation framework that subdivides the indoor space into occupied, functional and navigable spaces. However, in this work, we do not try to enrich the spaces with more semantics, such as room functions, automatically. Furthermore, the automatic detection of furniture type is out of the scope of our work. Some of the large pieces of furniture are selected using a connected component algorithm and an oriented bounding box is generated to present a clearance around the furniture as obstacles. To extract free space in 3D, some floating objects such as lamps are also included. Staircases are another essential element in our 3D models to test the room's connectivity on separate floors. Detecting and modeling stairs or staircases is a complex task because stairs can have versatile structure (e.g., with walls, metal bars or glasses). In our workflow, we explain how first to identify stair ramps and then detect individual steps. Finally, with several constraints, the consistency of the 3D model is evaluated. The navigation graph is generated between a pair of spaces to control the connectivity of all spaces, including doors and stairs. The final navigation graph demonstrates a connectivity network and does not show a turn by turn graph as generating a detailed graph is not the focus of our work.

Our pipeline is a combination of algorithms from generating a 3D model from point clouds of multistory buildings to presenting a flexible space subdivision for dynamic navigation. This work goes beyond the simple assumption of a Manhattan-World, vertical walls and a clutter-free environment. Modeling stairs, ramps and slanted ceilings are all integrated into our reconstruction workflow. In contrast to most of the indoor navigation work, which is based on synthetic models, our methods are tested on real complex use cases.

Moreover, a consistency control method is applied to check the model correctness against several constraints. The results show that scanning a building with a mobile laser scanner and using our pipeline enables the rapid generation of a coarse 3D model, including spaces for first responders. The contribution of our work is as follows:

A pipeline for 3D reconstruction from point clouds of complex multistory buildings including stairs, ramps and furniture is presented. Reconstruction of non-horizontal floors, ceilings and slanted walls is explained and tested. The pipeline reconstructs both volumetric walls and rooms polyhedra and provides a semi-automatic method to improve the reconstruction process. Furniture are included in the subdivision pipeline to demonstrate realistic navigation scenarios in multistory buildings. Several heuristic rules are implemented to check the consistency of generated models.

The remainder of this chapter is organized as follows: Section 4.2 describes related work, section 4.3 gives an overview of the pipeline. Section 4.4 explains the methodology for 3D reconstruction from point clouds, and section 4.5 discusses the flexible space subdivision. In Section 4.6, a consistency control of the 3D model is discussed. In section 4.7, our methods are tested on real data and the results are discussed. Section 4.8 is the future work and the conclusion.

4.2 Related Work

The topic of indoor 3D reconstruction from point clouds, RGB-Depth and images have been studied from different aspects and applications. For example, in the robotic domain, the navigation and interaction of robots with the indoor environment is the goal (Pinheiro et al., 2015). Problems such as scene understanding, object detection and localization of robots are investigated by the researchers (Bormann et al., 2015; Bowman et al., 2017; Koppula et al., 2011; Pinheiro et al., 2015; Rusu, 2013b; Rusu et al., 2009). Using images for indoor scene interpretation, façade reconstruction and modeling floor plans are investigated by many researchers (Dehbi et al., 2017, 2016; Liu et al., 2018; Wang et al., 2013). Another upcoming research is change detection in the permanent structures during different epochs of the renovation for example for indoor 3D cadaster applications (Koeva et al., 2019; Nikoohemat et al., 2018b). Besides, the significant progress in Wearable Mobile Laser Scanners (WMLS) and Indoor Mobile Mapping Systems (IMMS) provides a fast-growing source of data for researches in the domain of indoor modeling (Chen et al., 2010; Karam et al., 2019; Wen et al., 2016).

Reconstruction of walls is an important step to create a correct 3D model. A wall representation in the model depends on the application of the model and

3D modeling standards such as IFC (ISO 16739, 2018) and IndoorGML (Lee et al., 2014) and those which are not following a specific standard (Ikehata et al., 2015). Volumetric representation and cellular representation of walls are close to IFC and IndoorGML standards, respectively; and suitable for scan-to-BIM applications. Studies which deal with building retrofitting (Ochmann et al., 2019, 2016), indoor navigation (Diakité and Zlatanova, 2016; Jung and Lee, 2015) and automation in construction (Bassier et al., 2018; Macher et al., 2017; Thomson and Boehm, 2015) are cases based on IFC standards. In contrast, in some of the literature, a piecewise-planar representation of the model generates walls as planar objects (Boulch et al., 2014; Mura et al., 2016; Oesau et al., 2014) and rooms as the polyhedra. Accordingly, the method for room segmentation (also known as space subdivision or space partitioning) would be different. If the application of space is more important than surrounding elements, then a cell decomposition approach or voxel-based method should be suitable for applications such as indoor navigation (Ambruş et al., 2017; Diakité and Zlatanova, 2018) and perception of semantic space for robots (Bormann et al., 2015). However, if the enclosing elements such as the correct geometry of walls, doors and windows is the goal of a 3D model to be used in BIM software, then the process of reconstruction should pay more attention in generating correct walls and details such as windows (Bassier et al., 2018; Becker et al., 2015; Ochmann et al., 2016).

In most of the literature, there are several main assumptions when detecting and reconstructing the permanent structures. One general assumption is based on vertical walls (Macher et al., 2017; Murali et al., 2017; Ochmann et al., 2019; Oesau et al., 2014; Wang et al., 2017) and regular room layout (Manhattan assumption) (Ikehata et al., 2015; Khoshelham and Diaz-Vilarino, 2014; Tran H. et al., 2019). For the detection and the reconstruction of the floor or ceiling, it is presumed that the height is not changing (Mura et al., 2014; Ochmann et al., 2016). There are few works which explore beyond the Manhattan World assumptions (Mura et al., 2016) and deal with the 3D environment with a high number of reflective surfaces (Nikoohemat et al., 2017; Nikoohemat et al., 2018a).

Doors are essential for disaster management, indoor navigation and checking the regulations for safety in a building. Detection of doors is studied by (Adan and Huber, 2011; Diaz-Vilarino et al., 2015; Nikoohemat et al., 2017; Quintana et al., 2018) for 3D reconstruction and navigation. Experimentally, identifying open doors in the data is less complex than closed doors. Ray casting (also known as the occlusion reasoning) is one of the solutions which is used by researchers (Adan and Huber, 2011; Ochmann et al., 2016; Xiong et al., 2013) to detect openings (windows and doors). By this method, a ray is cast from the position of the laser scanner to the surface where a door or opening can be identified. If the ray intersects the surface and hits an object behind the

surface, then the intersection point on the surface is labeled as an opening pixel. However, the exact border of the opening remains unclear in ray casting method depending on the resolution of data. To detect closed doors, Diaz-Vilarino et al. (2015) use images in combination with point clouds. In the absence of images and color for the data, an altered method is required to identify closed doors. In a different approach, applicable for mobile laser scanners, Nikoohemat et al. (2017) and Elseicy et al. (2018a) use the trajectory of the mobile laser scanner to find the points above the trajectory which represent the top of the door and to locate the door frame. Using their method is possible to identify closed and open doors which are crossed by the MLS trajectory.

Stair modeling is another challenge in the reconstruction of a multistory building. Although less attention is devoted to the detection and the modeling of stairs and their surroundings, they are considered a complex but yet important structure of a building. Stairs can be modeled using shape grammars since they have a regular structure (Boulch et al., 2013; Schmittwilken et al., 2007). Another approach is by fitting a slanted plane to the stair ramp and finding the orientation and extension of the stairs (Sanchez and Zakhor, 2012). A plane segmentation method is used by Oßwald et al. (2011) to model stairs for humanoid climbing stairs. When using a wearable mobile laser scanner such as a backpack, it is possible to use the trajectory to detect the stairs (Nikoohemat et al., 2018a; Staats et al., 2017). Other methods are developed for stair modeling, pathfinding and robotics based on voxels, octree and histograms (Bansal et al., 2011; Fichtner et al., 2018).

From navigation aspects, many works have been carried out on presenting the conceptual models and standardization of navigation (Brown et al., 2013; Jung and Lee, 2015). Unfortunately, less attention is devoted to using cluttered data in complex environment for indoor modeling. Most of the 3D models for indoor navigation are simple and based on synthetic models (Khan and Kolbe, 2013) or represent simple cases such as shortest path (Broersen et al., 2015; Girard et al., 2011; Yang and Worboys, 2015). In a different approach by Alattas et al. (2017), the access right is used for indoor navigation. This is a new approach to design more flexible navigation routs. Similarly but with a different approach, Diakité and Zlatanova (2018) introduce the Flexible Space Subdivision (FSS) concept for dynamic use of the space and context-aware pathfinding. Based on their method, space can have different functions and based on each function, a navigation graph can be generated.

Consistency control of the 3D model after reconstruction is not investigated sufficiently in the literature. Just few researchers (Gröger and Plümer, 2010, 2009; Liu et al., 2014) suggest methods such as using grammar to control the topology and correctness of the generated model. The consistency of 3D city

modeling (Gröger and Plümer, 2009) can be applied for indoor modeling in some cases. In (Liu et al., 2014), a probabilistic grammar is used to generate consistent semantic information of an indoor scene. A constrained grammar is suggested by Gröger and Plümer (2010) for controlling the correctness of a model-driven model. However, the authors do not use real data to demonstrate the robustness of their method. Validating the 3D model from semantic and geometry aspects is an important line of research that needs more attention in the future.

4.3 Overview

In this section, we give a general overview of the 3D reconstruction pipeline and its application for emergency cases. Figure 4.1 shows the overview of steps (stairs modeling is not reflected and is explained in detail in the Section 4.5). To keep the overview short, important parameters are discussed in the methodology and at the end of the chapter in section 4.7.

The input data of our pipeline is a point cloud collected by mobile laser scanning (MLS) or terrestrial laser scanning systems (TLS). When the data is from an MLS the trajectory of the system is required for some of the algorithms (e.g., building levels separation and door detection). A trajectory is a separate point cloud representing the continuous location of the scanner's location during the acquisition. The trajectory is synchronized with the point cloud and the time attribute in both point cloud and trajectory is used to associate the point in the point cloud with their corresponding device location at a certain time. Each dataset is subsampled to reduce the number of points to accelerate the processing time and for a smoother visualization. For subsampling the average point spacing of less than 0.05 m is preserved.

Our pipeline starts with separating the building levels and stairs. Similar to (Nikoohemat et al., 2018a), the trajectory is segmented to horizontal and sloped segments, where each horizontal and sloped segment represents the levels and stairs, respectively. Each segment in the trajectory is used to cluster the associated point in the point cloud belonging to the same level or to the staircase in case of sloped segments, see section 4.1 for more details. We call each cluster a building segment (Figure 4.1b) which is processed separately for reconstruction of permanent structures or stairs. Each building segment is processed to planar segments using a surface growing method (Vosselman et al., 2004) and a smoothness threshold of 8 to 10 cm is set depending on the noise of the data for MLS datasets and 4 cm for the TLS dataset (Figure 4.1c). All segments are divided into almost-horizontal and almost-vertical using a threshold of 45 degrees. An adjacency graph is created based on the adjacency of segments using a threshold of 10 cm (d_{adj}) and minimum supporting points of 500 per segment where nodes represent the segments (Figure 4.1c).

Analyzing the topology relation of nodes in the graph using suggested heuristic rules result in labeled segments to permanent structures (including wall, floor and ceiling) and clutter containing unknown points and furniture (Figure 4.1d). Then a semi-automatic approach is applied including two main steps: 1. a visual correction of labels for misclassified walls and 2. the automatic extension of permanent structure segments to their plane's intersection for disjoint segments (sections 4.2.2 and 4.2.3). In section 4.4.3, parallel wall surfaces are recognized and merged into one segment as a volumetric wall which is represented by a parametric rectangle. The rectangle is located in the center of the wall and the width is thickness of the wall. Figure 4.1e shows volumetric walls represented by boxes as solids. To this end, the permanent structure of the building is reconstructed in a 3D model showing the walls, floor and ceiling. However, for emergency responses, the notion of rooms is required in addition to knowledge about doors, obstacles and access to the stairs.

We exploit the trajectory of the MLS device to detect the doors. A method from (Nikoohemat et al., 2017) is adopted where the doors crossed by the trajectory can be identified (Figure 4.1f) and a parametric door model with fixed dimensions is added to the model. After recognizing the permanent structure and separating the clutter, furniture is selected using a connected component analysis on points which are labeled as clutter. The details are explained in section 4.4.4. A new method is suggested in section 4.4.5 for reconstruction of the stairs where the input data is the coarse location of the stairs separated as a building segment from the rest of the point clouds. Stair detection and modeling have two steps: 1. detection of the stair ramp using a surface growing segmentation and selecting ramps with a slope of 25 to 50 degrees, 2. applying another segmentation with finer parameters on detected ramps and creating a constrained adjacency graph. The nodes in the graph represent segments in the ramp which are perpendicular with a tolerance of 10 degrees and having a supporting point more than 500 points.

Section 4.5 is dealing with reconstructing the rooms' polyhedra from permanent structures and when adding the furniture, extracting the remaining spaces as navigable space. In previous steps, we reconstructed walls, floor and ceiling and assured the enclosure of the spaces by extending disjoint structures. Rooms modeling is based on a 3D Boolean Operations on solids to extract the enclosed spaces encapsulated by permanent structures. Each permanent structure (wall, floor and ceiling) is a solid which was reconstructed in section 4.4. In Figure 4.1, room extraction is applied in Figure 4.1e and the result is shown in Figure 4.1g, skipping Figure 4.1f as an intermediate step for door detection. Obviously, during the room reconstruction, doors are reconstructed as a closed-door or part of the wall to guarantee an enclosed space. Following reconstruction of the rooms, by including the bounding box of furniture in the model, it is possible to apply the Flexible Space Subdivision

(Diakité and Zlatanova, 2018) framework (FSS) and identify the Object Spaces (O-Spaces), Functional Spaces (F-Spaces) and Remaining Spaces (R-Space), see Figure 4.1(h, i). Unlike Diakité and Zlatanova (2018) which uses simulated data to demonstrate the FSS for 3D indoor navigation, we use real data created from point clouds reflecting the as-built situation of the buildings. Finally, three heuristic rules are suggested to control the consistency of the model for emergency applications. Based on those rules, every room should have at least one door and every two spaces in the model at different levels should be connected by a network in the connectivity graph (Figure 4.1i). The results are evaluated on four datasets where the buildings have complex structures including arbitrary room layout, non-horizontal ceilings, ramps, and glass surfaces. The following sections discuss the methods in detail.

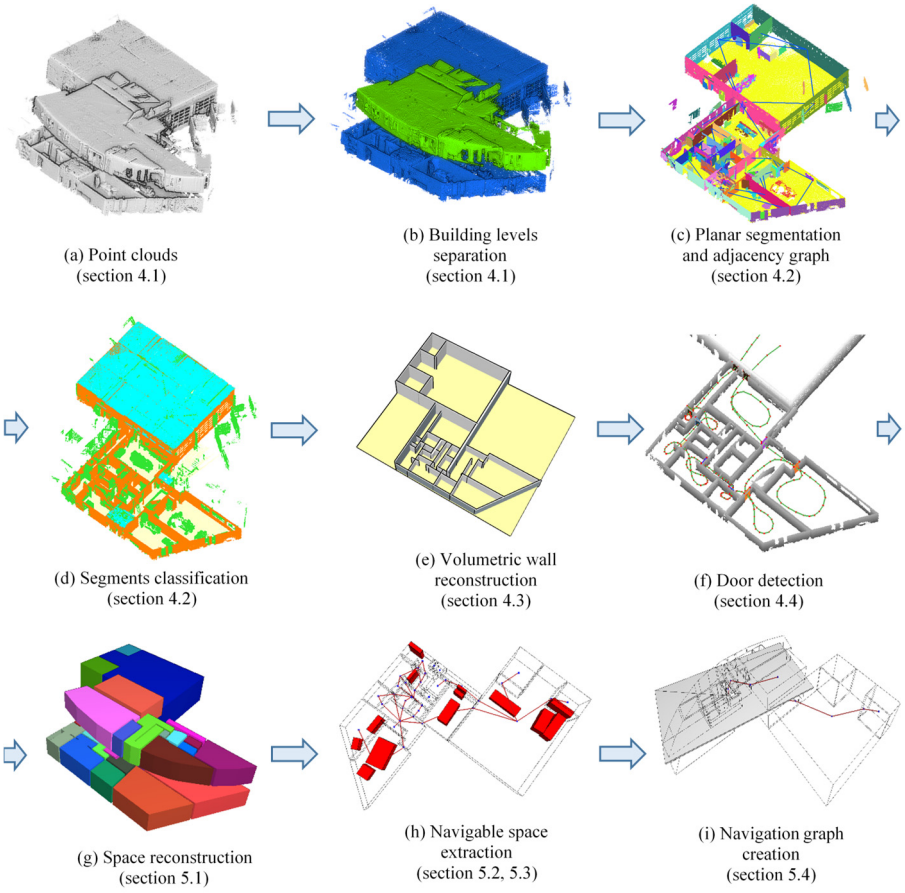


Figure 4.1. *Pipeline overview.*

4.4 3D Reconstruction of Permanent Structures, Openings and Stairs from Point Clouds

Our pipeline starts with preparing the data for the detection of permanent structures. Walls, floors and ceilings are the main focus of the permanent structure detection algorithm. Later other important objects such as doors, stairs and furniture will be automatically identified and included in the model. The data is mainly collected by MLS devices, including pushing-cart and handheld systems. The trajectory of the mobile laser scanner is a useful source to interpret the scene, for example, for separating the levels of buildings in case of a multistory building. The data preparation is mainly purging the noise caused by the reflective surfaces, as explained in (Nikoohemat et al., 2017). The process of classification of point clouds to walls, floors and ceilings starts by a piecewise planar segmentation and then creating an adjacency graph. A heuristic method (Nikoohemat et al., 2018a) is applied to analyze the adjacency graph to separate the permanent structure from the clutter. Furniture is classified as the clutter in our pipeline. In the following, the methods are explained in the details.

4.4.1 Separating the levels and stairs

Every level of a building is a horizontal space which is connected via stairs to the other levels. In complex buildings, sometimes space is extended vertically to the other levels and in the architecture is referred to as a Mezzanine. Therefore merely using a histogram generated from z-axis (Oesau et al., 2014; Turner and Zakhor, 2014) is not sufficient to separate the floors. Moreover, distributed points over the z-value in a sloped ceiling does not create a pick in the z-histogram. Nikoohemat et al. (2018a) suggest using the MLS trajectory to separate the point clouds associated to each level of the trajectory because separating the trajectory of the MLS into the levels is simpler than separating the point clouds. Their method is based on using the trajectory and the timestamp attribute which is synchronized with the timestamp in the point clouds. Obviously, this method can be only used for mobile laser scanners because they have a trajectory and the time attribute is available in both point clouds and the trajectory. The trajectory is a separate point set representing the discrete positions of the laser scanner during the data acquisition. By segmenting the trajectory to separated levels and slopes (representing the stair's ramps) the associated point cloud can be segmented as well using the timestamp for each segment of the trajectory. Segments in the trajectory with slopes are selected to collect associated points with the stairs. After separation of levels and stairs, each set is processed separately for detection of permanent structures.

4.4.2 Detection of Permanent Structures

Permanent structures are walls, floor and ceilings. Intuitively, walls are below the adjacent ceiling and above the adjacent floors. In contrast to the other works that assume walls are vertical and ceilings and floors are horizontal, we lift these constraints in our pipeline. Moreover, any arbitrary wall structure can be used as input to the algorithm and there is no need to align the data to the axis (non-Manhattan-World). The only assumption is the z-axis as the gravitational axis.

4.4.2.1 Adjacency Graph

To create an adjacency graph, the point clouds are segmented using a planar surface growing algorithm (Vosselman et al., 2004). Each node in the graph is associated with a segment, and each edge represents two adjacent segments (Figure 4.2). Two segments are adjacent if their points are within a specific distance defined as the adjacency distance (d_{adj}). Then all the nodes in the graph are attributed to *almost-vertical* and *almost-horizontal* based on the normal vector angle of their segments using a threshold of 45 degrees. For example, the nodes which the difference between the normal vector angle of their segment and the positive direction of z-axis is less than 45 degrees are attributed as almost-horizontal. Ramps and floors are almost-horizontal. Edges of the graph also are attributed based on the type of adjacency which they represent. We identified three types of adjacency: *wall-wall*, *wall-floor* and *wall-ceiling*. An edge is wall-wall if both nodes are almost-vertical. An edge is wall-floor if one node is almost-vertical and the other almost-horizontal; and the center of the almost-vertical segment is higher than the center of the almost-vertical segment. This implies that a wall-candidate should be above the floor-candidate. The same applies to a wall-ceiling edge with the difference that the segment of the almost-vertical (wall-candidate) should be below the almost-horizontal (ceiling-candidate).

After creating the adjacency graph, we use several heuristic rules according to Nikoohemat et al. (2018a) to label the nodes and their associated segments. Four labels are considered for the nodes: walls, floors, ceilings and clutter. In the adjacency graph, each node is examined one time and based on the number of edges and their attributes; the node obtains a label. Rule 1 suggests that if a node has one or more than one wall-ceiling edge, then it is a wall. Rule 2 and rule 3 suggest if a node has more than two wall-ceiling or more than two wall-floor edges, then it is a ceiling or floor, respectively. Clearly, a node with a floor or ceiling label should not have any wall-wall edge. Some extra soft rules control the labeling process to avoid false-positive detections. For example, almost-horizontal segments which are labeled as a ceiling should not have more than 90% overlap with each other. This soft rule excludes some horizontal segments such as ventilation canals and shelves near the ceiling or

tables which are attached to walls to be classified as a permanent structure. Moreover, the floor and ceiling nodes should have a distance of more than 1.5 meters. The label of each node is assigned to the associated points in the segment. Consequently, the output of the adjacency graph is labeled point clouds with four different classes as walls, floors, ceiling and clutter. Note that creating the adjacency graph and exploiting it to identify the permanent structures are not restricted to the assumption of vertical walls or horizontal floor and ceiling. Slanted walls, ceilings and ramps can be identified as well by analyzing the adjacency graph. For example, in the TU Delft building (Figure 4.3) part of the floor is lowered, and it is connected with two ramps to other floors. Since the ramp is classified as an almost-horizontal surface and is connected to other floors, it can be classified and modeled as part of the floor. The next step is generating the volumetric walls for a watertight 3D model.

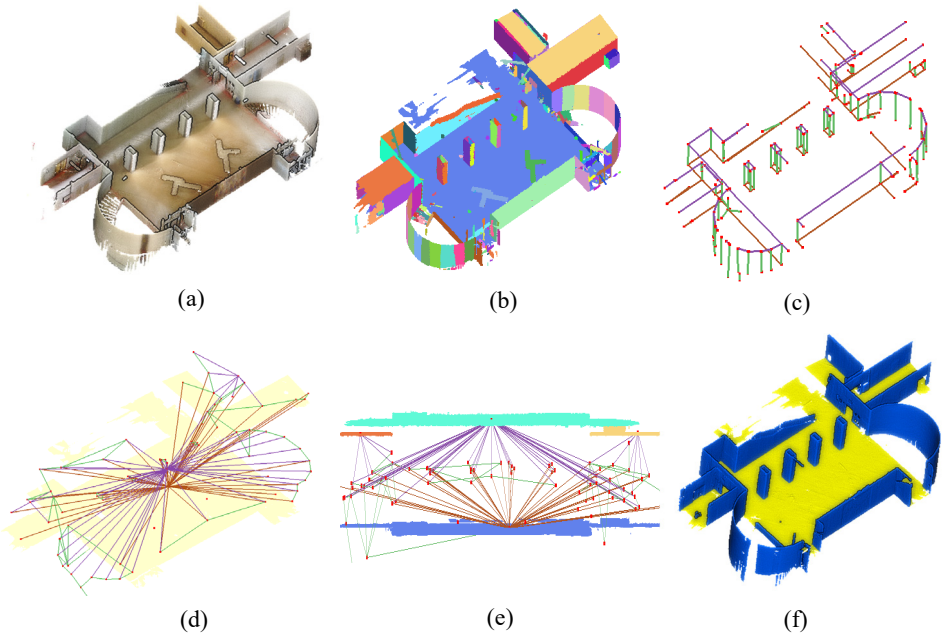


Figure 4.2. *The process of identifying a permanent structure. (a) point clouds, (b) points segmented using the surface growing algorithm, (c) intersection between adjacent segments, (d) and (e) the adjacency graph where edges are colored by three classes (wall, floor and ceiling), (f) detected walls (blue) and floor (yellow). For simplicity of the figure, an area with minimal clutter is selected. The dataset is acquired by NavVis Trolley system* (Huitl et al., 2012).

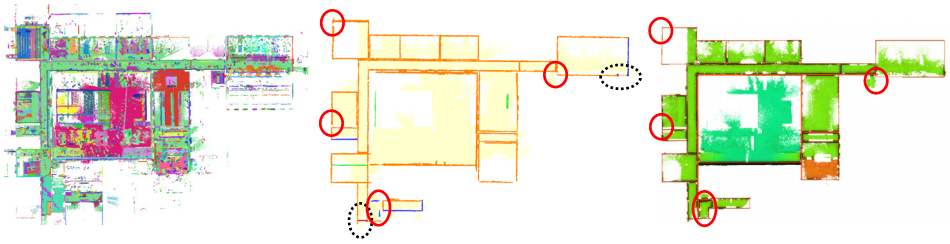
4.4.2.2 Visual Correction of Segments

The labeled segments from the previous section need to be visually inspected to avoid unexpected errors. For example, some of the clutter attached to the ceiling could be misclassified as a wall. The visual inspection contains two main

operations: 1. changing the label of a misclassified segment, 2. extension of two segments to the intersection of their planes in case of large data missing. The first correction is a semantic refinement and the second correction is a geometry refinement. Since the segments are color-coded based on their semantic label (Figure 4.3b, green and blue), changing the class label is quick and it takes several minutes for a dataset of 20 rooms. The extension correction is performed after the automatic undershoot correction (section 4.2.3), where the gap between two segments is larger than the extension threshold. This correction happens in rare cases where part of the data is missing because of large occlusions or unreachable regions in the room (Figure 4.3b, dashed circles).

4.4.2.3 Automatic Undershoot Correction

An undershoot error happens when two permanent structures are not connected either because of the occlusion or missing data during the scanning. These disjoint structures (segments) should be connected to reconstruct a topologically correct 3D model (Figure 4.3b, red circles). Finding undershoots in the data is not simple and fixing them needs a user interface and an expert user. Therefore, we find and fix undershoots automatically. First, for each segment, an oriented minimum rectangle is generated and the best fitting plane of the segment is calculated from the supporting points. For the automatic correction of undershoots, all the rectangles are sorted by their area. Every two nearby rectangles are intersected using their planes if the intersection line is within a distance of the edges of each rectangle, then two rectangles are extended to the intersection line (Figure 4.3c). The extension threshold (d_{ext}) should not be larger than a narrow hallway. Otherwise, the walls on two sides of the hallway will be incorrectly extended. For example, the corridor in Figure 4.3 has a width of almost three meters and the extension threshold is set to less than three meters. Consequently, an extension operation is required in the location of dashed black circles in Figure 4.3b. Experimentally, running the undershoot correction process for each semantic class separately results in a better outcome. We perform the algorithm first individually per class wall, floor and ceiling, then between wall and ceiling, and wall and floor.



(a) (b) (c)

Figure 4.3. *The process of visual and automatic improvements. (a) The top view of segmented point clouds, (b) detected walls (orange), false-positive walls (green) and missed walls (blue). During the visual inspection, the labels are changed and large data gaps are repaired by an extension operation (dashed black circles). Red circles show the walls which are not connected. (c) Automatic extension of walls to the intersection of their planes repairs the disjoint walls. The points for the floor are colored by height.*

4.4.3 Reconstruction of volumetric walls

The goal of this section is to reconstruct volumetric walls from wall segments. A volumetric wall is composed of smaller segments on both sides of a wall and in this part of the pipeline the algorithm identifies which segments belong to the same wall. In the previous step, wall segments are corrected by automatic undershoot correction. A correct 3D model should have consistent room layout and topology, which means there should not be a gap between rooms. In this step, we reconstruct the walls by detecting two parallel faces of a wall and merging them into one volumetric wall. A volumetric wall is a parametric object containing *width*, *height*, *length*, *normal vector* and *center*. The width represents the thickness of the wall and the height and length define the spatial extension of the wall. The normal vector and the center identify the orientation and location of the wall. To reconstruct a volumetric wall, every two rectangles are checked together. A new rectangle is created from nearby parallel (d, θ) rectangles, where the d is the distance between two planes, calculated from the center of the smaller segment to the plane of the larger segment, and the θ is the angle between normals (Figures 4.4a and b). The parameters of the new rectangle should be recalculated in a way that spatial extensions are extracted from the larger face and the width is the distance between two parallel faces (d). The normal vector and the center of the new rectangle are obtained by computing the weighted average of the normal vectors and of the centers of the two faces based on their area. Note that in practice because of the clutter and the occlusion in the data, on each face of the wall there could be more than one segment which should be merged into one rectangle based on their proximity and co-planarity. Therefore, one face of the wall can grow larger and other smaller co-planar faces are merged into it and each time parameters are updated.

Façade walls and walls which are not accessible from both sides (e.g., because of the occlusion) only have one face in the data. In such cases the measured face is offset to the opposite direction of the normal, assuming the normal vectors are flipped towards the position of the laser scanner. The offset distance can be user-defined or can be extracted from the neighboring walls. The floor and ceiling are reconstructed with a user-defined thickness. For multistory buildings, the thickness of the ceiling of the lower level can be calculated from the floor of the upper level. Volumetric reconstruction of walls, floors and ceiling assures that no empty space is generated between the levels of a building or the rooms of the same level.

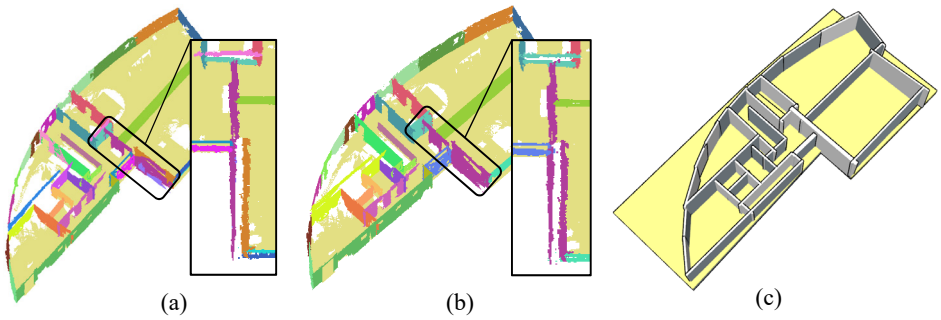
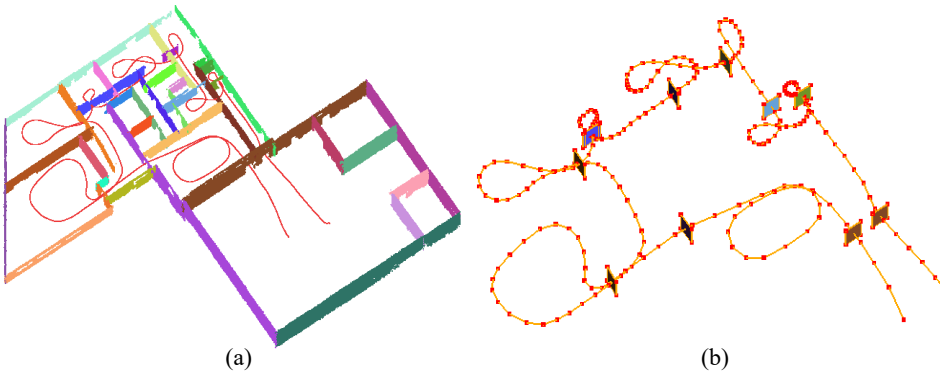


Figure 4.4. *Shows the process of generating volumetric walls; (a) detected walls before the merging process. (b) Two faces of walls are identified and merged into one wall and the distance of the planes is stored as the thickness of the wall. The small insets show the top view of walls before and after the merging process, (c) the generated volumetric walls. Notice the curved wall is modeled as smaller rectangular faces.*

4.4.4 Detection and addition of doors and furniture

Doors in our model are the openings which connect two spaces. Since we use an MLS device to collect the data, at least one of the doors of each visited room is crossed by the trajectory of the mobile laser scanner. Crossing the door with the trajectory is exploited by Nikoohemat et al. (2017) to detect doors. This method can be specifically useful even if the door appears as a closed-door in the data because a door can be closed before or after scanning. However, their method is for the cases that walls are unknown in the data and for detecting doors in the whole unlabeled point clouds. Since the walls are known for us, we adapt their method and intersect the detected walls with the trajectory to identify the approximate center of a door candidate. Before that, the trajectory should be sorted by using the time attribute and be converted to line segments by connecting the successive points (Figure 4.5b). Note that the intersection point necessarily is not in the center of the door, but it gives an estimation of the door location. Obviously, doors which are not traversed by the trajectory during the scanning, remain unidentified in our method. Some researchers (Adan and Huber, 2011; Nikoohemat et al., 2017) use ray casting to detect openings, but we limit the scope of our work to the doors crossed by the

trajectory, which is sufficient for navigation purposes. After detecting the location of the door, an oriented bounding box aligned with the direction of the wall is inserted in the model. The extension of the door (height and width) is user defined and the thickness is inherited from the wall thickness. Therefore, a double door and a single door are modeled the same in our 3D model.



