University of Zurich[UZH]

Department of Informatics

Room-aware Architectural 3D Modeling of Building Interiors from Point Clouds

Dissertation submitted to the Faculty of Business,
Economics and Informatics
of the University of Zurich

to obtain the degree of
Doktor / Doktorin der Wissenschaften, Dr. sc.
(corresponds to Doctor of Science, PhD)

presented by
Claudio Mura
from Cagliari, Sardinia, Italy

approved in July 2017

at the request of
Prof. Dr. Renato Pajarola
Dr. Enrico Gobbetti

University of Zurich[UZH]

The Faculty of Business, Economics and Informatics of the University of Zurich hereby authorizes the printing of this dissertation, without indicating an opinion of the views expressed in the work.

Zurich, July 19, 2017

Chairman of the Doctoral Board: Prof. Dr. Sven Seuken

ABSTRACT

The widespread diffusion of 3D acquisition systems has made it possible to obtain three-dimensional digital models (such as 3D point clouds) of objects and environments with ease and in a cost-effective manner. Among the other application scenarios, the availability of scanned 3D models of indoor environments promises to have a strong impact in domains such as engineering, architecture and property management. In these domains, compact 3D models representing the as-built structure of building interiors (often organized in so-called *Building Information Models (BIMs)*) can be used in tasks as diverse as renovation planning, energy performance analysis and emergency management.

However, the creation of such models is still by and large a manual process. Despite the efforts profused so far, a satisfactory automatic or semi-automatic pipeline to extract higher-level 3D models of interiors from raw measurements has not been developed yet. The challenges left open by the initial research are manifold. Besides the general problems of coping with missing regions and occlusions in the input data and of detecting the permanent components of the environment, three important challenges are to be addressed: first, how to detect the individual rooms in the environment and to integrate this computation in the modeling pipeline; second, how to effectively model environments with general wall orientations; third, how to process in a scalable manner large environments composed of many, individual rooms.

This thesis proposes a solution to these issues by introducing three processing pipelines, each taking as input a point-based model of an indoor environment and

producing as output a set of watertight 3D meshes, one for each room of the environment. The proposed approaches, presented in order of increasing complexity and generality, are all designed to work on real-world data and to cope with the typical defects of such inputs. Moreover, each pipeline focuses on tackling a specific problem that is left unsolved by the state-of-the-art.

The first work presented in this thesis focuses on the robust modeling of 2.5D interiors. It introduces an occlusion-aware pruning technique for discarding non-permanent elements and – more importantly – integrates a room detection approach in the reconstruction process, thus going beyond the modeling of the interior space as a single, unstructured entity. While capable of capturing many more real-world environments than the previous works, the method assumes a 2.5D structure in the input environments, requiring that all walls are vertical and that ceilings and floors are horizontal. The second research contribution lifts this assumption and allows to model multi-room interiors with arbitrary wall orientations. In this approach, the permanent structures are selected by analyzing the spatial configuration of adjacent planar parts of the scene and by removing the parts that are not coherent with the structural stability of the building. To cope with the increased complexity of a *full-3D* reconstruction, the extraction of the final room shapes is based on a more effective optimization technique. Compared to the 2.5D case, the three-dimensional data structures used in this method are significantly more expensive to build and can become prohibitively complex if applied to large environments. To address this problem, the last research contribution included in this thesis proposes exploiting the subdivision of buildings into separate rooms to reduce the complexity of the computations. By moving the detection of rooms at the beginning of the processing, this approach allows to reconstruct each sub-environment independently; this drastically reduces the computational resources needed to handle large-scale building interiors with many rooms and allows for a more conservative extraction of the permanent structures of the environment, thus also leading to more precise reconstruction results.

KURZFASSUNG

Dank der weiten Verbreitung von 3D-Messsystemen ist es heutzutage möglich, dreidimensionale digitale Modelle (wie 3D Punktwolken) von Objekten und Umgebungen kostengünstig und auf einfache Weise zu erstellen. Unter anderem verspricht die Verfügbarkeit von aufgenommenen 3D Modellen von Innenräumen starke Auswirkungen auf Gebiete wie Ingenieurwissenschaft, Architektur und Immobilienmanagement zu haben. Kompakte 3D Modelle, die die *as-built* Struktur von Innenräumen darstellen und oft als sogenannte *Building Information Models (BIMs)* organisiert sind, können in solchen Gebieten für so unterschiedliche Aufgaben wie Umbauplanung, Energieeffizienzanalyse und Notfallmanagement verwendet werden.

Jedoch ist die Erstellung von solchen Modellen noch weitgehend ein manueller Prozess. Trotz der bisher geleisteten Anstrengungen ist eine ausreichende Pipeline zur Erzeugung von high-level 3D Modellen von Innenräumen aus Rohmessdaten noch nicht entwickelt worden. Die Herausforderungen, die von den ersten Forschungsleistungen noch offen gelassen wurden, sind vielfältig. Neben den allgemeinen Problemen, mit fehlenden Regionen und Okklusionen in den Eingabedaten zurechtzukommen und die permanenten Bestandteile des Raums zu entdecken, sind drei wichtige Herausforderungen anzusprechen: Erstens, wie man die einzelnen Räume der Umgebung extrahieren und wie diese Berechnung in die Modellierungspipeline integriert werden kann; zweitens, wie man Innenräume mit beliebigen Wandorientierungen effektiv modellieren kann; drittens, wie man grosse Umgebungen mit vielen einzelnen Räumen skalierbar bearbeiten kann.

Diese Arbeit zeigt Lösungen zu diesen Fragen auf, indem sie drei neue Verarbeitungspipelines einführt, die je ein Punktwolkenmodell eines Innenräumes als Eingabe nimmt und eine Gruppe von wasserdichten 3D-Meshes (eine für jeden Raum der Umgebung) als Ausgabe produziert. Die vorgeschlagenen Ansätze werden in aufsteigender Reihenfolge ihrer Komplexität und allgemeinen Anwendbarkeit präsentiert und sind so entworfen, dass sie alle auf Datensätze aus der realen Welt angewandt werden können und den typischen Problemen dieser Eingabedaten gerecht werden. Darüber hinaus ist jede Pipeline für eine spezifische Herausforderung konzipiert, die von bisherigen Methoden noch nicht gelöst wurde.

Der erste Beitrag dieser Arbeit konzentriert sich auf die robuste Modellierung von 2.5D Innenräumen. Er führt eine Methode zur Entfernung der nicht-permanente Elemente ein, die Okklusionen berücksichtigt. Vor allem jedoch integriert unsere Arbeit einen Zugang zur Raumentdeckung in den Rekonstruktionsprozess, was über die reine Modellierung eines Innenraumes als einzige und unstrukturierte Gesamtheit hinaus geht. Obwohl diese Methode deutlich mehr Arten von Umgebungen aus der realen Welt als die bisherigen Pipelines erfassen kann, geht sie davon aus, dass die zu modellierende Umgebung eine 2.5D Struktur hat und verlangt deshalb, dass alle Wände senkrecht und alle Decken und Böden waagerecht sind. Der zweite Forschungsbeitrag lässt diese Annahme fallen und erlaubt, Innenräume mit verschiedenen Zimmern und beliebigen Wandorientierungen zu modellieren. Permanente Strukturen werden bei diesem Ansatz dadurch entdeckt, dass die räumlichen Konfigurationen von adjazenten planaren Teilen der Szene analysiert und die mit der strukturalen Stabilität des Gebäudes nicht übereinstimmenden Teile entfernt werden. Um mit der erhöhten Komplexität einer *full-3D* Rekonstruktion zurechtzukommen, basiert die Produktion der finalen Formen der Räume auf einer effektiveren Optimierungsmethode. Im Vergleich zum 2.5D Fall ist der Aufbau der dreidimensionalen Datenstrukturen, die in dieser Methode verwendet werden, deutlich rechenintensiver, was die Behandlung von grossen Umgebungen unmöglich machen kann. Um dieses Problem zu lösen, zeigt der letzte Forschungsbeitrag dieser Arbeit auf, wie die Unterteilung von Gebäuden in getrennte Räume ausgenutzt werden kann, um die Komplexität der Berechnungen zu verringern. Diese Methode nimmt die Entdeckung der Räume in der Pipeline vorweg und erlaubt dadurch, jede Subumgebung unhabhänging von den Anderen zu rekonstruieren. Dies senkt die erforderlichen Rechenressourcen für die Verarbeitung von weiträumigen Innenräumen mit vielen Zimmern beträchtlich und erlaubt eine konservativere Entdeckung der permanenten Strukturen der Umgebung, was auch zu genaueren Rekonstruktionsergebnissen führt.

ACKNOWLEDGMENTS

This doctoral dissertation is the result of many years of work and would not have been possible without the collaboration and the support of many people.

First of all, I would like to express my gratitude to my advisor Renato Pajarola, who gave me the opportunity to pursue this doctorate at the University of Zurich and to join the DIVA project. The resources at my disposal and the travel opportunities of the last years were amazing. Among the other things, I thank him for the patient support and flexibility shown on many occasions and for running the VMML so that it is not only a team of researchers, but also a group of people who get on well, help each other and enjoy many leisure activities together.

I would also like to thank Enrico Gobbetti for being the co-advisor of this thesis and for all the guidance provided during my studies (and before). I am grateful for his support, especially in the context of the DIVA project and during my time at CRS4.

A huge thank you goes to all my colleagues, both at VMML and in the DIVA project, with whom I shared many great moments and experiences both at work and outside work. Special thanks go to Oliver Mattausch, for his many technical insights and for the great exchange of ideas we have had throughout these years, and to Alberto Jaspe, for the invaluable help he provided me at the beginning of this research project.

Last, but not at all least, I deeply thank my mother, my friends and my relatives for their constant and unconditional moral support during all these years.

CONTENTS

Abstract i

Kurzfassung iii

Acknowledgments v

List of Figures xi

List of Tables xiii

1 Introduction 1
- 1.1 Background and Motivation . . . 2
- 1.2 Research Context . . . 3
- 1.3 Open Challenges . . . 6
- 1.4 Objectives . . . 7
- 1.5 Contributions . . . 8
- 1.6 Dissertation Overview . . . 9

2 Data Processing Pipeline 11
- 2.1 General Processing Pipeline . . . 12
- 2.2 Input Data . . . 14
  - 2.2.1 3D Point Clouds . . . 14
  - 2.2.2 Adjacency Graph of Bounding Rectangles . . . 19

2.3 Output Data 24

3 2.5D Modeling of Multi-room Environments 25
3.1 Motivation and Background 26
3.2 Method Overview 28
3.3 Occlusion-aware Detection of Permanent Components 30
3.4 Construction of Floorplan-based Space Partitioning 31
3.4.1 Computing Representative Lines 32
3.4.2 Cell Complex Construction 32
3.5 Room Detection and Reconstruction 33
3.5.1 Diffusion Embedding 33
3.5.2 Iterative Clustering 37
3.5.3 Post-processing 40
3.5.4 Final Reconstruction 41
3.6 Results 42
3.7 Discussion and Outlook 47

4 Full-3D Modeling of Multi-room Environments 49
4.1 Motivation and Background 50
4.2 Method Overview 51
4.3 Graph-based Detection of Permanent Components 52
4.3.1 Interactive Refinement 58
4.4 Construction of 3D Space Partitioning 59
4.5 Room Detection 61
4.6 Room Reconstruction 63
4.6.1 Data Term 63
4.6.2 Smoothness Term 64
4.7 Results 65
4.8 Discussion and Outlook 73

5 Room-based Modeling of Large-scale Environments 75
5.1 Motivation and Background 76
5.2 Method Overview 78
5.3 Room Detection by Clustering of View Probes 79
5.3.1 View Probes Generation 79
5.3.2 Selection of Interior Probes 81
5.3.3 Room Detection by Clustering of Interior Probes 83
5.3.4 Fuzzy Assignment of Bounding Rectangles 83
5.4 Detection of Candidate Permanent Components 85
5.5 Construction of Space Partitionings 87
5.5.1 Computation of Dominant Planes 87

5.5.2 Construction of Space Partitioning Structures 89
5.6 Room Reconstruction 90
5.6.1 Data Term 90
5.6.2 Smoothness Term 91
5.6.3 Merging of Reconstructed Rooms 92
5.7 Results 92
5.8 Discussion and Outlook 99

6 Conclusions **101**
6.1 Summary 102
6.2 General Discussion 104
6.2.1 Scope of Application of the Proposed Pipelines 104
6.2.2 Discussion on Parameters 104
6.3 Directions for Future Work 107

A Description of Datasets **109**

Bibliography **119**

Curriculum Vitae **127**

LIST OF FIGURES

1.1 Application scenarios for 3D models of building interiors 2
1.2 Simplified overview of the *scan-to-BIM* pipeline. 3

2.1 General pipeline followed by the proposed methods 13
2.2 Description of the *grid set* representation used in the thesis 16
2.3 Definition of the *k-nn* of a point 17
2.4 Consistent alignment of horizontal bounding rectangles. 22
2.5 Definition of the reference frame for the bounding rectangles . . . 23
2.6 Contact graph for the point cloud of an indoor environment 23
2.7 Example of input and output of the modeling pipelines 24

3.1 Overview of our 2.5D reconstruction pipeline. 29
3.2 Computation of the unoccluded vertical extent of a rectangle. . . . 31
3.3 Computation of the coverage of an edge of the 2D cell complex. . 33
3.4 Heatmap visualization of the diffusion distances 35
3.5 Scatterplots of a 3D projection of the Euclidean embedding 35
3.6 Pairwise distances between cells of the 2D cell complex 36
3.7 Illustration of the iterative partitioning process 39
3.8 Post-processing step to correct room over-segmentation. 41
3.9 Reconstruction results for real-world datasets 43
3.10 Reconstruction results for synthetic datasets 44
3.11 Reconstruction results under noise and registration errors 45

3.12 Color-coded visualization of the reconstruction errors for 'Synth 1' 46
3.13 Accuracy of the final wall planes reconstruction for 'Synth 1' . . . 46
3.14 Some limitations of our 2.5D reconstruction approach 48

4.1 Overview of our *full-3D* modeling pipeline. 53
4.2 The six structural patterns used in the structural region growing . . 54
4.3 Example of a valid structural graph in the adjacency graph 56
4.4 Lateral recovery of occluded rectangles 58
4.5 Results of structural region growing and comparison 59
4.6 Examples of interactive refinement of the automatic results. 60
4.7 Extraction of dominant planes by clustering of structural rectangles 61
4.8 3D cell complex, viewpoint cells and clusters of viewpoint cells . 61
4.9 Effects of gravity and room separation terms 66
4.10 Reconstruction results for 2.5D datasets and comparison 68
4.11 Comparison with our 2.5D modeling approach 69
4.12 Reconstruction results for general 3D datasets 70
4.13 Robustness with respect to increased measurement noise 72
4.14 Robustness to varying adjacency threshold in the graph 73
4.15 Some limitations of our *full-3D* method 74

5.1 Overview of our room-based modeling pipeline. 80
5.2 Generation of interior view probes 82
5.3 Room detection by clustering of interior view probes 84
5.4 Reconstruction results of our room-based modeling pipeline . . . 95
5.5 Complexity of the cell complexes 96
5.6 Computation time for the cell complex construction 97
5.7 Scalability of the cell complex construction 98
5.8 Some limitations of our room-based approach 100

LIST OF TABLES

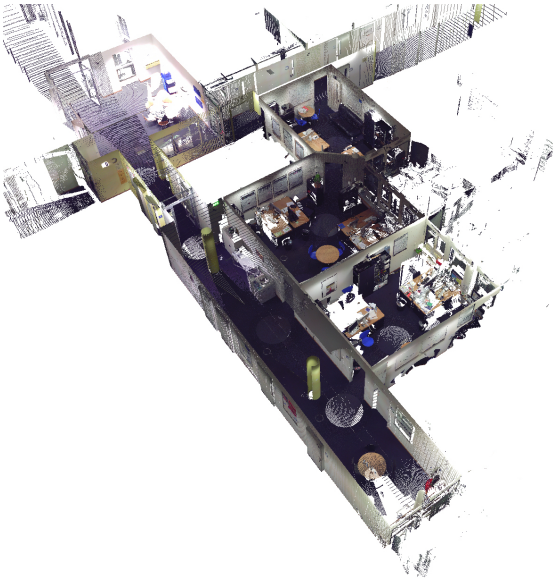
3.1 Relevant statistics and running times of the reconstruction 42
3.2 Reconstruction accuracy for dataset ‘Room 1’ 47

4.1 Relevant statistics on the reconstruction process 67

5.1 Main statistics for the room-based reconstruction 94

C H A P T E R 1

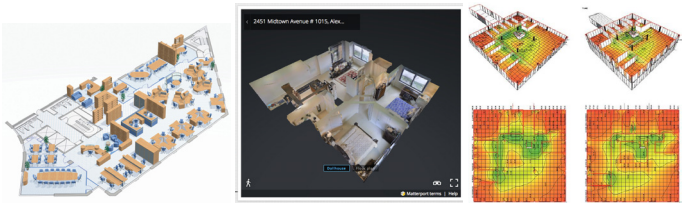
INTRODUCTION



1.1 Background and Motivation

The availability of 3D acquisition systems has increased dramatically in the last years. Not only are low-quality 3D sensors becoming available in cheap, handheld devices such as tablets and smartphones, but also high-quality static laser scanners, once bulky and expensive and therefore only manageable by larger companies, are becoming affordable and easy to use also for the larger public. As a consequence, it is nowadays possible to obtain three-dimensional digital models (such as 3D point clouds) of objects and environments with ease and in a cost-effective manner.

