TUM School of Engineering and Design
Technische Universität München

Creating an information-rich digital twin of indoor environments by interpretation and fusion of image and point-cloud data

Yuandong Pan

Vollständiger Abdruck der von der School of Engineering and Design der Technischen Universität München zur Erlangung eines akademischen Grades eines

Doktors der Ingenieurwissenschaften (Dr. -Ing.)

genehmigten Dissertation.

Vorsitzender:		Prof. Dr.-Ing. Christoph Holst
Prüfer der Dissertation:	1.	Prof. Dr.-Ing. André Borrmann
	2.	Prof. Ioannis Brilakis, Ph.D., University of Cambridge

Die Dissertation wurde am 13.01.2023 bei der Technischen Universität München eingereicht und durch die School of Engineering and Design am 07.03.2023 angenommen.

Zusammenfassung

Digitale Zwillinge, die aus dem Fertigungssektor stammen, haben begonnen, in gebaute Infrastrukturen vorzudringen, da sie allen Beteiligten kontinuierlich erhebliche Vorteile bieten können. In dieser Dissertation wird ein digitaler Zwilling als eine zweckgerichtete digitale Darstellung von Vermögenswerten oder Systemen definiert, die durch regelmäßig übertragene Daten mit den entsprechenden physischen Anlagen oder Systemen verbunden ist. Ein informationsreicher digitaler Zwilling einer Innenraumumgebung bezieht sich auf häufig aktualisierte digitale Modelle der Gebäude, die nicht nur große raumbegrenzende Elemente, sondern auch kleine Elemente umfassen. Die zugehörigen Informationen umfassen geometrische Informationen, Kategorieinformationen und entsprechende Textinformationen.

Es ist von Vorteil, über einen informationsreichen digitalen Zwilling eines Gebäudes zu verfügen, der den aktuellen Zustand des Objekts darstellt. Der Prozess der Erstellung eines solchen digitalen Zwillings ist jedoch extrem zeitaufwändig und erfordert einen sehr hohen menschlichen Aufwand, so dass die in die Erstellung eines digitalen Zwillings investierten Kosten den potenziellen Nutzen, den er bieten kann, übersteigen. Diese Arbeit zielt darauf ab, den menschlichen Aufwand in diesem Prozess zu reduzieren, indem Methoden vorgeschlagen werden, die modernste Deep-Learning-Technologien zur Automatisierung des Prozesses anwenden.

Der vorgeschlagene Gesamtansatz verwendet lasergestützte Punktwolken und Bilder als Eingabe. Er beginnt mit der Segmentierung einer Punktwolke eines mehrstöckigen Gebäudes in einzelne Stockwerke. Anschließend werden zwei verschiedene Ansätze für Gebäude vorgeschlagen, je nachdem, ob sie die Manhattan-Welt-Annahme erfüllen oder nicht. Beide Ansätze nutzen die semantischen Informationen, die durch Deep Learning aus Punktwolken extrahiert werden. Die für Manhattan-World-Gebäude konzipierte Void-Growing-Methode beginnt mit der Extraktion der Hohlräume innerhalb von Zimmern. Die raumbegrenzenden Elemente werden dann auf der Grundlage der extrahierten Leerräume extrahiert. Für Gebäude, die die Manhattan-Welt-Annahme nicht erfüllen, werden Ebenen in Punktwolken extrahiert, geschnitten und durch eine auf Energieoptimierung basierende Methode ausgewählt.

Der nächste Schritt ist die Rekonstruktion kleiner Objekte, wobei Bilder zur Verbesserung der Ergebnisse verwendet werden, da die Erkennung kleiner Objekte in Bildern besser ist als in Punktwolken. Die von der Kamera aufgenommenen Fotos können mit den von einem Laser gescannten Punktwolken durch das photogrammetrische Verfahren registriert werden, was bedeutet, dass die photogrammetrische Punktwolke als Brücke für die Zusammenführung von Daten aus verschiedenen Quellen dient. Anschließend werden moderne künstliche neuronale Netze zur Bildsegmentierung eingesetzt. Dann werden die erkannten semantischen Informationen in den Bildern mit Hilfe der Kameramatrizen aus dem photogrammetrischen Prozess auf die lasergescannten Punktwolken abgebildet. Vordefinierte geometrische Primitive werden dann an die Punktcluster mit semantischen Informationen angepasst und dem digitalen Zwilling als vereinfachte geometrische Information hinzugefügt. Bei der Modellanreicherung werden moderne Deep-Learning-

Modelle zur Texterkennung auf die Bilder angewendet. Textinformationen, wie z. B. Seriennummern von Objekten und Raumnummern, werden aus den Bildern extrahiert und dann den rekonstruierten Objekten im 3D-Raum zugeordnet.

Zusammenfassend wird in dieser Arbeit ein Gesamtansatz zur Erstellung eines informationsreichen digitalen Zwillings von Gebäuden durch die Kombination von Objekterkennungsmethoden in 3D-Punktwolken und 2D-Bildern vorgestellt, der zu einer Endausgabe führt, die große raumbegrenzende Elemente und kleine Objekte mit geometrischen Informationen, Kategorieinformationen und Textinformationen enthält.

Abstract

Digital twins, deriving from the manufacturing sector, have been starting to penetrate built environments because they can continuously provide substantial benefits to all stakeholders. In this dissertation thesis, a digital twin is defined as a purpose-driven digital representation of assets or systems, which is linked to the corresponding physical asset or system by regularly-transferred data. An information-rich digital twin of an indoor environment refers to frequently-updated digital models of the facility, which includes not merely large space-bounding elements but also small elements. The related information includes geometric information, category information, and corresponding text information.

It is beneficial to have an information-rich digital twin of a building that represents the current state of the asset. However, the process of creating such a digital twin is extremely time-consuming and requires very high human effort, which makes the cost invested in creating a digital twin exceed the potential benefits it can provide. This thesis aims to reduce the human effort in this process by proposing methods that apply state-of-the-art deep learning technologies to automate the process.

The overall proposed approach uses laser-scanned point clouds and images as input. Firstly, it starts with segmenting a point cloud of a multi-storey building into individual storeys. Subsequently, two different approaches are proposed for buildings depending on whether they do or do not fulfil the Manhattan-world assumption. Both approaches use the semantic information extracted by point cloud deep learning. The void-growing method, designed for Manhattan-world buildings, starts with extracting the void spaces inside rooms. Space-bounding elements are then extracted based on the extracted void spaces. For buildings that do not fulfil the Manhattan-world assumption, planes in point clouds are extracted, intersected, and selected by a method based on energy optimization.

The next step is reconstructing small objects, where images are used to improve the results based on the finding that recognition of small objects in images outperforms that in point clouds. The photos taken by the camera can be registered with laser-scanned point clouds by the photogrammetric process, which means the photogrammetric point cloud works as a bridge to fuse data from different sources. Subsequently, state-of-the-art artificial neural networks of image segmentation are implemented. Then the recognised semantic information in the images is mapped to the laser-scanned point clouds by the camera matrices from the photogrammetric process. Predefined geometric primitives are then fitted to the point clusters with semantic information and added to the digital twin as simplified geometric information. In model enrichment, state-of-the-art deep learning models for text detection and recognition are applied to the images. Text information, such as serial numbers of objects and room numbers, is extracted from images and then mapped to the reconstructed objects in 3D space.

In conclusion, this thesis presents an overall approach to creating an information-rich digital twin of buildings by combining the object detection methods in 3D point clouds

and 2D images, which results in the final output that contains large space-bounding elements and small objects with geometric information, object category information, and text information.

Acronyms

AEC Architecture, Engineering and Construction.

AI Artificial Intelligence.

ANNs Artificial Neural Networks.

AR Augmented Reality.

BIM Building Information Modelling.

CAD Computer-Aided Design.

CNN Convolutional Neural Network.

CRF Conditional random field.

CSG Constructive Solid Geometry.

DNN Deep Neural Network.

FPFH Fast Point Feature Histogram.

HMD Head Mounted Display.

IoT Internet of Things.

IoU Intersection over Union.

KMS Keyboard Mouse Screen.

LiDAR Light Detection And Ranging.

MEP Mechanical, Electrical, and Plumbing.

mIoU mean Intersection over Union.

MRF Markov Random Field.

MVS Multi-View Stereo.

PCA Principal Component Analysis.

R&M Repair and Maintenance.

RANSAC RANdom SAmple Consensus.

RCNN Region-based Convolutional Neural Network.

S3DIS Stanford 3D Indoor Scene.

SfM Structure from Motion.

SLAM Simultaneous Localization And Mapping.

TLS Terrestrial Laser Scanning.

TUM Technical University of Munich.

VR Virtual Reality.

Acknowledgements

I sincerely thank Prof André Borrmann for the guidance and support throughout my PhD study. It is really a pleasure to work in the Chair of Computational Modeling and Simulation. I really enjoy the working atmosphere in the group. I would also like to sincerely thank Prof Ioannis Brilakis for his guidance and help throughout the PhD study. Thank you for having me visit your lab it was a fantastic experience working with you and your team members. I also want to thank Prof Christoph Holst for accepting to chair the examination board of my PhD!

My PhD project is conducted with the support of the Institute for Advanced Study (IAS) at TUM. I would like to thank the IAS for funding the project, which made it possible for me to work with Prof Borrmann and Prof Brilakis together.

I also want to thank the colleagues I worked with in the last few years. It was a pleasure to work with the fantastic teams. And the collaboration experience conducted in the digital twinning group led by Dr Alexander Braun is a precious memory for me. Especially I also want to thank Fiona Collins and Florian Noichl. The time that we spent together and the discussions we had in the office made my whole PhD study a fantastic journey for me.

At last, I want to thank my parents, and I cannot do this without your support and encouragement throughout the process.

Contents

1 Introduction 1

1.1 Definitions of Digital Twins . . . 4

1.1.1 Digital Twins and Building Information Modelling . . . 10

1.2 Societal Problem Statement . . . 10

1.2.1 Benefits of Digital Twins in Different Applications . . . 10

1.2.2 Difficulties of Creating Digital Twins . . . 14

1.3 State of Practice . . . 15

1.4 Thesis Structure . . . 16

2 Background 17

2.1 State of the Art . . . 17

2.1.1 Technologies of Data Collection . . . 17

2.1.2 Data Processing Methods . . . 22

2.1.3 Reconsruction of Space-bounding Elements . . . 32

2.1.4 Reconstruction of Small Objects . . . 39

2.1.5 Digital Twin Visualisation . . . 40

2.2 Research Gaps . . . 45

2.3 Research Questions . . . 46

3 Proposed Solution 49

3.1 Scope of the Thesis . . . 51

3.2 Overall Solution . . . 51

3.3 Hypotheses . . . 55

4 Segmentation of multi-storey point clouds into different storeys 57

4.1 Proposed Methods and Experiments . . . 58

4.1.1 Storey with Identical Ceiling Height . . . 58

4.1.2 Storey with Varying Ceiling Heights . . . 61

4.2 Results and Conclusions . . . 65

5 Approach for Manhattan-world Environments 69

5.1 Proposed Approach . . . 69

5.1.1 Point Cloud Segmentation by Deep Learning . . . 70

5.1.2 Generating Seeds for Growing . . . 70

5.1.3 Growing from Each Seed . . . 72

5.1.4 Stopping Conditions . . . 74

5.1.5 Detect Window and Door Openings . . . 79

5.1.6 Merge Connected Cuboids . . . 83
5.1.7 Extract Space-bounding Elements . . . 83
5.1.8 Extract Windows and Doors . . . 83
5.2 Experiments and Results . . . 84
5.3 Conclusions and Discussions . . . 94

6 Approach for non-Manhattan-world environments 95
6.1 Proposed Approach . . . 95
6.1.1 Point Cloud Segmentation by Deep Learning . . . 97
6.1.2 Plane Extraction . . . 97
6.1.3 Candidate Plane Selection . . . 102
6.1.4 Energy Terms . . . 106
6.1.5 Optimisation . . . 107
6.2 Experiments and Results . . . 107
6.2.1 Parameter Study . . . 112
6.3 Conclusions and Discussions . . . 112

7 Method to Enrich Digital Twins with Small Elements 115
7.1 Proposed Approach . . . 117
7.1.1 Object Detection in Image . . . 119
7.1.2 Creating Photogrammetric Point Clouds . . . 121
7.1.3 Point Clouds Alignment . . . 122
7.1.4 Find Visible Points in Images . . . 124
7.1.5 Map 2D Semantic Information to 3D Space . . . 130
7.1.6 Find Best-fitting Labels for Points . . . 133
7.1.7 Fit Shape to Point Custers . . . 136
7.1.8 Text Detection and Recognition . . . 139
7.2 Experiments and Results . . . 140
7.2.1 Implementation . . . 140
7.2.2 Point Cloud Segmentation Result . . . 141
7.2.3 Reconstruction Result . . . 143
7.2.4 Text Recognition Result . . . 147
7.2.5 Parameter Study . . . 150
7.2.6 Discussions . . . 151
7.3 Conclusions . . . 154

8 Implementation Details and Experimental Setup 155
8.1 Implementation Details . . . 155
8.2 Data Description . . . 157
8.2.1 Point Cloud Data . . . 157
8.2.2 Image Data . . . 157
8.3 Data Structure Design for Output . . . 158

9 Results and Conclusions 161
9.1 Results . . . 161
9.2 Contributions and Limitations . . . 166

9.3 Conclusions and Future Work 167

Appendix A Parameter study on coefficients of energy terms in Section 6.2.1 171

Appendix B Parameter study on coefficients of point labels in Section 6.2.1 176

Appendix C Text recognition result in Section 7.2.4 182

Chapter 1

Introduction

This PhD thesis is about developing methods to create an information-rich digital twin of indoor environments of buildings. In this thesis, a digital twin of the building is defined as an updated digital representation of different categories of objects in the indoor environment of the building throughout its life cycle. An information-rich digital twin refers to a digital twin with more valuable information. Specifically speaking, the final created information-rich digital twin in the thesis contains the following:

- detailed and simplified geometric information for various categories of objects in buildings, from large space-bounding elements (like ceilings, walls, etc.) to small-scale elements (such as Mechanical, Electrical, and Plumbing (MEP) elements).
- semantic information, including object categories and corresponding texts (for example, text on door signs and object IDs).

The concept of the digital twin, deriving from the manufacturing sector, is usually described as consisting of three parts: a physical entity part, a digital representation part, and a linking between them. The number of publications in the engineering domain with the key term "digital twin" in the last 20 years is illustrated in Figure 1.1. It is obvious to see that the number of publications almost increases every year, and especially exploded in the last five years. If looking into different branches in the engineering domain, the increasing tendency stays the same, as shown in Figure 1.2. However, there is not any commonly-agreed definition of digital twins in the built environment, despite the fact that digital twins have been attracting increasing attention and investigation in academia.

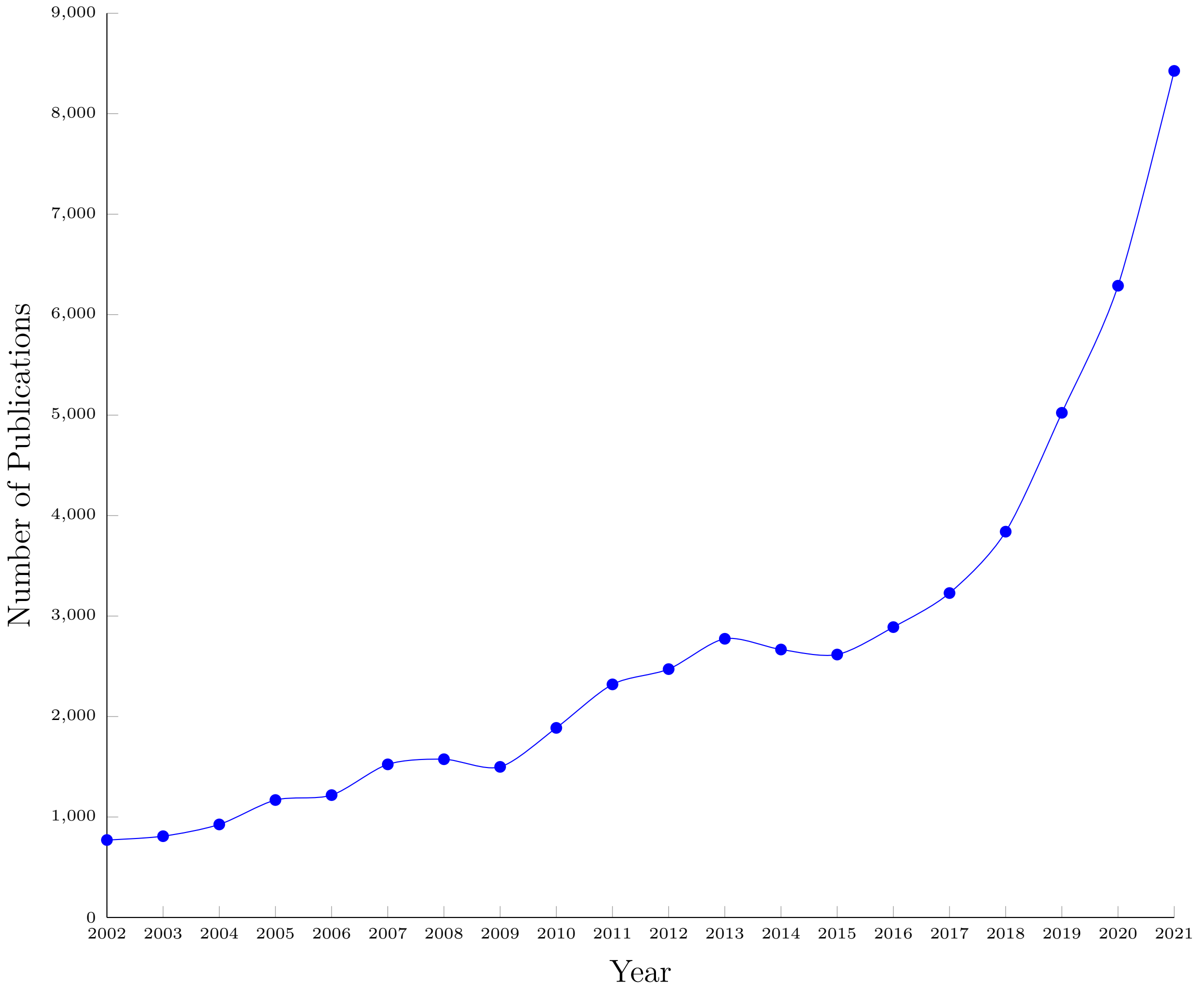


Figure 1.1: Number of publications in engineering in last 20 years with keyword "digital twin" (data source: https://app.dimensions.ai)

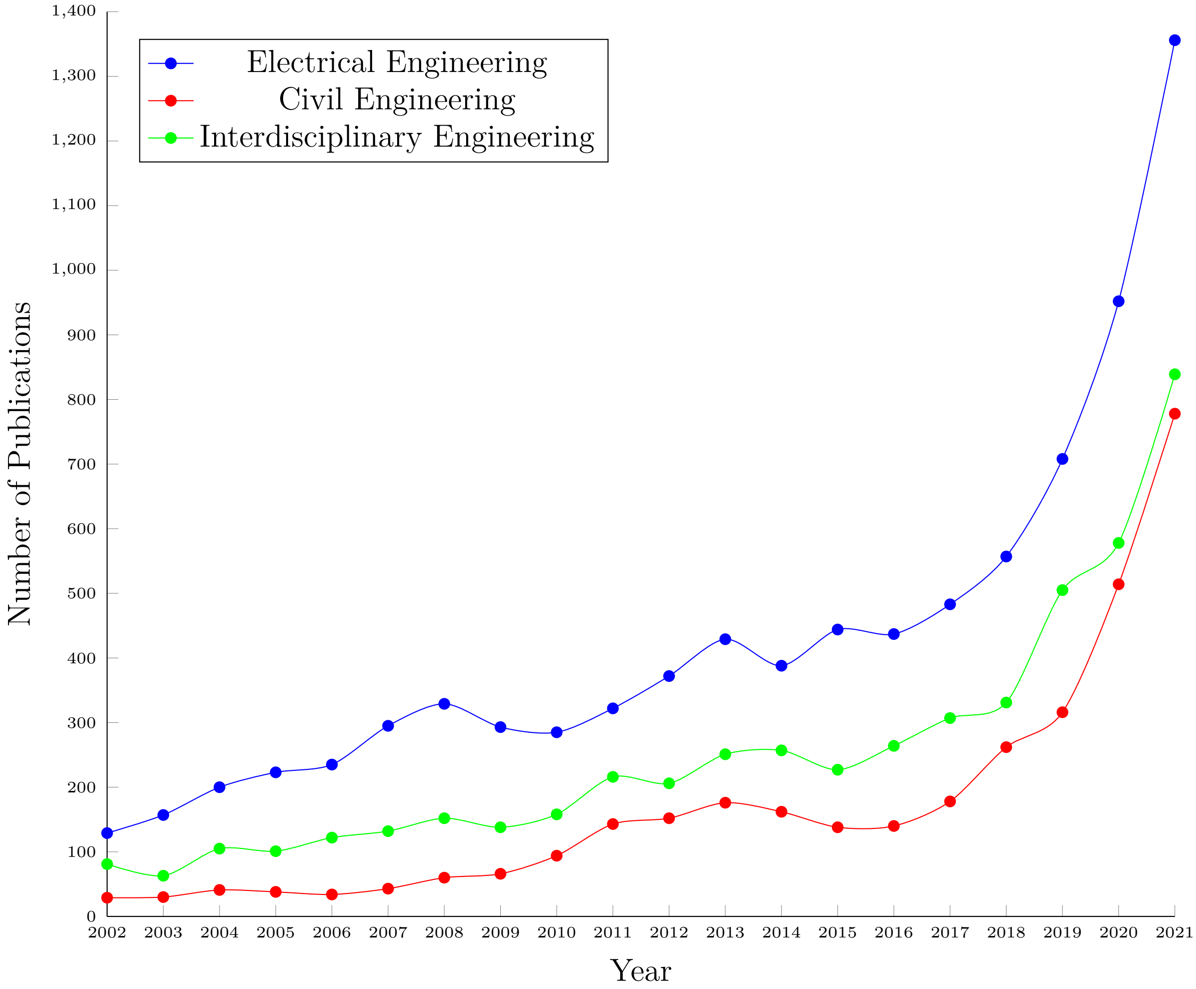


Figure 1.2: Number of publications in different engineering sectors in last 20 years with keyword "digital twin" (data source: https://app.dimensions.ai)

1.1 Definitions of Digital Twins

Despite the increasing attention to digital twins, digital twins probably refer to different concepts in different sectors. Meanwhile, even in the same field, there is also no guarantee that researchers are discussing the "same" digital twin concept in their research. The reason behind this is that the digital twin concept is comprehensive, which makes its definition manifold. Some representative definitions are selected and listed in Table 1.1. Almost all these definitions mention that three parts are essential: the physical model, the digital representation, and the linking between these two parts. Apart from the three key components, these definitions differentiate a bit in the physical model types, applied domains, potential use cases, typical properties, etc. Therefore, it is hard to make a unified definition that can cover or fit all the mentioned aspects in these definitions.

author	definition
Glaessgen and Stargel (2012)	a digital twin is an integrated multiphysics, multiscale, probabilistic simulation of an as-built vehicle or system that uses the best available physical models, sensor updates, fleet history, etc., to mirror the life of its corresponding flying twin.
Lee et al. (2015)	coupled model of the real machine that operates in the cloud platform and simulates the health condition with an integrated knowledge from both data-driven analytical algorithms as well as other available physical knowledge.
Zhuang et al. (2017)	a digital twin refers to the process and method of describing and modelling the characteristics, behaviour, formation process, and performance of physical entity objects using digital technology.
Alam and El Saddik (2017)	a digital twin is an exact cyber copy of a physical system that truly represents all of its functionalities, which can be used for monitoring, diagnostics, and prognostics purposes.
Grieves and Vickers (2017)	the digital twin is a set of virtual information constructs that fully describes a potential or actual physical manufactured product from the micro atomic level to the macro geometrical level. at its optimum, any information that could be obtained from inspecting a physically manufactured product can be obtained from its digital twin.
Söderberg et al. (2017)	using a digital copy of the physical system to perform real-time optimization.
Bachiega (2017)	a digital twin is a real-time digital replica of a physical device.
Tao et al. (2018)	digital twin is a real mapping of all components in the product life cycle using physical data, virtual data, and interaction data between them.
Bolton et al. (2018)	a dynamic virtual representation of a physical object or system across its life cycle, using real-time data to enable understanding, learning, and reasoning.
El Saddik (2018)	a digital twin is a digital replica of a living or non-living physical entity. By bridging the physical and the virtual world, data is transmitted seamlessly, allowing the virtual entity to exist simultaneously with the physical entity.

Bolton et al. (2018)	in the context of digital built Britain, a digital twin is a realistic digital representation of assets, processes or systems in the built or natural environment.
Ayani et al. (2018)	a digital twin is a multi-physics and multi-scale simulation model reflecting the corresponding physical model, with emphasis on the promotion of DT high-fidelity simulation to the industry.
Brilakis et al. (2019)	a digital twin is a digital replica of a physical built asset. what a digital twin should contain and how it represents the physical asset are determined by its purpose. It should be updated regularly in order to represent the current condition of the physical asset.
Liu et al. (2019)	digital twins refer to virtual objects or a set of virtual things defined in the digital virtual space, which has a mapping relationship with real things in the physical space.
Leng et al. (2019)	a digital twin is an exact and real-time cyber copy of a physical manufacturing system that truly represents all of its functionalities.
Borth et al. (2019)	a digital twin is a connected and synchronised digital replica of physical assets which represent both the elements and the dynamics of how systems and devices operate within their environment and live throughout their life cycle.
Sacks et al. (2020)	digital twin construction is a new mode for managing production in construction that leverages the data streaming from a variety of site monitoring technologies and artificially intelligent functions to provide accurate status information and to proactively analyse and optimize ongoing design, planning, and production.
Aheleroff et al. (2021)	a digital twin is a digital replica of a physical entity with the two-way dynamic mapping between a physical object and its digital model, which has a structure of connected elements and meta-information.
Fotland et al. (2020)	a digital twin is a digital copy of a physical asset, collecting real-time data from the asset and deriving information not being measured directly in the hardware.
Liu et al. (2021)	a digital twin is a digital entity that reflects physical entity's behaviour rule and keeps updating through the whole life cycle

Pan and Zhang (2021)	digital twins refer to a mirror and digital depiction of the actual production process, which can imitate all aspects of physical processes under the integration of physical products, virtual products, and relevant connection data.
VanDerHorn and Mahadevan (2021)	a digital twin is a simulation process that makes full use of physical models, sensor updates, operation and maintenance history and other data to integrate multi-disciplines, multi-physical quantities, multi-scales, and multi-probabilities
Kamble et al. (2022)	a digital twin is a digital counterpart of the physical systems based on a simulation that deals with design systems and optimizes them for improved efficiency.

Table 1.1: Different definitions of digital twins

Even if a comprehensive definition of digital twins that tries to include many aspects from a different perspective is proposed, it is still doubtful whether it is realistic to use the same definition in real use cases. It is almost impossible to create a digital twin that covers all those potential aspects in the real world, which covers information from small-scale components to larger ones, from objects to processes, from geometric representation to semantic information, from object relations to their interactions, from real current state to simulation results, etc. Therefore, the author states that one important property that should be added to the definition of digital twins is "purpose-driven".

A purpose-driven digital twin means the digital twin should be defined according to the purpose. Every time when we dive into the topic of digital twins, an essential point that needs to be clarified is what purpose the digital twin is supposed to achieve. In other words, a digital twin is not clearly defined without defining its purpose in advance. Depending on differentiated purposes, the extremely broad concept of digital twins can be scaled down in order to make a clear definition of the digital twin. Only when the definition is clear enough to clarify the scope of the problem does it make sense to work on the following topics, such as the contents that should be included, potential technologies that could be applied, etc.

For example, suppose we want to create a digital twin whose purpose is indoor navigation. In that case, the possible definition could be "an updated digital indoor 3D model that represents the current geometry of indoor space and contains important semantic information that is valuable for navigation". Meanwhile, as the digital twin is designed for indoor navigation, only reconstructing large-scale elements of the indoor environment (like walls, floors, staircases, etc.) should be enough. Extra work to reconstruct smaller elements is probably not necessary in this case. In comparison with that, if we want to build a digital twin that helps asset maintenance, these small scale-elements are also important based on the fact that in the Repair and Maintenance (R&M) activities of a facility, MEP costs usually constitute the largest share of total costs (Adán et al., 2018). Therefore, a digital twin for facility maintenance would be more valuable if it contains those elements that are frequently required in facility management processes. Meanwhile, facility management involves more accurate data about floor plans, space utilization, asset location, and technical plants (D'Urso, 2011). Text information such as room numbers and serial numbers (IDs) next to assets that can identify the corresponding assets is very beneficial, especially when managing large facilities. These objects should be linked with their object IDs or serial numbers if possible. In this case, a digital twin for facility maintenance should be a regularly-updated digital 3D model of the facility that contains the geometric and semantic information.

Besides "purpose-driven", another essential property of digital twins is to be regularly updated throughout the life cycle of the physical asset. There are three terms that need to be clarified. The term "updated" means that there must be data transfer between the physical and digital parts, which is mentioned in all definitions of digital twins. The term "regularly" indicates the updating frequency, which is determined by many factors like the physical asset type, user requirements, cost budget, etc. For a digital twin of roads that is used for autonomous driving, the updating frequency should be much higher than the digital twin to maintain the relevant roads. In any case, the digital twin should be updated regularly to fulfil the corresponding requirements. "Life cycle" indicates the time range that digital twins should be operated. For example, a digital twin for a building should be created and updated in all the phases, which include design, construction, operation, demolition, and waste treatment (Kotaji et al., 2003).

In summary, I define the digital twin in my thesis as follows:

> The digital twin is a purpose-driven digital representation of assets or systems, which is linked to the corresponding physical asset or system by regularly-transferred data. More specifically, the digital twin of buildings refers to regularly-updated 3D digital geometry representations of the facility that includes both large-scale structural elements and small-scale elements, which are further enriched with semantic information.

The individual digital representation at every updating time that represents the condition of the facility at the corresponding time is called "digital twin status" in this thesis. While a digital twin is supposed to represent a facility throughout its life cycle, it consists of multiple digital twin statuses at an updated frequency throughout the time, which is illustrated in Figure 1.3. It should be noted that geometry is an essential and basic constitution of a digital twin of buildings upon which other information can be built. In this case, a Building Information Modelling (BIM) model, which contains basic geometric and semantic information, can be seen as a starting point for a digital twin. More comparisons with regard to BIM and digital twins will be discussed in Section 1.1.1. Apart from the basic geometric and semantic information, the digital twin can also be further enriched by other information (for example, text information) to make it an information-rich digital twin. As mentioned above, the digital twin definition is purpose-driven, which makes its content depend on the selected use cases. This thesis focuses on the information-rich digital twin that is defined in this paragraph.

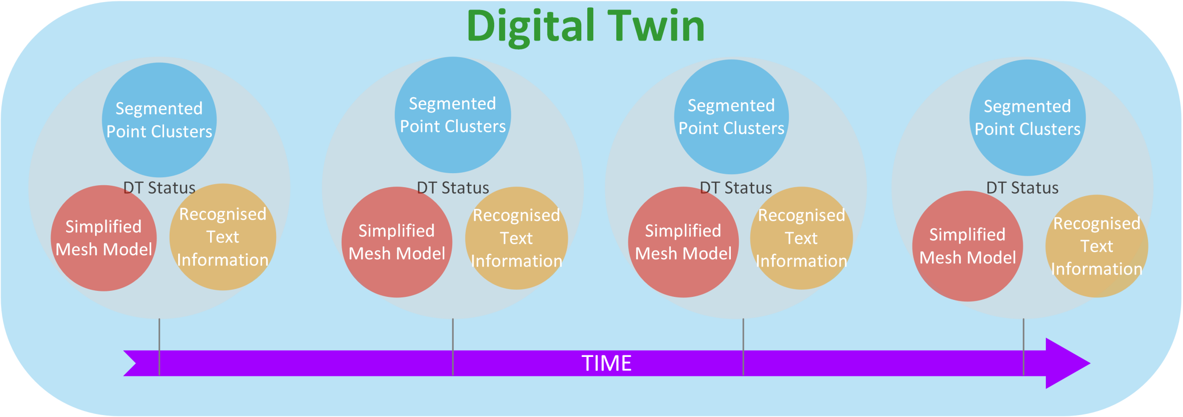


Figure 1.3: Digital twin consists of its individual status through life cycle

1.1.1 Digital Twins and Building Information Modelling

When talking about digitisation in the Architecture, Engineering and Construction (AEC) domain, one term cannot be ignored: Building Information Modelling (BIM), which has strong relationships to the Digital Twin concept. There is not a commonly-agreed answer to this question of "what is the difference between BIM and digital twins" either.

Most researchers agree that a BIM model fulfils the definition of a digital model according to the categorisation of Kritzinger et al. (2018): "A Digital Model is a digital representation of an existing or planned physical object that does not use any form of automated data exchange between the physical object and the digital object." While the BIM methodology covers the design, construction and operation phases of a facility, digital twins often include broader concepts that can focus on very large-scale facilities and integrates information from other sectors. It can link the facility itself with many other sectors or systems, such as water systems, waste systems, power systems, etc. A digital twin provides the possibility to achieve information exchange among different systems or sectors that we used to treat independently in the past. In addition, the notion of constantly updating the model received much less attention than for Digital Twins. As such, BIM can be seen as a concept fundamental for Digital Twins in the Built Environment. In particular, research under the heading "Scan-to-BIM" (Collins et al., 2022) (Bosché et al., 2015) (Bloch and Sacks, 2018) (Tuttas et al., 2017) is also valuable and transferable in the process of creating digital twins of built environments.

Relevant information to enrich BIM models ranges from geometric changes in the building layout over monitoring the condition of structural components (degradation) to the occupancy and usage information of individual rooms and spaces. Although the sensors to be employed differ significantly, a digital twin should be based on integrating BIM with Internet of Things (IoT) technology to allow seamless integration of various devices and the data they produce. In this way, digital twins can be seen as BIM models extended by means of capturing real-world data and feeding it back into the model, thus closing the information loop as demanded by the digital twin concept.

1.2 Societal Problem Statement

A digital twin is valuable for all stakeholders in different aspects. In this section, the benefits of having a digital twin and the difficulties of creating a digital twin are discussed.

1.2.1 Benefits of Digital Twins in Different Applications

As a valuable digital asset of the built environment, digital twins provide possibilities to help all stakeholders with a variety of use cases, including construction progress monitoring, facilities management and operation, asset condition monitoring, sustainable development, etc. Especially with regard to decision-making, digital twins benefit the whole process by providing more reliable and useful information.

In this section, different applications and the corresponding benefits of digital twins are discussed. Most applications based on digital twins can be performed throughout the related asset's life cycle. For historical assets which have been completed for many years but do not yet have any digital records, digital twins can help to start and keep a recording of their performance for better maintenance and renovation. For facilities under construction at the current state, keeping digital twins dynamic and up-to-date can perform the asset's real-time progress monitoring, quality control, diagnostics, and prognostics. In addition, digital twins can also be used in potential futures for capital investment projects before the design and construction of the facility. It is efficient to simulate the performance with its digital twin to assist decision and strategy-making in various predictive scenarios.

As discussed in Section 1.1, one important characteristic of digital twins is purpose-driven. How the physical and the digital twins are synchronized in real use cases should depend on the purpose of the digital twin, which determines the content of digital twins, like what elements and processes are digitalised, what level of detail the digital model is required, how frequent the model is supposed to be updated, etc. As the concept of digital twins is very broad, it is almost impossible to propose a precise and detailed definition of a digital twin that aims to cover everything without thinking about its purpose. Some potential applications of digital twins are discussed as follows.

Condition monitoring

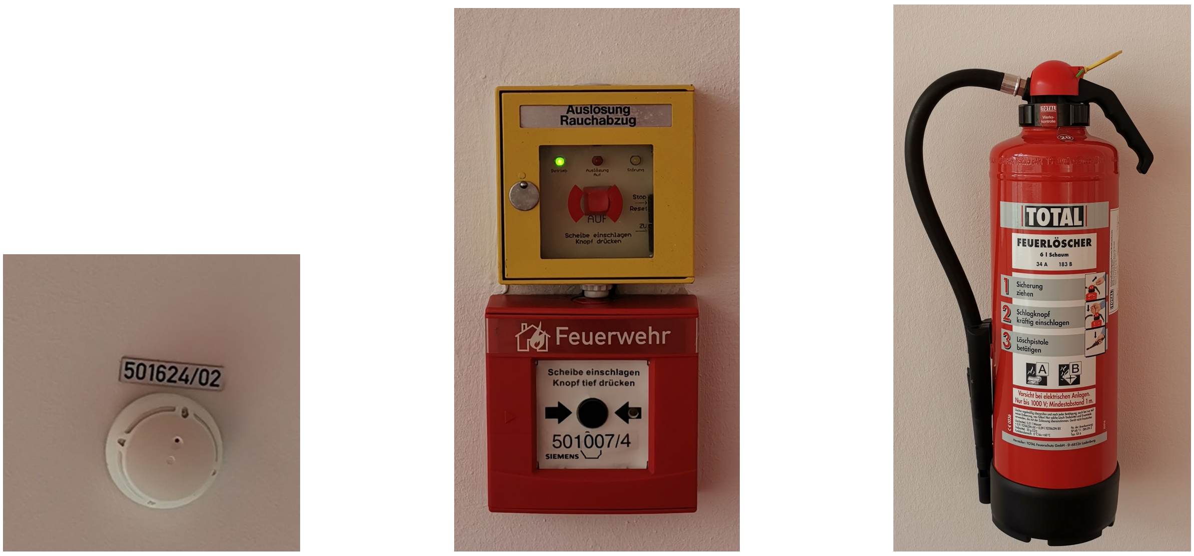
A digital twin can be used to monitor the current condition of a facility. By capturing geometric information by different sensors, the current condition can be visualized and represented by digital twins. The geometry of facilities can be monitored by comparing the current condition with previous asset conditions over time, which allows a digital twin to give maintenance suggestions to the asset holders and managers.

Apart from monitoring the geometry change of an asset, large-scale complex systems, for example, a sewer system of a city, can also be monitored by a digital twin. Predictive maintenance operations can be utilized to identify potential blockages. Some values, like the current state of flow in pipes, can be recorded and compared with the historical values, which are able to predict or locate disruption locations in the system. The predictive maintenance suggestions or alerts can be sent to facility managers to make a more suitable and quick decision by data support.

Facility management

There is a very broad spectrum of facility management, which includes but is not limited to operation management of Mechanical, electrical and plumbing (MEP) components (Hu et al., 2016) (Cheng et al., 2020), internal environment monitoring (Cao et al., 2015) (i.e. air quality), working productivity (Meerman et al., 2014), etc. With the increasing adoption of the Internet of Things (IoT) and Artificial Intelligence (AI), facility management is becoming more and more intelligent. In addition, Augmented Reality (AR) and Virtual Reality (VR) provide great potential to visualise the environment and improve efficiency (Baek et al., 2019) (Chen et al., 2020) (Chen et al., 2021) (Zhang et al., 2020).

The concept of digital twins embeds almost all of these aspects in facility management. Relevant objects and values are captured and represented in a detailed digital model by capturing devices like laser scanners and cameras. For example, it is extremely important to maintain fire safety equipment to guarantee the safe use of the facility. Some common fire safety equipment is shown in Figure 1.4. In facility management, it is very helpful to efficiently locate and reconstruct these devices in 3D space, especially in the management of large facilities. In particular, only recognising the object categories is not sufficient. It is necessary to identify the object to the "instance" level by linking the object ID or serial number to its position. One example of the object IDs or serial numbers is shown in Figure 1.4a, which are common in buildings nowadays.



(a) Smoke alarm

(b) Fire alarm switch

(c) Fire extinguisher

Figure 1.4: Fire safety equipment

Apart from the information that can be extracted with visual information, by applying various IoT sensors such as thermometers, hygrometers, and carbon dioxide sensors, different values (like temperature, humidity, and carbon dioxide level) that represent internal environment conditions can be recorded and then updated in the digital model regularly. AI-relevant technologies can be used to help the process of creating the initial model as well as updating the model throughout a facility's life cycle. Facility managers can check the visually assistive information provided by AR and VR devices, which is capable of lightening their workload. The digital-twin-centred concept is illustrated in Figure 1.5.

Data are continuously accumulated throughout the life cycle of the asset in the update process. But it needs to be noted that more data cannot always guarantee a better representation of the facility. If we use the term "usability" to describe how accurate and useful the data in a digital twin, it needs to be noticed that although the data are

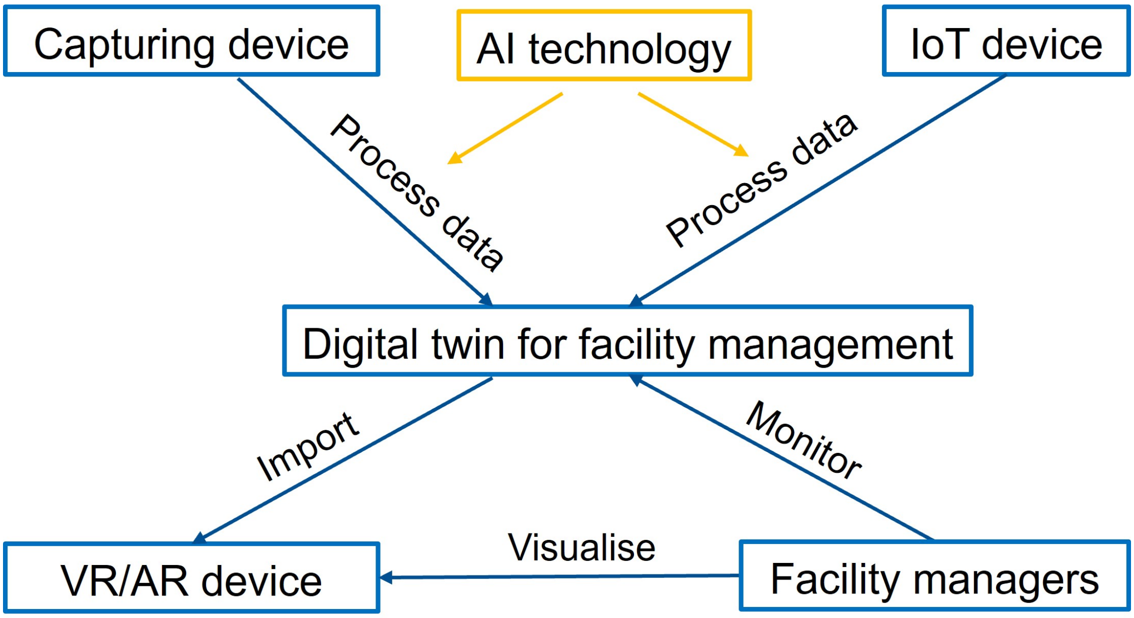


Figure 1.5: Digital twin for facility management

continuously added to the digital twin throughout the life cycle of a facility, the usability of the digital twin would start to drop while the data are starting to be outdated. Subsequently, usability starts to restore when the newly updated data flows into the digital twin. This concept is illustrated in Figure 1.6.

In the process of creating digital twins, raw data is firstly captured by various sensors, which makes the data flow from physical assets to a digital representation. The usability status of digital twins is proceeded by accumulating the amount of data. Subsequently, pre-processing technologies like denoising and registering are performed, which make the data more mature by increasing its usability of data. Then data is further processed by technologies like image segmentation, point cloud segmentation, geometry extraction, etc. Semantic and geometric information is added to the digital twin. At last, depending on the requirements of different use cases, relevant information that helps in the corresponding application is attached to the digital twin, which makes the digital twin reach the required usability level of the use case.

Environment simulation

Designers and engineers can use digital twins in the renovation phase of a project. Different scenarios can be simulated in a digital twin without modifying the real asset, such as natural light design, artificial lighting, heating simulation, and so forth. By only modifying facilities in the digital twin, how these changes would impact the above factors without implementing the modifications in the real world can be predicted. Similarly, by means of VR/AR devices, designers and customers can make use of the digital twin to visualize their own designs and show these changes and modifications (like lighting), which benefits the decision-making of renovation and makes communication between designers and clients much easier. For instance, different lighting atmospheres

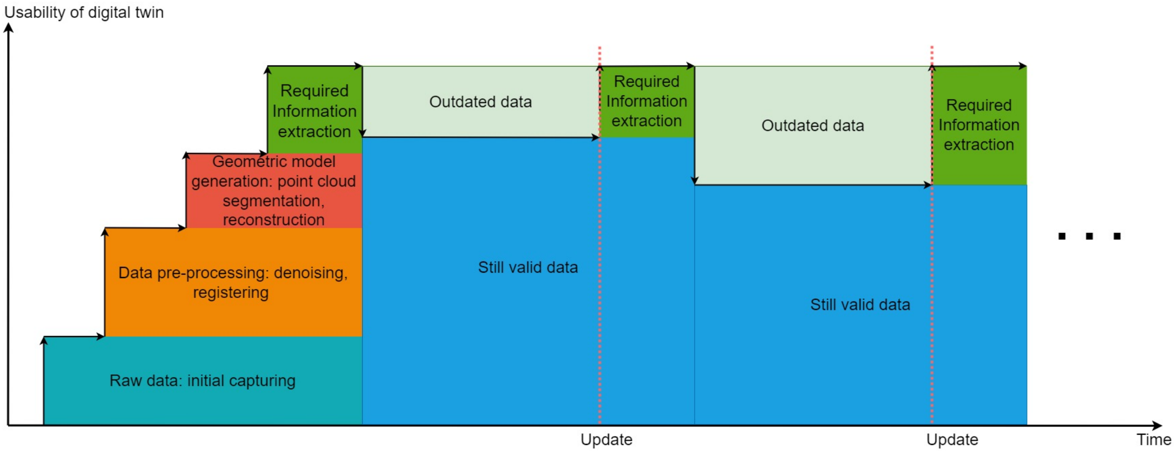

Usability of digital twin
Required Information extraction
Outdated data
Required Information extraction
Outdated data
Required Information extraction
Geometric model generation: point cloud segmentation, reconstruction
Data pre-processing: denoising, registering
Still valid data
Still valid data
Raw data: initial capturing
. . .
Update
Update
Time

Figure 1.6: Concept of usability changing for digital twin of facility management over time

are visualized, helping designers aesthetically assess the design and present the outcomes of the setup to their clients. This can be seen as a very basic use case of a digital twin to simulate the environment (Natephra et al., 2017).

1.2.2 Difficulties of Creating Digital Twins

Despite an information-rich digital twin being valuable for building management and maintenance, few existing assets have one. There are mainly two reasons for this situation. The first one is that many buildings were constructed decades ago. There were no authoring tools to construct a 3D digital model or the concept of a digital twin when they were built. The other reason is that even though some new buildings have a digital design model, this model was not updated when the asset was modified throughout the lifecycle of the asset. Therefore, most buildings do not have any valuable digital twins.

However, the process of creating a valuable digital twin requires huge human effort. Creating an information-rich digital twin of existing assets is a process that consists of the following steps: a) capturing raw visual and spatial data in the form of RGB images and laser-scanned point clouds; (2) detecting geometric objects and geometric relationships of objects in the raw data. Step 1 of this process is significantly more automated than step 2 and requires much fewer labour hours (Agapaki and Brilakis, 2021). The cost and effort needed to complete step 2 for most assets appear to counteract the perceived value of the resulting digital twin. Existing capturing technologies, such as laser scanning and photogrammetry, make it possible to efficiently collect point clouds that contain geometric information about the as-is state in the built environment. Compared with acquisition time, it is much longer to extract and reconstruct a basic 3D model that contains components like columns, slabs, and walls from the input point cloud. The potential benefits of having a digital twin and the difficulties of creating one make this domain need solutions to reduce the cost in this process.

1.3 State of Practice

As discussed in the previous section 1.2.2, the process of creating a digital twin consists of two steps, capturing raw data and detecting geometric objects with relationships. The second step can be broken down into the detection of large objects (such as ceilings, floors, and walls) and small objects (such as fire extinguishers and smoke alarms) by their scale.

Some leading 3D Computer-Aided Design (CAD) vendors (like Autodesk, Bentley, and ClearEdge3D) have developed software that has a variety of 3D modelling features, which enable modelling large objects from point cloud data. According to the author's experience, it took around two hours to collect the point cloud in a working area of 20 rooms by using a NAVVIS scanner (https://www.navvis.com/). But one junior modeller spent more than 80 hours modelling basic elements (ceilings, floors, walls, doors, windows, stairs) in this area using Autodesk Revit software. In (Lu and Brilakis, 2019), authors captured 10 bridges by the laser scanner and counted the as-is modelling time in Autodesk Revit. The average as-is modelling time is around 28 hours, while the corresponding capturing time is 2.82 hours. In (Agapaki et al., 2018), authors state that 64% of person-hour savings can be achieved by using the state-of-the-art, semi-automated commercial software EdgeWise (https://www.clearedge3d.com/edgewise/). But 2,382 manual labour hours are still needed to model an example of a small petrochemical plant with 240,687 objects and 53,834 pipes. In summary, despite the fact that some commercial software solutions can reduce the modelling time, it is still much higher than the time for capturing, which makes the cost and effort to generate a digital model exceed its benefits manually. Therefore, researchers are trying to automate the process of digital twinning in built environments in order to reduce human effort.

Apart from large-scale elements, small objects, referring to elements that are smaller in scale in comparison with structural elements (like walls, floors, and ceilings), are important in facility maintenance and management, such as fire safety equipment (smoke alarms, fire extinguishers, fire alarm switches), electrical elements (light switches, light fixtures, speakers), etc. In the Repair and Maintenance (R&M) activities of a building, Mechanical, Electrical, and Plumbing (MEP) costs usually constitute the largest share of total costs (Adán et al., 2018). Therefore, a building twin would be more valuable if it were to contain those elements that are frequently required in facility management processes. In addition, facility management involves more accurate data about floor plans, space utilization, asset location, and technical plants (D'Urso, 2011). Text information such as room numbers and serial numbers (IDs) next to assets that can identify the corresponding assets (as shown in Figure 7.1) is very helpful, especially when managing large facilities. These IDs exactly represent the corresponding object instances in an asset and make the link between physical assets and digital twins much clearer. Therefore, it is valuable to add the information to an enriched digital twin of buildings. Unfortunately, this work is currently mostly manual work, and the author cannot find software solutions to automate the process of enriching digital twins with small objects.

1.4 Thesis Structure

In the thesis, in order to reduce the manual work in the process of creating digital twins from collected raw data, the author presents a pipeline that automates the process. More specifically, the thesis presents methods to extract relevant information from raw data and subsequently generate a detailed 3D geometric model that is enriched with semantic information (including text information and object IDs). An information-rich digital twin that represents the condition of the facility at a specific time point can be created by running through the proposed pipeline. By applying the pipeline to all data captured throughout the time, the proposed pipeline provides the possibility to create a digital twin representing the facility throughout its life cycle.

The rest of this thesis is organised as follows: The background, including the literature review, is presented in Chapter 2. The overall proposed solution for creating an information-rich digital twin of an indoor environment is introduced in Chapter 3. Subsequently, the proposed methods are presented in individual chapters in detail. The method of segmenting the input point cloud of multi-storey buildings into different storeys is shown in Chapter 4. Then the proposed approach for Manhattan-world indoor environments is described in Chapter 5 (the Manhattan-world assumption states that there is a predominance of a triple of mutually orthogonal directions in the environment (Coughlan and Yuille, 1999)). The author published this concept as a conference paper (Pan et al., 2021) and the extended method as a journal paper (Pan et al., 2023). The proposed approach for non-Manhattan-world indoor environments is described in Chapter 6. The method that creates an information-rich digital twin of small objects in indoor environments and enriches the digital twin with semantic information is presented in Chapter 7. The author published this method in a journal paper (Pan et al., 2022) and its implementation improvement in a conference paper (Pan et al., 2022). After introducing the methods in individual chapters, the overall research methodology is presented in Chapter 8. At last, the results and conclusions of the thesis are summarised in Chapter 9.

Chapter 2

Background

Digital twins have been entering the conversation in the built environment as they can continuously offer substantial value to all stakeholders. Especially the related technology boom in the last years has started to make it possible to create and utilise digital twins in practice in a much more efficient way. In this chapter, these digital-twin related technologies are introduced in Section 2.1. The research gaps with regard to creating digital twins from raw data are discussed in Section 2.2. The research questions and hypotheses are summarised in Section 2.3.

2.1 State of the Art

In this section, various technologies that are used in the process of creating and utilising digital twins are discussed. Specifically, this section is organised as follows. Data collection technologies that are widely used to capture environments in the process of creating digital twins are introduced in Section 2.1.1. Data processing methods to process captured raw data are discussed in Section 2.1.2. Previous research with regard to creating digital twins of space-bounding elements (like ceilings, floors, and walls) is introduced in Section 2.1.3. Methods with regard to reconstructing small objects in indoor environments are discussed in Section 2.1.4. Technologies that are widely used to visualise digital twins in different applications are presented in Section 2.1.5.

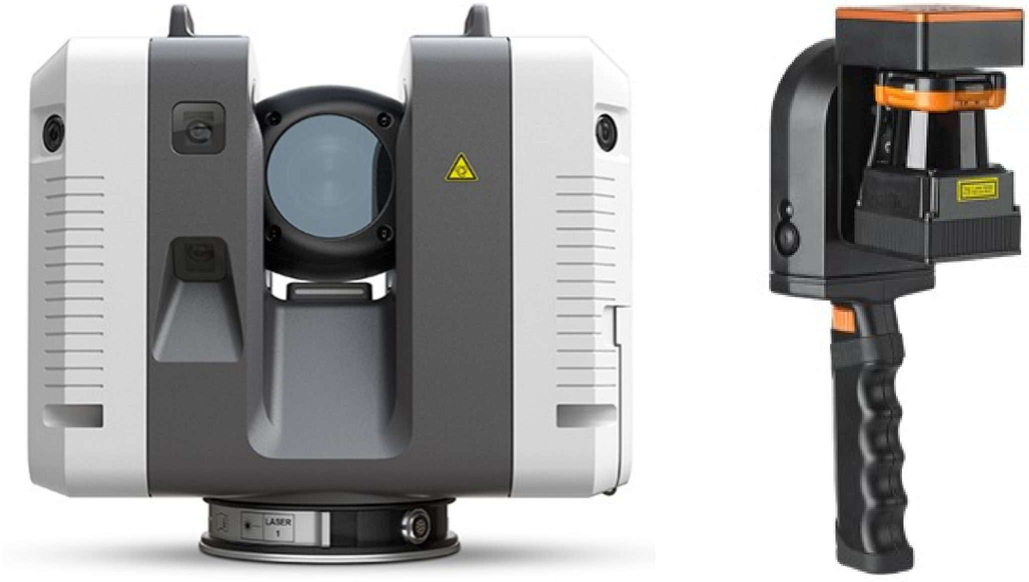
2.1.1 Technologies of Data Collection

The process of creating a digital twin starts with collecting relevant data from various devices. The collected data can be used not only to create digital twins (if no digital twins exist) but also to update digital twins (if there is one already). While different data can be collected according to different user requirements, as digital twins are "purpose-driven", geometric information of an asset is usually a basic but essential constitution of a digital twin. In this section, capturing technologies to collect raw geometric data for digital twins are introduced. Laser scanning and imaging technologies are two widely used technologies to capture the geometric information of the environments.

Laser Scanning

Terrestrial Laser Scanning (TLS), sometimes also called terrestrial Light Detection And Ranging (LiDAR) or topographic LiDAR, is a basic technique to measure the location and dimension of surfaces in 3D space by emitting laser pulses toward these surfaces of objects. The data generated by these sensors can reflect the physical surfaces of objects in the real world. However, the raw data provided by scanners is also discrete due to the discrete laser beams. The output of laser scanners is point clouds which are basically sets of points in 3D space. Each point is defined by three coordinates (x, y, z) and additional information depending on the device used, which could contain intensity, normal, colour information, etc.

Terrestrial laser scanners are often used to capture large outdoor spaces and indoor environments because they are able to measure points from a long distance with high precision. Compared with terrestrial laser scanners, which are usually heavy and need to be fixed on a tripod, mobile scanners are usually smaller. A terrestrial laser scanner and a mobile laser scanner are shown in Figure 2.1. While the capturing precision of mobile laser scanners is usually lower than that of terrestrial laser scanners, the measuring speed is much faster. Users can select different laser scanners according to their requirements, such as measuring accuracy, measuring speed, and cost.



(a) Leica RTC360 laser scanner

(b) GeoSLAM ZEB Go mobile laser scanner

Figure 2.1: Terrestrial laser scanner and mobile laser scanner

The working principle of a laser scanner can be briefly described as follows: the laser scanner emits a beam of laser onto a rotating mirror that effectively covers the surrounding environment, capturing millions of discrete data points and producing detailed 3D information about the surrounding surfaces. The distance between the surface and the device can be computed by reflecting the laser beam back to the device. In general, there are two methods for distance measurement in laser scanning technology: time of flight and phase shift measurement. In the time of flight distance measurement, the distance between the sensor and the target can be described by the following equation (Suchocki, 2020).

$$distance = \frac{c}{2} \cdot \Delta t, \tag{2.1}$$

where c is the speed of light between the device and the target, Δt is the measured time interval from emitting to receiving the light signal.

Apart from the time of flight technology, the phase shift measurement uses an amplitude-modulated continuous sinusoidal laser beam. The distance can be computed by the phase shift between the reference and return signals (Yoon et al., 2011) by the equation

$$distance = \frac{c}{2} \cdot \frac{\varphi}{2\pi f}, \tag{2.2}$$

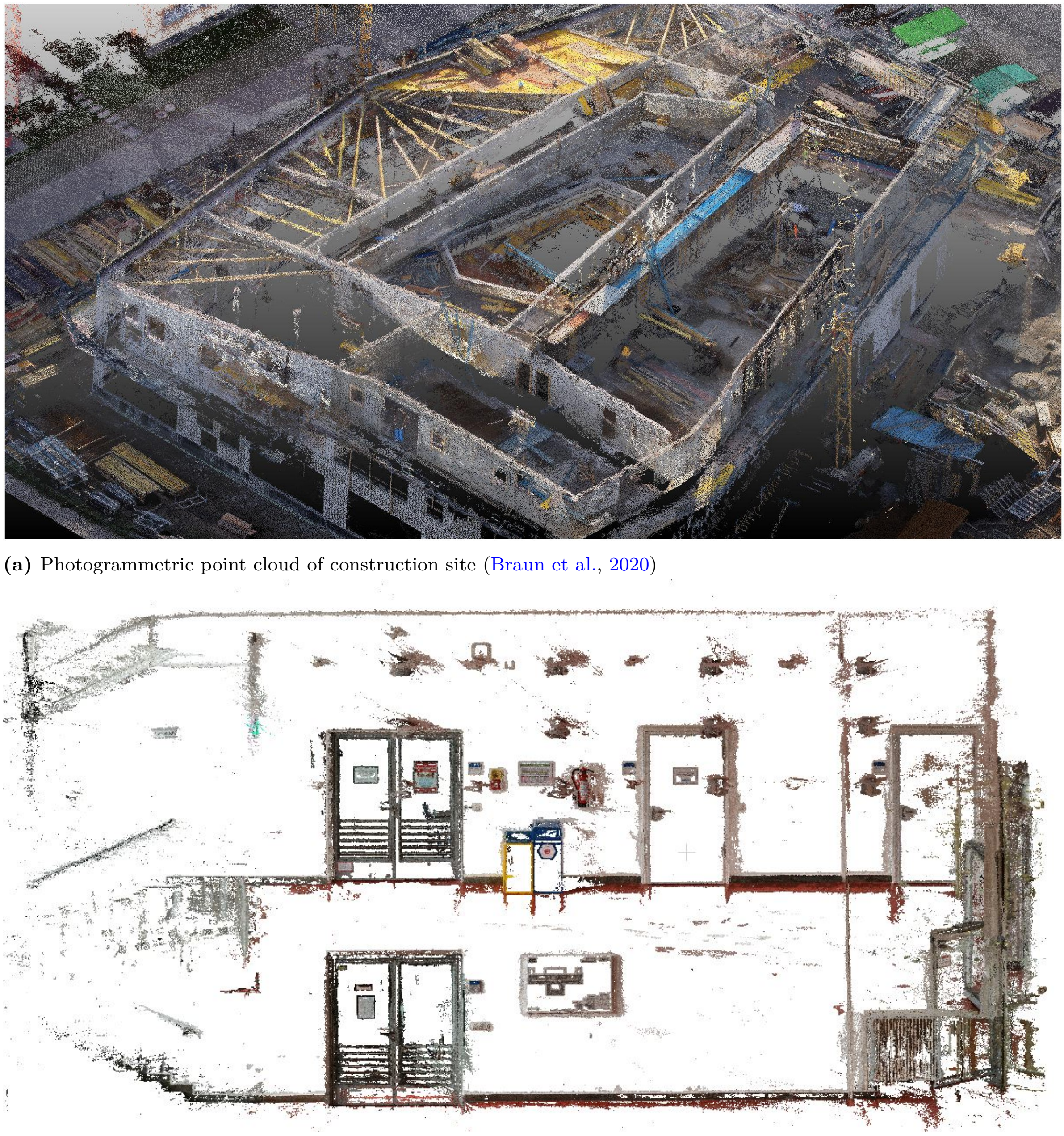
where c is the light speed between the device and the target, φ is the phase shift, and f is the modulation frequency.

Imaging

In contrast to laser scanning, which represents targets with three coordinates, imaging usually produces a 2D visual representation, i.e., images. There are advantages of using cameras to capture the indoor environment compared with using laser scanners. The first one is that the capturing device is usually less expensive. A huge amount of different camera options can be selected, and even modern cell phones nowadays also have good camera lenses. The second advantage is that it requires much less learning time to use cameras than using a scanner, which makes it possible for all stakeholders of the facility to capture the current status of the asset. However, the 2D images contain only 2D information. If we did not locate and record the camera position and orientation when taking the image, the camera information, which is important to reconstruct the 3D objects, cannot be found directly. In order to represent objects' 3D information, like locations and dimensions in 3D space, different approaches to processing images have been proposed.

Photogrammetry is one category of these approaches that reconstructs 3D information from 2D images. It is also called videogrammetry when capturing videos instead of photos. Basically, a video can be seen as a set of photos taken with a given frequency. The whole process consists of two steps: Structure from Motion (SfM) (Özyeşil et al., 2017) and Multi-View Stereo (MVS) (Seitz et al., 2006). The input of SfM is a set of overlapping images taken from different viewpoints. It starts with feature detection and extraction through feature matching and geometric verification and then reconstructs the object in 3D space, including the reconstructed intrinsic and extrinsic camera parameters of all images. MVS takes the output of SfM to compute depth and normal information for pixels in all images and creates a dense point cloud of the scene. The performance of photogrammetry relies on detecting and matching feature points in images. It aims to find key points representing the same point in 3D space on different 2D images.

Photogrammetric methods detect points where the colour changes, making the weakly textured and homogeneous surfaces hard to reconstruct. However, weakly textured surfaces are quite common in the indoor environment, like white wall surfaces. As shown in Figure 2.2, the reconstructed photogrammetric point cloud for images taken in a construction site is much denser and more complete than that using images taken in the indoor environment, as there are almost no homogeneous areas in images of the construction site. Although the reconstruction result of the indoor environment is not as good as the outdoor environment, almost all the key feature points are found, i.e., boundary points of objects. The reconstructed 3D key points are able to show the location and dimension of objects in the environment.



(a) Photogrammetric point cloud of construction site (Braun et al., 2020)

(b) Photogrammetric point cloud of indoor space

Figure 2.2: Comparison between photogrammetric point clouds of weakly and strongly textured surfaces (Construction site: weakly textured; indoor space: strongly textured)

2.1.2 Data Processing Methods

In this section, different technologies that are used to process the captured raw data and extract useful information are discussed. As point clouds and images are two important data sources for capturing the built environments, the methods here are also divided into two categories. While point cloud processing methods (mainly focusing on methods of segmentation) are introduced in Section 2.1.2, methods of processing images are discussed in Section 2.1.2.

Point Cloud Segmentation

Point cloud segmentation is the process of grouping points into subsets (normally called segments) characterized by having characteristics in common (Grilli et al., 2017). Sometimes, it can also be seen as an important pre-processing step, which aims to generate different types of data that can be integrated into further processes such as classification. Segmentation itself is formulated particularly as graph clustering in computer vision (Douillard et al., 2011). Most methods of segmenting point clouds are developed from segmenting 2D pictures. As shown in Figure 2.3, the author divides methods of point cloud data segmentation roughly into five categories: edge-based approaches, shape-based approaches, region-based approaches, graph-based approaches, and deep learning approaches.