(a) (b)

Figure 4.5. *(a) Top view of wall segments and part of the trajectory. (b) The doors are detected with the intersection of the line segments of the trajectory and walls.*

Including furniture in the model is useful for creating an insightful space subdivision that considers potential obstacles. In case of emergency, the auditing experts can evaluate if there is enough navigable area in one space or whether the emergency exits are not occluded by furniture. In our work, the furniture is classified as clutter in section 4.2. We include the larger pieces of furniture by using a connected component algorithm. First, the cluttered points in a neighborhood of the permanent structure are removed to make a clearance between furniture and adjacent the permanent structures. Then a connected component analysis is applied to the remaining points. A maximum distance threshold of 10 cm between points is set for this analysis, noticing that this threshold should be larger than the average point distance and smaller than the clearance between furniture and the permanent structures. The components with a larger number of points are chosen as the obstacles to be included in the model. The reason that we make a selection is that there is a lot of clutter near the ceiling or on the walls, such as pipes, small lamps, shelves, curtains or noise from reflection, which make the space subdivision cluttered. An oriented bounding box (OBB) is generated from each object and included in the model representing the occupied space with the furniture (Figure 4.6).

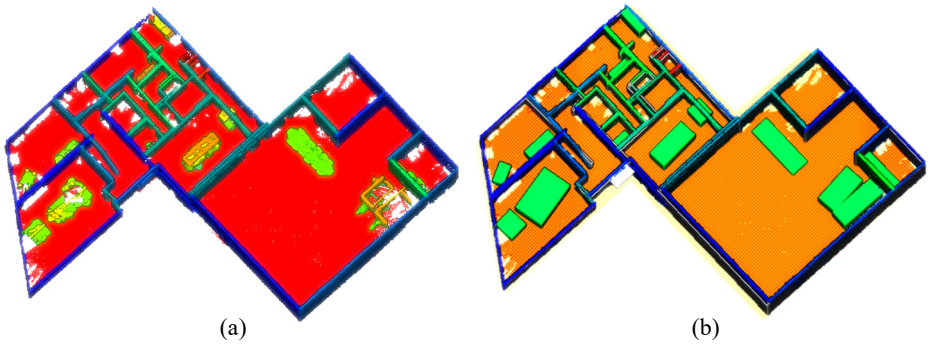


Figure 4.6. *(left) Including large pieces of furniture in the model and (right) creating an oriented bounding box (green boxes) for a flexible space subdivision.*

4.4.5 Stairs Modeling

Although the coarse location of staircases can be separated from the rest of the point clouds, the number of steps and the exact model of the stairs need to be reconstructed. Therefore, we develop a method using the adjacency graph *G (V, E)* to detect the exact size and location of stairs. Our method for stair modeling has two steps: 1. similar to (Sanchez and Zakhor, 2012) a planar segmentation is employed to search the ramp of the stairs (Figure 4.7b).; 2. a fine planar segmentation is applied on the detected ramp to identify planar segments representing each rise and tread in the stairs (Figure 4.7c). A planar segment is a valid staircase ramp if it inclines between 25 and 50 degrees are respecting the xy-plane. For the planar segmentation of the ramp a surface growing algorithm is applied with a point-to-plane distance set to 20 cm. We choose a larger value for planar segmentation for ramp detection to make sure points belonging to the steps fall into the ramp segment. A minimum enclosing rectangle is created for each valid ramp and the inliers (considering a buffer) would be the points which will be processed in the second step (see Figure 4.7b). The risers and treads should be identified in this ramp. In the second step, a surface growing segmentation with finer parameters (point-to-plane distance= 5cm) generates the segments which represent the nodes (V) in the adjacency graph (see Figure 4.7c). If two segments are adjacent and create a perpendicular angle with a threshold of 10 degrees, then an edge ($e \in E$) connects two nodes. Smaller segments with fewer than n supporting points (e.g., n = 500 points) are excluded from the graph. On the stairs, there is clutter (e.g., people during the scanning), and walls and bars are attached to the steps. Therefore, to extract the exact steps which form the stairs, *the longest path* is extracted from the undirected graph (Karger et al., 1997). We start from a random node ($v \in V$) and find a node ($x \in V$) with the longest distance from v using a breadth-first search (BFS). The discovered node x is an end node in the graph. By applying another breadth-first search from node x, the longest path in the graph is identified (see Figure 4.7d). Note

that the edges are unweighted and the graph should not have cycles. Nodes belonging to the longest path represent the steps of the stairs. For each node, the corresponding segment is selected, and a minimum enclosing rectangle is derived. Since the shape and size of the segments are varied, the width and length of a step are approximated from the majority of the rectangles. Then all the rectangles are adjusted with the new width and length. To extract the number of steps, as every step of stairs has two nodes in the graph, one for the riser and one for the tread, the number of steps is the round of *n/2* where *n* is the number of nodes in the longest path. Eventually, vertical space is generated from the minimum rectangle of the stairs, which represents the staircase space. This vertical space is a virtual space that connects the levels in a building with an extension from the floor of the lower level to the floor of the upper level. Note that in most buildings, the surrounding walls of stairs are not connected to the ceiling and sometimes they are made of metal bars or glass. Therefore, we create this virtual space to subdivide the space of stairs from other spaces for navigation purposes.

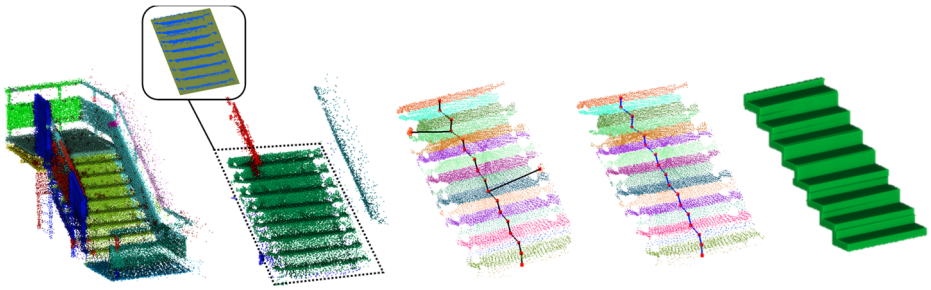


Figure 4.7. *The process of detecting and modeling stairs. (a) Segmented point clouds, (b) detected stair ramp, and selection of inlier points in the oriented minimum rectangle of the ramp (the small inset), (c) and (d) a constrained longest path graph showing the correct steps, (e) the reconstructed model with correct inclination and number of steps.*

4.5 Room Reconstruction and Flexible Space Subdivision (FSS)

Having the permanent structures created in the previous step, now a method is explained to generate enclosed spaces as polyhedra. Remaining spaces in the presence of obstacles are generated as well, to create more detailed spaces for disaster management cases. In the context of disaster management, several aspects of indoor navigation become critical. One of them is a good knowledge of the occupied/unoccupied spaces and openings' configuration. The Flexible Space Subdivision (FSS) (Diakité and Zlatanova, 2018) is a framework aiming to provide a space partitioning that reflects the complexity of the indoor environment. The approach produces three main subspaces: the occupied (O-Spaces), the functional (F-Spaces) and the remaining free spaces (R-Spaces).

These latter allow describing the space complexity in terms of spatial occupancy induced by the physical and functional characteristics of the indoor objects. The FSS thereby provides a spatial model that supports fine-grained indoor navigation and makes it possible to account for advanced constraints during the navigation, while ensuring enough granularity for describing precise localization.

In order to implement the FSS, one of the most critical features needed is the explicit representation of the indoor spaces. Such features can be generally obtained natively from BIM models such as IFC (e.g., the IfcSpace class) However, since we start from a point cloud, the indoor spaces are not readily available in our workflow. Therefore, we present a way to recover them. Similarly, other critical features to the FSS, already identified in the previous steps, will also be considered, such as the openings and obstacles.

4.5.1 Reconstructing the indoor spaces

The indoor space can be conceptually thought as the space that is encapsulated by the permanent structures of a building (walls, floors, ceilings, etc.). The features that we are explicitly representing correspond to that description. Starting from the wall, floor and ceiling volumes, we reconstruct the volumes that could be described as the rooms of the building. Our approach is to extract the closure of the model's interior using regularized Boolean operations (Tilove and Requicha, 1980). Such operations produce uniquely closed volume and discard Boolean results of lower dimension (faces, edges or points). In other words, the closed volumes encapsulated by the inner parts of the structural elements are sought. The process consists in successively uniting the volumes of the scene (boxes of the walls, slabs, ceilings, etc.) so as to end up with either several connected components corresponding to the enclosed volumes and the shell of the whole input set, or simply one unique connected component if there is no enclosed volume. Thus, only indoor spaces with full closure can be reconstructed.

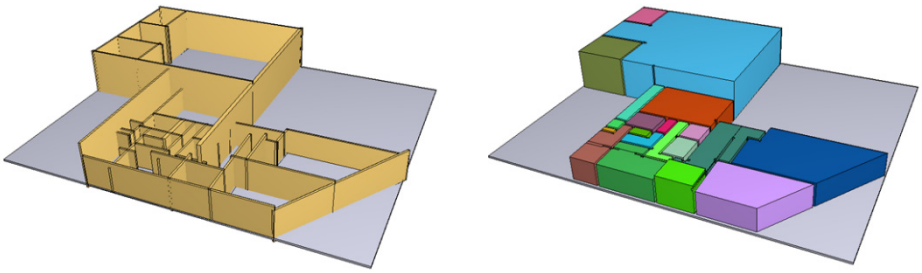


Figure 4.8. *(Left) Walls and the slab is forming the rooms of a building floor (the ceilings are hidden for the sake of clarity). (Right) Reconstructed rooms are resulting from the space closure of the structural elements.*

Figure 4.8 shows an example of the resulting spaces from the room reconstruction process. After extracting the walls, slabs and ceilings in the form of closed volumes from the point cloud, we have to ensure that any of those elements have physical contact with its neighboring elements (Figure 4.8, left). Because Boolean operations are generally very sensitive to precision issues, the intersections between the structures are exaggerated to ensure the contact. This does not disrupt the generation of the indoor spaces as it guarantees the formation of the space closure where they should happen in the model (Figure 4.8, right). A limitation to this approach is that spaces which are not entirely bounded by structural elements will not be reconstructed. Furthermore, because our 3D models are reconstructed from the point cloud, areas that are not scanned may cause gaps that make it difficult to determine with certainty if there should be a closure or not, without prior knowledge of the actual building.

The geometry of the volumes resulting from the process is similar to those of the space features that can be found in BIM models, such as the *IfcSpace* class in IFC. Such features do not purposely consider the indoor obstacles but maintains a spatial link of containment with them. In our case, as we rely on the FSS framework, we proceed to a further subdivision of the space in order to explicitly distinguish between the free and the occupied space.

4.5.2 Identification of the occupied spaces (O-Spaces) and the remaining free space (R-Spaces)

The indoor objects populating the space occupy a central role in the FSS framework as they are critical to indoor navigation applications, even more for emergency response. The point cloud which were classified as clutter during the modeling process are further classified to furniture (e.g., tables, sofa, chairs) and clutter (e.g., objects on the wall and ceiling, curtains, shelves), see section 4.4, from which larger pieces of furniture are selected and each is replaced by an oriented bounding box. In fact, as explained in (Diakité and Zlatanova, 2018), accounting for the detailed geometry of the furnishing elements would lead to error-prone Boolean operations and a too complex subspace geometry to work with, for a negligible added value. Thus, the simplification of furniture into OBB fits the purpose (see Section 4.4). Furthermore, the resulting simplified volumes correspond directly to the O-Spaces as defined in the FSS, and spatially intersecting ones are aggregated into a single O-Space accordingly. Figure 4.9 (left) shows the O-Spaces contained in their respective spaces.

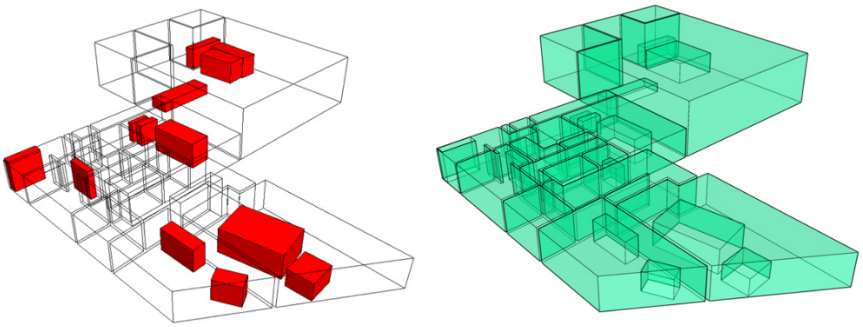


***Figure 4.9**. Obstacles identified as O-Spaces (Left). Non-occupied spaces identified as R-Spaces (Right).*

Subtracting the O-Spaces from the initial indoor spaces leads to the generation of the remaining non-occupied spaces (R-Spaces), which represent the space where all types of navigation can potentially be performed. Figure 4.9 (right) gives an illustration of the R-Spaces generated from the spaces of Figure 4.8, with a transparent view to make visible the parts of the spaces that have been carved accordingly to the detected O-Spaces. The carving operation is a Boolean difference between each room and the O-Spaces that it contains. For this reason, similarly to the walls and slabs, it is necessary to guarantee the spatial contact between the O-Spaces and their containing rooms to avoid having them as flying objects, because this would lead to empty set results from the Boolean operations.

4.5.3 Identification of the functional spaces (F-Spaces)

F-Spaces correspond to the spaces that are induced by the function of an object. Considering such parameters during the navigation allows taking into account the occupancy caused by the usage of indoor objects. From a more technical point of view, F-Spaces allows producing proper position nodes for agents concerning the function of indoor objects. For example, in a context of emergency response, F-Spaces of objects such as extinguishers would allow navigating the agent up to where the object would be accessible. Similarly, an F-Space of a highly flammable object would stand as a space to avoid during the navigation. However, the semantic level of our model does not allow to get enough information about the function of the furnishing elements. For this reason, F-Spaces are limited to the doors in this work.

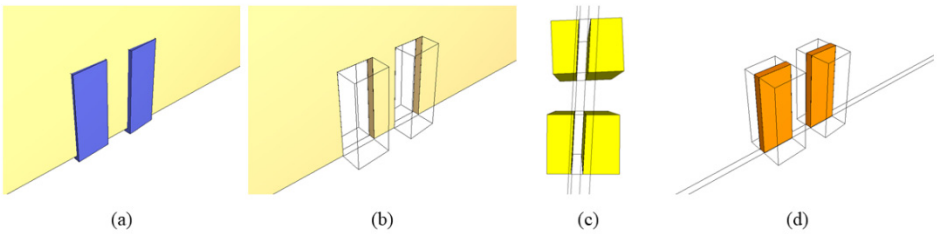


Figure 4.10. *Generation of the opening space and the F-Spaces of doors. (a) Original doors (blue) from the point cloud reconstruction intersecting with the wall (light yellow). (b) F-Spaces (wireframe) obtained by extruding the door volumes of the size of door width from both sides and intersecting them with the surrounding rooms and carving of the door spaces in walls. (c) Top view of the resulting F-Spaces (gold-yellow). (d) Opening spaces (orange) obtained as the intersection between the extruded doors and the wall which can be used as more accurate door volumes.*

From a navigation point of view, openings are transition spaces that connect two spaces. For openings such as doors, their F-Spaces can be seen as the space that is traversed when the door performs an opening movement or simply the space required to access and interact with it (open, close, hold, etc.). Therefore, a door has an F-Space in each part of the free spaces that surround it (in other words, from both of its sides). Figure 4.10 shows how we extract the F-Spaces for door features. The original doors obtained from the point cloud reconstruction, although not very precise, provide a good indication of the doors' location and size. Thus, on that basis, we extrude them in both of their main sides to make sure that we reach the indoor spaces surrounding them (see Figure 4.10c). Hence, we perform Boolean operations again to extract their intersection with the walls which correspond to the opening spaces (it can also be seen as a better estimation of the actual doors, see Figure 4.10b and d, and their intersections with the R-Spaces which correspond to their F-Spaces (see Figure 4.10b and c).

4.5.4 Using the model to support indoor navigation

An indoor navigation system requires several components, among which the most critical ones are a map combined with a spatial model and a localization technology. The former provides a spatial description of the environment that will be explored, and the latter allows to locate and track the guided agent along the suggested paths. From the spatial model, a navigation network (graph) is extracted that reflects the connectivity of the spaces of the environment. Navigation and path planning algorithms will, therefore, rely on that graph to compute the paths that lead to a chosen destination from a chosen starting point, based on specific constraints (shortest path, fastest path, etc.). Most existing navigation systems rely on simplified networks to provide navigation services, which means a graph in which the nodes are the centroid of rooms in the building and the edges symbolize their connectivity. Simple topographic information of buildings (e.g., floor plan) is used to

determine, for example, which room is connected to which corridor, etc. This then leads to a connectivity graph that provides minimal insight into the reality of the indoor environment. A typical path resulting from such system would result in similar navigation instructions: "To go to room D from room A, go to room B, then corridor C and then get to room D." Such guidance heavily assumes that the agents are aware of what room A, B, C and D are, which is not a reasonable assumption in the context of emergency response, where the first responders may interact with the environment for the first time.

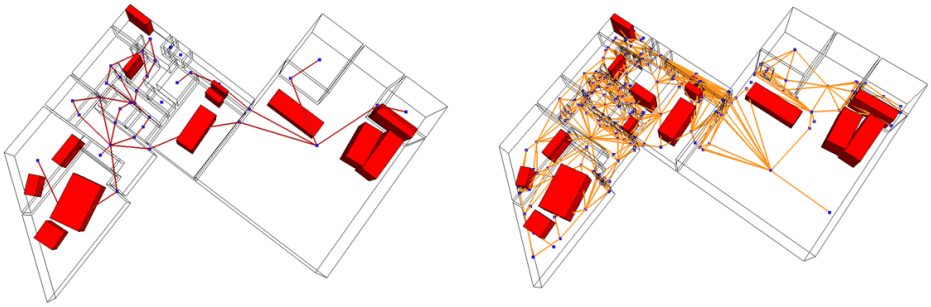


Figure 4.11. *(Left) Simple connectivity graph that simply connects the spaces through their shared doors. (Right) FSS navigation network that uses F-Spaces and R-Spaces for a finer description of the indoor space.*

In contrast, fine-grained navigation networks provide more useful insight into the indoor environment and better support for advanced navigation systems (Afyouni et al., 2012). The 3D models resulting from our reconstruction process, enriched with the FSS framework, are suitable for supporting fine-grained indoor navigation and can be used as spatial support. In fact, the nodes in our network correspond to the centroid of the navigable subspaces (F-Space and R-Spaces). Thanks to their semantic information and topological links with the occupied spaces (O-Spaces), the network offers finer and smarter navigation capabilities. Figure 4.11 shows an example of a simple network in comparison with the FSS network. The advanced network illustrated in Figure 4.11 (right) is resulting from the convex subdivision of the R-Space to allow the generation of more nodes at navigable locations. In comparison with (Gröger and Plümer, 2010) where a regular grid is proposed instead, our approach allows optimizing the subdivision with more nodes only where obstacles or objects of interest are located, keeping the size of the minimum necessary size of the graph. Indeed, our navigation networks, while providing all the possibilities of a basic connectivity graph, also offer the accessible areas in 3D through the R-Spaces. Furthermore, thanks to the available semantic information and with the identification of the F-Spaces, it is possible to derive paths with much more intelligence, considering the navigation context. For example, one could extract a path that maximizes the free space (to account for the equipment of first responders) while crossing as many extinguishers as possible on the way. Another advantage of the FSS as a fine-grained spatial

model, in comparison with other models, is the possibility to fully exploit the 3D space and thereby enable the extraction of navigation graph for agents with a different kind of mobility (Diakité and Zlatanova, 2018). For example, a robot or a drone could be considered as agents and all the spaces close to the ground or the ceiling are then used to compute their dedicated paths. The subspaces in Figure 4.11 (right) result from the F-Spaces of openings and the R-Spaces (which are the free spaces that exclude the indoor objects and the F-Spaces). These latter are further subdivided into convex space cells to ensure that their nodes in the network lay exclusively inside their boundaries.

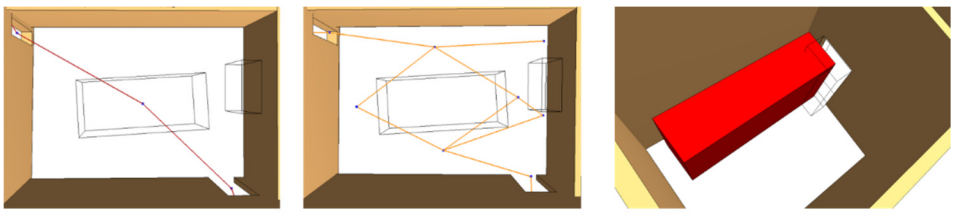


Figure 4.12. *Some advantages of the FSS in the navigation context. (a) A simple connectivity network that does not consider the space occupancy may provide position nodes that are not reachable to an agent (O-Spaces are wire-framed). (b) By relying on the R-Spaces, the FSS network provides exclusively position nodes where space is free of obstacle (O-Spaces are wire-framed). (c) The intersection between the O-Spaces (red) and the F-Spaces of openings (wireframe) indicates potential occlusion.*

Having nodes exclusively on free spaces makes a critical difference in terms of navigation, as illustrated in Figure 4.12. Indeed, as the navigation network is the support for path computation, if it does not reflect the indoor occupancy, the provided path may not be accessible to the agent. The network in Figure 4.12a provides positions that are unreachable for an agent, as it is the place of O-Space. By relying on the FSS (mainly the F-Spaces and the R-Spaces), the positions provided by the network would guarantee unoccupied spaces (see Figure 4.12b). In terms of path planning, this means more reliable paths. Furthermore, when combined with other spatial properties of the subspaces such as their size or volume, it becomes possible also to estimate the suitability of the path regarding its accessibility and the navigation comfort it can provide to a given agent. The usefulness of the F-Spaces of the openings appears more when obstacles lie in front of openings (see Figure 4.12c). Indeed, in such case, the F-Space is truncated to reflect the occupancy. Such information can play a critical role in emergency response navigation, as it may indicate that a room is blocked or hardly accessible, mainly with equipment of important size.

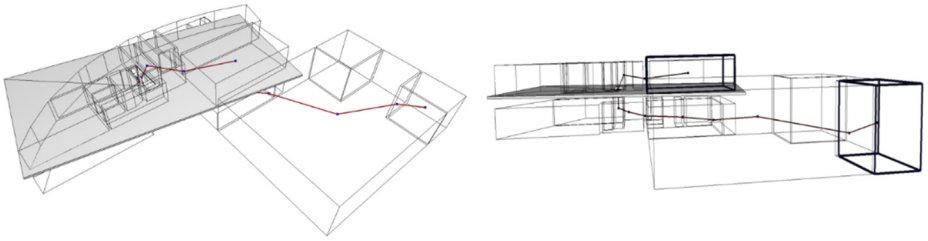


Figure 4.13. *Multi-story navigation network. (Left) The top view shows the slab of the upper floor and the simple network that links a room at the ground floor to a room on the upper floor. (Right) Front view of the generated multi-story network with the highlighted origin and destination spaces.*

Finally, because we could reconstruct the stairs and the space occupied by the staircases, we can generate a multistory navigation network, which is a challenging task from point clouds. Figure 4.13 illustrates a navigation path generated between two spaces at different levels of the building through the opening created in the slab of the upper floor. The latter was made similarly as the doors, by carving a hole using the staircase's space. The FSS can be similarly applied to the model for a fine-grained multi-story navigation network.

4.6 Consistency Check of the Model

As the last step of our pipeline, we verify the consistency of our generated model against three defined heuristic constraints (C). The accuracy of the model regarding the detected doors, stairs and reconstructed spaces can be improved by this consistency check. The focus of this check is the validation of generated spaces in terms of navigation. The three constraints, which are checked on the data sequentially, are described as follows:

> C1. Each room should have an area larger than A m^2 and a volume greater than V m^3.
> C2. Each room should be connected to at least one door.
> C3. There should be a route connecting every pair of rooms in the graph.

To verify C1, a 2D boundary projection on the xy-plane is generated for each reconstructed space. The area of the room is calculated and it should be larger than a threshold, its volume is also checked against another given threshold. This is necessary because, with the Boolean regularization approach, quasi-flat spaces can occur in thin gaps left by structural elements. Thus, only relying on the surface area would not allow invalidating such spaces. Once a room is validated, it passes for the next check. Otherwise, it should be flagged for further control. Later on, flagged items are returned for visual inspection

(4.2.2) to see whether a wall is mislabeled in the process. After correction, the pipeline is repeated to create a consistent model.

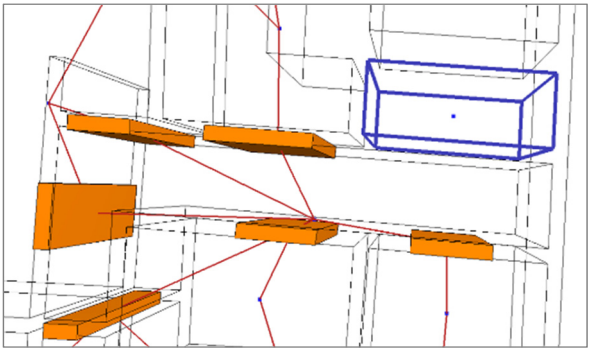


Figure 4.14. *Illustration of the constraint C2. A room (emphasized in bold blue lines) for which the node is connected to no other node of the graph characterize a missing door.*

The constraint C2 checks if each room is connected to at least one door. Otherwise, a missing door is flagged for the room. Although the door detection algorithm could have missed more than one door per room, a room without any door reflects necessarily a problem in the reconstruction. This is also an obvious limitation for navigation purposes. As illustrated in Figure 4.14, by relying on the navigation graph, it is straightforward to identify the problem, as it only requires detecting rooms associated with isolated nodes. Similar to C1, spaces detected in this checking are tagged for further improvement of the model reconstruction.

The third constraint C3 verifies the connectivity of two spaces. It also relies on the navigation network and covers the cases that cannot be detected with C2. We assume that every two spaces in a correct 3D model should be reachable and connected through the navigation graph. Therefore, the process checks the existence of a path between every two rooms in the model. For example, if the connection of level *n* through the reconstructed stairs to the level *n+1* is broken in part of the 3D model, it is possible to identify the broken path and fix it by generating a connectivity graph between two rooms in two different levels. This typically happens when the number of steps in the stairs is not reconstructed correctly and the stairs are not connected properly to the floor of each level. Similarly, if two rooms on the same floor are not connected, it can be because of a missing door or an invalid room.

4.7 Results and Discussion

We conducted our experiments on different types of buildings that go beyond the simple grid and regular structure. To demonstrate a full 3D reconstruction, the selected datasets cover a range of different cases such as the non-

Manhattan World structures, slanted walls and non-horizontal ceilings, ramps, large glass walls, multiple floors and ceilings with different heights. To compare the robustness of our algorithm regarding different laser scanning systems, the data sets are acquired by various laser scanners and represent different ranges of noise (e) from 0.01 to 0.06 meter (Table 4.1). All the datasets contain furniture and a high level of clutter to evaluate the algorithms in terms of occlusion and missing data problems. We tested our workflow on four datasets (Table 4.1) and the results are represented in Figure 4.15. The data of three datasets is collected by mobile laser scanners and belongs to the project Smart Indoor Models in 3D (Nikoohemat, 2019). The Penthouse dataset is acquired by a terrestrial laser scanner which belongs to the related work (Mura et al., 2016) and is selected to test our algorithm on slanted walls and ceilings. For one of the datasets (Fire-brigade 2), we use the professionally made BIM model for the comparison. The noise in the datasets collected by a mobile laser scanner varies between 4 and 6 cm. For pushing-cart systems, the data is less noisy than handheld devices.

Table 4.1. *Results of the different datasets. The third column shows the scanning device. The fourth column is the acquisition system precision which is related to the sensor noise and the MLS algorithm. The fifth and sixth columns show the correct and detected numbers of rooms and doors, respectively using our methods. The doors in Penthouse dataset is not reported as it is acquired by a TLS and is out of the scope of this research. The seventh column indicates the number of planar surfaces with supporting points more than 500 and in the brackets, the number of permanent structures and clutter segments. The visual operations (eighth column) are either changing the label of a segment or extending a wall in the data gaps. Each human intervention is one operation. The last dataset (Penthouse) is from related work* ***(Mura et al., 2016)****.*

Datasets	#Points	Scanning Device	Scanner Precision	#Rooms/ #Correctly reconstructed	#Doors/ #Detected	#Segments (structure/clutter)	#Visual operations	Figures
Fire brigade #1	7.4 M	Zeb1 (handheld)	e < 0.06 m	25 /23	26 /22	818 (323 / 495)	32	15
Fire brigade #2	5.6 M	Viametris (push-cart)	e < 0.04 m	27 /24	30 / 27	909 (186 / 723)	49	1, 4, 5, 6, 8, 9,11, 13,15, 16
TU Delft	3.2 M	ZebRevo (handheld)	e < 0.04 m	18 /16	25 /18	796 (112 / 684)	25	3, 7, 15
Penthouse	2.5 M	FaroFocus (TLS)	e < 0.01 m	4 /4	--	330 (79 / 251)	15	15, 17, 18

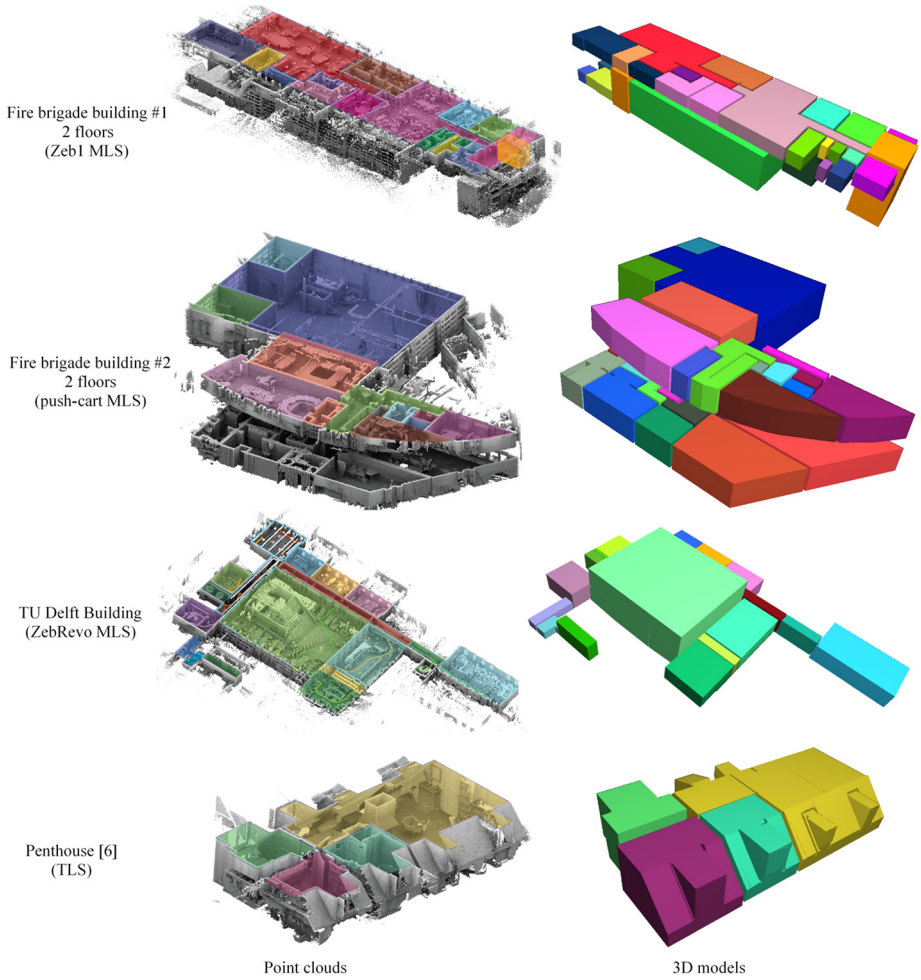


Figure 4.15. *3D models of datasets in the table1. The left column shows the point clouds, masked by the spaces from the model on the right for a better interpretation. The right column shows the 3D models reconstructed with our pipeline (the colors are random). The datasets of Fire brigade #1 and #2 (3rd and 4th rows) are representing two floors. The first dataset belongs to the related work* (Mura et al., 2016) *and demonstrates how our algorithm is capable of handling slanted walls and reconstruction of dormers. For a clear visualization, volumetric walls and slabs are removed in the left images. The empty space between rooms on each floor is filled with volumetric walls. For 3D models with walls refer to other images.*

Comparison with IFC and related work. We compare our result with a professionally made IFC model for Fire brigade building #2. The results show that 95% of the rooms are reconstructed correctly. The precision and recall of the permanent structure are 91% and 95%, respectively. The calculation of precision and recall in Table 4.2 is based on comparing the labels per point which are manually labeled with the labels which are predicted by the

algorithm. F1-score is a harmony of the correctness (precision) and completeness (recall) of values for each class. The room on the second floor has a curved structure wherein our planar reconstruction algorithm is reconstructed with smaller planar surfaces. The firetruck hall is extended from the first floor to the second floor and it shows that our algorithm can reconstruct a fully 3D model of spaces which are extended vertically. The ramp of the stairs is correctly detected and a staircase is reconstructed as a virtual space which connects the first floor to the second floor and the ramp is used for navigation purposes. Individual stairs are not reconstructed for this model because the stairs are scanned with a pushing-cart system only from the lower and upper floor which is not capable of scanning stairs consequently and adequately the data for all steps is not available. 90% of the doors are correctly detected and represented in Figure 4.15 and Table 4.1. The trajectory crosses most of the doors, just a few doors which are not intersected by the trajectory during the scanning are not recognized by our algorithm. One door is recognized as a false positive. The reason is a false positive wall which is crossed by the trajectory. The corridor on the second floor is separated into two spaces because of a false positive wall (Figure 4.15, upper row). The reason is that clutter in the ceiling was identified as part of the wall and was extended to the neighbor walls during the automatic extension process. Consequently, the corridor is subdivided with this wall. The rest of the spaces is correctly reconstructed. To see the spaces, refer to Figure 4.15, fire brigade building #2.