The resulting widespread diffusion of 3D models has opened new frontiers in domains such as engineering, architecture and property management. In particular, practitioners in these fields would greatly benefit from the easy availability of compact, accurate and semantically rich digital representations of buildings, with a special focus on their interior shape. Architects and interior designers can use such models for space planning (see Figure 1.1(a)) or to virtually evaluate new furnishing solutions, while real estate companies are testing web-based 3D visualization platforms to showcase properties online, allowing customers to explore a 3D model of a property before visiting it in person (see Figure 1.1(b)). In the Architecture, Engineering and Construction (AEC) domain, digital models of buildings are often organized in so-called *Building Information Models (BIMs)*, which document many aspects of the construction process and are therefore largely focused on permanent structures such as walls, floors and ceilings. Particularly useful are *as-built* BIMs, which are meant to represent the shape of the structures as they were constructed and must therefore be created from real-world measurements. The applications for these models are manifold and include renovation planning, energy performance analysis (shown in Figure 1.1(c)) and emergency management [Volk et al., 2014].



(a) (b) (c)

Figure 1.1: *Application scenarios for 3D models of building interiors include office space planning (a), virtual property showcasing (b) and energy performance analysis (c).*

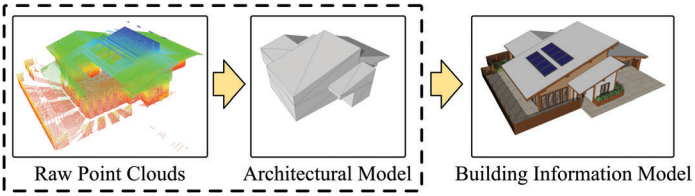


Figure 1.2: *Simplified overview of the* scan-to-BIM *pipeline. An input set of raw 3D point clouds (LEFT) is transformed into a compact 3D architectural model (MIDDLE), which serves as a basis for a higher-level BIM model (RIGHT). The focus of this thesis lies in the extraction of the architectural model from point clouds (shown in the dashed frame).*

In practice, the creation of 3D models of interiors is mostly based on the work of artists and designers, who model by hand the individual elements of the environment using scanned data as reference [Tang et al., 2010]. This results in tedious, expensive and error-prone workflows in which several weeks can be needed before a satisfactory output model is produced. This problem is particularly relevant when creating as-built BIMs of large buildings, for which the amount of objects to be modeled is vast.

For these reasons, there is a strong need for automatic or semi-automatic modeling pipelines that can transform raw and unstructured sets of measurements of interiors into compact and structured 3D architectural models. The output of such pipelines can be employed directly or it can be used as an intermediate result for more general workflows (such as the *scan-to-BIM* process, depicted in Figure 1.2), leading in both scenarios to a drastic reduction in the amount of man-hours required to obtain the final result.

1.2 Research Context

Due to its great practical importance, the problem of modeling indoor environments has been studied by many researchers in the last decade. The range of contributions proposed is wide and spans many sub-fields of architecture, engineering and computer science. Given the vastness of the literature on the topic, we restrict our analysis of the related work to the approaches that can be applied directly to *indoor* environments and that focus on the *architectural* elements of buildings.

In this section, we provide an initial, general overview of the state-of-the-art, in which the most relevant approaches are classified into three main categories based on their scope of application. A more focused analysis is included in the

forthcoming chapters (in particular, in Sections 3.1,4.1,5.1).

Interactive modeling and acquisition Many of the pipelines proposed so far explicitly include human intervention in the modeling process. In particular, as mentioned in the previous section, the creation of as-built BIMs from scanned point clouds (the *scan-to-BIM* problem) is largely a manual process, often requiring the use of several distinct software packages and involving tedious operations such as the manual selection of patches of input points [Tang et al., 2010]. In some more advanced pipelines, originally designed for modeling the outer shape of buildings, user intervention is limited to more intuitive tasks and is interleaved with optimization-based refinement steps; examples of interactive operations include sketching 3D boxes [Nan et al., 2010] or drawing polygonal outlines of shapes [Arikan et al., 2013]. Although these approaches entail intuitive manual operations, applying them to even moderately complex building interiors requires significant amounts of human work.

A number of other approaches integrate human intervention directly in the acquisition stage. Some researchers have proposed custom acquisition setups consisting of off-the-shelf components like depth-cameras and projectors [Kim et al., 2012]; others rely on standard smartphones for the initial capture of the indoor environment and expect that the user performing the acquisition marks relevant features on the acquired images or registers events such as a transition from a room to another [Sankar and Seitz, 2012; Pintore and Gobbetti, 2014]. Bringing the user in the loop already at acquisition time makes it easier to cope with problems like occlusions and missing data. However, besides being labor-intensive, such approaches can only be applied if an ad-hoc acquisition of an environment is possible. In many real-world scenarios, measurements of an environment of interest are already given or must be performed with specific technologies, in order to meet some minimum accuracy levels required by third-party institutions. Similar considerations hold for more general SLAM-based indoor reconstruction pipelines that work on input RGB-D videos [Newcombe et al., 2011; Salas-Moreno et al., 2013; Choi et al., 2015]. In addition to this, most of these methods are aimed at creating general-purpose 3D representations of indoor scenes. Further processing is required to obtain the compact and structured models that are needed in many real-world applications.

Reconstruction of outdoor building structures The problem of modeling buildings interiors is closely related to the more well-studied problem of reconstructing outer urban scenes [Musialski et al., 2013]. While the challenges to be tackled are different, many of the specific techniques used in urban reconstruction methods can be successfully applied to indoor settings. In particular, a

common approach is to extract the geometric model of a building by performing a binary inside/outside partitioning of a spatial structure conveniently built around the object of interest. Several structures have been proposed, including BSP-like space partitionings [Chauve et al., 2010] or tetrahedral subdivisions [Lafarge and Alliez, 2013]. The analogy between such a scheme and some recent indoor reconstruction pipelines [Oesau et al., 2014] is strong. However, datasets of outdoor urban scenes are relatively free of occlusions, which allows to use the intersection parity of visibility rays to reliably steer the inside/outside segmentation. In fact, similar visibility criteria are used also in pipelines that do not strictly follow this segmentation scheme and rely, for instance, on the use of box-like shapes to represent the structures of interest [Vanegas et al., 2012]. However, the visibility-based formulations used in these approaches are bound to fail if directly applied to cluttered indoor settings. Similarly, assumptions like repetitions and structural regularity in the data, commonly used when dealing with outdoor scenes [Zheng et al., 2010; Ceylan et al., 2012; Nan et al., 2015], do not necessarily hold in indoor environments, for which specialized approaches need to be employed.

Automatic modeling of building interiors Although indoor scenes have been object of active research for many years [Udeshi and Hansen, 1999; Gregor and Whitaker, 2001], the interest in the accurate reconstruction of their permanent components, originally mostly limited to applications in the robotics domain [Surmann et al., 2003], has increased significantly only recently. Researchers have tackled the problem of mapping the architectural shape of interiors, aiming not only at recognizing individual permanent structures (e.g. wall segments, columns, stairs) but rather at capturing the global shape of the environment. Although several interactive pipelines have been developed, the holy grail of the research is the *automatic* modeling of entire building interiors.

The solutions initially proposed focus on reconstructing the interior space as a whole, without recovering its room structure, thus limiting the modeling to an inside/outside partitioning typical of outdoor reconstruction. Most approaches manage the complexity of the problem by restricting their scope to the extraction of 2D floorplans [Turner and Zakhor, 2012] or by using restrictive priors on the expected structure of the environment. The so-called *Manhattan-World assumption*, that is, the assumption that cities and indoor scenes are built on a Cartesian grid [Coughlan and Yuille, 2000], has been extensively used both when the input is a set of pictures [Furukawa et al., 2009] and when it consists of dense 3D measurements [Okorn et al., 2010; Budroni and Böhm, 2010; Sanchez and Zakhor, 2012; Turner and Zakhor, 2013]. This prior simplifies the modeling task, but strongly limits the range of environments that can be captured correctly. With some notable exceptions [Okorn et al., 2010], most approaches disregard the problem of

viewpoint occlusions and assume that input scenes are relatively free of clutter – an assumption that is bound to fail when dealing with real-world interiors.

1.3 Open Challenges

Despite the initial research efforts, a satisfactory (semi-)automatic approach for the extraction of structured 3D descriptions of interiors from measured data has not been presented yet. This is due to the high complexity of the problem, in which the many issues to be addressed are often intertwined with each other.

Although the challenges left unsolved by traditional state-of-the-art methods are manifold, they can be broadly categorized into the following groups.

- **Handling noise and incomplete data**

  Raw input data are normally corrupted by measurement noise and contain outliers, which can appear either as isolated spurious measurements or as dense artifacts. Moreover, the presence of clutter in the environment at acquisition time can lead to viewpoint occlusions, which result in missing regions in the input model. It is important that indoor modeling methods produce valid reconstruction results even in the presence of such issues.

- **Reconstruction of the permanent structures**

  The raw input representations of building interiors typically include non-permanent elements (such as chairs, tables and other furniture) that are not relevant to their architectural structure. In this sense, they represent *clutter* that should be discarded and that should not influence the modeling process. For this reason, pipelines for indoor architectural modeling should detect and remove such objects and ensure that the final model only consists in the permanent structures of the environment.

- **Faithful representation of the architecture**

  Modern buildings exhibit a huge variety of architectural shapes and are composed of both planar and curved surfaces that are very often oriented in a non-trivial manner. However, to simplify the modeling process, most state-of-the-art approaches make use of very restrictive assumptions on the properties of the buildings processed, often limiting the range of environments that can be handled to those that have planar walls aligned to three orthogonal directions (the Manhattan-World assumption). A valid modeling pipeline should be able to capture faithfully the variety of architectural shapes that are found in real-world environments, without being limited by restrictive priors.

- **Extraction of the structure of the environment**

  Building interiors are structured into a number of sub-spaces, whose layout and interconnection reflect the intended function of the environment. However, traditional approaches consider an indoor environment simply as a single space separated from the outside by the permanent structures of the building. The modeling process should go beyond such a simple extraction of the interior space and extract the structure of the environment as well.

- **Efficient processing of large-scale environments**

  For a number of application scenarios (especially in the facility management domain), the indoor environments to be modeled are very large and often composed of dozens of individual sub-spaces. State-of-the-art pipelines do not consider this problem explicitly, largely due to the fact that the restrictive assumptions they make allow for the use of low-complexity algorithm and data structures. An important research problem is therefore how to process large-scale interiors efficiently without compromising on the descriptive power of the reconstructed models.

1.4 Objectives

The goal of this thesis is to advance the state-of-the-art in architectural modeling of interiors by tackling some of the most pressing aspects of the open challenges outlined in the previous section. In particular, the specific research objectives of the thesis can be summarized as follows.

1) Robustness to artifacts and occlusions

Dense groups of outliers and missing data caused by viewpoint occlusions are common problems in raw models of interiors, especially in those acquired with static laser range-scanners. A common goal of all pipelines presented in this thesis is therefore to achieve robustness to such issues, ensuring that their effects on the final output is reduced as much as possible.

2) Detection of permanent components

Input measurements corresponding to furniture and other non-architectural elements do not contribute to the desired reconstruction and increase the computational complexity of the modeling process. For this reason, all the proposed approaches aim to detect the permanent elements of the environment and discard those that belong to cluttering objects.

3) Detection of individual rooms

The subdivision into single rooms represents a fundamental piece of information in structured descriptions of interiors. In this work, we aim at performing a room-aware reconstruction, thus going beyond the sheer recovery of the volume delimited by the bounding surfaces of the building.

4) Modeling of arbitrarily oriented planar surfaces

A large array of real-world interiors can not be properly represented using only vertical walls and horizontal floors and ceilings. For this reason, one important goal of this thesis is to capture wall surfaces with arbitrary orientations, using piecewise planarity as only assumption on the architectural shape of the environment.

5) Scalable modeling of large environments

Large-scale building interiors include a large set of distinct architectural surfaces. If considered in its entirety, this information can lead to an unmanageable computational complexity of the modeling process. Therefore, an objective of this thesis is to ensure that the modeling process scales well with respect to the size of the environment, exploiting the subdivision into separate rooms to reduce the amount of data that must be processed together.

1.5 Contributions

The goals set for this thesis are achieved by means of three main research contributions, each described in a specific chapter and here shortly summarized. They are presented in order of increasing complexity and generality, listing for each of them the objectives targeted and (if applicable) the scientific publications in which the contribution is also described.

- **2.5D modeling of multi-room environments**
  Robustness to artifacts and occlusions, Detection of permanent components, Detection of individual rooms

  This work introduces an occlusion-aware pipeline for modeling multi-room building interiors. Compared to the state-of-the-art, it includes a step that selects candidate wall structures in an occlusion-aware fashion and integrates in the reconstruction a room extraction procedure that automatically finds the number of rooms in the environment. Moreover, it lifts the Manhattan-World assumption and can capture (vertical) walls with arbitrary angles between them. The approach, described in detail in Chapter 3, has been presented as a conference contribution [Mura et al., 2013] and has appeared in an extended version as an article in a scientific journal [Mura et al., 2014].

- **Full-3D modeling of multi-room environments**
  Robustness to artifacts and occlusions, Detection of permanent components, Detection of individual rooms, Modeling of arbitrarily oriented planar surfaces

  In this work, the first pipeline for reconstructing multi-room interiors with general 3D architectures is presented. The approach can handle wall structures with arbitrary orientations, thus lifting all assumptions on the alignment of the permanent structures. The increased complexity arising from the full-3D nature of the problem is managed by discarding non-meaningful features from the adjacency graph and by using an optimization-based formulation to reconstruct the final models of the rooms. This contribution (presented in Chapter 4) has appeared as a journal article [Mura et al., 2016] with associated conference talk.

- **Room-based modeling of large-scale environments**
  Robustness to artifacts and occlusions, Detection of permanent components, Detection of individual rooms, Modeling of arbitrarily oriented surfaces, Scalable modeling of large environments

  This contribution focuses on achieving scalability to large environments that contain many different rooms. To do so, the rough spatial extent of each room is detected before performing the actual reconstruction. This allows to perform the bulk of the computations separately on each room, thus drastically limiting the overall complexity of the modeling process. Chapter 5 provides a complete description of this work, which represents novel material introduced in this thesis and has not yet appeared as a separate scientific contribution.

1.6 Dissertation Overview

The rest of this thesis is structured as follows.

Chapter 2 provides an overview of the general data processing pipeline used in this thesis, presenting the general computational steps and describing the nature of the input and the output of the pipeline.

Chapter 3 describes a method for reconstructing 3D models of multi-room building interiors with a 2.5D structure (that is, only containing vertical walls and with horizontal floors and ceilings). The approach is fast, easy to implement and can handle a large number of real-world environments. However, the starting assumptions make it unsuitable for the processing of buildings with more general wall arrangements.

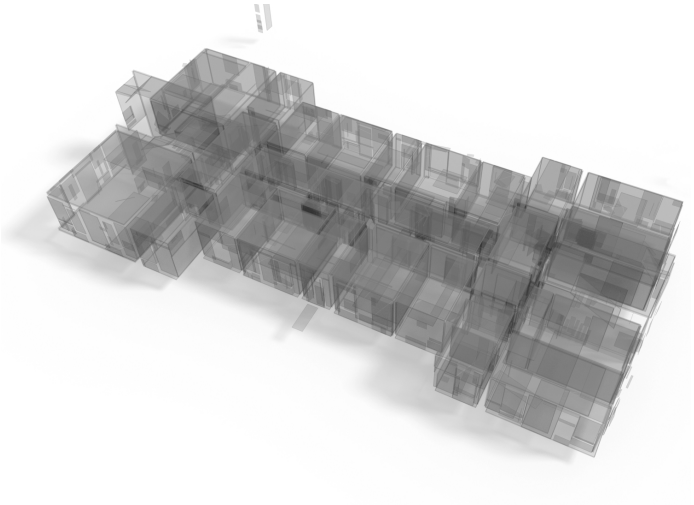
In Chapter 4, a more general approach is presented that lifts the assumptions previously made and allows for the reconstruction of multi-room environments with arbitrary wall orientations.

Scalability to large environments is the key objective of the approach discussed in Chapter 5. In this work, the subdivision of an indoor environment into separate sub-spaces is exploited to allow for efficient processing of buildings composed of many individual rooms, using a process that limits the complexity of the computation to that of the individual sub-spaces.

Finally, Chapter 6 concludes this thesis with an outlook on the results achieved and highlighting some promising directions for future work.