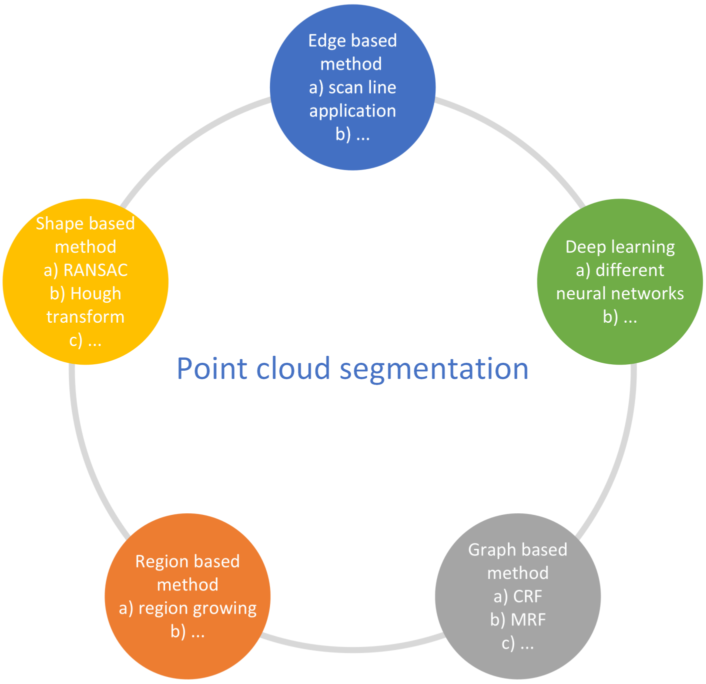


Figure 2.3: Different approaches to segment point clouds

Edge-based Approaches

As the name implies, edge-based approaches detect the edges of different regions to create segmented point clouds. Two approaches are proposed to detect 3D range data:

the line fitting approach and the surface normal approach (Bhanu et al., 1986). In the line fitting approach, the unit direction vector from each point to its nearest neighbouring points is calculated. If two direction vectors point in exactly opposite directions, these two vectors lie in a straight line. If two or more than two straight lines are found within a certain threshold, this point is not an edge point. If only one straight line is found, then it is an edge point. In the surface normal approach, the change in the neighbourhood of a point would be calculated. If the normal vectors change significantly, it means this point is an edge point.

A scan line grouping technique is proposed to segment a range image into planar regions (Jiang and Bunke, 1994). Then it is extended to both planar, and curved surfaces (Jiang et al., 1996). A scan line segmentation first splits each scan line (or row of a range image) into pieces and then merges these scan line segments with segments of adjacent scan lines based on some similarity criteria. Another method that generates a binary edge map based on the scan line grouping technique is designed to segment large range images (Sappa and Devy, 2001). After generating the binary edge map, a contour detection strategy is applied to extract the boundaries of surfaces. In this process, only information in the binary edge map is used. In conclusion, edge-based approaches are implemented firstly to locate edge points, then link them to obtain the contour that defines each region. So, the main problem of these approaches occurs if some edges can not be detected and no closed region can be created. In addition, all these approaches are sensitive to the noise of data, which is quite common in point clouds.

Shape-based Approaches

Edge-based approaches attempt to find points in the point cloud that fit geometric shapes. A well-known algorithm called RANdom SAmple Consensus (RANSAC) can be used to detect geometric shapes like lines, circles, and planes in point clouds (Fischler and Bolles, 1981). It selects a minimal initial data set and enlarges this set with consistent data when possible. For example, when fitting a circle to a set of 2D points, the RANSAC approach would select three points, calculate the centre and radius of the implied circle, and count the number of points that are close enough to that circle. If there are enough points that are closed enough to the circle, RANSAC calculates the parameters of the circle now. That means some points of the circle are detected. In conclusion, the RANSAC extracts shapes by randomly taking minimal sets from the point data and constructing shape primitives. The resulting candidate shapes are tested against all points in the data to determine how many points are well approximated by the primitive. After a given number of trials, the shape which approximates the most points can be extracted.

Many subsequent methods designed for different scenarios are developed. The RANSAC is applied to detect cylinders in range images (Bolles and Fischler, 1981). Another method to detect geometric primitives is proposed to process multiple range images (Reisner-Kollmann and Maierhofer, 2012). It starts with generating a global graph of sample points covering all input frames. Then, the RANSAC approach is implemented iteratively, optimizing shape parameters. Besides applications in range images, RANSAC and Gaussian images are used to detect cylinders in point clouds (Chaperon and Goulette, 2001). A large number of extensions of RANSAC have been proposed. For example, MLESAC proposed a new score function, which chooses the solution that maximizes the likelihood rather than just the number of inliers (Torr and Zisserman, 2000). This method improves the robustness of RANSAC. Another algorithm based on RANSAC is used to detect basic shapes in unorganized point clouds (Schnabel et al., 2007). It can detect surfaces composed of basic shapes like planes, spheres, cylinders, cones, and tori. Furthermore, this algorithm is robust even in the presence of many outliers and a high degree of noise.

Another RANSAC-based method that is designed to handle the data noise is proposed by considering local and global relations simultaneously (Li et al., 2011). The whole process from noisy input point data to the final reconstructed model contains several iterations of RANSAC fitting and constrained optimization. Starting with a set of locally fitted primitives based on RANSAC, relations across the primitives are progressively learned. In each local RANSAC stage, a set of feasible relations are extracted from the possible relations and then aligned to the input data. The global coupling corrects the primitives obtained in the local RANSAC stage and brings them to precise global alignment.

The well-known Hough transform is used to detect lines and circles in 2D datasets. Several variants of the Hough transform with respect to their applicability are evaluated to detect planes in 3D point clouds (Borrmann et al., 2011). In 3D Hough transformation, every non-vertical plane can be described with three coordinates in object space and three parameters in parameter space. For example, in this equation, $z = ax + by + c$, (x, y, z) denotes points in this plane and (a, b, c) denotes the plane parameters which define the plane. That means every point (a, b, c) in the parameter domain corresponds to a plane in the object space. So, the detection of planar surfaces starts from mapping points to planes in the parameter domain. The position where most planes intersect in the parameter domain describes the plane equation in the object domain. Normal vectors can be used in Hough transform to accelerate the calculation. A plane in the object domain can be completely defined by a point and the normal vector. So, the parameters of this plane are mapped to a single point instead of a plane in the parameter domain. The cost of computing can be reduced in this way.

Cylinders and spheres can also be detected by the Hough transform. A cylinder is described by five parameters, and a sphere is described by four parameters. If normal vector information is used, the dimension of the parameter domain can also be reduced. Although the calculation of detecting cylinders and spheres is complex, the basic idea is the same. They are all variants of the Hough transform for line detection.

Hough Transform and extended RANSAC algorithms are compared when detecting 3D building roof planes (Tarsha-Kurdi et al., 2007). In this application, RANSAC is more efficient than the Hough Transform, and Hough Transform is very sensitive to the segmentation parameter values. A method of using filtered normals and voxel growing is proposed in (Deschaud and Goulette, 2010). The first step of this method is estimating better normals in the point cloud data. The second step is to use a score function to estimate local planarity. Then the best local seed plane would be selected, and region growing, which would be introduced later, is started.

In conclusion, model-based methods usually use purely mathematical principles so that they are fast and robust with outliers. Even in most recent research, planar geometric primitives are fitted by the RANSAC-based or Hough-transform-based methods.

Region-based Approaches

Region-based approaches combine nearby points that have similar properties to retrieve isolated regions. The region growing method is proposed to segment images (Besl and Jain, 1988). It starts by giving each pixel a label based on its value and the values of its neighbouring pixels. This label can only take on values based on two surface-curvature signs and indicates the qualitative shape of an approximating surface that best fits the image data surrounding that point. After getting this surface type label image, iterative region growing proceeds using the surface curvature sign to determine a surface type label for each pixel.

The surface growing in point clouds is quite similar to the region growing in images. In some literature, objects in point clouds are considered to be polyhedral. In that case, the segmentation algorithms should determine planar surfaces. To apply surface growing, the identification of seed surfaces and criteria for extending these surfaces should be determined (Vosselman et al., 2004).

A surface growing method for airborne laser scanning that firstly picks a seed point randomly and then examines the n nearest neighbouring points and whether they fulfil certain criteria is proposed to segment buildings, and vegetation (Tóvári and Pfeifer, 2005). These criteria here include the similarity of normal vectors, the distance of the candidate point to the adjusting plane, and the distance between the current point and the candidate point. For terrestrial laser scanning, a surface growing method is used to segment the building facade (Pu et al., 2006). The algorithm selects an arbitrary unclassified point and tests if a minimum number of nearby points can be fitted to a plane. If this is the case, these points constitute the seed surface. The region growing starts from the seed surfaces based on some criteria, such as the distance to the seed surface. An extended approach based on region growing is used to identify structural components in synthetic scenes, as well as a collapsed bridge scene (Walsh et al., 2013). It can locate elements of a collapsed bridge, such as piers and pier caps. But it can not detect the edge between a pier and a pier cap, and key points for region growing are selected manually actually.

A two-stage region growing method is proposed to segment objects, and architectural components (Ning et al., 2009). The first-stage region growing, named rough segmentation is used to extract main objects based on the consensus of the normal vector in the same plane. The second-stage region growing, named detail segmentation, aims to extract detailed information for components. Seed points for both stages need to be selected manually. Another region-growing method based on geometrical continuities is proposed for the robust segmentation of building point clouds. It requires a single input from the user on the desired level of abstraction (Dimitrov and Golparvar-Fard, 2015). The most obvious improvement is that seed points here are selected adaptively.

In conclusion, region-based approaches are highly dependent on selected seed points and the thresholds of similarity criteria. Furthermore, over- or under-segmentation problems often occur in the result.

Graph-based Approaches

Graph-based approaches consider point clouds as graphs. A simple graph-based approach is considering each point in the data as a vertex and connections between pairs of neighbouring points as edges. An algorithm based on pairwise region comparison is firstly proposed to segment 2D images (Felzenszwalb and Huttenlocher, 2004). In this algorithm, the pairwise region comparison considers the minimum weight edge between two regions in measuring the difference between them. Thus two regions will be merged even if there is a single low-weight edge between them.

A k-nearest neighbours graph is presented to represent the structure of the point cloud, and the edges of this graph have weights that decrease with distance (Golovinskiy and Funkhouser, 2009). This method assumes a background prior, adds constraints of hard foreground, and finds the min-cut to compute a foreground and background segmentation. Strom et al. (2010) propose an algorithm for segmenting a coloured point cloud derived from a laser scanner and a camera that combines the previous 2D segmentation with spatial features such as surface normals. The experiment shows that it is more robust than segmenting either laser data alone or colour image alone. If there are not any RGB scanners available, the co-registration process must be implemented to produce the joint point cloud.

Many previous researchers combine graph-based methods with probabilistic models. The idea of Conditional random field (CRF) (Lafferty et al., 2001) is firstly proposed to segment and label sequence data. A method that is able to successfully segment and label 3D point clouds is presented by defining classes of 3D geometric surfaces and making use of contextual information using CRFs (Rusu et al., 2009). In this method, a Fast Point Feature Histogram (FPFH) is proposed to create a feature space in which 3D points lying on primitive geometric surfaces (e.g., planes, cylinders, spheres, etc.) can be easily identified and labelled. The computational properties of this approach exhibit a favourable integration for fast 3D classification of laser data.

Schoenberg et al. (2010) propose an algorithm is proposed to segment 3D points in dense range data generated from the fusion of a single optical camera and a laser scanner. This method uses Markov Random Field (MRF) to estimate a 3D point corresponding to each image pixel. The fusion process is implemented by combining the capture of low-resolution range images with the acquisition of registered high-resolution camera images (Diebel and Thrun, 2006). The textured 3D dense point cloud is segmented based on evidence of a boundary between regions of the textured point cloud. Clusters are discriminated based on Euclidean distance, pixel intensity, and estimated surface normal using a fast, deterministic, and near linear time segmentation algorithm. A simplified MRF model is used to segment facades of models, and it uses the contextual relations between points, where the node potentials are calculated from point-wise classification results using some classifiers, such as Support Vector Machine (SVM) (Lu and Rasmussen, 2012). Building roof contours are identified from among the above-ground objects by optimizing an MRF-based energy function that embodies roof contour attributes and spatial constraints (Galvanin and Dal Poz, 2011). The optimal configuration of building roof contours is found by minimizing the energy function using a simulated annealing algorithm. In general, these methods show satisfactory performance by detecting building facades and roof contours.

In conclusion, graph-based methods can segment complex scenes in different scenarios with good results. In addition, it is possible to co-register data from sensors and camera systems together which may potentially improve the segmentation performance by using the more collected information.

Deep learning Approaches

Artificial Neural Networks (ANNs) are computer programs that aim to simulate the way in which the human brain processes information (Agatonovic-Kustrin and Beresford, 2000). The concept ANNs is proposed decades ago. With the popularity of deep learning, ANNs have been mentioned and discussed more frequently in the last years (Kim, 2017). As there are many neurons in our brain that can transmit and process information, ANNs consist of lots of neurons (also called nodes) as well. Specifically, some neurons constitute layers, and these layers constitute an ANN. As shown in Figure 2.4, input information goes into the input layer, through hidden layers, and ends at the output layer.

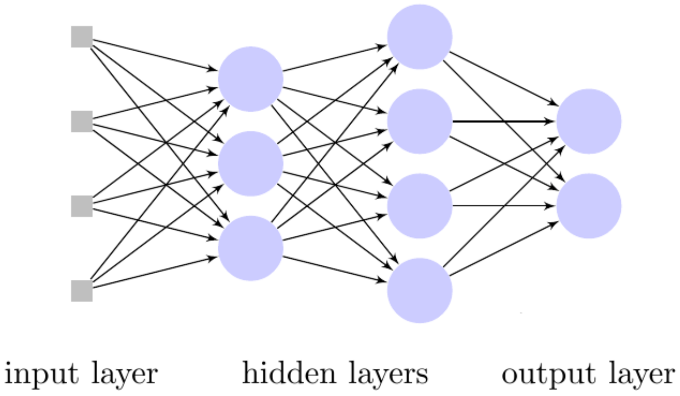


Figure 2.4: Neural network with two hidden layers

There are two most common problems in deep learning: classification and regression. In classification problems, the aim is to predict the classes to which the data belongs based on a set of features present in the input data. In comparison, the goal of regression problems is to predict or estimate a value rather than a class. While predicting unknown labels is the classification problem in deep learning (Goodfellow et al., 2016), point cloud segmentation can also be seen as a classification problem for each point in the point cloud. Therefore, many networks are designed to solve the classification problem and segmentation problem in one architecture framework when processing point clouds.

Some neural networks work on voxel structure which requires point cloud voxelisation first. Prokhorov (2010) uses 3D CNNs to solve a binary classification problem. Maturana and Scherer (2015) proposed a more general network with 3D CNN architecture called VoxNet to detect classes of objects for 3D point cloud data. The input of VoxNet is a point cloud, and VoxNet aims to predict a class label for the input. A volumetric grid that can represent spatial occupancy is calculated first and then applied to 3D CNNs.

Compared with methods requiring voxel input, many networks process point clouds as input directly. The neural network architecture, PointNet, is the first network that is directly proposed for 3D deep learning in the point cloud Qi et al. (2017). It takes point clouds as input directly and outputs labels for the entire input or labels for each point. The basic but novel idea of PointNet is to consider each point (x-, y-, z-coordinate) independently at the first stage in the network. Apart from the three spatial coordinates values (x,y,z), additional dimensions could be added by computing normals and other local or global features. PointNet processes features of individual points independently and extract global features of the entire point set. An improved network based on PointNet considering spatial information of point sets called PointNet++ is proposed (Qi et al., 2017). In PointNet++, the set of points is grouped into overlapping local regions by the distance metric. Then local features are extracted by capturing fine geometric structures from small neighbourhoods. These local features are grouped together into larger units and processed to produce higher-level features.

Other approaches inspired by PointNet, which processes the point cloud directly, are proposed to improve the performance. Dynamic graph CNN (Wang et al., 2019), differing from networks working on individual points like PointNet, constructs local geometric structures by a local neighbourhood graph and applies convolution-like operations to the graph. Its name "dynamic" means that compared with graph CNNs, the graph in this model is not fixed but is dynamically updated after each layer of the network, thereby performing convolutions not only in the metric neighbourhood but also in the semantic one. Li et al. (2018) propose a novel convolution operator called χ- operator to extract features from the point cloud.

Thomas et al. (2019) present another novel way to apply convolution operation in the point cloud, which is called kernel point convolution. In this method, the convolution operation is done in kernel points and points close to them, which is shown in Figure 2.5. Hu et al. (2020) introduce a novel local feature aggregation module, which works faster in large-scale point clouds. Fan et al. (2021) design a module that learns spatial contextual features from the point cloud and embeds it in an encoder-decoder architecture. Qiu et al. (2021) augment the local context of points and interpret the distinctness of the points from multiple resolutions to achieve semantic segmentation. Some networks adapt the transformer architecture (Vaswani et al., 2017) from natural language processing and apply the idea in point cloud segmentation (Guo et al., 2021) (Zhao et al., 2021) (Engel et al., 2021). These networks show that the transformer architecture is also powerful in point cloud processing. Perez-Perez et al. (2021) design a network for the Scan-to-BIM process to segment the structural, architectural, and mechanical components.

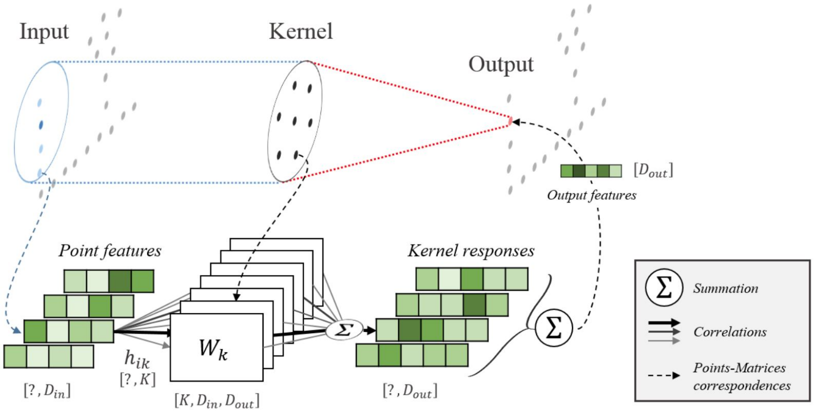


Figure 2.5: Kernel point convolution operation (Thomas et al., 2019)

For point cloud segmentation of buildings, the Stanford 3D Indoor Scene (S3DIS) dataset that contains point clouds of six areas of over $6,000m^2$, is widely used to evaluate the performance of different architectures (Armeni et al., 2016). The performance evaluation of the above-mentioned architectures on the S3DIS dataset is collected and shown in Table 2.1. It is obvious that most networks perform well for classes like ceiling, floor, and wall, while the performance for other categories is not at the same level. For example, if we use semantic information in reconstructing buildings, the predicted labels of a ceiling class (Intersection over Union (IoU) around 90%) are more reliable than those of a window class (IoU around 60%). Therefore, how to define and extract useful information computed from neural networks is still an ongoing research topic.

model	mIoU	ceil.	wall	floor	wind.	door	colu.	beam	chair	table	book.	sofa	board	clut.
PointNet (Qi et al., 2017)	47.6	88.0	69.3	88.7	47.5	51.6	23.1	42.4	42.0	54.1	38.2	9.6	29.4	35.2
SPG (Landrieu and Simonovsky, 2018)	62.1	89.9	76.4	95.1	55.3	68.4	47.1	62.8	73.5	69.2	63.2	45.9	8.7	52.9
RSNet (Huang et al., 2018)	56.5	92.8	92.5	78.6	51.6	68.1	34.4	32.8	60.1	59.7	50.2	16.4	44.9	52.0
PointCNN (Li et al., 2018)	65.4	94.8	75.8	97.3	58.4	57.2	51.7	63.3	71.6	69.1	39.1	61.2	52.2	58.6
KPConv (Thomas et al., 2019)	70.6	93.6	83.1	92.4	66.1	76.6	54.3	63.9	57.8	64.0	69.3	74.9	61.3	60.3
Point Transformer (Zhao et al., 2021)	70.4	94.3	84.7	97.5	66.1	78.2	58.1	55.6	74.1	77.6	71.2	67.3	65.7	64.8

Table 2.1: Segmentation mean Intersection over Union (mIoU) on S3DIS dataset (evaluated with 6-fold cross-validation)

In summary, occlusion caused by furniture occurs quite often in indoor environments, which leads to geometry missing in point clouds. The majority of methods are sensitive to occlusion because they only use geometric information in point clouds. They first detect surfaces of elements in point clouds utilizing Hough transform (Adan and Huber, 2011), RANSAC (Wang et al., 2017) (Ochmann et al., 2019) (Murali et al., 2017), or region growing (Xiong et al., 2013) (Mura et al., 2014) (Stambler and Huber, 2014). When occlusions occur in the environment, the performance of these methods declines, especially when there are complex rooms (like L-shape rooms and U-shape rooms) in the point cloud. The main reason is that reconstructing surfaces solely based on geometric information makes it hard to distinguish a large surface of furniture (like a cupboard surface) from a relatively small wall surface. Semantic information extracted by deep learning provides the possibility to improve the reconstruction. However, with regard to applying 3D deep learning in the process of creating digital twins in practice, we still need to extract useful and precise information from imprecise and redundant information predicted by neural networks.

Image Processing Methods

In this section, methods extracting semantic information from images are introduced. More specifically speaking, object detection methods and the methods of extracting text information are reviewed.

Object Detection Networks and Transfer Learning

In computer vision, object detection refers to identifying an object and precisely estimating its location (Szegedy et al., 2013). One of the most widely used algorithms in object detection is Region-based Convolutional Neural Network (RCNN) (Girshick et al., 2014). In RCNN, regions of interest are identified first and then classified by Convolutional Neural Network (CNN) to detect objects in the regions. Since original RCNN is relatively slow, some variants of RCNN have been proposed, like fast-RCNN (Girshick, 2015), mask-RCNN (He et al., 2017).

In the AEC domain, researchers have also applied and proposed different network architectures to achieve their research objectives, for example, defect and damage detection ((Jiang et al., 2021), (Tan et al., 2021), (Wu et al., 2021)), worker detection on construction sites ((Jeelani et al., 2021), (Son and Kim, 2021), (Nath et al., 2020)).

A neural network can be trained from scratch on a specific dataset. However, in order to achieve optimal results, it requires a large training set, as well as substantial processing time (Kolar et al., 2018). Therefore, transfer learning (Pan and Yang, 2009) is proposed to overcome the problems and improve performance. Transfer learning is a process where a neural network is pre-trained on a related larger dataset and re-trained on a user-specific dataset. Currently, there are several large, publicly available datasets that are used to pre-train a neural network, such as ImageNet (Deng et al., 2009), which contains more than one million images for training, the Pascal VOC 2012 dataset that contains more than 20,000 images (Everingham et al., 2010), the COCO dataset contains more than 300,000 images (Lin et al., 2014) with 2.5 million instances.

Text Detection and Recognition

In a building, some elements contain texts and numbers that are also valuable for facility management, such as room numbers on a door sign. In large facilities, entities of some electrical elements (such as smoke alarms and emergency switches) usually have a unique serial number in order to clearly label entities and make facility management more efficient. It is also very helpful to attach this information to the objects in the building twin, recognising and identifying objects at an instance level. There are usually two steps to extract the information from images: text detection and text recognition.

With regard to text detection, neural networks that are used in object detection can also be used to detect text in an image, such as Mask-RCNN (He et al., 2017) because text area can also be considered a type of object. Researchers have also proposed neural networks that aim to detect text in an image like (Long et al., 2018), (Wang et al., 2019), (Wang et al., 2019), (Liao et al., 2020), (Zhu et al., 2021). These networks were proposed to detect arbitrary-shaped text in an image and can be trained on large, publicly available datasets like ImageNet (Deng et al., 2009).

With regard to text recognition, some neural networks have been proposed to recognise regular and irregular text in an image, like (Shi et al., 2017), (Li et al., 2019), (Sheng et al., 2019), (Yue et al., 2020). These networks can be trained on text image datasets, such as the SynthText dataset (Gupta et al., 2016), which contains approximately 800 thousand synthetic scene-text images, the COCO-Text dataset (Veit et al., 2016) with more than 60 thousand real images and around 239 thousand annotated text instances.

In the field of building reconstruction, only a few previous works deal with text detection and recognition, and these focus on CAD drawings. In (Lu et al., 2020), the authors used Optical Character Recognition (OCR) technology to extract text information from CAD drawings and then added detected information to the as-is digital model of buildings. In (Zhao et al., 2021), the authors applied OCR to extract the object information from the images of structural drawings (i.e. grids, columns and beams) and generate Industry Foundation Class (IFC) models for buildings.

2.1.3 Reconsruction of Space-bounding Elements

In this section, the state of the art in the process of creating digital twins of space-bounding elements is discussed. In recent years, although some approaches have been proposed to automate the process of reconstructing 3D models from the point cloud, this topic is still solved only partially, and most previous work can be applied only to specific types of buildings and is restricted to reconstructing specific kinds of objects. This section is organised as the following four parts: single room reconstruction, reconstruction of multiple rooms, reconstruction of multiple storeys, and reconstruction of window/door openings.

Single Room Reconstruction

Some approaches are proposed to reconstruct individual rooms. Budroni and Boehm (2010) use plane sweeping to segment horizontal surfaces and vertical structures. Positions of the floor, ceilings, and walls are automatically detected. Then the 3D model can be generated from the detected ground contours. Adan and Huber (2011) use a histogram to determine surfaces of ceilings and floors and then use Hough transform to detect wall surfaces. Xiong et al. (2013) propose a region growing method that is applied to form different patches and implement a stacked learning approach to classify the detected patches. As digital twin is more valuable in large facilities than in a single room, and apparently, the process is much more complex for larger buildings as well, more researchers focus on the reconstruction process for buildings with multiple rooms.

Multiple Rooms Reconstruction

More approaches aim to reconstruct multiple rooms. Sanchez and Zakhor (2012) propose an approach that employs principle component analysis and RANSAC to detect large-scale architectural structures, such as ceilings and floors, as well as small-scale architectural structures like staircases. In this approach, all points are classified into doors, ceilings, walls, and remaining points. Monszpart et al. (2015) extract planar structures in a point cloud that follows regularity constraints and then optimise the plane arrangement. Authors use this approach to extract planes in many scenes, such as urban scenes, the exterior and interior of buildings. Oesau et al. (2013) use horizontal slicing, and volumetric-cell labelling is proposed to reconstruct watertight surface meshes. The binary labelling of the volumetric cells is formulated as energy minimisation and solved by the graph-cut method.

Xiao and Furukawa (2014) propose a method named "inverse Constructive Solid Geometry (CSG)" that detects planar surfaces and then fits the cuboid primitives to the point cloud. Mura et al. (2014) use an approach based on the diffusion process of space partitioning. They extract patches using a simple region-growing process based on normal deviation and plane offset. Wang et al. (2017) use Principal Component Analysis (PCA) to estimate the normal for each point, RANSAC to fit linear primitives, and graph-cut to identify walls. The proposed hierarchical clustering method can identify each room without knowing the number of rooms. Murali et al. (2017) use the RANSAC-based method to detect vertical and horizontal planes. Then they create a wall graph and fit cuboids into rooms. (Ochmann et al., 2016) propose a method that explicitly represents buildings as interconnected volumetric wall elements. They determine an optimal room and wall layout by graph-cut-based multi-label energy minimisation.

Some approaches use prior knowledge explicitly to reconstruct walls and rooms. Stambler and Huber (2014) propose the concept of the enclosure, reasoning that premises rooms are cycles of walls enclosing free interior space. They use the region growing method to segment the point clouds and simulated annealing to optimise rooms and walls. Tran et al. (2019) use the shape grammar approach to model indoor environments. They generate 3D parametric models by placing cuboids into point clouds and classifying them into elements and spaces. The wall candidates are obtained from pairs of adjacent peaks in the histogram of point coordinates. Truong-Hong and Lindenbergh (2022) propose a method to extract elements like floors, ceiling slabs, columns and beams by rough extracting the candidate points of the component and then fine-filtering the surface points of the components from the construction site. As digital twins always aim to reflect the current status of the whole building, some methods are proposed to reconstruct multi-storey buildings.

Multi-storey reconstruction

Only a few methods target multi-storey buildings. Macher et al. (2017) propose a semi-automatic reconstruction approach for multi-storey buildings. The automatic part of their work is to segment the input data into sub-spaces (rooms) and planes. After segmentation, the 3D geometry of the elements is translated to the BIM model manually. Ochmann et al. (2019) reconstruct volumetric models of walls and slabs in multi-storey buildings. Planes are detected first by employing RANSAC, and then the detected planes are classified as horizontal slab surfaces and vertical wall surfaces. A 3D plane arrangement is constructed by intersecting all planes, yielding a cell complex. An integer linear programming approach, in which binary variables for each cell are interpreted as room, outside, and wall, is used to find an optimal label for all cells.

Some of the literature mentioned above are collected in Table 2.2. In summary, most approaches are not full-automatic, which means they still require human effort in the process of reconstruction, And most of them are only relying on geometric information only. The performance, when applied in the point cloud with a high occlusion level, would decrease because of geometric information deletion caused by the occlusion of furniture.

In summary, the methods of reconstructing space-bounding elements can be roughly divided into two categories: bottom-up approach and top-down approach. In the bottom-up approach, we start from the point level. By clustering and fitting, we can get surfaces or clusters, and then get to the object level by recognising the objects. In the top-down approach, we hypothesize that many infrastructure object classes (e.g. wall) are more uniquely distinguishable through their pose and relationships to other objects than their local features (e.g. plane, colour). In this case, a complete point cloud is segmented into sections (e.g. vertical point clusters) and then into components (e.g. walls, windows, doors).

Table 2.2: Summary of the representative approaches in 3D building reconstruction

Task	Method	Type of reconstructed model	Sensor	Element
Buildings reconstruction in urban scene (Vosselman et al., 2001)	3D Hough transform and using ground plan	planar-surface model (Roofs and facades)	laser scan	point
Floor plan reconstruction (Okorn et al., 2010)	Line fitting by Hough transform	Walls in 2D floor plan	laser scan	point
Floor plan reconstruction (Liu et al., 2018)	PointNet with points, CNN with density images, CNN with RGB images	alls in the 2D floor plan (Different rooms are labelled)	RGBD stream	point, image
Planar reconstruction in man-made scene (Monszpart et al., 2015)	Regular arrangement of planes	Planar-surface model (walls, stairways, chairs, etc.)	laser scan	point
Indoor scene reconstruction (Budroni and Boehm, 2010)	Plane sweeping method	Extruding 2D contour to 3D model (walls for one room)	laser scan	point
Indoor scene reconstruction (Xiao and Furukawa, 2014)	Horizontal slicing 3D space sing Hough transform to fit rectangle primitives then generate 3D primitives	volumetric model (walls)	laser scan	voxel
Indoor scene reconstruction (Sanchez and Zakhor, 2012)	PCA and RANSAC	Planar-surface model (walls, floors, staircases)	laser scan	point

Indoor scene reconstruction under occlusion (Previtali et al., 2014)	RANSAC, labelling occlusion and openings by Ray-tracing	planar-surface model (walls, ceilings, floors)	laser scan	voxel
Indoor scene reconstruction under occlusion (Adan and Huber, 2011)	Hough transform, labelling occlusion by ray-tracing, detecting opening by SVM, reconstructing occlusion by MRF	planar-surface model (walls, ceilings, floors)	laser scan	voxel
Indoor scene reconstruction under occlusion (Xiong et al., 2013)	Getting patches by region growing, labelling patches by stacked learning, labelling occlusion by ray-tracing, detecting opening by SVM, reconstructing occlusion by MRF	planar-surface model (walls, ceilings, floors)	laser scan	voxel
Indoor scene reconstruction (Oesau et al., 2014)	Hough transform, graph-cut	planar-surface model (walls, ceilings, floors)	laser scan	voxel
Indoor scene reconstruction (Turner et al., 2014)	Finding planes by PCA, labeling rooms by graph-cut	planar-surface model (walls, ceilings, floors)	laser scan	voxel
Indoor scene reconstruction (Mura et al., 2014)	Extracting patches by region growing, generating walls by PCA, labelling rooms by space diffusion	planar-surface model (walls, ceilings, floors)	laser scan	voxel
Indoor scene reconstruction (Wang et al., 2017)	Extracting primitives by RANSAC, labelling rooms by graph-cut, reconstructing doors by ray-casting	planar-surface model (walls, doors)	laser scan	voxel
Indoor scene reconstruction (Murali et al., 2017)	Extracting planes by RANSAC, detecting cuboids for rooms, matching door models to empty space	planar-surface model (ceilings, walls, doors)	laser scan	point

Indoor scene reconstruction (Mura et al., 2016)	Fitting rectangles as shape proxies, detecting components by graphs, reconstructing rooms by MRF	planar-surface model (slanted ceilings and walls)	laser scan, RGBD	voxel
Indoor scene reconstruction (Ambruş et al., 2017)	Detecting primitives by RANSAC, labelling rooms by energy minimization	2D floor plan (different rooms are segmented and labelled)	laser scan	point, voxel
Indoor scene reconstruction (Stambler and Huber, 2014)	Region growing, reconstructing walls by wall score, detecting doors by Hough transform	Volumetric model (ceilings, floors, walls, doors)	laser scan	voxel
Indoor scene reconstruction (Thomson and Boehm, 2015)	Detecting planes by RANSAC, fitting IFC walls to the detected planes	volumetric model (walls)	laser scan	voxel
Indoor scene reconstruction (Ochmann et al., 2016)	RANSAC to detect planes, global optimization of graphs	volumetric model (walls including openings for one storey)	laser scan	voxel
Indoor scene reconstruction (Macher et al., 2017)	Detecting planes by RANSAC, generating components by prior knowledge	volumetric model (slabs, walls for multi-storey building)	laser scan	point
Indoor scene reconstruction (Ochmann et al., 2019)	Detecting planes by RANSAC, clustering by Markov clustering, arranging planes and labelling cells	volumetric model (slabs and walls for multi-storey buildings)	laser scan	voxel
Construction site facade reconstruction (Xu et al., 2015)	Detecting planes by RANSAC, classifying points by histogram and random forest	planar-surface model (tubes and toe boards)	Photogrammetry	point

Construction site facade reconstruction (Xu et al., 2018a)	Slicing planar surface to extract points, extracting features by histogram	planar-surface model (tubes, toeboards, decks)	photogrammetry	point
Construction site facade reconstruction, (Xu et al., 2018b)	Geometric cues among voxels by probabilistic model, clustering voxels to components	planar-surface model	photogrammetry, laser scan	voxel

Opening detection

When collecting data with a laser scanner in an indoor environment, doors are usually opened. The transparent glasses of windows cannot be captured by a laser scanner. So, there are usually door and window openings in existence in laser-scanned point clouds of indoor environments. Some methods have been proposed to detect the openings in point clouds. Mayer and Reznik (2005) interpret the building facade from images and detect windows by predefined shape. Pu and Vosselman (2007) extract windows from terrestrial point clouds of building facade by grouping points in planar segments and then fitting rectangles to the boundary of hole points. In their following research (Pu and Vosselman, 2009), prior knowledge such as the wall, opening and roof features' sizes, positions, orientations, and topology is used to recognise these objects. In Ripperda (2008), the structure of facades is reconstructed by a facade grammar and a reversible jump Markov Chain Monte Carlo process. Truong-Hong et al. (2013) propose a sample method to categorise points as boundary or interior points based on an angle criterion. Holes caused by occlusions are distinguished if they do not fit the predefined opening dimensions. Haghighatgou et al. (2022) propose an approach that investigates the house facade where openings have different shapes, sizes, and non-symmetrical positions. But the occlusion holes having characteristics similar to window opening still cause problems in the detection.

2.1.4 Reconstruction of Small Objects

Apart from space-bounding objects such as ceilings, floors and walls, an information-rich digital twin of buildings should also contain other small but important objects, for example, objects from the energy and fire-safety sub-systems such as smoke alarms, emergency switches, etc. In the previous Section 2.1.3, the state of the art of reconstructing structural elements (ceilings, floors, and walls) is discussed already. Compared to structural elements in a building, other components are usually small in size and have different geometry properties, which makes it hard to apply the same methods to detect those small-scale elements. Therefore, 2D information from images and 3D information from laser-scanned point clouds are connected and integrated into the proposed approach. We believe that this combination provides a significant advantage over using the laser-scanned point cloud alone, especially for detecting small-scale components in a building. In addition, text information, including serial numbers and IDs, can also be extracted from 2D images, and the detected information can be used to enrich the digital twin further.

With regard to elements located on wall surfaces, such as sockets and light switches, (Meeussen et al., 2010) designed a robot that can recognise doors, door handles, and sockets to achieve the door task and plugging task. The electrical outlet pattern is detected in camera images by feature detection, and a laser scanning sensor is used to find the pose of a wall. In (Krispel et al., 2015), the authors detect light switches and sockets in orthographic 2D images by a random forest classifier. They use a feature descriptor pool to measure the probability of the detection. A method was designed in (Kang et al., 2010) that allows a mobile robot to get on/off an elevator in a multi-storey building. An algorithm is presented for recognising elevator buttons, where the input image is first converted to a binary image, and then the candidates of buttons and floor numbers are filtered out and ambiguous candidates are rejected by applying a neural network. While most of these methods are used to help robots recognise specific objects in the environment and perform a given task, little work has been done in the AEC domain. In (Adán et al., 2018), the authors proposed a method to detect objects such as switches, ducts and signs in a coloured point cloud. Depending on whether the objects have geometric discontinuities or colour discontinuities in the wall area, potential regions of interest are computed in colour images and depth images with regard to the wall plane, respectively. The region of interest is then matched to a predefined depth model database and a predefined colour model database that contain object classes in the scene.

With regard to elements mounted on the ceiling such as lighting, Kim et al. (2017a) proposed a recognition method based on thermal-mapped point clouds for building elements consisting of electrical systems and heating, ventilation, and air-conditioning (HVAC) components. Assuming the temperatures of these elements are different from other parts of the ceiling, the points of corresponding elements can be extracted from the point cloud. (Kim et al., 2017b) used two steps to recognise objects in thermal-mapped point clouds: segmentation with thermal information and classification with geometric information. The target objects are light fixtures on the ceilings, monitors on the wall and humans in the environment. In (Díaz-Vilariño et al., 2015), the authors extract the ceiling plane first and then convert the laser-scanned point cloud to an image of the ceiling. Fluorescent lighting and circular low-energy bulbs are detected from the image by the Harris corner detector and Hough transformation. In (Puente et al., 2014), a method to detect tunnel luminaires from the point cloud is proposed. In this approach, they use assumptions that are only valid in the tunnel, for example, luminaires are located at higher points at the side of the tunnel and have brighter colour patterns than their surroundings. With regard to identifying pipes, Czerniawski et al. (2016) proposed a method to detect pipe spools in a cluttered point cloud. The method used curvature estimation, points clustering, and feature matching to extract pipe spool objects. In an office building, pipes are rarely visible because they are usually located inside the walls or behind suspended ceilings. In (Agapaki and Brilakis, 2020), the authors used deep learning to detect and differentiate between different pipes in industrial facilities.

2.1.5 Digital Twin Visualisation

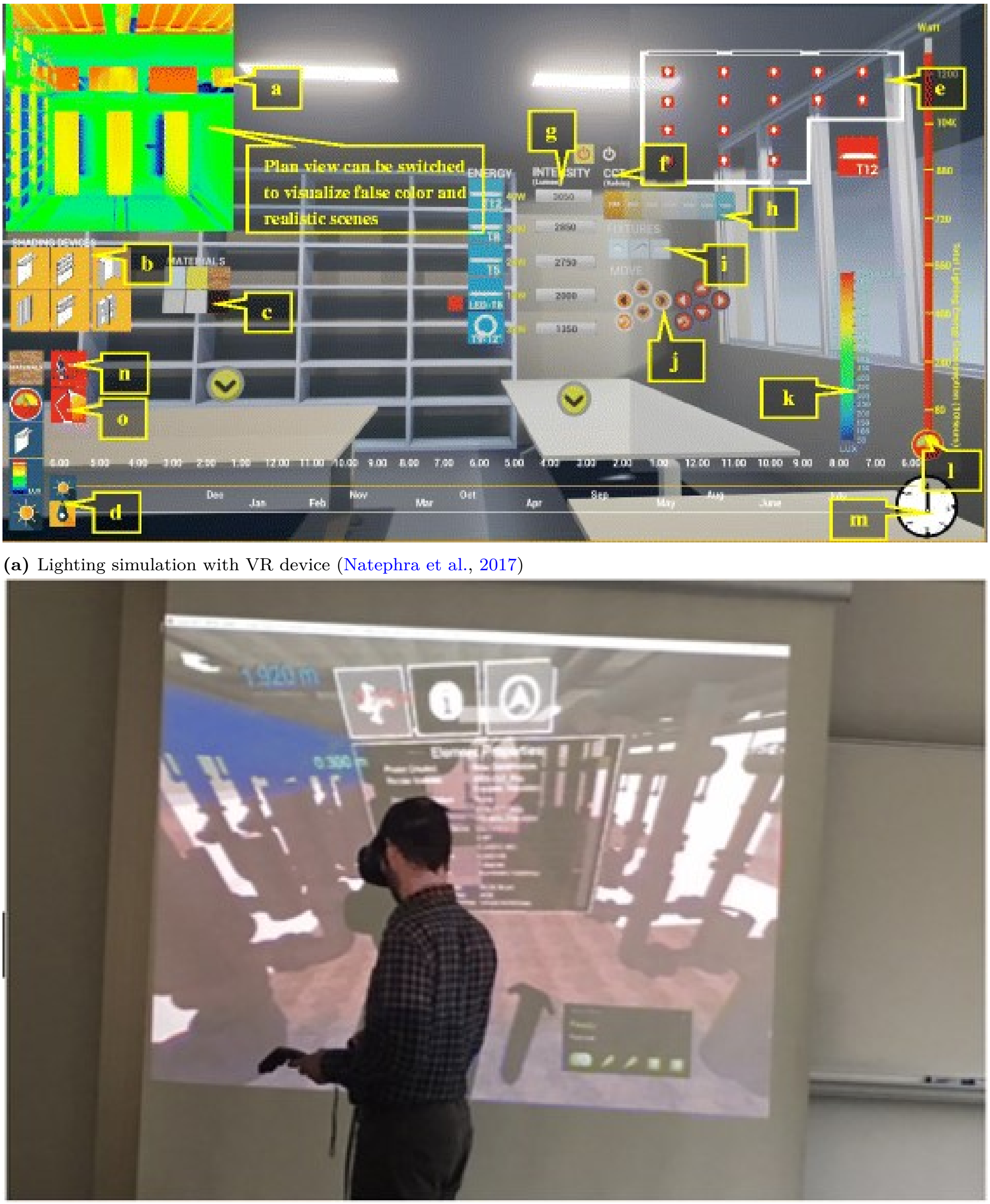
Nowadays, digital twins can be utilised by different technologies in various use cases. Especially the adoption of Virtual Reality (VR) and Augmented Reality (AR) technologies provides the possibilities to improve the user experience when using the digital twin in practice, for instance, design review, construction management and maintenance, and environment simulation. In addition, with the help of VR/AR technologies, digital twins of facilities can also be used in worker training, such as safety training and on-site operator training. The corresponding use cases for VR/AR technologies are discussed in the following sections.

Virtual Reality

VR is based on a set of technologies that puts a person in a digital simulation of an environment, which could be a simulation of the real world or just an imaginary environment. The VR devices make a simulated environment where the user is able to interact with the environment. There are four different kinds of VR devices (Schiavi et al., 2022):

- 3D Keyboard Mouse Screen (KMS), which comprises interaction with a keyboard and a mouse through the PC interface (Imaizumi, 2017);
- VR CAVE, which is a large screen projected with a wide range of views (Goulding et al., 2012);
- VR Standalone, which refers to the all-in-one headset which has built-in screens, processors, and storage;
- VR tethered Head Mounted Display (HMD), which is connected to a computer.

VR technologies can be used to simulate the digital twin of different assets, which makes it possible to achieve different tasks. For example, the distances of elements in the simulated model can be measured. As shown in Figure 2.6a, a tethered Head Mounted Display device is used to measure the space between model elements. Another example is shown in Figure 2.6b, lighting simulation is achieved using a VR standalone device.



(a) Lighting simulation with VR device (Natephra et al., 2017)

(b) Distance measurement with VR device (Zaker and Coloma, 2018)

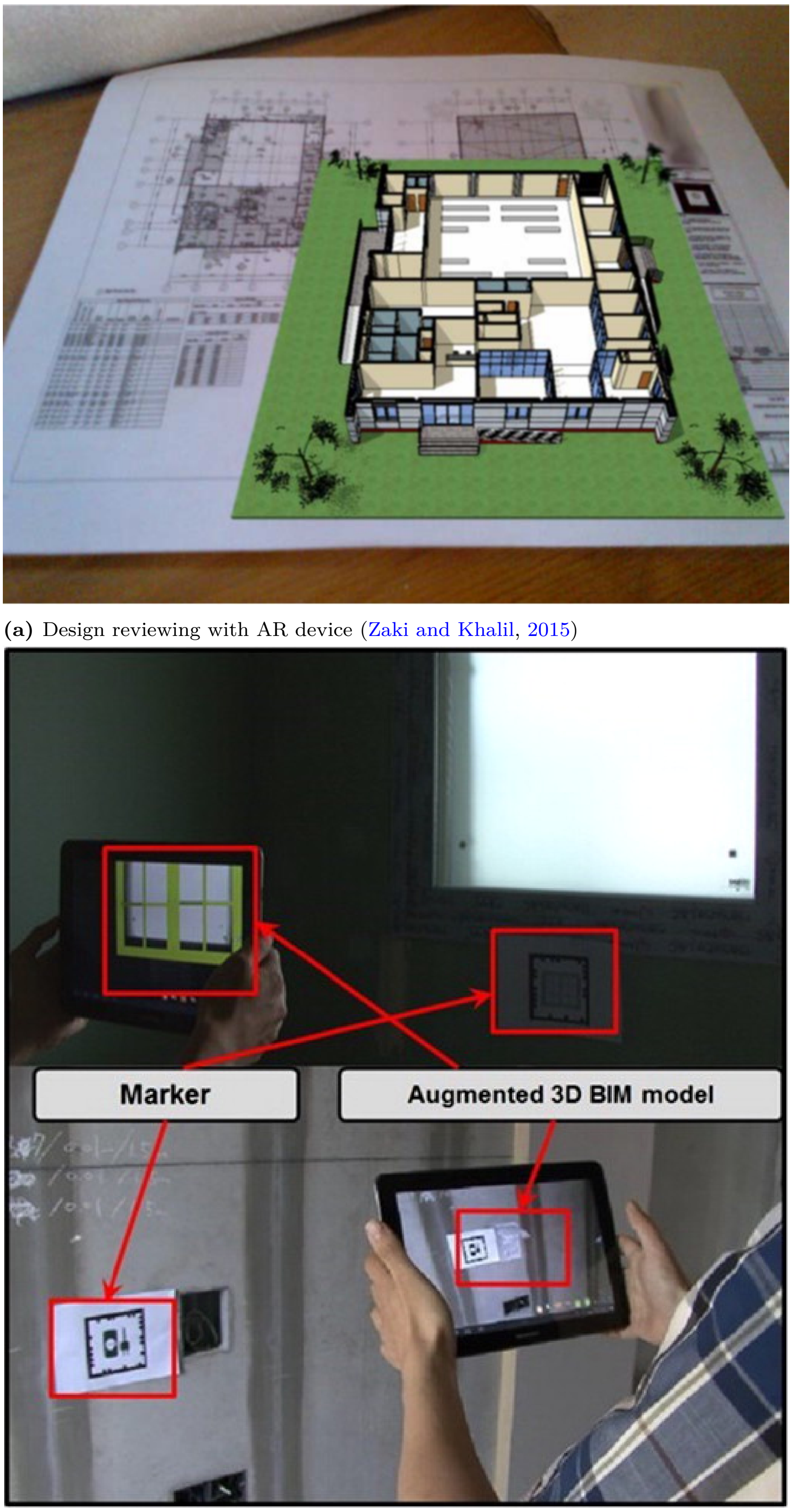
Figure 2.6: Example use cases with VR devices

Augmented Reality

AR overlays 3D virtual content onto the environment in the real world, which makes the user always see the real world. There are three categories of AR devices (Schiavi et al., 2022):

- AR fixed, which refers to a system setup with a fixed camera that streams the real world to a monitor with additional virtual elements (Xiang et al., 2019);
- AR mobile handheld, which refers to smartphones/tablets with AR marker-based or AR Simultaneous Localization And Mapping (SLAM)-based solutions (Olbrich et al., 2013);
- AR mobile smart glasses, which refers to devices that are optically transparent, like Google Glasses, Microsoft HoloLens, or HMD on optical see-through mode (Kivrak and Arslan, 2019).

Similar to VR technologies, AR technologies can also be applied in different use cases. For example, it can be used in the field of quality control. An example is shown in Figure 2.7a, where inspections to identify a void at the wrong location in a reinforced concrete wall before the concrete has been cast in the real world is achieved by an AR mobile handheld device. Another example is shown in Figure 2.7b, where the AR technology is used to review designs by projecting the 3D view from the 2D drawing.



(a) Design reviewing with AR device (Zaki and Khalil, 2015)

(b) Quality control with AR device (Kwon et al., 2014)

Figure 2.7: Example use cases with AR devices

2.2 Research Gaps

We summarise the research gaps in enriching a geometric digital twin of buildings as follows:

- In the data capturing process, sometimes only one single file of point cloud data for a multi-storey building with modern data capturing devices is generated nowadays. In this case, how to segment the point cloud of multi-storey buildings into individual storeys.
- In indoor environments, occlusion caused by furniture occurs quite often, which leads to geometry missing in point clouds. The majority of methods are sensitive to occlusion, especially when there are complex rooms (like L-shape rooms and U-shape rooms) in the point cloud.
- Artificial Intelligence (AI) has been penetrating the Architecture, Engineering and Construction (AEC) domain in recent years and provides possibilities to reduce human effort in the process of creating digital twins from a different perspective, especially Deep Neural Network (DNN) (Krizhevsky et al., 2012) provides a different and efficient solution to achieve point cloud semantic segmentation. However, the question of how exactly to use AI techniques and their output results to accelerate and improve the process is still an ongoing topic. With regard to applying 3D deep learning in the process of creating digital twins, it is not easy and clear how to extract useful and precise information from imprecise and redundant information predicted by neural networks.
- Buildings can be divided into two categories depending on whether they fulfil the Manhattan-world assumption, there is still a lack of methods that provide convincing results, especially for buildings that do not fulfil the Manhattan-world assumption under occlusion.
- Previous work focuses solely on structural elements and does not consider other smaller but valuable objects in a building. While some researchers detect geometric and colour discontinuities to find specific classes of small objects in images, these approaches do not apply AI-based methods to enhance performance. Moreover, most previous works dealt with only some classes of objects, and there is still a lack of comprehensive object categories when creating a building twin.
- Most previous work used only point clouds to achieve object detection and reconstruction. Because methods of object detection in 2D images are more mature and can provide better performance than those in 3D point clouds, there is a potential performance improvement when concatenating the information from various input sources.
- While text information attached to corresponding objects is also important in a rich building twin, none of the previous works considered adding text information. There is still a lack of creating a comprehensive information-rich building twin with geometric and semantic information.

2.3 Research Questions

Although an information-rich digital twin can benefit all stakeholders of the assets, there are only a few built facilities with available basic geometric models. There are mainly two reasons for this situation:

– many facilities were constructed years ago, and they have no pre-existing digital models from when they were constructed;

– those assets with a digital design model are never updated through the life cycle of the asset. Hence it is missing all asset modification information, dramatically reducing the data reliability.

Current capturing technologies such as laser scanning or photogrammetry (capturing technologies are discussed in Section 2.1.1 in detail) allow us to capture the geometric data efficiently, but the process of creating a digital representation for different kinds of objects from collected raw data is still time-consuming.

This PhD thesis aims to automate this process to reduce human effort in creating an information-rich digital twin. The author separates the whole process into the following sub-processes.

– Given a laser-scanned point cloud of a multi-storey building, segment it into several point clouds of individual storeys.

– Given a point cloud of one single storey of a building, create a basic digital twin that contains space-bounding elements of buildings.

– Given images taken in the same area, add small-scale but still important objects to the created digital twin and further enrich it with useful text information extracted from images.

Based on the process to create an information-rich digital twin, the author summarises the research questions of the thesis as follows.

RQ1 As AI has been used to solve various problems in different domains in recent years and provides possibilities to reduce human effort in many applications, how to extract and select reliable information from AI approaches in processing point cloud data.

RQ2 As there is usually strong occlusion (for example, caused by furniture) existing in the indoor environment of buildings in the real world which causes some surfaces of space-bounding elements to be missing or incomplete, how to improve the result of extracting the target elements in an occluded environment.

RQ3 As are two types of buildings depending on whether they fulfil the Manhattan-world assumption, how to extract the space-bounding elements for buildings that fulfil the assumption as well as those that do not.

RQ4 Based on the experiment result of point cloud segmentation where AI works better for classes of large-scale objects (details can be found in Section 5.2), how to use images as an additional data source (like images) to improve the reconstruction result.

RQ5 As some text information (like object IDs) is useful in a digital twin of a building, how to further enrich the digital twin with text information?

Chapter 3

Proposed Solution

In this chapter, the overall proposed approaches to create an information-rich digital twin are presented. This process can be roughly divided into the following steps:

- collect data with different capturing devices, like laser scanners and cameras.
- extract elements that bound individual spaces, like walls, ceilings, etc.
- detect and reconstruct small-scale objects like smoke alarms, light switches, etc.
- extract useful semantic information like object IDs, serial numbers, etc.

This thesis focuses on Step 2 to Step 4, and the first step of data collection is not in the scope. Figure 3.1 illustrates the overall proposed pipeline. The input to the proposed approach includes laser-scanned point clouds and images or videos captured in the same environments. Both data sources (images and point clouds) are used in the proposed approach. Laser-scanned point clouds are used as input to extract space-bounding elements which are usually large structural elements that form spaces in a building. As the detection results of detecting small elements in point clouds are as good as large elements (shown in Section 7.2.2 in detail), images are used as the complementary data source to detect and recognise small elements. After getting the labelled point clusters with semantic information for space-bounding elements and small elements (the two blocks on the top in Figure 3.1), simplified meshes are generated for those elements (the block at the bottom). Subsequently, texts are detected and recognised to enrich the created digital twin further.

The following parts of this chapter are organised as follows: The scope of the proposed solution is shown in Section 3.1. The overall solution for the whole thesis is presented in Section 3.2. The hypotheses are shown in Section 3.3.

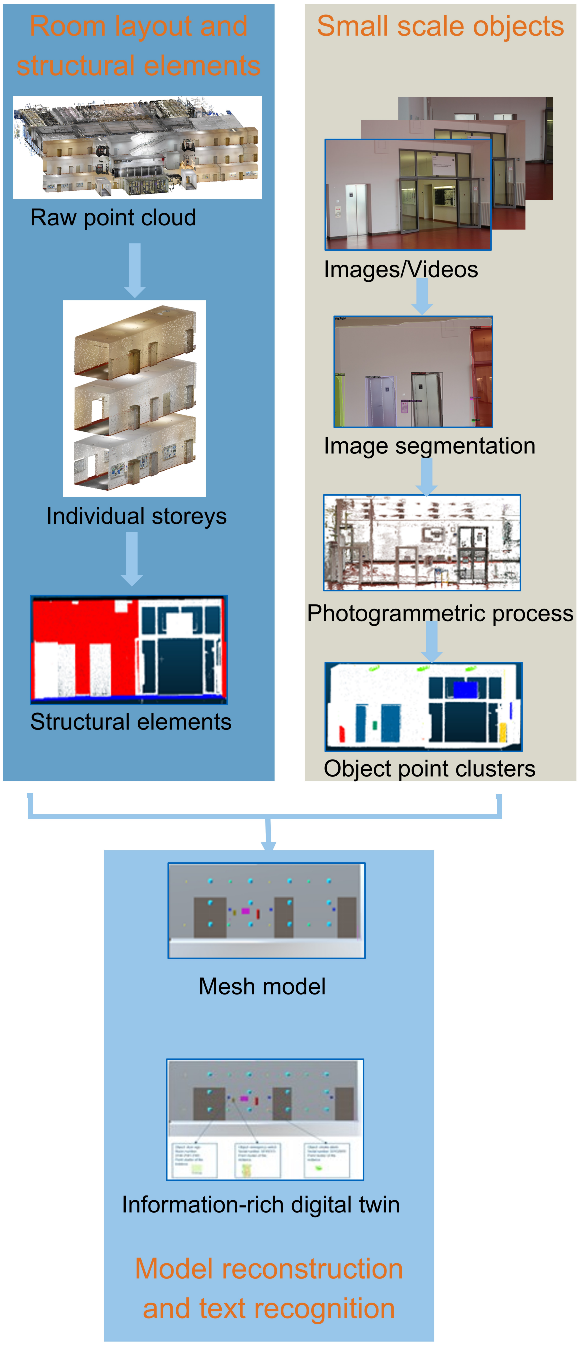


Figure 3.1: Proposed approach to create an information-rich digital twin

3.1 Scope of the Thesis

As discussed in Section 1.1, the concept of the digital twin is manifold. In this thesis, the proposed solution focuses on geometric and semantic information of different objects in a building. The research scope of the thesis is also presented in three parts: object categories, geometric information, and semantic information.

Object Category

With regard to categories of object classes, the following classes are considered in the thesis:

- space-bounding objects, including floors, ceilings, and walls.
- enriched objects, including windows, doors, columns, light switches, emergency switches, light fixtures, smoke alarms, escape signs, speakers, fire extinguishers, sockets, pipes, boards, door signs, elevator buttons, and trash bins.

The objects that form spaces in a building are included (walls, ceilings, floors). In addition, it also includes those objects that are important in facility maintenance and management, such as fire safety equipment (smoke alarms, fire extinguishers, fire alarm switches), electrical elements (light switches, light fixtures, speakers), etc.

Geometric Information

With regard to geometric information, the author considers detailed and simplified geometry for all objects. Both detailed and simplified geometric information is necessary to fit various applications. Therefore, the author decides to keep both two versions in the created output. While the detailed geometric information of one object is represented by the clustered point cloud for the object, the simplified geometric information is represented by a simplified mesh model for the corresponding object.

Semantic Information

With regard to semantic information, the author refers to two different kinds of information. The first one is the label of an object, which represents the object category. The second kind of semantic information is the information detected and recognised from texts (like object IDs, room numbers on the door sign, etc.), which is used to enrich the created output of the solution further.

3.2 Overall Solution

This section presents the overall proposed solution to create an information-rich digital twin. The input to the solution is the captured laser-scanned point cloud of a multi-storey building and images of the corresponding environments. By running through the proposed pipeline, the output is the information-rich digital twin, which includes both large-scale space-bounding elements and small elements, detailed and simplified geometric information, and semantic information. The individual steps of the process are presented later in the thesis, from Chapter 4 to Chapter 7. This whole process is illustrated in Figure 3.2.

As shown in the figure, different methods are designed for two data sources. Given the input of a laser-scanned point cloud of a multi-storey building, the first step is to segment the point cloud into individual storeys (Chapter 4). Subsequently, semantic segmentation is applied by deep learning on point clouds of individual storeys (Section 5.1.1). In this step, point clouds with predicted labels are generated. As the deep learning on point cloud to recognise small-scale elements does not perform as well as that to recognise large-scale elements (shown in Section 7.2.2 in detail), only predictions on those space-bounding elements are used for the next steps. Depending on whether the indoor environments fulfil the Manhattan-world assumption, two methods are proposed for individual cases (Section 5.1.2 to 5.3 for environments that fulfil the assumption and Chapter 6 for environments that do not fulfil the assumption). Simplified mesh models of space-bounding elements are created for the environments at this step.

For the other input of images of the environments, semantic segmentation on images is applied. The extracted information from images is then mapped to laser-scanned point clouds of individual storeys. In this way, the semantic information is transferred from the 2D image plane to the 3D space, and this process is presented from Section 7.1.1 to 7.1.5. Based on the extracted point clusters for small objects, pre-defined mesh models are fitted to those clusters to get lightweight simplified mesh models (Section 7.1.6 to 7.1.7). As images and point clouds are already aligned together, other information in images can be extracted from the 2D plane and mapped to the 3D space without much extra effort. Texts (like object IDs, room numbers, etc.) next to objects in images are detected, recognised, and then linked to the objects. This step is presented in Section 7.1.8.

The final output of the proposed solution contains extracted information from these described steps. Roughly speaking, while the large-scale space-bounding elements are extracted from laser-scanned point clouds, the small elements are extracted from images. The labels extracted from point clouds and images for different objects are selected and combined together. These labels are considered semantic information in the information-rich digital twin. The point clusters represent the detailed geometry of the corresponding objects. Then simplified meshes reconstructed from point clouds and images are combined and represent the lightweight geometry of the environments. Extra semantic information, recognised information from texts in images, is used to further enrich the created output.

The individual steps are presented in the following chapters in detail. The proposed approach to segment point cloud data of a multi-storey building into individual storeys is presented in Chapter 4. While the method to reconstruct environments that fulfil the Manhattan-world assumption is shown in Chapter 5, Chapter 6 presents the proposed solution for buildings that do not fulfil the Manhattan-world assumption. The approach that combines 2D and 3D detection in images and point clouds to detect small-scale objects is presented in Chapter 7.

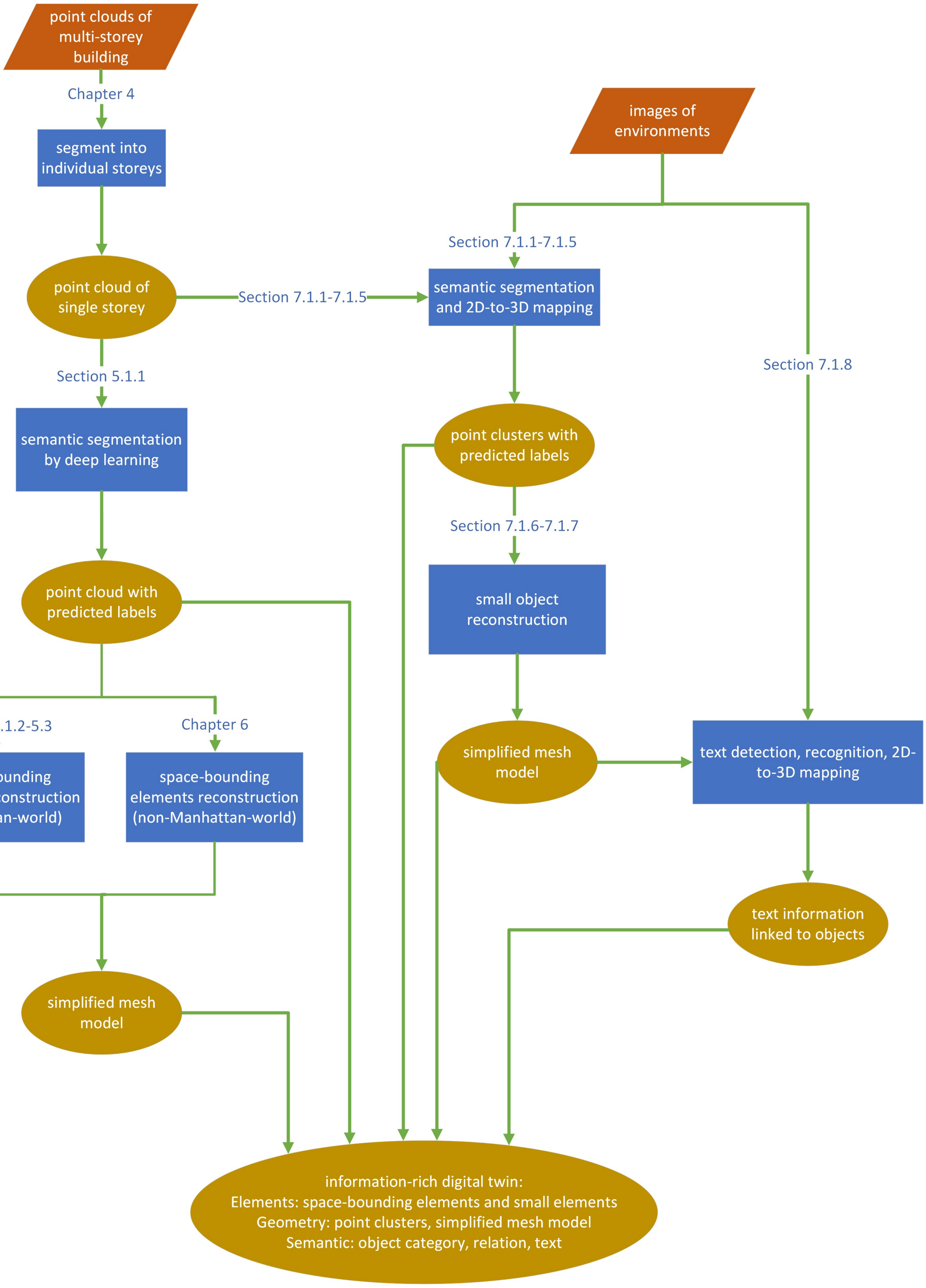

point clouds of multi-storey building
Chapter 4
segment into individual storeys
point cloud of single storey
Section 7.1.1-7.1.5
Section 5.1.1
semantic segmentation by deep learning
point cloud with predicted labels
Section 5.1.2-5.3
Chapter 6
space-bounding elements reconstruction (Manhattan-world)
space-bounding elements reconstruction (non-Manhattan-world)
simplified mesh model
images of environments
Section 7.1.1-7.1.5
Section 7.1.8
semantic segmentation and 2D-to-3D mapping
point clusters with predicted labels
Section 7.1.6-7.1.7
small object reconstruction
simplified mesh model
text detection, recognition, 2D-to-3D mapping
text information linked to objects
information-rich digital twin:
Elements: space-bounding elements and small elements
Geometry: point clusters, simplified mesh model
Semantic: object category, relation, text

Figure 3.2: Diagram of the overall proposed solution

3.3 Hypotheses

The hypotheses underlying this thesis are summarised as follows.