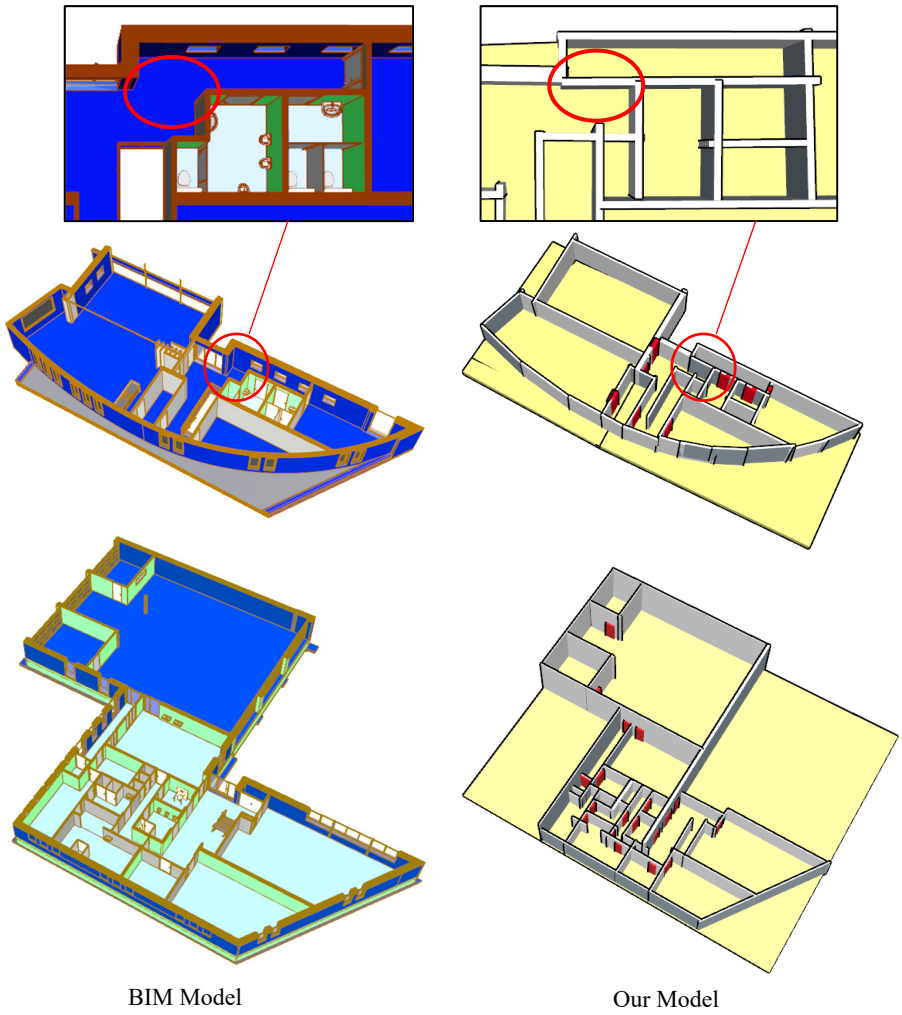


Figure 4.16. *Comparison of a professionally made BIM model (left) with our model (right). The lower row is the 1st floor and the top row is the 2nd floor. Doors are shown in red. Most of the rooms are successfully constructed. The insets show the top view of the corridor, which is divided because of a false positive wall.*

Table 4.2. *The accuracy results for Fire brigade building #2. The results are calculated based on point to point comparison of manually labeled points with automatic classification.*

Class of the permanent structure	Precision	Recall	F1-score
Wall	0.90	0.96	0.93
Floor	0.98	1.0	0.99
Ceiling	0.88	1.0	0.94

The second dataset for comparison is selected from the related work (Mura et al., 2016), which has a challenging structure with slanted walls, dormers, chimney and built-in bookshelves. Because the data is collected with a terrestrial laser scanner, the noise is less than one centimeter and the planar segments are finer than segments in other datasets. This is important when the algorithm generates the adjacency graph and later the minimum rectangles. The results show that our algorithm successfully reconstructs the dormers and slanted walls (Figure 4.17). The walls belong to the chimney are detected in our algorithm, but the chimney space is not reconstructed as space because the walls are not connected to the floor and it does not make an enclosure in the space (Figure 4.18, left). Our method reconstructs the green space in the left corner more accurately. Also, the empty space between the green room and the purple room is modeled more accurately in our model. The wall in the purple room is completely occluded by the built-in bookshelves (Figure 4.17 and Figure 4.18 right). A wall is added to enclose the space during the visual inspection. Note that detection of the doors in datasets which are acquired by TLS (Penthouse) is not part of this work. Therefore, the doors in Penthouse dataset are not reconstructed.

Figure 4.17. *The comparison of our method with the related work (Mura et al., 2016). The point cloud on the right is masked by the spaces from our model for a better reference. The red and yellow circles show the chimney and the built-in bookshelves in Figure 4.18. The chimney is not reconstructed in our model, and the wall behind the bookshelves is inserted manually.*

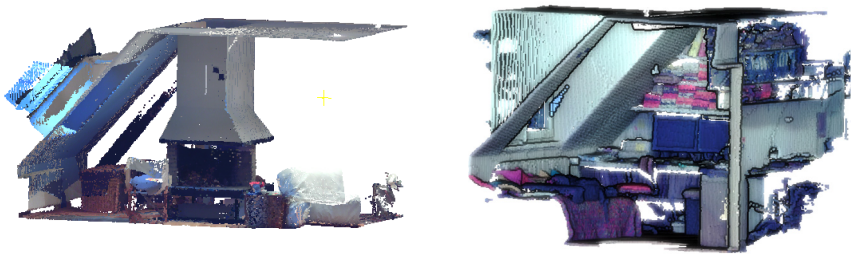


Figure 4.18. *Special cases for modeling the penthouse dataset: Left is illustrating the chimney descended from the ceiling. The walls of the chimney can be detected in our permanent structure algorithm, but its space is not reconstructed because of disconnection to the floor. The right image shows the built-in bookshelves occluding the wall and the dormer, which are reconstructed correctly in our model (purple room in Figure 4.17).*

Visual correction. A visual inspection is performed after the labeling of the permanent structures. This mainly includes checking if some of the clutter near the ceiling are classified as wall and change the label to clutter. Additionally, if a space is not enclosed with walls because of large data missing or the occlusion similar to the built-in shelves in penthouse dataset (Figure 4.18, right), then a wall is added manually. In such cases, the thickness and the orientation of the wall are inherited from the adjacent walls. Additionally, our pipeline can export the volumetric walls and spaces to the standard BIM software formats such as Wavefront format (.obj) to import it into CAD software for further improvements, for instance, for adding windows. We calculate the time and percentage of operations for visual inspection for Fire brigade building #2 dataset. It takes 5 minutes per floor (for a floor with almost 15 rooms) and less than 6% of the total number of segments are modified.

Navigation graph. The navigation graph in this work is a connectivity graph to show the relation between 3D spaces, which are categorized into navigable and non-navigable using the FSS framework. Showing a turn to turn detailed graph is not our goal and it is addressed in many previous works. Our focus is showing the thorough process of subdividing the space into more semantic divisions using a model reconstructed from point clouds and considering obstacles and the full 3D space for optimal routing of different kind of agents. Additionally, we showed how the FSS-based navigation graph could improve the navigation by revealing doors occluded by obstacles (see Section 4.4) and also how it could help to improve the 3D model regarding the presence of the doors per space (see Section 4.5). If the connection between levels or rooms is broken in the navigation graph, further inspection will be performed.

Important parameters. Parameters are reported in Table 4.3 and Table 4.4. The crucial parameters for permanent structure detection belong to the surface

growing and adjacency graph, which are the smoothness and proximity, respectively. The smoothness parameter or point-to-plane distance in surface growing segmentation is chosen considering the noise of the sensor, MLS data acquisition noise and point density. For example, for handled scanners the noise is higher than pushing-cart systems and similarly for pushing-cart systems is higher than terrestrial scanners. For surface growing segmentation a threshold of 8 to 10 cm is chosen for MLS devices and a threshold of 4 cm for TLS devices (Penthouse dataset). The exact values are reported in Table 4.4. The adjacency graph generates the best result with a threshold of 0.10 m for the proximity of adjacent segments. To classify the segments into almost-vertical and almost-horizontal, the angle of their normals with the positive direction of z-axis is set to 45 degrees for most datasets. For automatic undershoot correction of walls, the extension threshold is set based on the data. Experimentally, this threshold should be less than the width of a narrow corridor. For most datasets, a value of 1.0 meter is optimal. During the reconstruction of volumetric walls, a value of 0.80 m is selected for the maximum distance of two parallel planes to be considered as one wall, and their normal vector should not deviate more than 5 degrees.

Table 4.3. *Parameters and their value for permanent structure reconstruction.*

Parameter	**Value**
Adjacency graph	0.10 m
Maximum distance of wall faces	0.80 m
Maximum angle deviation of wall faces	5 degrees
Undershoot extension threshold	1.0 m
Minimum # of points in a segment	500 points
Max point distance for Conn. Comp. Analysis	0.10 m

Table 4.4. *Parameters for surface growing segmentation (Vosselman et al., 2004) and running time per dataset.*

Datasets	# points (million)	Segmentation time	Point-to-plane distance (meter)	Surface growing radius and # of neighbors in kd-tree
Fire brigade #1	7.4	648 s	0.10	1.0, 20
Fire brigade #2	5.6	426 s	0.08	0.8, 20
TU Delft	3.2	264 s	0.08	1.0, 20
Penthouse	2.5	204 s	0.04	0.5, 20

Runtime. All the algorithms are written in C++ and tested on a Lenovo ThinkPad workstation with an Intel core i7 (2.5 GHz, 16 GB RAM). Surface growing segmentation runtime is calculated per dataset and is reported in Table 4.4. The whole pipeline runtime, excluding segmentation, for a dataset with 7 million points and an average of 800 surfaces with 25 rooms takes 10 minutes. Most of the time is devoted to processing pairs of adjacent segments to build the adjacency graph, extending segments to their intersection, and eventually merging both faces. Other algorithms operated on minimum rectangles such as reconstructing the volumetric walls, generating the spaces from enclosed walls, and the algorithm for the detection of doors and stairs take less than a minute.

Limitations. Our volumetric reconstruction method for walls does not reconstruct columns in walls and walls with many intrusions and extrusions. Similarly, a wall with engraved windows such as façade walls are approximated by a planar surface and the details of windows frames are not reconstructed in the model. For the reconstruction of the permanent structures and to model the non-Manhattan World structures, our algorithm does not enforce any vertical, horizontal, or perpendicularity constraints. As a consequence, some walls can be slightly skewed when the surface is segmented with little clutter on the wall. However, this limitation does not jeopardize the space subdivision result and subsequent analyses. Some of the noise outside the building layout caused by the strong reflection of glass surfaces can disturb the detection of a permanent structure and lead to, for example, misclassified walls. However, the final 3D model is correct because these misclassified walls do not enclose a space during the reconstruction. Authors in (Nikoohemat et al., 2018a) propose a method to identify and to prune the noise of reflective surfaces as a solution, however, in this work, we did not apply their solution because the noise was not disrupting the final 3D model. The algorithm for stair modeling can fail if several steps are missing during the data acquisition. This can happen

because the BFS-search detects the longest path which is not representing the correct number of steps. Although, this problem can be disregarded if the data is scanned properly with a mobile laser scanner and the least amount of occlusion. Separating levels of the building using the trajectory is limited to mobile laser scanners which can go to the stair cases (backpacks and handheld). Therefore, for TLS and pushing-cart scanners the z-histogram can be used to separate the levels. Another limitation of using the trajectory is that in the presence of a lot of transparent surfaces can be problematic because laser rays penetrate to other levels, but yet it provides a coarse separation of complex buildings to the levels and stairs. Regarding the extraction of the room spaces, it is important to keep in mind that a lack of closure in the bounding elements of such spaces would not allow their reconstruction. While this can be seen as a limitation in highly occluded scenes, it remains a good indicator of the structural elements to reconsider for model correction or improvement.

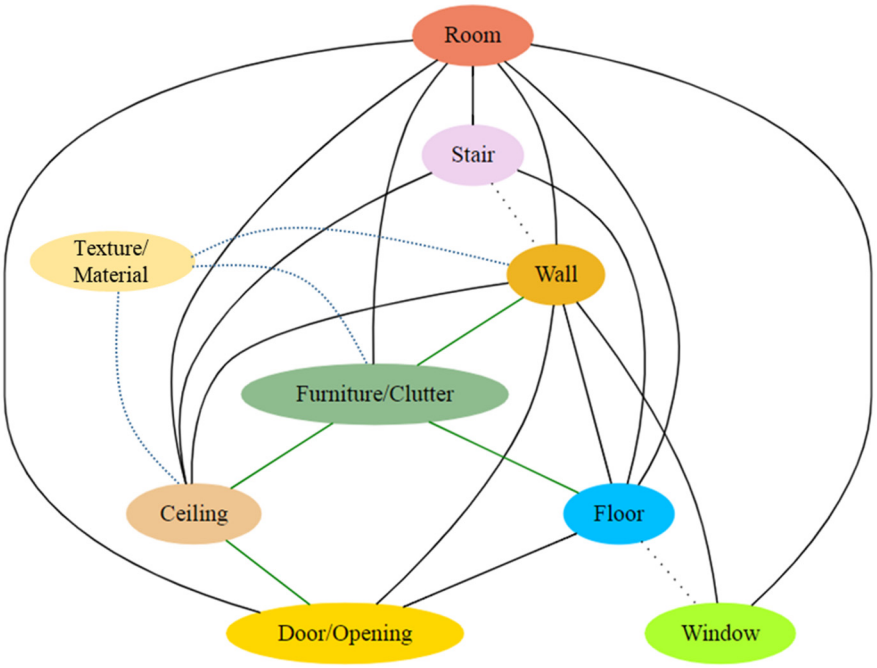
4.8 Conclusion and Future Work

In this chapter, we introduced a complete workflow that allows extracting fine-grained navigation networks from 3D models obtained through an advanced reconstruction process. The resulting model is meant to support disaster management and navigation in the context of emergency response. Since we generate volumetric walls and spaces, our results are also suitable to be used in BIM software for further improvements. Facility maps can be co-registered and included in our model for facility management. A safety audit can be performed without the need of sending inspectors to the building and this can be updated regularly for buildings, simply by scanning and generating new models. We use MLS devices for most of our datasets, which enable us to scan and generate the model on the same day. The generated model can be imported in CAD software and can be further improved by adding windows, missing doors, beams and columns. Firefighters can use the spaces to have an overview of the layout of different floors. When the spaces are enriched with the Object, Functional and Remaining (O, F, R) Spaces, a subspace-based navigation graph which reflects the indoor occupancy and functional spaces of indoor objects can be designed for supporting navigation of different types of agent. Our results show that 90% of the rooms are reconstructed correctly and this can be improved with a simple user intervention to 95%. We demonstrate a fully 3D reconstruction pipeline for multistory buildings with slanted walls, ceilings and ramps. The reconstruction of volumetric walls enables us to have a better geometry model close to IFC standards rather than thin-wall 3D models. Therefore, our pipeline can be an initiative for scan-to-BIM problems. Applying the closure-based method to extract spaces does not require the position of a scanner or a 3D cell complex partitioning. Moreover, the FSS approach provides a higher level of details for space subdivision comparing to previous 3D models. The consistency control constraints make sure that the

generated spaces are validated for navigation, and rooms are accessible from different locations in the building.

Future work could include the automatic semantic identification and enrichment of the furniture. This would help in the creation of more advanced FSS models with more detailed functional spaces and would also allow automatic identification of the room function. The evaluation of the whole pipeline with a flexible navigation network for emergencies in a real case scenario is not a trivial task and needs cooperation of several stakeholders and is part of our future plan. Moreover, automatic reconstruction of other items such as windows, beams and columns are another important improvement that needs to be considered in our 3D models. Although we have introduced a basic process to check the consistency, more in-depth investigations for further constraints checking could improve the pipeline by additionally considering, for example, topological constraints on the 3D models reconstructed from the point clouds.

Chapter 5 - Consistency Control of Indoor 3D Models Using a Control Grammar



Abstract

Indoor 3D models are a digital presentation of building interiors. Such 3D models are mainly reconstructed from 3D scans acquired by laser scanners and RGBD cameras. Because of the variety of algorithms in automating the reconstruction of 3D models, it is difficult to validate the correctness of final model and its suitability with the application. On the other hand, there are standards such as BIM, IFC and ISO 19107 which can guide the engineers when modeling the buildings. However, there is not any consistent approach which in the lack of ground truth to apply automatically on a certain model and to verify its consistency. We propose a conceptual framework based on formal grammars to control the consistency of a 3D model (generated from scans) in terms of semantic, geometry and topology. The proposed solution starts with the decomposition of the model to its composing components and takes three steps to validate the model: 1. Checking the correctness of individual instances in the model (e.g., a wall object), 2. verifying the consistency of the instances in interaction with each other (e.g., a door on a wall) and 3. the consistency of the model for the application (e.g., navigation). Our method does not fix the problem but by using a control grammar (in)validates the components in the model given the rules form the current standards and expert knowledge. A 3D model is consistent if it passes all three steps. Otherwise, in each step it is rejected with the flagged components. This proposed framework acts as a compiler for validation of 3D models regardless the level of details and reconstruction method.

5.1 Introduction

In recent years there has been an impressive progress in indoor data collection technologies such as mobile laser scanners and RGBD scanners. Such sensor systems that provide high quality images, point cloud and depth information enable us to reconstruct and update indoor 3D models through semi-automatic and automatic methods. These models are digital representation of the buildings interiors and are useful for architectures and engineers in planning and construction. Indoor 3D models are reconstructed from different sources such as floor plans, images, LIDAR and RGB-Depth data using automatic approaches. Exploiting multiple sources and automatic approaches make such models inconsistent in terms of current standards. Although, existing standards and data models such as industrial Foundation Classes (IFC), ISO 16739 and ISO 19107, CityGML and IndoorGML (Gröger et al., 2012; ISO, 2019, 2018; Lee et al., 2014) provide a suitable guideline to generate 3D models, the complex structure of indoor environment and noisy data obstruct the creation of a consistent 3D model in an automatic workflow. The question is how we can have a systemic way to check the consistency of our model without the presence of ground truth and with the minimal interaction of humans.

The generated model should comply with standards of current indoor models (IndoorGML, CityGML, IFC) considering the geometry, topology and data models of entities to provide a reliable platform for applications such as evacuation and safety management in large buildings. Obviously, some of the consistency checks could be inherited from standards but sometimes the generated 3D models do not follow any specific standard but yet they are giving a thorough overview of the building and interiors (Ikehata et al., 2015; Xiao and Furukawa, 2014). However, a correct and consistent indoor 3D model can be used for many applications. Indoor 3D models can differ regarding the applications or the standards which they comply with. For example, for navigation applications and considering IndoorGML standards, reconstructing indoor spaces and doors should be suffice and reconstruction of walls is not necessary. On the other hand, according to the BIM models, walls are important components of a 3D model and should therefore be reconstructed in volumetric shapes. Similarly, when considering the energy consumption of buildings for heating purposes a model with the presence of windows and air outlets is necessary.

A 3D model can have different level of complexity regarding the application, the reconstruction method and the input data. Using a 3D model for 3D GIS or BIM applications can lead to different representation of the geometry primitives. Integration of models between CAD and GIS has been studied in the literature (de Laat and van Berlo, 2011; Pu and Zlatanova, 2006). Similarly, standards (e.g., IFC, CityGML and IndoorGML) provide a guideline

that how a 3D model should be in terms of geometry and semantic for more specific applications such as construction or navigation. However, it is not trivial to check the consistency of a complex 3D model against a specific standard because all the components should be present in the model otherwise the comparison is rejected at the first place. For example, a 3D model which represents walls as surfaces and does not provide spaces is not compatible with both IFC and IndoorGML, yet could be a consistent model for specific applications. Therefore, we need a system which checks 3D models against some common knowledge present in the building language as well as existing standards. Such a system should not be sensitive to the type of geometry and should be scalable and adaptable to different applications. The tool that we choose for this purpose is a type of formal grammar where the components of a 3D model compose the symbols in the grammar, we name it the *control grammar*, and the constraint rules are controlling the consistency of the model.

Previous works verify a model by comparing them against ground truth either with a point-to-point comparison (Lehtola et al., 2017) or plane-to-plane comparison (Tran et al., 2019). The consistency of an indoor 3D model consists of completeness and correctness and should be checked in terms of topology, geometry and semantics even there is no ground truth. While completeness of the model is more dependent on external resources for example knowing the correct number of rooms, the correctness of components can be controlled individually and in relation with each other. From mathematical point of view geometries and their topology can be validated. Staircases and floors, for instance, are *disjoint* or *connected*. Each floor is *coveredBy* rooms and each room has at least one door. For example, if the accessibility of rooms and interiors to an (emergency) exit is defected because of topological errors in the model then the indoor navigation for evacuation fails. The control grammar does not correct or change the components but identifies, formalize and (in-) validate the inconsistency of components in relation with each other and their semantic. A control grammar acts as a compiler for 3D models and after parsing the 3D model to components analyses the correctness of them in three phases: individually(instances), in interaction with each other, and in application. Incorrect components are flagged and rejected to previous steps (e.g., semantic labeling or geometric modeling) for further check. The correction of inconsistent items is based on the judgment of the modeler and can be corrected manually or automatically. However, our system allows some of the constraints be passed if they are not crucial for the integrity of the 3D model. For example, if a model allows gap between the rooms because the walls are not modeled, then this can be passed to the system and system continues with further checks.

In this chapter, we introduce a control grammar which allows one to use different geometric representation and source of data for 3D model

reconstruction and yet be able to identify, formalize and verify the consistency of the model. Our formalization is independent on standard specifications and can be extended into a variety of applications.

The following of this chapter is as follows: section 5.2 gives an overview of the current methods for indoor 3D modeling from the literature and compare some of the methods. Section 5.3 explains the methodology of our work and section 5.4 gives some examples to illustrate the problem and solution. Section 5.5 is conclusion and future work.

5.2 Scientific background

Indoor models have been investigated from two main perspectives: indoor model reconstruction and indoor model applications. While the former studies the methods for building 3D models (façade and indoor) from the data such as RGBD, LiDAR and imagery, the latter tries to improve indoor model standards such as Industry Foundation Classes (IFC), CityGML (LoD4, LoD3) and IndoorGML. However, one important aspect of indoor 3D models is their consistency which is not addressed sufficiently in the literature. Gröger and Plümer, (2009) study how to preserve the geometric-topological consistency of 3D city models. Some of their solutions on primitive shapes geometry and topology can be applied for indoor objects but it is not suitable for complex indoor scenes. Horna et al., (2009) use Generalized Maps (G-map) as a topological basis to define a set of constraints for keeping the consistency of 3D models generated from 2D floor plans. Their solution is interesting in terms of applying the constraints to keep the production process generic to various type of buildings but it is only usable for models generated from floor plans. Gröger and Plümer, (2010) introduce a *Constraints Store* for derivation of 3D indoor models by grammars. The constraint store prevents errors to contradict with consistency-reachability rules. For instance, by using the grammar, a new split face should not traverse a door inserted with previous rule. These constraints are generated by rule applications and explicitly formalize the concepts of adjacency, reachability and semantics. Their proposed constraint store maintains three main rules: 1. Equality Constraints that means two boxes share the same face or wall. 2. Aggregation Constraints means face F is the aggregation of face F1 and F2 where F is split with a face or wall. 3. Inside Constraint that means face F1 is inside face F2, for instance a door inside a wall. Their solution is a good example of using grammar to keep the consistency of a model during the production process. However, the proposed approach has no example of a real case study or a data-driven model and applying their methods on non-Manhattan World buildings can be challenging because of the box representation for rooms.

In using grammar approaches for 3D modeling, Becker et al. (2015) use an L-System combined with a shape grammar to reconstruct interiors from point clouds. The authors define interior structure as two main subdivisions: rooms and hallways. Per each structure, they apply a specific grammar to reconstruct the interior environment. Another example of using grammar is given by Ikehata et al. (2015). They encode indoor space first into a structure graph and then transform it to a structure grammar. Their work is a practical representation of inverse procedural modeling for indoor environments but limited to Manhattan-World structures. Split grammar is the famous method proposed in instant architecture by Wonka et al., (2003) to automate the process of urban design for a variety of building styles. The authors claim that the power of split grammar lies in the restriction of allowed rules that allows a control on the derivation process while keeping it simple. Similarly, procedural modeling is used by several researchers for façade modeling and indoor modeling but in most cases they produce synthetic models for virtual cities and gaming applications (Aliaga et al., 2016; Mathias et al., 2011; Müller et al., 2006). There are some recent examples for indoor modeling and using grammar which result in more complex models from point clouds (Dehbi et al., 2016; Tran H. et al., 2019) or inferring the semantics of rooms (Hu et al., 2019), but none of these methods offer a complementary solution to check the correctness of the final model. Figure 5.1 illustrates the evolution of grammar applications from the early stage to this time. The figure shows that shape grammar has increasingly been used in building architecture and modeling. Therefore, we decided to apply grammar for consistency control of the 3D models. Note that our suggested grammar, unlike the mentioned grammars, does not apply production rules to create a 3D model, but it uses rules to verify the components of a model.

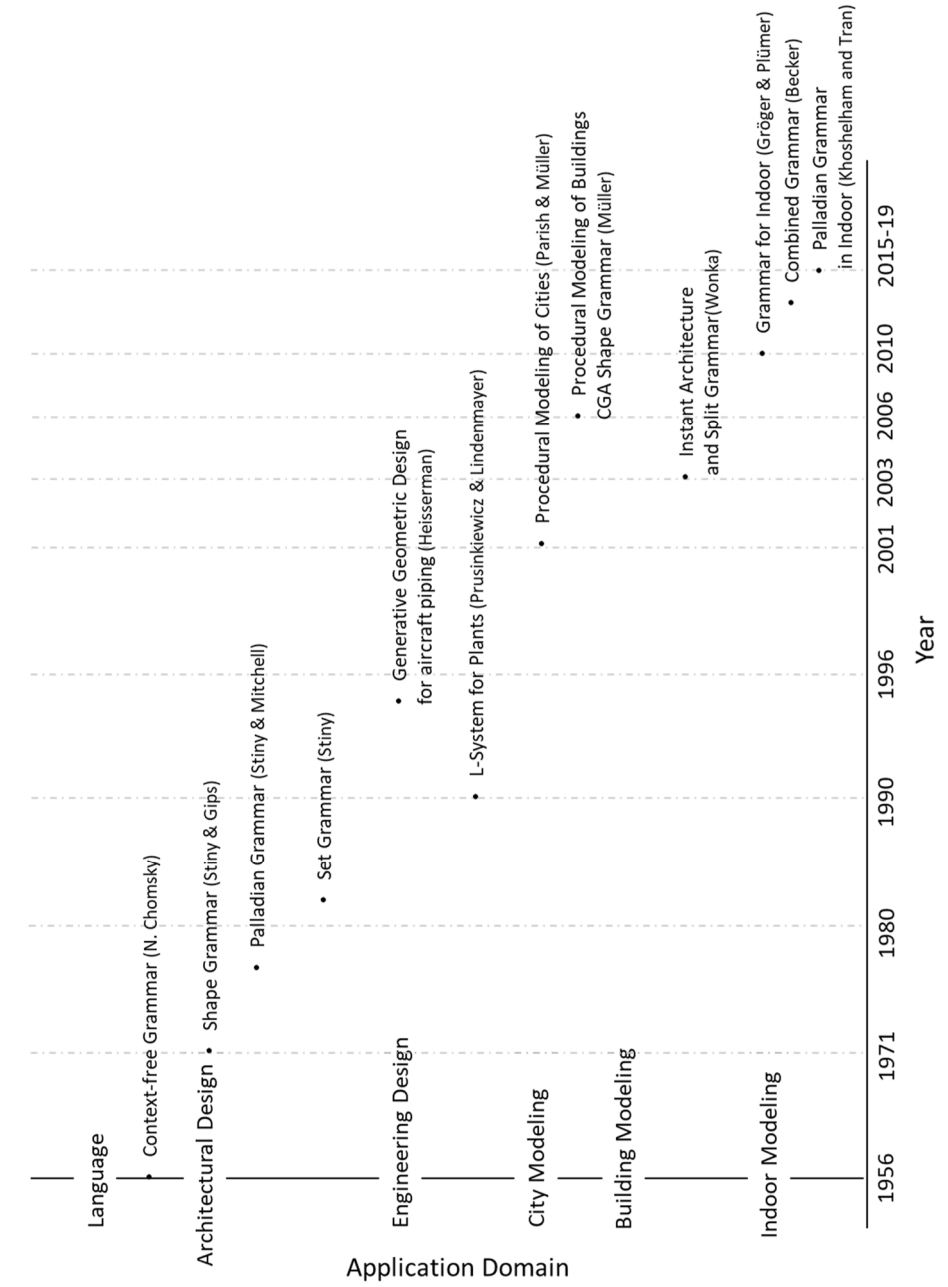


Figure 5.1. *The figure illustrates the application domain of grammar since the 1950's. The evolution of grammar births with language grammar and continues in design and building architecture as well as computer graphic. The application in indoor modeling domain has started since 2006 which expresses the novelty of research.*

Apart from grammar approaches for indoor 3D modeling, there are other methods such as piecewise planar reconstruction (Mura et al., 2016), mesh based methods (Xiao and Furukawa, 2014) and binary space partitioning methods (Ochmann et al., 2016). Each of the methods result in different types of models and levels of details. We classify these models into three main categories based on the type of and the existence of the walls: 1. Models with volumetric walls, 2. Models with paper-thin walls or surface walls, and 3. Models with no wall and just room representation. Table 5.1 shows the present layers in each model from the literature and Figure 5.2 categorizes them based on the type of the wall. Also, we included IndoorGML and IFC in Table 5.1, to compare their differences.