C H A P T E R 2

DATA PROCESSING PIPELINE



The approaches presented in this thesis follow a common processing pipeline and share a number of commonalities in the input and output models used. This chapter describes such common aspects, introducing the general processing pipeline (Section 2.1) used by the methods of Chapters 3, 4 and 5 and discussing the formats used to represent the input (Section 2.2) and the output (Section 2.3) of this pipeline. When discussing the input representation, special attention is devoted to explaining how the raw input measurements are augmented with a more abstract structure that simplifies the subsequent modeling process.

It should be noted that the data representations described in this chapter, as well as the related processing techniques, do not constitute an original research achievement of this thesis, but are relatively standard components also used in other state-of-the-art approaches.

2.1 General Processing Pipeline

The common goal of the methods proposed in this work is to transform an input representation of an indoor environment into a compact 3D model that describes its architectural shape and its room-based structure. The input model, consisting of a set of raw measurements augmented by a more abstract adjacency graph of rectangles (both described in Section 2.2) is transformed into a set of 3D polyhedra (Section 2.3) by applying a sequence of processing steps. While some details of the processing vary depending on the specific approach, it is possible to identify a general pipeline (shown in Figure 2.1) composed of the following main steps.

Detection of permanent components The parts of the environment that are good candidates for permanent structures (e.g. walls, ceilings, floor) are detected and brought forward to the next stages. The remaining elements are considered to originate from clutter (such as furniture) and are therefore discarded.

Construction of space partitioning The space surrounding the input scene is divided into convex sub-regions according to the permanent components selected in the previous step, in such a way that the boundaries of the sub-regions align to the primitives of the permanent components. A single sub-space (i.e. a room) of the input environment can be obtained as the union of a specific set of convex sub-regions of the space partitioning.

Room detection The individual rooms of the environment are detected, along with a set of positions that are inside each room and roughly define its location.

Room reconstruction The shape of each room, defined in terms of its bounding walls, is extracted by detecting the connected set of sub-regions that correspond to the space inside the room and by performing the union of such regions.

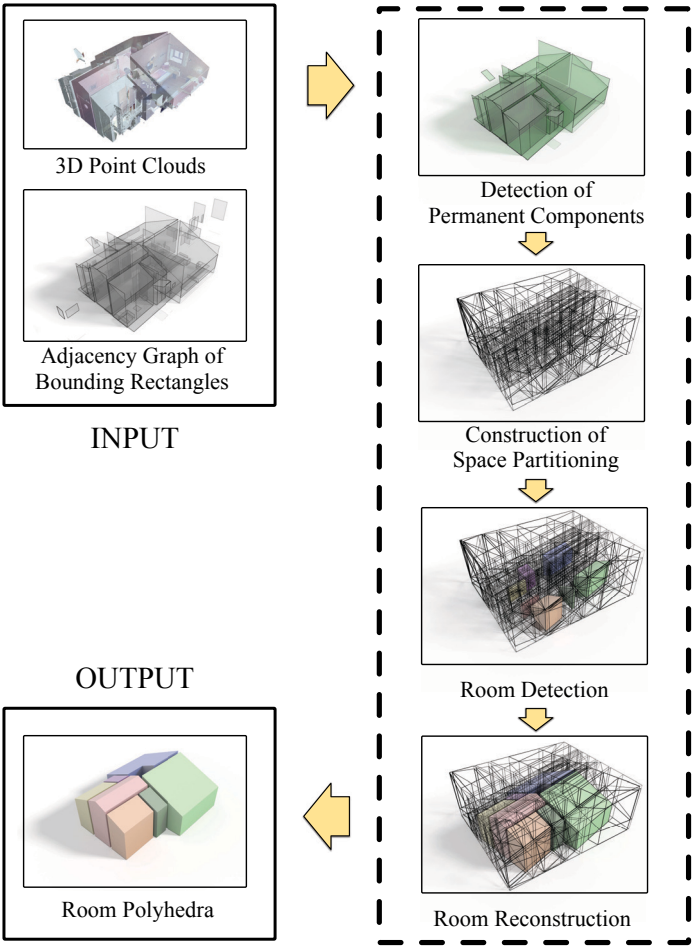


Figure 2.1: *General pipeline followed by the proposed methods. The input representation of the environment (i.e. a set of 3D point clouds and the associated adjacency graph of bounding rectangles) is transformed into a set of 3D polyhedra describing the rooms in the environment by means of a four-step pipeline.*

2.2 Input Data

As shortly introduced in the previous section, the environments to be processed are represented with one or more sets of raw 3D measurements (each commonly known as a *3D point cloud*) accompanied by a graph-based structure that encodes the planar parts of the scene and their adjacencies. This section describes this representation, discussing first how the input 3D point clouds are organized and pre-processed (Section 2.2.1) and then presenting the graph-based structure that is assembled from them (Section 2.2.2).

2.2.1 3D Point Clouds

Many 3D acquisition systems work by sensing along a set of pre-defined directions the distance from their center of projection to the closest real-world surface. Since the center of projection and the set of directions are known, the distance measurements collected can easily be transformed into a set of 3D points, which form a *3D point cloud*.

Traditionally, the sources for this kind of data have been high-end, static laser range-scanners, capable of delivering high-quality and dense measurements. The main downsides of these devices, that is, their high cost and their operational complexity, have been overcome in the last years: today, terrestrial laser scanners like the Faro Focus 3D[1] have become affordable and easy to operate. Cheaper 3D acquisition systems based on consumer-level RGB-D cameras, such as the Matterport Pro 3D Camera[2], are also available for cost-effective acquisition of 3D models of interiors. Moreover, an increasing number of tablets and hand-held devices are equipped with 3D sensors; these, used in combination with the other sensors and with the necessary sensor fusion technology and loop-closure algorithms, can turn even a regular smartphone into a 3D mapping system. This is the case, for instance, of the smartphones equipped with the Google Tango technology[3]. Although with different quality levels, any of these approaches allows to acquire a point-based digital model of an environment, thus making 3D point clouds a convenient and generic initial representation for higher-level 3D modeling pipelines.

Using a mathematical notation, a 3D point cloud P can be defined as a collection of n points in $\mathbb{R}^3$:

$$P = \{\, \mathbf{p}_i = (x_i, y_i, z_i) \mid i = 1, \ldots, n \;,\; x_i, y_i, z_i \in \mathbb{R} \,\}. \tag{2.1}$$

[1] http://www.faro.com/products/3d-surveying
[2] https://matterport.com
[3] https://get.google.com/tango

Since in this format no ordering between the individual points is given, in the following we refer to it as an *unstructured* point cloud. Each point $\mathbf{p}_i$ represents a sample on a surface of the real-world. In order to represent an object or an environment in a faithful manner, a sufficient number of samples must be included in the point cloud to represent the surfaces of interest. Depending on the complexity of the scene to be captured, the point cloud needs to be created by merging subsets of samples acquired from different viewpoints. Each of these subsets is itself a point cloud, often referred to as a *scan* to highlight the coherent origin of its points.

Points from a same scan are acquired by the scanning device using a specific pattern; in particular, points in space that project to nearby locations in the view plane of the device are scanned subsequently. While the specific details vary depending on the device, it is often possible to keep the samples in a scan arranged in a way that reflects the acquisition pattern. A frequently used format is the so-called *range-grid* (see Figure 2.2), in which points are arranged in a grid that represents a 2D parametrization of the field-of-view of the scanning device. In this representation, each point is associated not only with a triple of coordinates in $\mathbb{R}^3$, but also with a pair of coordinates (u, v) that describe its position in the parametrization. Using a more formal notation, a range-grid P^{grid} can be described as follows:

$$P^{\text{grid}} = \{\, \mathbf{p}_i = (x_i, y_i, z_i, u, v) \mid i = 1, \ldots, n,\ x_i, y_i, z_i \in \mathbb{R}, \\ u, v \in \mathbb{N},\ 1 \leq u \leq n_{\text{rows}}, 1 \leq v \leq n_{\text{cols}} \,\}. \tag{2.2}$$

where n_{rows} and n_{cols} are the dimensions of the discretized parametrization of the field-of-view of the device. In practice, the coordinates of each point inside the parametrization can be represented implicitly by storing the points in a specific order (e.g. row-major or column-major).

Each scan contains samples that are expressed in a local reference system. To create a single point cloud from multiple scans it is necessary to align the individual scans into a single reference frame. This process, often denoted as *registration*, associates each scan with a matrix $\mathbf{T}^{\text{reg}}$ expressing the rigid body transformation that brings the scan from its local reference system to the global one.

A point cloud can be represented as a *grid set* consisting of m individual range-grids, as shown in Figure 2.2 and expressed by the following notation:

$$P^{\text{grid set}} = \{\, (P_i^{\text{grid}}, \mathbf{T}_i^{\text{reg}}) \mid i = 1, \ldots, m \,\}. \tag{2.3}$$

For a number of applications, especially those involving view-dependent computations, it is convenient to represent the input point cloud using this format.

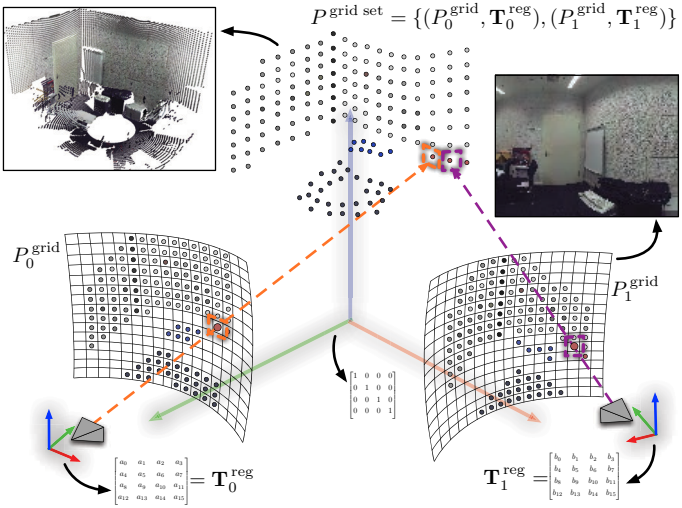


Figure 2.2: *Description of the* grid set *representation used in the thesis. The points of a grid set point cloud ($P^{\text{grid set}}$ in the figure) result from the union of a set of* range grid *point clouds (P_0^{grid} and P_1^{grid}). Each range grid is associated with a transformation matrix ($\mathbf{T}_0^{\text{reg}}$ and $\mathbf{T}_1^{\text{reg}}$) that aligns its points to the common reference system of the point cloud.*

Local Features Estimation

The point clouds described above represent the typical output of many 3D acquisition devices. In such representations, each sample only carries basic geometric information about its position in the 3D space. Even for basic point-based geometry processing it is useful to augment every sample with additional features that describe the properties of the underlying real-world surface around the sample. For the tasks presented in this thesis, we augment each sample with two features: the *normal vector*, which points in the direction perpendicular to the surface at the point, and the *radius of influence*, which represents the spatial extent over which the sample approximates the underlying surface.

Since normal vector and radius of influence are *local* properties of the surface, they are typically estimated from the set of points that lie in the spatial neighborhood of a point or, alternatively, from its k nearest points (also known as k

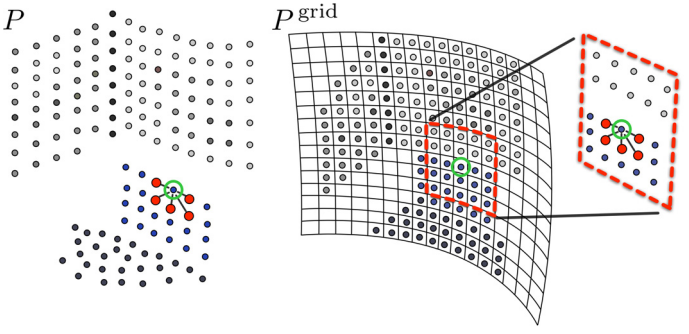


Figure 2.3: *Definition of the $k = 5$ nearest neighbors of a point in an unstructured (a) and in a range-grid point cloud (b). The query point is highlighted by the green circle*

nearest neighbors or *k-nn*). In unstructured point clouds (see Equation 2.1), this set of points is often obtained by querying a pre-built spatial data structure (e.g. a kd-tree [Gross and Pfister, 2007]). If a grid parametrization is available, however, a practical approach is to approximate this set of points by considering the points on the grid that are contained in a rectangular window centered at the point of interest [Steder et al., 2011]. This is justified by the fact that points that are close in 3D space project to adjacent locations in the view plane of the acquisition device. Due to a number of factors (variable orientations of real-world surfaces with respect to the view direction, discontinuities caused by the projection process, different distortion factors of the different regions of the parametrization), the points in the window do not coincide with the exact spatial neighbors. A good approximation, however, can be obtained by considering the points that lie inside an over-sized window around the target point and by selecting from this set the k points that are nearest to the target (see Figure 2.3). This strategy allows to avoid the overhead of building and querying a space partitioning structure (which have, respectively, a complexity of $O(n \log n)$ and $O(log\, n)$, with n being the number of points in the point cloud) and allows to find the local neighborhood of a point in a constant amount of operations.

Given a point and its neighbors, the radius of influence at that point can be computed as (half of) the average distance to its neighbors, while the normal vector can be estimated as the eigenvector corresponding to the smallest eigenvalue

of the covariance matrix created from the neighbors [Hoppe et al., 1992]. Note that, for most applications, the normal vectors need to be oriented so that they consistently point either to the inside or to the outside of the object they belong. If the viewpoint from which the points were acquired is given (which is trivially the case for the formats described in Equations 2.2 and 2.3), this can be achieved by computing the vector from the viewpoint to the target point (the view vector) and by flipping the normal vector if its dot product with the view vector is positive. In fact, even in the case of unstructured point clouds (Equation 2.1) it is sufficient that each scanned point is accompanied by the view vector to be able to compute the correct orientations for the normal vectors.

A more detailed discussion of the issues connected to feature estimation in point clouds is outside the scope of this work. In the context of this thesis, the input point clouds (regardless of their specific format) are assumed to contain consistently oriented normals (pointing towards the inside of the environment they describe) and radii of influence.

Test Datasets

In order to test the methods proposed in this thesis, a variety of point-based models of building interiors were used. The test datasets, described in detail in Appendix A, are represented either as unstructured point clouds (Equation 2.1) or as grid sets (Equation 2.3). The main reason for this is that, depending on the intended processing pipeline, one or the other format is more suitable. In particular, the methods described in Chapters 3 and 4, which make use of the visibility information of the scanned scene, expect the format described in Equation 2.3 for the input point cloud. The approach presented in Chapter 5, on the other hand, is oblivious of the viewpoints from which the scene was acquired and works directly on input point clouds expressed as in Equation 2.1.

Our test suite includes models obtained using different acquisition systems. Most real-world datasets (obtained by physically acquiring an existing environment) come from high-end terrestrial laser scanning, although some representatives of lower-quality technologies are also included.

In addition to real-world models, a number of *synthetic* datasets were generated by modeling an environment by hand using a 3D modeling software [Blender Foundation, 2016] and by then ray-casting in software the resulting model from a set of fixed positions. To simulate the noise that is present in real-world data, the result of each ray-surface intersection operation is corrupted by adding a variable offset (obtained from a Gaussian distribution with standard deviation σ_{noise}) along the ray direction. This process effectively simulates the scanning modality of time-of-flight static laser scanners and generates point clouds similar to the ones obtained with this technology.

2.2.2 Adjacency Graph of Bounding Rectangles

The point clouds output by scanning devices can contain millions of individual samples. Their distribution in the scene is a by-product of the way the view plane is covered by view rays and does not take into account the properties of the scene itself. For this reason, many geometrically simple parts of the scenes such as planar regions are represented redundantly by a large number of co-planar points. Since many man-made objects and in particular building interiors are mostly composed of piecewise-planar surfaces, it is convenient to replace the input point-based cloud with a more compact representation.

For this reason, the methods presented in this thesis perform a planar decomposition of the input point clouds, based on the use of *oriented bounding rectangles* as shape proxies for the planar regions of the scene. Each bounding rectangle lies on the plane of the region it represents and fully encloses the set of points of such region; its orientation on the plane is defined based on the global up-vector of the scene, resulting in an overall alignment of the rectangles that matches the main shape of the environment.

In addition to this, the bounding rectangles extracted are arranged in a graph structure that encodes their adjacency relations. This higher-level representation abstracts from the individual input measurements and allows for a simplified part-based analysis of the environment.