- Given a point cloud file of a multi-storey building, some features of points (such as density) in the point cloud provide the possibility to segment the whole point cloud into individual storeys.
- AI-based methods, especially deep learning on point clouds can improve the process of reconstructing space-bounding elements in occluded indoor environments that do or do not fulfil the Manhattan-world assumption.
- Apart from point clouds, images can be considered as an additional data source that provides complementary information obtained by deep learning, adding small-scale objects into the digital twin.
- By registering images with point clouds together, information recognised from texts in images can be extracted and linked to corresponding elements.

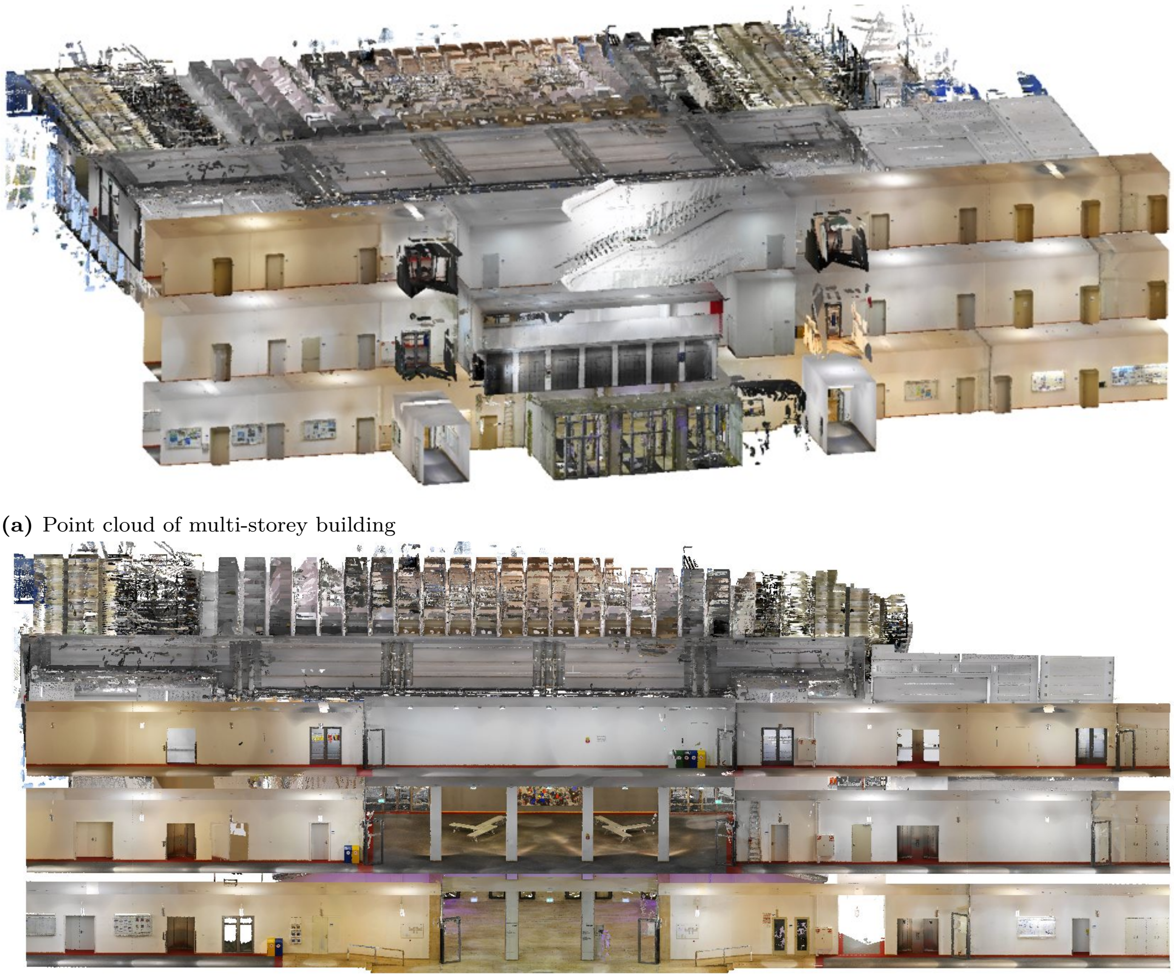
Chapter 4

Segmentation of multi-storey point clouds into different storeys

In this chapter, point clouds captured in multi-storey buildings are segmented into individual storeys. With modern laser scanners and the corresponding software, it is possible to represent a multi-storey building with one single file of the point cloud. A point cloud captured in a multi-storey building captured at the city centre campus at the Technical University of Munich (TUM) by a Navvis [1] laser scanner is shown in Figure 4.1.

There are several advantages of using one single file of a point cloud of buildings: 1) it can show the structure of the whole building more clearly; 2) it is easier to share and transfer data among all stakeholders of the facility with one single file. However, the hardware requirements for visualising and processing large files are very high because the number of points becomes very high. The point cloud shown in Figure 4.1 contains more than 300 million points. In addition, it is more complicated to extract useful information from the more complicated 3D environment that consists of multiple storeys. Therefore, it is very straightforward to propose segmenting the multi-storey point clouds into individual storeys before any further point cloud processing steps. The problem is narrowed down to process point clouds of individual storeys after applying the storey segmentation.

[1] www.navvis.com



(a) Point cloud of multi-storey building

(b) Point cloud of a multi-storey building (remove front surfaces for better visualisation)

Figure 4.1: Point cloud of a multi-storey building captured at TUM city centre campus (data captured by Baro Pfannenstein GmbH / Vokal + Partner Beratende Ingenieure mbB)

4.1 Proposed Methods and Experiments

In this chapter, the storey structure of buildings is divided into two categories: simple structure and complicated structure. The simple structure of building storeys means that the ceiling and floor height is identical for the whole storey. However, because of indoor decoration, like different designs of suspending ceilings, the ceiling height could be differentiated in one storey. For example, the ceiling height of an office could be different from that of the hallway in an office building. Different methods are proposed to segment the simple and complex storey structures.

4.1.1 Storey with Identical Ceiling Height

A part of a building with identical ceiling height is visualised in Figure 4.2. This part of the point cloud is used as input to explain how to segment the multi-storey building with a simple structure into individual storeys.

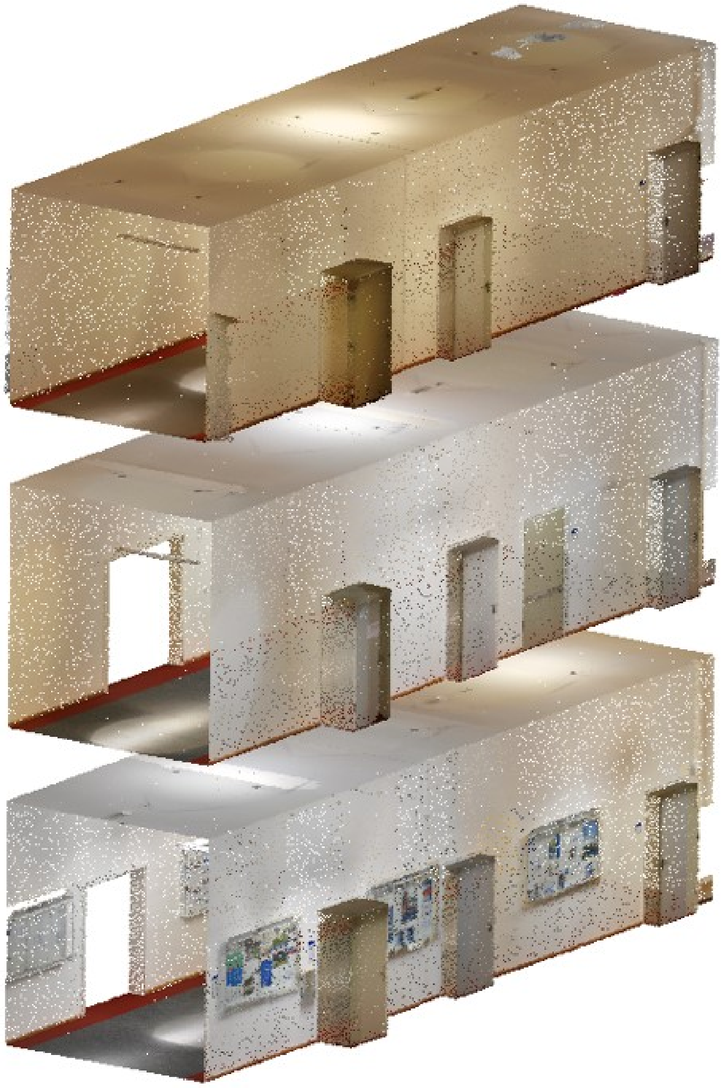


Figure 4.2: Part of building with identical height for individual storeys

For buildings in which storeys have identical ceiling heights, the position of the ceiling and floor of each storey can be identified clearly by analysing the z-coordinate of all points. The positions where the ceiling and floor planes are located are usually significantly higher in the distribution along the z-axis. In Figure 4.3, the point cloud for the facility shown in Figure 4.2 is colour coded by the z-coordinate of the points is presented, and the corresponding point distribution along the z-axis is illustrated. It is obvious to see that there are, in total, six peaks in the distribution. In this case, these six peak values in the z-axis are the height values for the floor of the ground floor, the ceiling of the ground floor, the floor of the first floor, the ceiling of the first floor, the floor of the second floor, and ceiling of the second floor. Different storeys can be segmented clearly by these peak values in the z-axis.

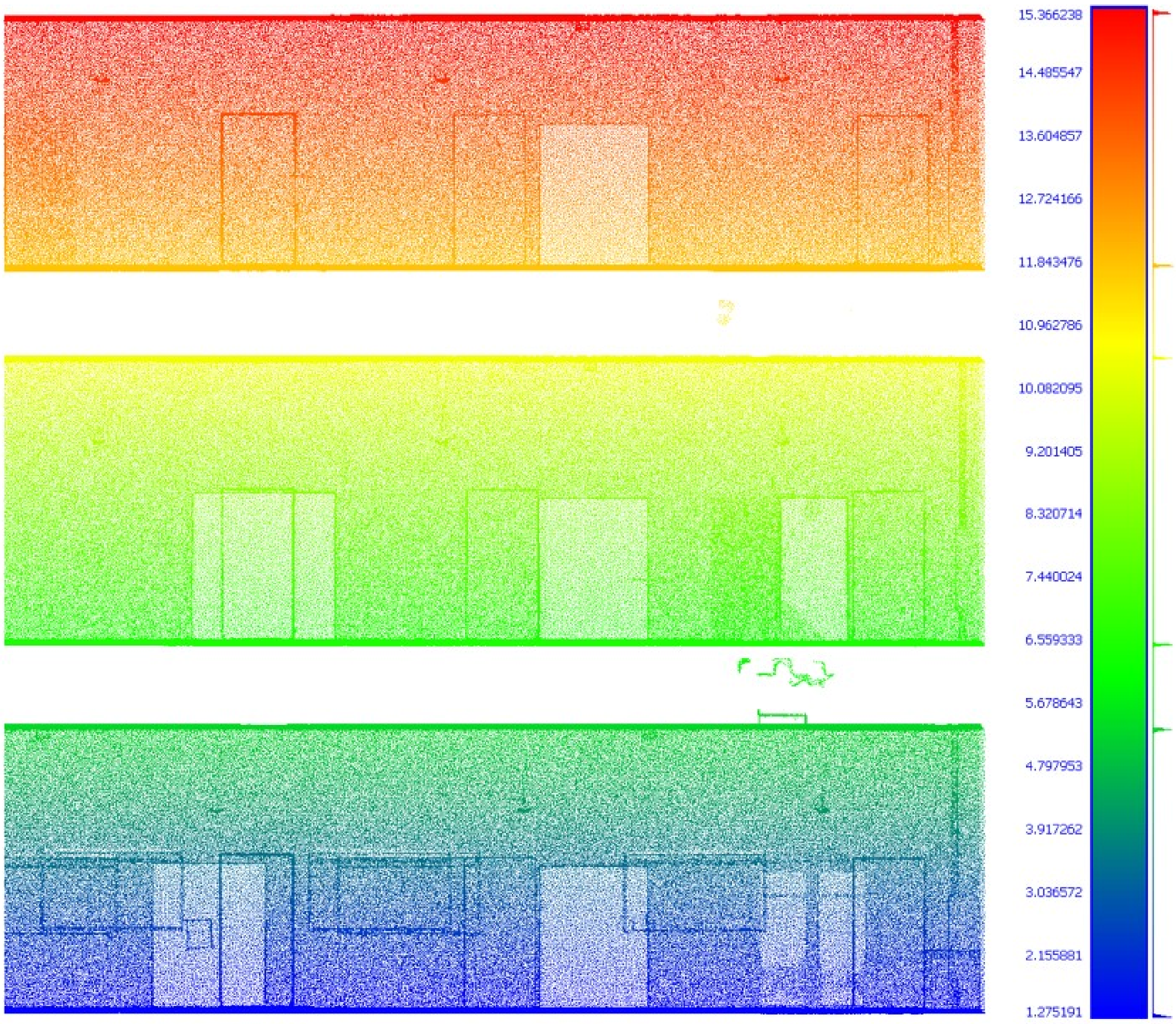
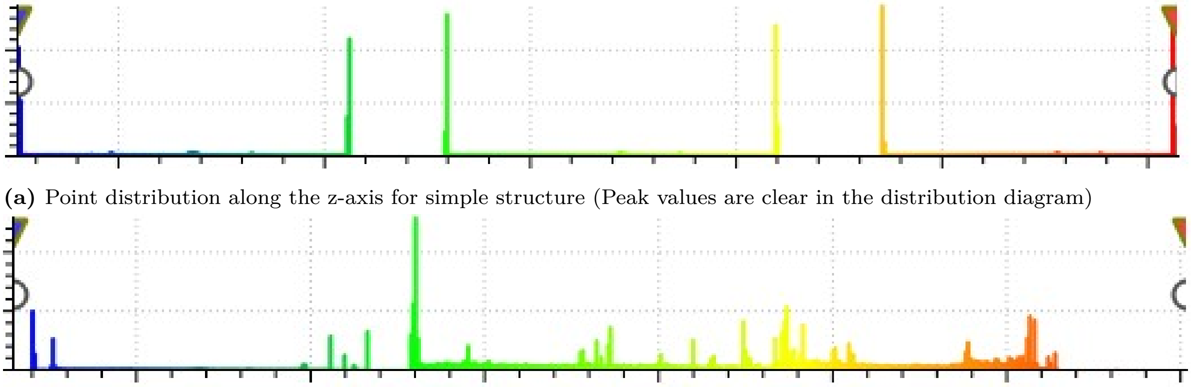


Figure 4.3: Point cloud distribution along its z-axis. The point distribution diagram shows six peak values for three storeys, as each storey contains a ceiling and a floor peak value.

However, if the storey height is not the same for each individual storey, the same strategy does not work. In Figure 4.4, the point distributions along the z-axis for multi-story buildings with simple and complex storey structures are compared (for point clouds shown in Figure 4.2 and Figure 4.1). The reason why the coordinate range number is omitted is that only the distribution rather than the exact values is important here. It can be seen that, unlike the distribution for simple storey structure, which has very clear peak values (in this case, three pairs of peak values for three storeys), the peak values in the distribution for complex storey structure are not clear and organised enough. It is almost impossible to tell where these floor or ceiling planes are located. Therefore, different strategies need to be applied to segment the point cloud with a more complex storey structure.



(a) Point distribution along the z-axis for simple structure (Peak values are clear in the distribution diagram)

(b) Point distribution along the z-axis for complex structure

Figure 4.4: Distribution comparison between multi-story buildings with simple and complex storey structure

4.1.2 Storey with Varying Ceiling Heights

In this section, an approach that aims to segment multi-storey buildings with different ceiling heights in individual storeys is proposed. It should be noticed that the approach is under the assumption that the ceilings and floors in the environment are horizontal. An example of a point cloud is shown in Figure 4.5, where the ceiling and floor heights change in each storey.

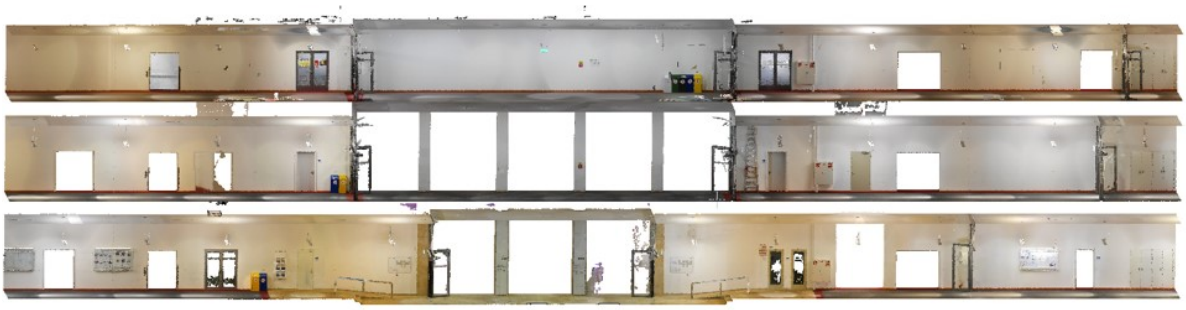


Figure 4.5: Point cloud of building with different ceiling heights in individual storeys

Although the point distribution along the z-axis described in Section 4.1.1 for these storeys is unclear, the local information along the z-axis is still valuable and can be used to identify the horizontal planes that could separate floors and ceilings. In order to extract the point height distribution along the z-axis in a local area (compared with distribution for all points), a grid design where the grid size is not identical in three directions is proposed. r_x, r_y, and r_z are used to denote the grid length in the x-direction, y-direction, and z-direction, respectively. As shown in Figure 4.6, the box sizes in the x-direction and y-direction stay the same and are moderated to the value larger than the size in the z-direction. The designed relationship among r_x, r_y, and r_z are described by the following equation:

$$r_x = r_y = n_1 \cdot r_z, \tag{4.1}$$

where n_1 is the ratio between the box length in the z-direction and the lengths in the other two directions. The parameter study with regard to choosing n_1 is discussed in Section 4.2. All points in the point cloud are located in their corresponding grids. The number of points inside each grid is counted, and the value is denoted by N. By this kind of "thin" local box design, the points that belong to horizontal planes are located in a grid with a relatively large N value and will subsequently be distinguished by the value N.

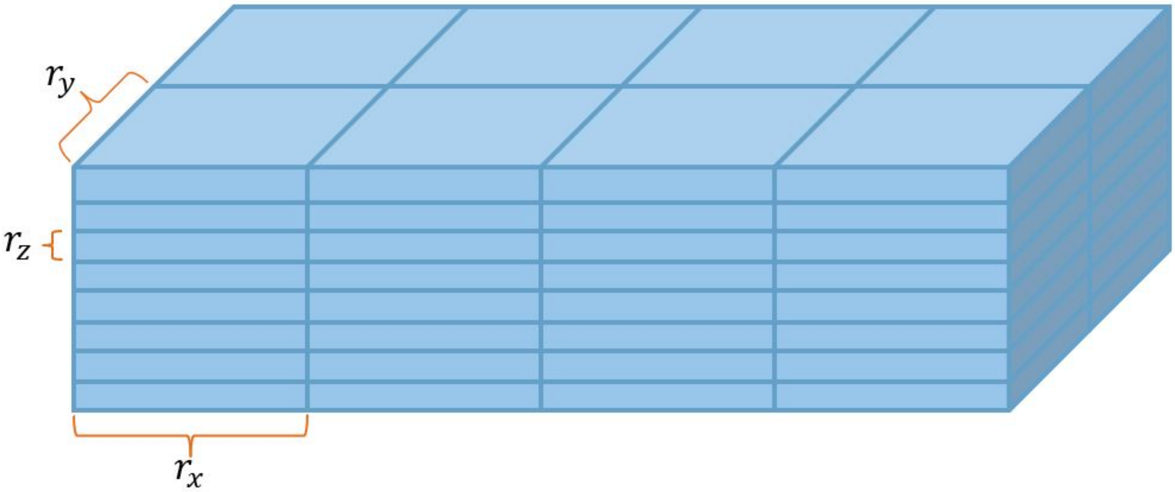


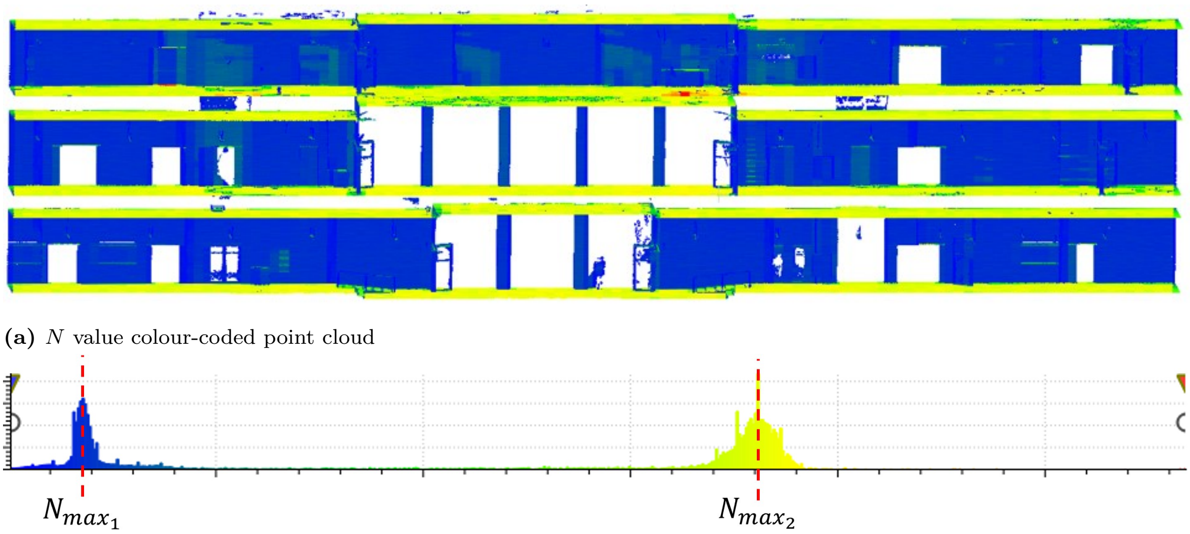
Figure 4.6: Different grid size in three directions

The colour-coded point cloud (shown in Figure 4.5) based on different N values is presented in Figure 4.7a. Meanwhile, the point distribution is illustrated in Figure 4.7b. It can be seen that points which belong to the ceiling or floor can be distinguished from other points in all storeys. And when looking into the distribution of all points, two significant "clusters" can be found clearly (one for points belonging to the vertical plane and one to the horizontal plane).

Subsequently, in order to filter the points which belong to the first cluster out, the following condition is applied:

$$N < \frac{N_{max_1} + N_{max_2}}{2}, \tag{4.2}$$

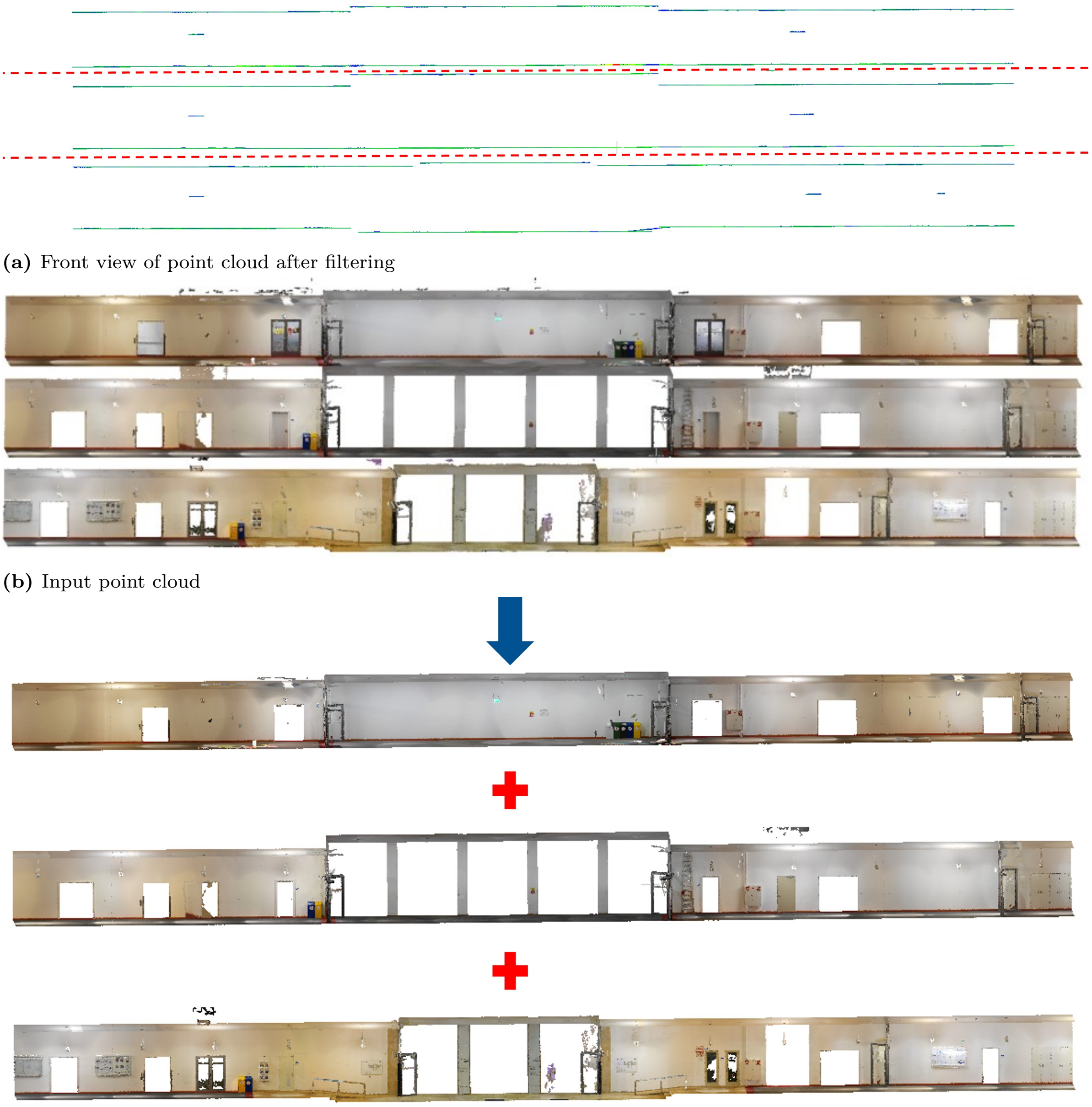
where N_{max_1} is the N value for the maximum value in the first cluster, and N_{max_2} is the N value for the maximum value in the second cluster, as marked in Figure 4.7b. The filtered point cloud is shown in Figure 4.8a. It can be seen that points belonging to the ceiling and floor can be extracted. Based on the prior knowledge that ceilings and floors in multi-storey buildings always are vertically separated by each other, points belonging to different storeys can be segmented accordingly (as marked by the red dashed lines). The final result is illustrated in Figure 4.8c.



(a) N value colour-coded point cloud

(b) Point distribution for N value

Figure 4.7: N value colour-coded point cloud and distribution



(a) Front view of point cloud after filtering

(b) Input point cloud

(c) Storey segmentation result

Figure 4.8: Segment multi-storey point cloud by filtered points of horizontal plane

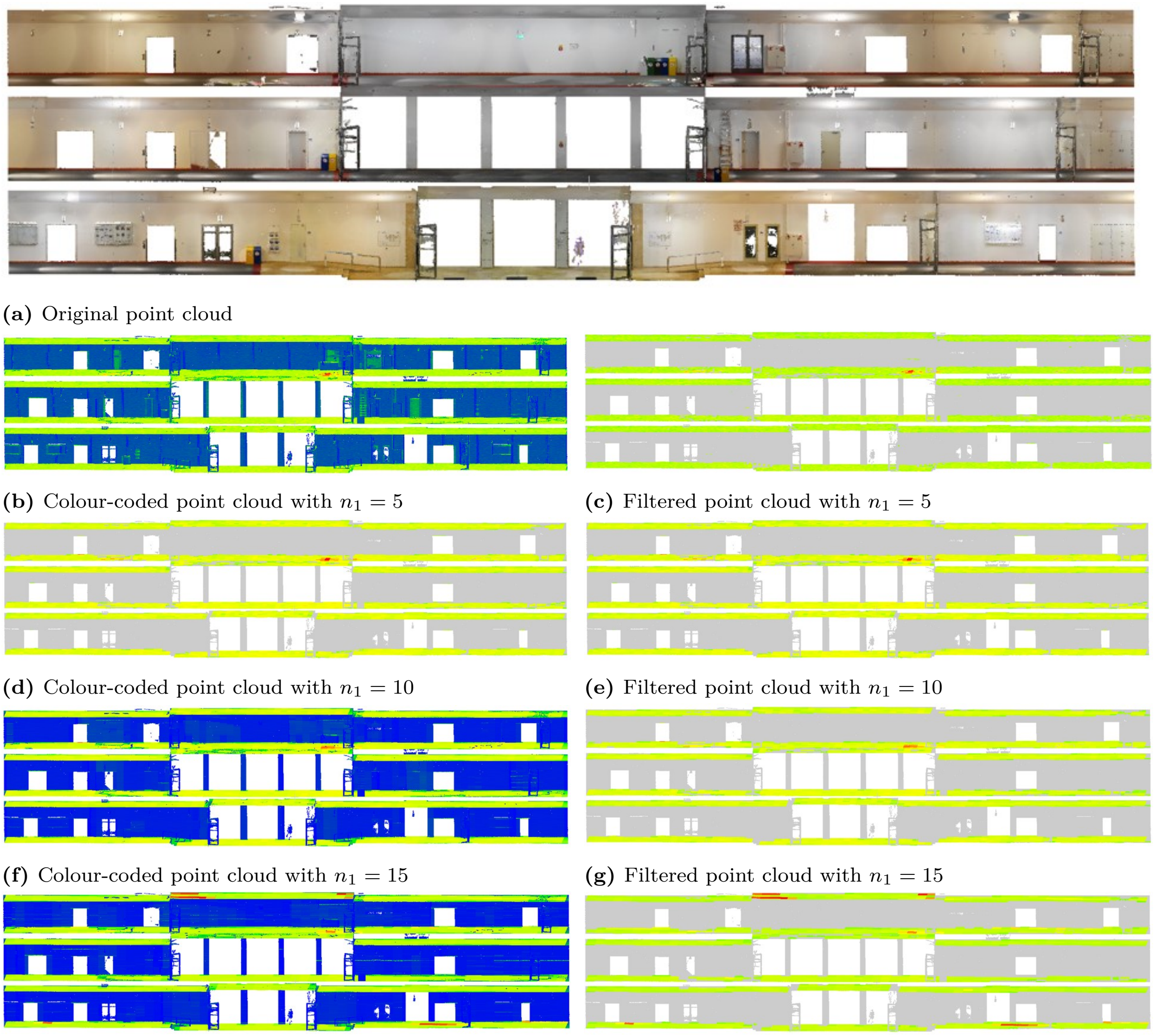
4.2 Results and Conclusions

In order to segment horizontal planes, which could be ceilings and floors from the whole point cloud, the local information is concatenated from the x- and y-direction by the designed voxel length. In this section, different results of choosing different n_1 values are compared.

Some results of different point clouds are shown in Figure 4.9, 4.10, and 4.11. And the corresponding original point clouds are presented in Figure 4.9a, 4.10a, 4.11a. It can be seen that the point clouds of the first two examples are cleaner, and not much occlusion or furniture exists. In this case, all chosen n_1 values can provide good results to segment the multi-storey point clouds. However, things get more complicated if some surfaces of the furniture are horizontal. For example, the furniture surfaces are marked in red circles in Figure 4.11. If only a small value is chosen (like $n_1 = 5$), it is really hard to distinguish the furniture surfaces from floor or ceiling surfaces. It is obvious to see that the floor and ceiling surfaces are better recognised when increasing the n_1 value.

The chosen value of n_1 determines how much more local information you want to extract from the horizontally neighbouring region than the vertically neighbouring region. For point clouds without much occlusion or large horizontal surfaces of furniture, it is not very strict to choose the range of n_1. But for scenes with those surfaces which could probably deteriorate the result, it is important to choose a proper value. If the value is too small, it can not distinguish smaller furniture surfaces from all horizontal surfaces. By enlarging the value, some horizontal surfaces of furniture can be filtered out. But in the meantime, if the value is too large, some small surfaces of the ceiling or floor would also be discarded.

In summary, point clouds of a multi-storey building are segmented into individual storeys in this chapter. The proposed method can handle both simple structures (individual floors have identical floor and ceiling heights) and complex structures (floor and ceiling heights can be differentiated in an individual storey). In this way, the problem can be narrowed to individual storeys, which reduces the complexity and benefits the computing.



(a) Original point cloud

(b) Colour-coded point cloud with $n_1 = 5$

(c) Filtered point cloud with $n_1 = 5$

(d) Colour-coded point cloud with $n_1 = 10$

(e) Filtered point cloud with $n_1 = 10$

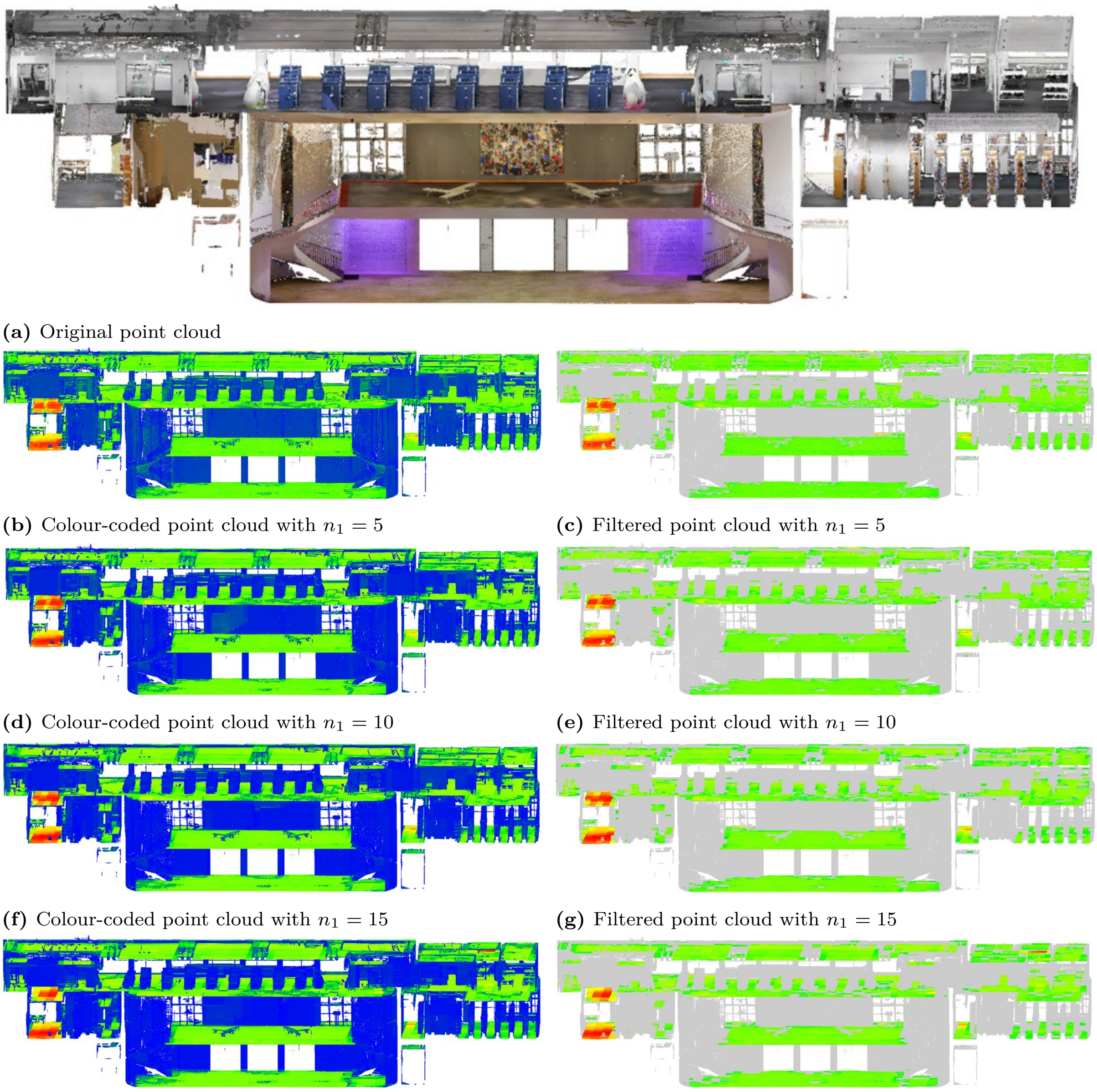
(f) Colour-coded point cloud with $n_1 = 15$

(g) Filtered point cloud with $n_1 = 15$

(h) Colour-coded point cloud with $n_1 = 20$

(i) Filtered point cloud with $n_1 = 20$

Figure 4.9: Storey segmenting results if different n_1 values



(a) Original point cloud

(b) Colour-coded point cloud with $n_1 = 5$

(c) Filtered point cloud with $n_1 = 5$

(d) Colour-coded point cloud with $n_1 = 10$

(e) Filtered point cloud with $n_1 = 10$

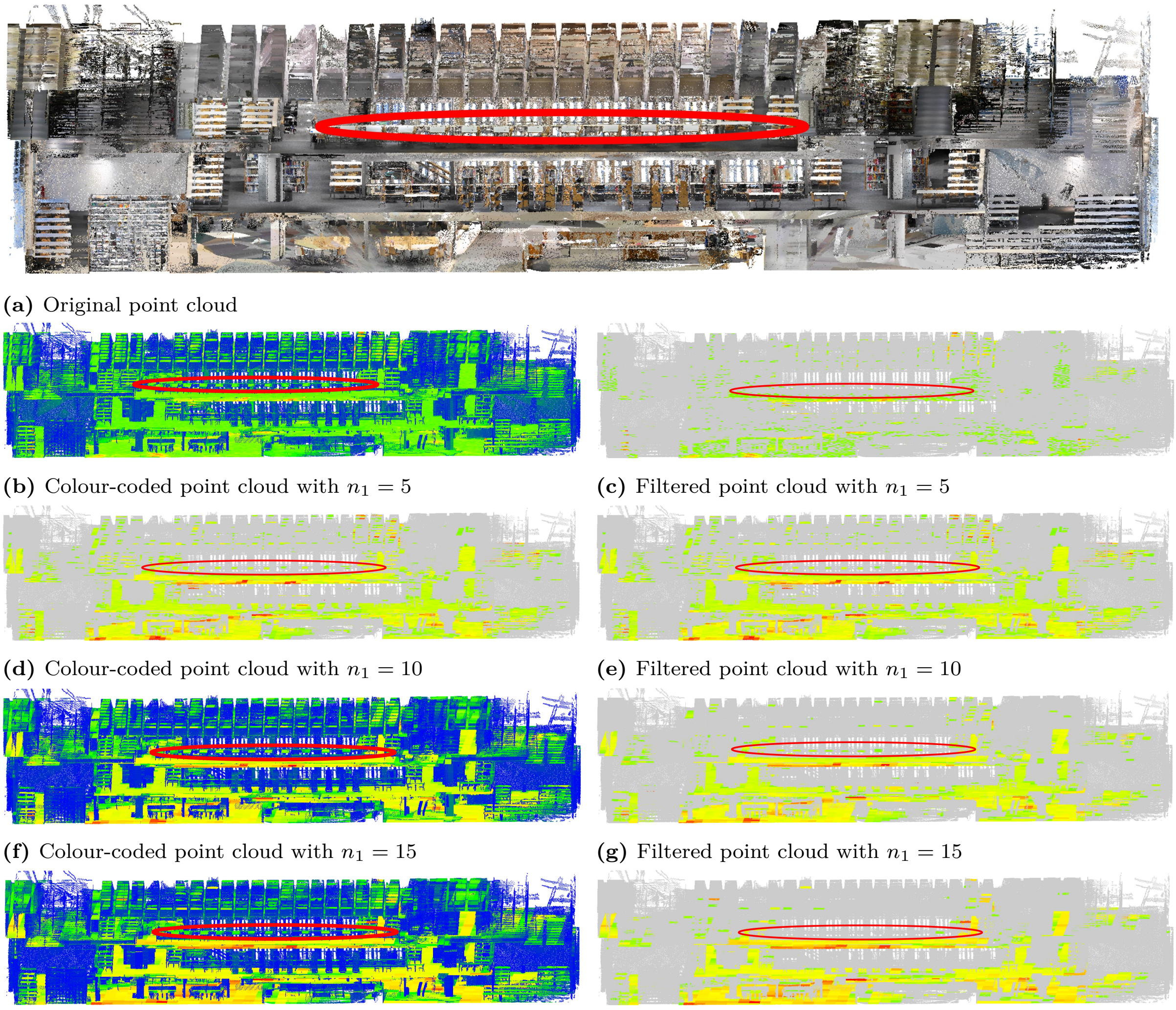
(f) Colour-coded point cloud with $n_1 = 15$

(g) Filtered point cloud with $n_1 = 15$

(h) Colour-coded point cloud with $n_1 = 20$

(i) Filtered point cloud with $n_1 = 20$

Figure 4.10: Storey segmenting results if different n_1 values



(a) Original point cloud

(b) Colour-coded point cloud with $n_1 = 5$

(c) Filtered point cloud with $n_1 = 5$

(d) Colour-coded point cloud with $n_1 = 10$

(e) Filtered point cloud with $n_1 = 10$

(f) Colour-coded point cloud with $n_1 = 15$

(g) Filtered point cloud with $n_1 = 15$

(h) Colour-coded point cloud with $n_1 = 20$

(i) Filtered point cloud with $n_1 = 20$

Figure 4.11: Storey segmenting results if different n_1 values

Chapter 5

Approach for Manhattan-world Environments

In this chapter, an approach of reconstructing space-bounding elements of buildings that fulfil the Manhattan-world assumption is proposed. The Manhattan-world assumption states that there is a predominance of a triple of mutually orthogonal directions in the environment (Coughlan and Yuille, 1999). This chapter is organised as follows: The proposed approach is introduced in Section 5.1 in detail. The experiments and results are presented in Section 5.2. Conclusions and discussions are presented in Section 5.3.

5.1 Proposed Approach

Instead of detecting surfaces in the point cloud first, the proposed approach aims to find the largest void space volume inside each room. Based on the found void volumes, space-bounding elements are reconstructed. Considering the void volumes at room level makes it easier to distinguish the surfaces of building elements from the surfaces of furniture in occluded indoor environments. Moreover, the detection process is improved, especially when adding semantic information from 3D deep learning. The overall proposed approach is illustrated in Figure 5.1.

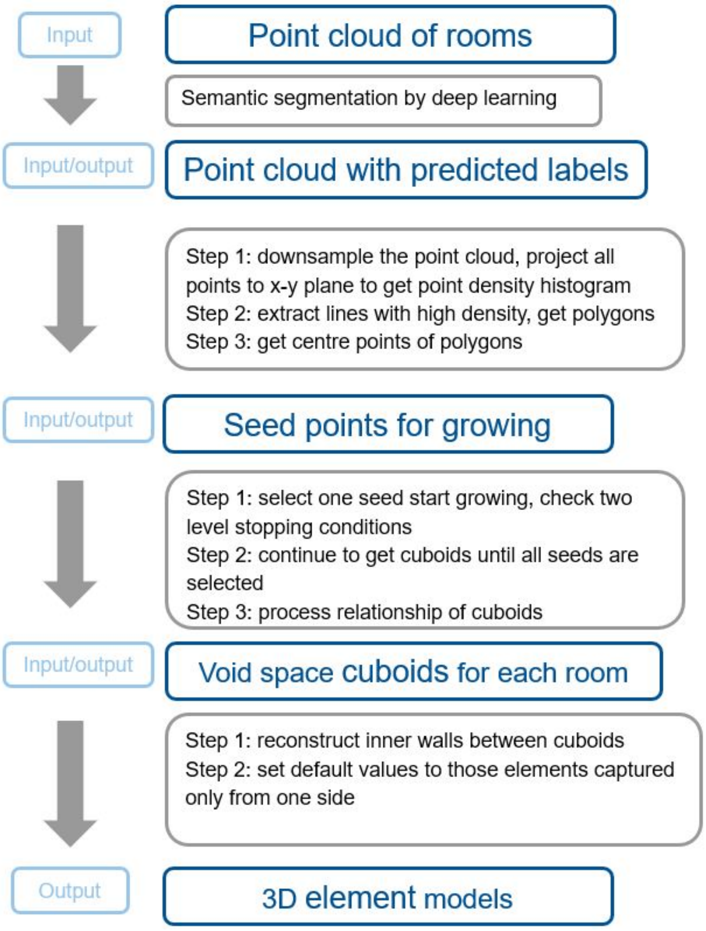

Input
Point cloud of rooms
Semantic segmentation by deep learning
Input/output
Point cloud with predicted labels
Step 1: downsample the point cloud, project all points to x-y plane to get point density histogram
Step 2: extract lines with high density, get polygons
Step 3: get centre points of polygons
Input/output
Seed points for growing
Step 1: select one seed start growing, check two level stopping conditions
Step 2: continue to get cuboids until all seeds are selected
Step 3: process relationship of cuboids
Input/output
Void space cuboids for each room
Step 1: reconstruct inner walls between cuboids
Step 2: set default values to those elements captured only from one side
Output
3D element models

Figure 5.1: Workflow of proposed approach

5.1.1 Point Cloud Segmentation by Deep Learning

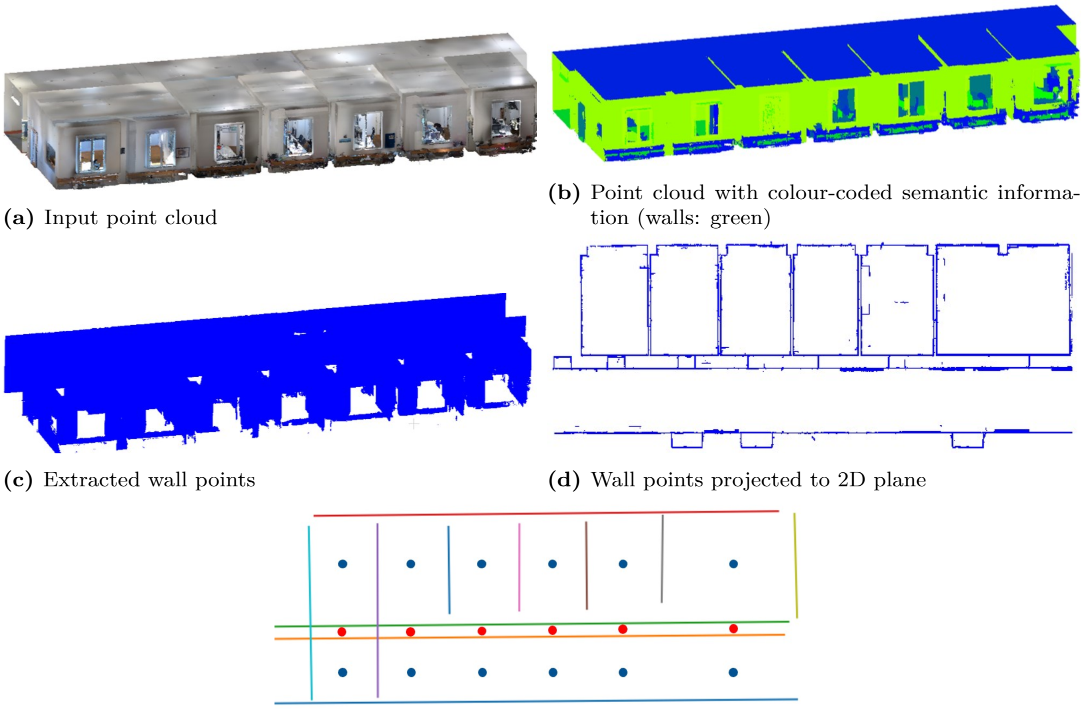
Because some of the important information can be occluded in a building with a high occlusion level, our proposed approach uses semantic information from deep learning is used to enhance performance. As the results on the S3DIS dataset (Armeni et al., 2016) are shown in Table 2.1, KPConv (Thomas et al., 2019) is one of the well-performing neural networks for point cloud segmentation in the indoor environment. As the similar indoor environment of our office building is captured, the author argues that KPconv can also perform well in our dataset.

More information on the dataset preparation and the results of point cloud segmentation are evaluated in Section 5.2.

5.1.2 Generating Seeds for Growing

In this step, the growing seeds for further steps are generated. The proposed method here is under the Manhattan-world assumption. Firstly, the method performs the voxelisation downsampling method to the point cloud with semantic information. All voxels in the voxelisation process are stored, including void voxels and non-void voxels. While void voxels represent empty space in the point cloud, a non-void voxel means there are points within the voxel. In order to find seed points for growing, all points that are predicted as walls by deep learning in the previous section are projected to the XY plane after the voxelisation downsampling. Subsequently, RANSAC is used to extract lines in the XY plane. According to the Manhattan-world assumption (Coughlan and Yuille, 2000), lines that are not parallel or perpendicular to X or Y coordinates are not extracted.

The next step is to extend the lines and calculate the intersecting points of these lines to get polygons. Some points (like those on large vertical surfaces of furniture) could also be predicted as wall points and projected on XY plane. Therefore, these polygons are potential floor plan representations, not real floor plans. Then the centroids of these polygons are computed subsequently and selected as the potential seed points in the 2D plane. In 3D space, a default value needs to be set for each potential seed point. In the experiment, $1.5m$ is used as the Z coordinate for seed points. Because the aim of the proposed approach is to grow a void space inside a room, the voxels of the seed points should be void voxels. If not, the neighbouring voxels can be selected as seeds, as in this step, seed points only need to be selected roughly. It is not necessary to distinguish polygons from different rooms. As described later in the algorithm of the growing method, if a seed point from a polygon is included in the grown space of another seed, it will not grow from this point anymore. This process is illustrated in Figure 5.2.



(a) Input point cloud

(b) Point cloud with colour-coded semantic information (walls: green)

(c) Extracted wall points

(d) Wall points projected to 2D plane

(e) Line extraction by RANSAC and seed point selection (blue points are seed points, red points are discarded because the polygon is too small to be considered as a single room)

Figure 5.2: The process of generating seeds

5.1.3 Growing from Each Seed

This part is the core of the proposed approach. As the aim is to find the largest void space volume inside a room, the final void volume is supposed to meet two requirements: a) it should enclose the points of furniture inside the room; b) it should not expand to the outside through window and door openings.

In this step, the void-growing process is in a "cuboid" way. That means the void space is grown from one seed (void voxel) in six directions (top, bottom, left, right, front, back). Similar to the region growing approach, neighbours of the seed are checked to determine whether they belong to the volume space at each step. In general, the growing process and the 26 neighbours of one voxel in 6 directions are illustrated in Figure 5.3.

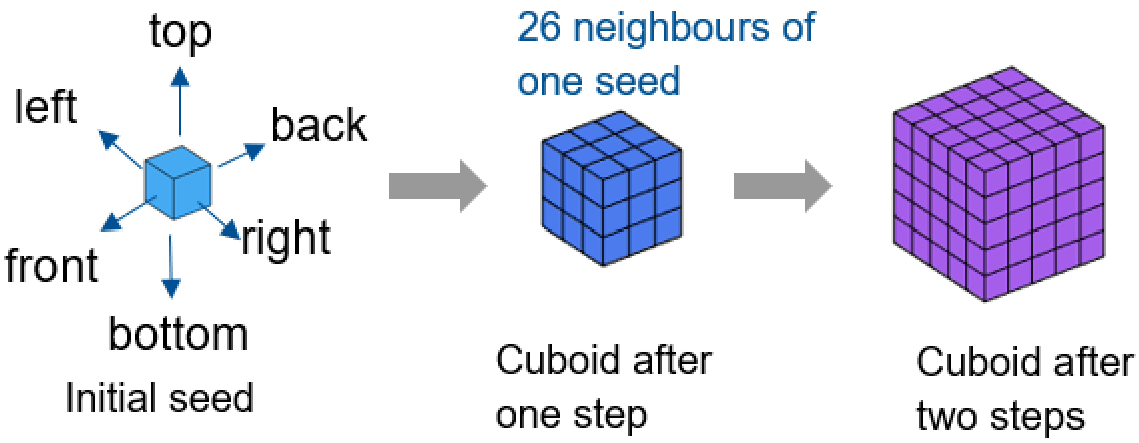


Figure 5.3: Proposed void growing process

It is vital to determine when to stop the growing process. One reason why the growing process is done in a "cuboid" way is that in this growing way, it is straightforward to check the frontiers in six directions during the growing process and use the information on frontiers to determine whether it should stop growing in each direction. Two different kinds of stopping conditions that determine when to stop growing are used: semantic stopping condition and geometric stopping condition.

In the previous step, all voxels from the input point cloud are categorised into two classes: void and non-void voxels. S is used to denote the set that contains all found seeds, and the algorithm picks a seed from S until there are no non-used seeds. This process is introduced as follows, and the pseudocode for the algorithm is shown in Algorithm 1:

– the algorithm checks whether the selected seed has been used before. If so, it would delete this seed and pick another seed;

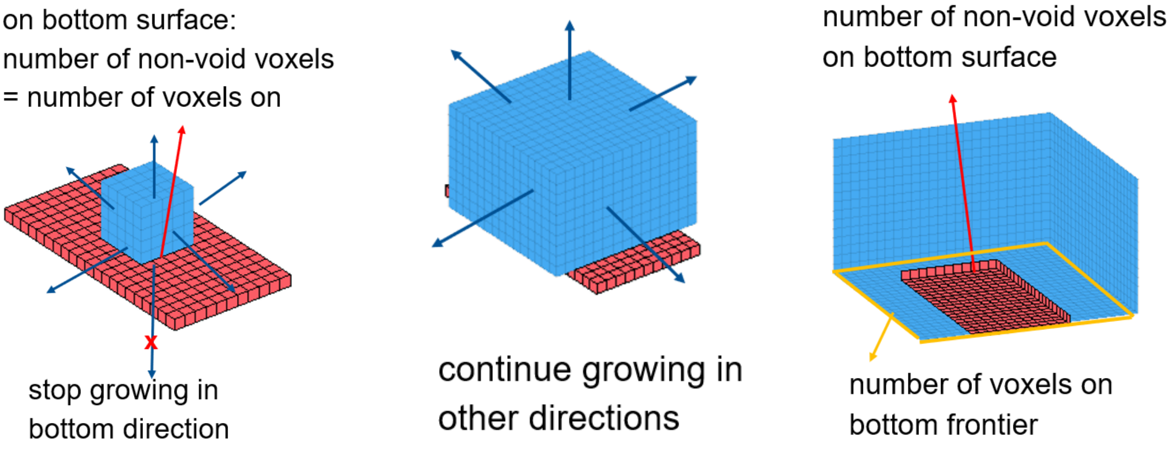
– the picked seed voxel is added to another set which is denoted by N_t. It represents the seed set at step t. N_0 denotes the initial set that contains only one seed from S;

- for every seed voxel in N_t at step t, the algorithm finds its 26 neighbours. The set that contains all neighbour voxels is denoted by P. It represents the potential seeds for the next step. And I use N to represent the union of all previous seed sets, from time step *0* to time step t;
- voxels that are already shown in N are removed from P;
- the algorithm checks whether it fulfils any of the stopping conditions on frontiers (stopping conditions will be introduced later in this chapter). The set of these voxels on the frontier is denoted by E. Only if it stops in all six directions, the growing process would be stopped;
- voxels in E that belongs to the frontier of stopping directions are removed from potential seed set P as it would not grow in the corresponding directions. Moreover, seed points from the previous step in this direction need to be added to the new seed set of the next step. Otherwise, it cannot grow in this direction anymore without the corresponding seeds. The newly generated seed set is denoted by N_{t+1}. Then go back to step 2);

Algorithmus 1 The void growing algorithm.

```
Input:
void voxels and non-void voxels of the point cloud;
initial seed, $S$;
the stopping conditions, $F()$;
functions to find all neighbors, $\alpha()$;
functions to get voxels on the frontier of any direction, $\beta()$;
Initialize:
voxel list of void volume space in the point cloud, $O \leftarrow \varnothing$
Algorithm:
while $S$ is not empty do
    select one initial seed $N_0$ from $S$
    if $N_0 \in O$ then
        while $F(N_t)$ is not fulfilled in six directions do
            find seeds' 26 neighbours $P \leftarrow \alpha(N_t)$
            $P \leftarrow P \setminus (P \cap O)$
            if $F(P)$ is fulfilled in any direction then
                find voxels in that direction $E \leftarrow \beta(N_t)$
                get seeds for next round $N_{t+1} \leftarrow P \setminus E$
                $N_{t+1} \leftarrow N_{t+1} \cup \beta(N_{t-1})$
                $O \leftarrow O \cup N_t$
            end if
        end while
    end if
end while
```

How the growing process works when it grows to a plane is illustrated in Figure 5.4 for an example. Here, the red voxels represent the non-void voxels that form a plane, while the blue voxels are the void voxels. In Figure 5.4a, when it grows to a position in the bottom direction that has many non-void voxels (like the top surface of a desk), at this step, the algorithm is not expected to know whether it grows to a desk surface or a floor surface. So, it stops growing in the bottom direction and continues growing in the other five directions. It will stop growing in the bottom direction until it does not meet the stopping condition in this direction anymore (for example, in Figure 5.4c).



(a) Stop growing in the bottom direction

(b) Continue growing in other directions

(c) Continue growing in bottom direction

Figure 5.4: The first-level stopping condition

5.1.4 Stopping Conditions

In the proposed approach, the stopping conditions are defined to determine when to stop the growing process in different directions separately. Despite the fact that the semantic information predicted from deep learning is very valuable, it is not a perfect result. Therefore, semantic information, as well as geometric information, is used as the stopping conditions. In the growing process, the information on the frontier in each direction is checked to determine whether it fulfils the stopping conditions at every step.

The basic idea is that the algorithm compares the ratio between the number of target points of all directions at every step and a predefined threshold value T (in our experiment, the default value is set $T = 0.1$ to make it does not neglect smaller non-void surface). If the ratio is larger than the threshold T, as there is a relatively large number of non-void voxels or there are some voxels with essential labels (like walls, windows, etc.), the algorithm will stop growing in this direction in this step.

Semantic Stopping Conditions

In semantic stopping conditions, labels predicted by deep learning are considered. For the top direction, a ratio value that is used to check whether it stops growing in this direction is defined as follows:

$$r_{top} = P_{ceiling}/Q_{top}, \tag{5.1}$$

where $P_{ceiling}$ denotes the number of voxels predicted as the ceiling in the top direction, and Q_{top} is the number of nonvoid voxels in the top direction. If $r_{top} > T$, it means there are a number of points predicted as ceiling and the algorithm would stop growing in this direction at this step, and continues checking the stopping conditions in further steps.

Similarly, for the bottom direction, a ratio value is defined as the stopping condition as follows:

$$r_{bottom} = P_{floor}/Q_{bottom}, \tag{5.2}$$

where P_{floor} denotes the number of voxels predicted as the floor in the bottom direction, and Q_{bottom} is the number of nonvoid voxels in the bottom direction.

For the other four directions (left, right, front, back) where windows and doors might exist in these directions, the ratio value is defined slightly differently:

$$r_{other} = (P_{wall} + P_{window} + P_{door})/Q_{other}, \tag{5.3}$$

where P_{wall}, P_{window}, and P_{door} denotes the number of voxels predicted as wall, window, and door in one of the other four directions, and Q_{other} is the number of nonvoid voxels in the same direction.

Geometric Stopping Conditions

As labels from deep learning are not always correct, it is helpful to consider other information rather than relying solely on semantic information. In geometric stopping conditions, the number of void and nonvoid voxels are considered. It aims to stop the growing process where a plane exists, but the points on the plane are wrongly predicted (for example, wall points are predicted as furniture points).

The ratio that determines whether it stops growing in one direction is defined as follows:

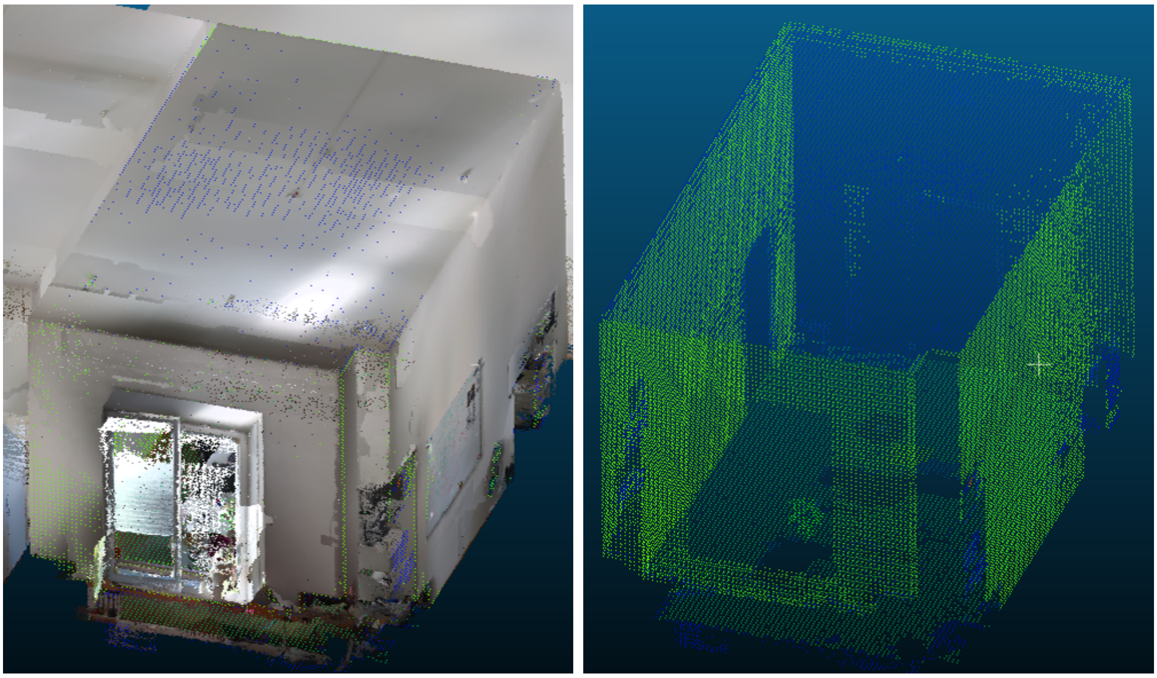
$$r = P/Q, \tag{5.4}$$

where P denotes the number of nonvoid voxels in one direction, and Q is the number of void voxels in the same direction. If $r > T$, it means there are a number of nonvoid voxels; despite that no semantic information is available, the algorithm should stop growing in this direction at this step and continues checking the stopping conditions in further steps.