Another important aspect when studying the indoor 3D models is the data stream from which the model is created. Most of the models presented in this related work and those which are the focus of this research are created from point clouds and RGBD images (Furukawa et al., 2009; Mura et al., 2016; Nikoohemat et al., 2019; Ochmann and Klein, 2019). However, our suggested solution can be useful as well for models generated from inverse procedural modeling methods and from floor plans (Horna et al., 2007; Okorn et al., 2010). As long as the 3D model can be decomposed to its reconstructing components and can be parsed into our grammar, it can be used in our framework.

Table 5.1. *Comparison of existing layers in 3D models in the literature and the difference in modeling of walls. Note that this comparison is based on the authors' interpretation of the models. The algorithms in the original literature may produce more details.*

	Wall	**Floor/Ceiling**	**Door**	**Window**	**Stairs**	**Rooms**
IndoorGML	no wall	yes	yes	no	yes	yes
IFC	volumetric	yes	yes	yes	yes	yes
Budroni And Boehm (2010)	paper-thin	yes	no	no	no	no
Xiong et al (2013)	paper-thin	yes	yes	yes	no	no
Mura et al (2014)	paper-thin	yes	no	no	no	yes
Oesau et al (2014)	paper-thin	yes	no	no	no	no
Xiao and Furukawa (2014)	volumetric	yes	no	no	no	no
Turner et al (2015)	no wall	yes	no	no	no	yes
Ikehata et al (2015)	paper-thin	yes	yes	no	no	yes
Becker et al (2015)	paper-thin	yes	no	no	no	no
Mura et al (2016)	no wall	yes	no	no	no	yes
Ochmann et al (2016)	volumetric	yes	yes	yes	no	yes
Ambrus et al (2017)	no wall	yes	yes	no	no	yes
Macher et al (2017)	volumetric	yes	no	no	no	no
Murali et al (2017)	paper-thin	yes	yes	no	no	yes
Wang et al (2017)	paper-thin	yes	no	no	no	yes
Wang et al (2018)	paper-thin	yes	yes	yes	no	no
Nikoohemat et al (2018)	no wall	yes	yes	no	no	yes
Ochmann et al (2019)	volumetric	yes	no	no	no	no
Tran et al (2019)	paper-thin	yes	no	no	no	yes
Nikoohemat et al (2019)	volumetric	yes	yes	no	yes	Yes

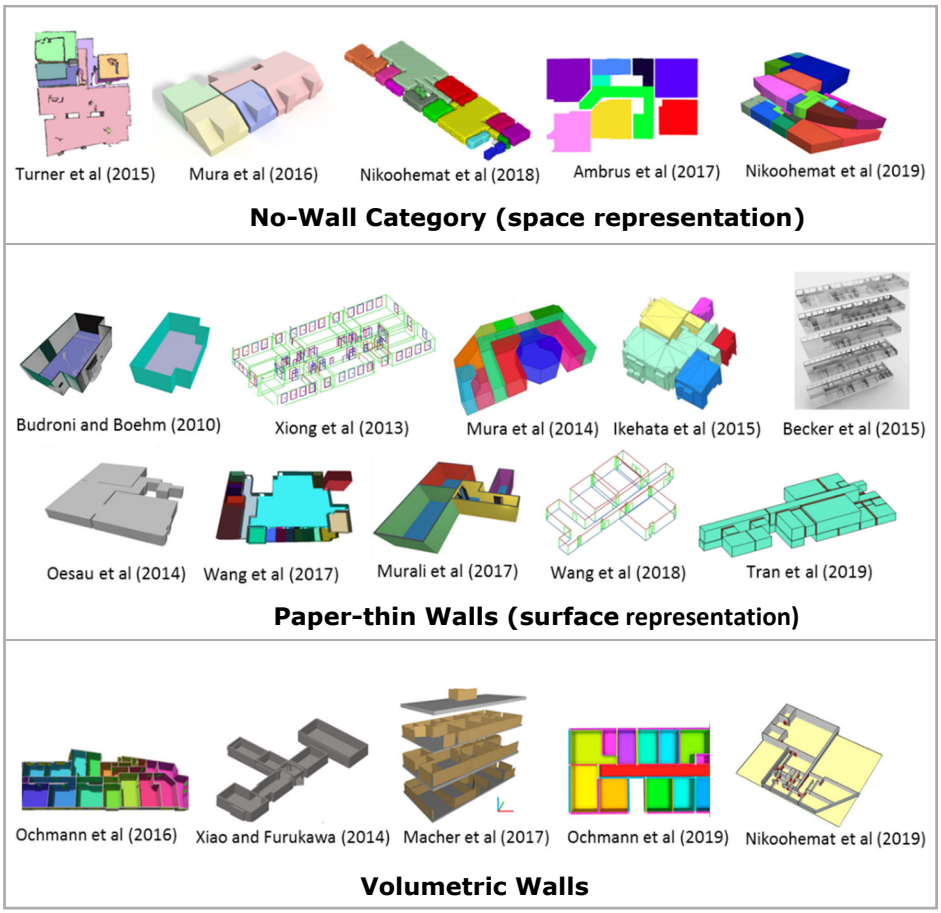


Figure 5.2. *Comparison of different 3D models in the literature. Indoor 3D models are categorized in no-wall, paper-thin wall and volumetric wall. This category is based on the author's interpretation of the final models and is just an example for comparison of the models. The algorithms in the original literature may produce more details.*

5.3 Methodology

Given a 3D model, the goal is to check the consistency of the model. A 3D model is a digital representation of the building interiors created from point clouds, RGBD images or floor plans. The consistency aims at checking the correctness and completeness of the model in the lack of ground truth by relying on the expert knowledge and existing standards.

First the model should be stored in a (context free) grammar structure. Here we explain several keywords. For clarification purpose a *class* is referred to as any component in the 3D model either abstract or real such as a permanent structure (a real component), or a space (an abstract component). An *instance* refers to an object of a given class. For example, each room is an instance of

the space class or each piece of furniture is an instance of furniture class. *Symbols* are terminals, non-terminals and the axiom in the grammar. *Constraints* are rules in the grammar. A 3D model can be represented as a vector model (mesh, polyhedron, etc.) or a raster (e.g., voxel). It can be acquired from digitizing floor plans and converting them to a 3D model, or from depth images or from Lidar scans. A 3D model should at least show the layout of spaces and interiors. According to Table 1, each model may have one or some of the below classes. In total, *eight classes* are considered in our grammar for the consistency control.

1. Permanent structure classes including walls, floor and ceiling are three classes.
2. Openings including doors and windows are considered as two classes.
3. Furniture, obstacles and objects are considered as one class.
4. Spaces or rooms are considered as one class.
5. Stairs or staircase are one class.

Note that rooms or spaces are the only class which can be *abstract components* and can be represented by the surrounding structure elements such as walls. Real components can be represented by geometric objects such as solids, surfaces and boundary representation. In the following, first we explain the components of control grammar and then the method for applying the grammar on a 3D model.

5.3.1 Control Grammar

Our grammar is a four-tuple *G= (N, T, R, S)* including set of non-terminals (N), set of terminals (T), set of constraint rules (C ϵ R) and a start symbol (S). Each rule is the form of $a \rightarrow b$ where (a, b) ϵ (N, T). Terminals and non-terminals are geometric primitives or geometric complex. For example, a wall can be represented by a solid (GM_Solid, ISO 19107 Standard for Geographic Information) or a surface. Rules are extracted from standards such as ISO 19107, IFC and IndoorGML and applied as constraints meaning they are not production rules. Each rule only (in)validates the objects in the model against a given constraint. Some of the rules have preconditions meaning a rule cannot be applied if the precondition is not satisfied. The constraints control the consistency of the 3D model in terms of topology and the integrity of the model in terms of interaction between instances.

A Manhattan Free Grammar: since most of the models in the recent literature are non-Manhattan and could have arbitrary wall layout, hence, our grammar does not enforce orthogonality and parallelism constraints to any of the geometries. A wall can be slanted and have arbitrary intersection angle with adjacent structures. However, one can include these constraints for the

aesthetic aspect of the model. For example, it can be included in the grammar that walls should be perpendicular to the building façade and to the floor.

5.3.2 Applying the grammar on a 3D model

3D models can be acquired from different sources and by using different methods. First, we explain a common approach for creating 3D models from point clouds. Having a dataset of point clouds from interiors, most of the algorithms extract permanent structures (walls, floors and ceilings) using methods such as cell complex, adjacency graph, piece-wise planar reconstruction followed by a room segmentation approach to create a 3D model. Some related works take it further and add more details such as openings, windows and stairs. Figure 5.3 shows a general workflow of extracting geometric primitives from point clouds and then adding semantics to provide a coarse 3D model. Our control grammar comes to the play at this stage and checks the components of the model concerning their geometry, topology and semantics. The assumption of our work is that we do not have external sources or ground truth for the consistency check and it is based on the common knowledge, existing standards and self-control. Any component which is rejected should be returned to an early stage of reconstruction (e.g., geometric derivation) and can be checked manually or if it is possible automatically for corrections. The process of correction is not part of the control grammar and it is related to the reconstruction algorithm. The objective of a control grammar is identifying inconsistency in the model.

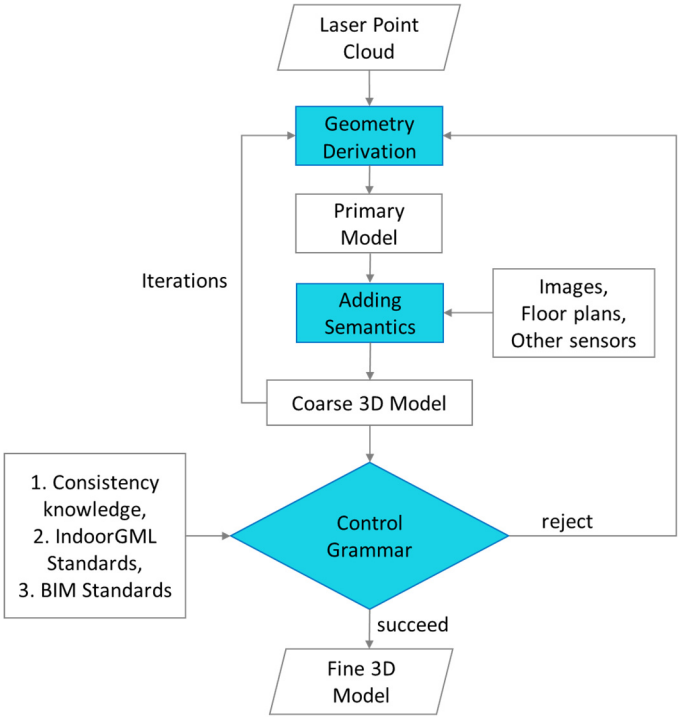


Figure 5.3. *Flowchart for proposed methodology. Colorful shapes represent main steps in the creation of a consistent 3D model.*

The rules check the validity of the component individually and in relation with other components in terms of geometry, topology and semantics. Before applying the rules, a 3D model parser decomposes the model into composing components where they are stored in the grammar (Non)Terminals. Then rules are applied on non-Terminals as instances or a pair of interacting instances. When an instance is not valid it is flagged as invalid and will not be checked further until the issue is resolved and it is returned to the geometric derivation step in the flowchart (Figure 5.3). However, some of the checks can be flagged as valid if the user (operator of the system) decides to pass the instance. We consider such rules as softer rules that can be relaxed to allow a certain amount of satisfaction, e.g., a door can be disjoint of the floor up to a threshold. This serves the flexibility of the system in terms of applications and the purpose of the modeling.

The control grammar is applied in three main steps (Figure 5.4):

1. Instances: meaning single components in the 3D model are controlled independently.
2. Interactions: proximate components which interact with each other are controlled.

3. Application: components are controlled based on the application of the model.

Each of above steps enforce some geometric, topological and semantic constraints inferred from the standards or which are defined by the experts. In the next subsection we explain how starting from a 3D model each of the steps can be applied.

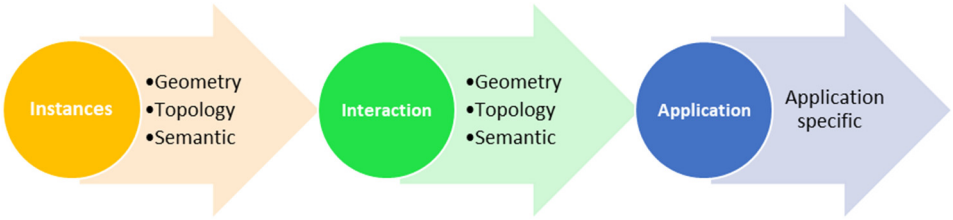


Figure 5.4. *Three stages of consistency control. In first two stages, three characters of instances is checked individually and in interaction with each other. In the third stage, instances are checked considering the application.*

5.3.3 A 3D-model parser

The control grammar takes the 3D model as input and decomposes it to the constituting classes. Each class has instances and each instance has attributes such as bounding box, local coordinate, color and so forth. This is the task of parser to recognize each instance in a class and store it as a non-Terminal symbol with the class name as the attribute. Note that the parser does not need to infer the type of class for each instance and this information should be provided in the model. For example, walls and doors have different labels in the original model. Some of the attributes such as volume can be calculated from the solids and stored with the instance in the non-terminal set. Other attributes should be provided with the geometry or if necessary, filled manually in the related fields, for example, the correct number of rooms in the model, if is known, or the number of steps in stairs. Attributes include but not limited to volume, area, width, height, length, color, location, function (kitchen, corridor, …), type (abstract, real), scope (permanent, temporary), number of levels, number of steps, number of rooms and materials. During parsing, the user can add more semantic to the component which would be useful for control grammar, for example separating façade walls from internal walls which can define the external shell of the building. Each instance per class is a geometric primitive or a geometric complex and is stored in a set of non-Terminals (N), where it is selected and checked with the constrained rules in the grammar.

5.3.4 Consistency Control of Instances

Since non-terminal instances correspond to 3D geometric entities, their validity should be ensured independently of each other. This can be done from a geometric, topological or semantic perspective. Geometric and topological validation of 3D objects is a well explored issue in the literature. This knowledge served as support to standards commonly used in GIS such as the ISO 19107. The latter specifies conceptual schemas for spatial description and consistent operations for (up to 3D) vector geometry and topology of geographic features. In this work, we will rely on the specifications of the ISO 19107 to define valid spatial entities (similarly to IFC and CityGML). A comprehensive study on the validation of solids is also presented in (Ledoux, 2013).

Based on our definition of 3D models, three types of geometries can be expected: (i) surface, (ii) volume (solid) and (iii) complex. At the beginning of the instance check process, every instance is initially assumed to be invalid. Hence, independently of its semantic nature, checking the geometric and the topological consistency of a non-terminal instance consists in applying several geometrical predicates (such as orientation or closure tests, etc.) that should result into Boolean results (true or false). In the case of surface elements like polygons, examples of predicates include simplicity check (the boundary of the polygon should not self-intersect or contain duplicated vertices), or degeneracy check (surface area should not be null), etc. In the case of volume elements like polyhedron, they should match the definition of a solid (Gröger and Plümer, 2009; Ledoux, 2013).This includes conditions such as their boundaries should be composed of valid polygons and they should form a full closure of the space. Checks that applies to polygons may also apply to polyhedron, with one more dimension to be considered (e.g., degeneracy test of a volume is to test whether it is a flat volume, with a magnitude null). Criteria that define valid 3D polygons and polyhedra can be found in the standards (ISO 19107, OGC, Ledoux 2013, etc.). Every check that results into a positive result would change the validity state of a non-terminal to true. Otherwise, the invalid instance is flagged and may be considered for a correction process. Complex objects consisting of several primitives, such as an L-shaped wall or stairs, need to be decomposed to simpler geometries in order to validate them.

Finally, the semantic control of an instance includes validating the instance knowing its semantic, class or function. This is rather more difficult than the two other checks on geometry and topology because there are less semantic standards for objects. By referring to IFC standards (e.g., IFCWall or IFCWindow), it is possible to infer some of the rules for the instances. Similarly, by using IndoorGML standards, we can check the sanity of indoor spaces and rooms (Cellular space in IndoorGML). Examples such as a wall is a vertical

element which bounds rooms and subdivides the space can be interpreted from the IFC standards. Likewise, a room should not be smaller than a certain volume or height are types of rules which can be inferred for semantic control of the instance.

The instance control grammar is summarized in Algorithm 1. Note that, Non-Terminals (N) are primitives containing the class type and extra attributes, such as bounding box, color, and height, which can be obtained with getters functions (lines 7 and 8 in the algorithm). The case of a rule can be defined as *r ∈ R, X → valid | invalid <C> {E}* where the C *∈ R* applies the constraints and E returns the type of error in case of invalid results. Rules (R) are a collection of Rs as semantic rules and Rg.t as geometric and topological rules. Note that the selection of a rule from the set of rules is decided based on the geometry and the class of the instances. The result of this step is a set of invalid instances with the type of error which needs to be resolved by the expert decision before proceeding to the next step (interaction control).

Algorithm 1 Instance grammar

$R_s \leftarrow \{semantic\ rules \in R\}$
$R_{g.t} \leftarrow \{geometry\ \&\ topology\ rules \in R\}$
$I \leftarrow \{set\ of\ instances \in (N \cup T)\}$ ▷ N=nonTerminals, T=Terminals
$I_{isValid} \leftarrow false$
$E_{map} \leftarrow \{map\ of\ errors\ and\ instances = \emptyset\}$
for each instance $I_i \in I$ **do**
  $I_{i.class} \leftarrow getClassType(I_i,\ CTypes)$
  $I_{i.geometry} \leftarrow getGeometryType(I_i,\ GTypes)$
  $I'_i \leftarrow \mathrm{I}_{i.class},\ I_{i.geometry}$
  $I'_i \leftarrow applyRule\ r \in (R_s \cup R_{g.t})$ ▷ Applying rules
  if $I'_i \neq \emptyset$ **then**
    $I'_{i.isValid} \leftarrow true$
  else
    $E_{map} \leftarrow insert(error,\ I'_i)$
  end if
end for
return E_{map}

5.3.5 Consistency Control of Interactions

The non-terminals from the previous step are validated individually considering their geometric, topological, and semantic information. Here, the grammar controls two instances either from the same class or from different classes in interaction with each other. Although, from a general perspective, all instance interactions correspond to spatial relationships, thus topological relations, the difference of geometry, topology, and semantics of the instances lead to more

specific types of interactions. For example, when checking the intersection between two volumes, the geometric and topological information prevail in the interaction. On the other hand, a semantic check is related to the interaction of instances considering their class type. For example, a furniture cannot be dangled in the model without attachment to a room or a permanent structure. Several topological models allow to extensively describe the spatial relationships linking the instances. Here we rely on the 9-intersection model (Egenhofer and Herring, 1990; Zlatanova et al. 2004) (see Figure 5.5) in our grammar to describe the valid and invalid interactions. Valid interactions include:

$$\{disjoint,\ meet,\ coveredBy\} \in R\langle C\rangle$$

Where R are rules defined by constraints C. The remaining relationships will be considered as invalid for interaction of two objects. For example, a room is coveredBy a furniture but a wall is met by a door. The meet relation happens when the boundaries of two 3D objects meet but not their interiors (Zlatanova et al., 2004).

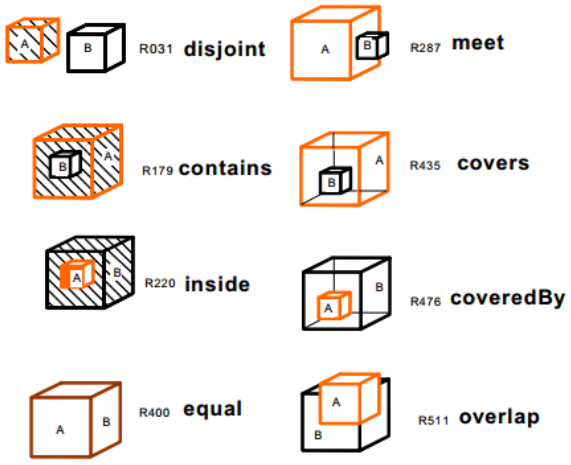


Figure 5.5. *9-intersection model for controlling the interaction of classes and possible relationship between 3D objects (Egenhofer and Herring, 1990; Zlatanova et al., 2004).*

A handy way to map the instance interactions is to use an interaction matrix. Table 5.2 illustrates one that lists few selected classes and shows their interactions, considering the information of their individual instances. Each row *i* and column *j* of the matrix refers to an instance in a specific class and each cell C_{ij} correspond to the interaction rule of an instance I_i with respect to an instance I_j. While an instance cannot be compared with itself, the instances of the same class can interact with each other. Therefore, the diagonal cells on the matrix are not empty.

The colors used in Table 5.2 allow to determine the level of interaction that can be expected between two instances. A green cell C_{ij} means that I_i must interact with I_j. This is the highest level of interaction constraint that makes the instance I_i invalid if it is not fulfilled. As an example, a wall instance is necessarily expected to touch other wall and floor instances. This corresponds to the *meet* relationship in the 9-intersection model. Similarly, the red cells represent interactions that should not happen, or said differently, *disjoint* relations. Otherwise the instances are flagged, as invalid. Yellow cells describe the most flexible constraint as the interaction of the two instances can eventually happen if some conditions are satisfied or may simply not happen. In both cases, that does not mean there is a problem, unless the identified relationship is an invalid one (e.g., *overlap*). For example, in some architectural designs a door and a window can *meet*. Therefore, yellow cells in the matrix can change to either green or red based on the expert decision and the building architecture.

The interaction matrix is translated to a graph where instances are the nodes and their interactions are edges. Nodes are connected by solid lines representing the green cells and by dashed lines representing the yellow cells. Clearly, for red cells no edges exist. Figure 5.6 shows how the colors are translated to the edge type in the graph. Green and yellow cells represent that two instances are adjacent and the line is a dashed line if the connection is conditional. We included three higher level of interactions in the grammar rules as a subset of *meet* in the 9-intersection model:

$$\{support, supported, attachment\} \in <meet>$$

A support relation happens when one object is not only adjacent and meet another object but also supports the object. A supported object always needs the adjacency of its supporter. For example, a lamp is always supported by a wall or ceiling or another furniture. An attachment happens when a room is an abstract component in the model and is represented with surrounding instances such as walls. For example, furniture always should be attached to a room. Similarly, for adding material and texture, there is no spatial relation and the connection is of the type *attachment*.

Figure 5.7 shows the translation of interaction matrix (Table 5.2) to the graph. This graph is an abstraction of a 3D model. Such a representation of classes and their instances in the form of a graph is scalable because the experts can add more nodes and can define the connections, such as adding furniture types or beams. A real graph which exactly reflects all the components in a 3D model looks much more complex, as each node represents an instance. Thus, for example, for each wall or door object there should be a separate node in the model. In the experiments section, an example of such a graph is shown.

Table 5.2. *The interaction relation between instances of each class. Interactions can occur for adjacent objects.*

Interaction matrix	wall	floor	ceiling	door	win.	stairs	room	furnit.
wall								
floor								
ceiling								
Door/opening								
window								
stairs								
room								
furniture/ clutter*								

* At least one of the yellow interactions should be met. This means a piece of furniture at least should have a connection (support or supported by) to one of the permanent structures or to another furniture.

graph edge		**type of adjacency**	**color**	**description**
solid line	————	Must be connected.		adjacent = share a common face or an edge
dashed line	– – – – – -	Conditional connection		conditional = for example through an opening
no line	-	No connection		not-adjacent

Figure 5.6. *the description of cells in Table 5.2.*

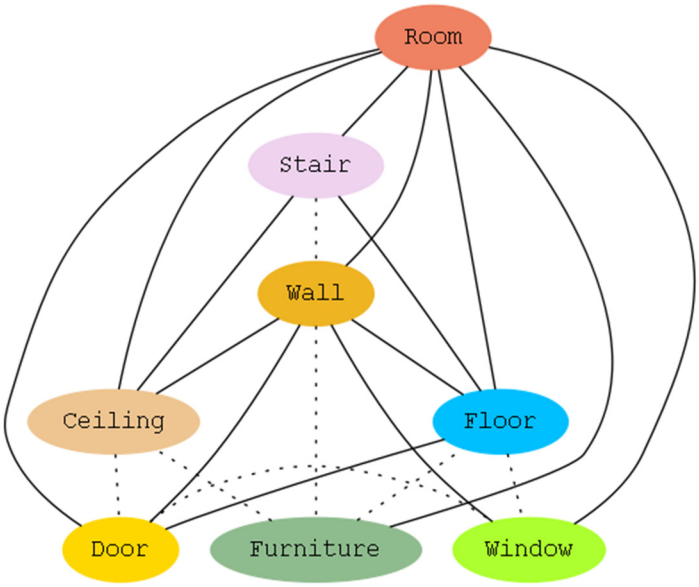


Figure 5.7. *The adjacency graph of classes. Some of the relations in the Table 5.2 are not shown for the simplicity of the graph. According to Table 5.2, almost all instances can be connected to the room. Edges with solid lines imply that two instances must be connected and dashed edges show the optional connection between two instances.*

The interaction control grammar is summarized in Algorithm 2. The algorithm traverses all the edges, and for each edge (e) the nodes (u, v) are selected. Similar to instance grammar, the class type and geometric type of the nodes (instances) are derived from the attributes and the corresponding cell in the interaction matrix can be identified. When the interaction type is recognized (green, yellow and condition), the corresponding rule(s) can be applied. Note that the *applyRule* function (line 7) in the algorithm is not limited to one rule but can sequentially check several constraints for the connection of the nodes. For example, a door and wall connection is verified in terms of geometric, topological, and semantic constraints. The output of the interaction control grammar is a set of errors reflecting the edge (and corresponding nodes) and type of errors. This step is passed if the error set is empty, meaning all the edges and their corresponding nodes are valid.

Algorithm 2 Interaction grammar

```
    R ← {interaction rules ∈ R}
 2: E_map ← {map of errors and edges = ∅}
    for each edge ∈ Edges do
 4:     edge ← getNodes(edge, AdjacencyGraph)
        u ← getNodeType(CTypes, GTypes)
 6:     v ← getNodeType(CTypes, GTypes)
        edge ← applyRule(u, v, R)                ▷ Applying rules
 8:     if edge ≠ ∅ then
            edge_isValid ← true
10:     else
            E_map ← insert(error, edge)
12:     end if
    end for
14: return E_map
```

5.3.6 Consistency Control of Applications

The last step of our pipeline consists of controlling the consistency of the model with respect to criteria specific to given applications. While the two previous steps can be considered as generic constraints commonly accepted for 3D models, the rules in this part of the grammar are more specialized to application domains. Consequently, a 3D model invalidated by this step for one application does not necessarily imply its invalidity for another application, and, vice-versa, the same applies to validated models. As an application example, we choose indoor navigation and define corresponding rules. We control the consistency of our generated model against a set of defined heuristic constraints (C_i), targeted to specific instances or combinations of instances critical to indoor navigation (e.g., detected doors, stairs and reconstructed spaces). The correctness of the model regarding those instances can be improved by this consistency check. The constraints, which are checked on the data sequentially, are described as follows:

C1. Each room should have spatial extents L, W and H greater than a given threshold.
C2. Each room should be connected to at least one door.
C3. There should be a route connecting every pair of rooms in the graph.
C4. Two floors are connected via at least one staircase
C5. Each space should have access to the exit door
C6. Emergency doors should not be blocked.
Optional constraints:

C7. Angular arrangements (e.g., all the corridors have the same turning angle)
C8. Wall thickness operations (e.g., all interior walls have the same thickness)
C9. Aesthetic operations (e.g., all doors in corridors have the same type)

The verification of C1 is necessary to make sure that rooms are spacious enough for navigating agents to fit them in. In fact, due to different methods possible for indoor space reconstruction, it can happen that reconstructed volumes are too narrow to be even considered as a space. Thus, this constraint helps detecting such features and guaranteeing that only reasonable paths will be delivered in the navigation process. While their threshold values can be determined based on input information of the navigation or generic assumptions (e.g., average human size), the estimation of the spatial extents, namely the length (*L*), width (*W*) and height (*H*) is not always trivial, in particular in non-Manhattan configurations, but extent checking on bounding boxes of rooms may often be enough. Once a room is validated, it passes for the next check. Otherwise, it should be flagged for further control. After correction, the pipeline is repeated to create a consistent model (see Figure 5.3).

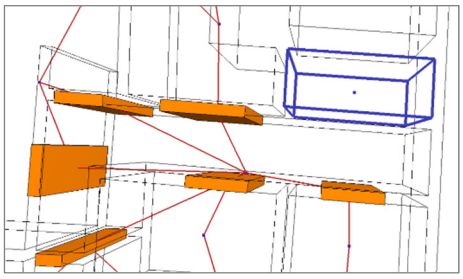


Figure 5.8. *Illustration of the constraint C2. A room (emphasized in bold blue lines) for which the node is connected to no other node of the graph characterize a missing door.*

The constraint C2 checks if each room is connected to at least one door. Otherwise, a missing door is flagged for the room. Door detection is a challenging process in 3D indoor reconstruction from a point cloud. However, although available algorithms could miss several of them in a scene, a room without any door reflects necessarily a problem in the reconstruction. This is also an obvious limitation for navigation purposes. As illustrated in Figure 5.8, by relying on the navigation graph, it is straightforward to identify the problem, as it only requires detecting rooms associated with isolated nodes. Similarly, to C1, spaces detected in this checking are tagged for further improvement of the model reconstruction.

The third constraint C3 controls the connectivity of two spaces. It also relies on the navigation network and covers the cases that cannot be detected with C2 (e.g., two rooms connected by a common door but disconnected from all the other spaces). We assume that every two spaces in a correct 3D model should be reachable and connected through the navigation graph. Therefore, the process checks the existence of a path between every two rooms in the model.

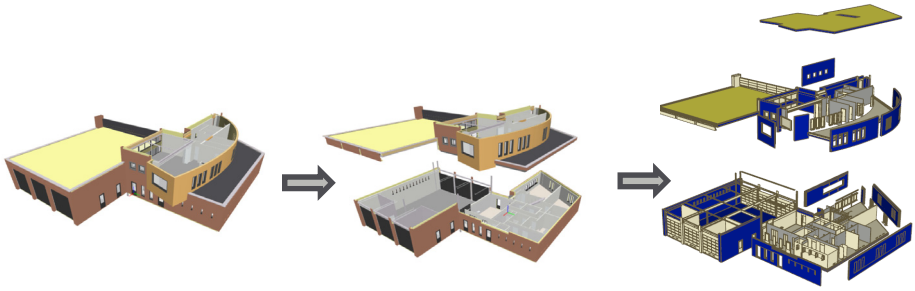
Constraint C4 completes C3 with the necessity of a staircase instance connecting two floor levels. In fact, rooms may be vertically connected due to holes or unreconstructed slabs. In such cases, while it may be assumed that the holes are elevator compartments, there is always a staircase built for use in case of emergency or elevator failure. On the other hand, despite the existence of staircase instances in the 3D model, they may not properly connect two levels. This typically happens when the number of steps in the stairs is not reconstructed correctly, breaking the C3 (for rooms in different floors) and C4.

Finally, C5 and C6 provide constraints important for navigation and related cases (e.g., emergency evacuation). By ensuring a connection to an exit from every room in the model, the consistency check guarantees that at least one of the critical doors of the building (emergency doors) is reconstructed and reachable and it is not blocked. This may not imply that the door is usable, however. Performing C5 comes down to verifying if, after all previous checks, there is at least one exit door that is connected to every space in the same division of the building. Note that using constraints such as C6 implies that the type of the door (emergency door, exit door, …) for the application should be added to the model.