Extraction of Planar Patches

The first step in the computation of the graph-based abstraction is the segmentation of the input point cloud into a set of co-planar patches of points. This is an important problem in the context of point cloud processing and a number of well-established techniques have been proposed, including methods based on region growing [Rabbani et al., 2006; Poppinga et al., 2008] and RANSAC-based approaches [Schnabel et al., 2007], as well as more complex optimization-based pipelines [Monszpart et al., 2015; Oesau et al., 2016]. In the context of this thesis, the co-planar patches are obtained by applying a region growing approach to the pre-processed input point clouds, which (as explained in Section 2.2.1) are augmented with oriented normal vectors. In this process, described in Algorithm 1, a set of locally planar points are selected as *seeds* for the growing process, that is, each of them is considered as a potential patch and is incrementally expanded by adding compatible points from its spatial surroundings. In practice, whenever a new point is added to a patch, its nearest neighbors are retrieved and added to a queue, which represents the *frontier* of the patch. A point of the frontier is considered compatible with the patch if its distance from the plane of the patch is smaller than a threshold θ_{off} and if its normal vector does not deviate significantly

from that of the patch plane, that is, if the angle between its normal and that of the plane is smaller than θ_{ang}. The result of this algorithm is a segmentation $\mathcal{S}$ of the input point cloud into a set of planar patches $\mathcal{P}_1, \ldots, \mathcal{P}_n$.

It is important to notice that this algorithm can be applied both to an unstructured point cloud (Equation 2.1) and to a point cloud in range-grid format (Equation 2.2). From an implementation point of view, the main difference lies in the definition of the query operation for the nearest neighbors (lines 4, 26 in Algorithm 1); in particular, for range-grid point clouds it is possible to use the approximate definition described in the previous section 2.2.1. Point clouds represented as a set of individual scans (Equation 2.3) can be segmented by simply applying the algorithm to each scan separately and by merging co-planar patches belonging to different scans in a subsequent step. This can be done by greedy merging of adjacent patches based on the parameters of the underlying plane or, in case of noisier input data, by applying statistical techniques to compare the point sets of the patches to be merged [Boulch and Marlet, 2014].

Computation of Oriented Bounding Rectangles

Each patch extracted is compactly represented using its bounding rectangle. To this purpose, the first step is to compute the parameters of the plane that best approximates each patch. A plane Π in $\mathbb{R}^3$ can be represented by a vector $\mathbf{n}_\Pi$, corresponding to its normal direction, and by a point $\mathbf{p}_\Pi$ that lies on the plane itself. The direction of the normal vector $\mathbf{n}_\Pi$ can be extracted by a *Principal Component Analysis* (PCA) of the set of points of the patch (using the same approach described in Section 2.2.1 to estimate the normal vectors of a point cloud); its orientation is chosen so that it matches the one of the point normals, which in the context of this thesis are assumed to be given and pointing inside the environment (Section 2.2.1). The choice of $\mathbf{p}_\Pi$ that corresponds to the best plane in a least-squares sense is the barycenter of the points [Gross and Pfister, 2007]. Note that, to make the estimation more robust to possible outlier points resulting from the region growing, we first perform a *Least Median of Squares* (LMS) regression [Rousseeuw, 1984] on the set of points of the patch and estimate the parameters of the plane only from the resulting set of inlier points.

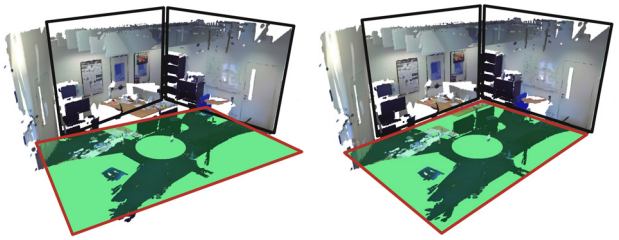
The points of the patch are then projected onto the plane and a bounding rectangle of the projections is computed. A critical point in this step is the choice of an appropriate orientation of the bounding rectangles. We have shown in the context of object detection [Mattausch et al., 2014] that simply using the two largest eigenvectors obtained by PCA can result in inconsistent orientations for the rectangles of the scene (see Figure 2.4).

To overcome this issue, every bounding rectangle is aligned to a pair $(\mathbf{v}_1, \mathbf{v}_2)$ of orthogonal 3D vectors that capture the orientation of the environment. For

Input : Point cloud P with vertex normals
Maximum plane offset θ_{off}
Maximum normal deviation θ_{ang}
Number of nearest neighbors k
Output: Segmentation $\mathcal{S} = \{\mathcal{P}_1, \ldots, \mathcal{P}_n\}$ of P into planar patches

PlanarSegmentation (P)
  $S \leftarrow$ `GenerateSeeds` (P)
  foreach $s \in S$ **do**
    $\mathcal{N} \leftarrow$ `GetNearestNeighbors` (s, P, k)
    $\Pi \leftarrow$ `FitPlane` ($\mathcal{N}$)
    $\mathcal{Q} \leftarrow \emptyset$, $\mathcal{P} \leftarrow \emptyset$
    foreach $p' \in \mathcal{N}$ **do**
      if p' not visited **then**
        $\mathbf{n}_\Pi \leftarrow$ `normal` (Π) , $\mathbf{n}' \leftarrow$ `normal` (p')
        if $\text{Cos}^{-1}(\mathbf{n}_\Pi \cdot \mathbf{n}') \leq \theta_{\text{ang}}$ $\wedge$ `dist` $(\Pi, p') \leq \theta_{\text{off}}$ **then**
          $\mathcal{Q} \leftarrow \mathcal{Q} \cup \{ p' \}$, mark p' as visited
        end
      end
    end
    while $\mathcal{Q} \neq \emptyset$ **do**
      $p' \leftarrow$ `PopItem` ($\mathcal{Q}$)
      if p' not visited **then**
        $\mathbf{n}_\Pi \leftarrow$ `normal` (Π) , $\mathbf{n}' \leftarrow$ `normal` (p')
        if $\text{Cos}^{-1}(\mathbf{n}_\Pi \cdot \mathbf{n}') \leq \theta_{\text{ang}}$ $\wedge$ `dist` $(\Pi, p') \leq \theta_{\text{off}}$ **then**
          $\mathcal{P} \leftarrow \mathcal{P} \cup \{ p' \}$
          $\Pi \leftarrow$ `UpdatePlane` (Π, p')
        else
          mark p' as not visited
        end
      end
      $\mathcal{N} \leftarrow$ `GetNearestNeighbors` (s, P, k)
      foreach $p' \in \mathcal{N}$ **do**
        if p' not visited **then**
          $\mathcal{Q} \leftarrow \mathcal{Q} \cup \{ p' \}$, mark p' as visited
        end
      end
    end
    $\mathcal{S} \leftarrow \mathcal{S} \cup \mathcal{P}$
  end
  return $\mathcal{S}$
end

Algorithm 1: Extraction of co-planar patches of points using region growing.



(a) (b)

Figure 2.4: *Consistent alignment of horizontal bounding rectangles. Orienting a horizontal rectangle based on the two largest eigenvectors of the underlying point set often results in an inconsistent orientation with respect to other parts of the scene (a). A more coherent orientation is obtained by deriving the reference system from that of a spatially close non-horizontal rectangle (b).*

a non-horizontal rectangle, this pair is defined based on the up-vector $\mathbf{v}_{\text{up}}$ of the scene (assumed, without loss of generality, to be $(0, 0, 1)$) and on the normal $\mathbf{n}_{\mathcal{P}}$ of the patch; for a horizontal rectangle, whose normal vector coincides with $\mathbf{v}_{\text{up}}$, the reference system is derived from the one of its closest non-horizontal rectangle (see Figure 2.5). As shown in Figure 2.4(b), this approach allows to recover a valid orientation of a region even in presence of large missing parts.

The result of this step is a set of rectangles $\mathcal{R} = \{r_1, \ldots, r_n\}$ that approximate the planar parts of the input scene. Although $\mathcal{R}$ constitutes a compact representation of the scene, some operations (e.g. the coverage computation mentioned in Section 4.4) require access to the lower-level segmentation into planar patches. An implementation of the procedure described in this section should therefore ensure that each rectangle $r_i \in \mathcal{R}$ can easily access the corresponding patch $\mathcal{P}_i$.

Construction of the Adjacency Graph

In order to reason about non-local properties of a scene it is often useful to consider the inter-relationships between its parts. For this reason, we arrange the bounding rectangles $\mathcal{R}$ according to their adjacency relations into an *adjacency graph* $G_{\text{adj}} = (V, E)$, as done in many shape analysis methods [Laga et al., 2013; Zheng et al., 2014]. In this graph, as shown in Figure 2.6, the set V contains one node for each bounding rectangle, while E is composed of all the pairs (v_1, v_2) for which the corresponding rectangles $v_1, v_2 \in V$ are adjacent, that is, the minimum distance between them is smaller than a threshold θ_{adj}. Under ideal sampling

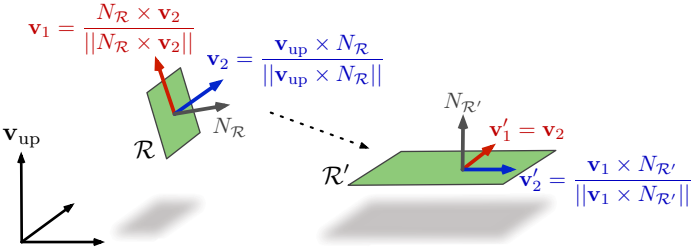
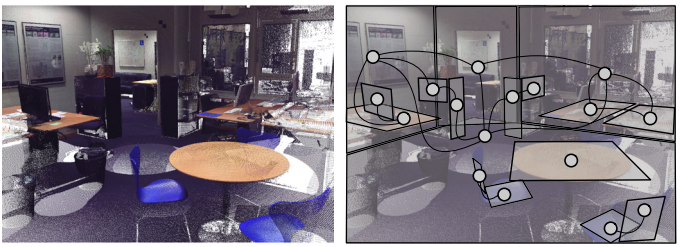


Figure 2.5: *Definition of the reference frame for the bounding rectangles. The reference frame of a non-horizontal rectangle (rectangle $\mathcal{R}$ on the left) is derived from the global up-vector $\mathbf{v}_{\text{up}}$ of the scene and helps define a consistent orientation for the nearby horizontal rectangles (see $\mathcal{R}'$, on the right).*

conditions (e.g. uniform sampling of all scene surfaces, absence of noise, no viewpoint occlusions), the value of θ_{adj} could be simply set to a value slightly larger than the sample distance. However, it is often necessary to increase the adjacency threshold to cope with the defects of real-world point clouds. In practice, θ_{adj} can be treated as an input parameter for a processing pipeline based on such an adjacency graph and adjusted based on the expected or estimated level of noise in the input data.

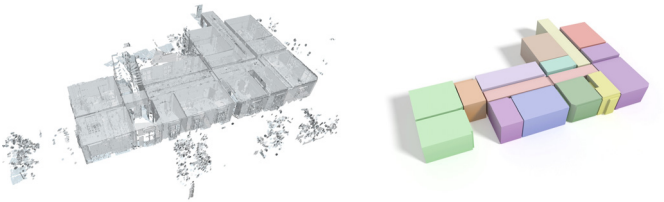


(a) (b)

Figure 2.6: *Contact graph for the point cloud of an indoor environment. An input point cloud (a) is associated with a graph (b) in which the nodes correspond to parts of the scene (represented by bounding rectangles) and the edges correspond to adjacencies between the parts. Note that in (b) some parts have been removed for clarity of illustration.*

2.3 Output Data

The space inside an indoor environment is bounded and completely separated from the outside by its main architectural surfaces (i.e., walls, ceilings, floor). For this reason, a typical way to describe building interiors is by means of a closed 3D polyhedron, whose boundary represents the interface surface between inner and outer space. In practice, a watertight polygonal mesh is used to represent this polyhedron [Turner and Zakhor, 2013; Oesau et al., 2014]. A similar representation can be used for an individual room inside a large environment. Since this thesis targets multi-room environments, our output reconstruction is represented as a set of watertight meshes (see Figure 2.7), one for each room in the environment. Note that, using this representation, a thick wall separating two rooms appears as two separate sheets, each in a different room polyhedron, and that the solid space between these corresponds to empty space between the rooms.

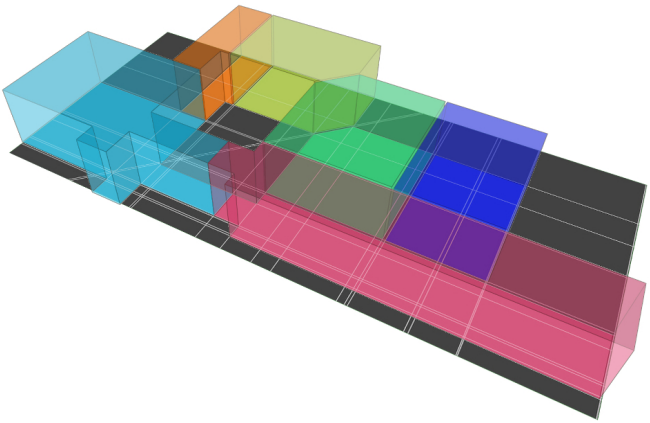


(a) (b)

Figure 2.7: *Example of input point clouds of building interiors (a) and of the reconstructed room polyhedra (b).*

C H A P T E R 3

2.5D MODELING OF MULTI-ROOM ENVIRONMENTS



This chapter describes the first research contribution of this thesis, which allows to robustly model multi-room building interiors that have a 2.5D structure. The method follows the general pipeline outlined in Chapter 2 and uses the 2.5D prior to achieve robustness to the defects of real-world data while keeping the complexity of the computation low. For this reason, it represents a practical approach that can be used to model a significant array of real-world environments while still being fast and simple to implement. To cope with the general case of interiors with arbitrary wall orientations, the more general approach of Chapter 4 should be used, at the cost of an increased complexity and of a higher computational cost.

3.1 Motivation and Background

A large number of modern buildings (e.g. blocks of apartments, office buildings) exhibit a regular architectural structure, with vertical walls and horizontal floors and ceilings. Such buildings are said to have a *2.5D structure*, meaning that their three-dimensional shape can be obtained by vertical extrusion of a two-dimensional entity, that is, the floorplan of the building.

Assuming a 2.5D structure provides some useful priors on the shape of the architectural structures to be reconstructed, which greatly simplifies the modeling process: since all wall structures are assumed to be vertical, the analysis of a building can be performed by considering the projection of the input model onto the horizontal plane of the floor. In particular, the input 3D points corresponding to the same vertical wall should appear on this projection as thin clusters of almost colinear 2D points; this allows for an early pruning of groups of points that do not exhibit a clear dominant direction. Moreover, the main wall directions can be extracted by detecting 2D lines in the projection, using well-established methods such as the *Hough transform* or the *mean-shift* clustering algorithm. In addition to this, the presence of horizontal floors and ceilings allows to consistently define the notion of *height* of the environment (i.e. the distance between the floor and the ceiling) and to use it to support the modeling process.

State-of-the-art The 2.5D assumption effectively allows to cast the modeling problem in terms of the extraction of the floorplan of the environment and is therefore a strong prior in itself. In spite of this, a large number of methods further simplify the task by relying on the even more restrictive Manhattan-World assumption, which additionally imposes that walls are either orthogonal or parallel to each other. This implies that walls can have only one of two dominant orientations, which can be found by first detecting the largest peak in the distribution of wall orientations and by then looking for the second largest peak near the

direction perpendicular to the first one [Okorn et al., 2010; Budroni and Böhm, 2010]. The same assumption is also exploited by methods that do not perform a floorplan-based modeling, for instance to simultaneously detect both large-scale and small-scale architectural structures [Sanchez and Zakhor, 2012] or to formulate the reconstruction as the labeling of an axis-aligned voxelization of the environment [Furukawa et al., 2009; Turner and Zakhor, 2013]. A voxel-based representation is also used by Adan et al. [Adan et al., 2013], who use it to guide the extraction of the wall structures. More interestingly, their voxelization is used to detect doorways and windows in an occlusion-aware manner.

Since these approaches are heavily limited in the array of environments that they can capture, more modern pipelines typically only rely on the 2.5D prior. Some approaches focus entirely on the extraction of the floorplan, arguing that a full 3D model can be generated from the floorplan itself [Turner and Zakhor, 2012], while others also extrude it vertically according to the recovered height of the environment [Cabral and Furukawa, 2014]. Oesau et al. [Oesau et al., 2014] go one step further and generate a 3D space partitioning by stacking lines detected in the vertical projection of the input model, then label the volumetric cells into inside/outside using a visibility-driven energy minimization.

Although more flexible than the approaches based on Manhattan-World, such methods generally suffer from two main drawbacks. First, most of them disregard the problem of occlusions in the input data, assuming either that the scene is almost completely visible or that parts of the building occluded from one view are available from another viewpoint. In fact, the few methods that achieve some degree of robustness to occlusions, either as a side-effect of the specific reconstruction algorithm used [Furukawa et al., 2009] or by explicit recovery of the occluded regions [Adan et al., 2013], rely on the orthogonality of the wall structures. Second, these methods perform a sheer recovery of the boundaries of the environment and do not extract any semantic information about the subdivision into different rooms, which (as explained in Chapter 1) is of fundamental importance in many real-world application scenarios, such as room asset planning and management or energy efficiency simulation.