If any stopping condition (semantic or geometric) is fulfilled in one direction, it stops growing in this direction and continues growing in other directions. Whether it stops growing in one direction does not influence the growth in other directions, and it can still enlarge the frontier of the stopped direction when continuing growing in other directions.

That means in further steps if the stopping condition is not fulfilled anymore, it can continue growing in this direction.

When it stops growing in all six directions, the result is supposed to be the void space, whose top surface is the ceiling, the bottom surface is the floor, and the other four surfaces are the wall surfaces. The growing results where it stops growing for one office room are shown in Figure 5.5. It is obvious to see that all surfaces that form the void space can be well detected.



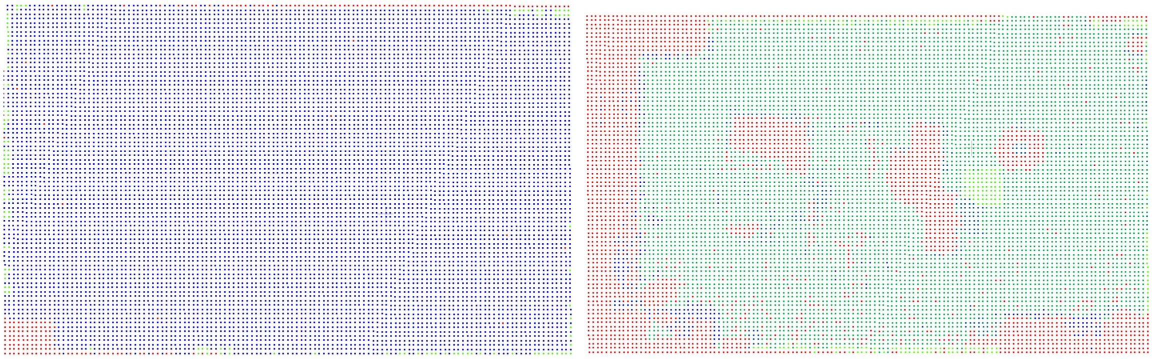
(a) Input point cloud

(b) Growing result where growing stops at room boundary

Figure 5.5: Growing result of an office

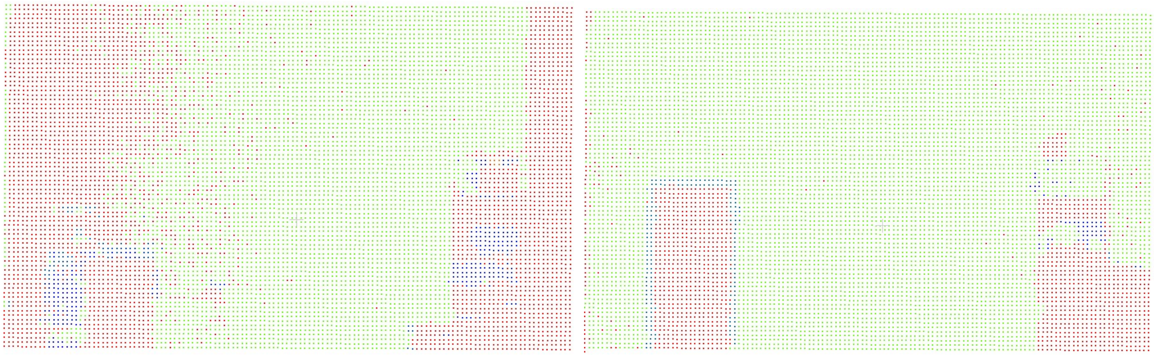
The surfaces where the algorithm stops growing in the bottom and top directions are shown in Figure 5.6. In this figure, red points denote the centre points of void voxels; blue points denote the centre points of voxels predicted as the ceiling; green points represent the centre points of voxels predicted as the floor. The algorithm stops growing when it grows to the ceiling and floor surfaces.

The surfaces where it stops growing in the other four directions are shown in Figure 5.7 and Figure 5.8. In these figures, red points denote the centre points of void voxels; green points represent the centre points of voxels predicted as the wall. The algorithm stops growing when it grows to wall surfaces, despite having door or window openings, because the growth would be stopped if the semantic information of windows and doors is found in the growing process. How to further extract door and window openings in the wall is discussed in Section 5.1.5.



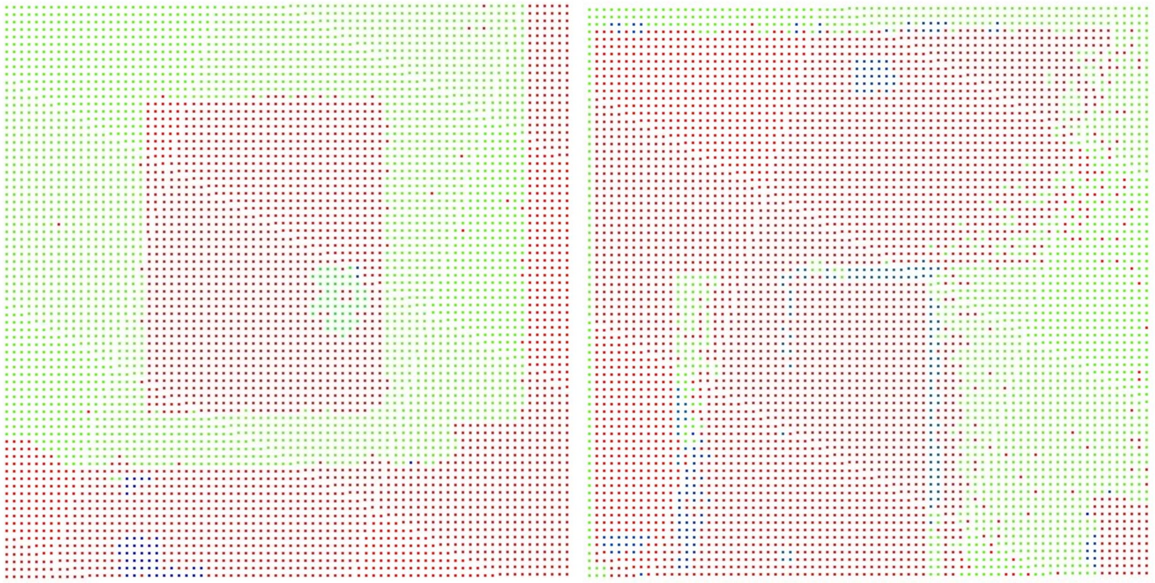
(a) Top stopping surface **(b)** Bottom stopping surface

Figure 5.6: Stopping surfaces in the top and bottom directions (Red points denote the centre points of void voxels; blue points denote the centre points of voxels predicted as the ceiling; green points represent the centre points of voxels predicted as the floor class)



(a) Left stopping surface **(b)** Right stopping surface

Figure 5.7: Stopping surfaces in left and right directions (Red points denote the centre points of void voxels; green points represent the centre points of voxels predicted as the wall; blue points are the centre points of voxels predicted as the door class)



(a) Front stopping surface

(b) Back stopping surface

Figure 5.8: Stopping surfaces in front and back directions (Red points denote the centre points of void voxels; green points represent the centre points of voxels predicted as the wall; blue points are the centre points of voxels predicted as the window class)

5.1.5 Detect Window and Door Openings

The patterns on frontiers in each direction when it stops growing are checked to identify whether there are openings on the wall in the corresponding direction. When capturing buildings with laser scanners, doors are usually open. This and the fact that the window glass surfaces do not reflect the laser beam lead to the fact that windows and doors are void areas without any points (like in Figure 5.9). As the proposed approach considers void space inside the point cloud, it is convenient to check the void area on the wall surfaces where the algorithm stops growing.

A wall surface with a door opening where it stops growing is shown in Figure 5.9. In this figure, red points are the centre points of void voxels; green points represent the centre points of non-void voxels predicted as the wall; blue points are the predicted door points. It is obvious to see that in the marked area, there is a red rectangle (opening) surrounded by blue points (door). Therefore, the door opening rectangle can be extracted by fitting a minimum rectangle that encloses all the points of void voxels.

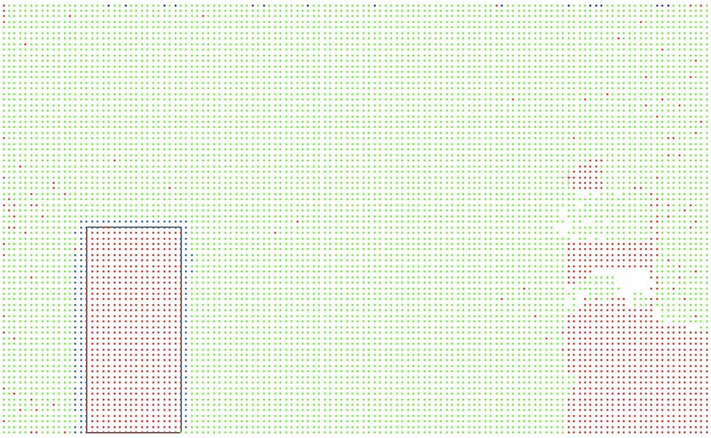


Figure 5.9: A wall surface with a door opening (Red points are the centre points of void voxels; green points represent the centre points of non-void voxels predicted as the wall; blue points are the predicted door points)

A wall surface with a window opening where it stops growing is shown in Figure 5.10. In this figure, again, red points are the centre points of void voxels; green points represent the centre points of voxels predicted as the wall; blue points are the predicted window points. In the marked area, it is obvious that there is a red rectangle (opening) and some window points at the boundary between wall points (green) and centre points of void voxels (red).

However, it would cause a problem if those predicted door/window points were used to extract a rectangle. As shown in Figure 5.9 and Figure 5.10, the predicted window and door points are not perfect rectangles because many points are wrongly predicted as other classes, and this would make the fitting result imprecise. The result for doors and windows (mIoU around 44%) is worse than that of the other three elements (ceiling, floor, and wall's mIoU larger than 80%).

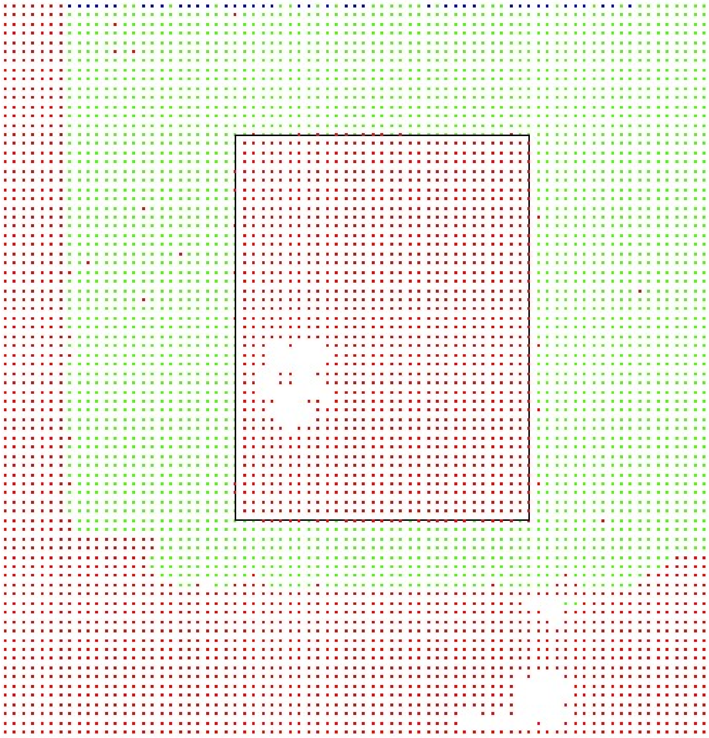


Figure 5.10: A wall surface with a window opening (red points are the centre points of void voxels; green points represent the centre points of voxels predicted as the wall; blue points are the predicted window points)

Therefore, in our proposed approach, semantic information about doors and windows is only used to check the existence of doors and windows, not used in the opening reconstruction. Rectangles are extracted based on geometric information of the void space inside walls under the assumption that all door and window openings are rectangles.

Semantic information from deep learning is quite helpful in distinguishing wall surfaces and furniture surfaces, despite the fact that some furniture surfaces are sometimes very large. These large furniture surfaces are quite hard to identify if using geometric information only. A large bookcase in front of a wall is shown in Figure 5.11. In this figure, red points denote the wall behind the bookcase; blue points are predicted as a bookshelf, while grey points next to the blue points are predicted as closet points. Although the neural network model predicts the points of a bookcase into two classes, it is understandable because closets and bookshelves are sometimes quite similar in real life. But it does not mislabel points of furniture to points of the wall, which is important to determine when to stop growing.

The process of fitting rectangles to find a window opening on the wall is illustrated in Figure 5.12. In the plane where a window exists, the window opening should be close to a rectangle, assuming all windows in the building are rectangles. Apart from the window opening, if the wall is occluded by some furniture which is quite common in the indoor environment, there would also be some furniture void voxels, as shown in Figure 5.12b. But usually, these void spaces have different shapes from windows and

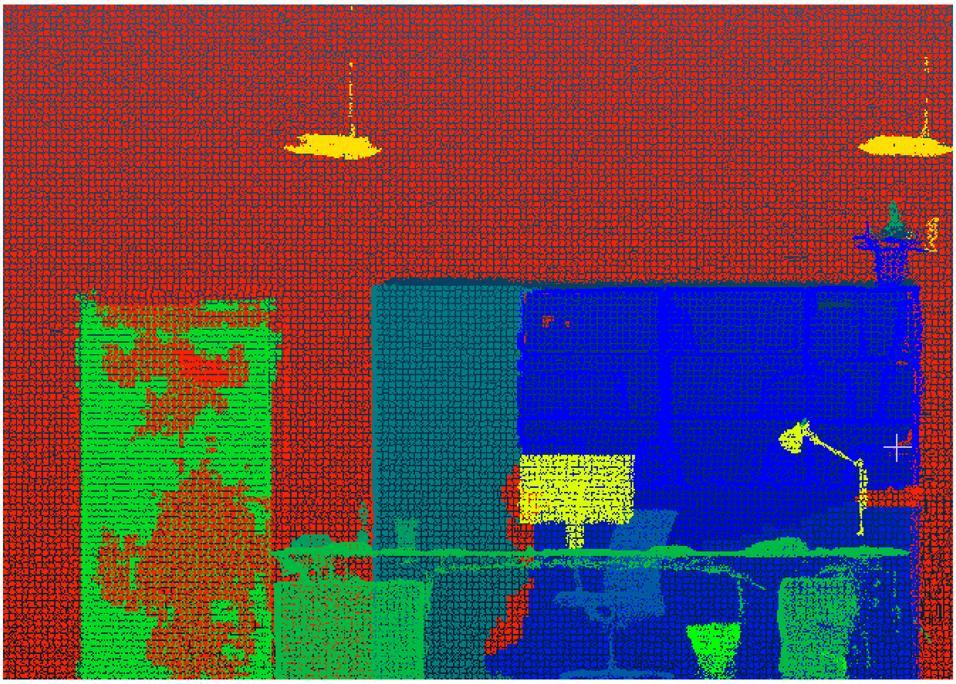
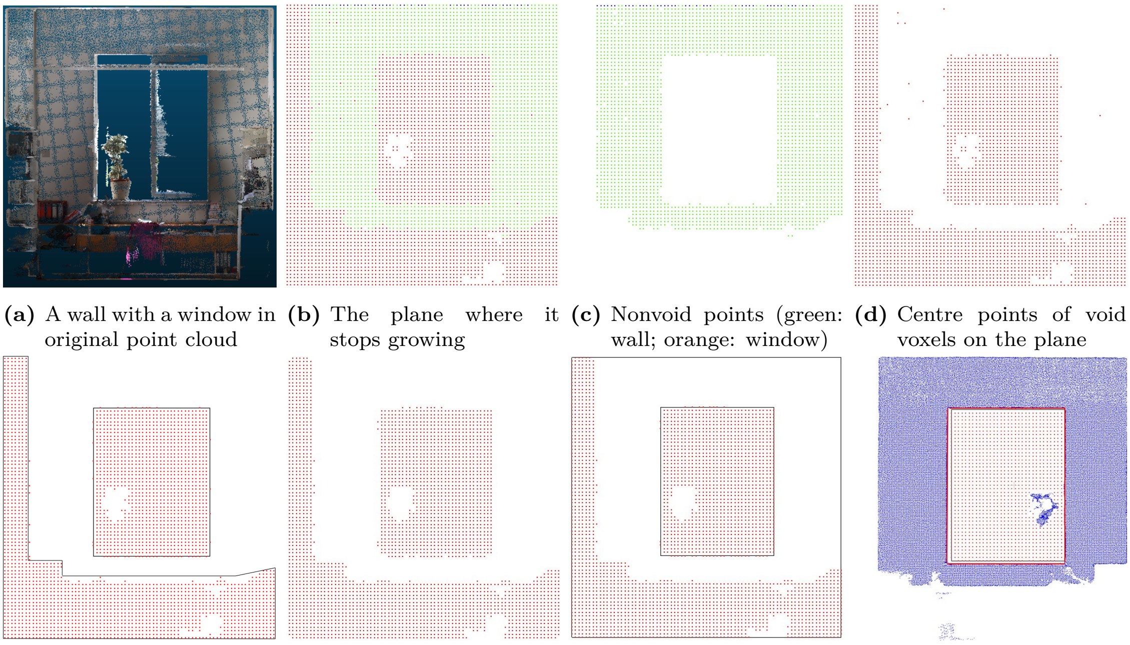


Figure 5.11: The prediction of a bookcase in front of a wall. Points in different colours represent different predicted classes.

doors. Moreover, on the boundary of the window void area and the wall area, there are some points predicted as window points by deep learning, like the orange points in 5.12c. The semantic information of window points can be used to distinguish the void space of a window opening from that caused by furniture. Then all void voxels (as shown in Figure 5.12d) in the plane are clustered into different point clusters based on the Euclidean distances between a point to its neighbours, as shown in Figure 5.12e. The next step is to remove the outliers in each cluster by applying the statistical outlier removal filter in Point Cloud Library (Rusu and Cousins, 2011) (shown in Figure 5.12f). At last, the minimal rectangle that can enclose all points in the cluster is fitted to the point cluster, as shown in Figure 5.12g. Because the larger rectangle in Figure 5.12g does not have any predicted window points on the boundary (not a window based on semantic information) and the void points inside the rectangle do not form a rectangle shape (not a window based on geometric information), the larger fitted rectangle can be removed from a window opening directly. As the dimension of door and window openings are relatively small compared with ceilings and walls, it is sensible to the voxel size that we used to downsample the point cloud. The detected rectangles are fitted again in the original input point cloud, as shown in Figure 5.12h. In the original point cloud, the sides of rectangles are adjusted horizontally or vertically within a voxel range until they fit the boundary of points and empty region.

The process of fitting rectangles to find a door opening on the wall is quite similar and illustrated in Figure 5.13. The main difference is that some points that are predicted as the door rather than the window are considered to detect the door opening.

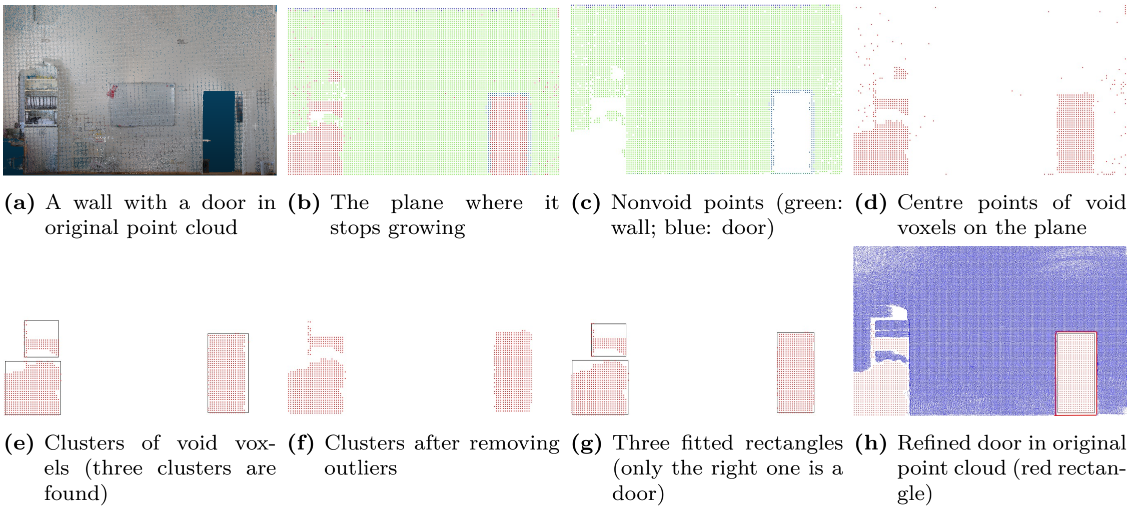


(a) A wall with a window in original point cloud **(b)** The plane where it stops growing **(c)** Nonvoid points (green: wall; orange: window) **(d)** Centre points of void voxels on the plane

(e) Clusters of void voxels (two clusters are marked) **(f)** Clusters after removing outliers **(g)** Two fitted rectangles (only smaller one is window) **(h)** Refined window in original point cloud (red rectangle)

Figure 5.12: The process of extracting a window opening

By considering whether there are window or door points on the boundary of the opening area and wall area, an opening can be classified as a door or window opening. If no semantic information is available, prior knowledge, such as that door openings are usually connected to the floor surface, is used to identify door openings from window openings.


(a) A wall with a door in original point cloud (b) The plane where it stops growing (c) Nonvoid points (green: wall; blue: door) (d) Centre points of void voxels on the plane

(e) Clusters of void voxels (three clusters are found) (f) Clusters after removing outliers (g) Three fitted rectangles (only the right one is a door) (h) Refined door in original point cloud (red rectangle)

Figure 5.13: The process of extracting a window opening

5.1.6 Merge Connected Cuboids

From the previous steps, cuboids for void volume inside rooms are extracted from the point cloud. Because there is at least one seed inside one room, one room could have multiple cuboids from different seed points. This usually happens in some complex rooms, like U-shape rooms, L-shape rooms, rooms with different ceiling heights, etc. Therefore, these cuboids should be merged into one. There are two circumstances in that two cuboids should be merged into one: a) one surface of a cuboid touches a surface of another cuboid: b) two cuboids have overlapped space.

5.1.7 Extract Space-bounding Elements

In this step, elements are reconstructed from the cuboids generated from the last step. Different strategies are used for different kinds of structural elements.

For ceilings, floors, and outer walls, it is common that only the inside surfaces are captured. For these elements that are only captured from one side, the surface where void volumes stop growing is considered the inner surface of these elements. The thickness of these elements is set to a default value because the information is not available in the point cloud.

In contrast, both sides of the inner walls usually are scanned when collecting data inside a building. That means two void cuboids are grown in the two adjacent rooms. The space between two void cuboids is considered the inner wall (as shown in Figure 5.14). The thickness of the inner wall is the distance between these two surfaces.

5.1.8 Extract Windows and Doors

In Section 5.1.5, the location and dimension of doors and windows can be extracted by fitting rectangles under the assumption that all doors and windows in the building are

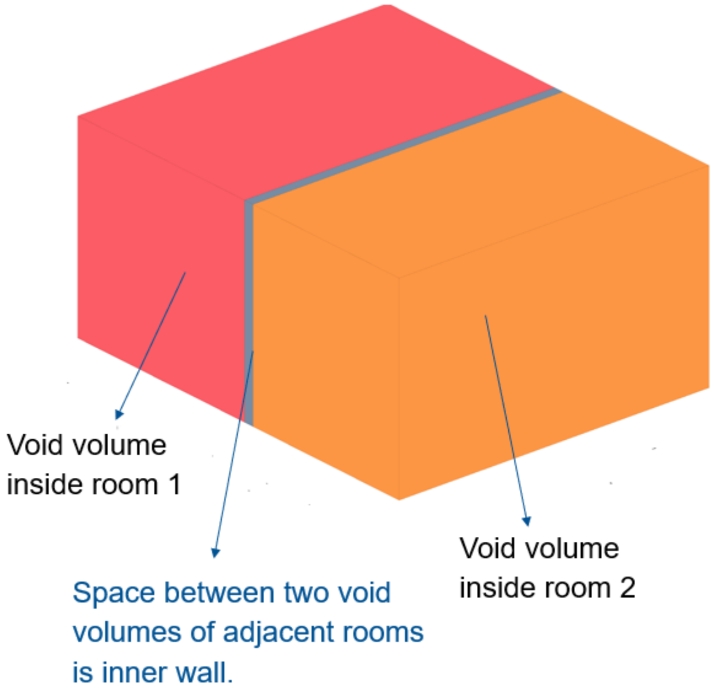

Void volume inside room 1
Space between two void volumes of adjacent rooms is inner wall.
Void volume inside room 2

Figure 5.14: One inner wall separates two adjacent volume spaces.

rectangles. Two rectangles extracted from the two sides of the wall should be identical in an ideal case. However, it is almost impossible to fit two exactly identical rectangles from the two sides. In order to get one rectangle that can represent the door or window openings, a new rectangle is computed from two old rectangles, whose width and height are the mean values of two old rectangles, and the new centre point is the middle point of the two old centre points.

At last, the author applies a label refinement step based on the detected rectangles of doors and windows. All points within the rectangles in the original point cloud are labelled as door or window points.

5.2 Experiments and Results

The proposed approach is written in C++. Point Cloud Library (PCL) 1.9.1 (Rusu and Cousins, 2011) and the Computational Geometry Algorithms Library (CGAL) 5.1 (The CGAL Project, 2020) are used in the implementation.

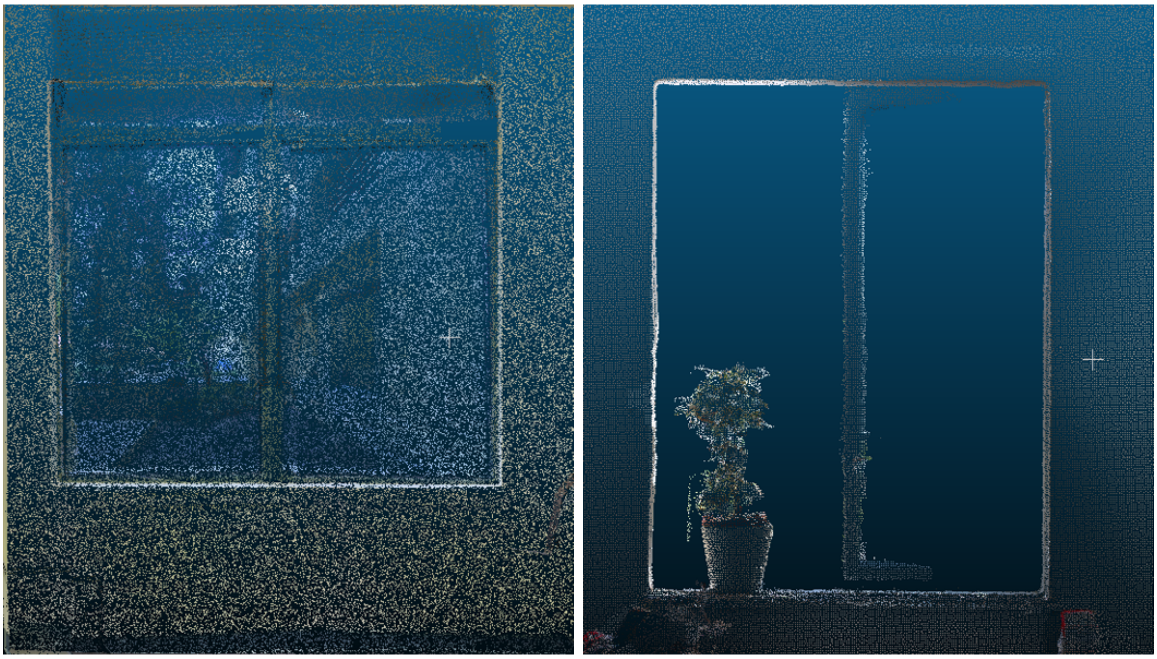
The point cloud that is used in the reconstruction process in this chapter is a laser scanning point cloud with 5mm resolution and captured in the office space at the Chair of Computational Modelling and Simulation (CMS) at the Technical University of Munich (TUM). This dataset is also the indoor environment of an office building, which is similar to that of the S3DIS dataset. In this chapter, the dataset captured at TUM is called the TUMCMS dataset in the following sections. The TUMCMS dataset is labelled manually and split into the training and validation set (the training set is around 70%

of the total area and the validation set is around 30%). Three training set settings are made: trained on the S3DIS dataset only, TUMCMS training set only, as well as S3DIS and TUMCMS training set. In addition, the author also shows the resulting improvement when applying door and window points refinement described in Section 5.1.8 for two different models. The results of different models are listed in Table 5.1.

model	training set	test set	ceiling	wall	floor	window	door
KPConv	TUMCMS(train)	TUMCMS (test)	94.7	81.8	96.6	54.4	48.8
KPConv	S3DIS	TUMCMS (test)	93.6	73.9	88.3	15.3	17.3
KPConv	S3DIS and TUMCMS (train)	TUMCMS (test)	93.5	82.9	97.3	48.7	47.1
KPConv result refinement	S3DIS and TUMCMS (train)	TUMCMS (test)	93.5	82.7	97.2	68.4	60.2
PointTransformer	S3DIS and TUMCMS (train)	TUMCMS (test)	94.4	83.1	97.8	52.2	50.1
PointTransformer result refinement	S3DIS and TUMCMS (train)	TUMCMS (test)	94.4	83.0	97.7	70.2	64.4

Table 5.1: Segmentation IoUs on TUMCMS test set

It is obvious to see that the model trained on both the S3DIS dataset and TUMCMS training set performs best in wall and floor classes. However, it performs worse in window class compared with training solely on the TUMCMS dataset. The main reason is that two datasets are captured by different technologies (S3DIS used RGBD camera and TUMCMS used laser scanner), which makes the property of point clouds non-consistent for different objects. One example is shown in Figure 5.15. While windows in the S3DIS dataset contain points and show outside scenes, windows in the TUMCMS dataset have no points and are actually "void" on the wall. Therefore, adding TUMCMS data to the training set can improve the result of the window class. In contrast, objects like walls, ceilings, and floors in the two datasets are quite similar. Training on both datasets can improve the performance of the network. By applying the window and door refinement, the annotated result of points can be improved, which is an essential part of the geometric digital twins that shows detailed geometric information.

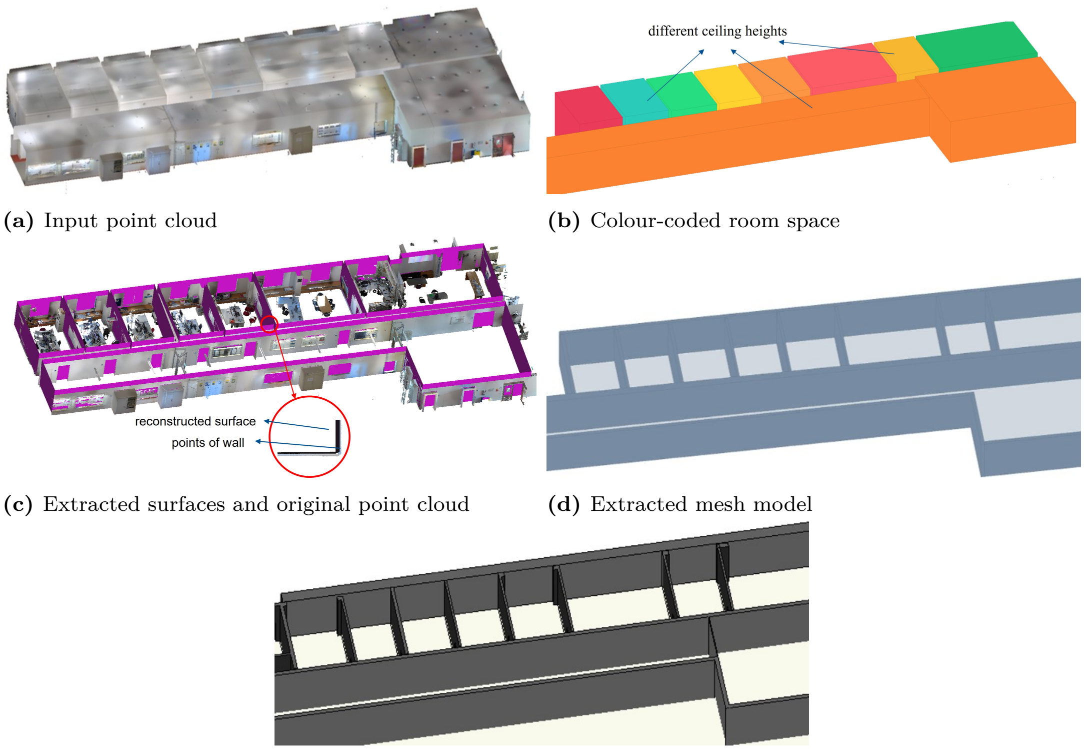


(a) Window in S3DIS dataset

(b) Window in TUMCMS dataset

Figure 5.15: Windows in S3DIS and TUMCMS dataset

The qualitative evaluation result on the part of the test set in the TUMCMS dataset is illustrated 5.16. The extracted room spaces are illustrated in Figure 5.16b. Each colour represents a void volume inside a room. It can be seen that not only standard cuboid rooms but also complex rooms can be detected. For example, the hallway can be seen as an L-Shape room. Furthermore, if focusing on the ceiling of the input cloud, the ceiling heights of some rooms and the hallway are not identical because of suspended ceilings which are quite common in the building industry nowadays. The different ceiling heights in the input point cloud can be clearly identified in the grown void volumes. In Figure 5.16c, it can be seen that the alignment of reconstructed surfaces and the original point cloud. The points representing the ceiling are removed to depict rooms of the input point cloud more clearly. It is obvious that the shift is small, and the quantitative evaluation is discussed later in this section. Based on the volume spaces found in previous steps,



(a) Input point cloud

(b) Colour-coded room space

(c) Extracted surfaces and original point cloud

(d) Extracted mesh model

(e) Manually created model

Figure 5.16: Qualitative evaluation on TUMCMS test set

walls, floors, and ceilings can be extracted. In the experiment, if only one surface of the elements is scanned, the default thickness of the element is set to 30cm. The reconstructed 3D model created by the proposed approach and the BIM model created manually are shown in Figure 5.16d and 5.16e.

In Figure 5.17, the points representing the ceiling are removed to depict rooms of the input point cloud more clearly. It is evident to see that a gap separates two adjacent room volumes. Moreover, the gaps between two adjacent rooms are supposed to be the wall that separates these rooms.

As shown in Figure 5.16 and Figure 5.17, the void growing algorithm performs quite well in our dataset, a typical indoor environment of offices with strong occlusion. The author states that our algorithm could apply to other point clouds of buildings without any modifications or with slight modifications. In most cases, if the buildings have similar door and window openings, we can apply our approach directly to new datasets. However, if the door and window openings have different shapes (like circles), the model has never been trained on this kind of door and window. It is understandable that the network cannot detect one type of element that it has never seen before. To find and fit these doors and windows, we could enlarge the training set or just add other predefined opening shapes as available prior knowledge.

By quantitative evaluation, the room areas in the TUMCMS test set are evaluated by first comparing area values, absolute and relative deviation, overlapped area, and

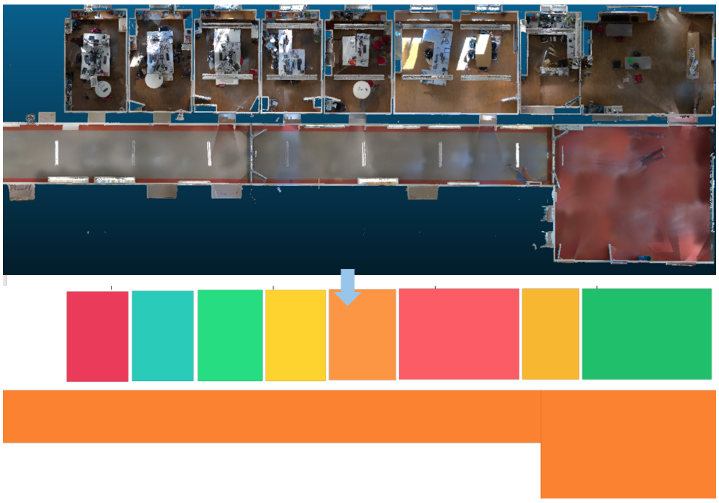


Figure 5.17: Input point cloud and void volumes inside rooms after removing ceilings

Intersection over Union (IoU) of individual rooms in Table 5.2. Basically, if using R_i to denote the room space of the extracted method for room i and S_i to denote the room space of room i in the manually created BIM model, the IoU for the room i is computed as

$$Room^{IoU}i = \frac{R_i \cap S_i}{R_i \cup S_i}. \tag{5.5}$$

The performance of choosing different voxel sizes is also compared in the table, and it shows a smaller voxel size performs better than the larger one. In addition, the result on another dataset is also evaluated, which is captured in a different office building at the Technical University of Munich (TUM), as shown in Table 5.3, marked as Dataset 2. The input point cloud and corresponding created model of the approach for Dataset 2 are illustrated in Figure 5.18 for qualitative evaluation as well.

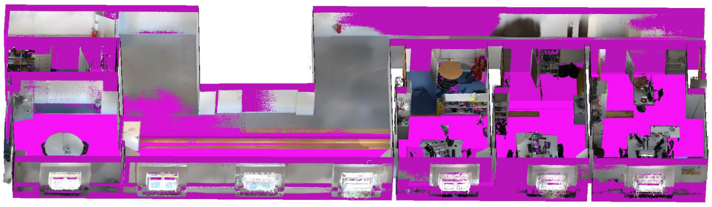


Figure 5.18: The input point cloud and corresponding created model of the approach for Dataset 2

Table 5.2: Area comparison between the void-growing model and BIM model on TUMCMS test set

office No.	voxel size (cm)	void-growing (m^2)	BIM (m^2)	abs. dev. (m^2)	rel. dev. (%)	overlap area (m^2)	IoU (%)
office 1	5cm	46.62	49.22	2.60	5.28	45.55	94.24
office 2	5cm	26.15	26.82	0.67	2.50	25.85	93.15
office 3	5cm	23.30	24.11	0.80	3.32	23.13	92.15
office 4	5cm	25.20	25.88	0.68	2.63	24.92	95.03
office 5	5cm	23.75	25.20	1.45	5.75	23.23	94.58
office 6	5cm	23.75	24.90	1.15	4.62	23.12	93.18
office 1	3cm	47.69	49.22	1.53	3.11	46.90	94.74
office 2	3cm	26.35	26.82	0.47	1.75	26.22	74.02
office 3	3cm	23.64	24.11	0.47	1.95	23.48	92.78
office 4	3cm	25.30	25.88	0.58	2.24	25.20	94.64
office 5	3cm	23.82	25.20	1.38	5.48	23.41	95.23
office 6	3cm	24.10	24.90	0.80	3.21	23.85	93.54

Table 5.3: Area comparison between the void-growing model and ground truth on Dataset 2 (5cm voxel size)

office No.	void-growing (m^2)	BIM (m^2)	abs. dev. (m^2)	rel. dev. (%)	overlap area (m^2)	IoU (%)
office 1	17.75	18.79	1.04	5.53	17.02	93.29
office 2	26.46	28.56	2.10	7.35	25.88	92.55
office 3	21.65	22.87	1.22	5.33	20.03	91.06
office 4	22.58	23.36	0.78	3.34	21.56	93.74

In addition, the thicknesses of some selected walls from the approach and manually created model from two datasets are shown in Table 5.4. The proposed algorithm could apply to other point clouds of buildings without any modifications or with slight modifications. In most cases, if the buildings have similar door and window openings, it can be applied directly to new datasets. However, if the door and window openings have different shapes (like circles), the model has never been trained on this kind of door and window. It is understandable that the network cannot detect elements that it has never seen before. To find and fit these doors and windows, the training set can be enlarged, or other predefined opening shapes can be used as available prior knowledge.

Table 5.4: Wall thickness comparison between the void-growing model and BIM model of two datasets with different voxel sizes

dataset No.	wall No.	voxel size	void-growing (m)	BIM (m)	abs. dev. (m)	rel. dev. (%)
1	wall 1	5cm	0.20	0.17	0.03	17.6
1	wall 2	5cm	0.20	0.16	0.04	25.0
1	wall 3	5cm	0.15	0.14	0.01	7.1
1	wall 4	5cm	0.20	0.17	0.03	17.6
1	wall 5	5cm	0.20	0.17	0.03	17.6
1	wall 1	3cm	0.18	0.17	0.01	5.9
1	wall 2	3cm	0.18	0.16	0.02	12.5
1	wall 3	3cm	0.15	0.14	0.01	7.1
1	wall 4	3cm	0.18	0.17	0.01	5.9
1	wall 5	3cm	0.18	0.17	0.01	5.9
2	wall 1	5cm	0.20	0.21	0.01	4.8
2	wall 2	5cm	0.20	0.22	0.02	9.1
2	wall 3	5cm	0.20	0.18	0.02	11.1
2	wall 4	5cm	0.15	0.20	0.05	25.0
2	wall 1	3cm	0.18	0.21	0.03	14.3
2	wall 2	3cm	0.21	0.22	0.01	4.5
2	wall 3	3cm	0.18	0.18	0	0
2	wall 4	3cm	0.18	0.20	0.01	5.0

5.3 Conclusions and Discussions

In conclusion, the chapter presents an automatic method that extracts void room space inside rooms first. In addition, state-of-the-art deep learning technologies are added to the initial void-growing method to improve the performance of extracting walls, ceilings, and floors. Semantic information from deep learning is also used to extract doors and windows. However, the window and door recognition by point cloud deep learning is not improved when adding more training data from different sources. The final digital twin includes point clusters with annotated semantic information, which shows the detailed information of captured surfaces and a simplified mesh model of the space-bounding elements. As the proposed method can automatically extract ceilings, walls with doors and windows, and floors in both simple cuboid rooms and rooms with complex structures, it can reduce a considerable amount of human effort to create a geometric digital twin of buildings. Human modellers only need to check the correctness and accuracy of the automatically created model and then add other objects into the model depending on the requirements.

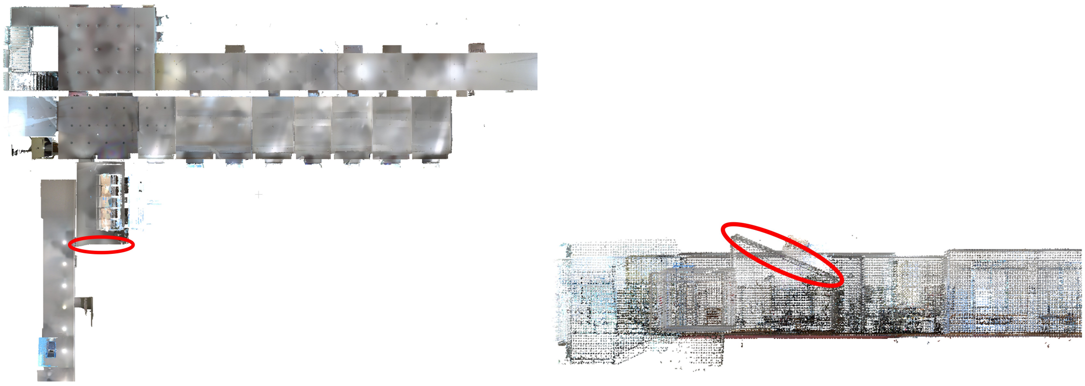
Most previous research starts with detecting elements by considering geometric information only and subsequently extracting room information. However, geometric information of target objects could be missing because of the occlusion, which makes these methods perform worse in an occluded environment. The proposed method detects spaces inside rooms first and then extracts building elements by considering geometric information and semantic information predicted from deep learning, which makes it possible to distinguish wall surfaces from furniture surfaces in an occluded environment, even when the furniture surfaces are large.

One limitation of the proposed method is that it needs to fulfil the Manhattan-world assumption because the growing process is implemented in a "cuboid" way (as shown in Section 5.1.3). The approach designed for non-Manhattan-world buildings is presented in Chapter 6. In addition, the voxel size used in the downsampling process limits the performance of the approach. When detecting objects with large dimensions, the impact is insignificant. However, it becomes vital to determine the thickness of elements because the thickness is usually not very large. Another limitation is that the proposed approach is suitable for laser scanning point clouds but not for point clouds from an RGBD camera because window openings in the point cloud are used to detect windows and find the opening dimension and location of the windows. The window openings are actually void in the laser scanning point cloud but not void in the point cloud captured from the RGBD camera (as shown in 5.15).

Chapter 6

Approach for non-Manhattan-world environments

In the indoor environments of buildings, the elements that usually make an environment a non-Manhattan-world are slanted ceilings or walls not-aligned to the x- or y-axis. Some examples are shown in Figure 6.1a and 6.1b. In this chapter, the author proposed an approach to reconstruct the indoor environments that do not fulfil the Manhattan-world assumption. This chapter is organised as follows: The proposed approach is shown in detail in Section 6.1. The experiments and results are presented in Section 6.2. Conclusions and discussions are shown in Section 6.3.



(a) Wall not aligned to x- or y-axis **(b)** Slanted ceiling

Figure 6.1: Elements that make environment non-Manhattan-world

6.1 Proposed Approach

In a non-Manhattan-world environment, the void spaces inside rooms are not cuboid anymore, which makes the void-growing introduced in Chapter 5 not work. Those steps that do not require Manhattan-world assumptions like 3D deep learning in Section 5.1.1 are still applied in the proposed approach. The overall pipeline of the proposed approach is illustrated in Figure 6.2.

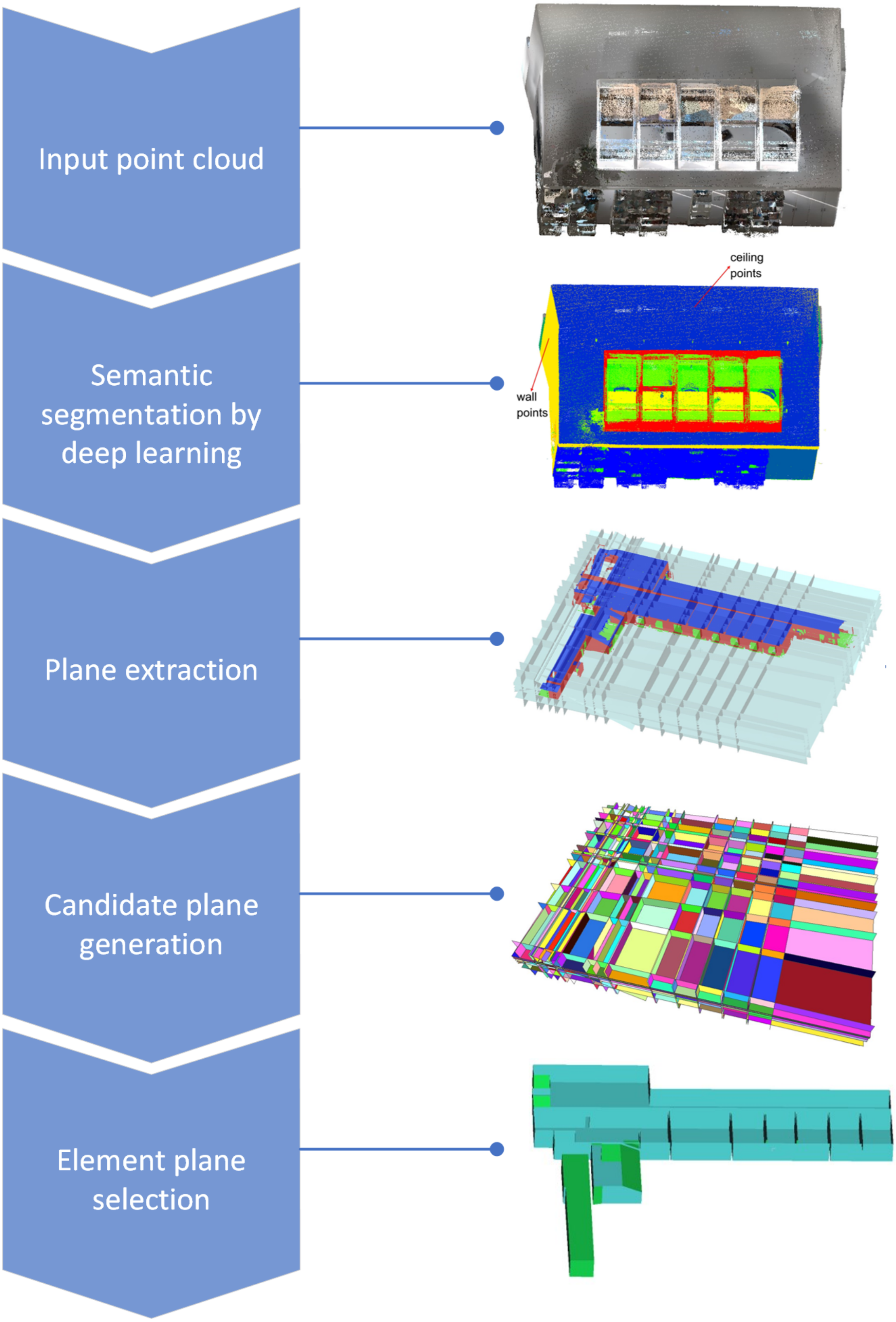
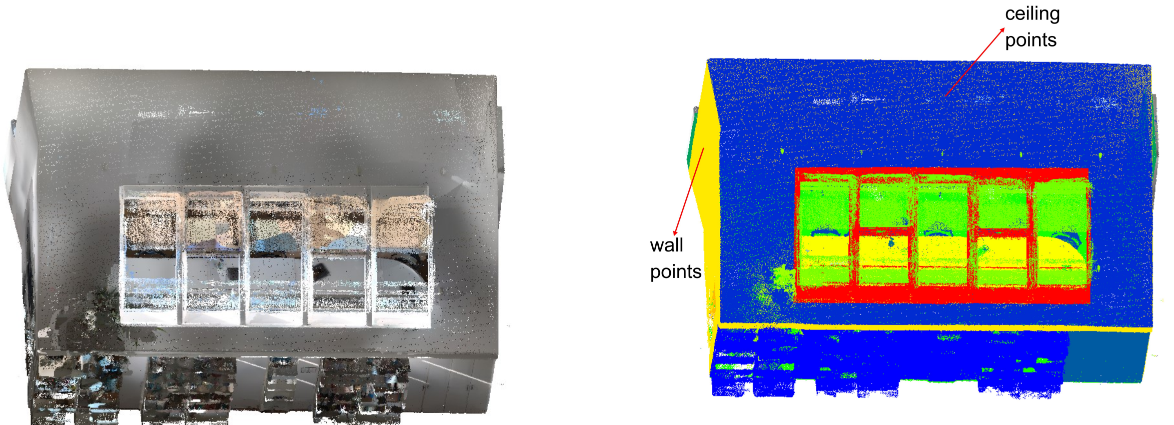


Figure 6.2: Proposed pipeline for non-Manhattan world environment

6.1.1 Point Cloud Segmentation by Deep Learning

As shown in Table 5.1, the prediction results by deep learning are relatively good for large structural elements like ceilings, walls, and floors (IoU for ceiling points, wall points, and floor points are around 90.4%, 83.0%, and 97.7% respectively), despite some rooms do not follow the Manhattan-world assumption. This kind of reliable semantic information is valuable and will be used in further processing steps.

In the training process of neural networks, there is no restriction of the Manhattan-world assumption, and point clouds of rooms with slanted ceilings are also included in the training set; the neural network can detect ceiling points not only for normal ceilings but also for slanted ceilings. An example of point cloud segmentation result of a non-Manhattan-world environment is shown in Figure 6.3. The original coloured point cloud is shown in Figure 6.3a, and the corresponding segmentation result is shown in Figure 6.3b. It can be seen that although the indoor environment is not under the Manhattan-world assumption, the slanted ceiling points and non-aligned wall points can be detected well. This qualitative evaluation is consistent with the quantitative evaluation, which presents the IoU values are not worse when applying the neural network model to a non-Manhattan-world environment.



(a) Example of non-Manhattan environment

(b) Segmentation result of non-Manhattan environment

Figure 6.3: Example of non-Manhattan environment and its segmentation result

6.1.2 Plane Extraction

In this section, large planes in the point cloud are extracted. In an indoor environment of a building, large planes are usually surfaces of structural building elements like walls, ceilings, and floors, as well as surfaces of large furniture like bookshelves and cupboards. Ideally, in this step, the goal is to extract all planes of structural elements and meanwhile discard furniture planes. But this task is not easy, and the reasons behind that are listed as follows: a) there are usually noisy points and outliers in the captured point clouds, which have a negative influence on the plane extraction; b) furniture surfaces could also be large so that it is not reliable to distinguish the surfaces based solely on the plane size; c) in the indoor environment of a building, there is occlusion which leads to some incomplete planes. This is also a negative factor in plane extraction.

Therefore, the semantic information extracted from point cloud segmentation by deep learning (in Section 6.1.1) is used to target these difficulties and benefit the plane extraction process. Neural networks can be used to target difficulties and improve the process. If noisy points, outliers, and furniture points are labelled as irrelevant classes in the training set when training a neural network model, these points can be distinguished from the points of structural elements.

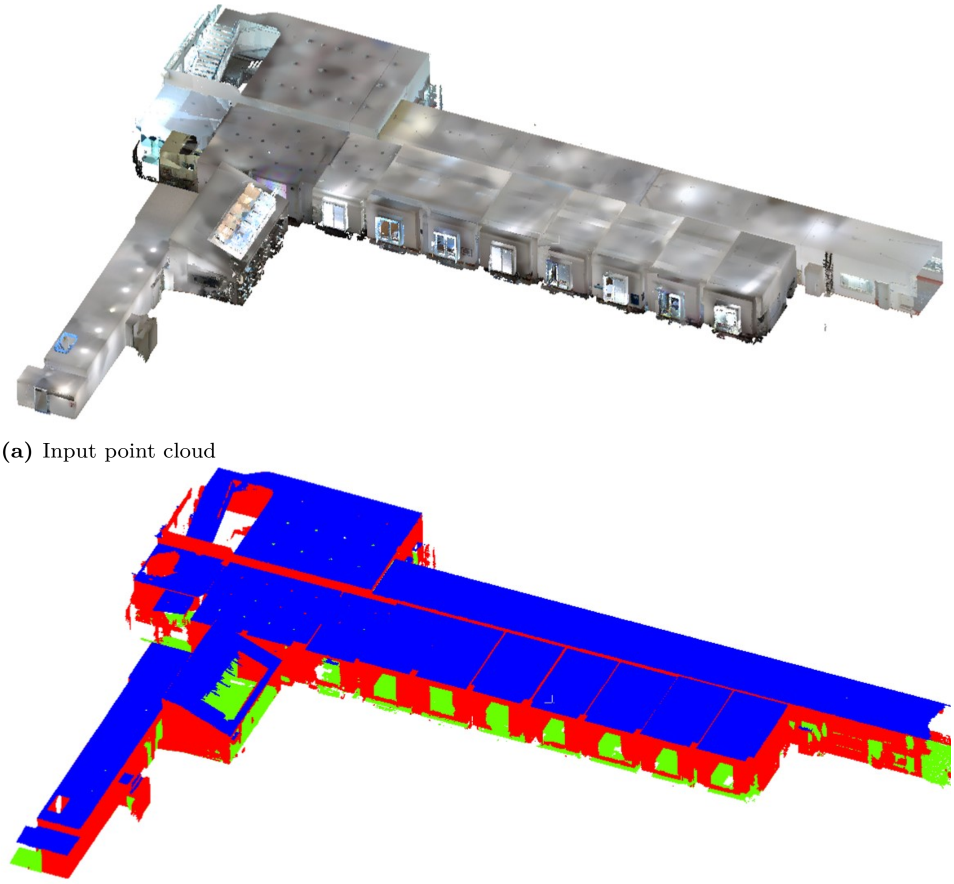
A variant of the RANSAC methods, the efficient RANSAC method (Schnabel et al., 2007), is applied in this step. Basically, the RANSAC algorithm extracts different shapes by randomly extracting point sets from the input point cloud and fitting corresponding shape primitives. The typical RANSAC algorithm consists of the following steps: a) select a point set from the input point cloud randomly; b) construct a geometric primitive to the selected point set; c) count the number of inliers to the fitted shape. Inliers are those points that are located within the error tolerance to the constructed shape in Step c. These three steps are repeated a predefined number of times, and the fitted primitive with the largest number of inliers is considered the extracted shape. The primitive with the maximal inlier number is continuously extracted until the termination condition is met. For example, the inlier number of the primitive with the largest number of inliers is smaller than a user-defined value, or the number of used points is smaller than a predefined threshold. However, it is not practical to apply basic RANSAC in large-scale point clouds because testing all possible primitive candidates against the whole input data in order to find the largest shape in a large amount of data is very slow. The efficient RANSAC solves this problem by testing the primitive against subsets of the input point cloud. The parameter study in the process of plane extraction by efficient RANSAC is discussed in Section 6.2.1. The shape candidates are extracted until the probability of missing the largest candidate is less than a pre-defined threshold. It should be noticed that RANSAC extracts a plane and provides the plane equation

$$ax + by + cz + d = 0, \tag{6.1}$$

where a, b, c, and d are the coefficients of the plane, which is computed by RANSAC. In theory, these extracted planes by RANSAC are infinitely large. A bounding box which encloses all points of the input point clouds is used as the boundary to crop these planes.

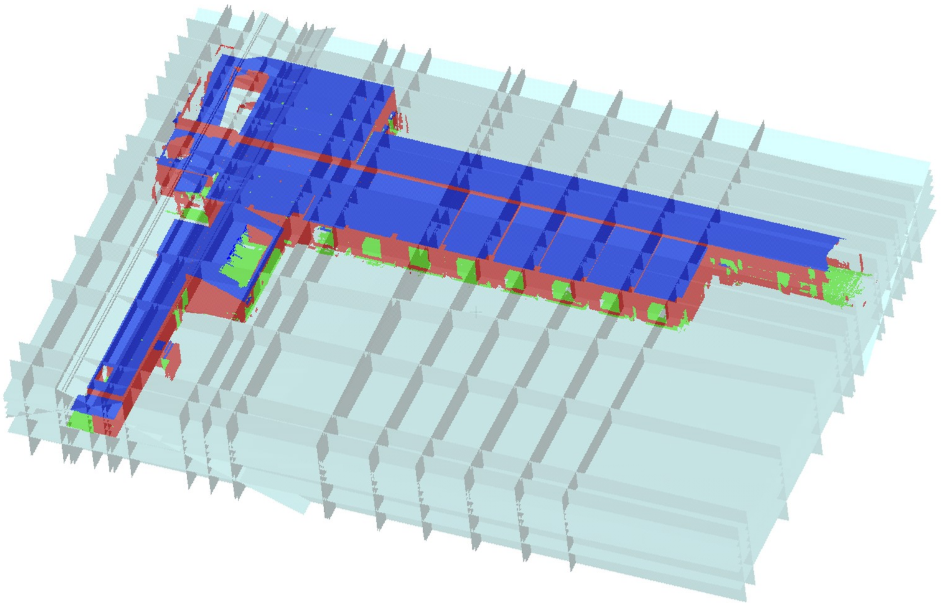
The planes extracted by efficient RANSAC and the corresponding point clouds are presented in Figure 6.4. The original point cloud is shown in Figure 6.4a. The point cloud of structural elements, which are filtered by the semantic information extracted in Section 6.1.1, is illustrated in Figure 6.4b, where blue points are ceiling points, green points are floor points, and red points are wall points. The extracted planes by efficient RANSAC when inputting the point cloud of structural elements are presented in Figure 6.4c. It can be seen that surfaces of structural elements are extracted as planes, and the plane locations fit the point cloud well.

In comparison with the efficient RANSAC result of using extracted points of structural elements, extracted planes using the original point cloud (shown in Figure 6.4a) as input to efficient RANSAC are shown in Figure 6.5. It can be observed that the plane extraction result of using points of structural elements is much cleaner than that of using all points, which makes sense because irrelevant planes can be extracted as long as enough supporting points are found by efficient RANSAC. The supporting points of these irrelevant planes can be points of furniture, noisy points, and even points of structural elements. By removing points of irrelevant classes, the extracted candidate planes of structural elements are cleaner. The smaller number of plane candidates benefit from further processing steps at the same time.



(a) Input point cloud

(b) Extracted points of structural elements (blue points: ceiling, red points: wall, green points: floor)



(c) Extracted planes of the point cloud (planes are set transparent for better visualisation)

Figure 6.4: Extracted planes by efficient RANSAC using points of structural elements

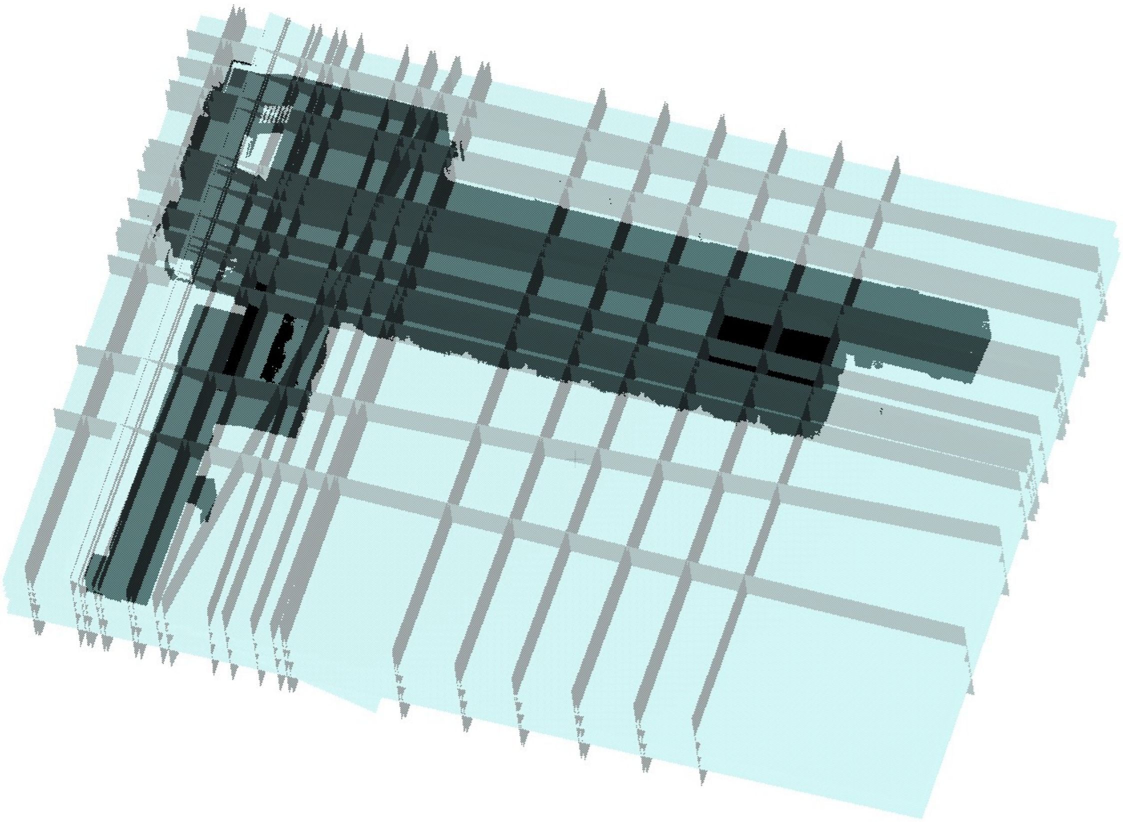


Figure 6.5: Extracted planes by efficient RANSAC in original point cloud (input shown in Figure 6.4a). Compared with the extraction result of using points of structural elements in Figure 6.4c, the result of using the original point cloud contains more non-relevant planes.

6.1.3 Candidate Plane Selection

In this step, the planes computed from the previous section are intersected to get individual polygonal faces, which are considered the potential faces of structural elements. As shown in Figure 6.6, large planes are over-segmented into different faces. These face candidates are colour-coded, which means different colours represent different faces. In order to select real faces of structural elements from all face candidates and inspired by previous work (Ochmann et al. (2016), Ochmann et al. (2019), Nan and Wonka (2017)), a binary linear programming problem is set up to get the final surface model.