In addition to C1 to C6 constraints, experts can define other constraints based on the application. Constraints C7 to C9 are examples for enforcing some geometric controls in the model. In many architecture styles walls have 90 degrees intersection angles. This can be checked with constraint C7. Similarly, constraint C8 controls whether the interior walls have the same thickness. Likewise, it is possible to control if the same type of door or window is reconstructed for some divisions of the building with constraint C9. For example, in some buildings, doors in the corridor are double doors or sliding doors. Note that although the rules for the application phase of the control grammar can be customized for different 3D models, they should not violate any of the previous rules or current geometric and topological standards.

5.4 Experiments

To test our pipeline, we use one of the models generated by the method described in (Nikoohemat et al., 2019). The selected model has walls, floors, ceilings, doors, stairs, furniture and rooms. One of the advantages of that work is that they produce two types of models, one with volumetric walls and one with spaces. Each reconstructed component is a volumetric feature, including walls, doors and rooms represented as solid geometry (polyhedron). Following our framework, the three steps of verification in the pipeline (see Figure 5.4) should be applied to the 3D model. It is therefore decomposed to its constructing components (Figure 5.9), and as a first step, the geometric and topological consistency of the solids (e.g., walls) are controlled. The decomposition of the model is done by 3D-model parser (section 5.3.3).



(a) (b) (c)

Figure 5.9. *Example of decomposing a 3D model to the components. (a) The 3D model. (b) The floor levels are separated. (c) Each level has components such as walls, floors, doors and windows which should be verified individually and in interaction with other components.*

5.4.1 Instances check

We used the model in Figure 5.10 to illustrate an example for the instance check. Based on what should be expected from 3D solids (Gröger and Plümer, 2009; Ledoux, 2013), the following rules are applied for each single component of the model:

1. Every polygon should be simple (no self-intersection).
2. The solid should be closed and there should be no dangling face or edge.
3. The normal vector of all faces of a volume should point towards the exterior of that volume.

Our 3D model is geometrically described as a Boundary Representation (B-Rep), along with information of its different classes of entities such that each single instance can be distinguished and is provided with its own list of 3D faces and semantic information. That information is used to implement the rules on the faces (polygons). The first rule means that no inconsistency on the boundary of the polygon should be accepted (e.g., self-intersecting

boundaries, duplicated vertices, etc.). No such issue could be found in the model because the whole model is triangulated, resulting in uniquely simple faces (triangles). The second rule concerns the topological closure of the solids, such that every edge of a given polygon is paired with the same edge of another polygon in the same polyhedron. Figure 5.10 illustrates a case where a volume is open because one of its sides has not been reconstructed properly and therefore has missing faces and consequently open boundaries. Such an instance should be corrected until a full closure is obtained. Finally, the last rule checks the consistency of the orientation of the volumes. The whole model should follow one convention, which is either all the face normals point inward or outward of the solid they belong to. We chose the second case, in accordance with the ISO 19107.

Once the consistency of the geometric entities is confirmed and the invalid instances are corrected, we go to the second step that check the interactions between the valid instances.

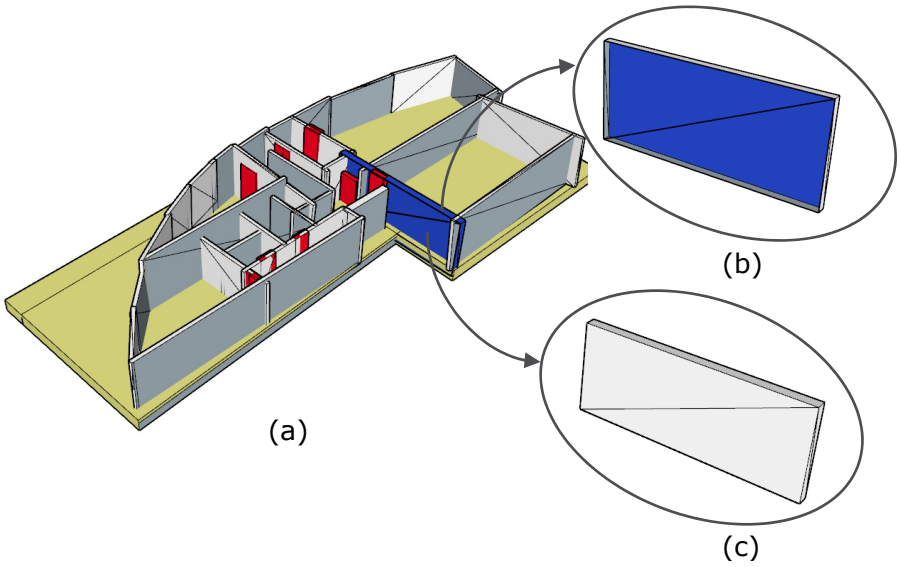


Figure 5.10. *Shows an example of geometry inconsistency for a wall object. (a) The 3D model. (b) A wall is checked and one of the faces is missing. (c) Shows the correct example of the same wall.*

5.4.2 Interactions check

As explained in consistency control of interactions (Section 5.3.5), we use an interaction matrix (Table 5.2) to determine the correct interactions in the model. First, an adjacency graph is generated from all the components in the model which have adjacency with each other. Figure 5.11 shows how a graph represents a 3D model. Nodes are components (e.g., rooms, walls, doors, and

windows) and edges are the interactions. To exemplify an interaction problem, Figure 5.12 shows one issue in the reconstruction of our test model. While in Table 5.2, it is specified that a door (opening) is expected to be adjacent to a wall, we found a door that does not share a face with its nearest wall instance. This is due to imprecision in the reconstruction, for example the wall thickness is changed without fixing its topology with a door. Other interaction inconsistencies that can be checked include walls not connected to the ceiling or stairs not attached to floors. Such issues are avoided in our model by exaggerating the extents of walls and slabs. However, this leads to other issues such as intersecting walls. Such interaction would correspond to the *overlap* relationship from the 9-intersection model, which is not considered as valid in our grammar, as it cannot happen in the reality of building elements. Therefore, this should be considered as an inconsistency. However, because the main purpose of the reconstruction of our test model was to obtain the indoor spaces, and this needed a proper space closure to be extracted (Nikoohemat et al., 2019), we can afford to ignore these interaction inconsistencies in the validation process. This shows the flexibility in the validation framework and makes a link to the next step of the process, as targeted applications may make some consistency rules more relevant or critical than others, that can just be ignored.

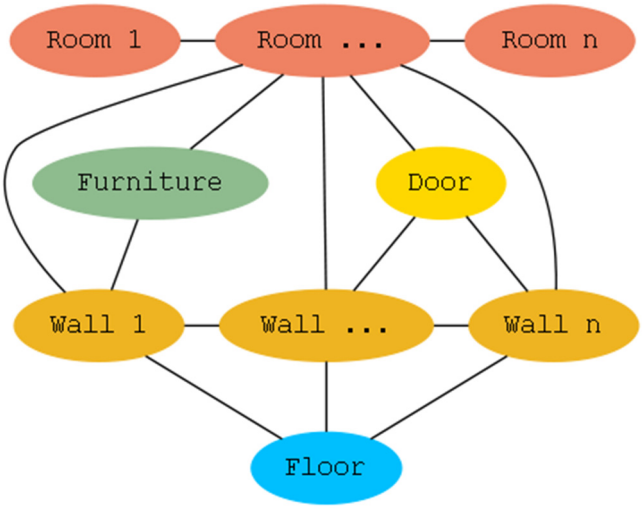


Figure 5.11. *A graph representation of a 3D model showing the connected instances. Selecting a room in the graph, the connections are stablished with walls, doors, furniture and other rooms. Here for simplicity we do not show windows and the ceiling.*

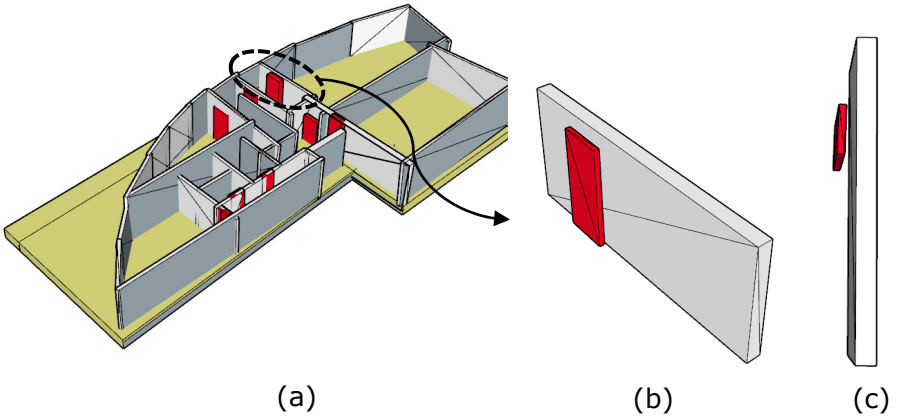


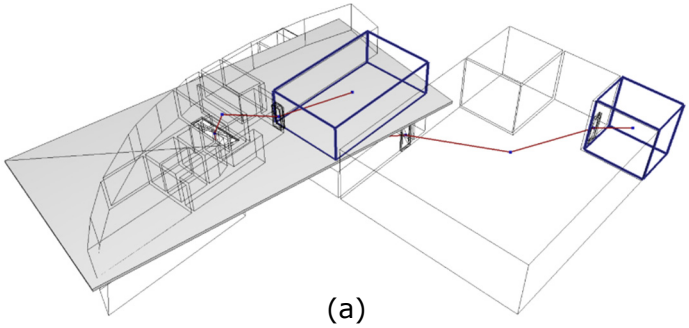
Figure 5.12. *Shows an example of checking the interaction of two instances from two different classes (wall and door). (a) The 3D model. (b) The wall and door object as adjacent objects. The door should be met or contained by the wall face. But the top view (c) shows that there is not any connection, and this is a topology error.*

5.4.3 Application consistency checks

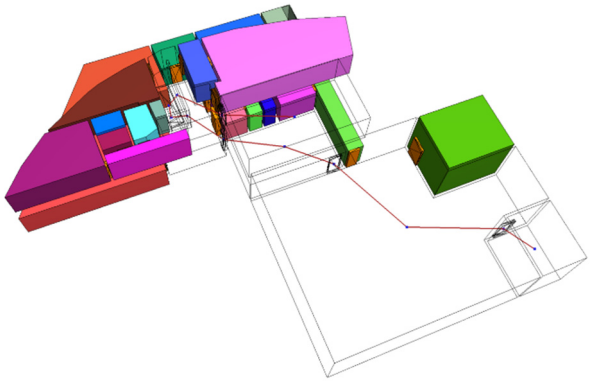
Our test model was made for indoor navigation purposes. Thus, the rules defined in Section 5.3.6 can be used for our experiment as well. While the check of rule C1 was successful, we have faced inconsistency issues related to C2. In fact, as shown in Figure 5.8, some spaces happen to be connected to no door from the whole model, which means that they are isolated spaces. This reveals missing doors that were not reconstructed or spaces that are wrongly reconstructed. In both cases, the issue needs to be addressed for an indoor navigation usage of the model and for the rest of the checks to be properly performed.

Indeed, the rule C3 for example depends on C2, because missing doors would logically lead to unconnected pairs of rooms. This case did not happen in our experiment model. Figure 5.13 illustrates the C4 check, where two floors are connected by at least one staircase. A navigation route from a space downstairs to a space upstairs goes through staircases (which is here represented by another space connecting both levels). The check C5 is performed, by defining one door as the main exit door of the building and computing the exit path from each room. Finally, C6 is checking if every emergency door has a clearance. However, this control can be applied for any door. To perform a door clearance, a buffer for the doors is considered and then the bounding box of the furniture is intersected with the buffer of the door. If an intersection occurs the door path is blocked (Figure 5.14). Adding external sources, such as the

footprint of the building or the number of rooms can help in invalidating some of the errors and checking the completeness of the model.



(a)



(b)

Figure 5.13. *In case of navigation as an application of 3D model, the pipeline checks whether every two spaces in two different levels are connected through a staircase.* (a) The wireframe of the spaces and the navigation network. (b) Another representation of the spaces and the network.

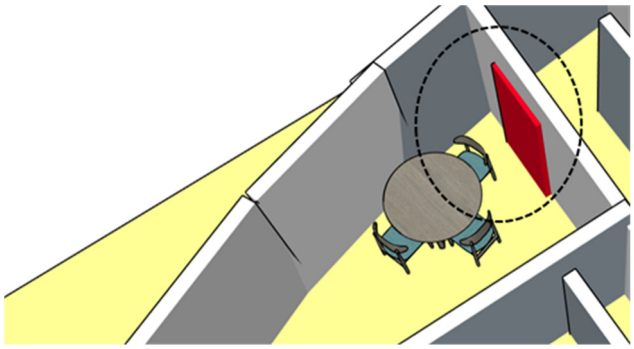


Figure 5.14. *Occlusion of an opening with the furniture can be detected by intersecting the bounding box of the furniture and the buffer of the door object.*

We proposed a framework to control the consistency of indoor 3D models in three steps. The consistency control applies the indoor standard data models and ISO Geometry standards to apply constraints in a formal grammar setting. Our control grammar does not fix the errors but helps detecting and invalidating them. The integration of correction routines can be included in a later version of the control grammar. One limitation of the current control grammar is the difficulty to detect and fix misclassified objects without user intervention. For example, if a large screen is misclassified as a wall, it is not trivial to identify it as an error. Similarly, if a room has two doors, and one door is detected but the other door is missed, because it was closed and coplanar with the wall, our control grammar cannot identify it. Although, our framework is scalable and flexible, similar to most of the grammar approaches, it needs expert knowledge for assembling suitable rules in each step and this can be time consuming. However, when it is done once, it can be applied for the same type of the model repeatedly.

5.5 Conclusion and Future work

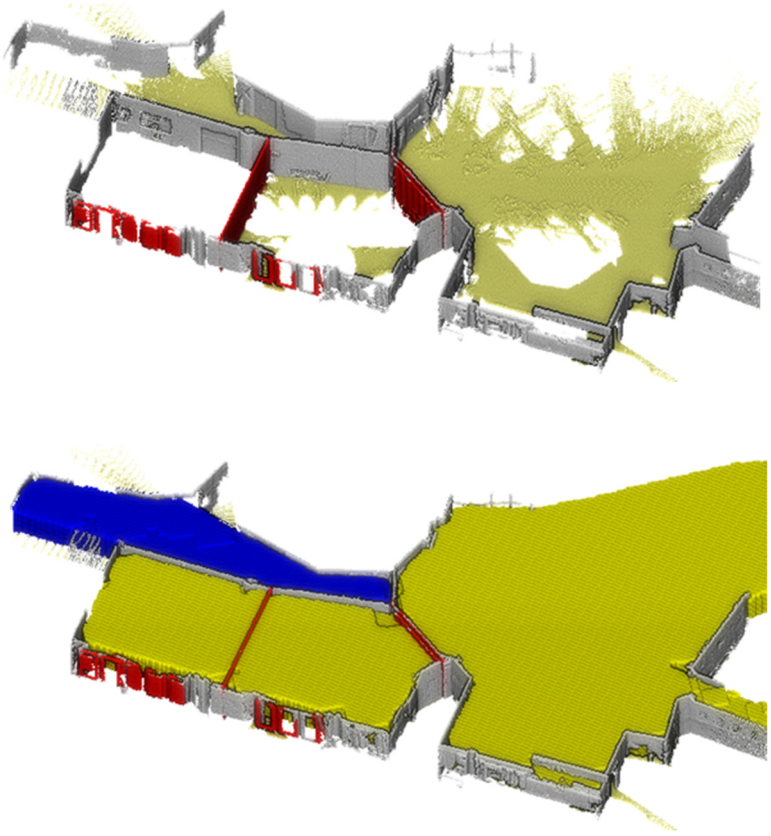
In this chapter, we have introduced a conceptual framework for consistency control of the indoor 3D models which can be deployed using a control grammar. A 3D model in our research is a digital representation of building interiors and consistency means correctness and completeness of the model. Although, the correctness and completeness cannot be checked completely in the lack of ground truth, the focus of this work is to promotes this by using the expert input and indoor standards. Our framework has a generic solution and therefore it is suitable for different types of 3D models reconstructed from point clouds, RGBD images, or floor plans. To this end, the available sources for controlling the model are the expert knowledge and indoor standards such as IFC, IndoorGML, and ISO 19107.

The final output of our grammar, after several iterations on the model, is a satisfactory 3D model which has passed several consistency tests. Our proposed solution does not correct primitives and 3D objects but it (in)validates them in three steps: first the individual instances are controlled for a geometric, topological and semantic consistency. Then every two adjacent objects are controlled using an interaction matrix and 9-intersection model schema and finally the model is verified for specific applications such as evacuation simulation. After each rejection, the flagged object should be fixed and the grammar needs the corrected component to continue the process. The correction of the errors, which may have several reasons, is the responsibility of the user. Therefore, we do not offer a correction process. However, it is possible to fix minor geometry errors such as a missing closure of a solid or intrusion of volumes. Because of the scalability and flexibility of our system, it is possible to add more nodes and define new rules. Similar to any grammar,

defining new rules needs expert knowledge. The suggested adjacency matrix and its corresponding graph help the users of the system to keep the integrity of the grammar when adding new rules. The control grammar is scalable to more complex architectures by adding more classes (pillars, furniture types) to the interaction matrix and the corresponding adjacency graph.

As the future work, our control grammar should be implemented with some handpicked rules from the ISO 19107 and/or IndoorGML or IFC for a real application. Then it is feasible to study the weaknesses of the framework for further improvements. As an extension to our control grammar it is logical to add automatic corrections instead of rejecting objects, for example, face-on-face or closeness of a polygon can be fixed automatically instead of rejection. One question is whether we could use the repetitiveness and symmetrical characteristics of building interiors to validate or correct some other parts of the model which suffer from missing data. This is something that can be integrated in the control grammar by a learning mechanism. Another future work could be the incorporation of texture and material, room functions, and agents (building occupants, wheelchair, drones) to demonstrate challenging scenarios.

Chapter 6 - Change Detection from Point Clouds in Indoor Environments *



* This chapter is based on:

1. Nikoohemat, S., Koeva, M., Oude Elberink, S. J., and Lemmen, C. H. J.: Change Detection from Point Clouds to Support Indoor 3D Cadaster, Int. Arch. Photogramm. Remote Sens. Spatial Inf. Sci., XLII-4, 451-457, https://doi.org/10.5194/isprs-archives-XLII-4-451-2018, 2018

2. Koeva, M. N., Nikoohemat, S., Oude Elberink, S. J., Morales Guarin, J. M., Lemmen, C. H. J., & Zevenbergen, J. A. (2019). Towards 3D Indoor Cadaster Based on Change Detection from Point Clouds. Remote sensing, 11(17), https://doi.org/10.3390/rs11171972

Abstract

Recently in The Netherlands, there are many examples of changes of the functionalities of buildings in time. Tracking these changes could be challenging when the building geometry will change as well; for example, a change from administrative to residential use of the space, or merging two spaces in the building without updating the functionality. To record the changes, a common practice is to use 2D plans for subdivisions and to assign new rights, restrictions and responsibilities for the changes in a building. In the meantime, with the advances of 3D data collection techniques, the benefits of 3D models in various forms are increasingly being researched. The current work explores the opportunities of using the 3D point clouds to establish a platform for 3D Cadaster studies in indoor environments. We investigate the changes in time in the geometry of the building that can be automatically detected from point clouds to update the 3D indoor cadaster. The permanent changes (e.g., walls, rooms) are automatically distinguished from dynamic changes (e.g., human, furniture) and will be linked to the space subdivisions. Finally, the results will be linked to the spatial units in an LADM. In addition to the cadaster application, the change detection methods which proposed in this work can be applied for monitoring the changes of buildings after any renovation.

6.1 Introduction

Change detection from point clouds is the common method for detecting the geometric and semantic changes over time in the urban canyons and forests. On the other hand, 3D Cadaster data models are trying to represent the physical objects in connection with the legal objects.

In the recent years there are many examples of changes of the functionalities of buildings in time. According to the statistics shared by Environmental Data Compendium (CLO, 2016)[1] in Netherlands 17% of the office, social and real estate buildings are empty in 2015. The Ministry of Interior and Kingdom Relations (BZK) and the Association of Dutch Municipalities (VNG) set up an expert team to support Municipalities in transformation from office to residential buildings. One of the examples is a nursing home, which were owned 40% by housing associations and 60% by health care organizations, changed into students' hotels or private owned apartments with private ownership. For example, the building on Figure 6.1, located in the City of Hoorn in 2015 was changed into a residential building.

It is challenging to monitor the changes in the physical objects during the lifetime of a building and accordingly update the 3D Cadaster. In this chapter, we study the changes inside buildings and try to make a bridge between the change detection and the indoor 3D Cadaster. The reason that point clouds are used for change detection and representation of the 3D Cadaster is the as-is status of the point clouds to the current situation of the cadaster (3D) spatial units. In other words, the point clouds reflect more details of the environment and they are close to the data acquisition. Furthermore, it is easy to convert the point clouds to other data representation forms such as voxel for usage in 3D Cadaster models (Oosterom, 2018).

There is a great potential in using point clouds during the life time of a building. By collecting point clouds before and after each renovation of buildings and refurbishments in the structure, for example using terrestrial laser scanner (TLS), it is fast and accurate to find the changes and update the corresponding documents and databases.

[1] Clo.nl, vacancy of office spaces, last access: Nov. 2019

Figure 6.1. *Changing from a nursing house to an apartment (Ministerie van BZK, 2015).*

This fact motivates us to use point clouds and monitor changes in support to updating the 3D Cadaster. Change detection from point clouds can be done merely on a low level of detail and just based on the geometry, or can be done on a higher level of detail by interpretation of the geometry to semantics. For example, the changes in two epochs could be because of differences in the furniture and not the permanent structure. This needs a higher level of interpretation from point clouds. On the other hand, if the differences are because of the coverage during the data acquisition, then it can be detected by a geometry comparison of two datasets. Therefore, only comparing the geometry of two-point clouds is not sufficient to interpret 3D Cadaster related changes. For understanding the changes, first, we classify the point clouds of each epoch to permanent structures (e.g., walls, floors, ceilings) and non-permanent (e.g., furniture, outliers) using the methods in (Nikoohemat et al., 2017). Additionally, we need to have an understanding of spaces inside the buildings to relate them to the 3D spatial units. Therefore, space subdivisions such as rooms, corridors, stair cases will be extracted from the point clouds of each epoch. Second, two epochs are co-registered and the geometry differences will be extracted. Third, the changes are classified to important changes such as permanent structure and non-important ones such as changes in the furniture or those caused by the acquisition coverage. Finally, the related

3D Cadaster records in the location of changes will be queried from the database and a cadaster expert can make a decision about the updating of the cadaster records.

While it is possible to automatically subdivide the indoor environments to space subdivisions from point clouds, it is not a trivial task to automatically link them to the 3D Cadaster model. Each space subdivision can represent a spatial unit or a group of spatial units in a building. These spatial units to some extent are supported in the Land Administration Domain Model (LADM) through four main classes: LA_Party, LA_RRR, LA_BAUnit and LA_SpatialUnit (LADM, ISO 19152:2012). Out of them LA_SpatialUnit which represents legal objects and LA_RRR which represents rights, restrictions and responsibilities are part of our interest. The reason that we use the LADM for our experiments is that it is more complete and recent than other cadastral data models such as FGDC (Cadastral Data Content Standard — Federal Geographic Data Committee, 2008), DM01 (Steudler, 2006) and The Legal Property Object Model (Kalantari, 2008). Additionally, unlike other cadastral data models that are based on 2D land parcels, LADM suggests modeling classes for 3D objects (Aien et al., 2013). However, there is a lack of support for 3D Cadaster in terms of data representation and spatial operations in the current 3D Cadaster models such as LADM. For example, cadaster parcels are mainly represented as a 2D parcel, while in a multi-story building there is a need to show the property as a volumetric object. The only class for supporting 3D spatial units in the LADM is the Class LA_BoundaryFace which uses GM_MultiSurface to model 3D objects. The problem of GM_MultiSurface is that it is not sufficient for 3D spatial analysis and representation (Aien et al., 2013).

Currently, there is no framework for connecting point clouds and the LADM, and there is no workflow or standard to connect a 3D model of a building to the LADM. Therefore, we set an external model between the attributed point clouds and LADM to execute 3D operations (e.g., check the topology, calculate the area) on the point clouds and feed the required data for LADM classes into the LADM. Two examples of non-residential buildings for which point clouds were acquired by means of two different MLS systems before and after renovation will be demonstrated.

In brief, our contributions are:

1. A method for comparison of two epochs of point clouds is represented which identifies and separates the permanent changes (rooms and walls) from the temporary changes (furniture and human).
2. The space subdivisions in two epochs are modeled and the changes are identified.

3. The use of change detection in indoor environments is demonstrated in 3D Cadaster applications. The space subdivision is connected to LADM and LA_SpatialUnit.

This research invites enthusiastic readers to use point clouds as a primary data representation to enrich the 3D Cadaster. The remainder of this chapter explains the related work in section 6.2 and the methodology of our work in section 6.3 followed by results and discussions on two use cases in section 6.4. The conclusion and future work are in section 6.5.

6.2 Related Work

Point clouds are a valuable source for decision makers in the domain of urban planning and land administration. Laser Scanner data including Aerial Laser Scanner (ALS), Mobile Laser Scanner (MLS) and Terrestrial Laser Scanner (TLS) have been used for reconstruction of 3D cities, building facades and roof reconstruction (Maas and Vosselman, 1999; Oude Elberink, S.J., 2009; Pu and Vosselman, 2009a). Another application of point clouds is damage assessment of the buildings before and after a disaster (Vetrivel et al., 2015). In the domain of forestry, point clouds are used for monitoring growth of trees and changes in the forest canopy. Xiao et al., (2012) use point clouds to monitor the changes of trees in urban canopies. Regarding building facades, some methods are combining images with laser scanner data to reconstruct the facades of buildings (Müller et al., 2007; Pu and Vosselman, 2009b; Teboul et al., 2010).

In the domain of cadaster, there is a need to subdivide the spatial units vertically and to have a 3D representation in 3D spatial databases. Oosterom, (2018) in "Best Practices 3D Cadasters" discusses different types of data representation for 3D models storage including voxels, vectors and point clouds. The flexibility of point clouds in conversion to voxel or vector formats makes it easier to use point clouds in cadaster. Additionally, point clouds can represent the 3D details of buildings from inside and outside.

Researchers have developed models to provide a common framework for 3D Cadaster. The main international framework for 3D Cadaster is the Land Administration Domain Model (LADM, ISO 19152, 2012). However, in LADM there is a lack of connection between spatial models such as Building Information Models (BIM) and IndoorGML. Oldfield et al., (2017) tries to fill this gap by facilitating the registration of the spatial units extracted from BIM into a Land Administration database. Aien et al., (2013) studies the 3D Cadaster in the relation with legal issues and their physical counterparts. The authors introduce a 3D Cadastral Data Model (3DCDM) to support the integration of physical objects linked with the legal objects into a 3D Cadaster.

Another application of LADM is for using the access rights for indoor navigation purposes. The access rights of spatial units are defined in the LADM and could be connected to IndoorGML for customized navigation in spatial units (Alattas et al., 2017).

In the literature, the use of point clouds for indoor cadaster is underestimated. With the recent improvements in automation of 3D modeling from point clouds (Mura et al., 2016; Nikoohemat et al., 2017; Ochmann et al., 2016) using the point clouds for indoor 3D Cadaster is promising. Additionally, there is incredible progress in subdividing the space from point clouds to semantic subdivisions such as offices, corridors, staircases and so forth (Bobkov et al., 2017; Jung et al., 2017). This progress makes it possible to connect the rights, restrictions and responsibilities (LA_RRR) to the spatial units (LA_SpatialUnits) from point clouds or after conversion to a vector model.

Another model built on LADM for supporting the 3D spatial databases in terms of land administration was developed by Rajabifard et al., (2018). The authors propose strategies for the implementation of the (3D) National Digital Cadastral Data Base (3D-NDCDB) in Malaysia. The proposed database, gives instructions for cadastral data collection, updating the data and storage. Their database is an open-source 3D database which is compliant to the LADM. Other researchers discuss the need for new spatial representations and spatial profiles (e.g., point cloud profile, for non-topological 3D parcels) (Kalogianni and Dimopoulou, 2018; Thompson, 2018).

Atazadeh et al., (2018) investigate the integration of legal information and physical information based on international standards. The LADM is used as the data model for modeling legal information while the Industry Foundation Classes (IFC) standard provides physical data elements for managing the lifecycle of buildings.

6.3 Methodology

In this section, the methods for detecting changes from point clouds will be explained. Furthermore, the relevant changes for 3D Cadaster will be distinguished from other changes and will be connected to the space subdivisions. Each space subdivision represents a semantic space that is associated with the 3D Cadaster attributes. Two exemplary buildings show the link of spatial units extracted from the point clouds to the land administration database in two epochs.

6.3.1 Case Studies

For the current research, two case study examples are used. The first case study is a building of the Technical University in Braunschweig (TUB) and the

second is the University of Twente Faculty of Geo-Information Science and Earth Observation (ITC) building. The floor plans of these buildings are shown in Figure 6.2. In Figure 6.2a, the highlighted area shows that a wall was removed and rooms were merged into one, and Figure 6.2b shows the two rooms before removing the walls. Point cloud data for the two case studies were collected with different scanners (Figure 6.3). The data for the Braunschweig building were collected with an ITC Indoor Mobile Mapping System (ITC-IMMS) (epoch1) (Karam et al., 2019) and a Zeb-Revo[2] (epoch2) For the ITC building, we used the Riegl terrestrial laser scanning system (VZ-400)[3] and a Viametris device (iMS3D)[4]. The accuracy of the point clouds varied from 0.01 to 0.06 meters depending on the laser scanner system. While the noise in the data of mobile mapping systems was larger than the terrestrial laser scanner (TLS), the scene coverage of a mobile mapping system was more than a TLS. The noise in the data could have been caused by sensors, data acquisition algorithms, and the reflective surfaces. For more information on the comparison of scanning systems, refer to the study by Lehtola et al. (2017). In the following subsections, the detailed methodology is explained based on the first case study.