Contribution The work described in this chapter – which was presented as a conference contribution [Mura et al., 2013] and as a journal article [Mura et al., 2014] – aims at filling the research gap with respect to the two points mentioned above (i.e. robustness to occlusions and recovery of room subdivision) while explicitly targeting the reconstruction of the architectural structures of the environment. In particular, we propose a robust modeling pipeline that is capable of coping with heavy occlusions and missing data and of automatically recognizing different rooms as separate parts of the environment. The approach is based on

the 2.5D prior, but does not impose other constraints on the mutual orientations of the walls, thus lifting the much more restrictive Manhattan-World assumption used by many state-of-the-art methods.

3.2 Method Overview

Coherently with the outline provided in Section 2.1, the method takes as input an adjacency graph G_{adj} of bounding rectangles that describes the scene to be modeled, together with the associated point clouds. Since the pipeline makes use of the visibility information during the detection of permanent components and in the room reconstruction stage, we arrange the point clouds according to the format described in Equation 2.3. The output model – a set of watertight meshes, one for each room in the environment – is obtained by applying a pipeline that follows a scheme similar to the one outlined in Section 2.1. The three steps of the pipeline, visually summarized in Figure 3.1, are shortly introduced in this section; a more detailed description is given in the remainder of this chapter.

Occlusion-aware detection of permanent components Vertical rectangles that are potential wall structures are selected from the input adjacency graph. Taking into account the per-scan visibility information, occluding rectangles are then projected onto the potential wall rectangles to recover their actual (unoccluded) vertical extent and get a robust indicator of whether they are genuine wall segments, pruning those which are more likely to be clutter.

Construction of floorplan-based space partitioning This step is performed entirely in 2D, on the plane of the ground. First, the projections of the candidate permanent components are clustered to get a smaller number of good representative lines for walls. Then, a cell complex is built from the intersections of the representative lines and its edges are weighted according to the likelihood of being genuine walls.

Room detection and reconstruction Diffusion distances are computed on the cell-graph of the complex and are used to drive an iterative clustering of the cells that extracts the boundaries of the individual rooms. For each room, the final geometry is obtained by robustly fitting planes to the points that lie along the room boundary and by intersecting the reconstructed wall planes with the planes of the floor and ceiling.

Differently from the general pipeline shown in Section 2.2.2, the last two steps are merged into one, as the room reconstruction technique used in this approach detects one room after another and at the same time reconstructs its 2D floorplan.

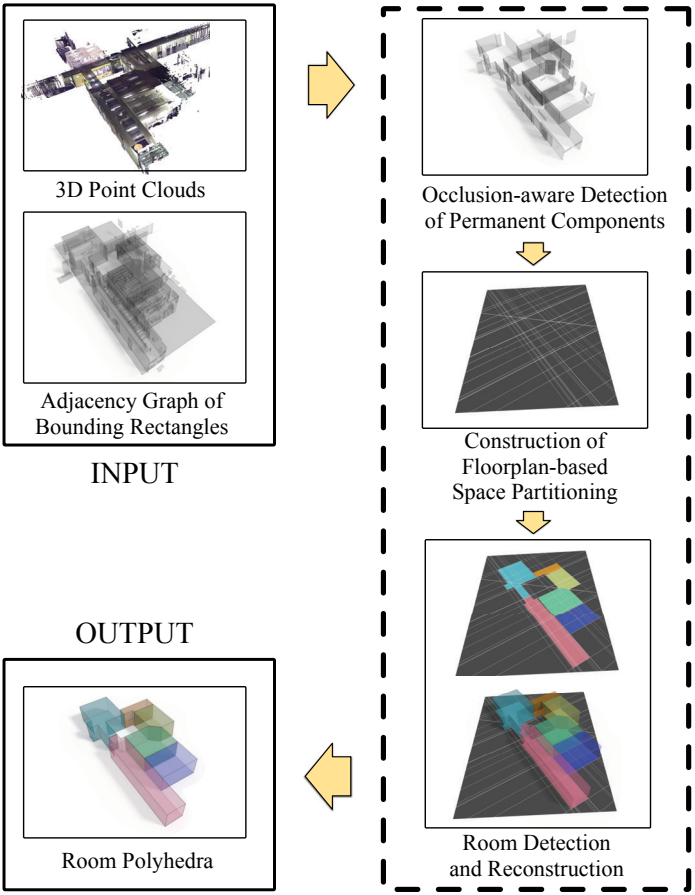


Figure 3.1: *Overview of our 2.5D reconstruction pipeline. With respect to the general outline in Section 2.1, the room detection and reconstruction steps are merged into one.*

3.3 Occlusion-aware Detection of Permanent Components

The adjacency graph G_{adj} corresponds to a general piece-wise planar representation of the environment considered; this also includes furniture and other non-permanent elements that do not belong to the architectural shape of the environment. Since we explicitly target 2.5D environments with vertical walls, we select as candidates for permanent components only the rectangles for which $|\mathbf{n} \cdot \mathbf{v}_{\text{up}}| < \epsilon$, where $\mathbf{n}$ denotes the normal vector of a rectangle of G_{adj} and $\mathbf{v}_{\text{up}}$ is the global up-vector of the scene. This simple criterion effectively rules out flat structures like tables. We also discard small cluttering rectangles for which the horizontal extent is smaller than a threshold θ_{hor} (set to $40cm$ in our experiments).

This process does not yet exclude large vertical cluttering elements such as large cabinets or bookshelves from the potential wall patches. We therefore perform a further pruning, following the intuition that genuine wall structures must cover a vertical extent that is approximately equal to the distance between the floor and the ceiling. Checking for this condition in real-world inputs such as 3D scans of offices or apartments is problematic, as obstacles located between the camera and the walls can severely limit the amount of structure visible. Taking more scans from additional viewpoints can only partially solve this problem and it cannot be considered a viable solution, especially for data sets that originate from static 3D laser range scanning.

For this purpose, we employ a lightweight visibility test to estimate the expected *unoccluded* vertical extent of each candidate permanent rectangle r. In our technique, an occlusion happens if a bounding rectangle r and an occluder rectangle r_{occl} overlap when seen from the scan position from which they were taken. We construct the *infinite shadow volume* [Crow, 1977] of each r_{occl} by casting rays from the scan position through its four corners. We then compute the intersection of this shadow volume with the plane of r. Finally, the quadrilateral resulting from the intersection of the shadow volume is tested for overlap with r. In practice, we scale each rectangle by a factor of 1.05 to ensure intersection between the shadow volume of r_{occl} and a potentially occluded rectangle r. The process is illustrated in Figure 3.2. If an occlusion between r and r_{occl} occurs, we consider the vertical extent of the projection of r_{occl} onto r and add it to the vertical extent of r itself. By repeating this check for every r_{occl}, we obtain the combined height h of r. We then prune r from the candidate list based on the following condition: $h \leq (1 - \eta) \cdot h_{\text{rooms}}$. Here η is a small number (set to 0.05 in our tests) and h_{rooms} is the distance between the floor and the ceiling. An accurate measure for h_{rooms} is obtained as a byproduct of the robust fitting of wall and ceiling planes described in Section 3.5.4.

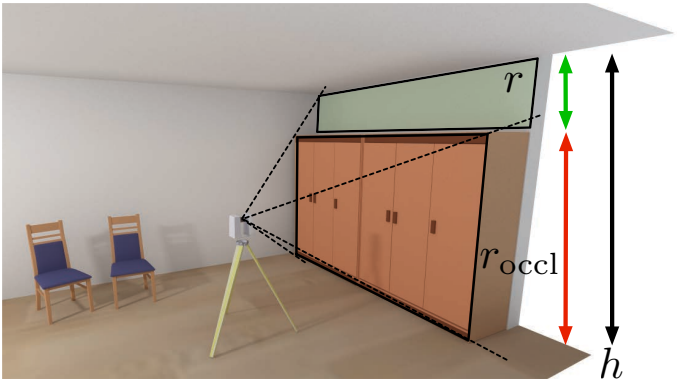


Figure 3.2: *Computation of the unoccluded vertical extent of a rectangle. An occluding bounding rectangle r_{occl} is projected onto the plane of a candidate permanent rectangle r to recover its unoccluded vertical extent h.*

Repeating this check for each vertical rectangle that satisfies the aforementioned condition yields a pruned list of actual candidate permanent components that are likely to belong to wall structures. The use of the unoccluded vertical extent significantly improves the selection of candidate walls in cluttered environments and all subsequent steps of the algorithm benefit from this. The approach works well even though it is based on an approximation of the scan visibility problem. On the other hand, a more sophisticated analysis would be inadequate for this task due to the imperfect nature of real world input data, which contain large holes and missing parts.

3.4 Construction of Floorplan-based Space Partitioning

The candidate permanent rectangles detected in the previous stage provide some evidence of the likely location of the main wall structures of the environment. As outlined in Section 2.1, these can be used to build a space partitioning structure that describes the space surrounding the input scans as a set of convex sub-regions. Thanks to the 2.5D hypothesis used in this method, the construction of this structure can be formulated entirely in terms of 2D computations on the floorplan of the environment, leading to a simplified process compared to a full-3D approach.

3.4.1 Computing Representative Lines

The candidate permanent rectangles detected in the previous step are projected vertically onto the plane of the floor to obtain a set of *2D line segments*, which are then clustered using mean-shift [Comaniciu and Meer, 2002]. Such an approach, employed in a similar manner also in other state-of-the-art approaches [Oesau et al., 2014], aims at merging almost co-linear segments into a reduced set of *representative lines* that describe the main orientations of the walls in the environment. A first clustering yields the main directions of the walls; for each direction obtained we then perform a 1D mean-shift clustering which identifies all possible offsets of parallel wall segments of that orientation. This way we extract a set of clusters of line segments $\mathcal{C} = \{\mathcal{C}_0, \ldots, \mathcal{C}_n\}$. Each cluster $\mathcal{C}_k$ corresponds to a particular wall structure and is associated with a representative line l_k. The list of representative lines $\mathcal{L} = \{l_0, \ldots, l_n\}$, together with the associated clusters of line segments $\mathcal{C}$, are used to construct the space partitioning structure and to compute the weights of its edges.

3.4.2 Cell Complex Construction

The representative lines $\mathcal{L}$ induce a partition of the plane that models the floorplan of the processed indoor environment. The resulting data structure is a standard *2D cell complex*, also known in the literature as an *arrangement of lines* [Edelsbrunner et al., 1986]. It is worth mentioning that, during the construction process, each edge of the complex is associated to the representative line l_k from which the edge originated, together with the corresponding cluster of 2D line segments $\mathcal{C}_k$.

Using the infinite representative lines to construct the complex contributes significantly to the robustness to occlusions of the whole method: even if part of a wall is missing due to an occlusion, the 2D region around the missing part will still be split along the actual wall, thus generating 2D cells whose boundaries follow the separation induced by the unoccluded wall structure. Since in our pipeline the shape of a room is obtained by aggregation of a set of 2D cells of the complex (see Section 3.5.2), the room reconstruction algorithm can still extract correct room shapes that adhere to the actual wall boundaries.

The edges of the cell complex are weighted in a way that allows for the subsequent extraction of the individual room shapes. In particular, given an edge $\mathbf{e}_{ij}$ between two faces f_i and f_j of the cell complex, we assign it a weight w_{ij} that corresponds to its likelihood of corresponding to a real wall structure. To do so, we consider all contributing candidate walls (i.e., all line segments of the cluster $\mathcal{C}_k$ associated to $\mathbf{e}_{ij}$) and project them onto $\mathbf{e}_{ij}$ itself. Let us denote with $\mathrm{cov}(\mathbf{e}_{ij})$ the fraction of the extent of $\mathbf{e}_{ij}$ that is covered by such projections. The weight w_{ij} is

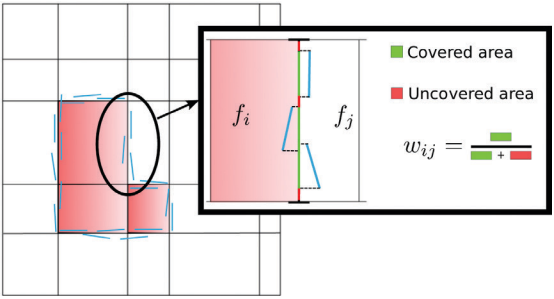


Figure 3.3: *Computation of the coverage of an edge of the 2D cell complex. The 2D line segments of candidate walls (in light blue) are first projected onto the edge, then the ratio of the occupied length to the total length is assigned as a weight to the segment.*

then defined as follows:

$$w_{ij} = \frac{\text{cov}(\mathbf{e}_{ij})}{\text{length}(\mathbf{e}_{ij})} \tag{3.1}$$

The computation of the weight w_{ij} of the edge between two faces f_i and f_j is shown in Figure 3.3.

Note that we also keep a so called *infinity face* f_∞, which corresponds to the outside and has an edge incident to each face on the boundary of the cell complex. This infinity face plays an important role in the termination criterion during the subsequent iterative clustering.

3.5 Room Detection and Reconstruction

The cell complex computed in the previous step partitions the plane of the floorplan into a set of faces. In this step, the shapes of the rooms of the environment are reconstructed first in 2D, as sub-regions of the floorplan corresponding to specific clusters of faces, then in 3D, by robustly extruding the 2D room shapes into 3D polyhedra. This procedure yields the final room models.

3.5.1 Diffusion Embedding

Given the cell complex representing the environment and the coverage weights w_{ij} of the edges between neighboring faces, it is possible to establish a global affinity measure for all pairs of faces. We use the coverage weights to derive a

sparse affinity matrix $\mathbf{L}$, with entries $\mathbf{L}_{ij}$ defined as follows:

$$\mathbf{L}_{ij} = \begin{cases} e^{-w_{ij}/\sigma} & \text{if } i \neq j \ \wedge \ f_i, f_j \text{ are adjacent,} \\ 1 & \text{if } i = j, \\ 0 & \text{otherwise} \end{cases} \tag{3.2}$$

where σ is a scaling parameter.

From matrix $\mathbf{L}$ we define a Markov probability transition matrix as $\mathbf{M} = \mathbf{D}^{-1}\mathbf{L}$, with $\mathbf{D} = \text{diag}(\sum_{j=1}^{n} \mathbf{L}_{ij})$. Each element $\mathbf{M}_{ij}$ can be seen as a local affinity value between faces f_i and f_j, as it is defined by considering only direct connectivity between faces. We propagate these local affinities by means of *diffusion maps* [Coifman and Lafon, 2006], which are known to be robust against noise [Lipman et al., 2010] and therefore well suited for our task. The diffusion map Φ embeds the faces in a multidimensional Euclidean space. Given a face f_i, its corresponding coordinate in the embedding space is

$$\Phi(f_i) = (\lambda_1^t \phi_1(f_i), \lambda_2^t \phi_2(f_i), \ldots, \lambda_m^t \phi_m(f_i)) \ , \tag{3.3}$$

where λ_k and ϕ_k are the k-th eigenvalue and eigenvector of $\mathbf{M}$ respectively. Two parameters control this diffusion process: the *diffusion time* t (a measure of smoothness that determines how much the affinities are propagated) and the number m of eigenvectors of $\mathbf{M}$ used in the diffusion map (corresponding to the dimensionality of the embedding). The Euclidean distance in this multidimensional space is a measure of dissimilarity between the faces of the cell complex. In other words, if $||\Phi(f_i) - \Phi(f_j)||_2$ is low then the faces f_i and f_j are likely to be in the same room.

This process has a physical interpretation as heat diffusion: it can be seen as a measure of how much heat can flow from f_i to f_j in a given diffusion time. This can be clearly understood by examining Figure 3.4, where the diffusion distances from a reference face to all the other faces are visualized as a heatmap. The heat diffuses quickly to the faces that belong to the same room, as they are close to the reference face in the Euclidean embedding. On the other hand, heat propagates slower to outer faces (indicated by a green color) and to faces that belong to other rooms, which in the embedding are far from the reference face (indicated by a blue/cyan color).

Visualizing the position of the faces in their embedding space is helpful towards understanding the importance of the diffusion process. However, this is not directly possible when $m > 3$. To give an intuition of how the Euclidean embedding is shaped, separate 3D projections of the embedded points could be considered, one at a time. In Figure 3.5 we show the projection of a diffusion embedding generated in our tests onto three of its dimensions, selected manually for the sake of illustration. Each object in the plot corresponds to a face of the 2D

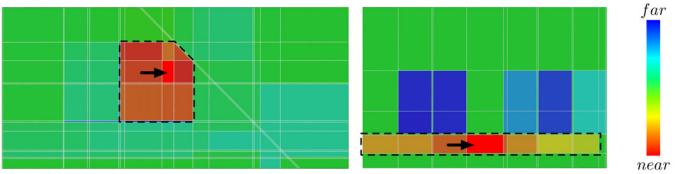


Figure 3.4: *Heatmap visualization of the diffusion distances for two different datasets. In both cases, the distances from a reference face (marked by the arrow) to all other faces in the complex are shown. Note how the cells in the same room (highlighted by the black dotted line) as the source are closer in terms of diffusion distances than the cells that belong to the outer space or to other rooms (shown in green and blue).*

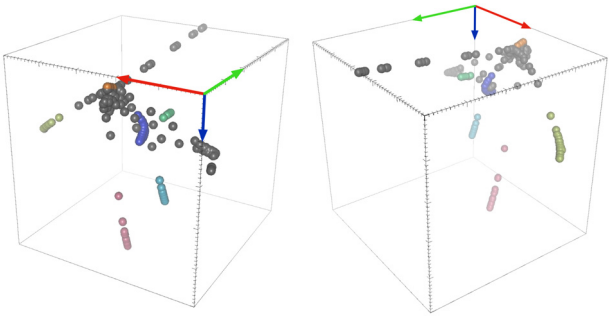


Figure 3.5: *Scatterplots showing two rotated views of a 3D projection of the Euclidean embedding. Each point in the plot represents a face of the 2D cell complex and is colored according to its final room assignment. The plot refers to the dataset 'Office 2' and the three dimensions have been manually selected for illustration purposes only.*

cell complex and is colored according to the room it is assigned as a result of the segmentation stage. While this offers only a partial view of the embedding space, it can be still understood how faces belonging to the same room are close in the diffusion embedding. The remaining $m - 3$ coordinates, not shown in this plot, make the separation between the faces belonging to different rooms fully accurate.