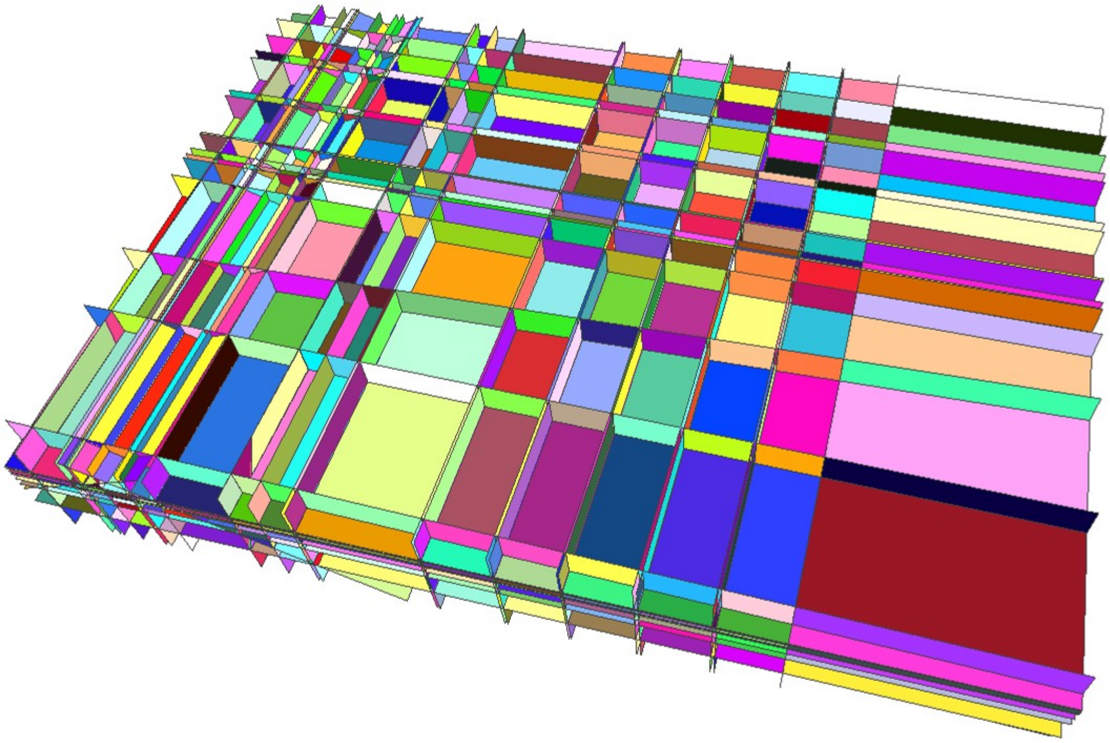


Figure 6.6: Candidate faces by intersecting planes (different faces are colour-coded, and input point cloud and planes are shown in Figure 6.4a)

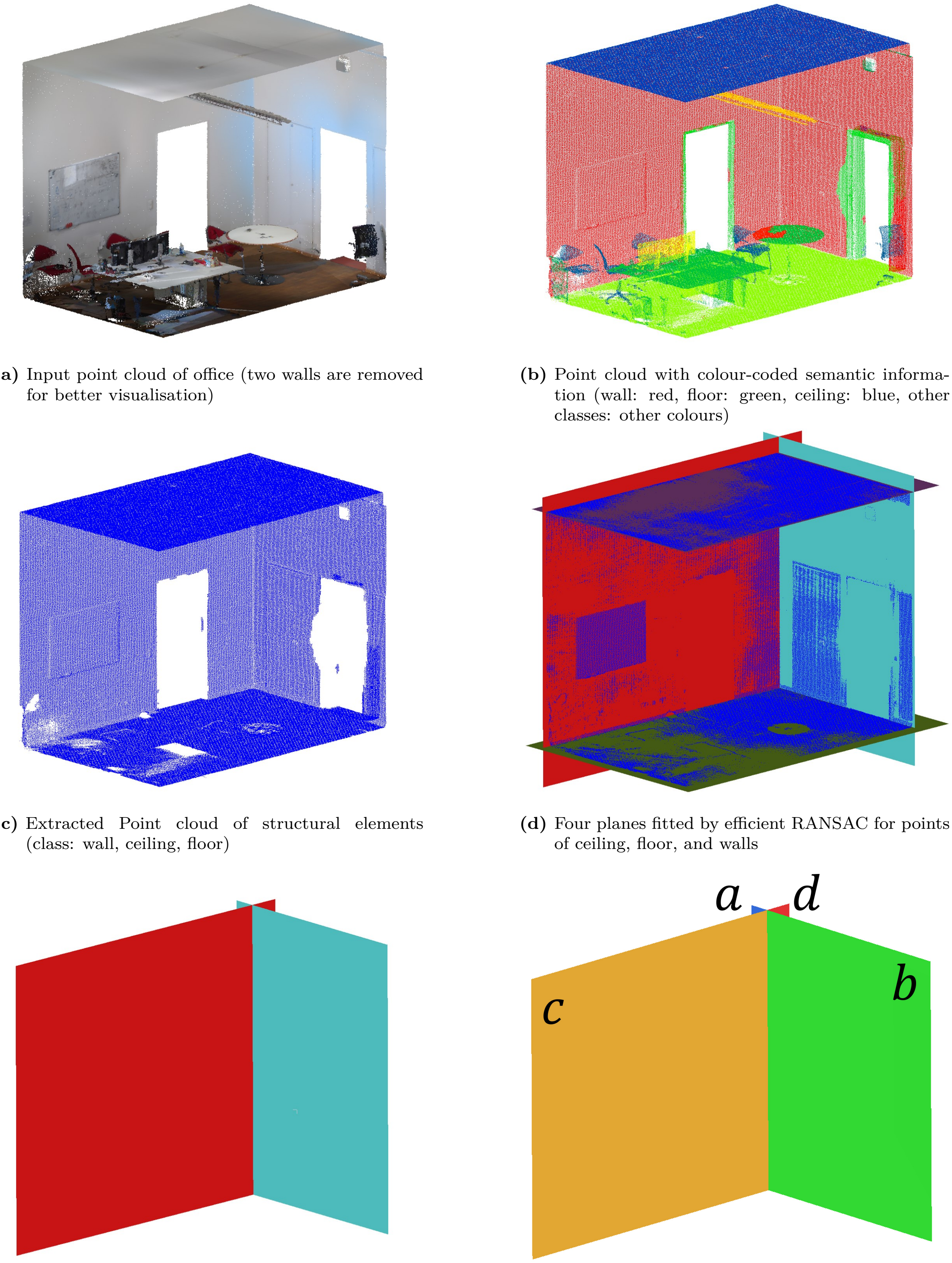
As the candidate faces are gotten by intersecting planes, one large plane is over-segmented into many small faces. Some of these small over-segmented planes are just redundant faces which do not represent any faces of elements in the real 3D world. The process of extracting intersected planes in the point cloud of an office is shown in Figure 6.7 for example. The original input point cloud, the colour-coded point cloud with semantic information, and the extracted point cloud of structural elements are presented in Figure 6.7a, Figure 6.7b, and Figure 6.7c, respectively. The four planes fitted by efficient RANSAC for the points of the ceiling, floor, and walls are shown in Figure 6.7d. Two planes out of the four are selected and shown in Figure 6.7e. Intersecting two planes results in four polygonal faces, as shown in Figure 6.7f.

The aim of this step is to select an optimal subset of all polygonal faces which can represent the surfaces of structural elements. If no boundary conditions apply, any number of these four faces resulting from intersecting two planes can be selected as real faces of elements. However, it needs to be considered whether these different selection options make sense when they are used to represent the surfaces of elements in the real world. In Figure 6.8, different circumstances of selecting four faces which are resulted from intersecting two planes are illustrated. The colour of selected faces is set solid, while the colour of unselected faces is set transparent. These circumstances are discussed here to explore whether it could happen in a real environment represented by point clouds. In Figure 6.8a, none of the four faces is selected. This circumstance could happen because all planes are extended and over-segmented, which makes it quite normal that the faces of two intersected planes do not represent any surfaces in the real world. Therefore, it could happen in the real environment to select 0 face to represent the surface of elements.

In Figure 6.8b, one of the four faces is selected. This circumstance can not happen because all elements captured in the point cloud are connected to others, which means all surfaces of elements are bounded by the edges which are resulted from intersecting with other elements. One single selected face could not represent any meaningful surfaces of elements in the environment. Therefore, it does not make sense to select 1 face to represent the surface of elements in the real environment.

In Figure 6.8c, two of the four faces are selected. This circumstance happens a lot in the environment. For example, when two selected faces are not in the same plane, the edge could be the intersection of two surfaces of intersected walls, while the selected faces represent the corresponding two surfaces. Similarly, this circumstance also applies to other intersections of surfaces, like ceiling and wall intersections, floor and wall intersections, etc. In addition, if the two faces are in the same plane, this plane could be the surface of elements, and the other two non-selected faces are just the extended parts of other surfaces, which do not represent any surfaces of elements. Therefore, selecting 2 faces to represent the surfaces of elements can represent the element surfaces in the point cloud.

In Figure 6.8d and Figure 6.8e, three and four of the four faces are selected. These circumstances cannot happen in the point cloud. Firstly, in the point cloud captured by the laser scanner, no points can be scanned behind the visible surfaces. If the number of the selected faces is more than 2, one or two faces which should not be visible in the real environment are added to the face subset to represent the surfaces of elements. Secondly, all elements in the real world cannot be represented as just one surface, as there is always thickness for objects. Even though some elements are captured just from one side, the captured surfaces of elements in the real world are always bounded by other surfaces. Three or four selected faces (shown in Figure 6.8e) of one edge cannot represent any surfaces of elements that make sense in the environment. Therefore, it cannot represent the surfaces of elements to select 3 and 4 faces. In summary, the number of selected faces which are resulted from intersecting two planes can only be 0 and 2.



(a) Input point cloud of office (two walls are removed for better visualisation)

(b) Point cloud with colour-coded semantic information (wall: red, floor: green, ceiling: blue, other classes: other colours)

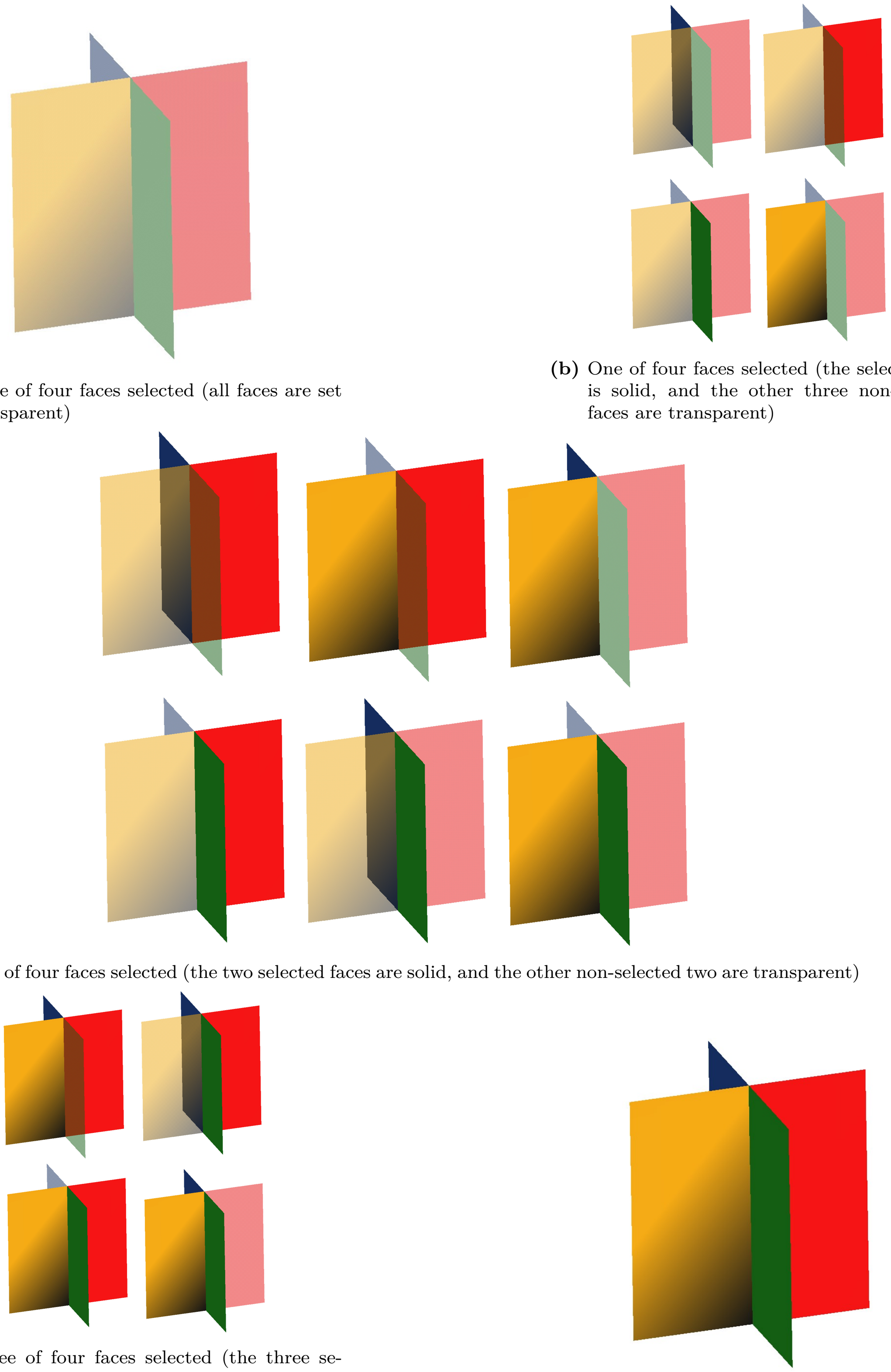
(c) Extracted Point cloud of structural elements (class: wall, ceiling, floor)

(d) Four planes fitted by efficient RANSAC for points of ceiling, floor, and walls

(e) Two intersected wall planes

(f) Four faces (a, b, c, and d) resulting from intersecting two planes

Figure 6.7: Example of extracting two intersected planes of walls in an office



(a) None of four faces selected (all faces are set transparent)

(b) One of four faces selected (the selected face is solid, and the other three non-selected faces are transparent)

(c) Two of four faces selected (the two selected faces are solid, and the other non-selected two are transparent)

(d) Three of four faces selected (the three selected faces are solid, and the non-selected one is transparent)

(e) All four faces selected (all faces are set solid)

Figure 6.8: Different circumstances of selecting four faces resulted by intersecting two planes (selected faces: solid, unselected faces: transparent)

6.1.4 Energy Terms

Inspired by previous work (Nan and Wonka, 2017), the objective function consists of two energy terms: data-fitting and model complexity.

Data-fitting term aims to describe how well the faces fit points in the point cloud. It is defined to represent a confidence-weighted and label-weighted percentage of points that do not support the final face reconstruction. The data-fitting term is shown by the equation

$$E_{fitting} = 1 - \frac{1}{P}\sum_{i=1}^{N} x_i \cdot F(f_i) + \frac{1}{area(M)}\sum_{i=1}^{N} x_i \cdot (area(f_i) - area(M_i^{\alpha})), \tag{6.2}$$

where $E_{fitting}$ is the energy term of data fitting, P is the number of points in the point cloud, N is the number of face candidates, x_i is the variable to represent whether the candidate face f_i is selected ($x_i = 1$) or not selected ($x_i = 0$), $F(f_i)$ is the confidence function of selecting face f_i. $area(M_{all})$, $area(f_i)$, and $area(M_i^{\alpha})$ represent the overall surface area of the final model, the area of the candidate face f_i, and the area of α-shape mesh M_i^{α} of f_i (α-shape is defined in (Kai and Da, 2021)). The confidence function $F(f_i)$ is defined at each point and considers its local neighbourhood and label predicted by deep learning as

$$F(f_i) = \sum_{distance(p,f)<\sigma} (1 - \frac{distance(p,f)}{\sigma}) \cdot G(p), \tag{6.3}$$

where p is the point, f is the face, $dist(p, f)$ is the distance between the point p and the face f, σ is the threshold which determines whether the point is considered (only points close to the plane are considered), and $G(p)$ is the confidence function of the point cloud at the point p, which is computed by examining the local covariance matrices defined at the point p and its label predicted by deep learning. The function is defined as

$$G(p) = coef(p) \cdot \frac{1}{3}(\sum_{i=1}^{3}(1 - \frac{3\lambda_i^1}{\lambda_i^1 + \lambda_i^2 + \lambda_i^3}) \cdot \frac{\lambda_i^2}{\lambda_i^3}), \tag{6.4}$$

where $coef(p)$ is the label coefficient depending on the semantic label. In the experiment, if the point p is predicted as structural elements $coef(p)$ is set to 1, and $coef(p)$ is set to 0.1 if the point is predicted as other classes. More discussions with regard to choosing the coefficients are shown in Section 6.2.1. The second term $\frac{1}{3}(\sum_{i=1} 3(1 - \frac{3\lambda^1}{\lambda^1+\lambda^2+\lambda^3}) \cdot \frac{\lambda^2}{\lambda^3}$ is from the previous work (Pauly et al., 2005), where $\lambda^1 \leq \lambda^2 \leq \lambda^3$ are the three eigenvalues of the covariance matrix. while the first term $1 - \frac{3\lambda^1}{\lambda^1+\lambda^2+\lambda^3}$ shows the quality of fitting a local tangent plane at the point p, the second term $\frac{\lambda^2}{\lambda^3}$ evaluates the local sampling uniformity.

In summary, the data-fitting term favours faces that are close to the dense point regions that are predicted as structural elements. **Model complexity** term aims to favour simple structures like large planes, which is proposed by (Nan and Wonka, 2017). It is defined as follows:

$$E_{complexity} = \frac{1}{E}\sum_{i=1}^{E} edge(e_i), \tag{6.5}$$

where E denotes the total number of plane intersections (number of edges), $edge(e_i)$ is the variable that shows whether the edge of planes exists. $edge(e_i)$ will be set to 1 if the faces connected to e_i is not in the same plane, while $edge(e_i)$ will be set to 0 if the faces connected to e_i are coplanar.

6.1.5 Optimisation

The optimal subset of faces is obtained by minimising the weighted sum of the energy terms under certain boundary conditions. The final term that needs to be optimised is

$$\lambda_{fitting} \cdot E_{fitting} + \lambda_{complexity} \cdot E_{complexity}, \tag{6.6}$$

where $\lambda_{fitting}$ and $\lambda_{complexity}$ are two weight coefficients for the two corresponding energy terms.

The restrictions that need to be followed are as follows:

$$\sum_{i \in f(e_i)} x_j = 0 \ or \ 2, 1 \leq i \leq E, \tag{6.7}$$

and

$$x_i \in \{0, 1\}, 1 \leq i \leq N, \tag{6.8}$$

where $f(e_i)x_j$ is the number of faces that are connected to the edge e_i. As discussed in Section 6.1.3, this value can only be 0 or 2 to make the selected faces able to represent surfaces of elements in point clouds. x_i denotes whether the face i is selected. While x_i is the value of 1 if it is selected, it is 0 if not selected.

The problem defined here is a binary linear optimisation problem. The subset of selected faces with the value $x_i = 1$ is the optimised face model of the structural elements from the input point cloud.

6.2 Experiments and Results

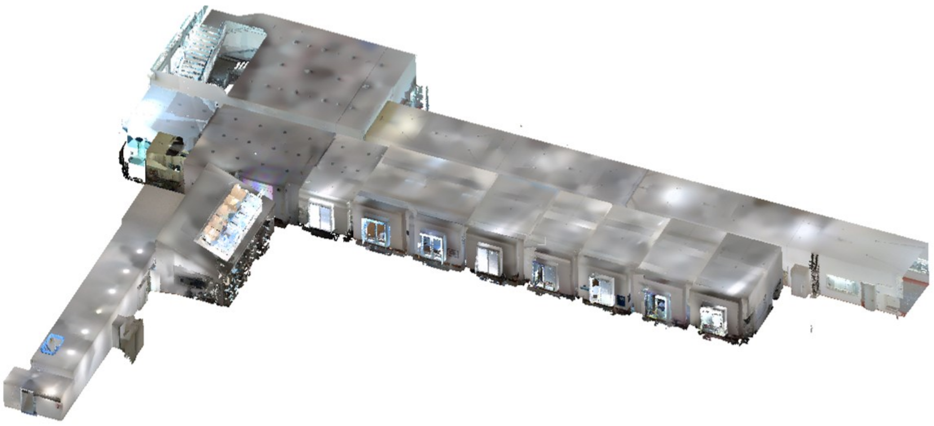
The implementation of extracting plane candidates in point clouds and selecting planes of structural elements is written in C++ and uses the Computational Geometry Algorithms Library (CGAL) (The CGAL Project, 2020) and Point Cloud Library (Rusu and

Cousins, 2011). And some parts of the implementation code are based on the code by Nan and Wonka (2017) in the CGAL library.

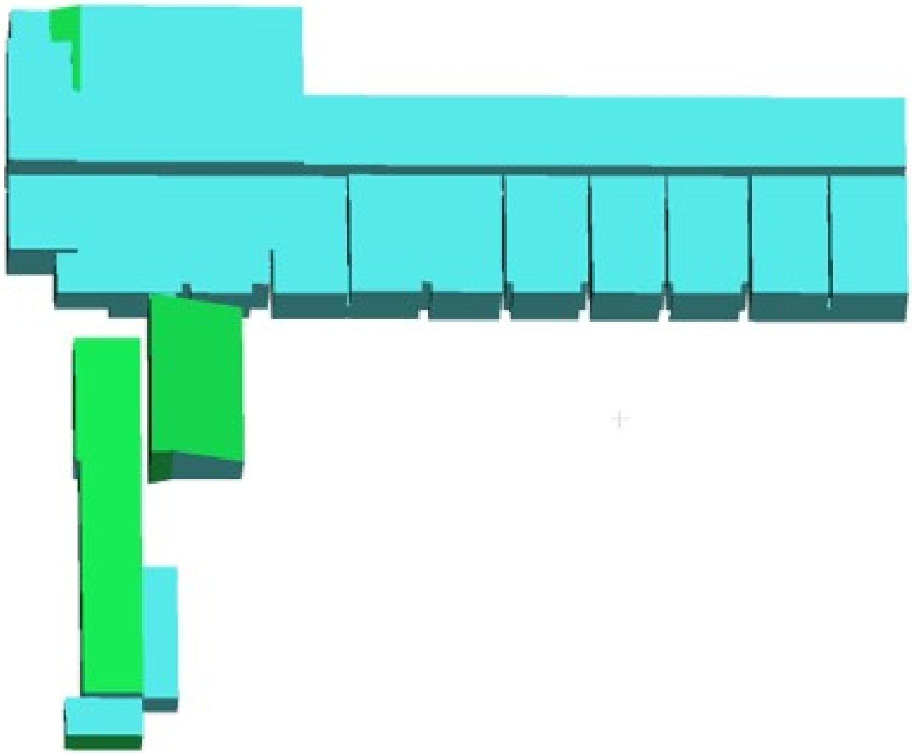
The result of the optimised plane selection of structural elements and the corresponding input point cloud is shown in Figure 6.9. It can be seen that all surfaces are reconstructed (including a slanted ceiling and a shifted wall inside one room).

In order to get better visualisation, part of the ceiling is removed in Figure 6.10a while all ceiling is removed in Figure 6.10b. It is obvious to see that all inner walls are also reconstructed well.

Another result of the non-Manhattan-world environment is shown in Figure 6.11. While the input point cloud is shown in Figure 6.11a, the point cloud with semantic information is shown in Figure 6.11b. The final created mesh model is shown in Figure 6.11c, where the slanted ceilings and their connections to walls are well represented.

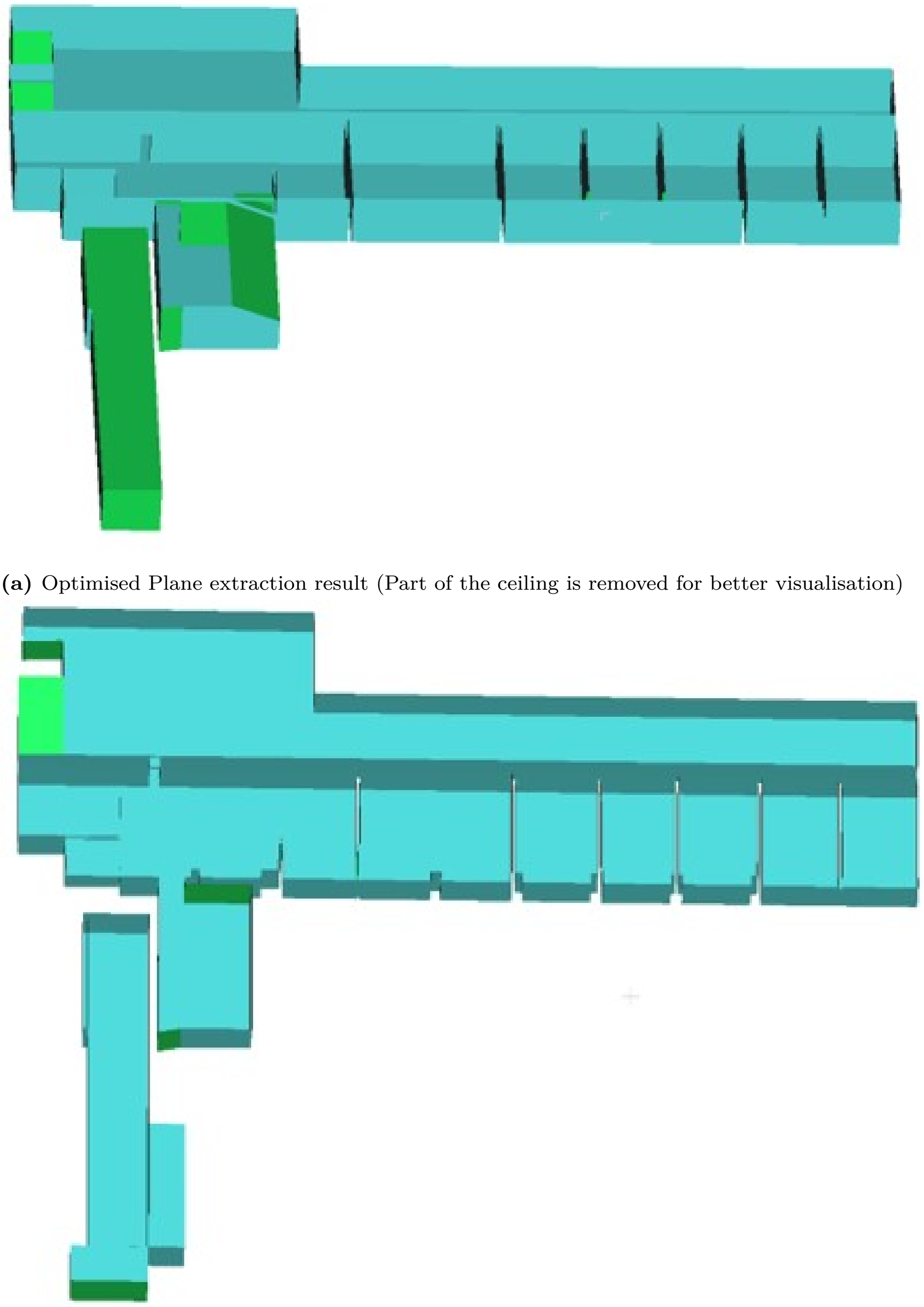


(a) Input point cloud



(b) Optimised selected planes of structural elements

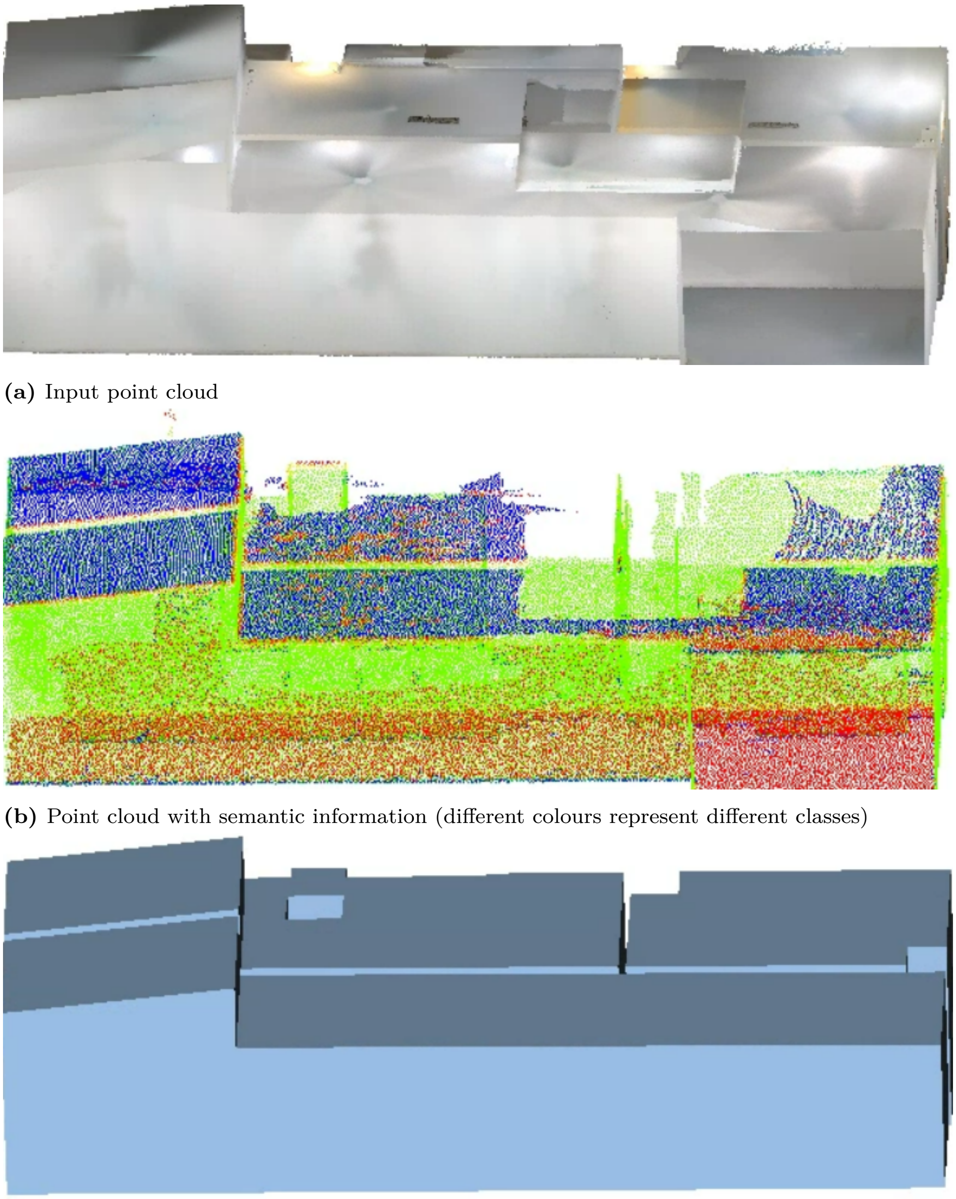
Figure 6.9: Plane selection result after optimisation (for each plane, when visualising from different directions, the colours are set differently: one side blue, one side green)



(a) Optimised Plane extraction result (Part of the ceiling is removed for better visualisation)

(b) Plane extraction result (all of the ceilings are removed for better visualisation)

Figure 6.10: Plane selection result of non-Manhattan environments



(a) Input point cloud

(b) Point cloud with semantic information (different colours represent different classes)

(c) Point cloud with semantic information

Figure 6.11: Rerult of another non-Manhattan-world environment

6.2.1 Parameter Study

Weighted Coefficients of Energy Terms

In Formula 6.6, $\lambda_{fitting}$ and $\lambda_{complexity}$ are two weight coefficients for the two corresponding energy terms which have impacts on the final optimisation results. Different results are shown from Figure A.1 to Figure A.4 (Figures can be found in Appendix A in order to save space in the main text). It can be observed that when increasing the weight coefficient of complexity, larger planes are extracted, which benefits the very large surfaces of walls. In the experiment, the $\lambda_{complexity}$ range between 0.4 and 0.8 is suitable for the point cloud.

Weighted Coefficients of Point Labels

In Equation 6.4, $coef(p)$ is the label coefficient depending on the label. In the experiment, if the point p is predicted as structural elements $coef(p)$ is set to 1, and $coef(p)$ is set to 0.1 if the point is predicted as other classes. As the objective is to extract the faces of structural elements, the coefficient for structural element classes should be higher than that for other classes. Therefore, the coefficient for structural element classes is kept to the value of 1 while the coefficient for other element classes is changed. In this section, the results of choosing different coefficients are shown from Figure B.1 to Figure B.5 (Figures can be found in Appendix B in order to save space in the main text).

It can be observed that with increasing the value of the coefficient for other classes, the points predicted as other classes have a larger impact on the final optimised model. In other words, some faces which do not belong to structural elements are included in the final optimised model, which does not match the goal of this step (extracting surfaces of structural elements). After analysing the results B.1 to Figure B.5, the author found the suitable value for the coefficient is around 0.1.

In summary, an approach to reconstruct structural elements from laser-scanned point clouds is proposed for non-Manhattan-World environments. Like the method proposed for Manhattan-world environments, it starts with extracting semantic information from 3D deep learning. Subsequently, planes are fitted by efficient RANSAC and then intersected to get over-segmented faces. Inspired by previous work (Ochmann et al. (2016), Ochmann et al. (2019), Nan and Wonka (2017)), the author sets up a binary linear programming problem to get the final surface model from intersected surfaces by considering both geometric information from point clouds and semantic information predicted by deep learning. As the steps to process raw data directly, like 3D deep learning and plane extraction, do not require the Manhattan-world assumption, the proposed automatic approach works and provides good results.

6.3 Conclusions and Discussions

In this chapter, an approach for reconstructing space-bounding elements for non-Manhattan-world buildings from laser-scanned point clouds is proposed. The proposed method starts with the point cloud semantic segmentation by deep learning first. Subsequently, planes are extracted from raw data and over-segmented into candidate faces. Then the face selection for the surfaces of elements is achieved by an optimisation problem. The approach uses geometric information and the semantic information extracted from neural networks because geometric information of space-bounding elements in the point cloud could be missing because of the occlusion, which makes previous methods perform worse in an occluded environment. By considering geometric information and semantic information predicted from deep learning,

In conclusion, the final output of the method includes point clusters with annotated semantic information, which shows the detailed information of captured surfaces and a simplified mesh model of the space-bounding elements. The non-Manhattan-world environments can be reconstructed well given a laser-scanned point cloud. However, there are still some limitations to the approach. Firstly, the proposed approach uses the prediction of deep learning as input, which makes its performance depend on the performance of deep learning. In order to get a well-performed deep learning model, a large amount of training data and complicated architecture designs are usually helpful. Secondly, the proposed approach can only reconstruct non-Manhattan-world buildings with planar surfaces. Curved surfaces are not considered in the method. At last, the method still only focuses on the space-bounding elements in the indoor environment. Other important elements are still missing, and the proposed method for these elements is presented in Chapter 7.

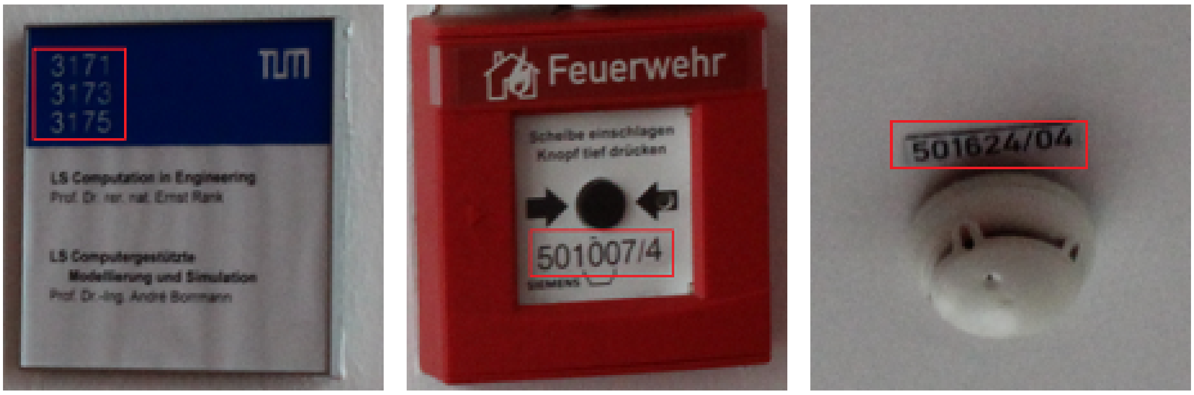
Chapter 7

Method to Enrich Digital Twins with Small Elements

This chapter proposes a novel approach that processes information from images and point clouds together to create an information-rich digital twin. The method focuses on 12 important and relatively small-scale elements (compared to walls, ceilings, and floors) in buildings: light switches, emergency switches, light fixtures, smoke alarms, escape signs, speakers, fire extinguishers, sockets, pipes, boards, door signs, elevator buttons, trash bins. In this part, I propose a novel framework to enrich a geometric building twin by fusing point cloud processing and object detection in images. The proposed method of information enrichment can be used to complete as-built models generated by other methods of creating geometric digital twins of space-bounding elements.

In particular, the proposed approach presents the following contributions:

- Because the performance of detecting small-scale elements directly in point clouds is significantly lower than in images, unlike most previous methods that exclusively use point clouds as input, the approach presented here extracts semantic information from images by deep learning and then maps the extracted semantic information to laser-scanned point clouds.
- While most of the previous approaches only detect primary elements (like ceilings, walls, floor, windows and doors), the proposed method includes small but highly relevant objects in the energy and the fire-safety sub-systems that are essential for maintaining and monitoring buildings (like smoke alarms, emergency switches).
- In order to create an information-rich building twin, other useful information (text and numbers) is detected in images by applying optical character recognition (OCR) technologies to detect object IDs and recognise object instances. Some examples of useful information in an indoor environment are shown in Figure 7.1. The detected machine-encoded texts include the room number on the door sign, as well as numbers or text corresponding to the detected objects, which helps to identify the object instance in the physical asset.


3171
3173
3175
TUM
LS Computation in Engineering
Prof. Dr. rer. nat. Ernst Rank
LS Computergestützte
Modellierung und Simulation
Prof. Dr.-Ing. André Borrmann
Feuerwehr
Scheibe einschlagen
Knopf tief drücken
501007/4
SIEMENS
501624/04

(a) Room number on door sign **(b)** Serial number on fire alarm switch **(c)** Serial number next to smoke alarm

Figure 7.1: Text information in building

This chapter is organised as follows: The proposed approach is introduced step by step in Section 7.1. The experiments and results are presented in Section 7.2. The conclusions and discussions are shown in Section 7.3.

7.1 Proposed Approach

Basically, the proposed method creates an information-rich digital twin of small objects in indoor environments and enriches the digital twin with semantic information. Specifically, the process includes creating more categories of objects in the digital twins of space-bounding elements in a building and adding relative text information to the final output. The overall process of the proposed method is illustrated in Figure 7.2. The inputs for the proposed method are point clouds acquired by laser scanners and videos or images captured in the same building area. The outputs are point clusters with labels and a mesh model for each element found. All points in one point cluster have an identical label. The overall goal is to create a comprehensive digital building model represented by mesh geometry and enriched with semantic information about the detected elements. To achieve this, information in 2D images is mapped onto a 3D laser-scanned point cloud. The method starts by detecting objects in images or videos by applying transfer learning technology. The next step is to construct a photogrammetric point cloud and align this point cloud to the laser-scanned point cloud. Subsequently, the semantic information from 2D images or videos is projected onto the 3D point cloud. After finding a best-fitting label for each point, the output point clusters of different objects can be obtained. In the final step, the author fits a pre-defined mesh model to each found instance.

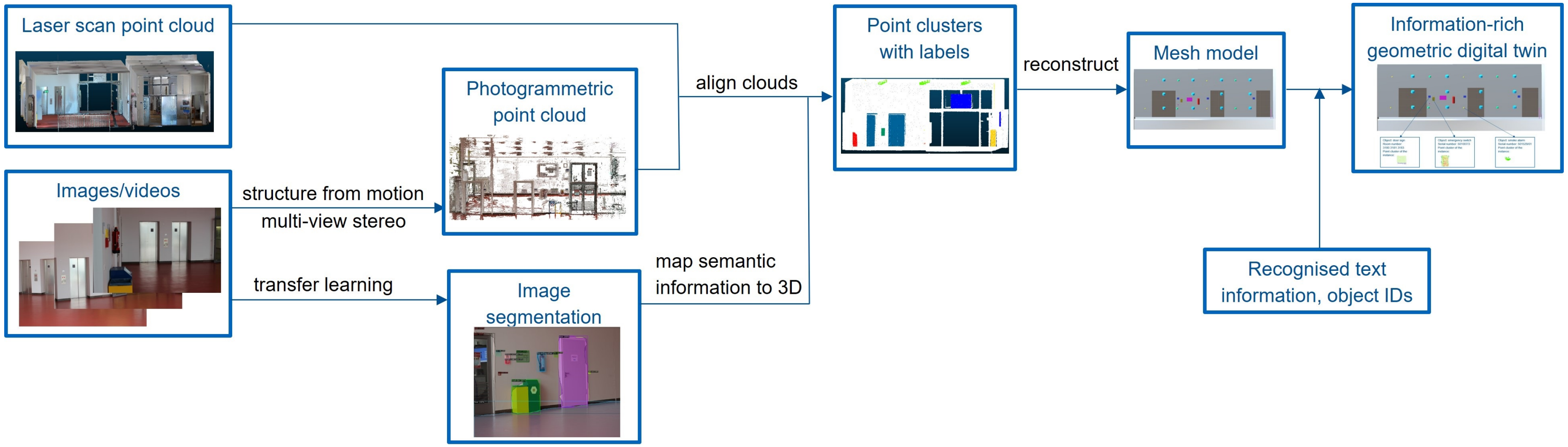

Laser scan point cloud
Images/videos
structure from motion
multi-view stereo
transfer learning
Photogrammetric point cloud
Image segmentation
align clouds
map semantic information to 3D
Point clusters with labels
reconstruct
Mesh model
Information-rich geometric digital twin
Recognised text information, object IDs

Figure 7.2: The overall procedure of the proposed method

7.1.1 Object Detection in Image

In this step, the method aims to detect the different objects from images or videos. Recently, the DNN (Krizhevsky et al., 2012), especially the proposed architecture of RCNN (Girshick et al., 2014), have proven effective in object detection in 2D images (Zhao et al., 2019). But the author still needs to prepare his own dataset because those publicly available datasets, such as Imagenet (Deng et al., 2009), one of the largest online available image datasets, do not contain all of the categories the author needs. Even if some of the target categories are present in Imagenet, such as fire alarms and fire extinguishers, there are no labelled instances available. Therefore, it is not possible to detect the target objects in images or videos that were captured in buildings by publicly available pre-trained models because these models are trained on a dataset lacking the categories required. The available networks must be re-trained for the application domain. In the conducted research, the author prepared his own dataset by manually labelling images that were captured in public buildings, more precisely office buildings on the inner-city campus of the Technical University of Munich (TUM). It should be noted that the images were not taken in the same building where the point clouds were captured.

In practice, there is no required minimum number of images when training a neural network. In Imagenet (Deng et al., 2009), categories like fire/smoke alarm and fire bell contain hundreds of labelled images. If the author follows a similar setup in that each category has hundreds of images, thousands of images are required for a dataset with 12 classes, which leads to a huge amount of labelling work. Considering the vast human effort to label these images manually, the author decided to use transfer learning techniques. As its name implies, transfer learning (Pan and Yang, 2009) means using the knowledge learned previously to solve new but related problems. When starting with a pre-trained model that has already been trained on thousands of images, the author does not need as many images as if training a network from scratch because the model has already "seen" and "learnt" from lots of images. In these cases, the author uses the pre-trained Mask-RCNN model (He et al., 2017) provided by Facebook (Wu et al., 2019) that has been trained on the COCO dataset (more than 100k images) (Lin et al., 2014).

Object detection in images results in finding a bounding box for a detected instance. Obviously, some regions within the bounding box do not belong to this instance, especially when the object is not a rectangle or inclined in the image. Since the author wants to map semantic information obtained in 2D images to the 3D point cloud in further steps, the author needs to reduce this kind of error here and apply image segmentation instead of instance detection. To this end, a variant of CNN called Mask RCNN (He et al., 2017) that detects objects in images by generating a mask for each instance is used. By doing so, a more precise contour of the object instance than the mere bounding box can be found. Some results of image segmentation and bounding box prediction of various objects are illustrated in Figure 7.3.

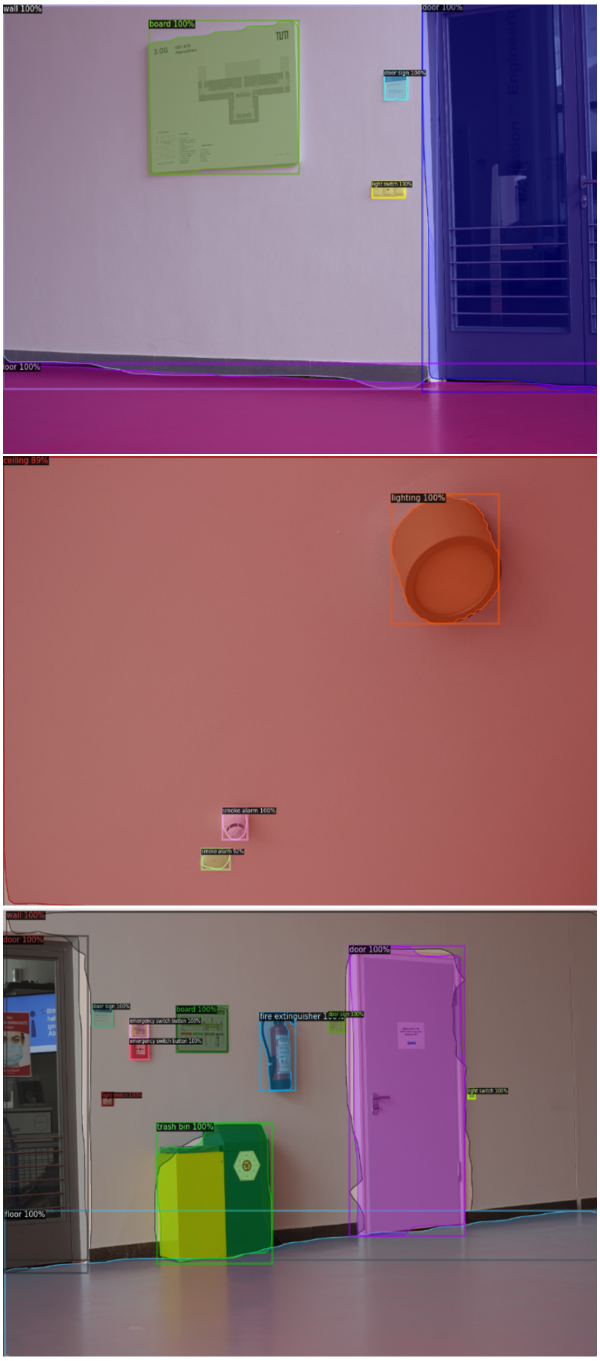
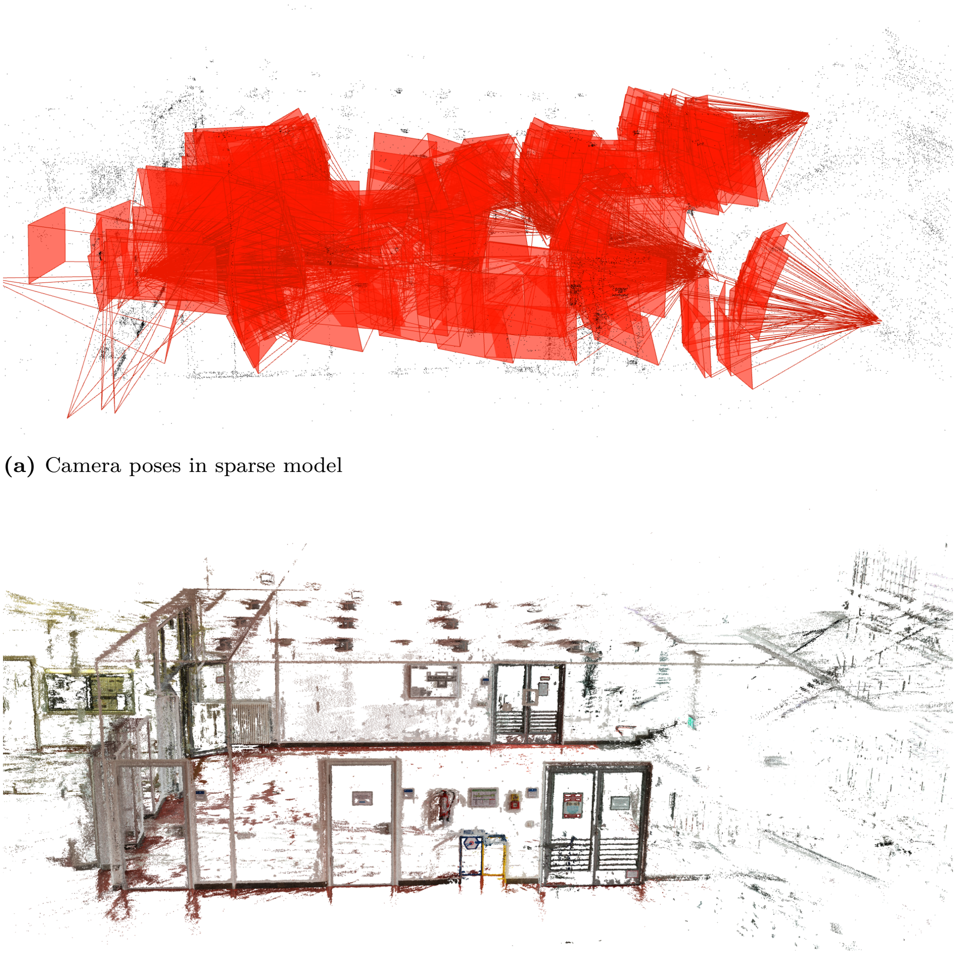


Figure 7.3: Object detection result by image segmentation mask and bounding box

7.1.2 Creating Photogrammetric Point Clouds

In (Braun and Borrmann, 2019), the author uses the photogrammetric point cloud to connect images and Building Information Modeling (BIM) models. Similarly, in the proposed approach, the photogrammetric point cloud acts as the bridge that connects 2D information in images with 3D information in the laser-scanned point cloud. In the reconstruction process, the extrinsic and intrinsic camera parameter matrices are estimated. Images or videos are supposed to be taken from different viewpoints within the area and cover as much information as possible. In this approach, the author applies COLMAP (Schönberger et al., 2016) (Schönberger and Frahm, 2016), an open-source Structure-from-Motion (SfM) and Multi-View Stereo (MVS) software, to reconstruct photogrammetric point clouds. The input of SfM is a set of overlapping images taken from different viewpoints. It starts with feature detection and extraction, continues with feature matching and geometric verification, and then reconstructs the object in 3D space, including the reconstructed intrinsic and extrinsic camera parameters of all images. MVS takes the output of SfM to compute depth and normal information for pixels in all images and creates a dense point cloud of the scene.

The estimated camera poses (position and orientation) of each image and the reconstructed sparse photogrammetric point cloud are illustrated in Figure 7.4. As we can see, the edges are reconstructed quite well, while the plane faces of elements like walls, ceilings, and floors are missing. This is because almost no features can be detected and extracted on these weakly textured surfaces, like a planar white wall, in the SfM process. However, these weakly textured surfaces can be captured quite well by laser scanners. This is one of the reasons why the author proposes the use of both laser-scanned point clouds and images to create sufficiently detailed and complete digital twins. In this way, all of the required information can be acquired by using both techniques to capture buildings.



(a) Camera poses in sparse model

(b) Reconstructed dense point cloud

Figure 7.4: Example of estimated camera poses and reconstructed point cloud

7.1.3 Point Clouds Alignment

Laser scanners measure the distance by transmitting light and sensing the return from objects (Oguchi et al., 2011) so that laser-scanned point clouds represent the actual scale of the environment. In contrast, photogrammetric point clouds extract information from 2D images – they do not represent the actual scale in world units unless additional information is considered, such as the size of an object. To perform the necessary registration of the two point clouds, the author aligns the photogrammetric point cloud with the laser-scanned point cloud so that the photogrammetric point cloud also represents the environment in its actual size.

The photogrammetric point cloud is transformed to the coordinate of the laser-scanned point cloud by

$$\mathbf{Q} = \mathbf{M}\mathbf{P}, \tag{7.1}$$

where $\mathbf{P}$ denotes the point set of the photogrammetric point cloud, $\mathbf{Q}$ denotes the point set of the photogrammetric point cloud transformed to the coordinate of the laser-scanned point cloud, $\mathbf{M}$ denotes the transformation matrix that transforms points from the coordinate of the photogrammetric point cloud to the coordinate of the laser-scanned point cloud.

4×4 transformation matrices are widely used to represent non-linear transformations in 3D space. In this approach, two steps are used to determine the 4×4 transformation matrix: the rough alignment step and the refinement step. In the rough alignment step, the author uses 4 pairs of points from the photogrammetric point cloud and laser-scanned point cloud by using

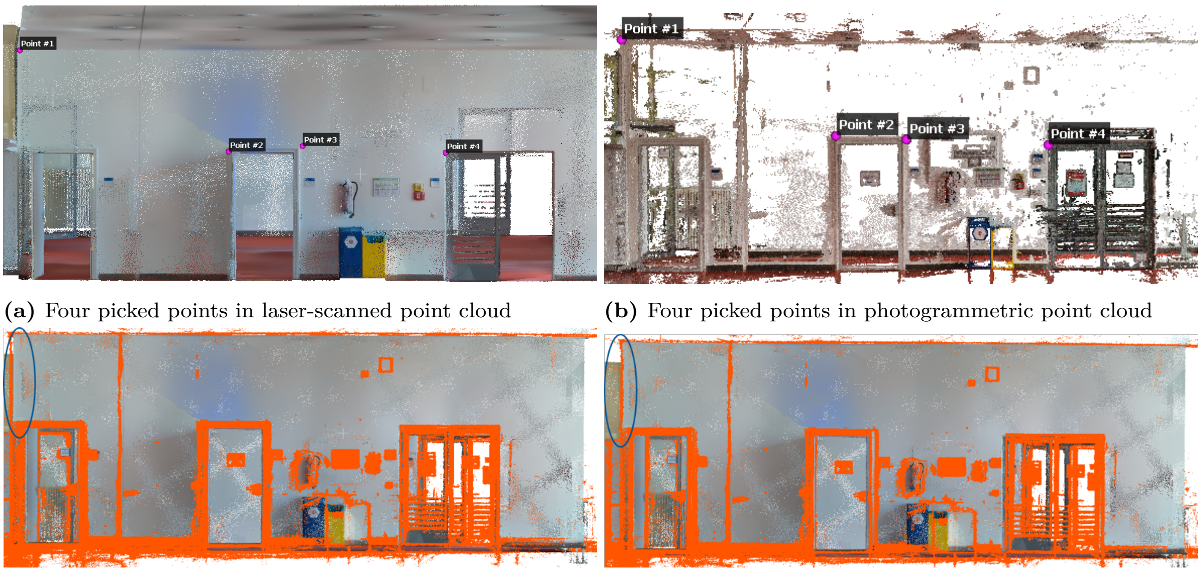
$$[\,\mathbf{q_1}\;\mathbf{q_2}\;\mathbf{q_3}\;\mathbf{q_4}\,] = \mathbf{M_1}[\,\mathbf{p_1}\;\mathbf{p_2}\;\mathbf{p_3}\;\mathbf{p_4}\,]^{-\mathbf{1}}, \tag{7.2}$$

where $\mathbf{M_1}$ denotes the roughly estimated transformation matrix from photogrammetric point cloud coordinate to laser-scanned point cloud coordinate, $\mathbf{p_1}$ and $\mathbf{q_1}$, $\mathbf{p_2}$ and $\mathbf{q_2}$, $\mathbf{p_3}$ and $\mathbf{q_3}$, $\mathbf{p_4}$ and $\mathbf{q_4}$ are four pairs of points in the photogrammetric point set $\mathbf{P}$ and laser scanning point set $\mathbf{Q}$.

In this step, the author only needs to select points roughly and get a rough alignment result. These point pairs can be chosen at random and could be any key points in point clouds, such as room and door corners, the centre of an object, etc. After rough alignment, the Iterative Closest Point (ICP) algorithm (Besl and McKay, 1992) is applied, to refine the alignment and obtain the refinement transformation matrix $\mathbf{M_2}$. The overall transformation matrix $\mathbf{M}$ can be computed by

$$\mathbf{M} = \mathbf{M_2}\mathbf{M_1}. \tag{7.3}$$

The photogrammetric point cloud can then be transformed to the coordinates of the laser-scanned point cloud by applying Equation 7.1. This alignment process is illustrated in Figure 7.5. When comparing the marked area in Figure 7.5c with that in Figure 7.5d, it is clear that the refinement step improves the alignment result.



(a) Four picked points in laser-scanned point cloud

(b) Four picked points in photogrammetric point cloud

(c) Rough alignment result (Photogrammetric point cloud is set to red colour)

(d) Refined alignment result (Photogrammetric point cloud is set to red colour)

Figure 7.5: The alignment process of photogrammetric and laser-scanned point cloud

7.1.4 Find Visible Points in Images

In this step, what is determined is whether a point from the laser-scanned point cloud is visible in each image that is used to reconstruct the photogrammetric point cloud. Because the photogrammetric point cloud and the laser-scanned point cloud are aligned already, the estimated parameters (extrinsic and intrinsic camera parameters) from the reconstruction process are also mapped into 3D space. The extrinsic camera matrix and intrinsic parameter matrix are known for each image or frame of a video. Based on the matrices, which points are visible at each camera position and captured in the corresponding image can be found.

As the transformation matrix that transforms points from a photogrammetric point cloud coordinate to a laser-scanned point cloud coordinate is $\mathbf{M}$, any point $\mathbf{p} = \left[x_0, y_0, z_0\right]^{\mathbf{T}}$ in the original laser-scanned point cloud $\mathbf{S}$ can be transformed to the coordinate of the photogrammetric point cloud by

$$\begin{bmatrix} x_1 \\ y_1 \\ z_1 \\ d_1 \end{bmatrix} = \mathbf{M^{-1}} \begin{bmatrix} x_0 \\ y_0 \\ z_0 \\ 1 \end{bmatrix}, \tag{7.4}$$

where $\left[x_0, y_0, z_0, 1\right]^T$ is the homogeneous coordinates of this point $\mathbf{p}$, $\mathbf{M^{-1}}$ is the inverse matrix of $\mathbf{M}$, and $\left[x_1, y_1, z_1, d_1\right]^T$ is the new calculated homogeneous coordinates of the point in the coordinate of the photogrammetric point cloud. Normalization is then applied by dividing each vector component by d_1,

$$\begin{bmatrix} x_2 \\ y_2 \\ z_2 \\ 1 \end{bmatrix} = \frac{1}{d_1} \begin{bmatrix} x_1 \\ y_1 \\ z_1 \\ d_1 \end{bmatrix}, \tag{7.5}$$

where $\begin{bmatrix} x_2, y_2, z_2, 1 \end{bmatrix}^T$ is the normalized homogeneous coordinate vector of point $\mathbf{p}$ in the coordinate of photogrammetric point cloud.

The next step is to transform every point from the coordinate of the photogrammetric point cloud to the camera coordinate of the image. In this chapter, the author uses $\mathbf{N}$ to denote the whole image set that is used to reconstruct the photogrammetric point cloud, $\mathbf{n}_i$ to denote the i^{th} image in the image set $\mathbf{N}$. For one single image $\mathbf{n}_i$, $\mathbf{M}^i_{ext}$ and $\mathbf{M}^i_{int}$ denote the corresponding camera extrinsic and intrinsic parameter matrices. The extrinsic parameter matrix can be defined as

$$\mathbf{M}^i_{ext} = \begin{bmatrix} & \mathbf{R}_i & & \mathbf{T}_i \\ 0 & 0 & 0 & 1 \end{bmatrix}, \tag{7.6}$$

where $\mathbf{R_i}$ is the 3×3 rotation matrix $\mathbf{R_i} = \begin{bmatrix} r^i_{11} & r^i_{12} & r^i_{13} \\ r^i_{21} & r^i_{22} & r^i_{23} \\ r^i_{31} & r^i_{32} & r^i_{33} \end{bmatrix}$,

and $\mathbf{T_i}$ is the 3×1 translation matrix $\mathbf{T_i} = \begin{bmatrix} t^i_1 \\ t^i_2 \\ t^i_3 \end{bmatrix}$ of the image $\mathbf{n}_i$.

The intrinsic parameter matrix can be represented by

$$\mathbf{M}^i_{int} = \begin{bmatrix} f_x & s & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix}, \tag{7.7}$$

where f_x and f_y are the effective focal length of the camera measured in units of image pixels in the horizontal and vertical directions, c_x and c_y are the pixel coordinates of the principal point. Additionally, s denotes the skew coefficient for the camera. This is zero if the image axis is perpendicular to the image plane. It should be noticed that no distortion is assumed here. 3D points can be then transformed into camera coordinates by

$$\begin{bmatrix} x_3 \\ y_3 \\ z_3 \\ 1 \end{bmatrix} = \mathbf{M}^i_{ext} \begin{bmatrix} x_2 \\ y_2 \\ z_2 \\ 1 \end{bmatrix} = \begin{bmatrix} r^i_{11} & r^i_{12} & r^i_{13} & t^i_1 \\ r^i_{21} & r^i_{22} & r^i_{23} & t^i_2 \\ r^i_{31} & r^i_{32} & r^i_{33} & t^i_3 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_2 \\ y_2 \\ z_2 \\ 1 \end{bmatrix} \tag{7.8}$$

and subsequently transformed to the image plane by computing

$$\begin{bmatrix} x_4 \\ y_4 \\ z_4 \end{bmatrix} = \mathbf{M}^i_{int} \begin{bmatrix} x_3 \\ y_3 \\ z_3 \end{bmatrix} = \begin{bmatrix} f_x & s & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_3 \\ y_3 \\ z_3 \end{bmatrix}, \tag{7.9}$$

where x_3, y_3, z_3 are coordinates in the camera coordinate, and x_4, y_4, z_4 are the perspective projected coordinates on the image coordinate. By homogeneous coordinate normalisation, the image coordinates of the projected point in the image plane can be obtained:

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \frac{1}{z_4} \begin{bmatrix} x_4 \\ y_4 \\ z_4 \end{bmatrix}, \tag{7.10}$$

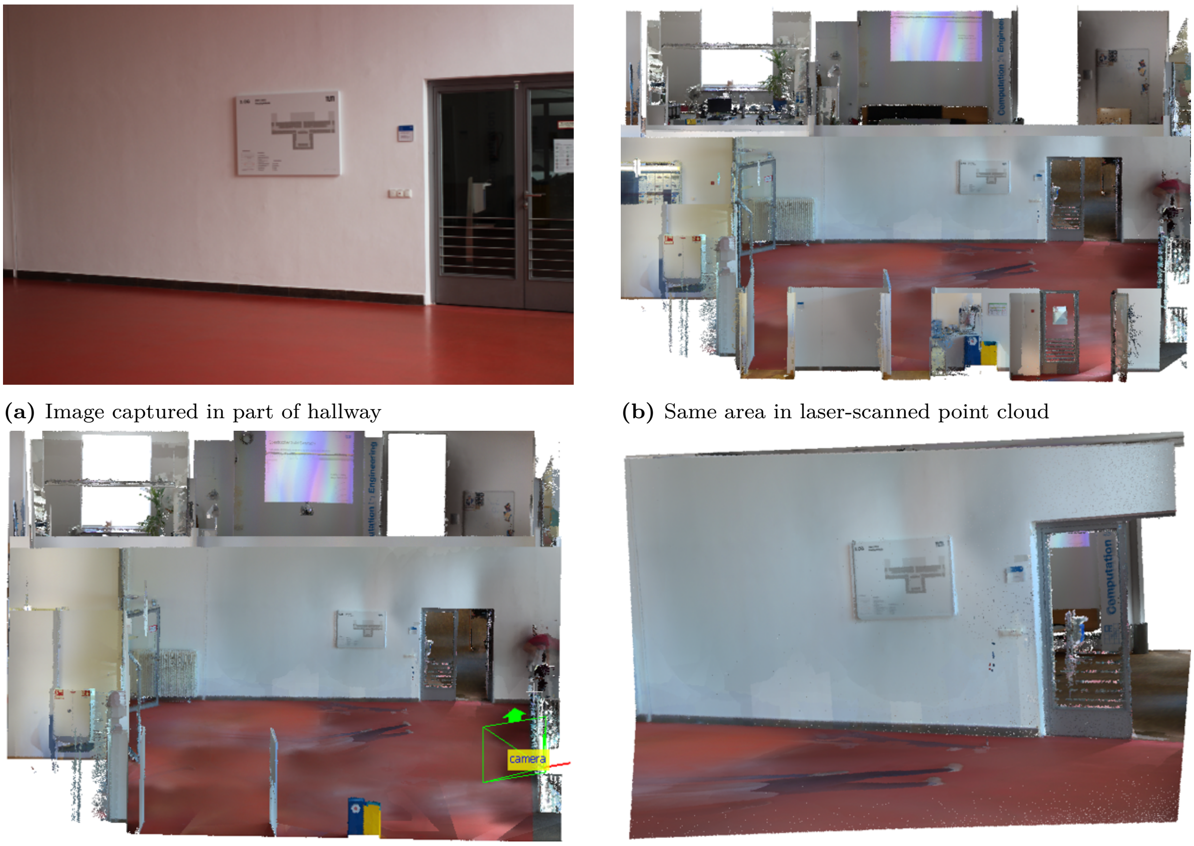
where u and v are the pixel coordinates in the horizontal and vertical direction in the image plane.