[2] www.geoslam.com/solutions/zeb-revo-rt
[3] www.riegl.com
[4] www.viametris.com

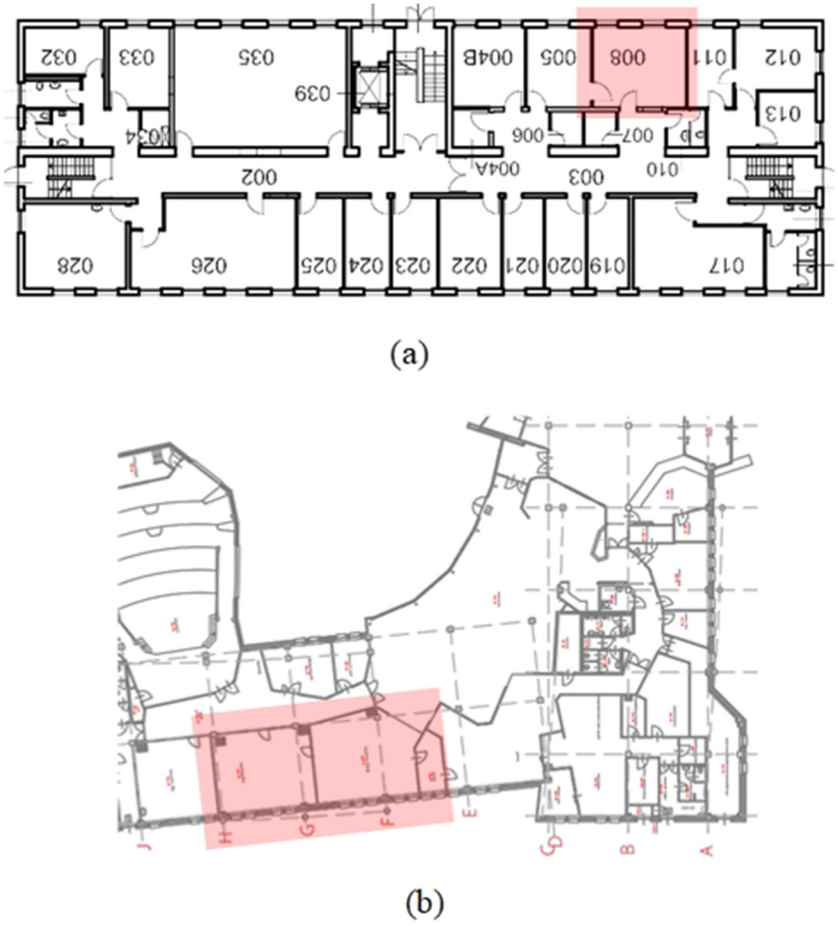


Figure 6.2. *The floor plans for our two case studies. (a) TU Braunschweig after the change (floor2). Note: the room numbers in the map are upside-down, because we want to keep the orientation of the floor plan consistent with other images. (b) University of Twente ITC building (floor 1) before the change. The highlighted areas show the rooms where the changes happened.*

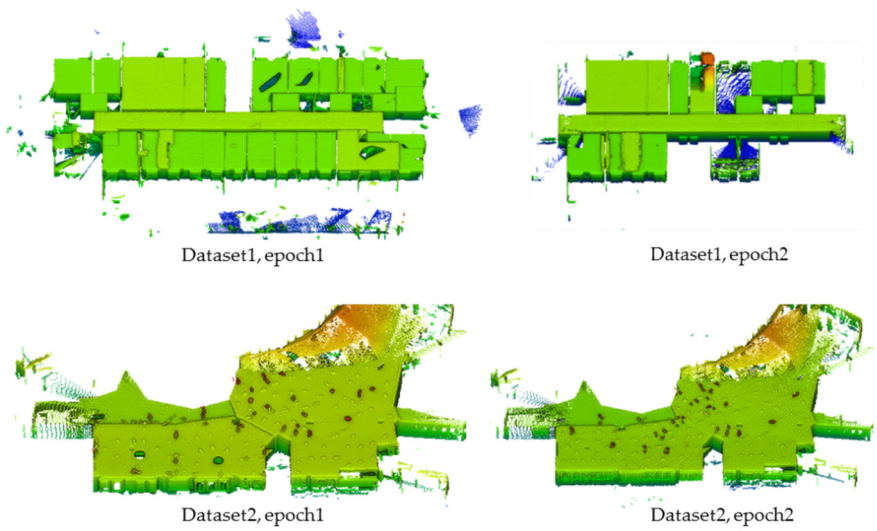


Figure 6.3. *The datasets for two different epochs. The first row is the dataset which belongs to the Braunschweig building and the second row is from the ITC building and is a more complex dataset, with furniture and large glass windows.*

6.3.2 Indoor change detection from point clouds

Differences in two epochs of point clouds inside the buildings can be categorized as:

1. Changes in the dynamic objects (e.g., furniture).
2. Changes in the permanent structure (walls, floors, rooms).

There are some other differences between two epochs of point clouds that are interpreted as:

3. Differences because of the acquisition coverage.
4. Differences because of the sensors and registration error.

In our approach, categories number 1 and 2 are dealt with as important changes for 3D Cadaster and categories 3 and 4 are just inevitable differences in two epochs that occurred because of data acquisition systems and are not relevant to the changes in the permanent structure of the building.

The process of change detection starts with the co-registration of two point clouds which are acquired with different laser scanners, one a Zeb-Revo handheld MLS and the other with a backpack system. The co-registration of two point clouds datasets is done with a straightforward approach such as ICP (Besl and McKay, 1992) and there is a lot of research in this domain (Makadia et al., 2006; Rabbani et al., 2007).

After the registration, two point clouds datasets are compared based on a distance threshold *d* to detect the differences caused by the registration error

and sensor differences (the fourth category mentioned earlier). The distance threshold can be chosen by summing up the *registration error* and *sensor noise*. The registration error and sensor noise already introduce some differences between the two datasets. The registration errors are defined after each co-registration process. The sensor noise is obtained from the specification of the systems. This threshold describes that points from two datasets with a distance of less than the threshold *d* are not considered as changes and they are in the 4th category because of the differences in the sensors. Points that have distances more than the threshold are in one of the other three categories. In our experiment, we define the distance threshold 10 cm. Let the point clouds from epoch one (acquired by a backpack) be PC1 (Figure 6.4a) and the point clouds from the second epoch (acquired by Zeb-Revo) be PC2 (Figure 6.4b).

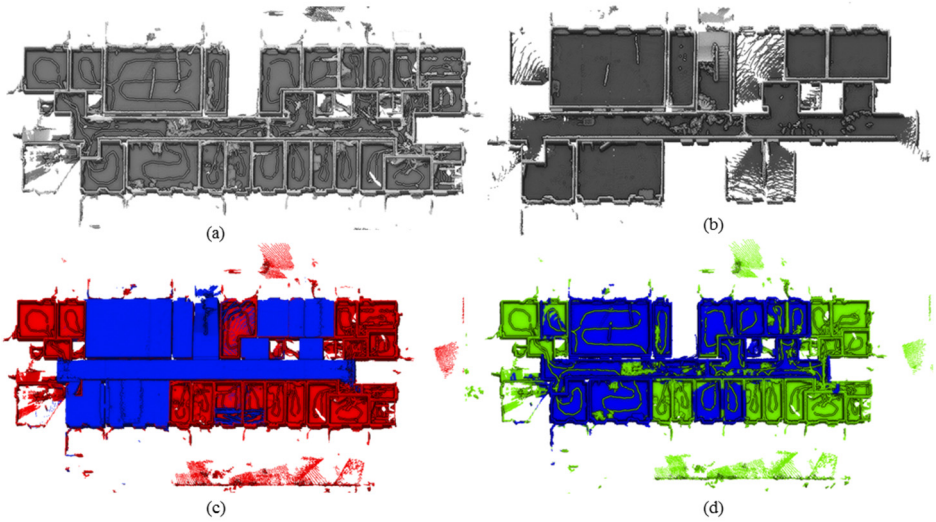


Figure 6.4. *(a) Point clouds from a backpack system from the first epoch. (b) Point clouds from a Zeb-Revo system, from the second epoch. (c) Co-registered point clouds. Blue shows the Zeb-Revo. The red areas indicate the differences in the coverage where PC1 is not covered by PC2. (d) Point clouds of epoch1 after the comparison with the epoch2. The blue points show the points with distance differences below the threshold and not changed. The green points show the differences because of coverage or furniture or a permanent change. The ceiling is removed for better visualization.*

The point to point comparison is based on the reconstruction of a Kd-tree (Friedman et al., 1977; Greenspan and Yurick, 2003) and comparing the distance of the points in PC1 from the PC2. This distance is stored in points in PC1. Using this method, the differences caused by an acquisition system and registration error are excluded from the real changes (Figure 6.4d). In the next step, the differences are further analyzed to detect and to exclude the acquisition coverage (3rd category). Our change detection method is based on analysing two geometric differences between two point clouds. This is done in a two-step approach:

1. The distinction is made between object changes and coverage differences.
2. The object changes are separated into changes on permanent structures and dynamic objects such as persons and furniture.

The geometric differences are calculated by determining the nearest 2D point, and the nearest 3D point in the other epoch. The first nearest point is based on the X, Y coordinates and the second on X, Y, Z coordinates. Figure 6.5 shows both geometric distances in 2D and 3D, as a point attribute categorized in three colors: green < 20 cm, yellow >20 cm and <50 cm, red > 50cm to the nearest.

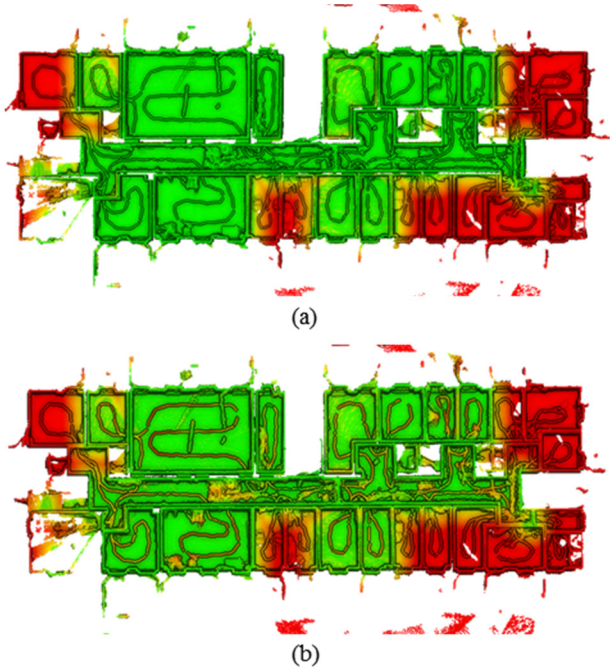


Figure 6.5. *The distance (green < 20 cm, yellow<50 cm, red > 50cm) to the nearest point in a) 2D and b) 3D.*

For both object changes and coverage differences, it is expected that the nearest 3D point is further than a certain threshold. However, the nearest 2D point is close for a changed object but not in case of coverage differences. Points are temporarily labeled as part of changed objects if the distance to the nearest point in 3D is larger than 50cm, but the nearest point in 2D is less than 50 cm. Next, the whole point cloud is segmented into planar segments, and only the vertical segments with a majority of points labeled as potentially changed are considered to be changed. The planar segmentation is performed by the region growing algorithm presented by Vosselman et al., (2004). Note that in this way also the points on a newly built wall near the ground or ceiling are included in the changed objects. The vertical segments labeled as changed

objects include permanent structures, such as walls, but also dynamic objects such as persons. In the second step, the aim is to separate permanent from temporary changes, by looking at a method described in (Nikoohemat et al., 2017).

6.3.3 Classify changes to permanent and temporary

The next step is to separate the changes that are part of the permanent structure from dynamic objects. This includes classifying the point clouds in each epoch to a permanent structure (e.g., walls, floors, ceilings) and temporary objects (e.g., furniture, clutter and outliers). We apply a method from (Nikoohemat et al., 2017) to semantically label the point clouds in each epoch (see Figure 6.7). This classification starts with a surface growing segmentation and generating an adjacency graph from the connected segments. By analysing the adjacency graph, it is possible to separate permanent structures such as walls because of their connection to the floor and ceiling. Figure 6.7c shows the permanent structure (walls and floor) is separated from the temporary objects (clutter, furniture and outliers).

After the classification of points in each epoch, by comparing the changes with the semantic labels (walls, floors, and ceilings), it is possible to distinguish relevant changes for 3D Cadaster. Each point in the set of changes is a possible change for 3D Cadaster if it is labeled as a wall, floor, or ceiling. Otherwise it is a change only because of furniture or dynamic objects or outliers. Table 6.1 shows how we identified changes with labels per point, respecting the permanent structure. According to the table, we have four types of labels which categorizes the points to permanent structures with a change (label 1) or no-change (label 3), and to non-permanent structures with a change (label 0) and no-change (label 2). Points with label 1 are important for change detection in 3D Cadaster because they represent a permanent change in the building. Figure 6.7c represents the changes with different colors according to their label.

Table 6.1. *The table shows how the point clouds are labeled regarding the changes and their role in the building structure. The points with label 1 are interesting for change detection of the 3D Cadaster.*

Labels of Points in the Data	Non-permanent Structure	Permanent Structure
Change	0	1
No change	2	3

6.3.4 Changes in the relation with Indoor Space subdivisions

The process of detection of permanent changes is continued by linking these changes to the volumetric space or space subdivisions. Space subdivisions represent the semantic space in the indoor environment such as offices,

corridors, parking space, staircases and so forth. Each space subdivision is connected to space in a spatial unit in the 3D Cadaster model and all laser points in the space subdivisions carry the attributes of the corresponding cadaster administration. In this step, we explain how these space subdivisions could be extracted from the point clouds and linked to the previously detected changes. Note that an apartment may consist out of one or more spatial units. A spatial unit may consist out of one or more spaces. A spatial unit may have invisible boundaries and needs to be checked by a cadaster expert.

To detect the spatial units from the point clouds, following the method in (Nikoohemat et al., 2017), after the extraction of the permanent structures in each epoch, a voxel grid is reconstructed from the point clouds including walls, floors and ceilings. Then using 3D morphology operation on the voxel grid, the space is subdivided into rooms and corridors. Each space subdivision is represented with the center of voxels as a point cloud segment. To find out in which space subdivisions the changes occurred, we intersect the space subdivisions of each epoch with the permanent changes detected in the previous step (see Figure 6.6). For example, in Figure 6.7, we can see that in the second epoch (Figure 6.7b) a wall is removed and two spaces are merged. Since this wall was detected as a change during the previous step (Figure 6.7c), by the intersection of changed objects with subdivisions, the conflicts in the two epochs are extracted (Figure 6.6). This conflict is linked to a space subdivision and each space subdivision or a group of them (e.g., a building level) may represent a spatial unit in the 3D Cadaster model.

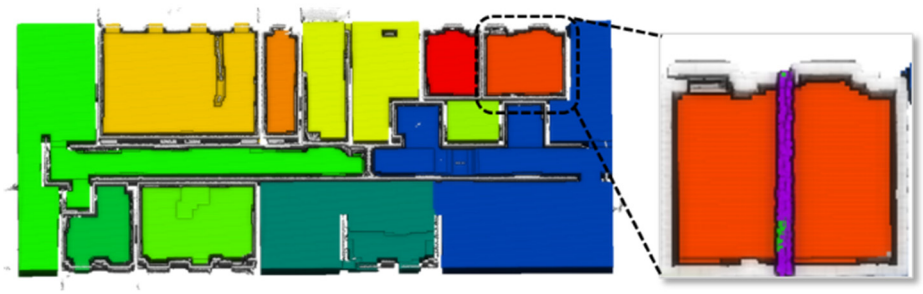


Figure 6.6. *The space subdivisions of PC2 (second epoch), after the change. The purple wall in the right image shows the intersection of a detected change with a space subdivision.*

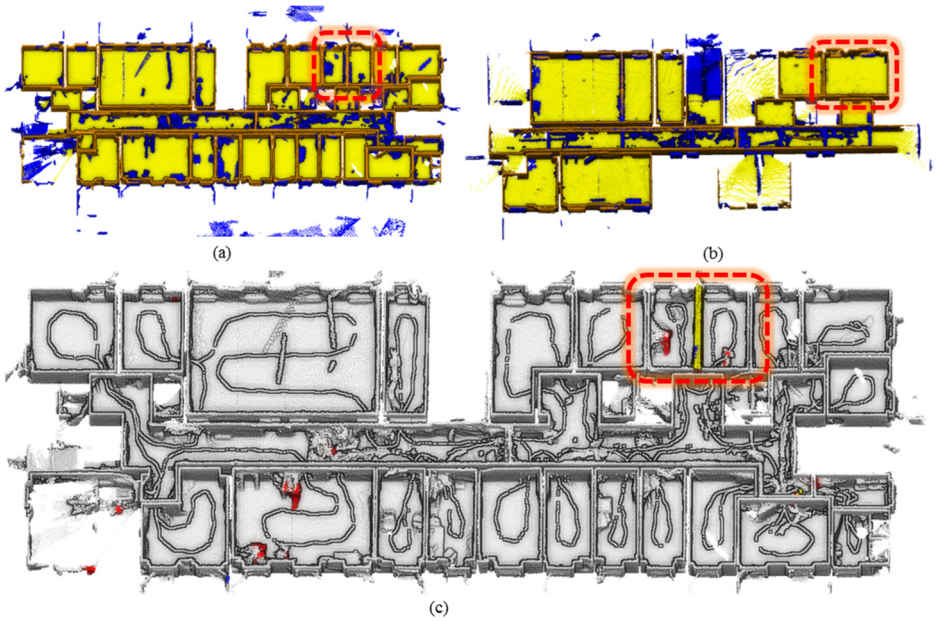


Figure 6.7. *(a) PC1 acquired by a backpack and (b) PC2 is acquired by a Zeb-Revo, walls (orange), floor (yellow), clutter (blue). (c) The changes are detected in PC1 and classified as permanent structure changes (yellow) and temporary changes (red). The red rectangle shows the wall that is showing the permanent change.*

6.3.5 Changes in the relation with 3D cadaster model

To link the cadaster to the detected changes, we assume every space subdivision in a point cloud is represented in the object description of the spatial unit in the LADM. We choose LADM (Figure 6.8) as our 3D Cadaster model because it is the most recent and the most complete 3D Cadaster data model and it has a class for supporting the 3D objects. Note that an interactive refinement on the space subdivision from the previous step is necessary to group some of the subdivisions according to the 3D Cadaster legal spatial units. For example, a group of offices that belong to the same owner has an invisible boundary that should be interactively corrected.

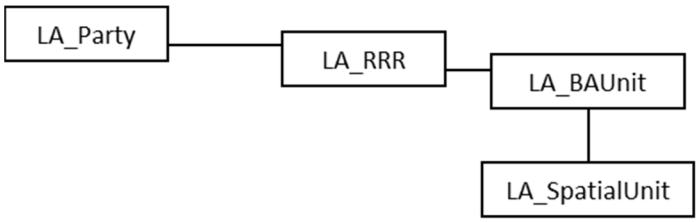


Figure 6.8. *Basic classes of the LADM (ISO 19152:2012)*

It is possible to setup the LADM in such a way that each space is represented as a spatial unit and then using the LADM class *LA_BAUnit* to associate those spatial units with a legal unit. Type of building units can be individual or shared. An individual building unit can be an apartment and can represent a legal space. In our example (Figure 6.7), the wall is removed and two spaces are merged, and their ownership is updated in the LA_BAUnit class.

Every spatial unit in LADM is modeled with *GM_MultiSurface*. 2D parcels are modeled by boundary face string (LA_BoundaryFace). The representation of 3D spatial units is done by boundary face (LA_BoundaryFace) and for the storage, a *GM_Surface* is used (see Figure 6.9). However, in our approach, we are aiming at keeping the point clouds until the last step for spatial analysis. Therefore, we use the calculated features such as volume, area and neighbor units for feeding the classes in the LADM.

All spatial attributes and legal issues such as rights and restrictions could be associated with the point clouds and the LADM. The measured spaces are important because apart from the floor space also the volumes are known. This is relevant for evaluation purposes of the individual spaces in apartments. Figure 6.10 illustrates the LADM representation of an apartment – in this case, owned by a party (right holder) named Frank. This party has an individual space and a share (1/100) in the common or shared space. Individual and shared space (including the ground parcel) compose the building as a whole.

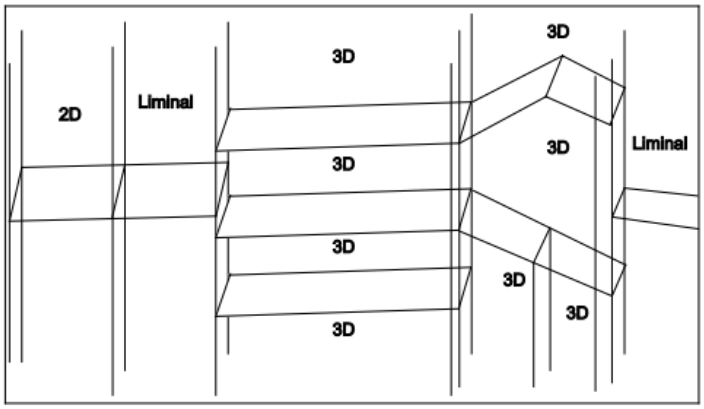


Figure 6.9. *Mixed use of boundary face strings and boundary faces defining both bounded and unbounded 3D volumes (LADM, ISO 19152:2012, Annex B).*

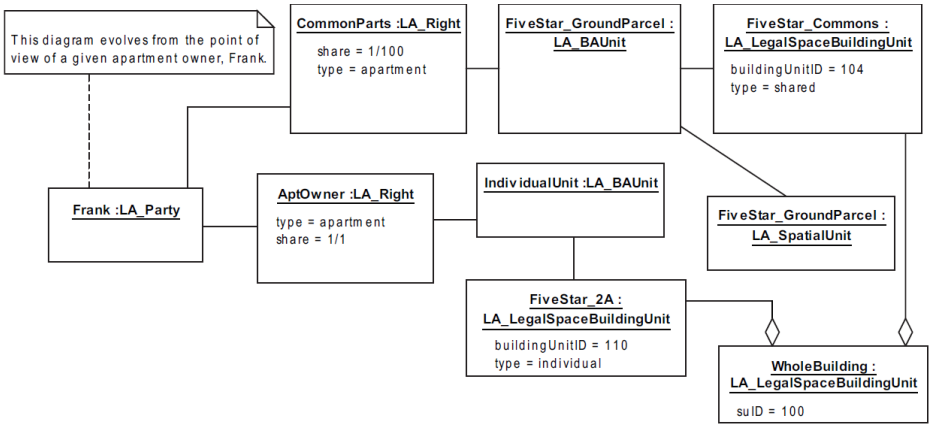


Figure 6.10. *An apartment building in LADM and its legal space (LADM, ISO 19152:2012).*

6.4 Results and Discussion

The proposed method is tested on two datasets. One dataset has a smaller amount of clutter and the shape of the building has a regular structure. Therefore, the separation of walls is easier. To challenge the robustness of our method with a complex structure and more furniture, a dataset with arbitrary wall layout and glass surfaces is selected (ITC restaurant, Figure 6.11). The details of the datasets for each epoch are in Table 6.2.

Table 6.2. *The details of the datasets and two case studies. The first and second rows belong to the first case study. The table shows the number of points and scanning device per dataset. The fourth column shows the number of changed rooms before and after the renovation of the building. The fifth column shows the items which are identified as changes.*

Dataset	# of Points	Scanner	# of changed rooms	Changed items	Figures
TU Braunschweig (epoch 1)	1.7 M	ITC-IMMS	2	Clutter, walls	2, 3, 4, 5, 7, 12
TU Braunschweig (epoch 2)	1.8 M	Zeb-Revo	1	Wall is removed.	2, 3, 4, 6, 7, 12, 14
ITC Restaurant (epoch 1)	2.8 M	Viametris, Riegl	3	Clutter, walls, curtains	2, 3, 11, 13
ITC Restaurant (epoch 2)	1.0 M	Viametris	1	Two walls are removed.	2, 3, 11

First, the datasets from two different epochs were co-registered using the iterative closest point ICP algorithm (Figure 6.11). Then the changes between two epochs were identified in 2D and 3D, as explained in the methodology. The classification algorithm separated the permanent changes from non-permanent changes and then we intersected the permanent changes with the reconstructed spaces from two epochs (Figure 6.12 and 6.13). In this way, the changes in the rooms in the second epoch of both datasets can automatically be identified. To identify the relation of physical changes with the 3D Cadaster, a user adds the ownership of the spaces as an attribute to each space. For example, the spaces which have the same rights and ownership obtain the same label and form a new physical space (Figure 6.14). Then it is possible to connect them to the basic class of the LA_Spatial Unit in the LADM and update the spatial unit class in the LADM. In dataset 2 (ITC restaurant), part of the curtain was identified as the permanent change because the curtains were covering the walls and they were detected as a permanent structure. However, this can be the inaccuracy of the classification method, for identifying the changes in the space is not problematic because it has a slight change in the space partitioning.

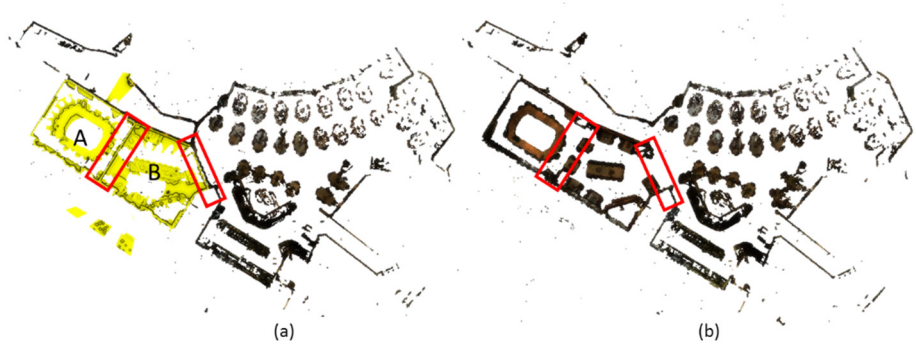


Figure 6.11. *The figure shows the top view of two epochs of our use case. The floor and ceiling are removed for a clear visualization. (a) The data is collected by a Riegl terrestrial laser scanners (TLS, VZ-400)[5] (rooms A and B in yellow) and is co-registered with the data collected by the Viametris system (iMS3D)[6]. (b) The second epoch is also collected by the Viametris system and the walls in the red rectangles are removed.*

[5] www.riegl.com
[6] www.viametris.com

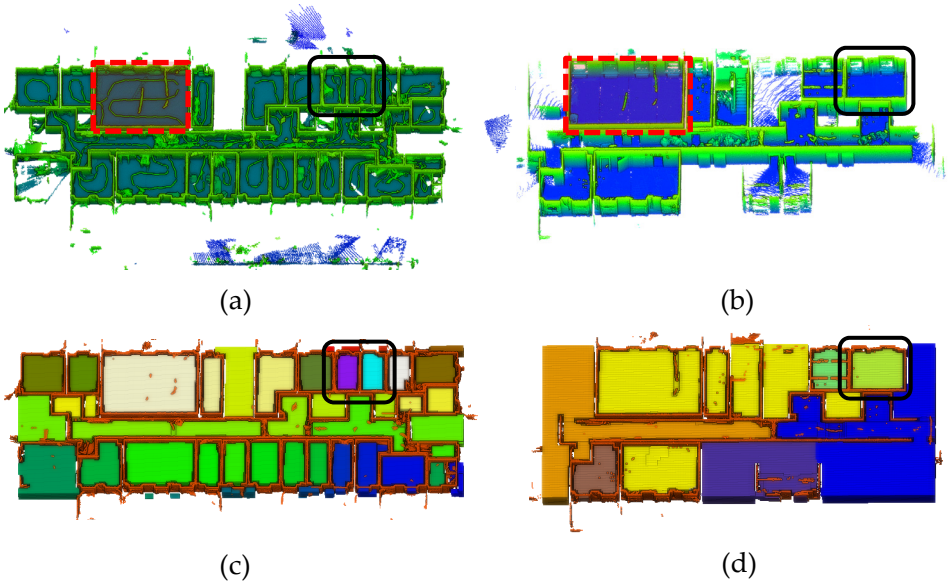


Figure 6.12. *The figure represents the changes in the detected permanent structure and then the spaces. (a) and (b) show the changes in the walls (black rectangles). The red transparent rectangle is for the orientation between two images. (c) and (d) show the detected walls in orange and space partitions in random colors. The black rectangles show how the room changed after removing a wall.*

The important parameter for the detection of changes is the distance threshold (d) to identify the changes from the differences caused by noise and registration errors. We set this parameter slightly larger than the sum of the sensor noise coming from the scanning device and the residuals coming from the ICP algorithm (less than 10 cm). In our experiments, we set this threshold on 10 cm, which implies that we cannot detect changes which are smaller than 10 cm. For planar segmentation of the point clouds, the smoothness parameter for a surface growing algorithm is important, which depends on the noise and point spacing in the data. We set the smoothness threshold to 8 cm because the noise from MLS systems (Viametris and Zeb-Revo) is around 5 cm. The smoothness parameter was set slightly larger than the sensor noise and point spacing. The point spacing was 5 cm, which meant we could subsample point clouds to reach 5 cm point spacing. The parameters for detecting the permanent structure were chosen according to (Nikoohemat et al., 2017). Segments with more than 500 supporting points were selected for creating the adjacency graph and smaller segments were discarded. The voxel size for space partitioning was 10 cm, which is an appropriate voxel size to have enough precision to identify changes and avoid expensive computations.

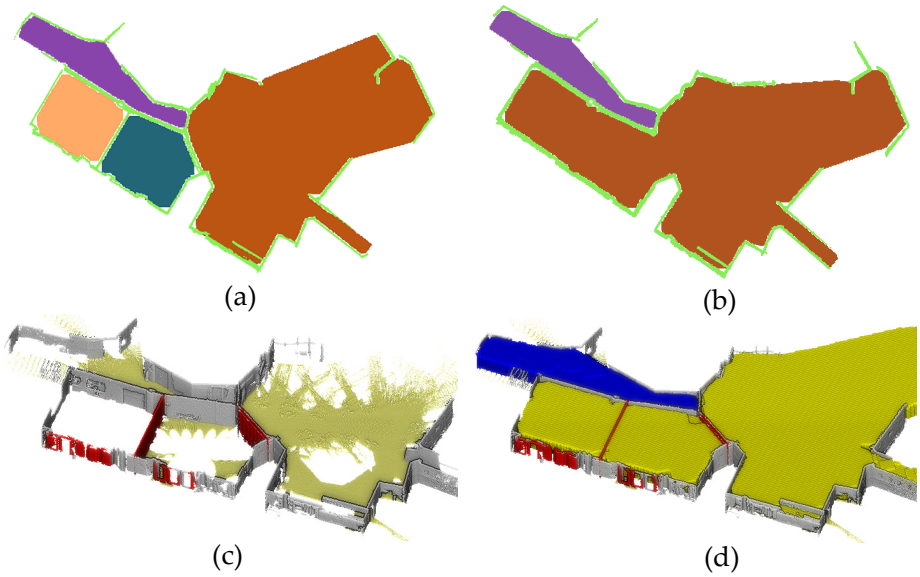


Figure 6.13. *The top view of the spaces and permanent changes. (a) Epoch one, walls are in green and four spaces in random colors. (b) After removing walls, two rooms in epoch one are merged with the large space in brown color, and, in total, it forms two spaces with the rest of the interiors. (c) Detected permanent changes are shown in red. (d) The spaces from the second epoch are intersected with the permeant changes to identify the changes in the space.*

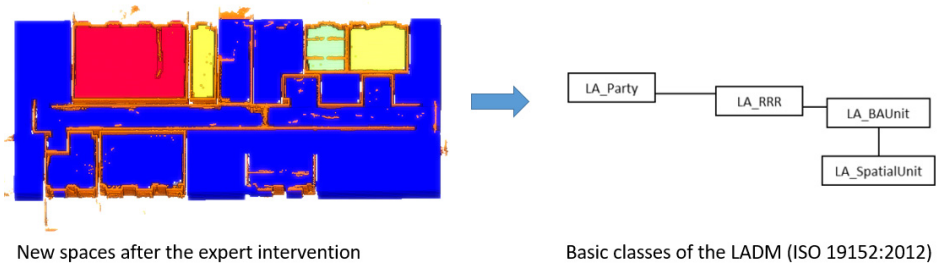


Figure 6.14. *New spaces with the same rights and ownership obtain the same label and color and form a new physical space that can be linked to the LA_SpatialUnit.*

The running time for surface growing segmentation, identifying the permanent structure, and detecting the changes for the first dataset with 1.7 million points took 2.4 min, 5.6 min, and 7 min, respectively. The space partitioning was computationally more expensive than other processes and it took 10 min for dataset 1 with the voxel size of 10 cm, and it depended on the volume of the building.

In our workflow, the challenge was detecting the permanent changes from the dynamic changes, which were not important for the Cadaster. According to

(Nikoohemat et al., 2017), this process can have an average accuracy of 93% for permanent structures and 90% for spaces (Nikoohemat et al., 2019). Furthermore, the extraction of spaces is really crucial in the process, because the volume and area are calculated from the space subdivision result. Therefore, an expert should check the results of space subdivision and merge or split some of the spaces that are extracted from the point clouds. The interactive corrections are less than 10% of the whole process and, for a building of three floors as large as our case study, it does not take more than 10 min.

The process of linking the spatial units to the 3D Cadaster model was not automated in our approach. This was because of the lack of possibilities for representation and visualization of 3D objects in the 3D Cadaster models. Therefore, our method was limited when it comes to the storage of 3D spatial objects in the Cadaster databases. As future work, linking the 3D objects and 3D Cadaster models, one solution we intend to investigate is using the point clouds as external classes and trying to keep the 3D objects as point clouds for all steps. The extraction of vector boundaries for the Cadaster models can be done with functions from the point clouds.

6.4 Conclusion and future work

We have presented a pipeline to detect geometrical changes in buildings from two epochs of point clouds captured by mobile laser scanners. In our approach, changes are recognized as dynamic changes (e.g., human, furniture) and permanent changes (e.g., walls, rooms). The permanent changes are then linked to the space subdivisions that are extracted from the point clouds of each epoch. A cadaster expert will interactively group some of the space subdivision according to their legal attributes. Then the spaces that are changed and identified during the process will be further analysed to extract spatial attributes such as boundary, area and volume. This process can be done on point clouds where changes occurred. Extracted spatial attributes can be exchanged between a cadaster model such as LADM and the point clouds. A cadaster expert should make decisions on updating the cadaster model according to the spatial changes. It should be noted that although the changes in the permanent structures for our use cases are few (two or three walls), identifying such changes is not trivial for an operator in large datasets with many rooms and more changes. Therefore, an automatic process can be a great assistant for experts.