The properties of the embedding can be analyzed indirectly by considering the matrix of pairwise distances that results from the diffusion process. In this matrix, the i-th row contains the diffusion distances from face f_i to all other faces in the

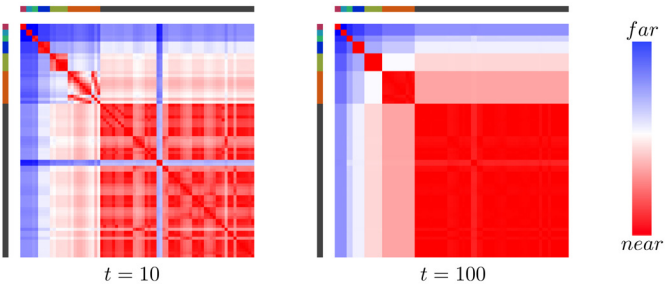


Figure 3.6: *Color-coded visualization of the pairwise distances for the cell complex of dataset 'Office 3' using two different values of the diffusion time t. The matrices have been rearranged so that rows corresponding to faces in a same cluster are adjacent; the colored bars to the sides of the matrices highlight the correspondence with the detected rooms. Each red square on the diagonal corresponds to the pairwise distances between faces in the same room cluster. Note how faces within the same cluster are close, and how increasing the diffusion time removes noise and emphasizes the structure of the clusters.*

complex. In Figure 3.6 a color-coded version of the distance matrix is shown. Note that the order of the rows has been modified so that the rows corresponding to faces in a same room appear one after another. The matrix has a block structure, with the red squares along the diagonal denoting the distances between faces of a same room. The first six squares correspond to rooms, while the last square represents the distances between the outer faces. It is worth noticing how the squares correspond to uniformly low distance values, which confirms that faces within a same room are close in terms of diffusion distances. The diffusion process corrects many errors due to missing data, clutter, and wrong candidate walls, emphasizing the room separations.

These arguments show that the diffusion formulation is very effective in highlighting the similarities between faces within the same room. We also noticed that the process is only slightly influenced by variations in the parameter settings. All the results shown in this paper have been obtained using $t = 40$ and $\sigma = 0.0625$. The maximum dimensionality of the embedding, which corresponds to the maximum number of eigenvalues that can be used in the diffusion maps, is bounded from above by the number of faces n_f of the complex. In practice, using such a high number of eigenvalues does not lead to better results and only increases the computation time. Since using an embedding of dimensionality 80 proved sufficient to correctly capture even the most complex environments in our test suite, we have conservatively set $m = \min(n_f, 80)$.

3.5.2 Iterative Clustering

The embedding distances between the faces of the 2D complex allow to apply a clustering algorithm to the set of faces, grouping them into a number of clusters, each corresponding to a single room of the environment.

We exploit the fact that the location at which each scan was acquired is known (see Section 3.2) to *automatically* extract the correct number of rooms. To this purpose, we perform an iterative clustering algorithm, described in pseudo-code in Algorithm 2 and explained visually in Figure 3.7. Each iteration executes a binary version of the k-medoids clustering [Kaufman and Rousseeuw, 1987], in which $k = 2$ (line 5 of Algorithm 2). The k-medoids algorithm works by alternating between two steps: computation of the cluster centers (the *medoids*) and assignment of the data points to the best cluster. Differently from the more commonly used k-means algorithm, the cluster centers are restricted to be items of the input dataset.

In particular, given two clusters $\mathcal{K}_0$ and $\mathcal{K}_1$, each medoid k_i is updated according to the rule $k_i = \text{argmin}_{\mathcal{K}_i} \sum_{k_j \in \mathcal{K}_i} ||k_i - k_j||^2$. At each step, the k-medoids is initialized by setting as initial medoids the two faces that are farthest away from each other in terms of diffusion distance. Of the two clusters that result from each partitioning step, one always corresponds to a new single room, denoted as $\mathcal{K}_R$ in Algorithm 2. For example, the split labeled 1 in Figure 3.7 extracts the room shown in red. For environments with genus ≥ 1 (such as 'Office 2', shown in Figure3.9), $\mathcal{K}_R$ can correspond to a *hole* (i.e. a set of outside faces disjoint from f_∞); since we assume that each room is scanned from at least one viewpoint (Section 3.2), we discard $\mathcal{K}_R$ if it does not include a face containing a viewpoint. The second cluster extracted by the k-medoids contains the faces not yet labeled (shown in gray in Figure 3.7 and denoted as $\mathcal{K}_\infty$ in Algorithm 2) and always includes the face f_∞. This behavior is linked to the properties of the diffusion embedding, which are analyzed in detail in the next paragraph. As the algorithm proceeds to iteratively cluster $\mathcal{K}_\infty$ (line 5 in Algorithm 2), it creates a split sequence like the one shown in the bottom of Figure 3.7.

The iterative partitioning has to be stopped when all rooms have been detected. To do so, we exploit the viewpoint information that comes with the input point clouds. In particular, we can assert that the faces that contain a scan position ($v \in \mathcal{V}$ in Algorithm 2) are certainly inside a room. Conversely, we can assume that each room was scanned from at least one location inside it (see Section 3.2). Hence, to check if $\mathcal{K}_\infty$ contains yet unlabeled rooms, it is sufficient to check whether it contains a face $v \in \mathcal{V}$ (line 7 in Algorithm 2). Otherwise we can conclude that all rooms have been detected, in which case the partitioning process is terminated. In Figure 3.7, this termination criterion is met after the last remaining room has been extracted in the split labeled 10.

Input : Set $\mathcal{F}$ of the faces in the Euclidean embedding
Set $\mathcal{V}$ of the faces containing viewpoints
Output: Set $\mathcal{K}$ of clusters defining the rooms

ClusterRooms ($\mathcal{F}, \mathcal{V}$)
  $\mathcal{K} \leftarrow \emptyset$
  $\mathcal{K}_\infty \leftarrow \mathcal{F}$
  repeat
    // create clusters $\mathcal{K}_R$ (new room) and
    // $\mathcal{K}_\infty$ (unlabeled faces containing f_∞)
    ($\mathcal{K}_R, \mathcal{K}_\infty$) $\leftarrow$ KMedoids ($\mathcal{K}_\infty$)
    // update set of room clusters
    $\mathcal{K} \leftarrow \mathcal{K} \cup \mathcal{K}_R$
  until $\exists v \in \mathcal{V} : v \in \mathcal{K}_\infty$
  return { $\mathcal{K}$ }
end

Algorithm 2: Iterative clustering algorithm for the extraction of room clusters.

The advantage of this method over other room clustering alternatives is that the subdivision is stopped when all rooms have been extracted, without the need of specifying the target number of rooms in advance. In fact, this approach effectively integrates the detection of the rooms into the reconstruction process.

Analysis of the diffusion-based clustering The clustering algorithm proposed exhibits the systematic behavior of extracting a single room in each iteration. Two factors are responsible for this behavior: the specific properties of the diffusion embedding and the characteristics of the k-medoids clustering. The two aspects are examined in detail in this section.

A relevant property of the embedding is that, for the set of faces of any given room, the faces representing the outside (and including f_∞) are closer than the faces inside any other room. Evidence for this is provided by the color-coded visualization of Figure 3.4: the faces of the room where the source face is located (indicated by the arrow) are closer to the outer faces than to the faces in other rooms. This property is even more evident if we consider the visualization of the pairwise distance matrix in Figure 3.6.

This happens because the faces that correspond to the inside and those that represent the outside are in contact along the whole perimeter wall of the building, which corresponds to many edges of the complex. Using the heat diffusion analogy, these edges represent the only barrier that prevents the heat from diffus-

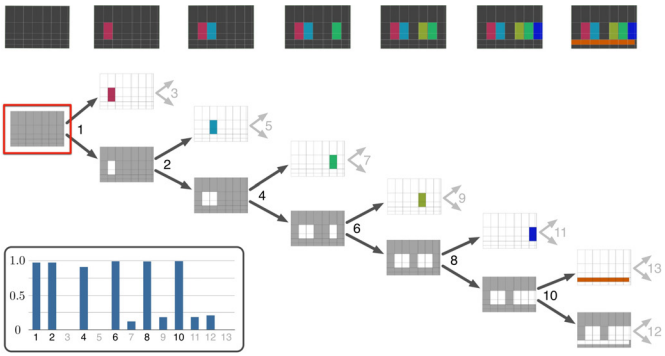


Figure 3.7: *Illustration of the iterative partitioning process. The input cell complex (framed in red) is iteratively divided into two clusters, generating an (unbalanced) binary tree of splits. At each split (identified by a number), the set of input faces is partitioned into a room cluster and a set of unlabeled faces (shown in gray). The partial partitioning corresponding to each split step can be seen in the top part of the figure. Note that the only splits that are actually performed in our algorithm are the ones numbered in black, while the gray numbers denote hypothetical splits. A measure of the quality of each split (including the hypothetical ones) in the bar plot in the inset.*

ing from the inside to the outside. On the other hand, any two individual rooms do not touch directly, but are separated either by empty space or by a thick wall; in either case, the heat can not diffuse directly from one room to another, but must travel across (possibly many) outer faces. For this reason, as long as the candidate walls are sufficiently solid, the faces of any room are much closer in the diffusion embedding to the outside faces than to the faces of any other room.

During each iteration of the clustering process, the first assignment of faces to the medoids will create two *unbalanced* clusters, with medoids k_1 and k_2. This is because the medoid that is closer to f_∞ (let us assume this is k_2) will become the pivot for most of the faces of the complex. As a consequence, k_2 will move towards f_∞. Since every room is closer to the outside than to any other room, every face in the complex will be assigned to the cluster of k_2, with the exception of the faces that belong to the room containing k_1.

The use of the k-medoids algorithm, which requires the cluster centers to be faces of the complex, further increases the robustness and the stability of this partitioning process. This is because a cluster center will either be a face inside a room (and hence the cluster is strongly bound to this particular room) or a face close to f_∞ (possibly, f_∞ itself). We have also experimented with an alternative

formulation based on applying the more traditional k-means clustering, where cluster centers can correspond to arbitrary positions in the embedded space. We discovered that k-means is much more dependent on the diffusion parameters (in particular, on the diffusion time t) and requires considerable adjustment of their values to yield correct results.

To further prove the effectiveness of our algorithm we have evaluated the quality of each split *a posteriori*. Let $\mathcal{B} = \{\mathbf{e}_1, \ldots, \mathbf{e}_n\}$ be the set of edges that separate the two components extracted in a single binary split. We define the *split quality* q_{split} of $\mathcal{B}$ as follows:

$$q_{\text{split}}(\mathcal{B}) = \frac{\sum_{i=1}^{n} w(\mathbf{e}_i) \cdot \text{length}(\mathbf{e}_i)}{\sum_{i=1}^{n} \text{length}(\mathbf{e}_i)}, \tag{3.4}$$

where $w(\mathbf{e})$ denotes the coverage of $\mathbf{e}$ due to candidate wall segments and follows the definition of Equation 3.1. This quality function measures how solid the boundary between two adjacent clusters is in the range $[0..1]$, where 1 corresponds to a split along a perfectly solid wall. The edge lengths act here as weights, ensuring that each edge contributes to the split cost proportionally to its importance.

The quality of the splits generated in the sequence of Figure 3.7 is shown as a bar plot in the inset of the same picture. The splits actually performed by our algorithm are identified by the black numbers. According to the quality measure, each of those splits generates a boundary that corresponds to an actual wall with high confidence. On the other hand, forcing a further (hypothetical) split when the algorithm would normally terminate would yield a boundary with clearly low quality. These hypothetical splits are denoted by light gray numbers. Since we are taking the viewpoints into account to check for the proper termination depth, no threshold has to be set to avoid such bad splits.

3.5.3 Post-processing

When the input environment contains long corridors, the diffusion process might fail to propagate the affinity values between the distant faces of such a structure, resulting in corridors being incorrectly split into several separate clusters (see Figure 3.8). This issue could be solved by increasing the diffusion time t in Equation 3.3, which however would require an interactive and adaptive tuning of this parameter. As an alternative, we propose a simple post-processing technique based on explicitly checking the goodness of the boundary between adjacent clusters. This is based on the fact that incorrect splits can be easily and robustly detected since they do not correspond to an actual wall segment.

The first step of the post-processing is to discover the adjacencies between the detected clusters of 2D cells, that is, to detect the pairs of clusters whose boundaries touch in at least one edge. For each such pair, we compute the (possibly

(a) (b)

Figure 3.8: *Post-processing step to correct room over-segmentation. Large rooms with an elongated shape such as corridors may be incorrectly split into multiple clusters (a). By applying our robust post-processing step, we are able to detect these cases and recover the correct shape of any over-segmented room (b).*

multiple) connected components of the shared boundary $\mathcal{B}$, then evaluate the split quality $q_{\text{split}}(\mathcal{B})$ as defined in Equation 3.4.

Two adjacent clusters are then split if for each of their boundary segments q_{split} is lower than a threshold, that we have set to 0.5 in our experiments. This method robustly and correctly merges connected structures like the corridor in Figure 3.8. Note that, in practice, the choice of this threshold did not require any fine tuning, as we have observed a clear-cut distinction between erroneous and correct splits. In all our experiments, fake boundaries always had $q_{\text{split}} < 0.22$, while real ones always had $q_{\text{split}} > 0.97$ (as shown by the plot of Figure 3.7).

3.5.4 Final Reconstruction

The final 3D model of the environment is obtained by extracting a 3D polyhedron from each detected room cluster. To do so, the boundary edges of each cluster are extracted, merging adjacent edges that are colinear. Then, the full 3D extent of the walls is recovered. Instead of simply extruding the 2D boundary edges vertically, we apply a different approach based on robust statistics to obtain a more accurate estimation of the wall parameters.

For each edge in the boundary of a cluster, we access the 2D line segments $\mathcal{C}_k$ associated to the edge (see Section 3.4.1) and we select the points of the corresponding rectangles. A robust plane fitting algorithm is then applied to these points to extract the final wall planes. The fitting is based on the *Iteratively Reweighted Least Squares* (IRLS) [Huber and Ronchetti, 2009], which is based on solving a sequence of weighted least squares problems until convergence. Robustness is achieved by using a suitable weight function, chosen so that outliers (which correspond to large residuals) have a reduced influence in the estimation. In our experiments, we used the function $w(x) = 1/|x|$ as weight function for the IRLS, which corresponds to minimizing the L_1 norm of the residuals.

Model	#Perm. rect.	#Rooms	Time
Room 1	21	1	6.7s
Room 2	25	1	8.6s
Office 1	58	2	10.5s
Office 2	113	6	9.5s
Office 3	134	6	34.4s
Synth 1	52	4	19.4s
Synth 2	60	3	18.6s
Synth 3	202	11	85.3s

Table 3.1: *Relevant statistics and running times of the reconstruction. For each input model, we show the number of candidate permanent rectangles detected (col.* Perm. rect.*), the number of rooms extracted (col.* Rooms*) and the processing time for the whole reconstruction process (col.* Time*)*

We use a similar fitting procedure for reconstructing the floor and ceiling planes. Since we assume that floor and ceiling are planar and orthogonal to the up-vector, we find the two horizontal rectangles r_{floor} and r_{ceil} with respectively minimum and maximum height. To increase the accuracy and robustness of the estimation, we propose the following strategy to fit the final planes. Given r_{floor} (respectively r_{ceil}), we take the horizontal rectangles whose distance from r_{floor} (and r_{ceil}) is less than a threshold and use their points as support set for an IRLS fit. Note that throughout this process for practical purposes we only consider rectangles with a diagonal larger than $50cm$.

Given the fitted planes of the walls and of the floor and the ceiling, the polygons of the final polyhedra are obtained by intersecting pairs of adjacent wall planes with the floor and ceiling planes.

3.6 Results

We evaluated the effectiveness of our approach on 8 models of building interiors. Of these, 5 correspond to real-world environments (‘Room 1’, ‘Room 2’, ‘Office 1’, ‘Office 2’, ‘Office 3’), while the remaining 3 (‘Synth 1’, ‘Synth 2’, ‘Synth 3’) are synthetic datasets, created by a 3D artist and sampled in software (see Section 2.2.1 for a description of this process). Detailed information on the input datasets are given in Appendix A. Relevant statistics about the reconstruction process are listed in Table 3.1.