By using the Equations 7.4 to 7.10, all points in the original laser-scanned point cloud can be projected into the image plane. However, there are points in the cloud that are not in the field of view of the given camera pose and intrinsic parameters. Assuming the dimension of the image in pixels is $W \times H$, if a point (x_0, y_0, z_0) in the original laser-scanned point cloud and its projected point in the image plane (u, v) can be seen in the image, the point should follow these conditions:

$$0 \leq u \leq W, 0 \leq v \leq H. \tag{7.11}$$

The process of checking the visibility of laser-scanned points for one image is illustrated in Figure 7.6. As we can see in subfigure 7.6d, the visible area shown in the laser-scanned point cloud is identical to the image scene.

Up to this step, the visibility of a point is only determined by the camera parameters. That means that as long as the points fulfil Condition 7.11, they are considered visible points, which makes the camera see "through" the wall. As shown in Figure 7.7, it is obvious that some points should not be visible, like points behind the surface of the wall.



(a) Image captured in part of hallway

(b) Same area in laser-scanned point cloud

(c) Transform point cloud to camera frame (camera at origin)

(d) Visible points from laser-scanned point cloud at camera pose

Figure 7.6: Process of finding visible points in the image (the ceiling points in the point cloud are removed for better visualisation)

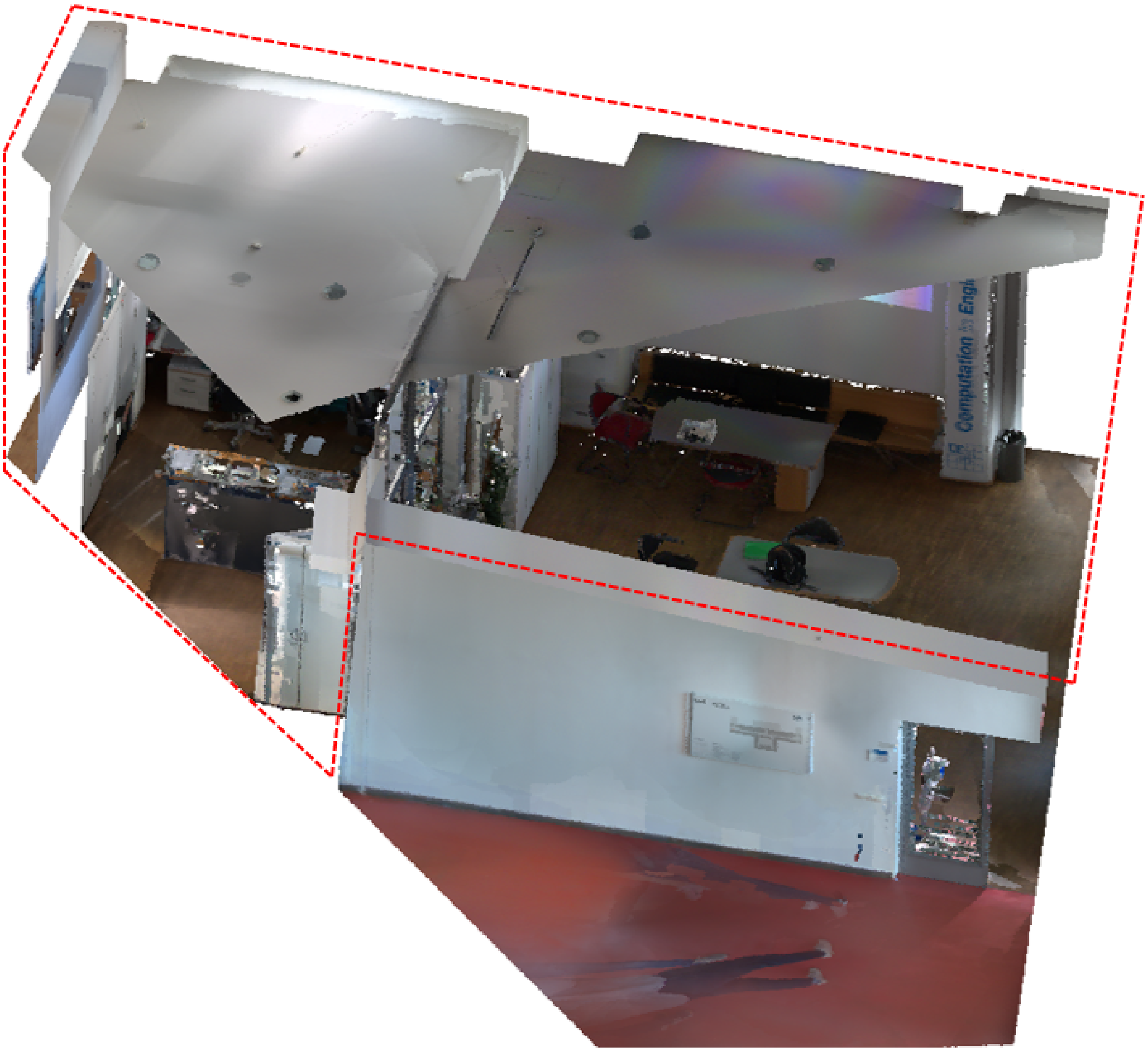


Figure 7.7: Top view of visible points at camera position in Figure 7.6. Points behind the wall (within the red dash line) are actually not visible from the camera pose.

The raycasting method (Roth, 1982) is used to remove those points that should not be seen at the current camera position. However, rays might pass through the point cloud without intersecting any points because point clouds are actually discrete points in 3D space. Therefore, point clouds are usually voxelised before raycasting (Laine and Karras, 2010). Figure 7.8 shows how raycasting works in a voxelised point cloud. Rays shoot from the camera position to each point in the point cloud. While a dark blue voxel means there are points within the voxel, a light blue voxel indicates no points in the voxel. If a ray starting from the camera does not pass through any other dark blue voxels, its target point is visible at the camera position. In contrast, if a ray passes through at least one other dark voxel before reaching the target point, this target point is occluded by other voxels in between.

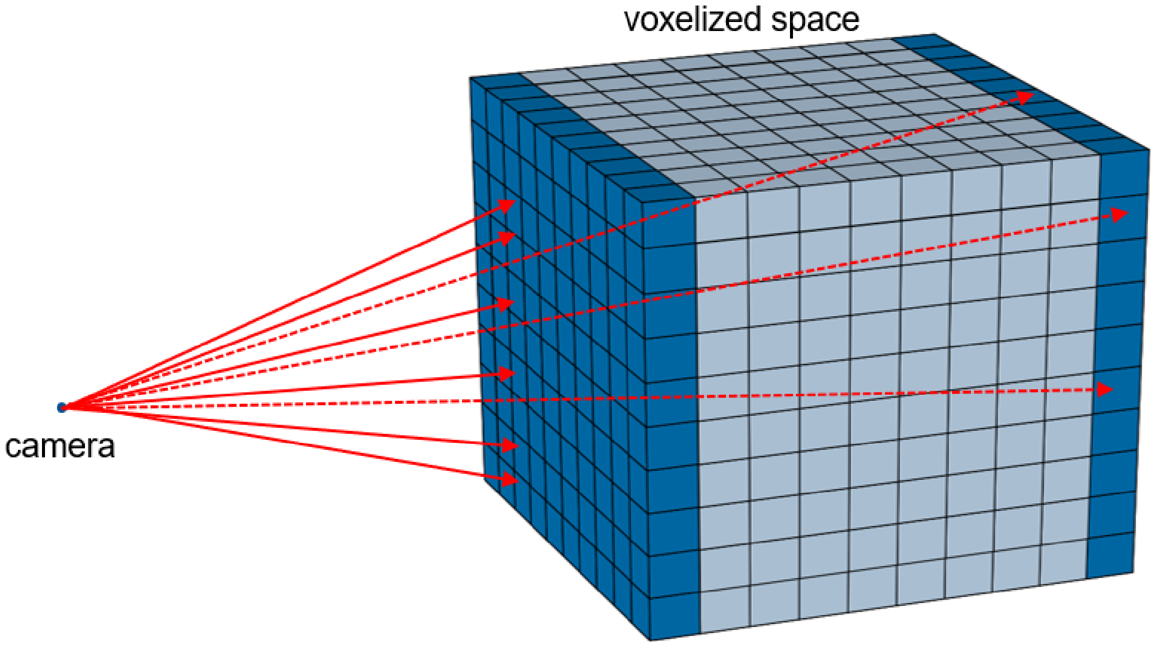


Figure 7.8: Raycasting method in voxelised point cloud. There are points in dark blue voxels but no points in light blue voxels. Rays of dotted lines starting from the camera intersect other dark blue voxels before reaching the target voxel. These target voxels are occluded by the voxels between the camera and themselves.

The remaining visible points after applying the raycasting method to the point cloud are shown in Figure 7.9. In the raycasting process, the voxel size has an enormous impact on performance. A further discussion on finding the best voxel size is presented in Section 7.2.5.

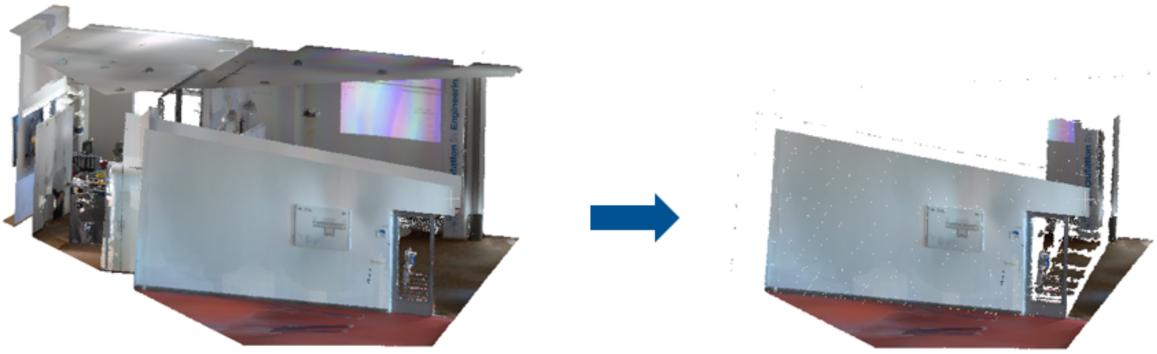


Figure 7.9: Apply raycasting to get visible points at camera position

7.1.5 Map 2D Semantic Information to 3D Space

In this step, the semantic information detected from 2D images or videos in Section 7.1.1 is mapped to the 3D space. The author uses Mask-RCNN (He et al., 2017) to detect objects in images, and the result for each detected instance (like a board, a smoke alarm, etc.) is a mask. The mask is a matrix that is exactly the same size as the input image but has only two values, 0 and 1. While pixels with a value of 0 are the background, pixels with a value of 1 are where the detected instance is located in the image. As shown in Figure 7.10c, 7.10e, and 7.10g, when a mask is applied to an image, only the image area that belongs to the detected area can be seen.

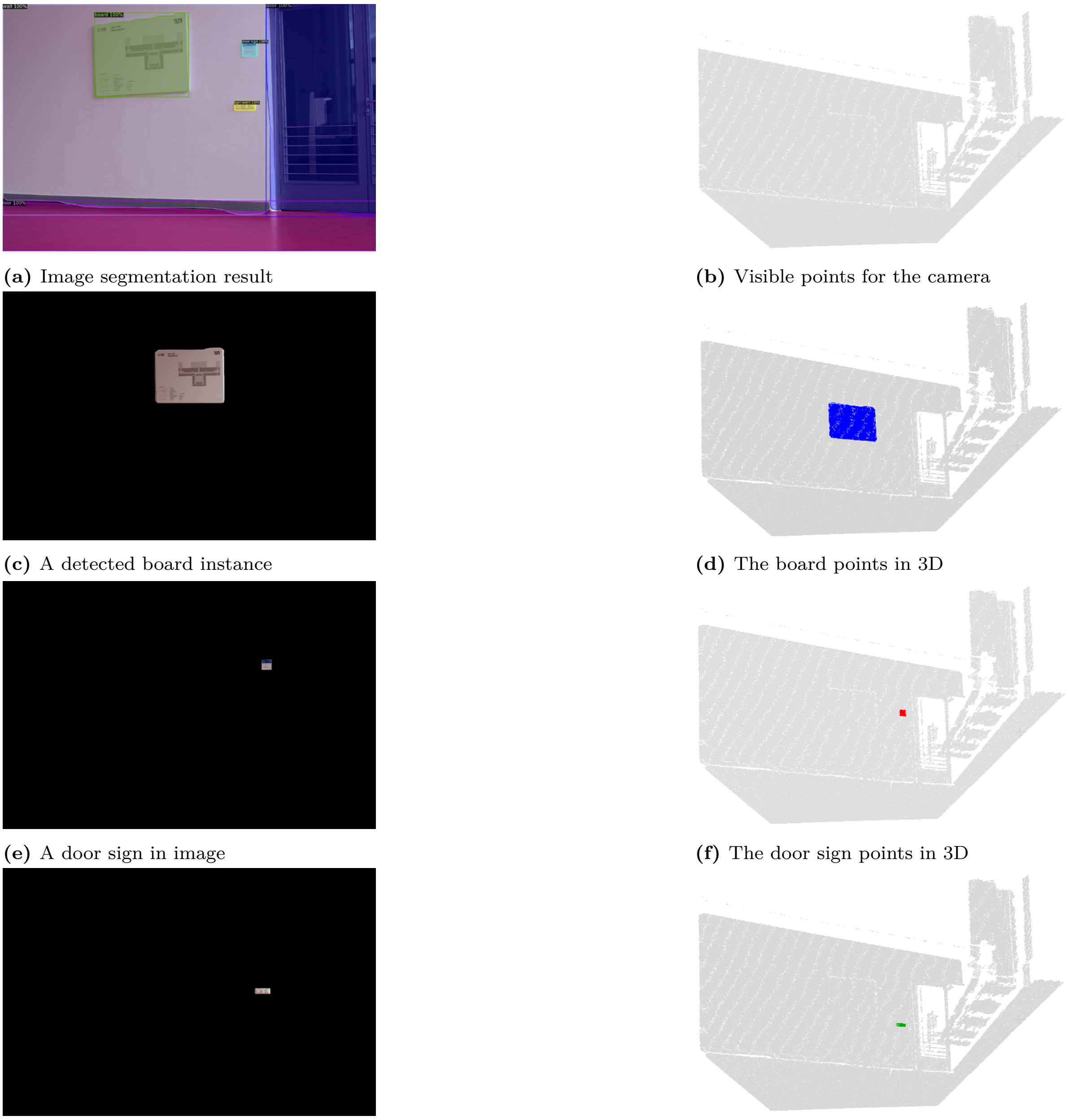
In the previous step, all visible points (x_0, y_0, z_0) in 3D space are already transformed to 2D coordinates (u, v) in the image plane. At this step, it checks whether every point in the image plane is in the predicted segmentation mask or the background area. Points located in the instance mask of three categories are shown in Figure 7.10c, 7.10e, and 7.10g for example.

Because images/videos are used to reconstruct the photogrammetric point cloud, many images have overlapping areas. In order to record semantic information from all images, an $M \times N$ matrix $\mathbf{L}$ is used to accumulate predicted information from all images, where M denotes the number of categories and N denotes the number of points in the laser-scanned point cloud. If the k^{th} point's projection in the image plane is within a mask of category j, the term $\mathbf{L}_{j,k}$ in the matrix $\mathbf{L}$ would be increased by 1, where $1 \leq j \leq M$ and $1 \leq j \leq L$.

One point in the laser-scanned point cloud is usually visible in multiple images, and the predicted labels from these images might be different. Therefore, it is necessary to retain all of the information and find the best-fitting label prediction for each point in later steps. The pseudocode of the method proposed in Section 7.1.3 to 7.1.5 is shown in Algorithm 2.

Algorithmus 2 The mapping algorithm from 2D to 3D.

```
Input:
One point $\mathbf{s}_k \in \mathbf{S}$, laser-scanned point cloud set $\mathbf{S}$;
Image set used to reconstruct the photogrammetric point cloud $\mathbf{N}$;
For image $\mathbf{n}_i \in \mathbf{N}$, camera extrinsic and intrinsic parameter matrices $\mathbf{M}^i_{ext}$ and $\mathbf{M}^i_{int}$;
Predicted segmentation mask $\mathbf{m}^i_j \in \mathbf{K^i}$ for image $\mathbf{n}_i$, category $j$, $\mathbf{K^i}$ denotes all predicted masks for image $\mathbf{n}_i$;
Transformation matrix from photogrammetric point cloud to laser-scanned point cloud $\mathbf{M}$;
Function to check whether a point is visible at a camera position $\alpha()$;
Function to check whether a point belongs to a mask $\beta()$;
Initialize:
Matrix used to count labels for all points in point cloud $\mathbf{L} \leftarrow \mathbf{O}$;
Algorithm:
for $\mathbf{s}_k \in \mathbf{S}$ do
    Point in the coordinate of photogrammetric point cloud $\mathbf{p}_k = \mathbf{M}^{-1} \times \mathbf{s}_k$
    for $\mathbf{n}_i \in \mathbf{N}$ do
        Point in image plane $\mathbf{c}_k = \mathbf{M}^i_{int} \times \mathbf{M}^i_{ext} \times \mathbf{p}_k$
        if $\alpha(\mathbf{c_k})$ is FALSE then
            continue
        end if
        for $\mathbf{m}^i_j \in \mathbf{K^i}$ do
            if $\beta(\mathbf{c_k})$ is TRUE then
                count label $j$ for point $k$ once, $\mathbf{L}_{j,k} = \mathbf{L}_{j,k} + 1$
            end if
        end for
    end for
end for
return L
```



(a) Image segmentation result

(b) Visible points for the camera

(c) A detected board instance

(d) The board points in 3D

(e) A door sign in image

(f) The door sign points in 3D

(g) A light switch in image

(h) The light switch in 3D

Figure 7.10: Image segmentation masks and corresponding points in 3D of different instances

7.1.6 Find Best-fitting Labels for Points

Given the results from previous steps, this step aims to find a best-fitting label for each point in 3D from the $M \times N$ label matrix $\mathbf{L}$.

Two values are used to determine the best label for each point. For one point $\mathbf{p}_i$ in the laser-scanned point cloud, N_i is the number of images where the point can be seen, $\mathbf{L}_{j,i}$ is the number of images where the point is within the predicted mask of category j. But it should be noted that N_i is not equal to the sum of N_i^j for all categories because a point could also be located in the "background" area instead of the mask area. Therefore, the author uses two values to represent how certain the label assigned to the i^{th} point $\mathbf{p}_i$ is:

$$U_i = \max_{1 \leq j \leq M} \mathbf{L}_{j,i} / N_i, \tag{7.12}$$

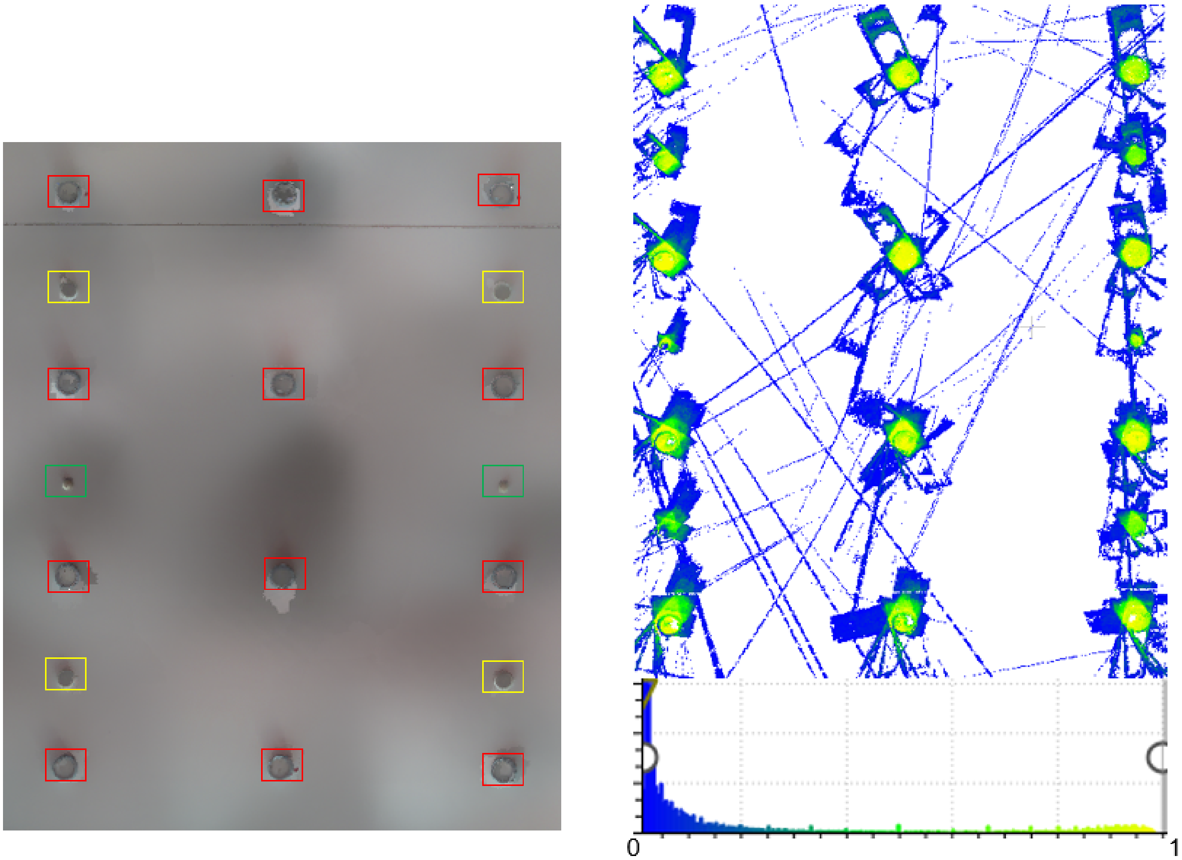
$$V_i = \max_{1 \leq j \leq M} \mathbf{L}_{j,i} / \sum_{j=1}^{M} \mathbf{L}_{j,i}. \tag{7.13}$$

Because the pixels at the border of the predicted mask area can probably be mapped to an object's surrounding points that do not belong to the object (for example, some points on the ceiling are predicted as points of a smoke alarm), these wrongly predicted points need to be removed. Unlike the points of an object, these neighbouring points do not appear in all images of the object. Moreover, some of them may only appear in one image but are predicted as object points. Therefore, it is not enough to rely solely on prediction accuracy from all images. The value U_i is used to filter the surrounding points out and how it works is illustrated in Figure 7.11.

Figure 7.11a is a part of the point cloud that shows the ceiling and three kinds of objects (lighting, speaker, smoke alarm) mounted to it from the bottom view. Figure 7.11b shows the distribution of U_i. Many points on the ceiling are predicted as a point of the object because the prediction is mapped from 2D images that are taken from different views.

Most of the surrounding points (ceiling points) are distributed in the low-value range of U_i. Figure 7.12a and Figure 7.12b show the points left after filtering out those points with the criteria $U_i > 0.5$ and $U_i > 0.7$. Objects' points can be extracted from their neighbouring points on the ceiling.

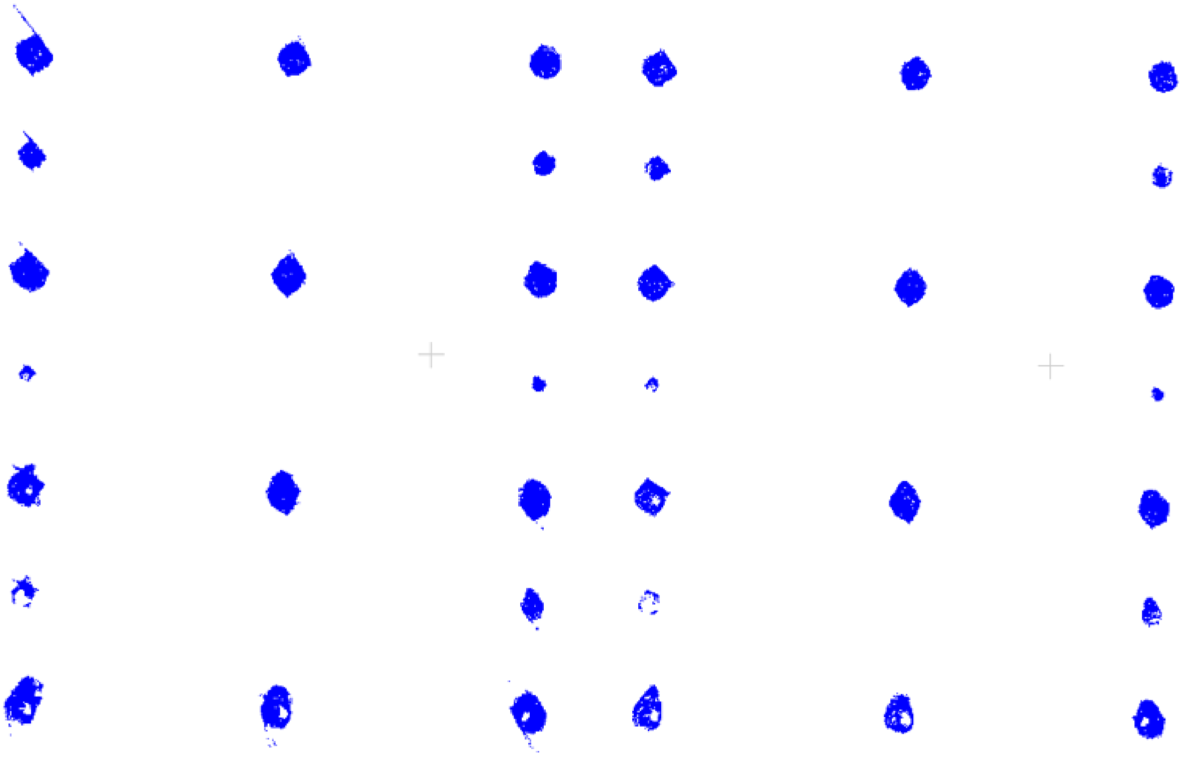
Unlike U_i, which aims to remove surrounding points of an object, V_i is used to show how certain the method is when assigning a class label with a point. Figure 7.13a shows the distribution of how certain the method is when assigning the label that occurs mostly as the class of the point for the same area. In this case, it is quite certain that the assigned labels are correct as most points are located in the range close to 1. Figure 7.13a shows points in different colours according to their assigned labels.



(a) Bottom view of a ceiling and mounted objects (red box: lighting, yellow box: speaker, green box: smoke alarm).

(b) Mapping prediction to point cloud and distribution of U_i

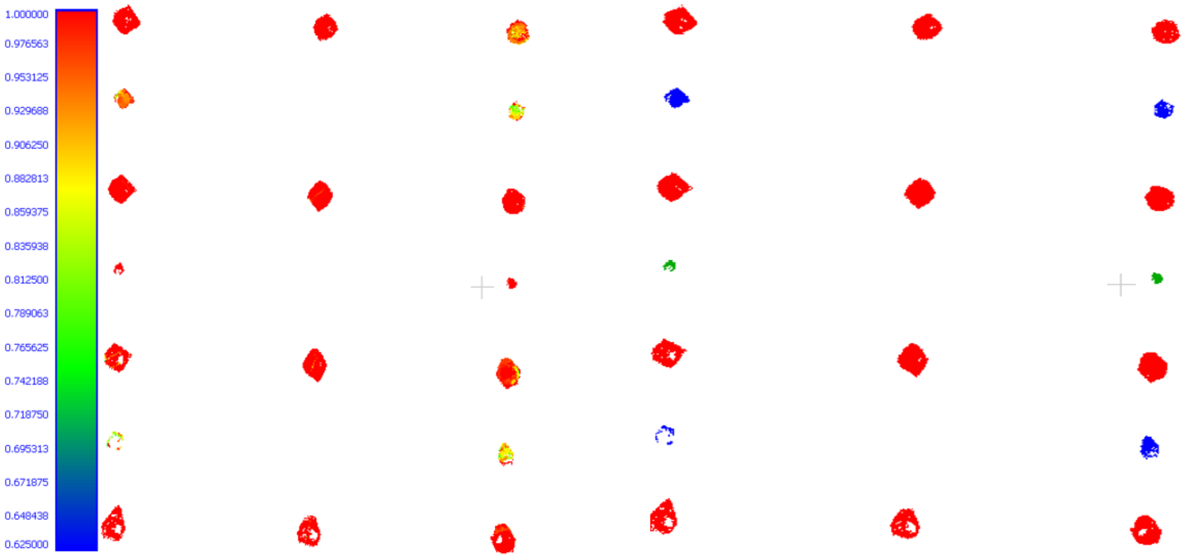
Figure 7.11: The distribution of U_i for a part of the point cloud of a ceiling



(a) Remaining point cloud after filtering $U_i > 0.5$)

(b) Remaining point cloud after filtering $U_i > 0.7$)

Figure 7.12: The remaining point cloud by filtering out ceiling points)



(a) The distribution of V_i (assigning the corresponding label in V_i to the point)

(b) Points of different classes (red: light, blue: speaker, green: smoke alarm)

Figure 7.13: The distribution of V_i and the extracted points of different classes

7.1.7 Fit Shape to Point Custers

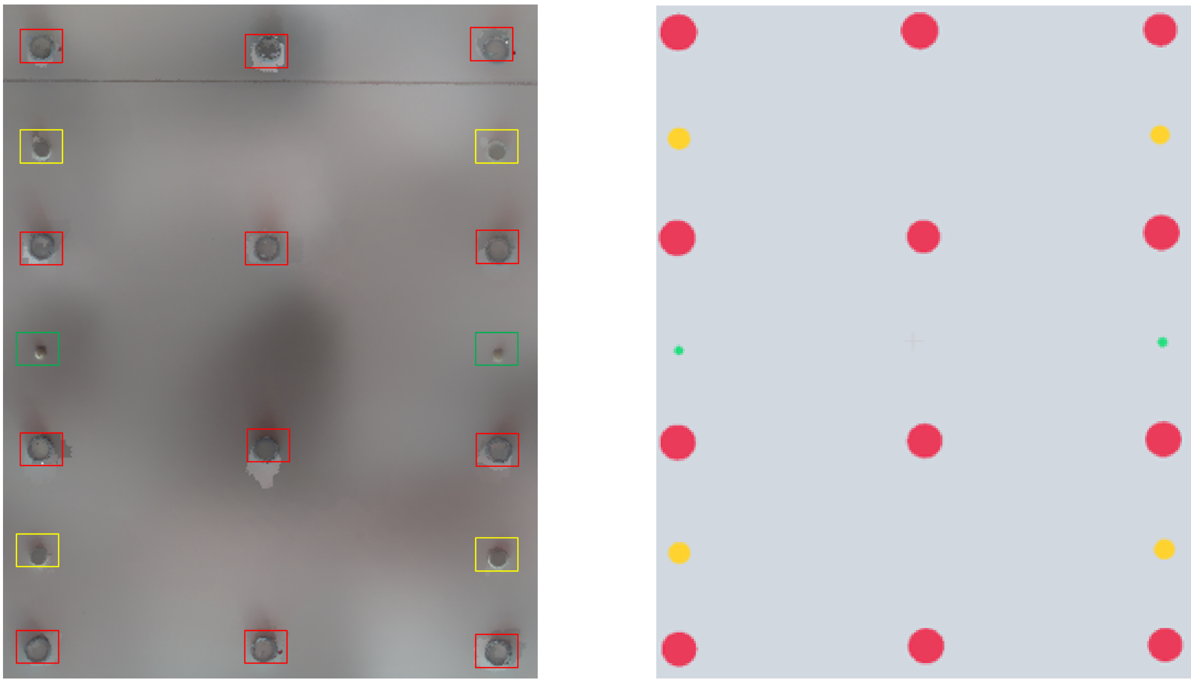
In this step, the author wants to fit a geometric shape to each extracted point cluster. Different object types are reconstructed by varying strategies.

For small objects mounted on the ceiling and wall (like smoke alarms, sockets, and switches), the extracted point clusters from the previous section are projected on the plane of the ceiling or wall. By then fitting simple geometric shapes (like circles and rectangles) in the wall or ceiling plane, the location and size in the 2D plane can be found. The reason that the author chooses to fit geometric shapes in 2D planes rather than in 3D point clouds is:

- Some surfaces of the elements might not be captured when capturing buildings with a laser scanner. It is hard to fit geometric shapes in the 3D point cloud directly, especially for small elements (like smoke alarms) that lack points on their surface.
- Some elements are commonly standardised elements (sockets, light switches, smoke alarms) whose instances are identical across the entire facility. Fitting shapes in the 2D plane can also reduce computing costs.

The random sample consensus (RANSAC) algorithm (Fischler and Bolles, 1981) is used to fit circles for cylindrical objects (such as a light, speaker, smoke alarm) and rectangles for "cuboid-like" objects (socket, switch, door sign, board, elevator button). Then the next step is to extrude the 2D shapes from the wall or ceiling plane by default thickness (if available) or estimate the thickness of the object in the 3D point cluster by finding the maximum distance to the plane. The fitting circles of three classes of objects (light, speaker, smoke alarm) on the ceiling plane are shown in Figure 7.14 and corresponding extruded cylinders are shown in Figure 7.15 by way of example.

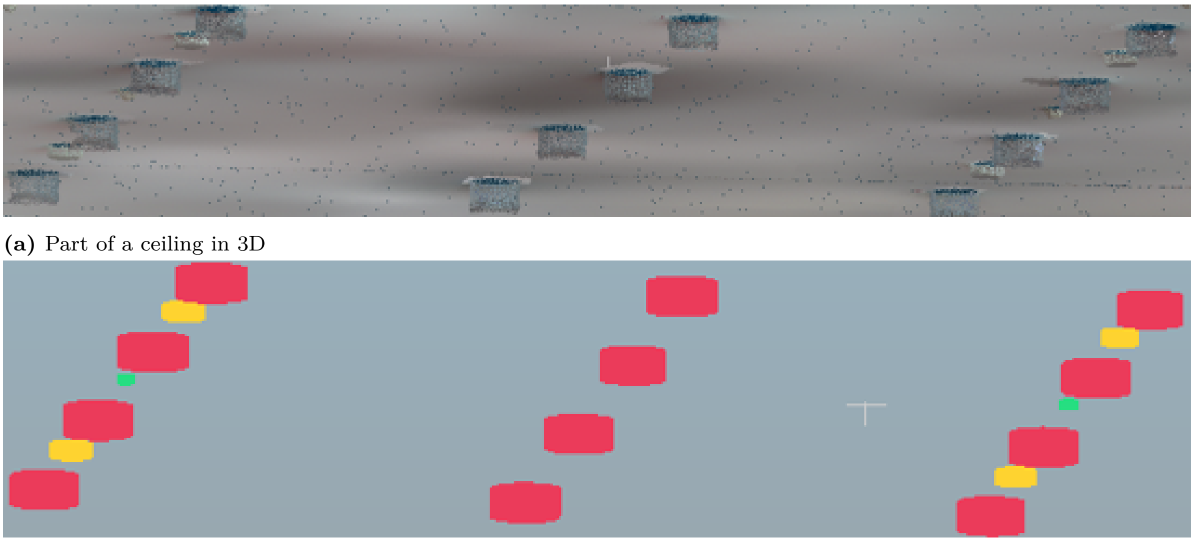
With regard to pipes and fire extinguishers that are usually cylindrical, RANSAC is used to fit a cylinder to the point cluster and find its dimension and position. The extracted cylinder of a fire extinguisher is illustrated in Figure, 7.16 for example. As shown in Figure 7.16c, only one cylinder is reconstructed in this step, based on the major part of the fire extinguisher body. A more detailed structure of the fire extinguisher body and hose pipe would be ignored.



(a) Bottom view of part of the ceiling (Red box: lighting, yellow box: speaker, green box: smoke alarm)

(b) Fitting result on ceiling plane (red: lighting, yellow: speaker, green: smoke alarm)

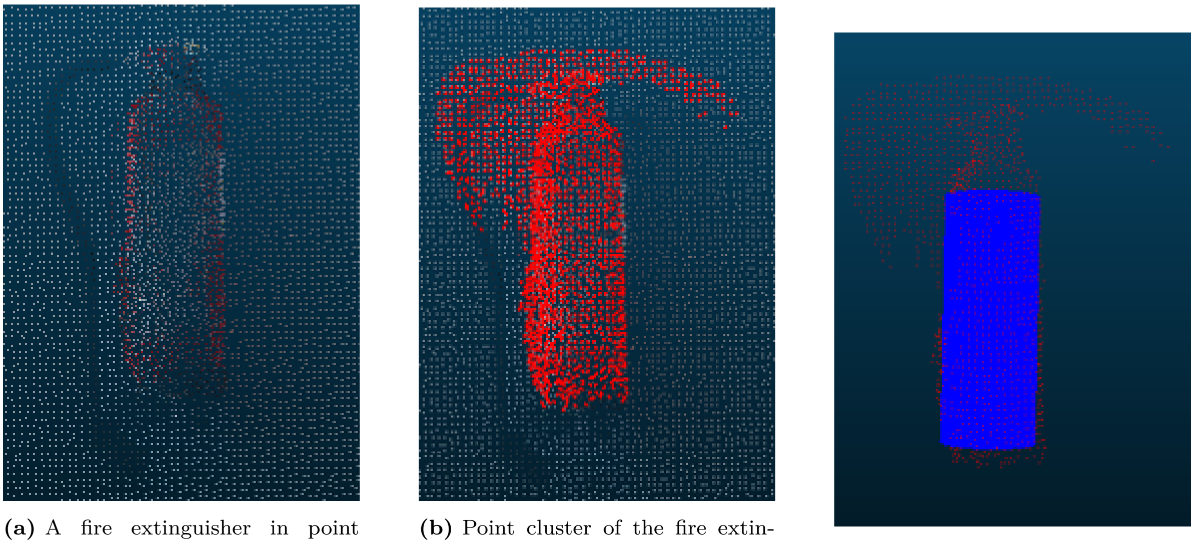
Figure 7.14: Bottom view of part of a ceiling and fitting result



(a) Part of a ceiling in 3D

(b) Fitting result in 3D (red: lighting, yellow: speaker, green: smoke alarm)

Figure 7.15: Part of a ceiling and fitting result in 3D



(a) A fire extinguisher in point cloud

(b) Point cluster of the fire extinguisher

(c) Fit a cylinder to the cluster

Figure 7.16: Part of a ceiling and fitting result in 3D

7.1.8 Text Detection and Recognition

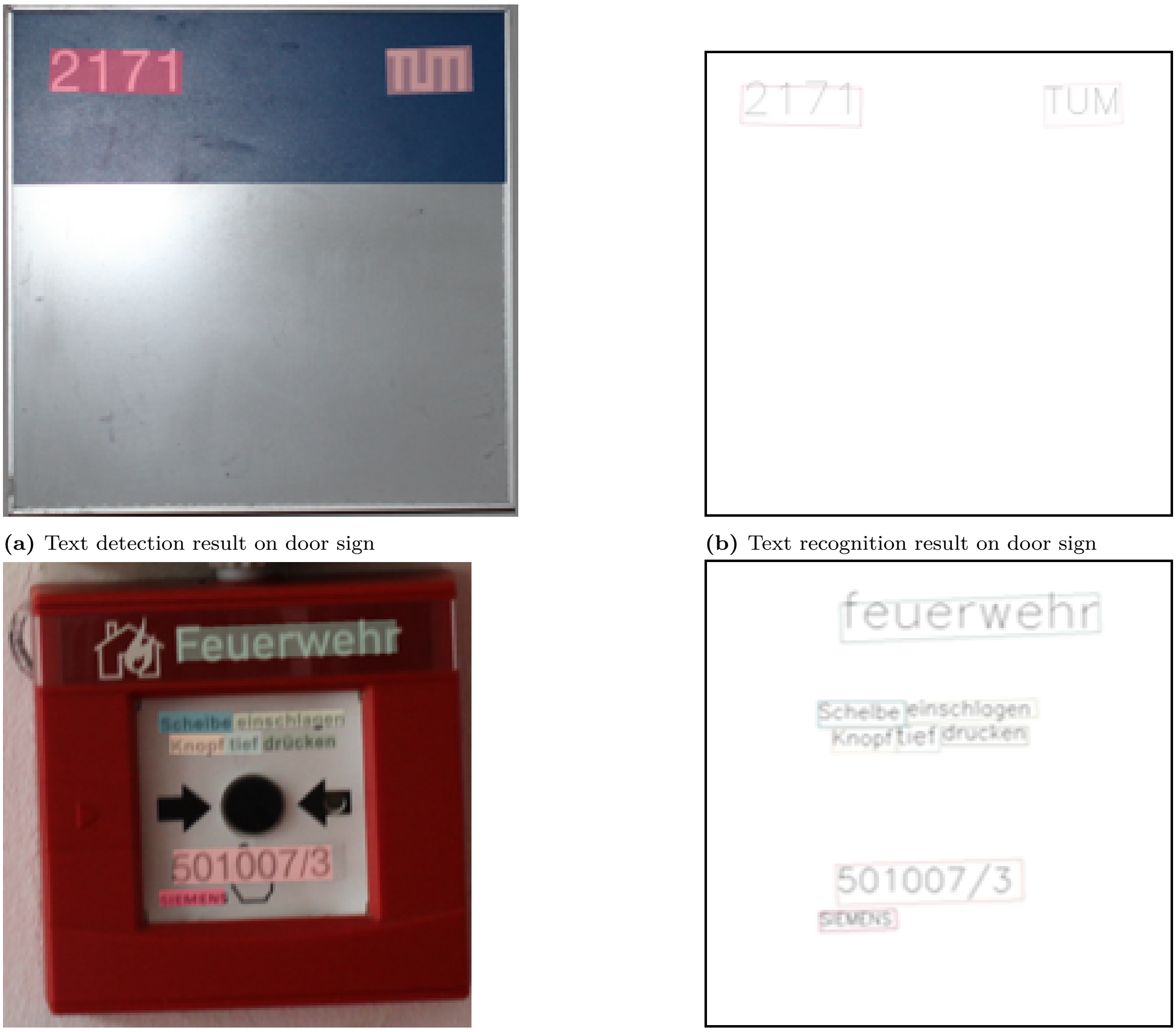
In this step, text information attached to objects is extracted from images. As shown in Figure 7.1, text information for facility management is available on or next to dedicated objects in a building, like the room number on a door sign (shown in Figure 7.1a), the serial number on an emergency switch (shown in Figure 7.1b), the serial number next to a smoke alarm (shown in Figure 7.1c). Apart from detecting and recognising texts, the aim of this step is also to link the detected information to the corresponding objects.

With regard to text detection, text can be located in the object area as well as next to the object (like numbers next to the smoke alarm in Figure 7.1). No valid result could be found for the second case if detecting text only within the object area. In order to solve this problem, the predicted object area is enlarged by increasing its width and length by 50%, assuming related texts to the object are within the enlarged region. The text detection network model with differentiable binarization (Liao et al., 2020), pre-trained on (Gupta et al., 2016), is applied within the enlarged area and outputs the corresponding text bounding boxes.

With regard to text recognition, the text recognition network model for irregular text (Li et al., 2019) is applied to detected text bounding boxes. The recognised text is the information related to the corresponding object that contains or is close to the text area. The text detection and recognition result of a door sign and an emergency switch is illustrated in Figure 7.17. Most texts can be recognised correctly, especially those numbers that are very useful for building management.

In summary, the input to the proposed processing pipeline are images/videos and point clouds. Point clusters with semantic information are created by mapping semantic information detected by deep learning to the 3D point cloud. The 3D mesh model is reconstructed by fitting geometric shapes to point clusters and then enriched by useful information that is valuable for maintaining the building by detecting and recognising text information on or close to objects.

Although the network used in this chapter is designed and trained to work with multi-oriented texts, the recognition result would suffer if texts were not horizontally-oriented. Non-horizontally-oriented texts usually occur in the images of the ceiling because it is hard to make sure the texts in all images are horizontally-oriented when holding a camera to collect images. In order to solve this problem, the author inserts an intermediate step between text detection and text recognition. In this step, the detected text bounding box would be rotated to the position where its longer side is horizontal by assuming texts are oriented along the longer side. Two angles (clockwise and counterclockwise) can rotate the bounding box to the horizontal position and produce two new bounding boxes. One of the angles would flip the text. The two new bounding boxes are then the input for the text recognition step. The flipped texts can be discarded by the lower prediction score, and the results are shown in Section 7.2.4.



(a) Text detection result on door sign

(b) Text recognition result on door sign

(c) Text detection result on emergency switch

(d) Text recognition result on emergency switch

Figure 7.17: Text detection and recognition result

7.2 Experiments and Results

7.2.1 Implementation

The proposed approach is implemented in C++ and Python and is tested in the point cloud collected in the Chair of Computational Modeling and Simulation at the Technical University of Munich (TUM) with the help of NAVVIS (www.navvis.com). The annotated dataset used for transfer learning contains more than 1000 instances, including 120 boards, 124 door signs, 34 elevator buttons, 52 emergency switches, 34 fire extinguishers, 30 escape signs, 357 lights, 94 light switches, 45 pipes, 137 smoke alarms, 123 sockets, and 91 speakers. These images were not taken in the same area of the building where the point clouds were taken.

In point cloud processing, the PCL library (Rusu and Cousins, 2011) is used to implement the proposed algorithm. Object detection in images is done with Detectron2

Technology	Language and library used	automatic or manual
Object detection in image by Transfer learning	Python, Detectron2 (Wu et al., 2019)	automatic
Creating photogrammetric point clouds	Python, COLMAP (Schönberger et al., 2016) (Schönberger and Frahm, 2016)	automatic
Point clouds alignment	None	manual
Extract visible points	C++, PCL library (Rusu and Cousins, 2011)	automatic
Map 2D information to 3D space	C++	automatic
Find best-fitting labels	C++	automatic
Fit shape to point clusters	C++, PCL library (Rusu and Cousins, 2011)	automatic
Text detection and recognition	Python, MMOCR (Kuang et al., 2021)	automatic

Table 7.1: Implementation details of each step

(Wu et al., 2019). In the experiment, the author uses the pre-trained Mask-RCNN model (He et al., 2017) provided by Facebook (Wu et al., 2019) that has been trained on the COCO dataset (more than 100k images) (Lin et al., 2014) and retrained on the annotated dataset. The photogrammetric point cloud is created by using COLMAP (Schönberger et al., 2016) (Schönberger and Frahm, 2016); text detection and recognition are implemented by means of the MMOCR tool (Kuang et al., 2021). The detailed implementation information, including the used technologies and frameworks, is listed in Table 7.1.

In this section, the author presents the results of the experiments from three aspects, point cloud segmentation result, reconstruction result and, text recognition result.

7.2.2 Point Cloud Segmentation Result

In the proposed approach, 2D semantic information detected from images is mapped to a 3D point cloud to identify the respective point clusters. The result is in the same format as that of point cloud segmentation of 3D deep learning. The author compares the segmentation results of the proposed approach with those of 3D deep learning. In this regard, the S3DIS dataset (Armeni et al., 2016) contains the point cloud of the indoor environment that is similar to the point cloud captured on the TUM campus. As

Method	mIoU
PointNet (Qi et al., 2017)	47.6
SPG (Landrieu and Simonovsky, 2018)	62.1
DGCNN (Wang et al., 2019)	56.1
RSNet (Huang et al., 2018)	56.5
PointCNN (Li et al., 2018)	65.4
KPConv (Thomas et al., 2019)	69.6
Point transformer (Zhao et al., 2021)	73.5

Table 7.2: Segmentation mIoUs on S3DIS dataset (evaluated with 6-fold cross-validation)

Model	wall	ceiling	floor	smoke alarm	light
KPConv (3cm)	89.0	96.5	97.6	29.1	69.4
KPConv (5cm)	88.2	96.2	97.8	18.6	65.2

Table 7.3: Segmentation IoUs of related classes in the point cloud

shown in Table 7.2, KPConv (Thomas et al., 2019) is one of the best-performing network architectures with the mean Intersection over Union (mIoU) around 70%. mIoU is a common evaluation metric for semantic segmentation, and a higher value means a better prediction result.

The author chooses KPConv for the experiments with the annotated laser-scanned point clouds captured at TUM and considers these are the reference values for further comparisons. The model is trained with two different downsampling sizes: 3cm and 5cm. It is plain to see that the performance for large objects (wall, ceiling, floor) is much better than that for smaller objects. This result is consistent with that of the S3DIS dataset (Armeni et al., 2016). For a small object like a smoke alarm, in particular, the performance is quite low, which means the current state-of-the-art network is not suitable for segmenting small objects. There are two possible explanations:

- the input point cloud resolution is too low for neural networks to understand small objects;
- small objects have much fewer points compared to larger ones (like a ceiling, floor, and wall), and the unequal class distribution means this has to be compensated during training, which could sacrifice the performance of some classes.

The performance of the proposed approach for different classes is shown in Table 7.4. As we can see, compared with the state-of-the-art network that only uses point clouds

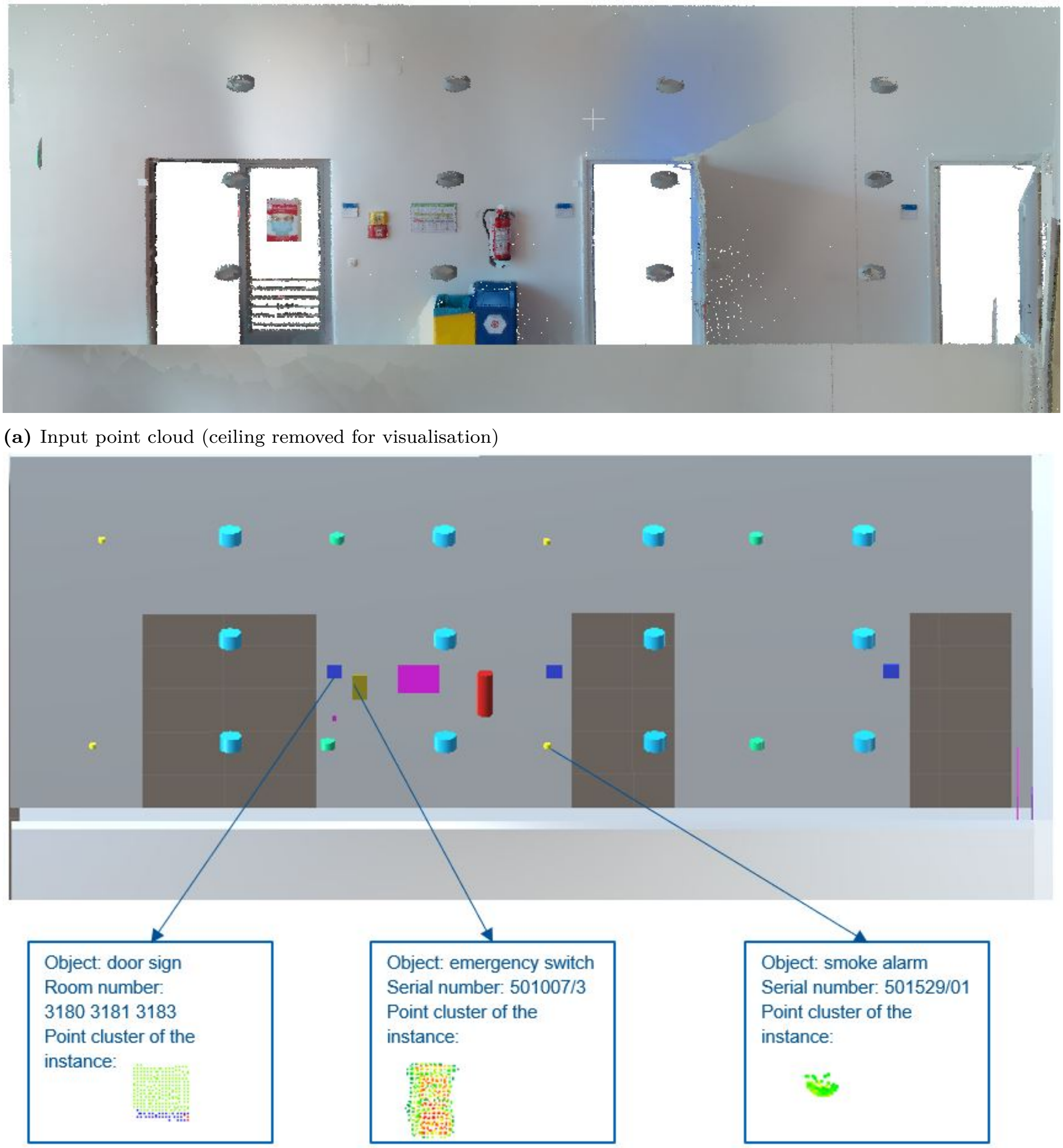
board	door sign	elevator button	emergency switch	fire extinguisher	escape sign	light	light switch	pipes	smoke alarm	socket	speaker
68.0	67.0	80.8	62.2	85.7	70.1	79.9	47.6	39.1	48.6	61.1	64.5

Table 7.4: Segmentation IoUs of small objects in test point cloud

as input, this approach with additional image input provides a significant improvement in the common classes which are available in the image as well as the point cloud (smoke alarm from 29.1% to 48.6%, light from 69.4% to 79.9%).

7.2.3 Reconstruction Result

One example of the information-rich digital twin that is created by applying the proposed approach is illustrated in Figure 7.18. The digital twin is a comprehensive model which includes geometric information (reconstructed 3D geometric models), and semantic information (point clusters of object instances with labels and useful text information).



(a) Input point cloud (ceiling removed for visualisation)

Object: door sign
Room number:
3180 3181 3183
Point cluster of the
instance:

Object: emergency switch
Serial number: 501007/3
Point cluster of the
instance:

Object: smoke alarm
Serial number: 501529/01
Point cluster of the
instance:

(b) The created information-rich building twin

Figure 7.18: Input point cloud and the created elements of the building twin

In Table 7.5, 7.6 and 7.7, the author compares the dimensions for some objects in three categories from one area against the corresponding manually created model from the laser-scanned point cloud. As most of the absolute deviations of the radius are less than $0.01m$, the performance is quite good, given the resolution of the point cloud used is $0.005m$. The relative deviations of smoke alarm diameters are relatively larger than those of the other two classes because the smoke alarms are smaller, which means an absolute deviation in a similar range results in a larger relative deviation value.

No.	radius	ground truth	deviation (abs.)	deviation (rel.%)
1	0.116	0.110	0.006	5.5
2	0.110	0.110	0	0
3	0.118	0.110	0.008	7.3
4	0.110	0.110	0	0
5	0.118	0.110	0.008	7.3
6	0.121	0.110	0.011	10.0
7	0.116	0.110	0.006	5.5
8	0.117	0.110	0.007	6.4
9	0.118	0.110	0.008	7.3
10	0.117	0.110	0.007	6.4
11	0.121	0.110	0.011	10.0
12	0.113	0.110	0.003	2.7

Table 7.5: Light radius comparison between model created from proposed approach and manually created model: (m)

No.	radius	ground truth	deviation (abs.)	deviation (rel.%)
1	0.072	0.070	0.002	2.9
2	0.063	0.070	0.007	10.0
3	0.068	0.070	0.002	2.9
4	0.073	0.070	0.003	4.3

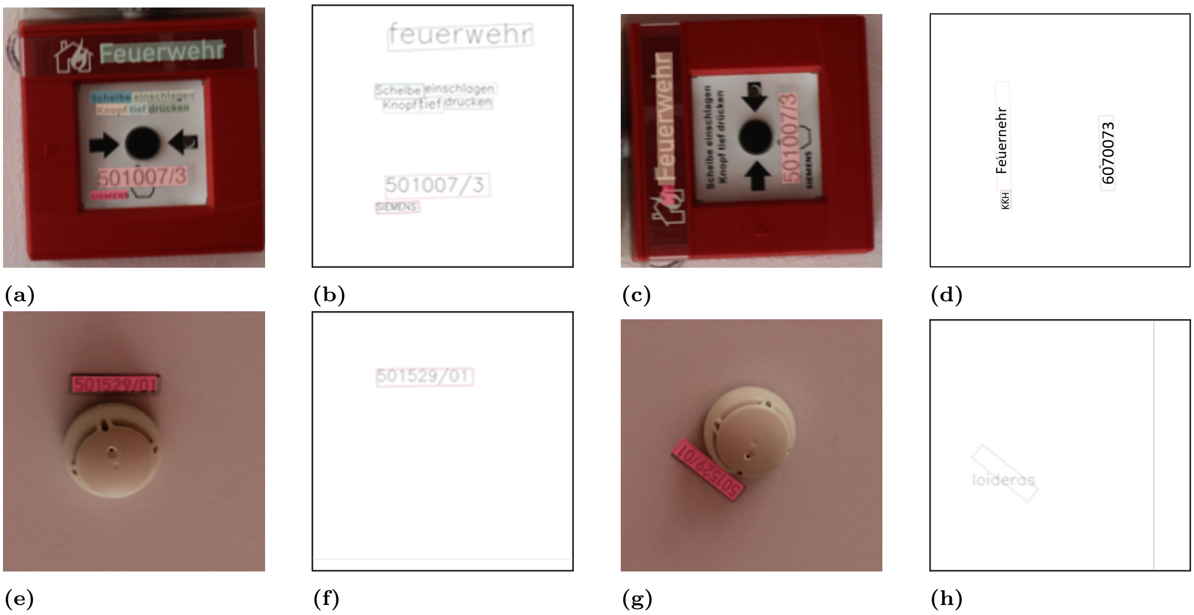
Table 7.6: Speaker radius comparison between model created from the proposed approach and manually created model: (m)

No.	radius	ground truth	deviation (abs.)	deviation (rel.%)
1	0.030	0.035	0.005	14.3
2	0.032	0.035	0.003	8.6
3	0.025	0.035	0.010	28.6
4	0.028	0.035	0.007	20.0
5	0.027	0.035	0.008	22.9

Table 7.7: Smoke alarm radius comparison between model created from proposed approach and manually created model: (m)

7.2.4 Text Recognition Result

In the proposed experiments, the text recognition network model (Li et al., 2019) works well if the text in an image is horizontally oriented and performs worse if the text is not horizontal. The comparison of recognition results for texts attached to two objects is shown in Figure 7.19.



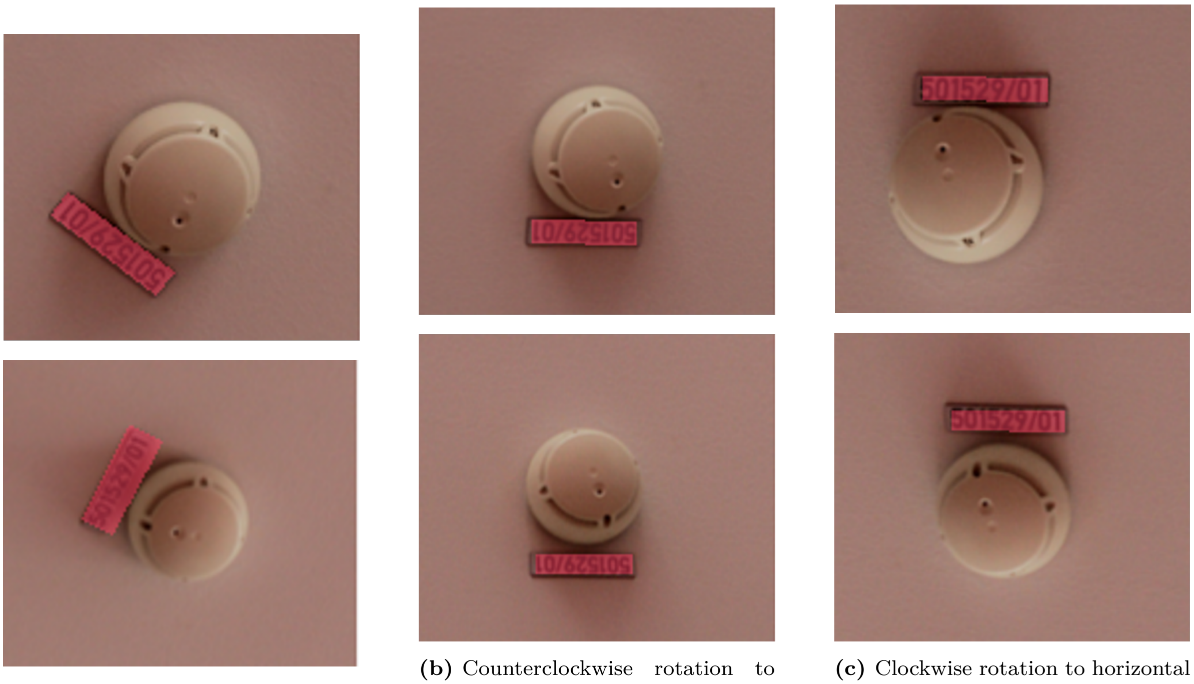
(a) (b) (c) (d)

(e) (f) (g) (h)

Figure 7.19: Comparison of recognition results between non- and horizontally-oriented text

In order to improve the recognition result, the author introduces a method of rotating the detected bounding boxes in Section 7.1.8. The corresponding result is shown in Figure 7.20, for example.

In order to discard the prediction of flipped texts, prediction scores are checked. The recognised texts and corresponding prediction score of four horizontal bounding boxes in Figure 7.20 are listed in Table 7.8. It is plain to see that two prediction scores



(a) Two images of the same object

(b) Counterclockwise rotation to horizontal position

(c) Clockwise rotation to horizontal position

Figure 7.20: Rotating the detected box to horizontal position

Image Nr.	Text	Score
1	501529/01	**0.99995**
2	LO/SEZSLOS	0.78154
3	501529/01	**0.99824**
4	LO/62SLOS	0.84252

Table 7.8: Recognised text and prediction score

(Nr.2 and Nr.4) are significantly lower than the other two (Nr.1 and Nr.3), which means the level of certainty is lower. And this lower prediction score comes from the flipped text. Therefore, it is very easy to identify the correct direction of text by analysing the prediction score. The texts from high score predictions are then chosen as the extracted text information if these predictions provide identical results (as in Table 7.8, where they both predict "501529/01"). If high-score predictions are in conflict with each other, which usually happens when multiple images for the same object are available, all predicted texts are stored with their prediction scores. So the final decision is left up to the human user.

More results of recognising texts in the original and rotated text bounding box are shown in Table 7.9, while the corresponding figures are shown from C.1 to C.3 in Appendix C. It can be seen that in one image (Image 2), the prediction scores are significantly higher than others, and the recognised texts are correct in this image as well.