Future work should determine the link between designed space by the architect and the real constructed space as measured with point clouds. This measurement is relevant for the composition of legal space in LADM – but also for building permits and other permits (e.g., for shops and companies). The

process of representing and linking 3D objects to the 3D Cadaster especially for indoor is an ongoing research.

The authors of this research hope that this work would introduce a new research avenue regarding the connection between point clouds and indoor cadastral models.

Chapter 7 – Synthesis

7.1 Scope of application of the proposed pipeline

This dissertation in addition to a thorough overview of the state-of-the-art in chapter 2, contributes in the reconstruction of 3D models from point clouds by suggesting a pipeline which becomes deeper in details and more sophisticated per chapter. Although this pipeline is mainly targeting disaster management applications, it generates a consistent 3D model from interiors which can be applied for other indoor use cases such as scan-to-bim.

The selected use cases are chosen with precaution to cover a wide range of problems in 3D reconstruction. Therefore, our proposed pipeline tackles problems such as: the noise from reflective surfaces, incomplete data caused by clutter, complexity of multistory buildings, various architecture layouts which are non-Manhattan-World, detection of closed doors and finally consistency control of the models.

The methodology in chapter 3 is at early stage and even though it can be applied for indoor navigation purposes and it provides the layout of the building, it does not reconstruct a fully 3D model. Instead, a space partitioning based on voxels is suggested where the navigable space is discerned from occupied space (e.g., furniture) and the partitions represent the layout of the building in 2D and 3D. However, since the space partitioning represented in chapter 3 is a voxel representation it may not be useful for some applications such as evacuation simulations which need a 3D geometry model (e.g., OBJ format). Similarly, the semantic labeling of the openings, door and windows, does not result in a geometry model of openings as it only provides the classification of the point clouds.

In chapter 4, the methods from previous chapter are improved to provide a watertight 3D model which complies with standard data-formats of 3D geometry. The output of chapter 4 is a complete pipeline for creating fully 3D models from multistory buildings including stairs and ramps. Additionally, the algorithms are improved in terms of performance to provide a 3D model in a shorter time in comparison with space partitioning using voxels. The output of this chapter can be used both for BIM applications because of volumetric representation of walls and for IndoorGML applications because of space representation. Furthermore, the modeling of furniture as obstacles provides a higher level of space subdivision for indoor navigation purposes. As an extension to the pipeline providing the function of each subspace and furniture brings more insight for an optimal routing in complex indoor environments.

In chapter 5, the methodology applies a grammar method to investigate the consistency of 3D models which are not only suggested by our method but also 3D models created from the state-of-the-art. This chapter does not dive into

details of the implementation and tries to conceptualize a workflow for checking the models in the absence of ground truth against three main available standards including IndoorGML, IFC, and ISO 19107, as well as the common knowledge of the modeler. The pipeline comes to its fulfilment by exemplifying the consistency control grammar on some cases from previous datasets. We believe consistency control of the pipeline can be more elaborated in the future on 3D model benchmarks to form comprehensive grammar rules for testing the integrity of the 3D models regardless of the method.

Chapter 6 goes beyond the proposed reconstruction pipeline and investigates an approach for change detection in indoor environments and applies it to indoor 3D cadaster cases. The findings of this chapter can be applied for maintenance and updating the 3D models by discerning the changes of the permanent structure than temporary changes. Change detection enables us to save time by updating the model in renovated places and avoiding a complete scan of unchanged areas. Change detection is a broadly researched topic in outdoor for monitoring the urban developments, while in indoor it is a new line of research and needs more attention in the future.

7.2 Conclusions per objective

This dissertation provides the state-of-the-art methods for indoor 3D modeling. This research focuses on using MLS data for indoor modeling because of the recent advances in mobile laser scanners and accordingly tackles the problems raised by such systems. The main objective of this research is proposing a pipeline for indoor 3D reconstruction from point clouds and showing applications of such models for disaster management.

In addition to the detailed discussions at the end of each chapter, here we summarize the findings and limitations of our approach per objective given in the first chapter.

7.2.1 Semantic labeling

This objective mainly addresses the research gaps including dealing with the noise, incomplete data and glass surfaces. In the presence of clutter, identifying the class of a point or a segment is more challenging. Likewise, understanding indoor scenes needs smart algorithms because indoor environments have a variety of architectures and man-made furniture. Our proposed solution for semantic labeling is presented in chapter 3. We prove that our heuristic method based on an adjacency graph and exploiting the trajectory of mobile laser scanners can reach high accuracy (average of 90%) in semantic labeling of six main classes: walls, ceilings, floors, stairs, doors and windows. This result is achieved in complex architecture styles, in cluttered environments, and for buildings with large reflective surfaces. Our method

outperforms other methods which use machine learning (Armeni et al., 2016) and deep learning (e.g., SegCloud, SnapNet, PointNet) for semantic learning (Boulch, 2018; Qi et al., 2016; Tchapmi et al., 2017), particularly in detecting doors and windows, without the need of training data. However, the number of classes in our method is fewer than other works because we are not focusing on labeling furniture.

We provide evidence that the trajectory of mobile laser scanner is a valuable source to understand the scene and should not be neglected. We used the trajectory for three cases: 1. separating the floors and stairs, 2. occlusion reasoning to detect openings, and 3. detection of closed and opened doors which are traversed by the trajectory. The main challenge of semantic labeling is when there is large glass surface in the environment or when the ceiling and walls are largely cluttered, such as built-in bookshelves, cabinets in the kitchen, pipes and ventilations in the ceiling and occlusion by curtains. Our permanent structure algorithm is robust as long as a good connection between the ceilings and walls can be identified in the data. The parameters for each algorithm are discussed in the discussion section of each chapter but generally defining robust parameters for an adjacency graph, or space partitioning is experimental and after several trials the robust algorithms can be extracted. Finally, this objective did not investigate the labeling of furniture types or the function of places. The future work can focus on adding the level of details to the model by including furniture types and room functions.

7.2.2 Geometric modeling

This objective deals with the research questions of modeling complex structures, non-Manhattan World buildings, slanted walls, ramps and sloped ceilings. Considering the fact that the proposed algorithms dealing with these problems should be scalable to large datasets. Several off-the-shelf computational geometry algorithms are adapted and further developed. In chapter 3, we explain a method for generalization of segments which share similar geometries to model wall segments. As the limitation of generalization, the small details of walls (less than generalization threshold) are not reconstructed in the model, for example a board or a picture frame can be generalized to the wall structure. In chapter 4, a method for modeling stair cases is proposed and doors are modeled with standard dimensions. The only objects that are not modeled in our pipeline are the windows frames and that is firstly because of the complexity of their shapes and secondly because modeling the layout of the buildings and curved walls have a higher priority in our pipeline. Furniture is modeled by oriented bounding boxes for space subdivision in chapter 4. An oriented bounding box is a satisfactory estimation for indoor navigation applications. However, for some applications such as interior design, the exact shape of the furniture is more useful. Modeling the specific shape of furniture can be considered in the future work.

7.2.3 Watertight 3D model reconstruction

This objective is further handling the results from previous objective and is a complementary goal after geometry modeling. Geometry modeling of walls does not necessary provide a topological correct model. For example, extracting the rooms from the walls required the topology of the walls. In most related work this is achieved by making a cell complex by extending all walls to the bounding box of the data (Mura et al., 2016; Ochmann et al., 2019) which makes unnecessary rooms to be merged or invalidated. Our method suggests a new approach based on the enclosure of spaces and exploiting other knowledge such as existence of a door per space. To reach this objective, we present two types of model: 1. a model with volumetric representation of walls suitable for BIM applications, and 2. a space representation of the model suitable for IndoorGML. The water-tightness is obtained by an automatic method which searches for disjoint structures (undershoots) and fixes them by extending to the intersection of their planes. This method guarantees the connection between walls, floor and ceilings to extract the room polyhedra using the enclosure of the space. One challenge in our method is large missing data (larger than a corridor width) which should be checked by the user. Other methods have a kind of user intervention to fix similar problems (Ochmann et al., 2019). Another challenge is to set user-defined parameters for undershoot correction, wall maximum thickness and surface growing segmentation, and voxel size for methods which use voxels. In each chapter we discuss how to achieve the best parameter settings based on the empirical knowledge.

We prove that our algorithms are robust to arbitrary room layout and walls with curves. This is exemplified with four use cases where a watertight 3D model is reconstructed from multistory buildings. Additionally, the methodology reconstructs stairs to establish the connection between floors as a crucial factor for seamless indoor navigation. Modeling stairs is neglected in most of the indoor 3D modeling related works and needs more research in indoor 3D modeling. Moreover, the modeling of volumetric walls vs. surface walls provides a standard model for BIM applications. The model can be exported to the standard geometry data-formats such as OBJ and it can be modified by the user, for example a large piece of data missing which is not identified by the algorithm can be fixed by the user. The future work in this domain should focus on reconstruction of small jagged walls instead of generalizing them as well as modeling columns and beams. One thing that can be investigated in regular and repetitive architectures is using (shape) grammar to reconstruct the model in parts with sparse data.

7.2.4 Consistency and accuracy control of 3D models

The goal of this objective is to conceptualize how the consistency of a model can be controlled in the lack of ground truth. The consistency refers to the

correctness of the model in terms of geometry, topology and semantics. The consistency control of the model is shortly investigated at the end of chapter 4 by testing three constraints on an indoor navigation graph. This objective is further studied in chapter 5 by proposing a control grammar. The grammar is a checkpoint in the pipeline where it validates the components of the model against three available standards: IndoorGML, IFC, and ISO 19107. If the component does not comply with the standards in terms of geometry, semantics or topology it will be flagged and the user (or modeler) should investigate the reason. This check is carried out in three steps: 1. instances (e.g., a wall), 2. interactions (e.g., a wall and a door) and 3. applications (e.g., for navigation). The goal of this objective is not to fix the flaws in the model but only identify them for the user. The consistency control grammar has a flexible approach which can be adapted to different models. Either a model represents volumetric walls, paper-thin walls or just spaces, all can be controlled by our method because unlike other grammar approaches, we do not define fixed rules, instead the rules are extracted from the current standards (e.g., 9-intersection model, ISO 19107) and the knowledge expert. The difference between our control grammar and existing model checkers, (e.g., Autodesk Revit Model Review, Solibri Model Checker) for e.g., BIM models, is that a model checker needs very fixed and defined data models and specific classes particularly for construction applications. While models that we investigate necessarily do not retain all classes and standards for such model checkers and thus our control grammar serves a generic and flexible approach to such models. We believe, our proposed solution is a stepstone for learning a suite of rules from different type of models and bring a standard workflow for checking the consistency of the 3D models. As the future work, the grammar can suggest corrections to the identified defects or modify them automatically while keeping the integrity of the model.

7.3 Reflections and Outlook

This PhD research was part of the project 'Smart Indoor Models in 3D' which itself is in the umbrella of a bigger research program called *Map4Scociety*. The goal of *Smart Indoor Models in 3D* project was to equip the fire brigades with an advanced means of media for disaster management in complex buildings. This is indeed a necessity for first responders to know what is the current status in large buildings such as a train station, hospital or university campus. Unfortunately, for many important public buildings just a blueprint of the complex is available and there is no digital means reflecting the details of layout and furniture of each floor. Therefore, in this PhD work, we tried to address this problem.

During this research, we continuously asked this fundamental question why governmental mapping agencies dedicate all their resources to provide maps

and geo information systems for outdoor environments when we spend most of our time inside the buildings? Luckily, in recent years with the improvements in indoor localization algorithms from one hand, and advances in indoor mobile mapping systems from the other hand, it is possible to scan the current status of the building in a short time and to guide the occupants inside the building. Furthermore, the national regulations in countries like UK is enforcing the property owners to provide the BIM when renovating or constructing a new building. Therefore, in coming years, it is expected more private and public buildings possess a digital twin of their property including a 3D model. By providing these digital models for fire brigade and auditing agencies, the inspection of buildings, for example clearance of evacuation doors, can be less time consuming.

Using mobile laser scanners (MLS) for indoor mapping is a two-fold system. On the one hand and as an advantage, they acquire the geometry of the building in short time. In most cases, this geometry is supported by 360° images which provide better visualization media for the users. On the other hand, MLS devices produce large amount of point clouds and images which require advanced and expensive software to visualize and process. While images are easier to understand for the user, point clouds need expert knowledge and customized tools for specific applications. Therefore, unlike images, labeling objects and creating 3D models from point clouds are not trivial tasks. Such a complexity raises the necessity of (semi-) automatic approaches for 3D modeling and classification of point clouds. Concerning data acquisition systems, drones and low-cost scanners are interesting devices which are finding their place in indoor mapping domain. Low-cost devices, notwithstanding their lower accuracy, can be accessible for a larger user group in comparison to expensive MLS and TLS devices. This low-cost advantage in the future can provide more applications of indoor space to ordinary users. For example, one can easily scan the room and virtually design the interiors before spending money on the furniture. Unlike low-cost systems which can cover only small areas because of the limitations of the sensor and processors, drones are more advanced devices and can autonomously scan the interiors. Therefore, in the near future data acquisition will not be a problem but dealing with large amount of data coming from all the sensors inside the building is the challenge. Thus, the need of automatic methods for delivering real time service to the occupants of buildings is necessary and 3D models are the important platform for location-based services.

In this regard, our pipeline tries to address several challenges which were as research problems in the literature. Including automatic reconstruction of fully 3D models without the assumption of horizontal floor and ceiling and with lifting Manhattan World assumptions. Additionally, as we use mainly an MLS device for scanning, dealing with the reflection from the glass surfaces was

another problem causing noise in the data which we discussed in chapter 3. Concerning the level of automation of the pipeline, one can ask how our pipeline helps the automation of 3D construction? It may be difficult to quantify the level of automation but we can claim that most of the developed algorithms need the minimum user interaction for 3D modeling of the interiors. In Chapter 4, section 4.4, there is a slight intervention of the user to resolve complex cases and large data gaps in the data. Apart from this user interaction, the expert knowledge is just required when tuning critical parameters. The critical parameters are discussed at the end of each chapter for each algorithm. In brief, from chapter 3, they are including parameter of voxel size for space partitioning, three parameters for door detection, the adjacency graph parameter and the planar segmentation parameter. We experienced, for most of the dataset these parameters can be adjusted based on the point density, point spacing, the computational time, and the dimensions of the structures (e.g., a door). In chapter 4, the parameters for undershoot correction are important to assure the closure of the spaces while avoiding extension of unnecessary elements. Parameter for merging two faces of a volumetric wall normally can be adjusted as a range between the thinnest wall and the thickest wall. The connected component analysis for extracting large furniture depends on the point spacing and for extracting stairs depends on the size and the inclination of the stairs. In chapter 5, no parameter is discussed as it is a conceptual framework for consistency control of the 3D models. In chapter 6, when using change detection, the parameter for detecting changes between two datasets is inferred from the co-registration errors and sensor noise. To conclude, we discovered our default values for most of the datasets can be generalized to other cases considering the accuracy of the data coming from the sensor. Since most of our parameters are tested for mobile laser scanners, the same parameters should work for terrestrial laser scanners as they have more accurate and dense data. Another important fact that was considered during the designing of our pipeline was the scalability of the algorithms. Because point clouds are delivered in large amounts, the algorithms should be scalable to large buildings with big data. As our pipeline is dealing with each floor independently in a building, thus the computations can be deployed in a parallel programming environment to handle large buildings.

With the recent advances in deep learning, we can ask this question that how do machine learning methods can help in the process of 3D modeling and learning parameters? Currently, there are not enough benchmarks to cover all types of building interiors and architecture styles, but considering the progress in data acquisition systems, soon we will have enough training samples to be able to classify indoor objects with high accuracy. Knowing the class of indoor objects including the permanent structures can help in geometric modeling of the interiors. Moreover, understanding the type of objects in a space can provide insight of the function of the room, for example in a classroom there

are many chairs. Therefore, although deep learning methods can not directly help to model furniture and rooms, they can indirectly provide knowledge about the interiors for mapping and modeling.

When it comes to 3D modeling from point clouds the question of the accuracy of the model is raised. It is important to consider the application of the 3D model to use a fit-to-purpose method. For example, when modeling for BIM applications, the accuracy of the permanent structures is critical, thus the data collection should be done with more accurate scanners and the walls should be modeled accurately while the spaces or subspace between furniture are not the purpose. Unlike BIM applications, extracting spaces and the room layout are more important when using the model for disaster management. For applications such as energy management, the thickness of wall, the size and location of windows, and material of the structure fit the purpose. Above all, the author believes creating an accurate and consistent 3D model should be the goal of any automatic pipeline to reach a reliable outcome. To this end, the framework in chapter 5 helps to validate the model in terms of applications.

The author recommendation is on both data collection and data processing. From our experience in this research, it is very important to choose a proper data acquisition system for the application. When deciding for digitalization of large buildings several factors should be considered: 1. budget, 2. time, 3. accuracy, 4. the complexity of the environment. The selection of laser scanner is a trade-off between budget and time. MLS devices are expensive but suitable for large projects, on the other hand TLS devices or RGBD cameras are less expensive but also less mobile. The scanning devices ranged according to their accuracy from high to low are: TLS, pushing-cart systems, back packs systems, handheld devices and RGBD cameras. The accuracy varies from 4 mm to 4 cm, thus based on the application the proper scanner can be chosen. The complexity of the environment can be a range of amount of occlusion, number of glass surfaces, number of moving objects and people. When using mobile laser scanners, it is important to consult with the manufacturer about the performance of the scanner in such environment. For data processing, most of the manufacturer deliver registered point clouds, or a software which can automatically in a short time register the points. However, for classification of the data and modeling still there is no fully automatic software. CAD Software such as Revit and PointCab facilitate the modeling process by providing tools, but it needs professional modelers with sufficient time to model a building. Therefore, automatic pipelines similar to ours bridge the gap between data collection and data modeling without the need to visualize large amount of data and provides a primary model for the modeler to be enriched further. By practicing on using such automatic approaches in corporation with CAD software, modelers can help developers to understand which part of the modeling process should be automated and to what extent.

By looking at the progress in indoor 3D modeling since 2010, in the near future, indoor modeling pipelines become more accurate in modeling details of the interiors and can replace the tedious work of modelers. Another future work can investigate seamless indoor-outdoor modeling where the 3D models of cities can be enriched by indoor models and the navigation of cars and people can be carried out seamlessly in indoor complexes. Additionally, with the emerge of autonomous drones, it is foreseen that in future, indoor environment can be monitored and modeled continuously. Therefore, real-time and light algorithms which can be deployed on limited processing devices are in demand.

Bibliography

Adan, A., Huber, D., 2011. 3D Reconstruction of Interior Wall Surfaces under Occlusion and Clutter, in: 2011 International Conference on 3D Imaging, Modeling, Processing, Visualization and Transmission(3DIMPVT).pp.275–281. doi.org/10.1109/3DIMPVT.2011.42

Adao, T., Magalhães, L., Peres, E., Pereira, F., 2014. Procedural Generation of Traversable Buildings Outlined by Arbitrary Convex Shapes. Procedia Technology 16, 310–321. doi.org/10.1016/j.protcy.2014.10.097

Afyouni, I., Ray, C., Claramunt, C., 2012. Spatial models for context-aware indoor navigation systems: A survey. JOURNAL OF SPATIAL INFORMATION SCIENCE Number 4, 85–123. doi.org/10.5311/JOSIS.2012.4.73

Aien, A., Kalantari, M., Rajabifard, A., Williamson, I., Wallace, J., 2013. Towards integration of 3D legal and physical objects in cadastral data models. Land Use Policy 35, 140–154. doi.org/10.1016/j.landusepol.2013.05.014

Alattas, A., Zlatanova, S., Van Oosterom, P., Chatzinikolaou, E., Lemmen, C., Li, K.-J., 2017. Supporting Indoor Navigation Using Access Rights to Spaces Based on Combined Use of IndoorGML and LADM Models. ISPRS International Journal of Geo-Information 6, 384. doi.org/10.3390/ijgi6120384

Aliaga, D.G., Demir, \.Ilke, Benes, B., Wand, M., 2016. Inverse Procedural Modeling of 3D Models for Virtual Worlds, in: ACM SIGGRAPH 2016 Courses, SIGGRAPH '16. ACM, New York, NY, USA, pp. 16:1–16:316. doi.org/10.1145/2897826.2927323

Ambruş, R., Claici, S., Wendt, A., 2017. Automatic Room Segmentation From Unstructured 3-D Data of Indoor Environments. IEEE Robotics and Automation Letters 2, 749–756. doi.org/10.1109/LRA.2017.2651939

Anand, A., Koppula, H.S., Joachims, T., Saxena, A., 2012. Contextually guided semantic labeling and search for three-dimensional point clouds. The International Journal of Robotics Research 0278364912461538.

Armeni, I., Sener, O., Zamir, A.R., Jiang, H., Brilakis, I., Fischer, M., Savarese, S., 2016. 3D Semantic Parsing of Large-Scale Indoor Spaces. Presented at the Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 1534–1543.

Atazadeh, B., Rajabifard, A., Kalantari, M., 2018. Connecting LADM and IFC Standards – Pathways towards an Integrated Legal-Physical Model 14.

Bansal, M., Matei, B., Southall, B., Eledath, J., Sawhney, H., 2011. A lidar streaming architecture for mobile robotics with application to 3d structure characterization, in: 2011 IEEE International Conference on Robotics and Automation. IEEE, pp. 1803–1810.

Barazzetti, L., 2016. Parametric as-built model generation of complex shapes from point clouds. Advanced Engineering Informatics 30, 298–311. doi.org/10.1016/j.aei.2016.03.005

Bassier, M., Klein, R., Van Genechten, B., Vergauwen, M., 2018. IFC Wall reconstruction from unstructured point clouds, in: Annals of the Photogrammetry Remote Sensing and Spatial Information Sciences.

Becker, S., 2009. Generation and application of rules for quality dependent façade reconstruction. ISPRS Journal of Photogrammetry and Remote Sensing 64, 640–653. doi.org/10.1016/j.isprsjprs.2009.06.002

Becker, S., Peter, M., Fritsch, D., 2015. Grammar-Supported 3d Indoor Reconstruction From Point Clouds For "As-Built" Bim. ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information Sciences 1, 17–24. doi.org/10.5194/isprsannals-II-3-W4-17-2015

Becker, S., Peter, M., Fritsch, D., Philipp, D., Baier, P., Dibak, C., 2013. Combined grammar for the modeling of building interiors. Proceedings of the ISPRS Acquisition and Modelling of Indoor and Enclosed Environments.

Besl, P.J., McKay, N.D., 1992. Method for registration of 3-D shapes, in: Sensor Fusion IV: Control Paradigms and Data Structures. International Society for Optics and Photonics, pp. 586–607.

Bobkov, D., Kiechle, M., Hilsenbeck, S., Steinbach, E., 2017. Room segmentation in 3D point clouds using anisotropic potential fields, in: 2017 IEEE International Conference on Multimedia and Expo (ICME). Presented at the 2017 IEEE International Conference on Multimedia and Expo (ICME), pp. 727–732. doi.org/10.1109/ICME.2017.8019484

Bokeloh, M., Wand, M., Seidel, H.-P., 2010. A Connection Between Partial Symmetry and Inverse Procedural Modeling, in: ACM SIGGRAPH 2010 Papers, SIGGRAPH '10. ACM, New York, NY, USA, pp. 104:1–104:10. doi.org/10.1145/1833349.1778841

Bormann, R., Hampp, J., Hägele, M., 2015. New brooms sweep clean - an autonomous robotic cleaning assistant for professional office cleaning, in: 2015 IEEE International Conference on Robotics and Automation (ICRA). Presented at the 2015 IEEE International Conference on Robotics and Automation (ICRA), pp. 4470–4477. doi.org/10.1109/ICRA.2015.7139818

Bormann, R., Jordan, F., Li, W., Hampp, J., Hägele, M., 2016. Room segmentation: Survey, implementation, and analysis, in: 2016 IEEE International Conference on Robotics and Automation (ICRA). Presented at the 2016 IEEE International Conference on Robotics and Automation (ICRA), pp. 1019–1026. doi.org/10.1109/ICRA.2016.7487234

Bosse, M., Zlot, R., Flick, P., 2012. Zebedee: Design of a spring-mounted 3-d range sensor with application to mobile mapping. Robotics, IEEE Transactions on 28, 1104–1119.

Boulch, A., Guerry, J., Le Saux, B. and Audebert, N., 2018. SnapNet: 3D point cloud semantic labeling with 2D deep segmentation networks. Computers & Graphics, 71, pp.189-198.

Boulch, A., De La Gorce, M., Marlet, R., 2014. Piecewise-Planar 3D Reconstruction with Edge and Corner Regularization. Computer Graphics Forum, Proceedings of EUROGRAPHICS Symposium on Geometry Processing 33, 55–64. doi.org/10.1111/cgf.12431

Boulch, A., Houllier, S., Marlet, R., Tournaire, O., 2013. Semantizing Complex 3D Scenes using Constrained Attribute Grammars. Computer Graphics Forum 32, 33–42. doi.org/10.1111/cgf.12170

Bowman, S.L., Atanasov, N., Daniilidis, K., Pappas, G.J., 2017. Probabilistic data association for semantic SLAM. IEEE, pp. 1722–1729. doi.org/10.1109/ICRA.2017.7989203

Broersen, T., Fichtner, F., Heeres, E.J., De LIefde, I., Rodenberg, O., Meijers, B.M., Verbree, E., Van der Spek, S.C., Ten Napel, D., 2015. Project Pointless: Identifying, visualising and pathfinding through empty space in interior point clouds using an octree approach.

Brown, G., Nagel, C., Zlatanova, S., Kolbe, T.H., 2013. Modelling 3D Topographic Space Against Indoor Navigation Requirements, in: Pouliot, J., Daniel, S., Hubert, F., Zamyadi, A. (Eds.), Progress and New Trends in 3D Geoinformation Sciences, Lecture Notes in Geoinformation and Cartography. Springer Berlin Heidelberg, pp. 1–22. doi.org/10.1007/978-3-642-29793-9_1

Budroni, A., Boehm, J., 2010. Automated 3D Reconstruction of Interiors from Point Clouds. International Journal of Architectural Computing 8, 55–73. doi.org/10.1260/1478-0771.8.1.55

buildingSMART International, 2013. Industry Foundation Classes Release 4 (IFC4).

Cadastral Data Content Standard — Federal Geographic Data Committee,2008.URL www.fgdc.gov/standards/projects/FGDC-standards-projects/cadastral/index_html (accessed 7.17.18).

Chauve, A.L., Labatut, P. and Pons, J.P., 2010, June. Robust piecewise-planar 3D reconstruction and completion from large-scale unstructured point data. In *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition* (pp. 1261-1268). IEEE. doi.org/10.1109/CVPR.2010.5539824

Chen, G., Kua, J., Shum, S., Naikal, N., Carlberg, M., Zakhor, A., 2010. Indoor localization algorithms for a human-operated backpack system, in: 3D Data Processing, Visualization, and Transmission. Citeseer.

Chen, J., Chen, B., 2008. Architectural Modeling from Sparsely Scanned Range Data. International Journal of Computer Vision 78, 223–236. doi.org/10.1007/s11263-007-0105-5

Chomsky, N., 1959. On certain formal properties of grammars. Information and Control 2, 137–167. doi.org/10.1016/S0019-9958(59)90362-6

Chomsky, N., 1956. Three models for the description of language. Information Theory, IRE Transactions on 2, 113–124.

Coughlan, J.M., Yuille, A.L., 1999. Manhattan World: compass direction from a single image by Bayesian inference, in: The Proceedings of the Seventh IEEE International Conference on Computer Vision,1999.,pp.941–947vol.2. doi.org/10.1109/ICCV.1999.790349

de Laat, R., van Berlo, L., 2011. Integration of BIM and GIS: The development of the CityGML GeoBIM extension, in: Advances in 3D Geo-Information Sciences. Springer, pp. 211–225.

Dehbi, Y., Hadiji, F., Gröger, G., Kersting, K., Plümer, L., 2016. Statistical Relational Learning of Grammar Rules for 3D Building Reconstruction. Trans. in GIS. doi.org/10.1111/tgis.12200

Dehbi, Y., Loch-Dehbi, S., Plümer, L., 2017. PARAMETER ESTIMATION AND MODEL SELECTION FOR INDOOR ENVIRONMENTS BASED ON SPARSE OBSERVATIONS, in: ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information Sciences. pp. 303–310. doi.org/10.5194/isprs-annals-IV-2-W4-303-2017

Diakité, A.A., Zlatanova, S., 2018. Spatial subdivision of complex indoor environments for 3D indoor navigation. International Journal of Geographical Information Science 32, 213–235. doi.org/10.1080/13658816.2017.1376066

Diakité, A.A., Zlatanova, S., 2016. Valid Space Description in BIM for 3D Indoor Navigation. IJ3DIM 5, 1–17. doi.org/10.4018/IJ3DIM.2016070101

Diaz-Vilarino, L., Khoshelham, K., Martínez-Sánchez, J., Arias, P., 2015. 3D Modeling of Building Indoor Spaces and Closed Doors from Imagery and Point Clouds. Sensors 15, 3491–3512. doi.org/10.3390/s150203491

Díaz-Vilariño, L., Verbree, E., Zlatanova, S., Diakité, A., 2017. Indoor modelling from SLAM-based laser scanner: Door detection to envelope reconstruction. ISPRS-International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences 345–352.

Egenhofer, M.J., Franzosa, R.D., 1991. Point-set topological spatial relations. International Journal of Geographical Information System 5, 161–174.

Egenhofer, M.J., Herring, J., 1990. A mathematical framework for the definition of topological relationships. Fourth international symposium on spatial data handling 803–813.

Elseicy, A., 2018. Semantic Enrichment of Indoor Mobile Laser Scanner Point Clouds and Trajectories. University of Twente Faculty of Geo-Information and Earth Observation (ITC).

Elseicy, A., Nikoohemat, S., Peter, M., Elberink, S.O., 2018a. Space Subdivision of Indoor Mobile Laser Scanning Data Based on the Scanner Trajectory. Remote Sensing 10, 1815. doi.org/10.3390/rs10111815

Espinace, P., Kollar, T., Roy, N., Soto, A., 2013. Indoor scene recognition by a mobile robot through adaptive object detection. Robotics and Autonomous Systems 61, 932–947. doi.org/10.1016/j.robot.2013.05.002

ESRI, 2012. Building Interior Space Data Model (BISDM) [WWW Document].URLwww.esri.com/industries/government/facilities/brochures_whitepapers

Fichtner, F.W., Diakité, A.A., Zlatanova, S., Voûte, R., 2018. Semantic enrichment of octree structured point clouds for multi-story 3D pathfinding. Transactions in GIS 22, 233–248. doi.org/10.1111/tgis.12308

Foster, P., Sun, Z., Park, J.J., Kuipers, B., 2013. VisAGGE: Visible angle grid for glass environments, in: 2013 IEEE International Conference on Robotics and Automation. pp. 2213–2220. doi.org/10.1109/ICRA.2013.6630875

Friedman, J.H., Bentley, J.L., Finkel, R.A., 1977. An algorithm for finding best matches in logarithmic expected time. ACM Transactions on Mathematical Software (TOMS) 3, 209–226.

Frueh, C., Jain, S., Zakhor, A., 2005. Data Processing Algorithms for Generating Textured 3D Building Facade Meshes from Laser Scans and Camera Images. Int J Comput Vision 61, 159–184. doi.org/10.1023/B:VISI.0000043756.03810.dd

Furukawa, Y., Curless, B., Seitz, S.M., Szeliski, R., 2009. Reconstructing building interiors from images, in: In Proc. of the International Conference on Computer Vision (ICCV.