Implementation We implemented a software prototype of the proposed pipeline in C++, using OpenMP to parallelize some processing stages, including

the weighting of the cell complex. All tests were run on an Intel Xeon E5-2670 2.6GHz processor. As shown in Table 3.1, the processing times vary from $6.7s$ for the smallest model ('Room 1') to less than $90s$ for the largest dataset ('Synth 3'), which contains almost $70M$ points. These timings also include the planar abstraction, i.e. the segmentation into planar patches and the extraction of bounding rectangles. Note that in this pre-processing step we set $\theta_{\text{off}} = 0.5$cm and $\theta_{\text{ang}} = 0.975$; the specific choice of θ_{adj} does not influence the computation, since this pipeline only considers the nodes (i.e., the rectangles) of the input graph G_{adj}, without taking into account the adjacency relations between them.

Correctness of reconstruction The behavior of our modeling pipeline was studied both for multi-room environments and for individual rooms. The reconstruction results for the datasets considered are shown in Figure 3.9 (real-world datasets) and in Figure 3.10 (synthetic datasets).

The datasets 'Room 1' and 'Room 2', each consisting of a single room, allow to specifically evaluate the reconstruction of the room shapes, independently of the room detection capabilities of the pipeline. Both cases represent challenging inputs, as the large window fronts cause many reflection artifacts and thus a large amount of outliers; moreover, the shape of 'Room 2' is concave and highly irregular. Nevertheless, the method can correctly extract the shape of the two rooms.

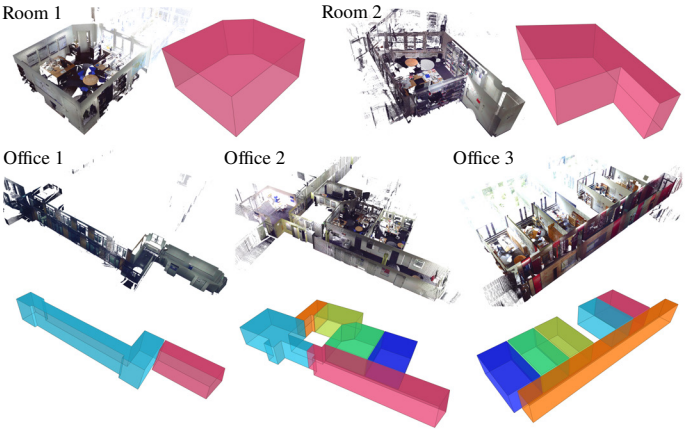


Figure 3.9: *Reconstruction results for real-world datasets. For clarity of visualization, the input models are shown with the ceiling removed.*

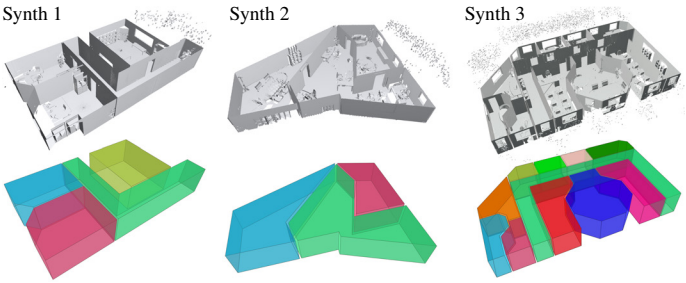


Figure 3.10: *Reconstruction results for synthetic datasets. Notice the presence of scattered points in the input models, added to simulate the artifacts from laser rays hitting reflective surfaces (e.g. windows). Ceilings have been removed for the sake of clarity.*

The datasets 'Office 1', 'Office 2' and 'Office 3' are part of three different office environments, all composed of multiple rooms. The highly anisotropic shape of the corridor in 'Office 1' is reconstructed correctly and separated from the neighboring room. 'Office 3' represents a larger environment composed of several rooms, all attached to a central corridor, whereas 'Office 2' has a more complex and irregular structure, lacking a single central corridor and containing some empty space completely surrounded by other rooms. In both cases, all the rooms are correctly detected and reconstructed.

In order to test the approach on more general architectural shapes, we performed additional tests using three synthetic models ('Synth 1', 'Synth 2' and 'Synth 3'). In all these models, the depth measurements produced during the ray-casting process were corrupted with additive Gaussian noise, using a standard deviation $\sigma_{\text{noise}} = 1mm$; this corresponds to a noise level comparable to that of static laser range-scanners. Moreover, we simulated the scattered outliers caused by window reflections. To do so, instead of discarding all the rays that exit through a window opening, for 0.5% of these rays we generate depth values from a uniform distribution in the range $6m - 8m$ (chosen so that the resulting points are outside of the room). 'Synth 1' and 'Synth 2' show two cases that clearly violate the Manhattan-World assumption; in particular, the rooms in 'Synth 2' exhibit a very irregular boundary and a high variation of incident angles between walls. Even more challenging is 'Synth 3', as it contains a higher number of rooms, many of which having a non-trivial shape composed of many separate wall segments (see, for instance, the central room colored in blue). In spite of these challenges, the proposed pipeline is capable of producing valid results for all three datasets.

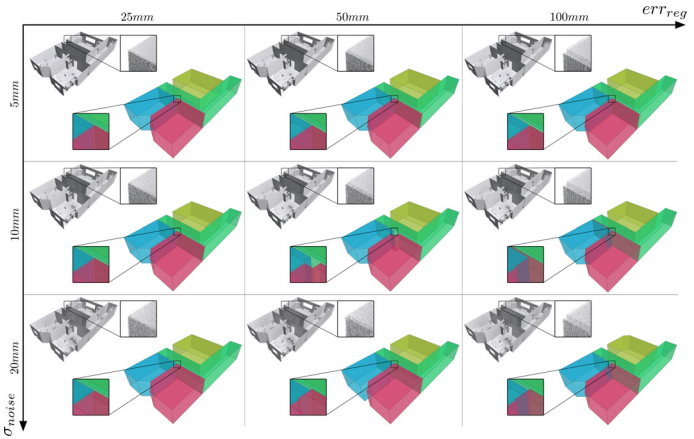


Figure 3.11: *Reconstruction results for dataset 'Synth 1' corrupted with increasing levels of measurement noise (σ_{noise}, in the vertical axis) and registration error (err_{reg}, in the horizontal axis).*

Robustness To assess the robustness of our method we have corrupted model 'Synth 1' with high levels of noise (using $\sigma_{\text{noise}} = 5, 10, 20mm$) and introduced an artificial registration error ($25, 50, 100mm$) in the translation part of the registration matrices. While such levels of degradation do not appear in data obtained by static laser scanning, they are not uncommon when using other acquisition technologies, such as hand-held cameras or cart and vehicle-based systems. As shown in Figure 3.11, for a combination of high levels of noise and high alignment error some reconstruction artifacts appear (e.g., walls separating adjacent rooms are attached to the actual rooms). Even under these challenging conditions our algorithm is able to successfully detect all rooms in the environment.

Reconstruction accuracy In addition to the results shown above, we have evaluated the reconstruction accuracy of our method, using both synthetic ('Synth 1') and real-world inputs ('Room 1').

In the first case, the 3D model created during the modeling process serves as a ground truth. We generated two virtual scans of this model, with and without noise; we then selected for each wall surface of the ground truth model the set of points lying on it. This made it possible to evaluate the distance from each reconstructed face to the real inlier points. As shown by the color-coded visual-

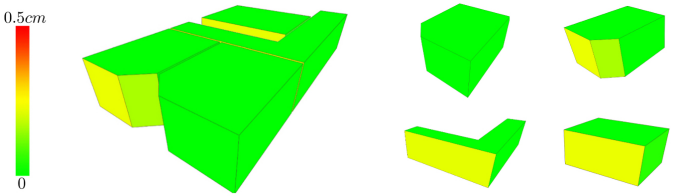


Figure 3.12: *Color-coded visualization of the maximum fitting error for the visible wall faces of the synthetic dataset 'Synth 1'. A detailed analysis of the errors for all reconstructed faces (including those not visible in this picture) is given in Figure 3.13.*

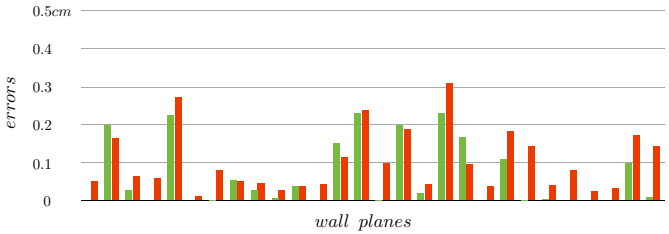
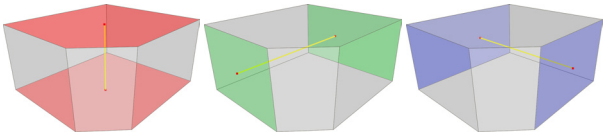


Figure 3.13: *Accuracy of the final wall planes reconstruction for synthetic dataset 'Synth 1'. The green bars refer to the IRLS fitting scheme proposed, while the red bars correspond to a more standard reference approach (LMS). For each wall plane of the reconstructed model (on the horizontal axis) the maximum distance from the plane to the set of points used for the fit is shown.*

ization in Figure 3.12, the accuracy of the reconstruction is generally very good. The more detailed plot in Figure 3.13, which quantifies the errors for all the faces of the model, shows that the maximum error exceeds $2mm$ for a few faces only, while for most of them is well below $1mm$. We also compared these reconstruction errors (obtained with our IRLS-based fitting scheme) against those resulting from the use of the Least Median of Squares (LMS) approach; the comparison shows that the accuracy of IRLS is comparable and generally better than the one of LMS.

Performing a quantitative evaluation for real-world datasets like 'Room 1' is more problematic, as a reliable ground-truth model of the acquired environment is unavailable. We therefore evaluated the distance between pairs of parallel walls of the real environment using a manually-operated laser distance measure device.



Real	$3.086m$	$6.170m$	$5.700m$
Model	$3.090m$	$6.160m$	$5.689m$
Disp.	4.81 / $4.91cm$	5.71 / $7.51cm$	3.22 / $5.11cm$

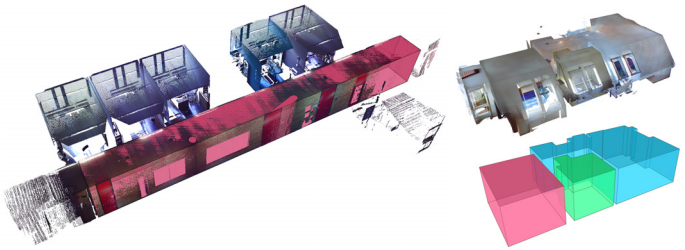
Table 3.2: *Reconstruction accuracy for dataset 'Room 1'. For each pair of parallel walls, we show their real-world distance (*Real*) and their distance in the reconstructed model (*Model*). The last row (*Disp.*) shows, for each wall plane in each pair of parallel walls, the spatial dispersion of the points that are used for the fit with respect to the plane itself.*

We then compared the measurements with the distance between the corresponding wall faces in the reconstructed model. The results shown in Table 3.2 confirm that also on real-world inputs our method is able to achieve very good accuracy levels.

3.7 Discussion and Outlook

This chapter presented a pipeline for the architectural reconstruction of real-world multi-room building interiors that exhibit a 2.5D structure. Our method is robust against clutter and occlusions and is able to partition the environment into the correct number of individual rooms. This approach represents an important step towards going beyond a plain geometric reconstruction of an indoor environment to extract semantic information as well. A number of more recent works complement and improve the capabilities of our pipeline [Turner and Zakhor, 2014; Ochmann et al., 2016], in some cases integrating the room reconstruction into a more extensive recovery of the semantic structure of the environment [Ikehata et al., 2015; Armeni et al., 2016].

Though innovative and of significant practical value, our method has several limitations. The approach focuses on the robust extraction of the basic room shapes and does not attempt to recognize fine architectural details, such as small recesses in the walls; these can be recovered separately, by performing a 2D analysis of each wall surface extracted. Moreover, the room detection relies on the assumption that each room contains at least one scan position; rooms that do not satisfy this condition can not be extracted by the iterative clustering procedure (Figure 3.14(a)). Note, however, that this assumption is typically verified in real-world datasets. The most significant limitation of this method is linked to the



(a) (b)

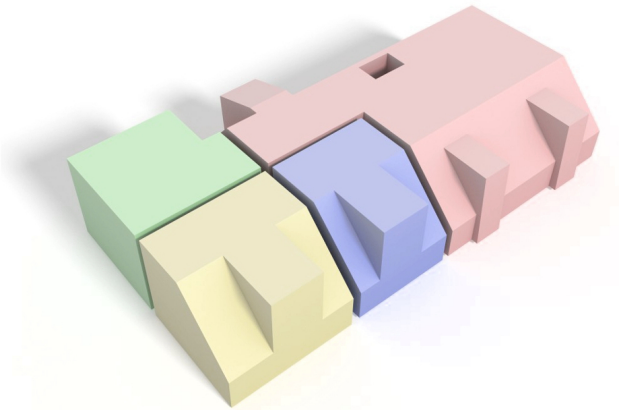
Figure 3.14: *Some limitations of our 2.5D reconstruction approach. Rooms that are only partially covered by scans originating in other rooms (such as the rooms attached to the corridor in (a)) are not recognized by our method. Moreover, our pipeline can not faithfully model environments with slanted walls (b) and different ceiling heights.*

2.5D prior used. While this allows to simplify the modeling problem and to make it more efficient, a number of real-world architectures with slanted surfaces (e.g. traditional roofed houses) can not be explained in terms of vertical walls and horizontal floors and ceilings (of the same height). In such cases, the approach can not deliver a satisfactory reconstruction, as clearly shown in Figure 3.14(b)).

The method described in Chapter 4 overcomes this latter, substantial limitation of the approach and allows for the modeling of more general architectures with arbitrarily oriented planar surfaces.

C H A P T E R 4

FULL-3D MODELING OF MULTI-ROOM ENVIRONMENTS



The approach described in Chapter 3 performs a room-aware reconstruction of an indoor environment starting from data containing artifacts and missing regions; however, due to the 2.5D prior used, it can not provide a meaningful reconstruction of a significant share of real-world building interiors. This chapter presents a pipeline that lifts the 2.5D assumption and can model environments with arbitrary wall orientations, thus allowing for a *full-3D* description of building interiors.

4.1 Motivation and Background

Many modern buildings exhibit a regular architecture, based on the presence of vertical wall surfaces and of horizontal floors and ceilings. This structure – better known in the literature as the 2.5D structure – is commonly used by automatic indoor modeling methods to simplify the reconstruction process and have a well-defined prior to guide the detection of permanent structures. Often, the condition that ceilings and floors have the same height is included into the 2.5D prior, as done in the approach presented in the previous chapter.

Although the methods based on this assumption have proven capable to reconstruct real-world interiors in a robust and often fully automatic manner, their applicability is limited to a subset of the settings that need to be covered by practitioners. A large array of cases (e.g. traditional roofed houses) have a *full-3D* nature, that is, they are bound by architectural surfaces with arbitrary orientations. When applied to digital models of such environments, 2.5D approaches often fail to produce a solution, or in the best case output a model that only roughly captures the real shape of the interiors (see Figure 3.14(b) in Section 3.7). In such cases, the required model must be created by hand using a 3D modeling software, possibly starting from the incorrect solution generated automatically. The manual reconstruction process is tedious, time-consuming and error-prone and is further complicated by the general orientations of the structures of the environment. For this reason, the need for automatic or semi-automatic pipelines for the modeling of indoor architecture is even stronger for cases that do not exhibit a 2.5D structure.

State-of-the-art Despite the great practical significance of the problem, most state-of-the-art reconstruction pipelines are based on the restrictive Manhattan-World and 2.5D assumptions. This is especially true for methods specialized in indoor environments. As already described in Section 3.1, traditional approaches assume axis-aligned wall orientations [Furukawa et al., 2009; Budroni and Böhm, 2010; Okorn et al., 2010; Adan et al., 2013; Turner and Zakhor, 2013]. More modern approaches (including the 2.5D pipeline proposed in Chapter 3) lift the orthogonality constraint between architectural surfaces [Turner and Zakhor, 2012; Oesau et al., 2014; Cabral and Furukawa, 2014] and are able to recover the seman-

tics of the environment [Ikehata et al., 2015; Ochmann et al., 2016; Armeni et al., 2016], but disregard the task of reconstructing more general 3D structures. The method of Oesau and colleagues [Oesau et al., 2014] goes in the direction of a more general 3D reconstruction, by creating a three-dimensional space partitioning from which the shape of the interior environment is extracted. However, this structure is built from vertical and horizontal planes only, thus leading in practice to a 2.5D reconstruction.