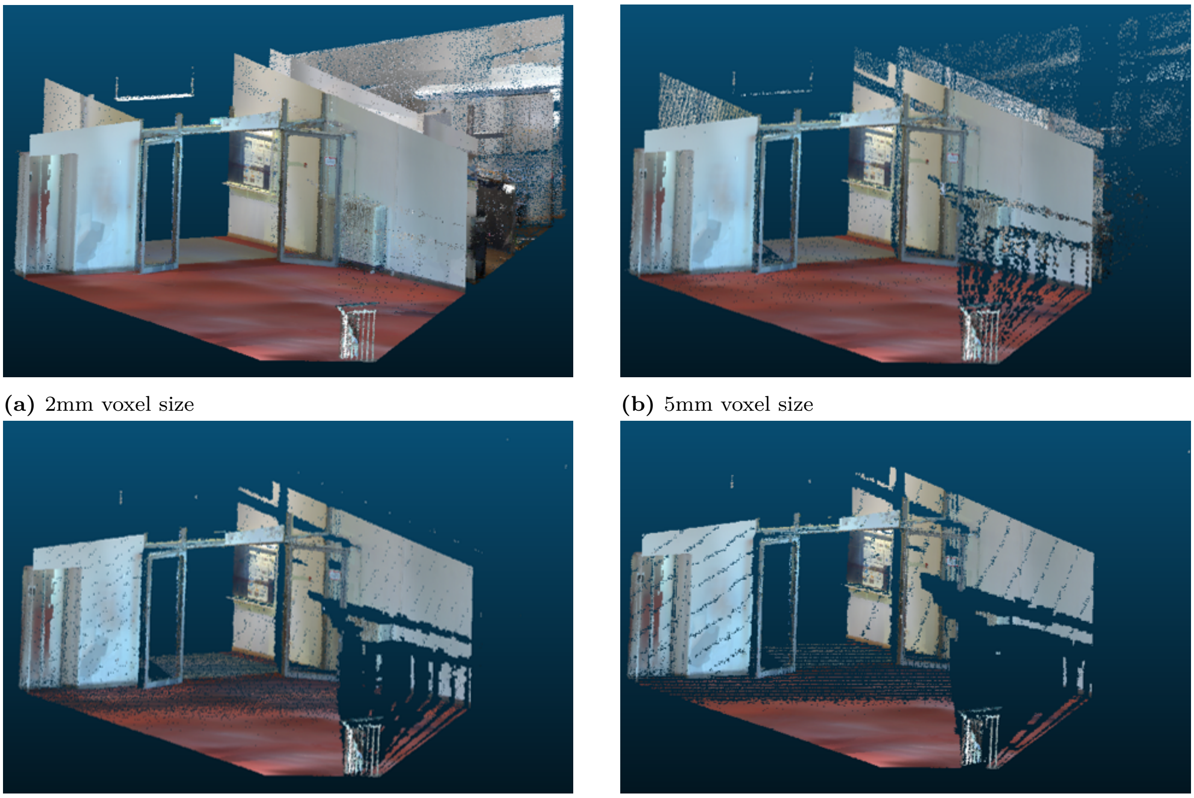
Image Nr.	Text	Score
1	Bunz	0.86509
1	perspy	0.71426
1	BIRTHDAY	0.90593
1	pecumy	0.73446
1	Yorkwood	0.77432
1	6012805	0.76081
2	Heizung	**0.99508**
2	Vorlauf	**0.99984**
2	heizung	**0.99575**
2	Rucklauf	**0.99997**
2	heizung	**0.99994**
2	Vorlauf	**0.99904**
3	Inorgin	0.47901
3	bunzlah	0.80703
3	ineptancy	0.56472
3	heinzi	0.66825
3	Iorasa	0.53956
3	BUNZIOH	0.83753

Table 7.9: Recognised text and prediction score of labels

7.2.5 Parameter Study

In Section 7.1.4, the ray-casting method is used to remove points that should not be visible at the given camera position. The aim of ray-casting is to make points visible in the real world that can also be seen in the point cloud. At the same time, it should not "look through" the wall either, seeing points that should be occluded. Therefore, the voxel size in Figure 7.8 is essential.

Figure 7.21 shows a comparison of four different voxel sizes: 2mm, 5mm, 1cm, and 2cm. As we can see, rays can still go through the wall with a resolution of 2mm and 5mm, which makes the scene behind the wall visible. With a resolution of 2cm, the handrail and its fence cause too much occlusion, making a relatively large part of the wall that should not be occluded invisible. In this case, the voxel size of 1cm provides the best result. Moreover, the test point cloud resolution is also 1cm in Figure 7.21. This is not a coincidence, because a 1cm resolution point cloud means the distance between neighbouring points is around 1cm. Therefore, it is appropriate that the voxel size chosen for ray-casting is the same as the resolution of a point cloud so that rays do not pass through a surface and at the same time avoid unnecessary occlusions.



(a) 2mm voxel size

(b) 5mm voxel size

(c) 1cm voxel size

(d) 2cm voxel size

Figure 7.21: Ray-casting result with different voxel sizes

7.2.6 Discussions

The described approach until now is automatic except for the step to co-register photogrammetric point cloud and laser-scanned point cloud, as shown in Table 7.1. This process can be improved as well in order to get a fully automatic process.

In the proposed method, images taken by the DSLR camera are registered to the laser scanning point cloud through the images taken by the laser scanner, which means an extra data source (images taken by scanners) is required. Many modern laser scanners can take images as well and the laser scanner used in this section is Leica RTC360 (as shown in Figure 2.1a). Using $\mathbf{I_c}$ to denote the DSLR camera image set and $\mathbf{I_l}$ to denote the whole laser scanner image set that is used to reconstruct the photogrammetric point cloud. For an image in camera image set $\mathbf{m}_i \in \mathbf{I_c}$, $\mathbf{M}^i_{ext}$ and $\mathbf{M}^i_{int}$ denote the corresponding camera extrinsic and intrinsic parameter matrices. These parameters are computed by SfM from the previous step and are in the coordinate of the photogrammetric point cloud. For an image in laser scanning image set $\mathbf{n}_i \in \mathbf{I_l}$, $\mathbf{N}^i_{ext}$ and $\mathbf{N}^i_{int}$ denote the corresponding camera extrinsic and intrinsic parameter matrices that are computed by SfM and referenced to the photogrammetric point cloud. Meanwhile, an image in laser scanning image set $\mathbf{n}_i \in \mathbf{I_l}$ also has the extrinsic and intrinsic parameters of the laser scanner camera, referenced to the coordinate of the laser scanning coordinate, denoted by $\mathbf{L}^i_{ext}$ and $\mathbf{L}^i_{int}$. Therefore, the images taken by the laser scanner work as a 'bridge'to connect the photogrammetric and laser scanning point cloud. As shown in Figure 7.22, the marked camera poses are images taken by the laser scanner and the rest are images taken by DSLR images.

For images $\mathbf{n}_i \in \mathbf{I_l}$, camera positions in the photogrammetric and laser scanning point cloud are available. By moving their centroids in the photogrammetric and laser scanning coordinate to the origin, and applying singular value decomposition (SVD) to the matrix of the product of the two position matrices, the translation matrix and rotation matrix can be computed. In this chapter, the author uses $\mathbf{M}$ to denote the transformation matrix that transforms points from laser scanning point cloud coordinates to photogrammetric point cloud coordinates. Any point $\mathbf{p} = \begin{bmatrix} x_0, y_0, z_0 \end{bmatrix}^{\mathbf{T}}$ in the original laser scanning point cloud $\mathbf{S}$ can be transformed to the coordinate of the photogrammetric point cloud by

$$\begin{bmatrix} x_1 \\ y_1 \\ z_1 \\ d_1 \end{bmatrix} = \mathbf{M^{-1}} \begin{bmatrix} x_0 \\ y_0 \\ z_0 \\ 1 \end{bmatrix}, \tag{7.14}$$

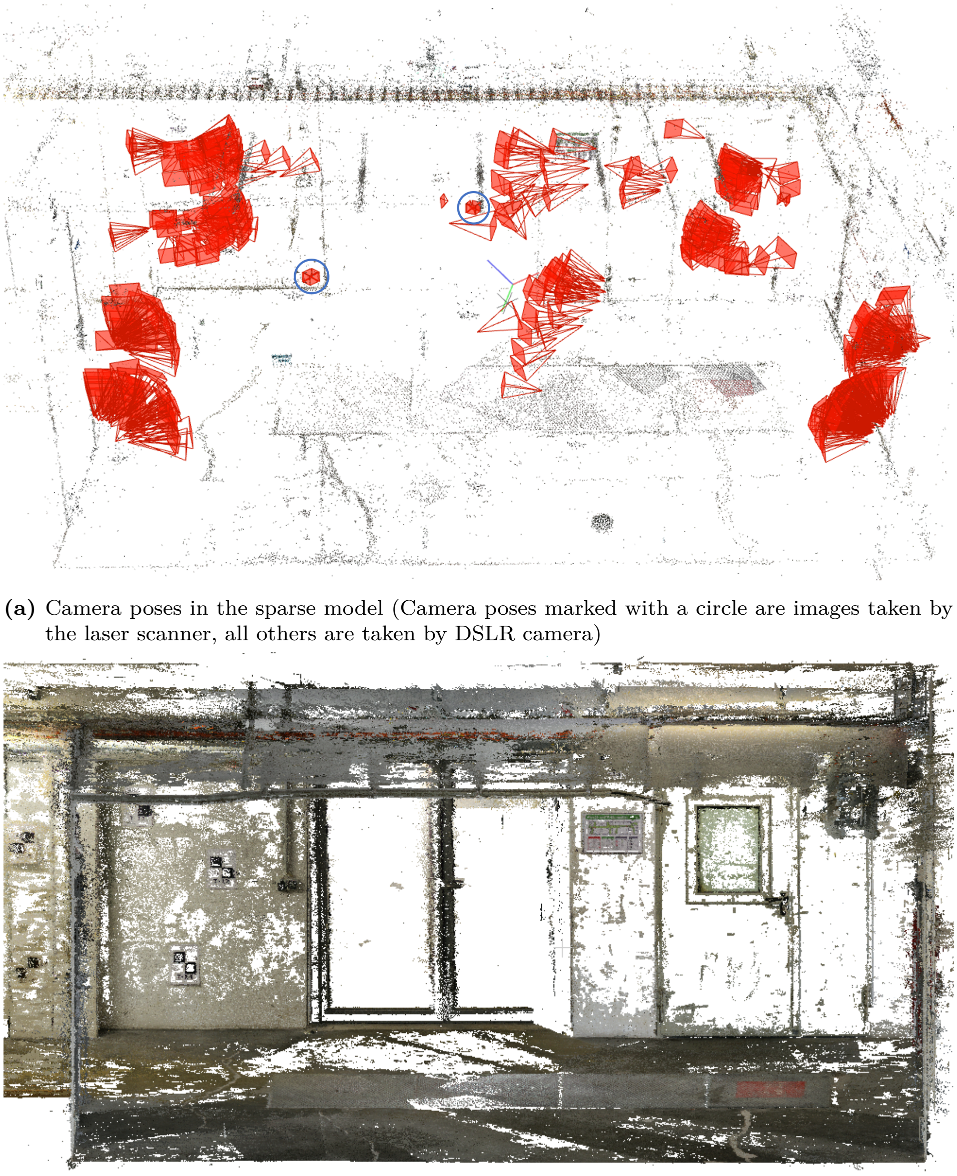
where $\begin{bmatrix} x_0, y_0, z_0, 1 \end{bmatrix}^T$ is the origin homogeneous coordinates of this point $\mathbf{p}$, $\mathbf{M^{-1}}$ is the inverse matrix of $\mathbf{M}$, and $\begin{bmatrix} x_1, y_1, z_1, d_1 \end{bmatrix}^T$ are the newly calculated homogeneous coordinates of the point in the coordinates of the photogrammetric point cloud.

Then normalization is applied by dividing each vector component by d_1,

$$\begin{bmatrix} x_2 \\ y_2 \\ z_2 \\ 1 \end{bmatrix} = \frac{1}{d_1} \begin{bmatrix} x_1 \\ y_1 \\ z_1 \\ d_1 \end{bmatrix}, \tag{7.15}$$

where $\begin{bmatrix} x_2, y_2, z_2, 1 \end{bmatrix}^T$ is the normalized homogeneous coordinate vector of point $\mathbf{p}$ in the coordinate of photogrammetric point cloud.

By using the images taken by laser scanners, the manual co-registration step in the pipeline can be implemented automatically. However, it requires that the laser scanner has the function of taking images during the scanning process.



(a) Camera poses in the sparse model (Camera poses marked with a circle are images taken by the laser scanner, all others are taken by DSLR camera)

(b) Reconstructed dense point cloud (points on the front wall are removed for better visualisation)

Figure 7.22: Overview of computed camera poses and reconstructed point clouds

7.3 Conclusions

As shown in Section 7.2, the proposed pipeline provides convincing results in creating an information-rich digital twin of small objects from laser-scanned point clouds and images. Meanwhile, the method could be applied to other facilities if the environment is captured by a laser scanner and a camera. However, it should be noted that the photogrammetric process only works if a sufficient amount of images were taken differently from different viewpoints. It is hard to say the minimum required number of images for the photogrammetric process because it depends on different aspects, such as the facility size, the number of objects, the camera lens, etc. But according to the author's experience, more images from various viewpoints usually improve the reconstruction result.

In addition, the author also tests the photogrammetric process with images and frames extracted from videos. In the proposed experiment, photogrammetric point clouds created by video frames are usually noisier than those from camera images. Furthermore, a camera with a higher resolution and larger field of view can also contribute to a higher-quality point cloud, which usually requires a longer computation time. As the photogrammetric process is only used to register images to laser-scanned point clouds, the strategies for increasing the quality of photogrammetric point clouds and reducing the cost are not in the scope of this chapter.

If the photogrammetric process in the pipeline fails, all the other parts can still proceed. However, an alternative way to provide a camera's intrinsic and extrinsic parameters should be included, for example, using the referenced images taken by modern laser scanners that have cameras during data capturing, manually recording camera poses and calibrating parameters.

Furthermore, there are still other limitations to the proposed method. Firstly, the object detection step can provide good results for standard objects like fire extinguishers, smoke alarms, etc. But it performs worse with objects that vary greatly in different environments, such as lights on the ceiling. More training pictures are required to solve this problem. Secondly, although the approach has already enlarged the number of reconstructed categories in the indoor environment, many other objects are still missing, such as desks, bookshelves, etc. These elements are also valuable for further enriching the digital twin. As this method can be extended to a fully automatic method, and human intervention is limited to data capture, it provides the possibility to generate and update the model frequently at a low cost. In addition, the method of registering 2D images taken by a camera to the laser-scanned point cloud also can be extended to registering images taken by other sensors, such as thermal cameras. In this case, the digital twin can be further enriched with other information.

Chapter 8

Implementation Details and Experimental Setup

The implementation details and experimental setup of the overall proposed solution are introduced in this chapter. The following parts of the chapter are organised as follows. The implementation details are presented in Section 8.1. The input data are described in Section 8.2. The designed data structure for the overall output is presented in Section 8.3.

8.1 Implementation Details

In this section, the implementation details of the overall proposed solution are presented. The proposed approach is implemented in C++ and Python. With regard to point cloud processing, the PCL library (Rusu and Cousins, 2011) and CGAL library (The CGAL Project, 2020) are used to implement the proposed methods. Semantic segmentation on the point cloud is implemented in Python, using the network architecture KPConv (Thomas et al., 2019) and PointTransformer(Zhao et al., 2021). Object detection in images by 2D networks is achieved with the Framework Detectron2 (Wu et al., 2019). The photogrammetric point cloud is created by using COLMAP (Schönberger et al., 2016) (Schönberger and Frahm, 2016). Text detection and recognition are implemented by means of the MMOCR (Kuang et al., 2021). The detailed implementation information, including the sub-processes and used frameworks, is listed in Table 8.1.

Subprocess	Section in the thesis	Language and library used
Segment point clouds of multi-storey buildings into different storeys	4	C++, PCL library (Rusu and Cousins, 2011)
Point Cloud Segmentation by Deep Learning	5.1.1	Python, KPConv (Thomas et al., 2019), PointTransformer (Zhao et al., 2021)
Reconstruct space-bounding elements for buildings that fulfil Manhattan-world assumption	5.1.2 to 5.1.8	C++, PCL library
Reconstruct space-bounding elements for buildings that do not fulfil Manhattan-world assumption	6	C++, CGAL library (The CGAL Project, 2020)
Object detection in image by Transfer learning	7.1.1	Python, Detectron2 (Wu et al., 2019)
Creating photogrammetric point clouds	7.1.2	Python, COLMAP (Schönberger et al., 2016; Schönberger and Frahm, 2016)
Map 2D information to 3D space	7.1.3 to 7.1.6	C++
Fit shape to point clusters	7.1.7	C++, PCL library
Text detection and recognition	7.1.8	Python, MMOCR (Kuang et al., 2021)

Table 8.1: Implementation details of all steps

8.2 Data Description

In this section, all data used in the proposed solution are introduced. As presented in the previous chapters in which two data sources are used: point clouds and images, this section is separated into two parts as well.

8.2.1 Point Cloud Data

Point cloud data used in this thesis can be divided into two categories: point clouds to train neural networks and point clouds of the environments to be reconstructed. The author uses publicly available data to train neural networks and prepares self-annotated data as well. S3DIS (Armeni et al., 2016) dataset that includes in total six areas of the office building is included in the training set. The self-annotated dataset, named TUMCMS set, consists of around 30 rooms and contains around 25 different classes, including ceiling, door, elevator, exit signs, floor, light, smoke detector, wall, window, radiator, beam, handrail, bookshelf, closet, desk, table, couch, chair, fire extinguisher, dustbin, fan, monitor, clutter. It needs to be noted that not all categories labelled in the dataset are reconstructed in the proposed solution. Object categories in the scope of the thesis are introduced in Section 3.1. The point cloud of the indoor environment to be reconstructed contains three storeys and is collected at Technical University as well. The detailed comparison is listed in Table 8.2.

8.2.2 Image Data

Similar to point cloud data, the image data used in the implementation also can be divided into two categories: images used to train the neural networks and images taken in the environments to be reconstructed. The images used for transfer learning are taken in a different building at the TUM city-centre campus and annotated manually. The annotated dataset used for transfer learning contains more than 1000 instances, including 120 boards, 124 door signs, 34 elevator buttons, 52 emergency switches, 34 fire extinguishers, 30 escape signs, 357 lights, 94 light switches, 45 pipes, 137 smoke

Dataset	Environment specification	Publicly available?	Annotated?
S3DIS (Armeni et al., 2016)	six areas of office building	Yes	Yes
TUMCMS	around 30 rooms of office building	No	Yes
Multi-storey building data at TUM	three storeys of office building	No	No

Table 8.2: Point cloud data used in the implementation

alarms, 123 sockets, and 91 speakers. In the experiment, the author uses the pretrained Mask-RCNN model (He et al., 2017) provided by Detectron2 (Wu et al., 2019) that has been trained on the COCO dataset (more than 100k images) (Lin et al., 2014) and retrained on the annotated dataset. These images were not taken in the same area of the building where the point clouds were taken.

The images to reconstruct the elements in the environment are collected as images or extracted from videos. For the process of reconstructing photogrammetric point cloud 7.1.2, around 300 images or frames from videos are used for one space, and 10 spaces are captured in total.

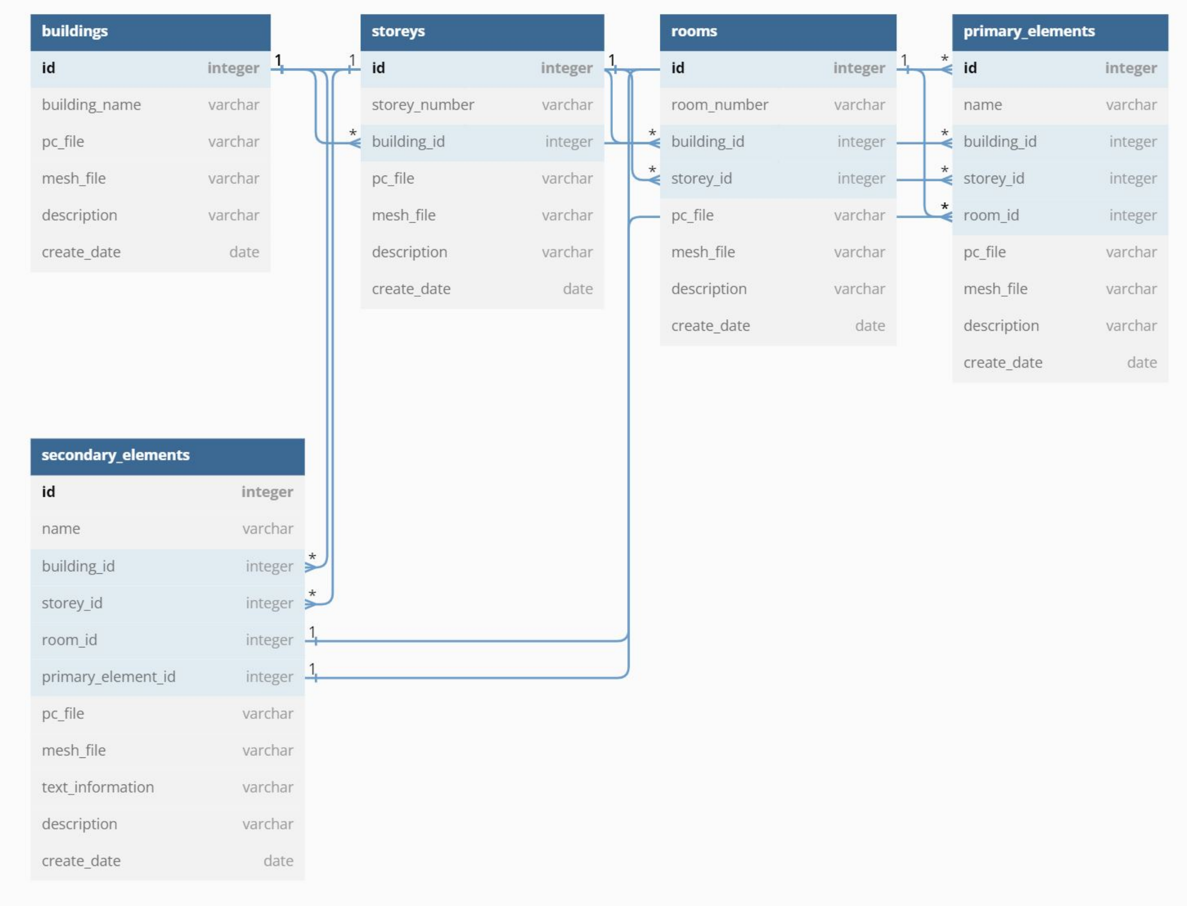
8.3 Data Structure Design for Output

The proposed solution creates an information-rich digital twin that includes both space-bounding elements and small-scale elements, detailed and simplified geometric information, and semantic information of object classes and texts. In order to store the extracted information and the corresponding relationships, the author presents a data structure for the output of the overall proposed approach. The data structure is illustrated in Figure 8.1.

As the whole method follows a "top-down" approach that starts from a multi-storey building, through individual storeys and rooms, to individual elements, the designed data structure follows the same logic. Basically, the whole building consists of multiple storeys, and each storey contains a number of rooms. Subsequently, a room contains various elements. The primary elements here means the space-bounding elements (like walls, ceilings, floors) that are extracted in Chapter and Chapter 5 and Chapter 6, and the secondary elements refer to the elements that are extracted in Chapter 7 (like smoke alarms, light switches, etc.), which are associate with primary elements (for example, a smoke alarm is mounted on the ceiling, and a light switch is usually mounted on the wall).

As shown in Figure 8.1, storeys are linked to buildings through "*building_id*", rooms are linked to buildings and storeys with "*building_id*" and "*storey_id*". In the element level, primary elements are linked to buildings with "*building_id*", storeys with "*storey_id*", and rooms with "*room_id*". Secondary elements are linked to buildings with "*building_id*", storeys with "*storey_id*", rooms with "*room_id*", and primary elements with "*primary_element_id*" in the same manner.

Results of segmenting point clouds of multi-storey buildings into individual storeys (in Chapter 4) are stored in the corresponding storeys under "*pc_file*", which contains the path of the segmented point clouds. Results of segmenting point clouds of one storey into individual rooms and then into space-bounding elements (in Chapter 5 and 6) are stored in the corresponding storeys under "*pc_file*", which contains the path of the segmented point clouds. The class information of elements is stored under "*name*", and the corresponding simplified mesh file path is stored under "*mesh_file*". Results of the method that combines object detection in images and point clouds (in Chapter 7) are



buildings	
id	integer
building_name	varchar
pc_file	varchar
mesh_file	varchar
description	varchar
create_date	date

storeys	
id	integer
storey_number	varchar
building_id	integer
pc_file	varchar
mesh_file	varchar
description	varchar
create_date	date

rooms	
id	integer
room_number	varchar
building_id	integer
storey_id	integer
pc_file	varchar
mesh_file	varchar
description	varchar
create_date	date

primary_elements	
id	integer
name	varchar
building_id	integer
storey_id	integer
room_id	integer
pc_file	varchar
mesh_file	varchar
description	varchar
create_date	date

secondary_elements	
id	integer
name	varchar
building_id	integer
storey_id	integer
room_id	integer
primary_element_id	integer
pc_file	varchar
mesh_file	varchar
text_information	varchar
description	varchar
create_date	date

Figure 8.1: Designed data structure

stored in "*pc_file*" and "*mesh_file*" under "*secondary_elements*". The extracted text information like object IDs or serial numbers goes to the "*text_information*".

Currently, the designed structure can fulfil the requirements of the created information-rich digital twin in the proposed solution. With regard to geometric information, the detailed geometry is stored under "*pc_file*", and the simplified geometry is stored under "*mesh_file*". With regard to semantic information, the extracted semantic information of the object class is stored in "*name*", and the extracted semantic information of texts is stored in "*text*". Objects are linked to buildings, storeys, rooms, and other objects to represent relationships.

The data structure can also be extended to fulfil more use cases of digital twins. For example, if other object categories in the buildings are required to enrich the created digital twin further (like furniture), other classes can be added to the database (like furniture). The corresponding objects of the classes can be linked to other objects using the same logic. Furthermore, if more data from other sources should be included (like thermal data), the thermal data and extracted information like temperature can be considered as properties and then added to the corresponding objects.

In summary, the designed data structure is capable of containing all the information extracted in the whole approach, which can store the extracted point clusters, simplified meshes, category labels, and text information. In addition, it still provides the possibility to add more properties to each element to fit other digital twin use cases.

Chapter 9

Results and Conclusions

In this chapter, the author presents the results and conclusions of the whole thesis. More specifically, this chapter is organised as follows. The results and outcome analysis of the overall proposed solution are presented in Section 9.1. The contributions and limitations are discussed in Section 9.2. Last, the conclusions of the thesis are presented in Section 9.1.

9.1 Results

The overall proposed solution is designed to create an information-rich digital twin that includes both space-bounding elements and small-scale elements, detailed and simplified geometric information, and semantic information (object classes and recognised texts). The data flow that shows how sub-processes presented from Chapter 4 to 7 are combined to get the final output of the information-rich digital twin is presented in Figure 9.1. In this Figure, the green arrows represent the data flow between the sub-processes. The red arrows represent how the point clusters of different objects representing the detailed geometric information get to the final output. The point clusters with labels extracted from two input sources (point clouds and images) are selected and combined to show the detailed geometry of the environments. Subsequently, the purple arrows represent how simplified mesh models that are created from corresponding point clusters are gathered in the final output. The mesh models of small-scale objects are added to the basic model of space-bounding elements. The orange arrow shows the extracted text information extracted from images is included and associated with the corresponding objects as well.

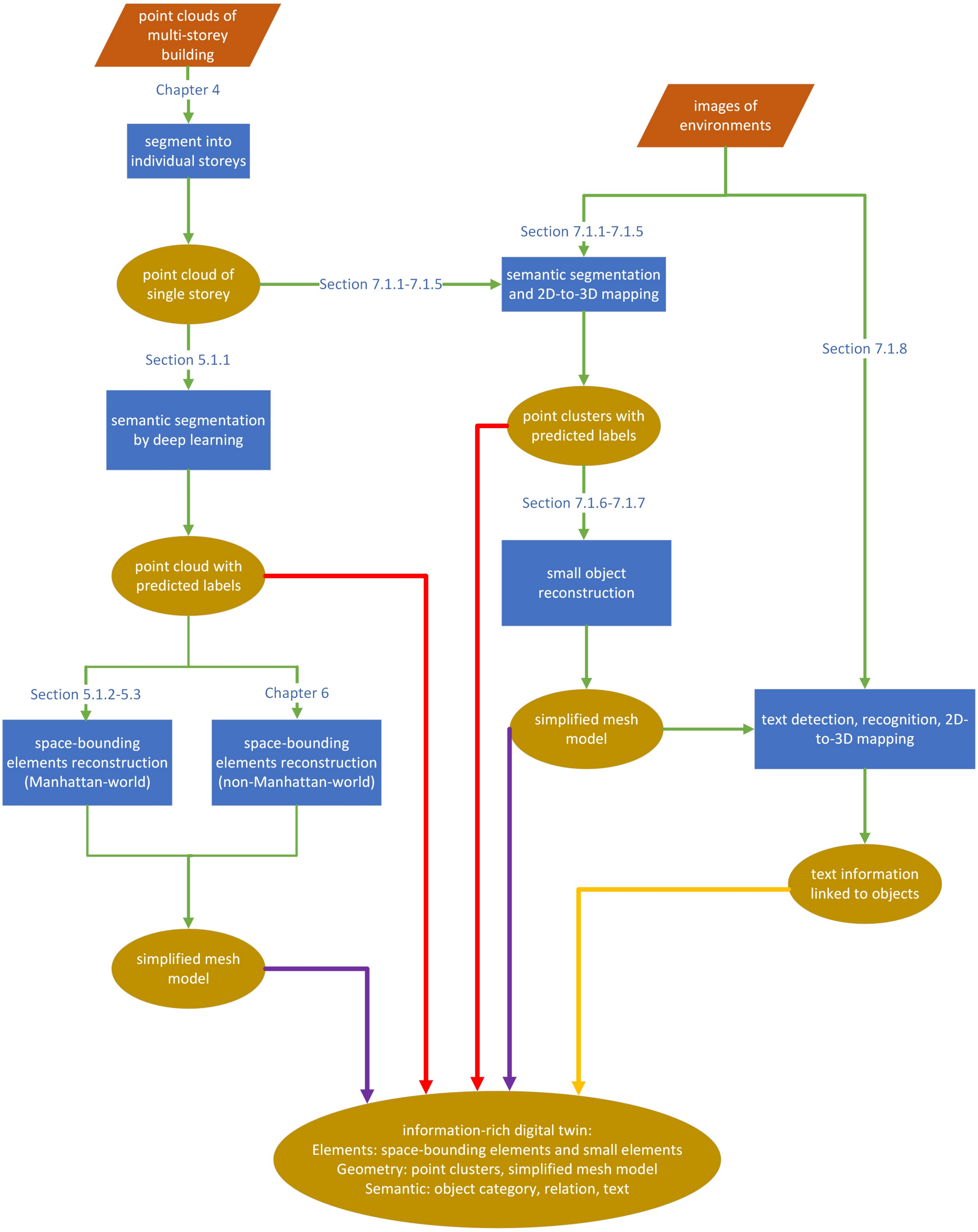
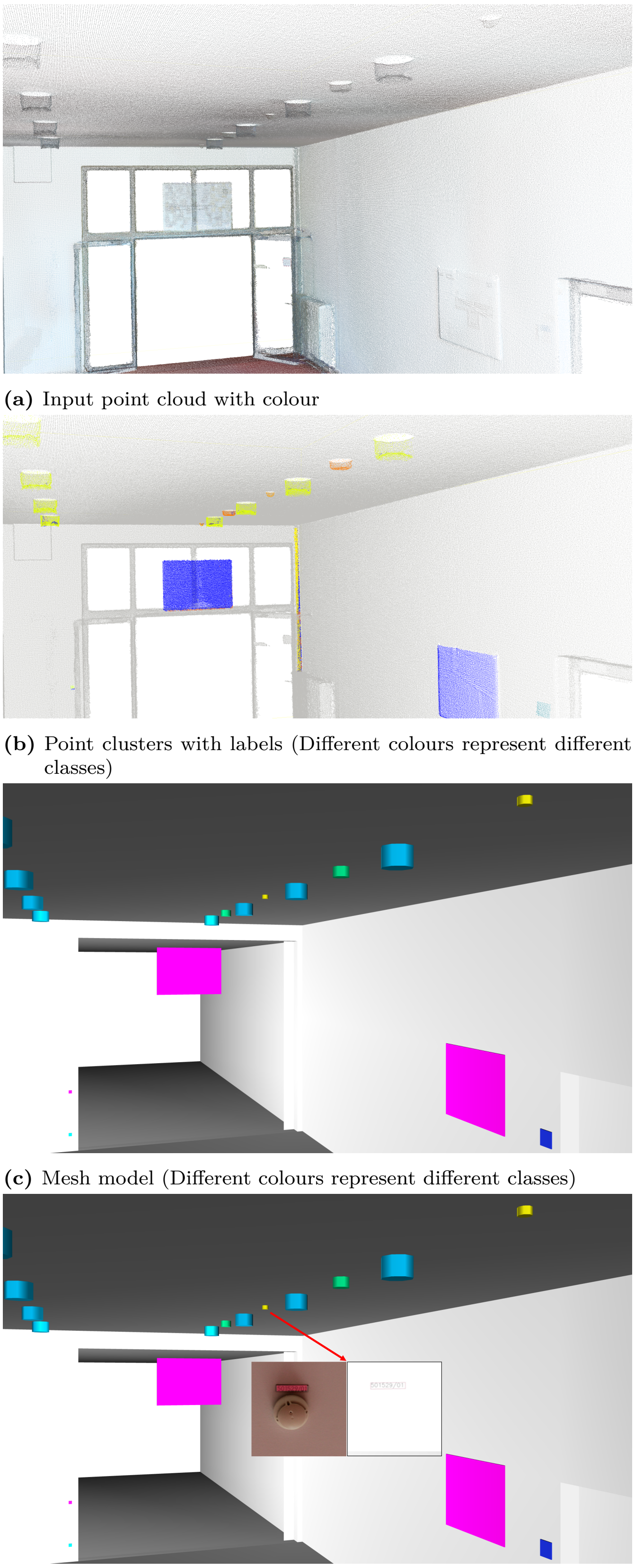

point clouds of multi-storey building
Chapter 4
segment into individual storeys
images of environments
Section 7.1.1-7.1.5
point cloud of single storey
Section 7.1.1-7.1.5
semantic segmentation and 2D-to-3D mapping
Section 7.1.8
Section 5.1.1
semantic segmentation by deep learning
point clusters with predicted labels
Section 7.1.6-7.1.7
point cloud with predicted labels
small object reconstruction
Section 5.1.2-5.3
Chapter 6
space-bounding elements reconstruction (Manhattan-world)
space-bounding elements reconstruction (non-Manhattan-world)
simplified mesh model
text detection, recognition, 2D-to-3D mapping
text information linked to objects
simplified mesh model
information-rich digital twin:
Elements: space-bounding elements and small elements
Geometry: point clusters, simplified mesh model
Semantic: object category, relation, text

Figure 9.1: Outline of the proposed solution with marked data flow (Green: data flow in sub-processes; Red: data flow for detailed geometry; Purple: data flow for simplified geometry; Orange: data flow for text information.)

Examples of the final output that gathers all information are shown in Figure 9.2 and Figure 9.3. The input point clouds are shown in Figure 9.2a and 9.3a. Given the input point clouds and images taken in the same area, the detailed geometric information of detected objects shown as point clusters of objects after running through the overall proposed solution, which is shown in Figure 9.2b and 9.3b. These two figures are colour-coded, which means different colours represent different categories of objects. The mesh models showing the simplified geometric information are then extracted based on the point clusters with labels. These mesh models are presented in Figure 9.2c and 9.3a, which are colour-coded as well, where different colours represent different object classes. At last, useful text information is also extracted as additional semantic information to enrich the created model further. Figure 9.2d shows the extracted object ID next to a smoke alarm can be extracted, and Figure 9.3d presents the recognised texts on a door sign. These texts are linked to the corresponding objects and included in the final output of an information-rich digital twin.

In summary, the overall proposed solution selects and combines the extracted information in the sub-processes (from Chapter 4 to Chapter 6) of the overall presented approach. The final created information-rich digital twin is then stored in the proposed data structure (in Section 8.3).



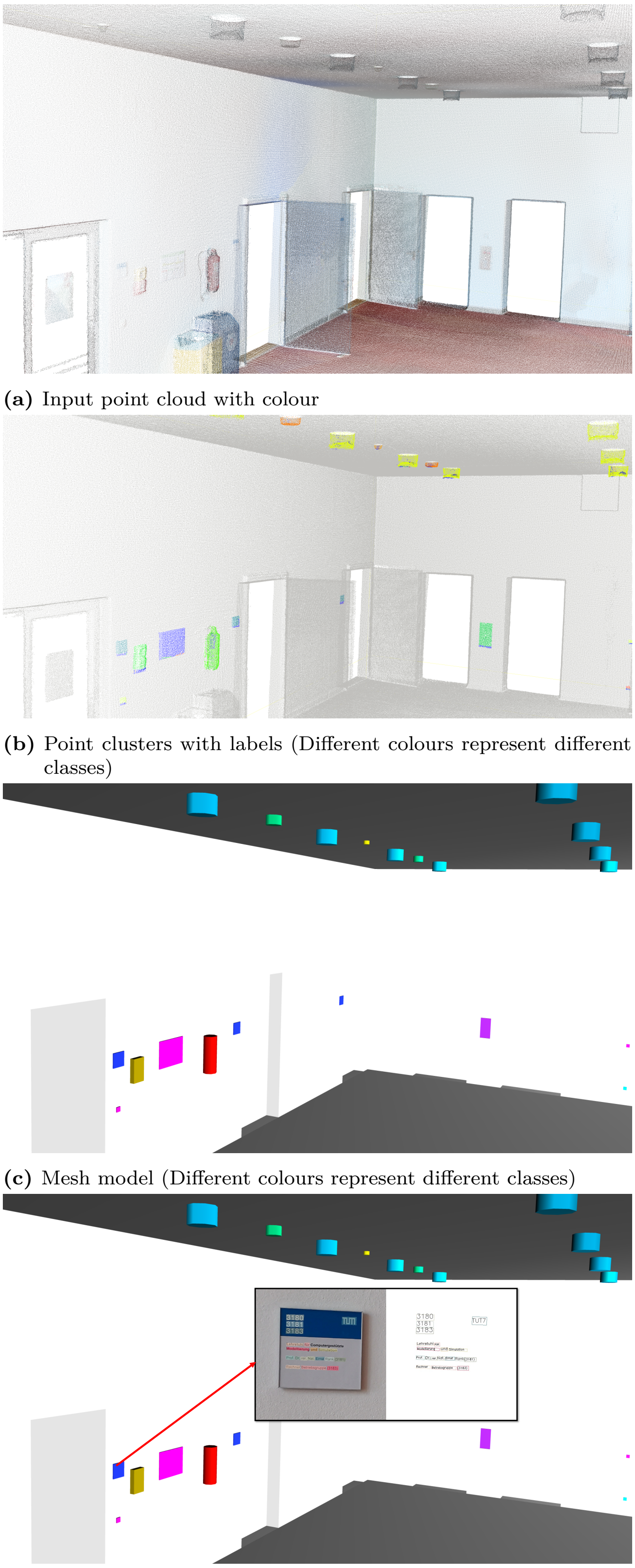
(a) Input point cloud with colour

(b) Point clusters with labels (Different colours represent different classes)

(c) Mesh model (Different colours represent different classes)

(d) Recognised object ID linked with a smoke alarm

Figure 9.2: Information contained in the final output


3180
3181
3183
Lehrstuhl für Computergestützte Modellierung und Simulation
Prof. Dr. rer. Nat. Ernst Rank (3181)
Rechner Betriebsgruppe (3183)
3180
3181
3183
TUT7
Lehrstuhl für
Modellierung und Simulation
Prof. Dr. rer. Nat. Ernst Rank (3181)
Rechner Betriebsgruppe (3183)

(a) Input point cloud with colour

(b) Point clusters with labels (Different colours represent different classes)

(c) Mesh model (Different colours represent different classes)

(d) Recognised text on a door sign

Figure 9.3: Information contained in the final output

9.2 Contributions and Limitations

As a valuable digital asset of the built environment, digital twins provide possibilities to help all stakeholders of the facilities with a variety of use cases, including facilities management and operation, asset condition monitoring, sustainable development, construction progress monitoring, etc. Especially in the process of supporting decision-making, digital twins benefit the whole process by providing more reliable and useful information. In this thesis, a method to create an information-rich digital twin of buildings from two different data sources (point clouds and photos), which provides complementary information that could be achieved different reconstruction objectives, is proposed.

The information-rich digital twins mainly contain the following constituents:

- the simplified geometric mesh model of different objects;
- point clusters with label information of corresponding objects;
- text information, including object IDs linked to objects.

Object categories include relatively large space-bounding elements and relatively small objects.

The proposed overall method mainly includes the following steps:

- given a multi-storey point cloud of indoor environments as input, the point cloud is segmented into individual storeys of point clouds;
- space-bounding elements are extracted and reconstructed directly from the laser-scanned point cloud using geometric information and semantic information extracted by point cloud deep learning; depending on whether the point clouds fulfil the Manhattan-world requirements or not, two different approaches are proposed;
- by object detection in 2D images and mapping from 2D image to 3D space, small objects are extracted and reconstructed to enrich the digital twin of buildings; text detection and recognition are performed to extract object IDs and other useful text information to enrich the model further.

The key contributions of this thesis are summarised as follows:

- approaches for extracting space-bounding elements from point clouds of the environments that do or do not fulfil the Manhattan-world assumption are presented;
- not only space-bounding elements but also smaller objects are reconstructed, which increases the detected object categories of indoor environments;

- deep learning is applied and merged in the approaches, and the extracted semantic information is selected and then used to reconstruct the objects;
- apart from laser scanned point clouds, photos are used to extract semantic information in 2D and then mapped to 3D to reconstruct smaller objects;
- text information is extracted by text detection and recognition technologies and linked to the corresponding objects.

There are still limitations to the proposed solution:

- as deep learning on 3D point clouds is applied in the proposed solution, which makes the performance of the proposed solution depends on deep learning. In order to train a well-performed model, a large amount of annotated data and computing power are required.
- more specifically, the proposed solution does not work for large curtain walls made of glasses. These surfaces made of glasses are extremely hard to be captured in data collection and difficult to be recognised in data processing as well.
- the proposed solution considers the objects in the scope shown in Section 3.1. The existing object categories in a building in practice are much larger, which means the created digital twin can still be continuously enriched with other small objects.

9.3 Conclusions and Future Work

The proposed approach uses point cloud data and images as input to create an information-rich digital twin of indoor environments. The final output of the approach contains geometric and semantic information for space-bounding elements and small elements. The created digital twin can be used in applied in different aspects throughout various assets' life cycles. For historical assets which have been completed for many years but do not yet have any digital records, the created digital twin can help to start and keep a recording of their current conditions for making better decisions, especially in facility maintenance and renovation. For those assets with available digital representations, the proposed approach provides the possibility to enrich the current digital model with more useful information as well. Keeping the digital twin dynamic and up-to-date can improve the asset's real-time progress monitoring, quality control, diagnostics, and prognostics for facilities. Furthermore, the created final output can also be used in capital investment projects before the design and construction of the facility. It is efficient and convenient to simulate the performance with its digital twin to assist decision and strategy-making in various predictive scenarios.

In addition, the proposed pipeline can be used not only to create digital twins from scratch but also to update the available digital twin. And it is also not necessary to start the approach always from the beginning. For example, if prior knowledge (like the space-bounding elements in the building do not change) is known in advance, it is possible to collect image data and run the sub-processes designed for small objects only. In this case, laser scanners are not even required. Instead, a much lighter photo-capturing device like a camera or even a mobile phone can be used to collect data, which could further reduce the cost of generating a model. This process is illustrated in Figure 9.4.

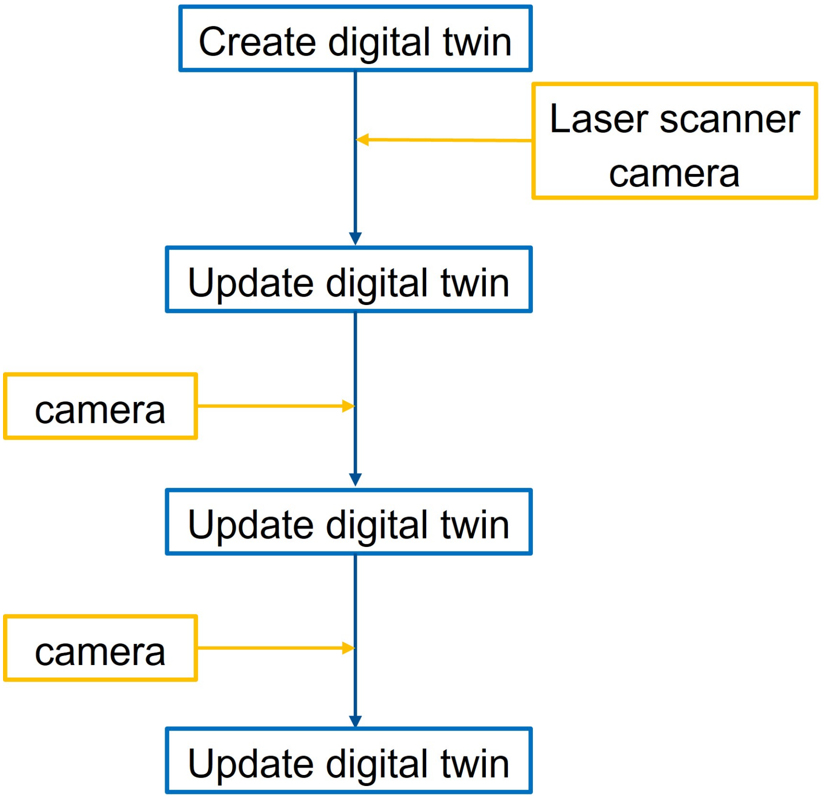


Figure 9.4: Updating digital twin can only use photos

In practice, the proposed approach can heavily reduce the human effort in the process of reconstructing indoor environments from point clouds and photos. The human modellers do not need to measure the locations and dimensions of elements in the environments. Instead, they could just check the automatically created models and fix the corresponding parts if errors or inconsistencies are found. The cost to create a digital twin for indoor environments could be heavily reduced, which also provides the possibility to digitise more facilities. These digital twins can be used in various use cases and bring benefits to all stakeholders of the facility.

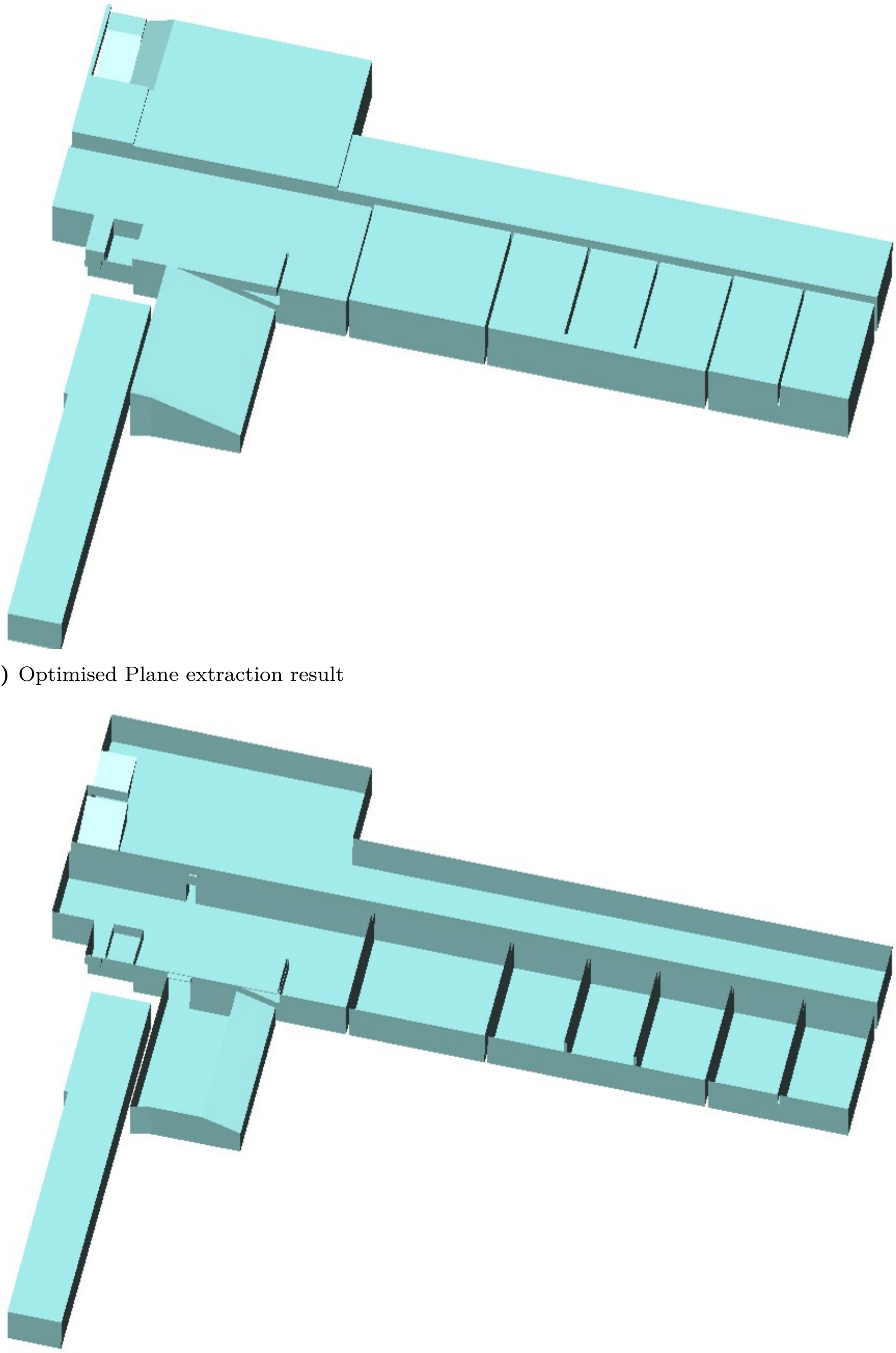
For the whole society, when more and more digital twins of various facilities are available, the current geometric conditions of these assets can be visually monitored. Furthermore, by putting different kinds of sensors into the physical assets and updating the corresponding data in digital twins, the data between digital and physical twins can be linked effectively. This could be a starting point to make smart buildings or smart cities. Decision-makers can effectively collect information for buildings and cities in the digital twin.

In conclusion, this thesis presents a pipeline to create an information-rich digital twin from point clouds and photos, which reduces the human effort in the process of creating digital twins. The approach starts with segmenting multi-storey point clouds into individual storeys, through space-bounding elements reconstruction, and ends with small objects and text information detection. By combining information extracted by 2D deep learning in images and 3D deep learning in point clouds, the complementary information contained in photos and point clouds is used to create the final output model, which contains simplified and detailed geometric information, object label information, and useful text information. The proposed methods in the sub-processes of the proposed approach can also be applied separately to different individual use cases. And the proposed method of registering 2D images with 3D point clouds can also be extended to register images from other sensors (like thermal images) to add more information to the created digital twin.

In the future, some potential research topics in the following aspects could be conducted. Firstly, more component categories could be considered, such as staircases, handrails, furniture, etc. The digital twin can be further enriched by including more object classes. Secondly, components with more complicated structures can be considered, for example, curved walls and ceilings. At last, data from other sensors can also be included, such as thermal sensors, which can enrich the information in digital twins. The multi-modal data in digital twins can provide more potential use cases.

Appendix A

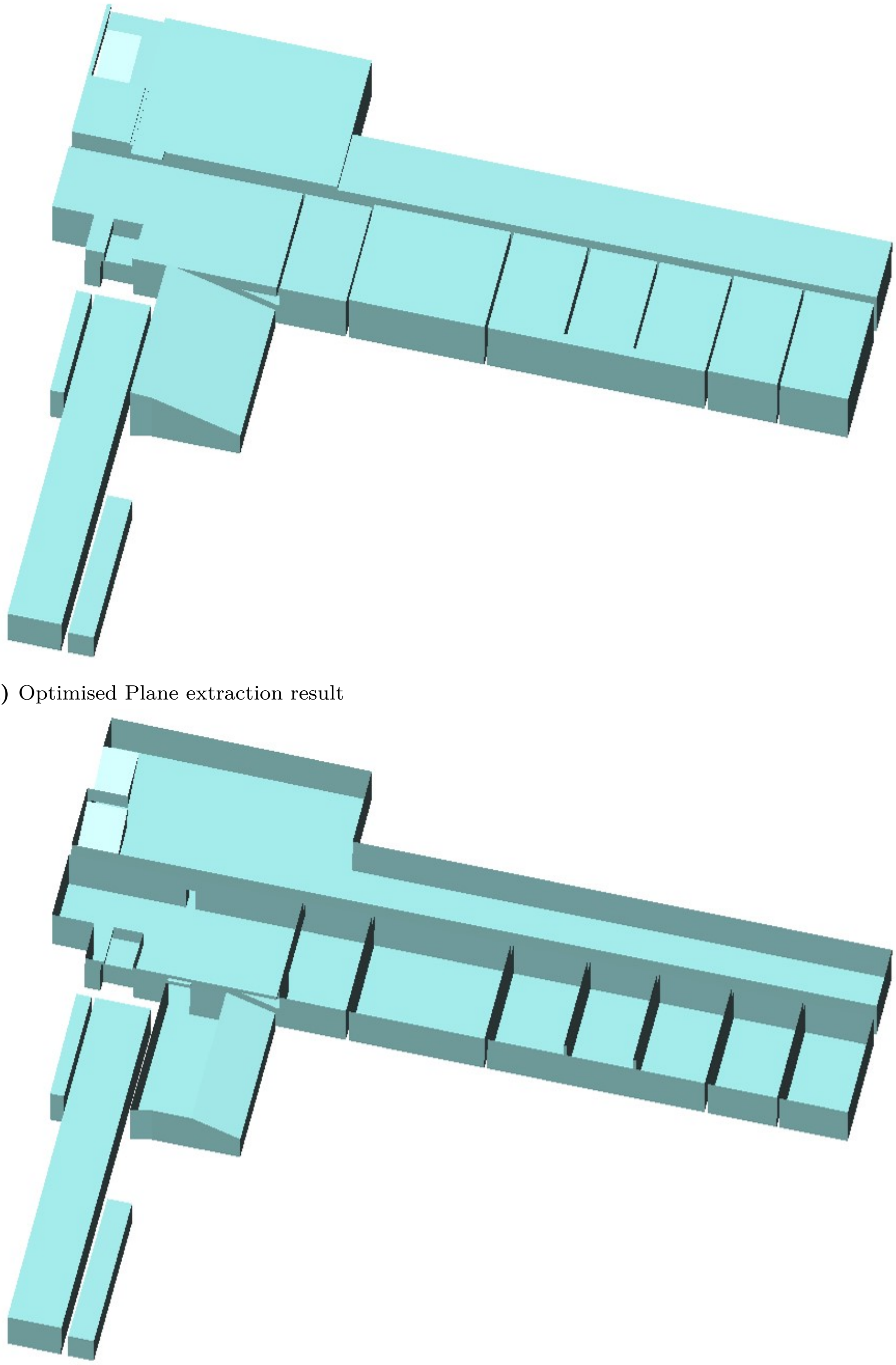
Parameter study on coefficients of energy terms in Section 6.2.1



(a) Optimised Plane extraction result

(b) Optimised Plane extraction result(part of the ceiling is removed for better visualisation)

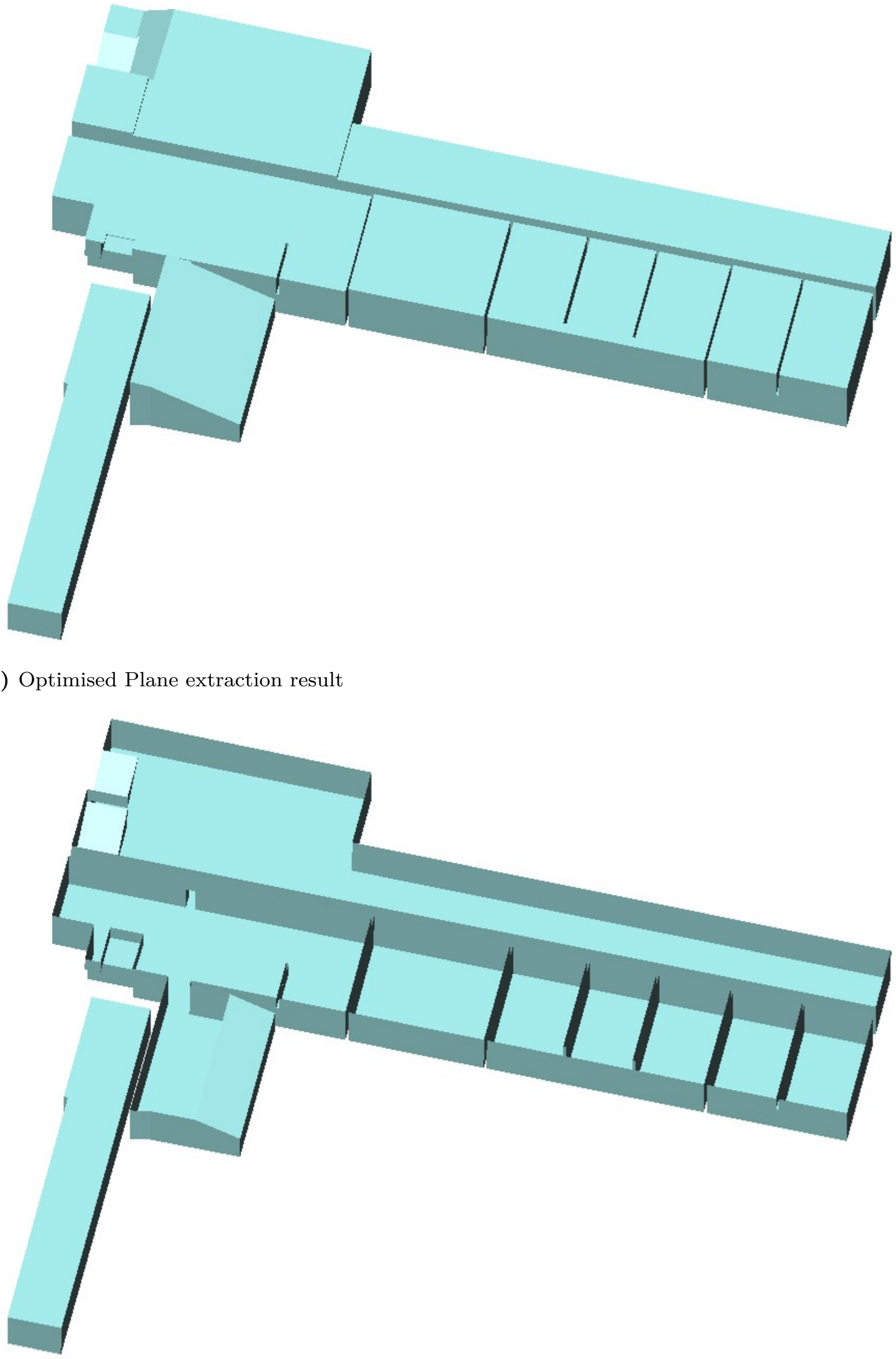
Figure A.1: Plane selection result of of setting $\lambda_{fitting} = 0.6$ and $\lambda_{complexity} = 0.4$



(a) Optimised Plane extraction result

(b) Optimised Plane extraction result(part of the ceiling is removed for better visualisation)

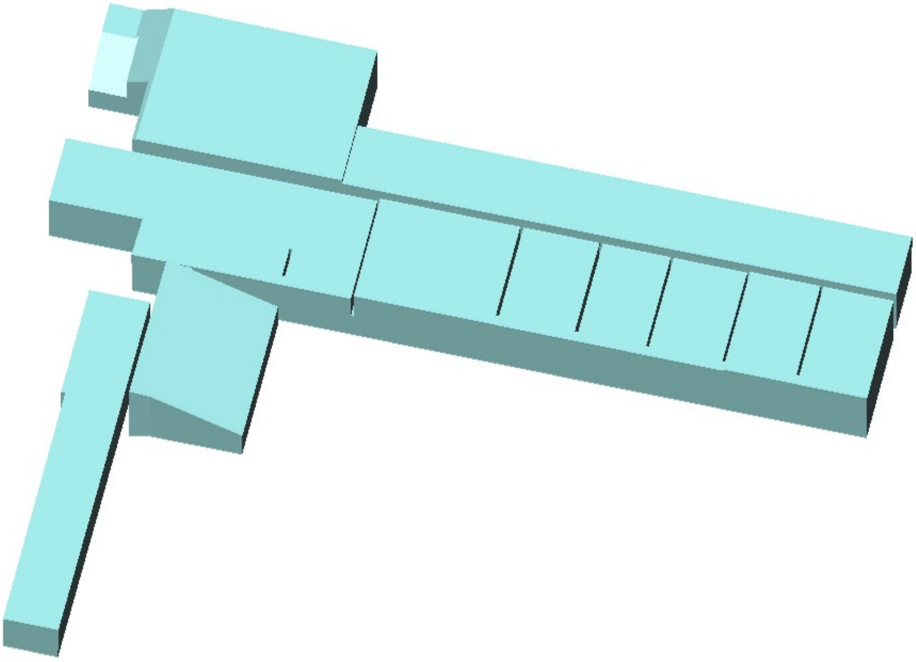
Figure A.2: Plane selection result of of setting $\lambda_{fitting} = 0.8$ and $\lambda_{complexity} = 0.2$



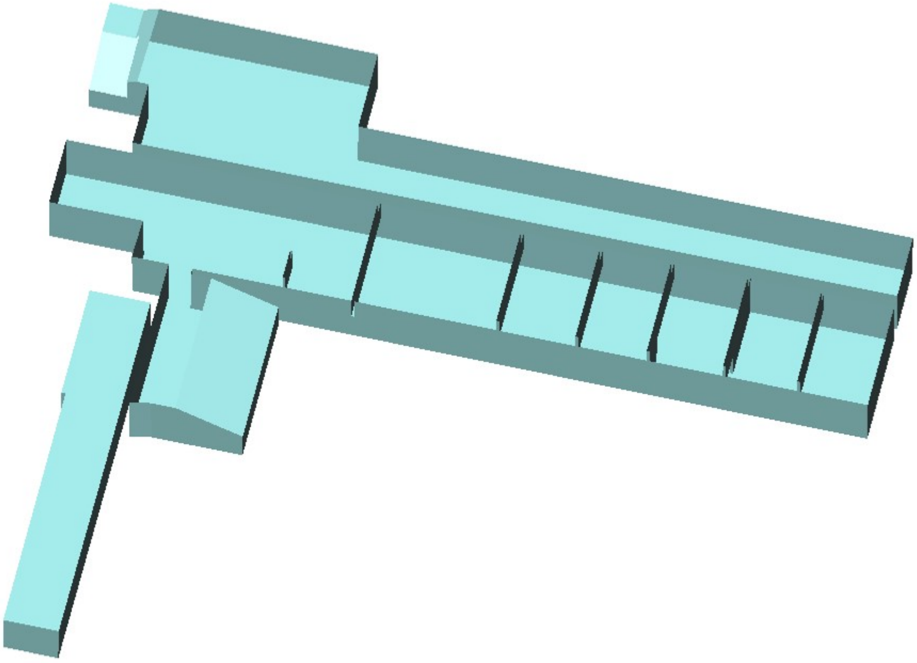
(a) Optimised Plane extraction result

(b) Optimised Plane extraction result(part of the ceiling is removed for better visualisation)

Figure A.3: Plane selection result of of setting $\lambda_{fitting} = 0.5$ and $\lambda_{complexity} = 0.5$