Gips, J., Stiny, G., 1980. Production Systems and Grammars: A Uniform Characterization. Environ Plann B Plann Des 7, 399–408. doi.org/10.1068/b070399

Girard, G., Côté, S., Zlatanova, S., Barette, Y., St-Pierre, J., Van Oosterom, P., 2011. Indoor Pedestrian Navigation Using Foot-Mounted IMU and Portable Ultrasound Range. Sensors 11, 7606–7624. doi.org/10.3390/s110807606

Goetz, M., Zipf, A., 2013. Indoor Route Planning with Volunteered Geographic Information on a (Mobile) Web-Based Platform, in: Krisp, J.M. (Ed.), Progress in Location-Based Services, Lecture Notes in Geoinformation and Cartography. Springer Berlin Heidelberg, pp. 211–231. doi.org/10.1007/978-3-642-34203-5_12

Greenspan, M., Yurick, M., 2003. Approximate kd tree search for efficient ICP, in: 3D Digital Imaging and Modeling, International Conference on (2003). IEEE, p. 442. http://doi.ieeecomputersociety.org/10.1109/IM.2003.1240280

Gröger, G., Plümer, L., 2010. Derivation of 3D Indoor Models by Grammars for Route Planning. Photogrammetrie - Fernerkundung - Geoinformation 2010, 191–206. doi.org/10.1127/1432-8364/2010/0049

Gröger, G., Plümer, L., 2009. How to achieve consistency for 3D city models. Geoinformatica 15, 137–165. doi.org/10.1007/s10707-009-0091-6

Gröger, Thomas H. Kolbe, Claus Nagel, Karl-Heinz Häfele, 2012. OGC City Geography Markup Language (CityGML) Encoding Standard.

Hedau, V., Hoiem, D., Forsyth, D., 2009. Recovering the spatial layout of cluttered rooms. IEEE, pp. 1849–1856. doi.org/10.1109/ICCV.2009.5459411

Horna, S., Damiand, G., Meneveaux, D., Bertrand, Y., 2007. Building 3D indoor scenes topology from 2D architectural plans, in: GRAPP, Proc. of 2nd International Conference on Computer Graphics Theory and Applications. Barcelona, Spain, pp. 37–44.

Horna, S., Meneveaux, D., Damiand, G., Bertrand, Y., 2009. Consistency constraints and 3D building reconstruction. Computer-Aided Design 41, 13–27. doi.org/10.1016/j.cad.2008.11.006

Hu, X., Fan, H., Noskov, A., Zipf, A., Wang, Z., Shang, J., 2019. Feasibility of Using Grammars to Infer Room Semantics. Remote Sensing 11, 1535. doi.org/10.3390/rs11131535

Huitl, R., Schroth, G., Hilsenbeck, S., Schweiger, F., Steinbach, E., 2012. TUMindoor: An extensive image and point cloud dataset for visual indoor localization and mapping, in: 2012 19th IEEE International Conference on Image Processing (ICIP). pp. 1773–1776. doi.org/10.1109/ICIP.2012.6467224

Ikehata, S., Yang, H., Furukawa, Y., 2015. Structured indoor modeling, in: Proceedings of the IEEE International Conference on Computer Vision. pp. 1323–1331. doi.org/10.1109/ICCV.2015.156

Isikdag, U., Zlatanova, S., Underwood, J., 2013. A BIM-Oriented Model for supporting indoor navigation requirements. Computers, Environment and Urban Systems 41, 112–123. doi.org/10.1016/j.compenvurbsys.2013.05.001

ISO, 2019. ISO 19107 [WWW Document]. ISO. URL www.iso.org/cms/render/live/en/sites/isoorg/contents/data/standard/06/61/66175.html (accessed 11.28.19).

ISO, 2018. ISO 16739-1:2018 [WWW Document]. ISO. URL http://www.iso.org/cms/render/live/en/sites/isoorg/contents/data/standard/07/03/70303.html (accessed 11.28.19).

ISO 19152:2012, 2012. Geographic Information – Land administration domain model (LADM). International Organization for Standardization, Geneva, Switzerland.

ISO, 2003. ISO 19107, Geographic information – Spatial Schema. International Organization for Standardization, Geneva, Switzerland.

Izadi, S., Kim, D., Hilliges, O., Molyneaux, D., Newcombe, R., Kohli, P., Shotton, J., Hodges, S., Freeman, D., Davison, A., Fitzgibbon, A., 2011. KinectFusion: Real-time 3D Reconstruction and Interaction Using a Moving Depth Camera, in: Proceedings of the 24th Annual ACM

Symposium on User Interface Software and Technology, UIST '11. ACM, New York, NY, USA, pp. 559–568. doi.org/10.1145/2047196.2047270

Jenke, P., Huhle, B., Stra\s ser, W., 2009. Statistical reconstruction of indoor scenes.

Jung, H., Lee, J., 2015. INDOOR SUBSPACING TO IMPLEMENT INDOORGML FOR INDOOR NAVIGATION. ISPRS-International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences 1, 25–27. doi.org/10.5194/isprsarchives-XL-2-W4-25-2015

Jung, J., Hong, Sungchul, Jeong, S., Kim, S., Cho, H., Hong, Seunghwan, Heo, J., 2014. Productive modeling for development of as-built BIM of existing indoor structures. Automation in Construction 42, 68–77. doi.org/10.1016/j.autcon.2014.02.021

Jung, J., Stachniss, C., Kim, C., 2017. Automatic Room Segmentation of 3D Laser Data Using Morphological Processing. ISPRS International Journal of Geo-Information 6, 206. doi.org/10.3390/ijgi6070206

Kada, M., 2007. Generalisation of 3D building models by cell decomposition and primitive instancing, in: Proceedings of the Joint ISPRS Workshop on "Visualization and Exploration of Geospatial Data", Stuttgart, Germany.

Kalantari, M., 2008. Cadastral Data Modelling-A Tool for e-Land Administration.

Kalogianni, E., Dimopoulou, E., 2018. 3D Cadastre and LADM – Needs and Expectations towards LADM Revision. 7 th International FIG Workshop on the Land Administration Domain Model 11-13 April 2018, Zagreb, Croatia 22.

Karam, S., Vosselman, G., Peter, M., Hosseinyalamdary, S., Lehtola, V., 2019. Design, Calibration, and Evaluation of a Backpack Indoor Mobile Mapping System. Remote Sensing 11, 905. doi.org/10.3390/rs11080905

Karger, D., Motwani, R., Ramkumar, G.D.S., 1997. On approximating the longest path in a graph. Algorithmica 18, 82–98. doi.org/10.1007/BF02523689

Khan, A., Kolbe, T., 2013. Subspacing based on connected opening spaces and for different locomotion types using geometric and graph based representation in multilayered space-event model (MLSEM), in: Proceedings of the 8th 3D GeoInfo Conference. doi.org/10.5194/isprsannals-II-2-W1-173-2013

Khoshelham, K., Diaz-Vilarino, L., 2014. 3D Modeling of Interior Spaces: Learning the Language of Indoor Architecture, in: Proceedings of the ISPRS Technical Commission V Symposium, Riva Del Garda, Italy. doi.org/10.5194/isprsarchives-XL-5-321-2014

Khosravani, A.M., 2016. Automatic modelling of building interiors using low-cost sensor systems. Institute for Photogrammetry, University of Stuttgart, Stuttgart, Germany.

Koch, R., May, S., Murmann, P., Nüchter, A., 2017. Identification of transparent and specular reflective material in laser scans to discriminate affected measurements for faultless robotic SLAM. Robotics and Autonomous Systems 87, 296–312. doi.org/10.1016/j.robot.2016.10.014

Koeva, M., Nikoohemat, S., Oude Elberink, S., Morales, J., Lemmen, C., Zevenbergen, J., 2019. Towards 3D Indoor Cadastre Based on Change Detection from Point Clouds. Remote Sensing 11, 1972. doi.org/10.3390/rs11171972

Kolbe, T.H., Gröger, G., Plümer, L., 2005. CityGML: Interoperable access to 3D city models, in: Geo-Information for Disaster Management. Springer, pp. 883–899.

Koppula, H.S., Anand, A., Joachims, T., Saxena, A., 2011. Semantic Labeling of 3D Point Clouds for Indoor Scenes, in: Shawe-Taylor, J., Zemel, R.S., Bartlett, P.L., Pereira, F., Weinberger, K.Q. (Eds.), Advances in Neural Information Processing Systems 24. Curran Associates, Inc., pp. 244–252.

Ledoux, H., 2013. On the Validation of Solids Represented with the International Standards for Geographic Information. Computer-Aided Civil and Infrastructure Engineering 28, 693–706. doi.org/10.1111/mice.12043

Lee, J., Li, K.-J., Zlatanova, S., Kolbe, T.H., Nagel, C., Becker, T., 2014. OGC® IndoorGML [WWW Document]. IndoorGML SWG. URL http://docs.opengeospatial.org/is/14-005r3/14-005r3.html#12 (accessed 9.8.15).

Lehtola, V.V., Kaartinen, H., Nüchter, A., Kaijaluoto, R., Kukko, A., Litkey, P., Honkavaara, E., Rosnell, T., Vaaja, M.T., Virtanen, J.-P., 2017. Comparison of the Selected State-Of-The-Art 3D Indoor Scanning and Point Cloud Generation Methods. Remote Sensing 9, 796. doi.org/10.3390/rs9080796

Liu, C., Wu, J., Furukawa, Y., 2018. FloorNet: A Unified Framework for Floorplan Reconstruction from 3D Scans. arXiv preprint arXiv:1804.00090.

Liu, T., Chaudhuri, S., Kim, V.G., Huang, Q., Mitra, N.J., Funkhouser, T., 2014. Creating Consistent Scene Graphs Using a Probabilistic Grammar. ACM Trans. Graph. 33, 211:1–211:12. doi.org/10.1145/2661229.2661243

Loch-Dehbi, S., Dehbi, Y., Plümer, L., 2017. Estimation of 3D Indoor Models with Constraint Propagation and Stochastic Reasoning in the Absence of Indoor Measurements. ISPRS International Journal of Geo-Information 6, 90. doi.org/10.3390/ijgi6030090

Maas, H.-G., Vosselman, G., 1999. Two algorithms for extracting building models from raw laser altimetry data. ISPRS Journal of Photogrammetry and Remote Sensing 54, 153–163. doi.org/10.1016/S0924-2716(99)00004-0

Macher, H., Landes, T., Grussenmeyer, P., 2017. From Point Clouds to Building Information Models: 3D Semi-Automatic Reconstruction of Indoors of Existing Buildings. Applied Sciences 7, 1030. doi.org/10.3390/app7101030

Makadia, A., Patterson, A., Daniilidis, K., 2006. Fully Automatic Registration of 3D Point Clouds, in: 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06). pp. 1297–1304. doi.org/10.1109/CVPR.2006.122

Manduchi, R., Castano, A., Talukder, A., Matthies, L., 2005. Obstacle Detection and Terrain Classification for Autonomous Off-Road Navigation. Autonomous Robots 18, 81–102. doi.org/10.1023/B:AURO.0000047286.62481.1d

Martinovic, A., Van Gool, L., 2013. Bayesian Grammar Learning for Inverse Procedural Modeling, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. Presented at the Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 201–208.

Mathias, M., Martinović, A., Weissenberg, J., Van Gool, L., 2011. Procedural 3D building reconstruction using shape grammars and detectors, in: 3D Imaging, Modeling, Processing, Visualization and Transmission (3DIMPVT), 2011 International Conference On. IEEE, pp. 304–311.

Mattausch, O., Panozzo, D., Mura, C., Sorkine-Hornung, O., Pajarola, R., 2014. Object detection and classification from large-scale cluttered indoor scans: Object detection and classification. Computer Graphics Forum 33, 11–21. doi.org/10.1111/cgf.12286

Ministerie van BZK, M. van A., 2015. Rapportage Transformatie zorgvastgoed naar Wonen - Rapport - Rijksoverheid.nl [WWW Document]. URL www.rijksoverheid.nl/documenten/rapporten/2015/06/30/transformatie-zorgvastgoed-tien-praktijkvoorbeelden (accessed 12.9.19).

Mozos, O.M., 2010. Semantic labeling of places with mobile robots. Springer.

Mozos, O.M., Stachniss, C., Burgard, W., 2005. Supervised Learning of Places from Range Data using AdaBoost, in: Proceedings of the 2005 IEEE International Conference on Robotics and Automation. Presented at the Proceedings of the 2005 IEEE International Conference on Robotics and Automation, pp. 1730–1735. doi.org/10.1109/ROBOT.2005.1570363

Müller, P., Wonka, P., Haegler, S., Ulmer, A., Van Gool, L., 2006. Procedural Modeling of Buildings, in: ACM SIGGRAPH 2006 Papers, SIGGRAPH '06. ACM, New York, NY, USA, pp. 614–623. doi.org/10.1145/1179352.1141931

Müller, P., Zeng, G., Wonka, P., Van Gool, L., 2007. Image-based procedural modeling of facades, in: ACM Transactions on Graphics (TOG). ACM, p. 85.

Mura, C., Mattausch, O., Jaspe Villanueva, A., Gobbetti, E., Pajarola, R., 2014. Automatic room detection and reconstruction in cluttered indoor

environments with complex room layouts. Computers&Graphics44,20–32. doi.org/10.1016/j.cag.2014.07.005

Mura, C., Mattausch, O., Pajarola, R., 2016. Piecewise-planar Reconstruction of Multi-room Interiors with Arbitrary Wall Arrangements. Computer Graphics Forum 35, 179–188. doi.org/10.1111/cgf.13015

Murali, S., Speciale, P., Oswald, M.R., Pollefeys, M., 2017. Indoor Scan2BIM: building information models of house interiors, in: IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS).

Newcombe, R.A., Davison, A.J., 2010. Live dense reconstruction with a single moving camera, in: Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference On. IEEE, pp. 1498–1505.

Nikoohemat, S., Diakité, A., Zlatanova, S., Vosselman, G., 2019. INDOOR 3D MODELING AND FLEXIBLE SPACE SUBDIVISION FROM POINT CLOUDS, in: ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information Sciences. pp. 285–292. doi.org/10.5194/isprs-annals-IV-2-W5-285-2019

Nikoohemat, S., Faculty of Geo Information Science and Earth Observation ITC, University of Twente, 2019. Smart Indoor Models in 3D (SIMs3D). DANS. doi.org/10.17026/dans-22n-w7s8

Nikoohemat, S, Peter, M., Elberink, S.O., Vosselman, G., 2017. Exploiting Indoor Mobile Laser Scanner Trajectories for Semantic Interpretation of Point Clouds. ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences 4.

Nikoohemat, S., Peter, M., Oude Elberink, S., Vosselman, G., 2018a. Semantic Interpretation of Mobile Laser Scanner Point Clouds in Indoor Scenes Using Trajectories. Remote Sensing 10, 1754. doi.org/10.3390/rs10111754

Nikoohemat, S., Koeva, M., Oude Elberink, S.J., Lemmen, C.H.J., 2018b. CHANGE DETECTION FROM POINT CLOUDS TO SUPPORT INDOOR 3D CADASTRE. International Archives of the Photogrammetry, Remote Sensing & Spatial Information Sciences. doi.org/10.5194/isprs-archives-XLII-4-451-2018

Ochmann, S., Vock, R., Klein, R., 2019. Automatic reconstruction of fully volumetric 3D building models from oriented point clouds. ISPRS Journal of Photogrammetry and Remote Sensing 151, 251–262. doi.org/10.1016/j.isprsjprs.2019.03.017

Ochmann, S., Vock, R., Wessel, R., Klein, R., 2016. Automatic reconstruction of parametric building models from indoor point clouds. Computers & Graphics, Special Issue on CAD/Graphics 2015 54, 94–103. doi.org/10.1016/j.cag.2015.07.008

Oesau, S., Lafarge, F., Alliez, P., 2014. Indoor scene reconstruction using feature sensitive primitive extraction and graph-cut. ISPRS Journal of Photogrammetry and Remote Sensing 90, 68–82. doi.org/10.1016/j.isprsjprs.2014.02.004

Oosterom, P., 2018. Best Practices 3D Cadastres. International Federation of Surveyors (FIG), Copenhagen, Denmark.

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.

Oldfield, J., van Oosterom, P., Beetz, J., Krijnen, T.F., 2017. Working with Open BIM Standards to Source Legal Spaces for a 3D Cadastre. ISPRS International Journal of Geo-Information 6, 351. doi.org/10.3390/ijgi6110351

Oßwald, S., Gutmann, J., Hornung, A., Bennewitz, M., 2011. From 3D point clouds to climbing stairs: A comparison of plane segmentation approaches for humanoids, in: 2011 11th IEEE-RAS International Conference on Humanoid Robots. pp. 93–98. doi.org/10.1109/Humanoids.2011.6100836

Oude Elberink, S.J., S.O., 2009. Target graph matching for building reconstruction. Int. Arch. Photogramm. Remote Sens. Spat. Inf. Sci 38, 49–54.

Peter, M., 2017. Modelling of Indoor Environments Using Lindenmayer Systems. International Archives of the Photogrammetry, Remote Sensing & Spatial Information Sciences 42.

Pinheiro, P., Cardozo, E., Wainer, J., Rohmer, E., 2015. Cleaning Task Planning for an Autonomous Robot in Indoor Places with Multiples Rooms. International Journal of Machine Learning and Computing 5, 86–90. doi.org/10.7763/IJMLC.2015.V5.488

Previtali, M., Barazzetti, L., Brumana, R., Scaioni, M., 2014. Towards automatic indoor reconstruction of cluttered building rooms from point clouds. ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information Sciences 1, 281–288.

Pu, S., Vosselman, G., 2009a. Knowledge based reconstruction of building models from terrestrial laser scanning data. ISPRS Journal of Photogrammetry and Remote Sensing 64, 575–584.

Pu, S., Vosselman, G., 2009b. Building facade reconstruction by fusing terrestrial laser points and images. Sensors 9, 4525–4542.

Pu, S., Zlatanova, S., 2006. Integration of GIS and CAD at DBMS level, in: Proceedings of UDMS. pp. 9–61.

Qi, C.R., Su, H., Mo, K., Guibas, L.J., 2016. PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation. arXiv:1612.00593 [cs].

Quintana, B., Prieto, S.A., Adán, A., Bosché, F., 2018. Door detection in 3D coloured point clouds of indoor environments. Automation inConstruction85,146–166. doi.org/10.1016/j.autcon.2017.10.016

Rabbani, T., Dijkman, S., van den Heuvel, F., Vosselman, G., 2007. An integrated approach for modelling and global registration of point clouds. ISPRS Journal of Photogrammetry and Remote Sensing 61, 355–370. doi.org/10.1016/j.isprsjprs.2006.09.006

Rajabifard, A., AGUNBIADE, M., KALANTARI, M., Bin, D.M.N., AIEN, A., OLFAT, H., SHOJAEI, D., ANARAKI, M.R., 2018. An LADM-based Approach for Developing and Implementing a National 3D Cadastre – A Case Study of Malaysia 20.

Rusu, R.B., 2013a. Identifying and Opening Doors, in: Semantic 3D Object Maps for Everyday Robot Manipulation, Springer Tracts in Advanced Robotics. Springer Berlin Heidelberg, pp. 161–175. doi.org/10.1007/978-3-642-35479-3_11

Rusu, R.B., 2013b. Table Cleaning in Dynamic Environments, in: Semantic 3D Object Maps for Everyday Robot Manipulation, Springer Tracts in Advanced Robotics. Springer Berlin Heidelberg, pp. 149–159. doi.org/10.1007/978-3-642-35479-3_10

Rusu, R.B., Marton, Z.C., Blodow, N., Holzbach, A., Beetz, M., 2009. Model-based and learned semantic object labeling in 3D point cloud maps of kitchen environments, in: IEEE/RSJ International Conference on Intelligent Robots and Systems, 2009. IROS 2009. pp. 3601–3608. doi.org/10.1109/IROS.2009.5354759

Sanchez, V., Zakhor, A., 2012. Planar 3D modeling of building interiors from point cloud data, in: 2012 19th IEEE International Conference on Image Processing (ICIP).pp. 1777–1780. doi.org/10.1109/ICIP.2012.6467225

Schmittwilken, J., Saatkamp, J., Forstner, W., Kolbe, T.H., Plumer, L., 2007. A semantic model of stairs in building collars. PHOTOGRAMMETRIE FERNERKUNDUNG GEOINFORMATION 2007, 415.

Schnabel, R., Wahl, R., Klein, R., 2007. Efficient RANSAC for Point-Cloud Shape Detection. Computer Graphics Forum 26, 214–226. doi.org/10.1111/j.1467-8659.2007.01016.x

Schwarz, M., Müller, P., 2015. Advanced Procedural Modeling of Architecture. ACM Trans. Graph. 34, 107:1–107:12. doi.org/10.1145/2766956

Silberman, N., Fergus, R., 2011. Indoor scene segmentation using a structured light sensor, in: 2011 IEEE International Conference on Computer Vision Workshops (ICCV Workshops). pp. 601–608. doi.org/10.1109/ICCVW.2011.6130298

Staats, B.R., Diakité, A.A., Voûte, R.L., Zlatanova, S., 2017. Automatic generation of indoor navigable space using a point cloud and its scanner trajectory. ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences 4, 393. doi.org/10.5194/isprs-annals-IV-2-W4-393-2017

Steudler, D., 2006. Swiss cadastral core data model—experiences of the last 15 years. Computers, environment and urban systems 30, 600–613.

Stiny, G., 1982. Spatial Relations and Grammars. Environ Plann B Plann Des 9, 113–114. doi.org/10.1068/b090113

Stiny, G., Gips, J., 1971. Shape Grammars and the Generative Specification of Painting and Sculpture., in: IFIP Congress (2).

Stiny, G., Mitchell, W.J., others, 1978. The palladian grammar. Environment and planning B 5, 5–18.

Tang, P., Huber, D., Akinci, B., Lipman, R., Lytle, A., 2010. Automatic reconstruction of as-built building information models from laser-scanned point clouds: A review of related techniques. Automation in Construction 19, 829–843. doi.org/10.1016/j.autcon.2010.06.007

Tchapmi, L., Choy, C., Armeni, I., Gwak, J., Savarese, S., 2017. Segcloud: Semantic segmentation of 3d point clouds, in: 3D Vision (3DV), 2017 International Conference On. IEEE, pp. 537–547.

Teboul, O., Simon, L., Koutsourakis, P., Paragios, N., 2010. Segmentation of building facades using procedural shape priors. IEEE, pp. 3105–3112. doi.org/10.1109/CVPR.2010.5540068

Thompson, R., 2018. Developing an LADM Compliant Dissemination and Visualization System for 3D Spatial Units. 7th International FIG Workshop on the Land Administration Domain Model 11-13 April 2018, Zagreb, Croatia 22.

Thomson, C., Boehm, J., 2015. Automatic Geometry Generation from Point Clouds for BIM. Remote Sensing 7, 11753–11775. doi.org/10.3390/rs70911753

Tilove, R.B., Requicha, A.A., 1980. Closure of boolean operations on geometric entities. Computer-Aided Design 12, 219–220. doi.org/10.1016/0010-4485(80)90025-1

Tran, H., Khoshelham, K., Kealy, A., 2019. Geometric comparison and quality evaluation of 3D models of indoor environments. ISPRS Journal of Photogrammetry and Remote Sensing 149, 29–39. doi.org/10.1016/j.isprsjprs.2019.01.012

Tran H., Khoshelham K., Kealy A., Díaz-Vilariño L., 2019. Shape Grammar Approach to 3D Modeling of Indoor Environments Using Point Clouds. Journal of Computing in Civil Engineering 33,04018055. doi.org/10.1061/(ASCE)CP.1943-5487.0000800

Tran, H., Khoshelham, K., Kealy, A., Díaz-Vilariño, L., 2017. EXTRACTING TOPOLOGICAL RELATIONS BETWEEN INDOOR SPACES FROM POINT CLOUDS. ISPRS Annals of Photogrammetry, Remote Sensing & Spatial Information Sciences 4.

Turner, E., Cheng, P., Zakhor, A., 2015. Fast, Automated, Scalable Generation of Textured 3D Models of Indoor Environments. IEEE Journal of Selected Topics in Signal Processing 9, 409–421. doi.org/10.1109/JSTSP.2014.2381153

Turner, E., Zakhor, A., 2014. Floor plan generation and room labeling of indoor environments from laser range data, in: Proceedings of International Conference on Computer Graphics Theory and Applications.

Vetrivel, A., Gerke, M., Kerle, N., Vosselman, G., 2015. Identification of damage in buildings based on gaps in 3D point clouds from very high resolution oblique airborne images. ISPRS Journal of Photogrammetry

and Remote Sensing 105, 61–78. doi.org/10.1016/j.isprsjprs.2015.03.016

Vosselman, G., 2014. Design of an indoor mapping system using three 2D laser scanners and 6 DOF SLAM. ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences 2, 173.

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

Wang, H., Gould, S., Roller, D., 2013. Discriminative learning with latent variables for cluttered indoor scene understanding. Communications of the ACM 56, 92. doi.org/10.1145/2436256.2436276

Wang, R., Xie, L., Chen, D., 2017. Modeling Indoor Spaces Using Decomposition and Reconstruction of Structural Elements. Photogrammetric Engineering & Remote Sensing 83, 827–841. doi.org/10.14358/PERS.83.12.827

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

Wolf, D., Prankl, J., Vincze, M., 2015. Fast semantic segmentation of 3D point clouds using a dense CRF with learned parameters, in: 2015 IEEE International Conference on Robotics and Automation (ICRA). Presented at the 2015 IEEE International Conference on Robotics and Automation (ICRA), pp. 4867–4873. doi.org/10.1109/ICRA.2015.7139875

Wonka, P., Wimmer, M., Sillion, F., Ribarsky, W., 2003. Instant Architecture, in: ACM SIGGRAPH 2003 Papers, SIGGRAPH '03. ACM, New York, NY, USA, pp. 669–677. doi.org/10.1145/1201775.882324

Xiao, J., Furukawa, Y., 2014. Reconstructing the world's museums. International Journal of Computer Vision 110, 243–258.

Xiao, W., Xu, S., Elberink, S.O., Vosselman, G., 2012. Change detection of trees in urban areas using multi-temporal airborne lidar point clouds, in: Remote Sensing of the Ocean, Sea Ice, Coastal Waters, and Large Water Regions 2012. International Society for Optics and Photonics, p. 853207.

Xiong, X., Adan, A., Akinci, B., Huber, D., 2013. Automatic creation of semantically rich 3D building models from laser scanner data. Automation in Construction 31, 325–337. doi.org/10.1016/j.autcon.2012.10.006

Yang, L., Worboys, M., 2015. Generation of navigation graphs for indoor space. International Journal of Geographical Information Science29,1737–1756. doi.org/10.1080/13658816.2015.1041141

Zhang, Z., Weiss, R., Hanson, A.R., 1994. Qualitative obstacle detection, in: Computer Vision and Pattern Recognition, 1994. Proceedings

CVPR'94., 1994 IEEE Computer Society Conference On. IEEE, pp. 554–559.

Zheng, Y., Peter, M., Zhong, R., Oude Elberink, S., Zhou, Q., 2018. Space Subdivision in Indoor Mobile Laser Scanning Point Clouds Based on Scanline Analysis. Sensors 18, 1838. doi.org/10.3390/s18061838

Zlatanova, S., Rahman, A.A., Shi, W., 2004. Topological models and frameworks for 3D spatial objects. Computers & Geosciences, Multidimensional geospatial technology for the geosciences 30, 419–428. doi.org/10.1016/j.cageo.2003.06.004

Author's Biography

Shayan received his BSc. in Geomatics and Surveying in Tehran, Iran in 2006 and his MSc. in Cartography in 2014 in a joint program with Technical University of Munich (TUM), Vienna (TUW) and Dresden (TUD). Prior to his master (2006-2011), Shayan worked in a consulting engineer's company in Tehran, Iran in the field of Geomatics where he gained expertise in GIS and photogrammetry by supervising national projects. After his master, because of his interest in entrepreneurship, he started working in a startup in Munich (NavVis GmbH), where he learnt about mobile laser scanners and lidar processing. His experience at NavVis, as a leading company in laser scanning, carried him to his PhD path which has resulted in the present manuscript. In July 2015, he started as a PhD candidate at the Earth Observation Department of Faculty ITC (University of Twente). His first paper won the best paper award in Geospatial Week 2017, Wuhan, China. As part of his PhD, he spent three months as a junior researcher at GRID Department in the Faculty of Built Environment (at UNSW), Sydney, Australia, which resulted in two publications. His PhD was part of the project Smart Indoor Models in 3D (SIMs3D project) and Map4Scociety Program with the goal of addressing the research area of managing big data, and application areas crisis management, smart cities and management for buildings. Shayan has published in various conferences and journals in Remote Sensing and Photogrammetry, supervised master students and has reviewed scientific manuscripts for journals. Currently he is the co-editor of the special issue in indoor modeling and mapping for IJGI MDPI Journal.

Besides the present manuscript, he (co-) authored the following publications:

Nikoohemat, S, Peter, M., Elberink, S.O., Vosselman, G., 2017. Exploiting Indoor Mobile Laser Scanner Trajectories for Semantic Interpretation of Point Clouds. ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences 4. doi.org/10.5194/isprs-annals-IV-2-W4-355-2017

Nikoohemat, S., Peter, M., Oude Elberink, S., Vosselman, G., 2018. Semantic Interpretation of Mobile Laser Scanner Point Clouds in Indoor Scenes Using Trajectories. Remote Sensing 10, 1754. doi.org/10.3390/rs10111754

Nikoohemat, S., Koeva, M., Oude Elberink, S.J., Lemmen, C.H.J., 2018. Change Detection from Point Clouds to Support Indoor 3D Cadastre.

International Archives of the Photogrammetry, Remote Sensing & Spatial Information Sciences. doi.org/10.5194/isprs-archives-XLII-4-451-2018

Nikoohemat, S., Diakité, A., Zlatanova, S., Vosselman, G., 2019. Indoor 3D Modeling and Flexible Space Subdivision from Point Clouds, in: ISPRS Annals of Photogrammetry, Remote Sensing and Spatial Information Sciences. pp. 285–292. doi.org/10.5194/isprs-annals-IV-2-W5-285-2019

Nikoohemat, S., Faculty of Geo Information Science and Earth Observation ITC, University of Twente, 2019. Smart Indoor Models in 3D (SIMs3D). DANS. doi.org/10.17026/dans-22n-w7s8

Nikoohemat, S., Diakité, A.A., Zlatanova, S., Vosselman, G., 2020. Indoor 3D reconstruction from point clouds for optimal routing in complex buildings to support disaster management. Automation in Construction 113, 103109. doi.org/10.1016/j.autcon.2020.103109

Nikoohemat, S., Diakité, A., Lehtola V., Zlatanova, S., and Vosselman, G.: Consistency grammar for 3D indoor model checking, Transactions in GIS (submitted Feb. 2020), under review.

Nikoohemat, S., Vosselman, G., Peter, MS., 2016, 'Permanent Structure Detection in Cluttered Point Clouds from Indoor Mobile Laser Scanners (IMLS): abstract' NCG Symposium 2016, Enschede, Netherlands, 30/11/16 - 30/11/16, pp. 1s-19s.

Nikoohemat, S, Peter, MS., Vosselman, G., 2017, 'Semantic labeling of point clouds in indoor environment: abstract' Annual Symposium of the Netherlands Centre for Geodesy and Geo-Informatics(NCG) 2017, Delft, Netherlands, 2/11/17 - 2/11/17, pp. 1s-17s.

Nikoohemat, S., Diakité, A., Zlatanova, S., and Vosselman, G.: 2019, 'Smart Indoor Models in 3D for Disaster Management, poster' Geospatial World Forum (GWF) 2019, Amsterdam, Netherlands, 07-09 April 2019.

Elseicy, A., Nikoohemat, S., Peter, M., Elberink, S.O., 2018a. Space Subdivision of Indoor Mobile Laser Scanning Data Based on the Scanner Trajectory. Remote Sensing 10, 1815. doi.org/10.3390/rs10111815

Koeva, M., Nikoohemat, S., Oude Elberink, S., Morales, J., Lemmen, C., Zevenbergen, J., 2019. Towards 3D Indoor Cadastre Based on Change Detection from Point Clouds. Remote Sensing 11, 1972. doi.org/10.3390/rs11171972

Invited talks

Nikoohemat, S., Indoor 3D Reconstruction from Point Clouds, 2019, invited talk at Department of Civil, Geo and Environmental Engineering, Cartography Alumni Meeting 2019 at Technische Universität München, Germany, Sep. 2019.

Nikoohemat, S., 3D Mapping and Modeling of Complex Buildings for Disaster Management, 2019, invited talk at Department of GIS and Surveying at University of Tehran, Iran, Aug. 2019.