(a) Optimised Plane extraction result

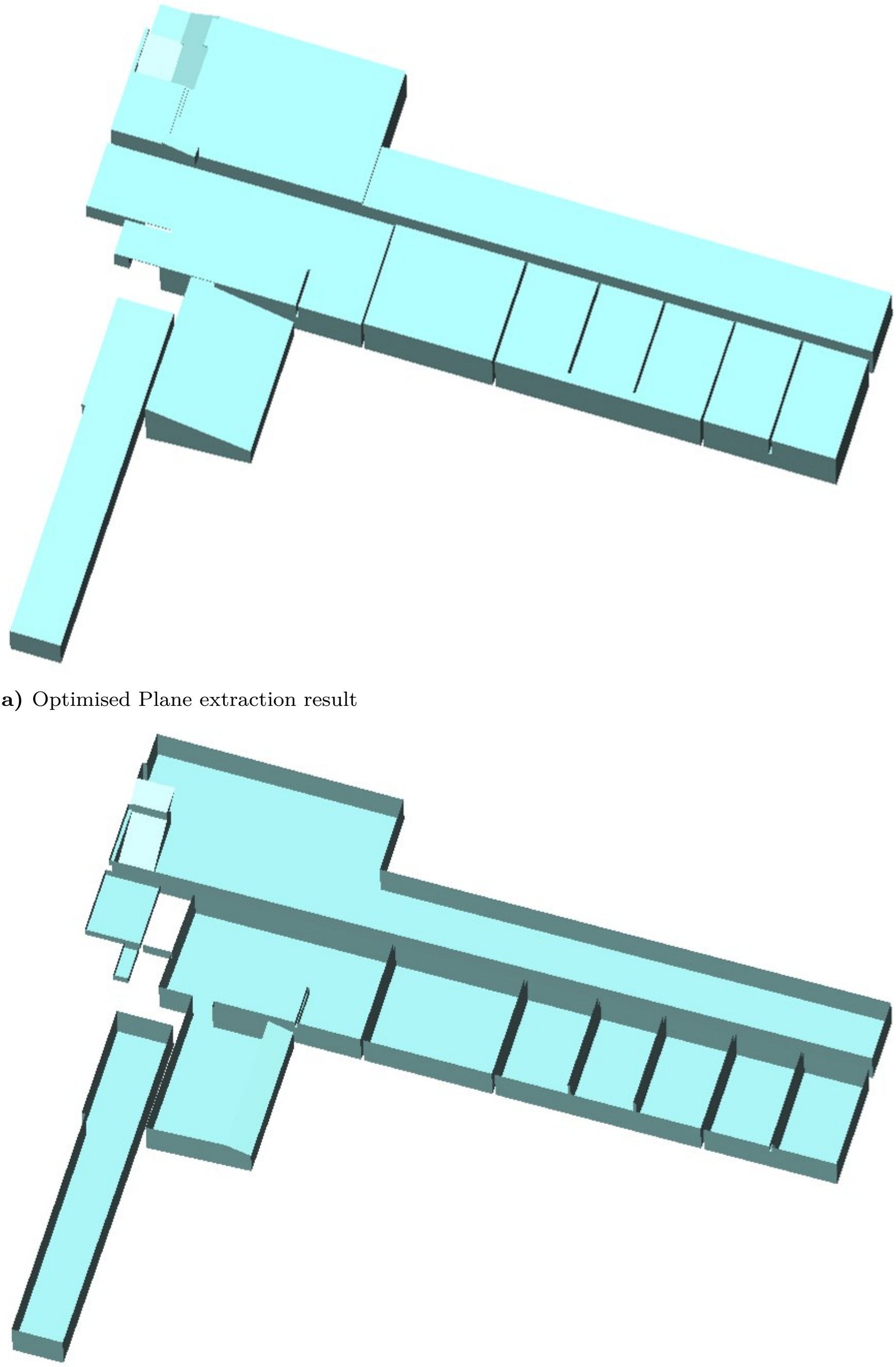


(b) Optimised Plane extraction result(part of the ceiling is removed for better visualisation)

Figure A.4: Plane selection result of of setting $\lambda_{fitting} = 0.2$ and $\lambda_{complexity} = 0.8$

Appendix B

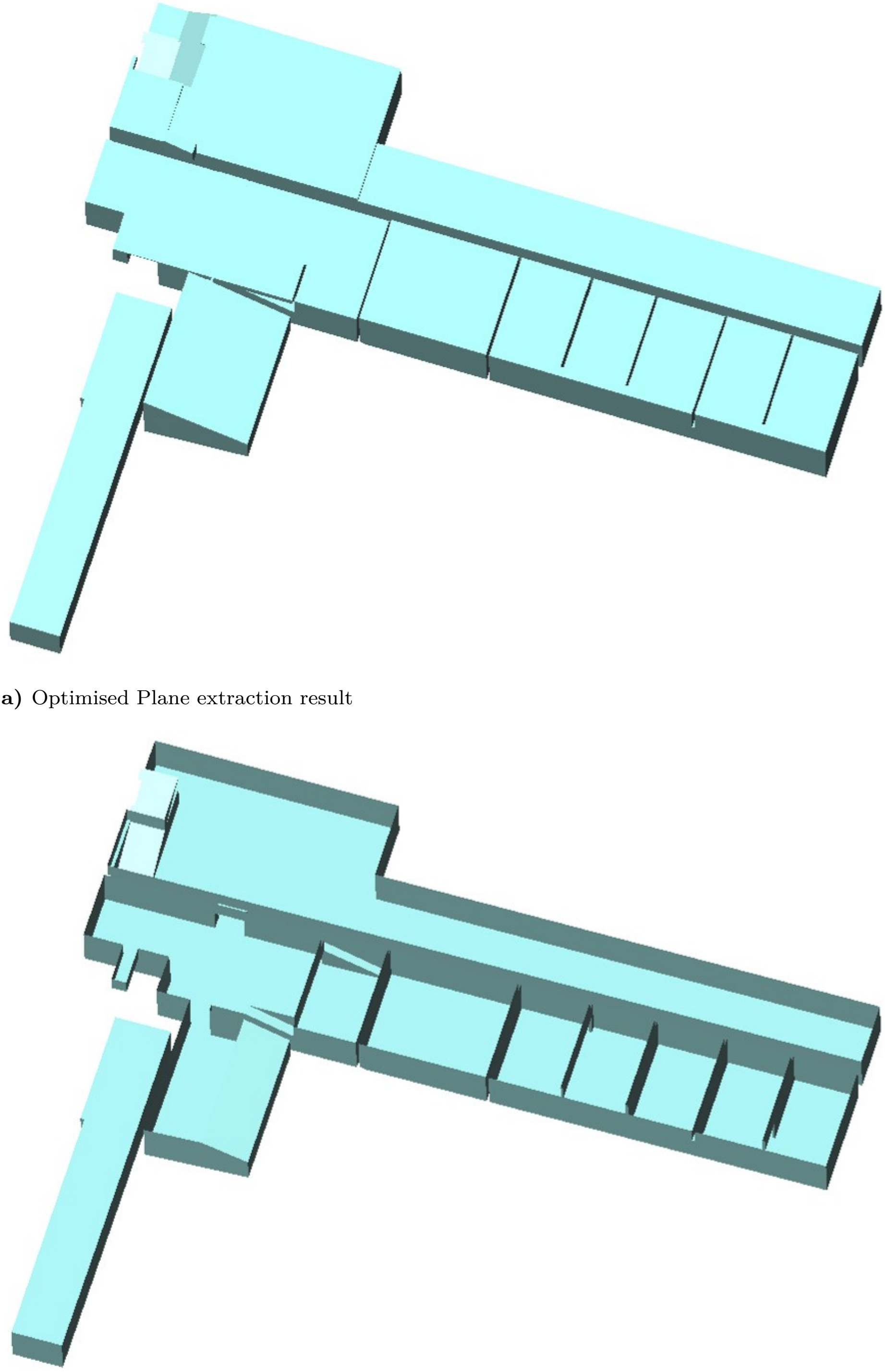
Parameter study on coefficients of point labels in Section 6.2.1



(a) Optimised Plane extraction result

(b) Optimised Plane extraction result(part of the ceiling is removed for better visualisation)

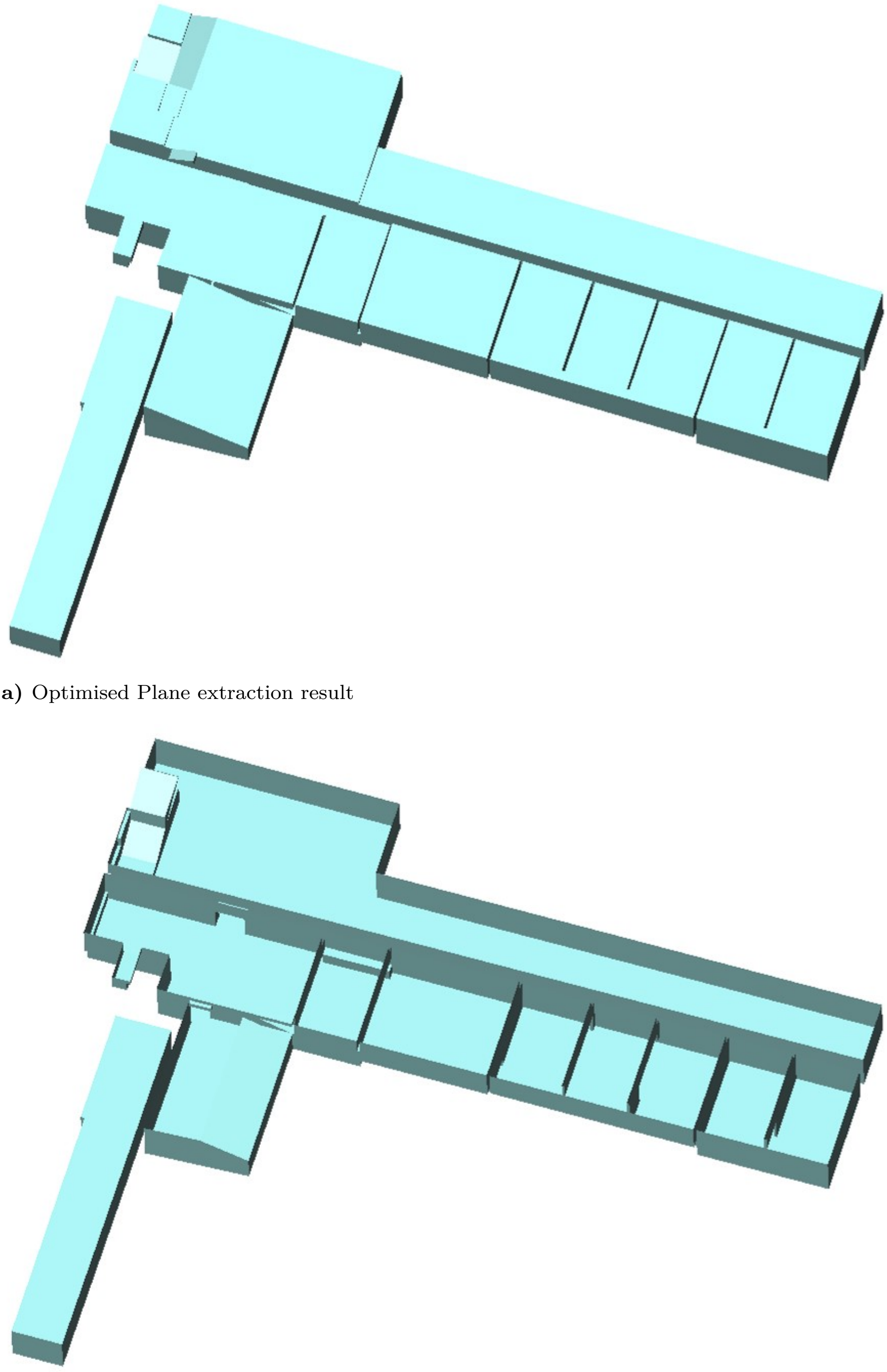
Figure B.1: Plane selection result of setting coefficient for the non-structural label to 0.1



(a) Optimised Plane extraction result

(b) Optimised Plane extraction result(part of the ceiling is removed for better visualisation)

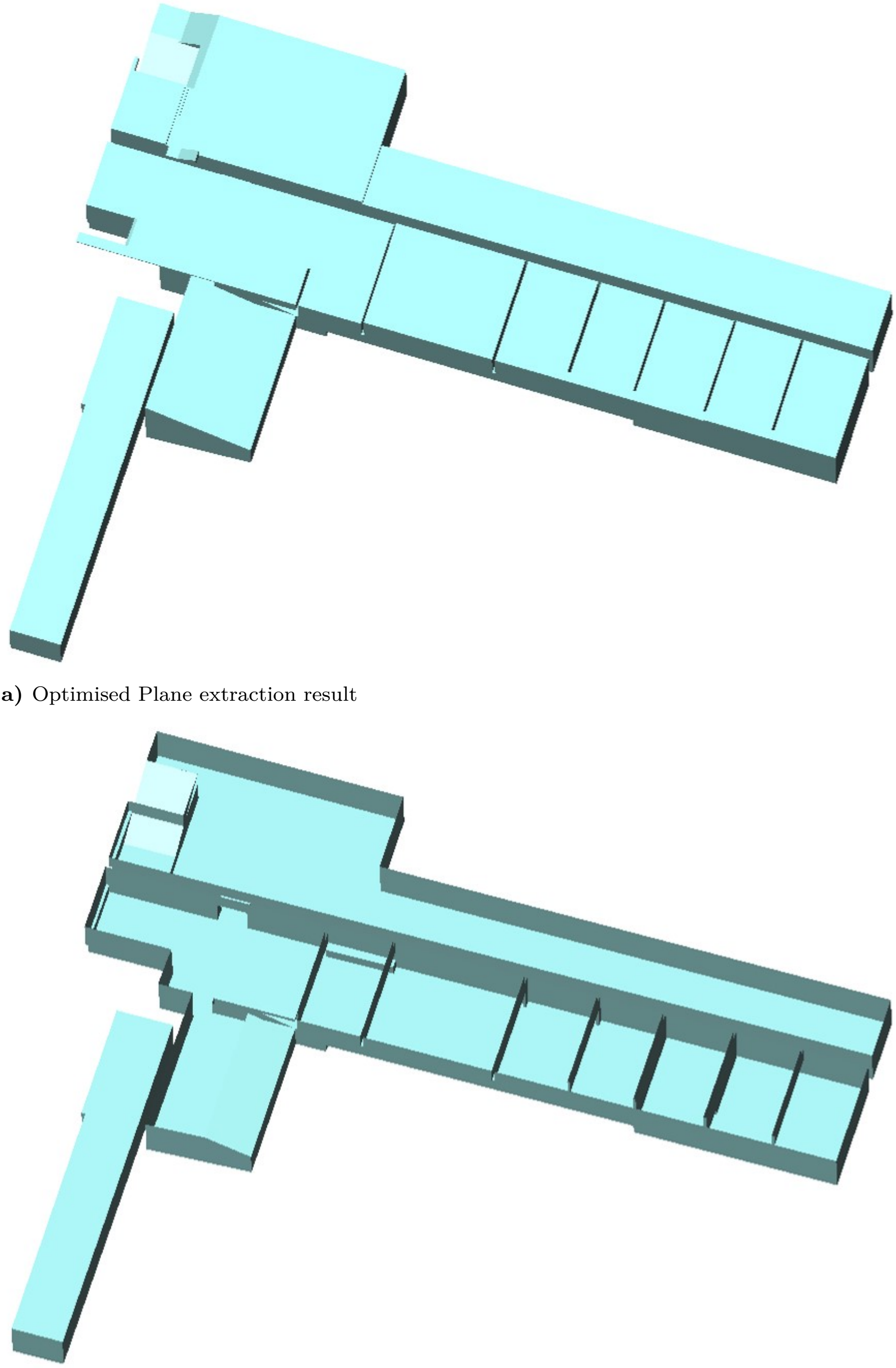
Figure B.2: Plane selection result of setting coefficient for the non-structural label to 0.2



(a) Optimised Plane extraction result

(b) Optimised Plane extraction result(part of the ceiling is removed for better visualisation)

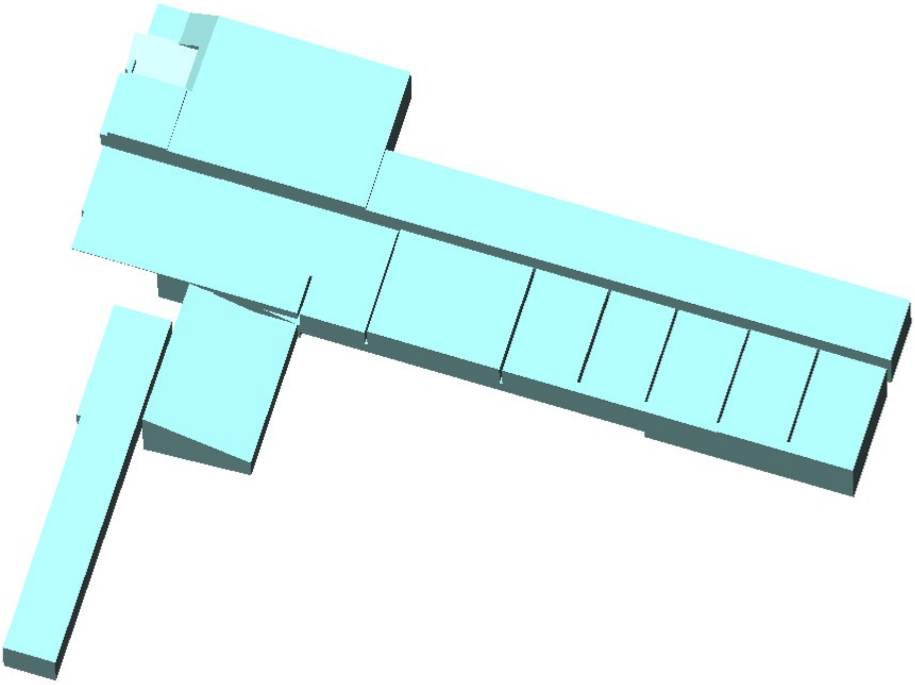
Figure B.3: Plane selection result of setting coefficient for the non-structural label to 0.3



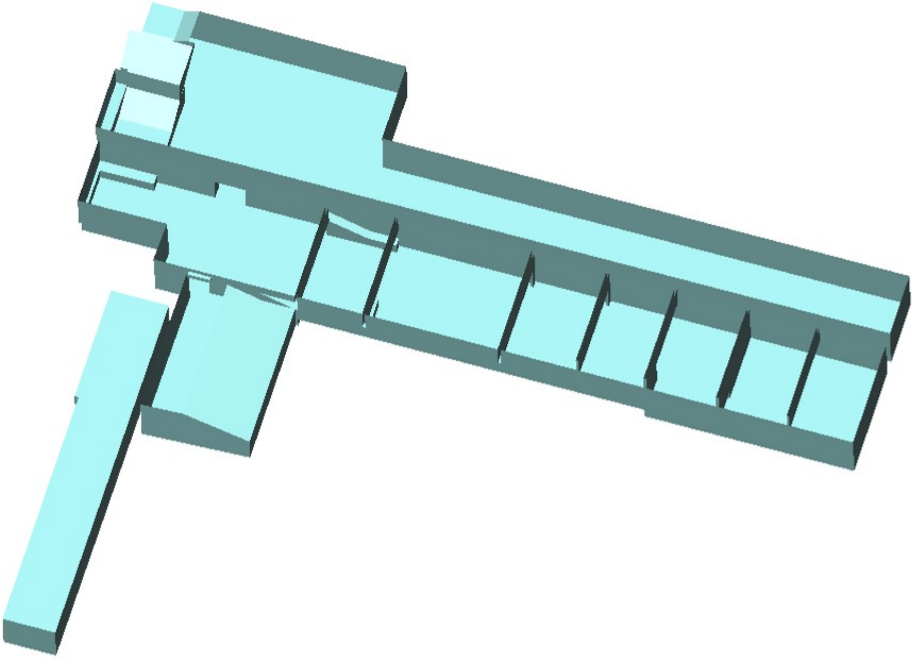
(a) Optimised Plane extraction result

(b) Optimised Plane extraction result(part of the ceiling is removed for better visualisation)

Figure B.4: Plane selection result of setting coefficient for the non-structural label to 0.4



(a) Optimised Plane extraction result



(b) Optimised Plane extraction result(part of the ceiling is removed for better visualisation)

Figure B.5: Plane selection result of setting coefficient for the non-structural label to 0.5

Appendix C

Text recognition result in Section 7.2.4

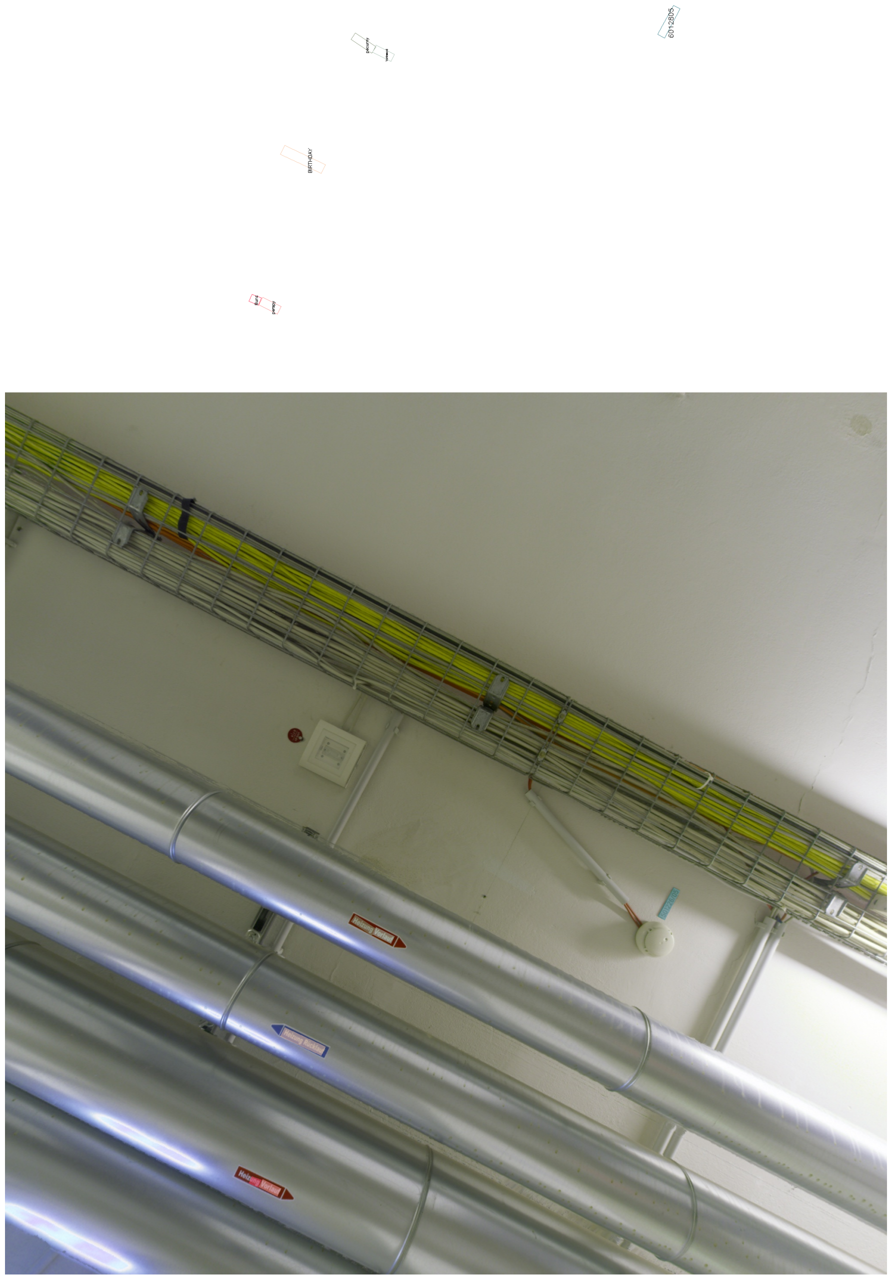


Figure C.1: Recognised texts in original image

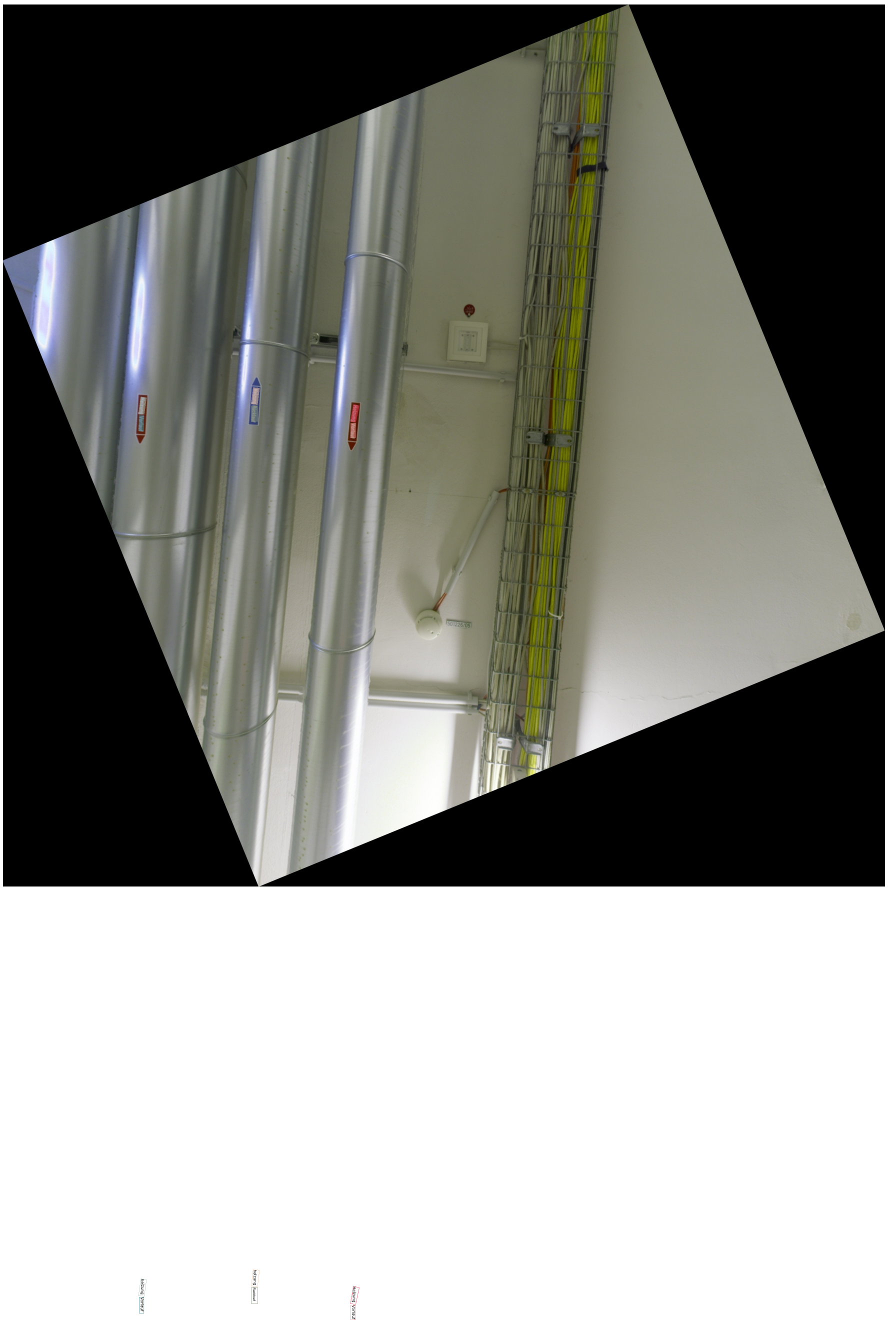


Figure C.2: Recognised texts in one rotated image

Figure C.3: Recognised texts in the other rotated image

Bibliography

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

Adán, A., B. Quintana, S. A. Prieto, and F. Bosché (2018). Scan-to-BIM for 'secondary' building components. *Advanced Engineering Informatics 37*, 119–138.

Agapaki, E. and I. Brilakis (2020). CLOI-NET: Class segmentation of industrial facilities' point cloud datasets. *Advanced Engineering Informatics 45*, 101121.

Agapaki, E. and I. Brilakis (2021). Instance segmentation of industrial point cloud data. *Journal of Computing in Civil Engineering 35*(6), 04021022.

Agapaki, E., G. Miatt, and I. Brilakis (2018). Prioritizing object types for modelling existing industrial facilities. *Automation in Construction 96*, 211–223.

Agatonovic-Kustrin, S. and R. Beresford (2000). Basic concepts of artificial neural network (ANN) modeling and its application in pharmaceutical research. *Journal of pharmaceutical and biomedical analysis 22*(5), 717–727.

Aheleroff, S., X. Xu, R. Y. Zhong, and Y. Lu (2021). Digital twin as a service (DTaaS) in industry 4.0: an architecture reference model. *Advanced Engineering Informatics 47*, 101225.

Alam, K. M. and A. El Saddik (2017). C2PS: A digital twin architecture reference model for the cloud-based cyber-physical systems. *IEEE access 5*, 2050–2062.

Ambruş, R., S. Claici, and A. Wendt (2017). Automatic room segmentation from unstructured 3-d data of indoor environments. *IEEE Robotics and Automation Letters 2*(2), 749–756.

Armeni, I., O. Sener, A. R. Zamir, H. Jiang, I. Brilakis, M. Fischer, and S. Savarese (2016). 3D semantic parsing of large-scale indoor spaces. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1534–1543.

Ayani, M., M. Ganebäck, and A. H. Ng (2018). Digital twin: Applying emulation for machine reconditioning. *Procedia Cirp 72*, 243–248.

Bachiega, G. (2017). Developing an embedded digital twin for HVAC device diagnostics.

Baek, F., I. Ha, and H. Kim (2019). Augmented reality system for facility management using image-based indoor localization. *Automation in Construction 99*, 18–26.

Besl, P. J. and R. C. Jain (1988). Segmentation through variable-order surface fitting. *IEEE Transactions on Pattern Analysis and Machine Intelligence 10*(2), 167–192.

Besl, P. J. and N. D. McKay (1992). Method for registration of 3-d shapes. In *Sensor fusion IV: control paradigms and data structures*, Volume 1611, pp. 586–606. International Society for Optics and Photonics.

Bhanu, B., S. Lee, C.-C. Ho, and T. Henderson (1986). Range data processing: Representation of surfaces by edges. In *Proceedings of the eighth international conference on pattern recognition*, pp. 236–238. IEEE Computer Society Press.

Bloch, T. and R. Sacks (2018). Comparing machine learning and rule-based inferencing for semantic enrichment of BIM models. *Automation in Construction 91*, 256–272.

Bolles, R. C. and M. A. Fischler (1981). A RANSAC-based approach to model fitting and its application to finding cylinders in range data. In *IJCAI*, Volume 1981, pp. 637–643.

Bolton, A., L. Butler, I. Dabson, M. Enzer, M. Evans, T. Fenemore, F. Harradence, E. Keaney, A. Kemp, A. Luck, et al. (2018). Gemini principles.

Bolton, R. N., J. R. McColl-Kennedy, L. Cheung, A. Gallan, C. Orsingher, L. Witell, and M. Zaki (2018). Customer experience challenges: bringing together digital, physical and social realms. *Journal of Service Management 29*(5), 776–808.

Borrmann, D., J. Elseberg, K. Lingemann, and A. Nüchter (2011). The 3D hough transform for plane detection in point clouds: A review and a new accumulator design. *3D Research 2*(2), 3.

Borth, M., J. Verriet, and G. Muller (2019). Digital twin strategies for SoS 4 challenges and 4 architecture setups for digital twins of SoS. In *2019 14th Annual Conference System of Systems Engineering (SoSE)*, pp. 164–169.

Bosché, F., M. Ahmed, Y. Turkan, C. T. Haas, and R. Haas (2015). The value of integrating Scan-to-BIM and Scan-vs-BIM techniques for construction monitoring using laser scanning and BIM: The case of cylindrical MEP components. *Automation in Construction 49*, 201–213.

Braun, A. and A. Borrmann (2019). Combining inverse photogrammetry and BIM for automated labeling of construction site images for machine learning. *Automation in Construction 106*, 1–13.

Braun, A., S. Tuttas, U. Stilla, and A. Borrmann (2020). Improving progress monitoring by fusing point clouds, semantic data and computer vision. *Automation in Construction 116*(C), 1–16.

Brilakis, I., Y. Pan, A. Borrmann, H.-G. Mayer, F. Rhein, C. Vos, E. Pettinato, and S. Wagner (2019). Built environment digital twining.

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

Cao, Y., X. Song, and T. Wang (2015). Development of an energy-aware intelligent facility management system for campus facilities. *Procedia engineering 118*, 449–456.

Chaperon, T. and F. Goulette (2001, November). Extracting cylinders in full 3D data using a random sampling method and the Gaussian image. In *Vision Modeling and Visualization Conference 2001 (VMV-01)*, Stuttgart, Germany.

Chen, H., L. Hou, G. K. Zhang, and S. Moon (2021). Development of BIM, IoT and AR/VR technologies for fire safety and upskilling. *Automation in Construction 125*, 103631.

Chen, K., J. Yang, J. C. Cheng, W. Chen, and C. T. Li (2020). Transfer learning enhanced AR spatial registration for facility maintenance management. *Automation in Construction 113*, 103135.

Cheng, J. C., W. Chen, K. Chen, and Q. Wang (2020). Data-driven predictive maintenance planning framework for MEP components based on BIM and IoT using machine learning algorithms. *Automation in Construction 112*, 103087.

Collins, F. C., M. Ringsquandl, A. Braun, D. M. Hall, and A. Borrmann (2022). Shape encoding for semantic healing of design models and knowledge transfer to scan-to-bim. *Proceedings of the Institution of Civil Engineers - Smart Infrastructure and Construction 175*(4), 160–180.

Coughlan, J. and A. Yuille (1999, Sep.). Manhattan world: compass direction from a single image by bayesian inference. In *Proceedings of the Seventh IEEE International Conference on Computer Vision*, Volume 2, pp. 941–947 vol.2.

Coughlan, J. and A. L. Yuille (2000). The manhattan world assumption: Regularities in scene statistics which enable bayesian inference. *Advances in Neural Information Processing Systems 13*, 845–851.

Czerniawski, T., M. Nahangi, C. Haas, and S. Walbridge (2016). Pipe spool recognition in cluttered point clouds using a curvature-based shape descriptor. *Automation in Construction 71*, 346–358.

Deng, J., W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei (2009). Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pp. 248–255. Ieee.

Deschaud, J.-E. and F. Goulette (2010). A fast and accurate plane detection algorithm for large noisy point clouds using filtered normals and voxel growing. In *3DPVT*. Hal Archives-Ouvertes Paris, France.

Díaz-Vilariño, L., H. González-Jorge, J. Martínez-Sánchez, and H. Lorenzo (2015). Automatic lidar-based lighting inventory in buildings. *Measurement 73*, 544–550.

Diebel, J. and S. Thrun (2006). An application of markov random fields to range sensing. In *Advances in neural information processing systems*, pp. 291–298.

Dimitrov, A. and M. Golparvar-Fard (2015). Segmentation of building point cloud models including detailed architectural/structural features and mep systems. *Automation in Construction 51*, 32–45.

Douillard, B., J. Underwood, N. Kuntz, V. Vlaskine, A. Quadros, P. Morton, and A. Frenkel (2011). On the segmentation of 3D lidar point clouds. In *2011 IEEE International Conference on Robotics and Automation*, pp. 2798–2805. IEEE.

D'Urso, C. (2011). Information integration for facility management. *IT Professional 13*(6), 48–53.

El Saddik, A. (2018). Digital twins: The convergence of multimedia technologies. *IEEE MultiMedia 25*(2), 87–92.

Engel, N., V. Belagiannis, and K. Dietmayer (2021). Point transformer. *IEEE Access 9*, 134826–134840.

Everingham, M., L. Van Gool, C. K. Williams, J. Winn, and A. Zisserman (2010). The pascal visual object classes (voc) challenge. *International journal of computer vision 88*(2), 303–338.

Fan, S., Q. Dong, F. Zhu, Y. Lv, P. Ye, and F.-Y. Wang (2021). SCF-net: Learning spatial contextual features for large-scale point cloud segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 14504–14513.

Felzenszwalb, P. F. and D. P. Huttenlocher (2004). Efficient graph-based image segmentation. *International journal of computer vision 59*(2), 167–181.

Fischler, M. A. and R. C. Bolles (1981, jun). Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Commun. ACM 24*(6), 381–395.

Fotland, G., C. Haskins, and T. Rølvåg (2020). Trade study to select best alternative for cable and pulley simulation for cranes on offshore vessels. *Systems Engineering 23*(2), 177–188.

Galvanin, E. and A. P. P. Dal Poz (2011). Extraction of building roof contours from lidar data using a markov-random-field-based approach. *IEEE transactions on geoscience and remote sensing 50*(3), 981–987.

Girshick, R. (2015). Fast R-CNN. In *Proceedings of the IEEE international conference on computer vision*, pp. 1440–1448.

Girshick, R., J. Donahue, T. Darrell, and J. Malik (2014). Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 580–587.

Glaessgen, E. and D. Stargel (2012). The digital twin paradigm for future nasa and us air force vehicles. In *53rd AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics and Materials Conference 20th AIAA/ASME/AHS Adaptive Structures Conference 14th AIAA*, pp. 1818.

Golovinskiy, A. and T. Funkhouser (2009). Min-cut based segmentation of point clouds. In *2009 IEEE 12th International Conference on Computer Vision Workshops, ICCV Workshops*, pp. 39–46. IEEE.

Goodfellow, I., Y. Bengio, and A. Courville (2016). *Deep learning.* MIT press.

Goulding, J., W. Nadim, P. Petridis, and M. Alshawi (2012). Construction industry offsite production: A virtual reality interactive training environment prototype. *Advanced Engineering Informatics 26*(1), 103–116. Network and Supply Chain System Integration for Mass Customization and Sustainable Behavior.

Grieves, M. and J. Vickers (2017). Digital twin: Mitigating unpredictable, undesirable emergent behavior in complex systems. In *Transdisciplinary perspectives on complex systems*, pp. 85–113. Springer.

Grilli, E., F. Menna, and F. Remondino (2017). A review of point clouds segmentation and classification algorithms. *The International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences 42*, 339.

Guo, M.-H., J.-X. Cai, Z.-N. Liu, T.-J. Mu, R. R. Martin, and S.-M. Hu (2021). Pct: Point cloud transformer. *Computational Visual Media 7*(2), 187–199.

Gupta, A., A. Vedaldi, and A. Zisserman (2016). Synthetic data for text localisation in natural images. In *IEEE Conference on Computer Vision and Pattern Recognition.*

Haghighatgou, N., S. Daniel, and T. Badard (2022). A method for automatic identification of openings in buildings facades based on mobile LiDAR point clouds for assessing impacts of floodings. *International Journal of Applied Earth Observation and Geoinformation 108*, 102757.

He, K., G. Gkioxari, P. Dollár, and R. Girshick (2017). Mask R-CNN. In *Proceedings of the IEEE international conference on computer vision*, pp. 2961–2969.

Hu, Q., B. Yang, L. Xie, S. Rosa, Y. Guo, Z. Wang, N. Trigoni, and A. Markham (2020). Randla-net: Efficient semantic segmentation of large-scale point clouds. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 11108–11117.

Hu, Z.-Z., J.-P. Zhang, F.-Q. Yu, P.-L. Tian, and X.-S. Xiang (2016). Construction and facility management of large MEP projects using a multi-scale building information model. *Advances in Engineering Software 100*, 215–230.

Huang, Q., W. Wang, and U. Neumann (2018). Recurrent slice networks for 3D segmentation of point clouds. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2626–2635.

Imaizumi, J. (2017). Assessment of the process for designing an apartment building through IM and VR. *International Review for Spatial Planning and Sustainable Development 5*(1), 45–54.

Jeelani, I., K. Asadi, H. Ramshankar, K. Han, and A. Albert (2021). Real-time vision-based worker localization & hazard detection for construction. *Automation in Construction 121*, 103448.

Jiang, X. and H. Bunke (1994). Fast segmentation of range images into planar regions by scan line grouping. *Machine vision and applications 7*(2), 115–122.

Jiang, X. Y., U. Meier, and H. Bunke (1996). Fast range image segmentation using high-level segmentation primitives. In *Proceedings Third IEEE Workshop on Applications of Computer Vision. WACV'96*, pp. 83–88. IEEE.

Jiang, Y., D. Pang, and C. Li (2021). A deep learning approach for fast detection and classification of concrete damage. *Automation in Construction 128*, 103785.

Kai, T. and F. Da (2021). 2D alpha shapes. In *CGAL User and Reference Manual* (5.1.5 ed.). CGAL Editorial Board.

Kamble, S. S., A. Gunasekaran, H. Parekh, V. Mani, A. Belhadi, and R. Sharma (2022). Digital twin for sustainable manufacturing supply chains: Current trends, future perspectives, and an implementation framework. *Technological Forecasting and Social Change 176*, 121448.

Kang, J.-G., S.-Y. An, W.-S. Choi, and S.-Y. Oh (2010). Recognition and path planning strategy for autonomous navigation in the elevator environment. *International Journal of Control, Automation and Systems 8*(4), 808–821.

Kim, P. (2017). *Neural Network*, pp. 19–51. Berkeley, CA: Apress.

Kim, P., J. Chen, and Y. K. Cho (2017a). Building element recognition with thermal-mapped point clouds. In *34th International Symposium on Automation and Robotics in Construction (ISARC 2017).*

Kim, P., J. Chen, and Y. K. Cho (2017b). Robotic sensing and object recognition from thermal-mapped point clouds. *International Journal of Intelligent Robotics and Applications 1*(3), 243–254.

Kivrak, S. and G. Arslan (2019). Using augmented reality to facilitate construction site activities. In I. Mutis and T. Hartmann (Eds.), *Advances in Informatics and Computing in Civil and Construction Engineering*, Cham, pp. 215–221. Springer International Publishing.

Kolar, Z., H. Chen, and X. Luo (2018). Transfer learning and deep convolutional neural networks for safety guardrail detection in 2D images. *Automation in Construction 89*, 58–70.

Kotaji, S., A. Schuurmans, and S. Edwards (2003). *Life-Cycle Assessment in Building and Construction: A state-of-the-art report, 2003.* Setac.

Krispel, U., H. L. Evers, M. Tamke, R. Viehauser, and D. Fellner (2015). Automatic texture and orthophoto generation from registered panoramic views. *The International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences 40*(5), 131.

Kritzinger, W., M. Karner, G. Traar, J. Henjes, and W. Sihn (2018). Digital twin in manufacturing: A categorical literature review and classification. *IFAC-PapersOnLine 51*(11), 1016–1022. 16th IFAC Symposium on Information Control Problems in Manufacturing INCOM 2018.

Krizhevsky, A., I. Sutskever, and G. E. Hinton (2012). Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems 25*, 1097–1105.

Kuang, Z., H. Sun, Z. Li, X. Yue, T. H. Lin, J. Chen, H. Wei, Y. Zhu, T. Gao, W. Zhang, K. Chen, W. Zhang, and D. Lin (2021). MMOCR: A comprehensive toolbox for text detection, recognition and understanding. In *Proceedings of the 29th ACM International Conference on Multimedia*, MM '21, New York, NY, USA, pp. 3791–3794. Association for Computing Machinery.

Kwon, O.-S., C.-S. Park, and C.-R. Lim (2014). A defect management system for reinforced concrete work utilizing BIM, image-matching and augmented reality. *Automation in Construction 46*, 74–81.

Lafferty, J. D., A. McCallum, and F. C. N. Pereira (2001). Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the Eighteenth International Conference on Machine Learning*, ICML '01, San Francisco, CA, USA, pp. 282–289. Morgan Kaufmann Publishers Inc.

Laine, S. and T. Karras (2010). Efficient sparse voxel octrees–analysis, extensions, and implementation. Technical report, NVIDIA Corporation.

Landrieu, L. and M. Simonovsky (2018). Large-scale point cloud semantic segmentation with superpoint graphs. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 4558–4567.

Lee, J., B. Bagheri, and H.-A. Kao (2015). A cyber-physical systems architecture for industry 4.0-based manufacturing systems. *Manufacturing letters 3*, 18–23.

Leng, J., H. Zhang, D. Yan, Q. Liu, X. Chen, and D. Zhang (2019). Digital twin-driven manufacturing cyber-physical system for parallel controlling of smart workshop. *Journal of ambient intelligence and humanized computing 10*(3), 1155–1166.

Li, H., P. Wang, C. Shen, and G. Zhang (2019, Jul.). Show, attend and read: A simple and strong baseline for irregular text recognition. *Proceedings of the AAAI Conference on Artificial Intelligence 33*(01), 8610–8617.

Li, Y., R. Bu, M. Sun, W. Wu, X. Di, and B. Chen (2018). Pointcnn: Convolution on x-transformed points. *Advances in neural information processing systems 31*, 820–830.

Li, Y., X. Wu, Y. Chrysathou, A. Sharf, D. Cohen-Or, and N. J. Mitra (2011). Globfit: Consistently fitting primitives by discovering global relations. *ACM transactions on graphics (TOG) 30*(4), 52.

Liao, M., Z. Wan, C. Yao, K. Chen, and X. Bai (2020, Apr.). Real-time scene text detection with differentiable binarization. *Proceedings of the AAAI Conference on Artificial Intelligence 34*(07), 11474–11481.

Lin, T.-Y., M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick (2014). Microsoft coco: Common objects in context. In *European conference on computer vision*, pp. 740–755. Springer.

Liu, C., H. Vengayil, Y. Lu, and X. Xu (2019). A cyber-physical machine tools platform using OPC UA and MTConnect. *Journal of Manufacturing Systems 51*, 61–74.

Liu, C., J. Wu, and Y. Furukawa (2018). Floornet: A unified framework for floorplan reconstruction from 3D scans. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 201–217.

Liu, M., S. Fang, H. Dong, and C. Xu (2021). Review of digital twin about concepts, technologies, and industrial applications. *Journal of Manufacturing Systems 58*, 346–361.

Long, S., J. Ruan, W. Zhang, X. He, W. Wu, and C. Yao (2018). Textsnake: A flexible representation for detecting text of arbitrary shapes. In *European conference on computer vision*, pp. 20–36.

Lu, Q., L. Chen, S. Li, and M. Pitt (2020). Semi-automatic geometric digital twinning for existing buildings based on images and cad drawings. *Automation in Construction 115*, 103183.

Lu, R. and I. Brilakis (2019, May). Recursive segmentation for as-is bridge information modelling. In *Proceedings of the Joint Conference on Computing in Construction (JC3)*, pp. 209–217.

Lu, Y. and C. Rasmussen (2012). Simplified markov random fields for efficient semantic labeling of 3D point clouds. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 2690–2697. IEEE.

Macher, H., T. Landes, and P. Grussenmeyer (2017). From point clouds to building information models: 3D semi-automatic reconstruction of indoors of existing buildings. *Applied Sciences 7*(10), 1030.

Maturana, D. and S. Scherer (2015). Voxnet: A 3D convolutional neural network for real-time object recognition. In *Ieee/rsj International Conference on Intelligent Robots and Systems*, pp. 922–928.

Mayer, H. and S. Reznik (2005). Building facade interpretation from image sequences. In *Proceedings of the ISPRS Workshop CMRT*, pp. 55–60. Citeseer.

Meerman, A., V. Lellek, and D. Serbin (2014). The path to excellence: Integrating customer satisfaction in productivity measurement in facility management. In *Advancing knowledge in Facilities Management: Promoting Innovation in FM*, pp. 201–211. European Facility Management Network (EuroFM).

Meeussen, W., M. Wise, S. Glaser, S. Chitta, C. McGann, P. Mihelich, E. Marder-Eppstein, M. Muja, V. Eruhimov, T. Foote, et al. (2010). Autonomous door opening and plugging in with a personal robot. In *2010 IEEE International Conference on Robotics and Automation*, pp. 729–736. IEEE.

Monszpart, A., N. Mellado, G. J. Brostow, and N. J. Mitra (2015). Rapter: rebuilding man-made scenes with regular arrangements of planes. *ACM Trans. Graph. 34*(4), 103–1.

Mura, C., O. Mattausch, and R. Pajarola (2016). Piecewise-planar reconstruction of multi-room interiors with arbitrary wall arrangements. *Computer Graphics Forum 35*(7), 179–188.

Mura, C., O. Mattausch, A. J. Villanueva, E. Gobbetti, and R. Pajarola (2014). Automatic room detection and reconstruction in cluttered indoor environments with complex room layouts. *Computers & Graphics 44*, 20–32.

Murali, S., P. Speciale, M. R. Oswald, and M. Pollefeys (2017). Indoor scan2BIM: Building information models of house interiors. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 6126–6133. IEEE.

Nan, L. and P. Wonka (2017). Polyfit: Polygonal surface reconstruction from point clouds. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 2353–2361.

Natephra, W., A. Motamedi, T. Fukuda, and N. Yabuki (2017). Integrating building information modeling and virtual reality development engines for building indoor lighting design. *Visualization in Engineering 5*(1), 1–21.

Nath, N. D., A. H. Behzadan, and S. G. Paal (2020). Deep learning for site safety: Real-time detection of personal protective equipment. *Automation in Construction 112*, 103085.

Ning, X., X. Zhang, Y. Wang, and M. Jaeger (2009). Segmentation of architecture shape information from 3D point cloud. In *Proceedings of the 8th International Conference on Virtual Reality Continuum and its Applications in Industry*, pp. 127–132. ACM.

Ochmann, S., R. Vock, and R. Klein (2019). Automatic reconstruction of fully volumetric 3D building models from oriented point clouds. *ISPRS journal of photogrammetry and remote sensing 151*, 251–262.

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

Oesau, S., F. Lafarge, and P. Alliez (2013, May). Indoor scene reconstruction using primitive-driven space partitioning and graph-cut. In *Eurographics Workshop on Urban Data Modelling and Visualisation*, Girona, Spain.

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

Oguchi, T., S. H. Yuichi, and T. Wasklewicz (2011). Chapter seven - data sources. In M. J. Smith, P. Paron, and J. S. Griffiths (Eds.), *Geomorphological Mapping*, Volume 15 of *Developments in Earth Surface Processes*, pp. 189–224. Elsevier.

Okorn, B., X. Xiong, B. Akinci, and D. Huber (2010). Toward automated modeling of floor plans. In *Proceedings of the symposium on 3D data processing, visualization and transmission*, Volume 2.

Olbrich, M., H. Graf, S. Kahn, T. Engelke, J. Keil, P. Riess, S. Webel, U. Bockholt, and G. Picinbono (2013). Augmented reality supporting user-centric building information management. *The visual computer 29*(10), 1093–1105.

Özyeşil, O., V. Voroninski, R. Basri, and A. Singer (2017). A survey of structure from motion. *Acta Numerica 26*, 305–364.

Pan, S. J. and Q. Yang (2009). A survey on transfer learning. *IEEE Transactions on knowledge and data engineering 22*(10), 1345–1359.

Pan, Y., A. Braun, A. Borrmann, and I. Brilakis (2021). Void-growing: a novel scan-to-BIM method for manhattan world buildings from point cloud. In *Proceedings of the 2021 European Conference on Computing in Construction*, Volume 2 of *Computing in Construction*, Online, pp. 312–321. University College Dublin.

Pan, Y., A. Braun, A. Borrmann, and I. Brilakis (2023). 3d deep-learning-enhanced void-growing approach in creating geometric digital twins of buildings. *Proceedings of the Institution of Civil Engineers - Smart Infrastructure and Construction 176*(1), 24–40.

Pan, Y., A. Braun, I. Brilakis, and A. Borrmann (2022). Enriching geometric digital twins of buildings with small objects by fusing laser scanning and ai-based image recognition. *Automation in Construction 140*, 104375.

Pan, Y., F. Noichl, A. Braun, A. Borrmann, and I. Brilakis (2022). Automatic creation and enrichment of 3d models for pipe systems by co-registration of laser-scanned point clouds and photos. In *Proceedings of the 2022 European Conference on Computing in Construction*, Volume 3 of *Computing in Construction*, Rhodes, Greece. University of Turin.

Pan, Y. and L. Zhang (2021). A BIM-data mining integrated digital twin framework for advanced project management. *Automation in Construction 124*, 103564.

Pauly, M., N. J. Mitra, J. Giesen, M. Gross, and L. J. Guibas (2005). Example-based 3D scan completion. In *Proceedings of the Third Eurographics Symposium on Geometry Processing*, SGP '05, Goslar, DEU, pp. 23–es. Eurographics Association.

Perez-Perez, Y., M. Golparvar-Fard, and K. El-Rayes (2021). Scan2BIM-net: Deep learning method for segmentation of point clouds for scan-to-BIM. *Journal of Construction Engineering and Management 147*(9), 04021107.

Previtali, M., L. Barazzetti, R. Brumana, and M. Scaioni (2014). Towards automatic indoor reconstruction of cluttered building rooms from point clouds. *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences 2*(5), 281.

Prokhorov, D. (2010, May). A convolutional learning system for object classification in 3-d lidar data. *Trans. Neur. Netw. 21*(5), 858–863.

Pu, S. and G. Vosselman (2007). Extracting windows from terrestrial laser scanning. *Intl Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences 36*, 12–14.

Pu, S. and G. Vosselman (2009). Knowledge based reconstruction of building models from terrestrial laser scanning data. *ISPRS Journal of Photogrammetry and Remote Sensing 64*(6), 575–584.

Pu, S., G. Vosselman, et al. (2006). Automatic extraction of building features from terrestrial laser scanning. *International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences 36*(5), 25–27.

Puente, I., H. González-Jorge, J. Martínez-Sánchez, and P. Arias (2014). Automatic detection of road tunnel luminaires using a mobile lidar system. *Measurement 47*, 569–575.

Qi, C. R., H. Su, K. Mo, and L. J. Guibas (2017). Pointnet: Deep learning on point sets for 3D classification and segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 652–660.

Qi, C. R., L. Yi, H. Su, and L. J. Guibas (2017). Pointnet++: Deep hierarchical feature learning on point sets in a metric space. In *Advances in Neural Information Processing Systems*, Volume 30. Curran Associates, Inc.

Qiu, S., S. Anwar, and N. Barnes (2021). Semantic segmentation for real point cloud scenes via bilateral augmentation and adaptive fusion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1757–1767.

Reisner-Kollmann, I. and S. Maierhofer (2012). Segmenting multiple range images with primitive shapes. In *2012 19th International Conference on Systems, Signals and Image Processing (IWSSIP)*, pp. 320–323. IEEE.

Ripperda, N. (2008). Determination of facade attributes for facade reconstruction. *International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences 37*(B3a), 285–290.

Roth, S. D. (1982). Ray casting for modeling solids. *Computer graphics and image processing 18*(2), 109–144.

Rusu, R. B. and S. Cousins (2011, May 9-13). 3D is here: Point cloud library (pcl). In *IEEE International Conference on Robotics and Automation (ICRA)*, Shanghai, China. IEEE.

Rusu, R. B., A. Holzbach, N. Blodow, and M. Beetz (2009). Fast geometric point labeling using conditional random fields. In *2009 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 7–12. IEEE.

Sacks, R., I. Brilakis, E. Pikas, H. S. Xie, and M. Girolami (2020). Construction with digital twin information systems. *Data-Centric Engineering 1*, e14.

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

Sappa, A. D. and M. Devy (2001). Fast range image segmentation by an edge detection strategy. In *Proceedings Third International Conference on 3-D Digital Imaging and Modeling*, pp. 292–299. IEEE.

Schiavi, B., V. Havard, K. Beddiar, and D. Baudry (2022). BIM data flow architecture with AR/VR technologies: Use cases in architecture, engineering and construction. *Automation in Construction 134*, 104054.

Schnabel, R., R. Wahl, and R. Klein (2007). Efficient RANSAC for point-cloud shape detection. *Computer Graphics Forum 26*(2), 214–226.

Schoenberg, J. R., A. Nathan, and M. Campbell (2010). Segmentation of dense range information in complex urban scenes. In *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 2033–2038. IEEE.

Schönberger, J. L. and J.-M. Frahm (2016). Structure-from-motion revisited. In *Conference on Computer Vision and Pattern Recognition (CVPR)*.

Schönberger, J. L., E. Zheng, M. Pollefeys, and J.-M. Frahm (2016). Pixelwise view selection for unstructured multi-view stereo. In *European Conference on Computer Vision (ECCV)*.

Seitz, S., B. Curless, J. Diebel, D. Scharstein, and R. Szeliski (2006). A comparison and evaluation of multi-view stereo reconstruction algorithms. In *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, Volume 1, pp. 519–528.

Sheng, F., Z. Chen, and B. Xu (2019). Nrtr: A no-recurrence sequence-to-sequence model for scene text recognition. In *2019 International Conference on Document Analysis and Recognition (ICDAR)*, pp. 781–786. IEEE.

Shi, B., X. Bai, and C. Yao (2017). An end-to-end trainable neural network for image-based sequence recognition and its application to scene text recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence 39*(11), 2298–2304.

Söderberg, R., K. Wärmefjord, J. S. Carlson, and L. Lindkvist (2017). Toward a digital twin for real-time geometry assurance in individualized production. *CIRP Annals 66*(1), 137–140.

Son, H. and C. Kim (2021). Integrated worker detection and tracking for the safe operation of construction machinery. *Automation in Construction 126*, 103670.

Stambler, A. and D. Huber (2014). Building modeling through enclosure reasoning. In *2014 2nd International Conference on 3D Vision*, Volume 2, pp. 118–125. IEEE.

Strom, J., A. Richardson, and E. Olson (2010). Graph-based segmentation for colored 3D laser point clouds. In *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 2131–2136. IEEE.

Suchocki, C. (2020). Comparison of time-of-flight and phase-shift TLS intensity data for the diagnostics measurements of buildings. *Materials 13*(2), 353.

Szegedy, C., A. Toshev, and D. Erhan (2013). Deep neural networks for object detection. In *Advances in Neural Information Processing Systems*, Volume 26. Curran Associates, Inc.

Tan, Y., R. Cai, J. Li, P. Chen, and M. Wang (2021). Automatic detection of sewer defects based on improved you only look once algorithm. *Automation in Construction 131*, 103912.

Tao, F., J. Cheng, Q. Qi, M. Zhang, H. Zhang, and F. Sui (2018). Digital twin-driven product design, manufacturing and service with big data. *The International Journal of Advanced Manufacturing Technology 94*(9-12), 3563–3576.

Tarsha-Kurdi, F., T. Landes, and P. Grussenmeyer (2007). Hough-transform and extended RANSAC algorithms for automatic detection of 3D building roof planes from lidar data. In *ISPRS Workshop on Laser Scanning 2007 and SilviLaser 2007*, Volume 36, pp. 407–412.

The CGAL Project (2020). *CGAL User and Reference Manual* (5.1.1 ed.). CGAL Editorial Board.

Thomas, H., C. R. Qi, J.-E. Deschaud, B. Marcotegui, F. Goulette, and L. J. Guibas (2019). Kpconv: Flexible and deformable convolution for point clouds. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 6411–6420.

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

Torr, P. H. and A. Zisserman (2000). MLESAC: A new robust estimator with application to estimating image geometry. *Computer vision and image understanding 78*(1), 138–156.

Tóvári, D. and N. Pfeifer (2005). Segmentation based robust interpolation-a new approach to laser data filtering. *International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences 36*(3/19), 79–84.

Tran, H., K. Khoshelham, A. Kealy, and L. Díaz-Vilariño (2019). Shape grammar approach to 3D modeling of indoor environments using point clouds. *Journal of Computing in Civil Engineering 33*(1), 04018055.

Truong-Hong, L., D. F. Laefer, T. Hinks, and H. Carr (2013). Combining an angle criterion with voxelization and the flying voxel method in reconstructing building models from LiDAR data. *Computer-Aided Civil and Infrastructure Engineering 28*(2), 112–129.

Truong-Hong, L. and R. Lindenbergh (2022). Extracting structural components of concrete buildings from laser scanning point clouds from construction sites. *Advanced Engineering Informatics 51*, 101490.

Turner, E., P. Cheng, and A. Zakhor (2014). Fast, automated, scalable generation of textured 3D models of indoor environments. *IEEE Journal of Selected Topics in Signal Processing 9*(3), 409–421.

Tuttas, S., A. Braun, A. Borrmann, and U. Stilla (2017). Acquisition and consecutive registration of photogrammetric point clouds for construction progress monitoring using a 4D BIM. *PFG–journal of photogrammetry, remote sensing and geoinformation science 85*(1), 3–15.

VanDerHorn, E. and S. Mahadevan (2021). Digital twin: Generalization, characterization and implementation. *Decision Support Systems 145*, 113524.

Vaswani, A., N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. u. Kaiser, and I. Polosukhin (2017). Attention is all you need. In *Advances in Neural Information Processing Systems*, Volume 30. Curran Associates, Inc.

Veit, A., T. Matera, L. Neumann, J. Matas, and S. Belongie (2016). Coco-text: Dataset and benchmark for text detection and recognition in natural images.

Vosselman, G., S. Dijkman, et al. (2001). 3D building model reconstruction from point clouds and ground plans. *International archives of photogrammetry remote sensing and spatial information sciences 34*(3/W4), 37–44.

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

Walsh, S. B., D. J. Borello, B. Guldur, and J. F. Hajjar (2013). Data processing of point clouds for object detection for structural engineering applications. *Computer-Aided Civil and Infrastructure Engineering 28*(7), 495–508.

Wang, R., L. Xie, and D. Chen (2017). Modeling indoor spaces using decomposition and reconstruction of structural elements. *Photogrammetric Engineering & Remote Sensing 83*(12), 827–841.

Wang, W., E. Xie, X. Li, W. Hou, T. Lu, G. Yu, and S. Shao (2019). Shape robust text detection with progressive scale expansion network. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 9336–9345.

Wang, W., E. Xie, X. Song, Y. Zang, W. Wang, T. Lu, G. Yu, and C. Shen (2019). Efficient and accurate arbitrary-shaped text detection with pixel aggregation network. In *IEEE/CVF International Conference on Computer Vision*, pp. 8439–8448.

Wang, Y., Y. Sun, Z. Liu, S. E. Sarma, M. M. Bronstein, and J. M. Solomon (2019). Dynamic graph CNN for learning on point clouds. *Acm Transactions On Graphics (tog) 38*(5), 1–12.

Wu, Y., A. Kirillov, F. Massa, W.-Y. Lo, and R. Girshick (2019). Detectron2. https://github.com/facebookresearch/detectron2.

Wu, Y., Y. Qin, Y. Qian, and F. Guo (2021). Automatic detection of arbitrarily oriented fastener defect in high-speed railway. *Automation in Construction 131*, 103913.

Xiang, S., R. Wang, and C. Feng (2019, May). Towards mobile projective AR for construction co-robots. In M. Al-Hussein (Ed.), *Proceedings of the 36th International Symposium on Automation and Robotics in Construction (ISARC)*, Banff, Canada, pp. 1106–1113. International Association for Automation and Robotics in Construction (IAARC).

Xiao, J. and Y. Furukawa (2014). Reconstructing the world's museums. *International journal of computer vision 110*(3), 243–258.

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

Xu, Y., J. He, S. Tuttas, and U. Stilla (2015). Reconstruction of scaffolding components from photogrammetric point clouds of a construction site. *ISPRS Annals of Photogrammetry, Remote Sensing & Spatial Information Sciences 2*, 401–408.

Xu, Y., S. Tuttas, L. Hoegner, and U. Stilla (2018a). Reconstruction of scaffolds from a photogrammetric point cloud of construction sites using a novel 3D local feature descriptor. *Automation in Construction 85*, 76–95.

Xu, Y., S. Tuttas, L. Hoegner, and U. Stilla (2018b). Voxel-based segmentation of 3D point clouds from construction sites using a probabilistic connectivity model. *Pattern Recognition Letters 102*, 67–74.

Yoon, H., H. Song, and K. Park (2011). A phase-shift laser scanner based on a time-counting method for high linearity performance. *Review of Scientific Instruments 82*(7), 075108.

Yue, X., Z. Kuang, C. Lin, H. Sun, and W. Zhang (2020). Robustscanner: Dynamically enhancing positional clues for robust text recognition. In *European Conference on Computer Vision*.

Zaker, R. and E. Coloma (2018). Virtual reality-integrated workflow in BIM-enabled projects collaboration and design review: a case study. *Visualization in Engineering 6*(1), 1–15.

Zaki, T. and C. Khalil (2015). QR-coded clash-free drawings: An integrated system of BIM and augmented reality to improve construction project visualization. In *Proceedings of the ICSC15—The Canadian Society for Civil Engineering's 5th International/11th Construction Specialty Conference, Vancouver, BC, Canada*, pp. 7–10.

Zhang, Y., H. Liu, S.-C. Kang, and M. Al-Hussein (2020). Virtual reality applications for the built environment: Research trends and opportunities. *Automation in Construction 118*, 103311.

Zhao, H., L. Jiang, J. Jia, P. H. Torr, and V. Koltun (2021). Point transformer. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 16259–16268.

Zhao, Y., X. Deng, and H. Lai (2021). Reconstructing BIM from 2D structural drawings for existing buildings. *Automation in Construction 128*, 103750.

Zhao, Z.-Q., P. Zheng, S.-t. Xu, and X. Wu (2019). Object detection with deep learning: A review. *IEEE transactions on neural networks and learning systems 30*(11), 3212–3232.

Zhu, Y., J. Chen, L. Liang, Z. Kuang, L. Jin, and W. Zhang (2021). Fourier contour embedding for arbitrary-shaped text detection. In *IEEE conference on computer vision and pattern recognition*.

Zhuang, C., J. Liu, H. Xiong, X. Ding, S. Liu, and G. Weng (2017). Connotation, architecture and trends of product digital twin. *Computer Integrated Manufacturing Systems 23*(4), 753–768.