In the context of outdoor urban reconstruction, several methods are capable of reconstructing more general 3D structures. Often, the final geometric model of a building is obtained by performing an inside/outside partitioning of a spatial structure built around the objects of interest. Different structures have been proposed, typically derived from a *binary space partitioning* (BSP) [Chauve et al., 2010] or from a tetrahedral space subdivision [Lafarge and Alliez, 2013]. While these partitioning schemes can represent arbitrary wall orientations, the overall approaches are meant to work on relatively occlusion-free outdoor scenes and are therefore bound to fail in cluttered and heavily occluded settings such as real-world indoor environments. An exception to this is the work by Boulch et al. [Boulch et al., 2014], which is based on a visibility formulation that is amenable for indoor settings. However, none of these methods performs a separation between structural components and clutter nor deals with the detection of rooms.

Contribution This chapter introduces a novel modeling pipeline that lifts the restrictive Manhattan-World and 2.5D assumptions and is capable of reconstructing multi-room building interiors with arbitrary wall orientations. The proposed solution, published as a journal article [Mura et al., 2016] and presented in a connected conference talk, employs a three-dimensional BSP structure to represent the input scene, as done in more general outdoor reconstruction settings. The increased complexity of the data structure is balanced by an early pruning of the parts of the scene that do not belong to permanent structures of interest. This allows to handle a significantly larger class of real-world environments and reconstruct prevalent 3D structures such as slanted walls and sloped ceilings *robustly* and in a *unified manner*. While more expressive in the type of structures that can be recognized, our approach includes the main capabilities of modern indoor modeling frameworks, such as room detection and separation between permanent components and clutter.

4.2 Method Overview

The *full-3D* pipeline proposed strictly adheres to the general scheme outlined in Section 2.2.2. In particular, the method is composed of four main computation

steps, which extract from the input representation of the environment (i.e. the adjacency graph of bounding rectangles G_{adj} and the associated point clouds) a set of 3D polyhedra representing the rooms in the environment. As in the method described in Chapter 3, the input point clouds are assumed to cover the environment in an adequate manner (that is, with at least one scan in each room) and are represented using the grid set format (Equation 2.3), which allows to access the viewpoints from which the environment was acquired.

The individual steps, visually summarized in Figure 4.1, are shortly introduced in this section; an in-depth description is provided in the rest of the chapter.

Graph-based detection of permanent components The rectangles G_{adj} are divided into permanent components (i.e., the static architecture) and clutter (non-structural components like furniture) by reasoning on the structural relations between adjacent elements based on six rules.

Construction of 3D space partitioning The planes of the permanent components are used to build a 3D BSP structure that partitions the scene into a set of convex polyhedral cells, which are then arranged into a *cell complex* based on their adjacency relations.

Room detection A visibility-based clustering is applied to a subset of the cells of the complex to find the approximate location of the individual rooms.

Room reconstruction The final room models are reconstructed by applying a multi-label energy minimization approach to the whole cell complex and by performing the volumetric union of the sets of cells with the same label.

4.3 Graph-based Detection of Permanent Components

As highlighted in Section 3.3, the input graph G_{adj} contains many rectangles that correspond to furniture and other non-permanent objects. Not only these rectangles are not relevant to an architectural modeling of the environment, but they also significantly increase the complexity of the reconstruction, in particular of the space partitioning step. This is especially true in the 3D case, for which the construction of a space subdivision structure can represent a significant bottleneck [Verdie et al., 2015]. For this reason, it is fundamental to separate the components that belong to the permanent architecture from clutter early on in the pipeline.

In the absence of a prior on the orientation of the wall structures, we consider the adjacency relations between parts of the scene encoded in G_{adj} and reason on

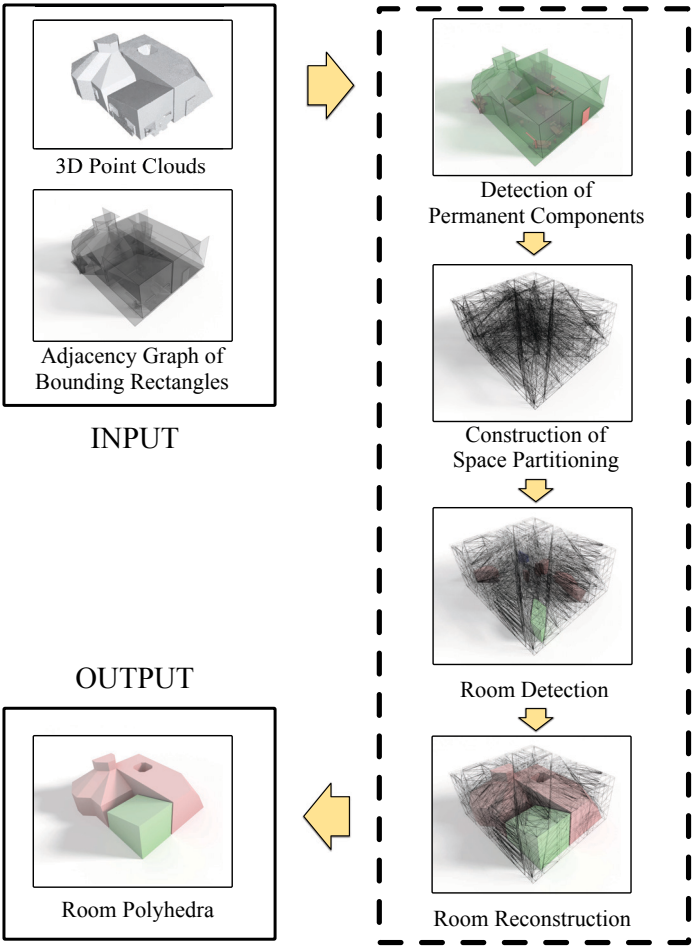


Figure 4.1: *Overview of our* full-3D *modeling pipeline. The computation strictly follows the general scheme outlined in Figure 2.1.*

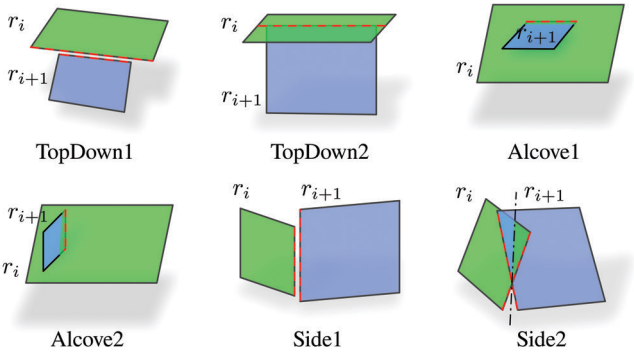


Figure 4.2: *The six structural patterns for pairs of neighboring bounding rectangles that are used in our structural region growing algorithm.*

the structural soundness of the environment. In particular, we observe that building interiors are composed of three main structural elements, namely *ceilings*, *walls* and *floor*, which are arranged from top to bottom in a consistent fashion, i.e. ceilings (on top) unload their weight onto the floor (bottom), typically transitioning through walls.

To exploit this intuition for the reconstruction, we introduce the concept of *structural paths* in the adjacency graph G_{adj}. A structural path is a sequence $W_S = (\, v_1,\, v_2,\, \ldots,\, v_{n-1},\, v_n\,)$, where $v_i \in G_{\text{adj}}$, v_1 corresponds to a rectangle in the ceiling, v_n to a rectangle on the floor and every edge $(\, v_i,\, v_{i+1}\,)$ is an edge in G_{adj} that is *structurally valid*. An edge $(\, v_i,\, v_{i+1}\,)$ is considered to be *structurally valid* if the two bounding rectangles r_i and r_{i+1} corresponding to the nodes v_i and v_{i+1} express a transition that is coherent with the top-bottom arrangement described above.

Valid transitions are encoded using a set of six spatial configurations, denoted as *structural patterns* and depicted in Figure 4.2. Patterns **1**-**4** capture the transition from the ceiling downwards to the floor (patterns **TopDown1** and **TopDown2**), including the special case of large alcoves that jut out of the main room structure (**Alcove1** and **Alcove2**). Patterns **5**-**6** (**Side1** and **Side2**) encode *lateral adjacencies* between walls. In particular, the mutual rectangle positions for these patterns can be described as follows.

1. TopDown1: at least one of r_i and r_{i+1} is not vertical and there exists a pair of edges s'_i, s'_{i+1} that are spatially close (i.e. their minimum distance is

$< \theta_{\text{adj}}/2$) and parallel to a same line; additionally, the projections of s'_i and s'_{i+1} onto said line overlap;

2. **TopDown2:** r_i rests on r_{i+1}, i.e., the top edge of r_{i+1} lies on the plane of r_i and its projection onto such plane is contained in r_i;

3. **Alcove1:** the top edge of r_{i+1} (any if r_{i+1} is horizontal) lies on r_i and the opposite edge lies on the negative half space of Π_i; note that Π_i has the same normal as r_i, which points towards the inside of the environment (as described in Section 2.2.2);

4. **Alcove2:** the bottom edge and either the left or the right edge of r_{i+1} intersect r_i;

5. **Side1:** r_i and r_{i+1} are both non-horizontal and have the same vertical slant (i.e., are either both vertical or are parallel); additionally, the left edge of r_i and the right edge of r_{i+1} (or vice-versa, the left edge of r_{i+1} and the right edge of r_i) are adjacent in the sense of **TopDown1**;

6. **Side2:** the intersection of the planes of r_i and r_{i+1} crosses the left edge of r_i and the right edge of r_{i+1} (or vice-versa, the left edge of r_{i+1} and the right edge of r_i).

To find structural paths in the contact graph, we apply the following base algorithm, described in pseudocode in Algorithm 3. For every ceiling node v_{ceil} (corresponding to a rectangle on a ceiling) we perform a region growing in G_{adj}, using v_{ceil} as starting node and only expanding along edges that are structurally valid in the sense of patterns **1**-**4** (line 12). If one or more ground patches are reached, we backtrack from each of them until v_{ceil} to extract the structural paths found.

Figure 4.3 shows a typical structural path that reaches the floor from a ceiling node, crossing edges that conform to **TopDown1**. In the same picture, the fixtures on the ceiling are not connected to the floor and are thus marked as clutter. Similarly, the cabinet represents an orthogonal extrusion from the nearby wall and does not represent structural support for the ceiling elements (TopDown1-2), hence it is not part of any structural path.

While this procedure works well when the whole extent of the structures of interest is visible, in practice furniture placed along the line of sight of the acquisition device often projects large shadows onto the scene, especially onto the lower parts of the walls. This means that the nodes in G_{adj} corresponding to such walls may have no edge connecting them to the floor. We therefore perform a second selection step (shown in Figure 4.4), starting the region growing from the structural patches found in the first pass and considering the lateral adjacencies

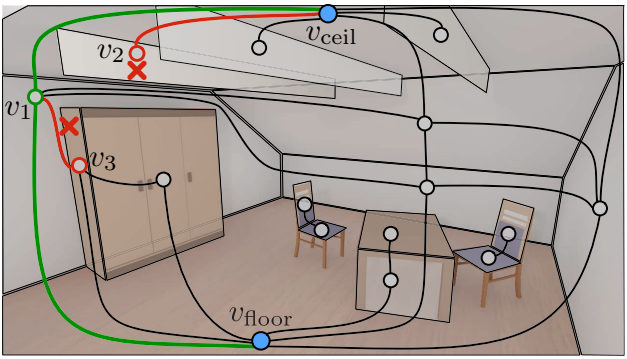


Figure 4.3: *Node v_1 belongs to a structural path ($(v_{\text{ceil}},\ v_1,\ v_{\text{floor}})$, in green) in the adjacency graph and is therefore marked as permanent. Nodes v_2 and v_3, reachable from the ceiling but not on a valid path to the floor, are classified as clutter.*

of **Side1** and **Side2**. Note that objects like cupboards and cabinets (see also Figure 4.3) are not wrongly added to a structural path by the side relations, as they do not follow a sequence of side-based adjacencies between structural elements (e.g. walls), but rather constitute blocks that protrude out of them.

This algorithm yields a set of bounding rectangles $\mathcal{R}_{\text{struct}}$ that correspond to the detected structural components of the environment. Although it does not make use of any assumption on the orientation of the wall surfaces, this approach is highly effective in separating permanent structures from clutter. Compared to the 2.5D approach presented in Chapter 3, the structural region growing achieves comparable or better results, as shown in the example in Figure 4.5.

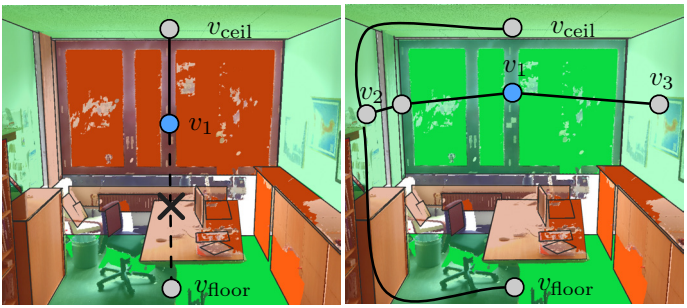
Note that, although this procedure consistently separates the main permanent elements from the clutter, it is not meant to perform a completely error-free extraction of the architectural structure of the environment. Its main goal is to simplify the problem for the subsequent stages, in particular for the final optimization-based reconstruction, which recovers the globally optimal room models.

Detection of floor and ceiling rectangles The extraction of the structural paths is guided by the sets $\mathcal{R}_{\text{ceil}}$ and $\mathcal{R}_{\text{floor}}$ of rectangles that lie on the ceilings and on the floors, respectively. It is therefore necessary to detect these two groups of rectangles before the structural region growing can be applied.

Input : Contact graph $G_{\text{adj}} = (V', E')$ of a scene
Set $\mathcal{R}_{\text{ceil}}$ of rectangles on the ceiling
Set $\mathcal{R}_{\text{floor}}$ of rectangles on the ground
Output: Set $\mathcal{R}_{\text{struct}}$ of rectangles of permanent structures

StructuralRegionGrowing (G_{adj} , $\mathcal{R}_{\text{ceil}}$, $\mathcal{R}_{\text{floor}}$)
  $\mathcal{R}_{\text{struct}} \leftarrow \emptyset$
  foreach $v_{\text{ceil}} \in \mathcal{R}_{\text{ceil}}$ **do**
    $Q \leftarrow \{v_{\text{ceil}}\}$
    mark v_{ceil} as visited
    $F \leftarrow \emptyset$
    $E_{\text{crossed}} \leftarrow \emptyset$
    while $Q \neq \emptyset$ **do**
      $v_{\text{curr}} \leftarrow Q$ .pop()
      $N \leftarrow$ nn_query (G_{adj} , v_{curr} , θ_{adj})
      foreach $v_n \in N$ **do**
        if (v_{curr}, v_n) *is structurally valid* **then**
          add (v_{curr}, v_n) to E_{crossed}
          if v_n *not visited* **then**
            if $v_n \in \mathcal{R}_{\text{floor}}$ **then**
              add v_n to F
            end
            else
              $Q \leftarrow Q \cup \{v_n\}$
              mark v_n as visited
            end
          end
        end
      end
    end
  end
  if $F \neq \emptyset$ **then**
    $\mathcal{R}_{\text{struct}} \leftarrow \mathcal{R}_{\text{struct}} \cup$ `Backtrack` (F , E_{crossed})
  end
  return $\mathcal{R}_{\text{struct}}$
end

Algorithm 3: Structural region growing for permanent components detection.



(a) (b)

Figure 4.4: *The rectangle corresponding to node v_1 is adjacent to a ceiling rectangle, but, due to viewpoint occlusions, it is not connected to the floor (a). However, using structural pattern **Side1**, v_1 can correctly be classified as structural, as it is connected by a lateral adjacency path between v_2 and v_3 that both belong to a structural path (b).*

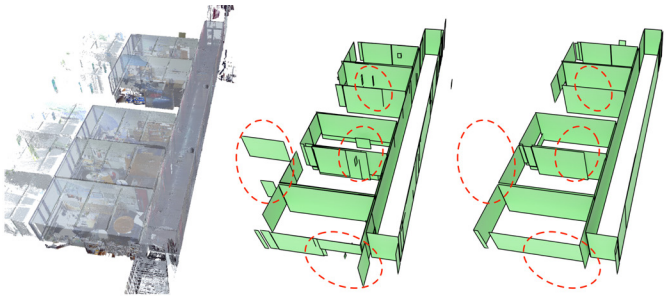
Following other state-of-the-art approaches [Turner and Zakhor, 2012; Oesau et al., 2014], horizontal floors and ceilings are detected by analyzing the z-histogram of the scene. However, since we also target buildings with non-horizontal roofs, we also analyze the rectangles that are non-horizontal. A non-horizontal rectangle is marked as belonging to a ceiling if the vertical projection of all the rectangles above it do not intersect it; here a rectangle r_i is considered to be *above* rectangle r_j if the vertices if r_i lie in the upper half-space defined by the plane of r_j.

4.3.1 Interactive Refinement

Ambiguous rectangles configurations caused by missing data can lead to unwanted results in the detection of the permanent components. For this reason, we provide the option to use interactive operations to correct such errors and to enforce specific completions of occluded regions.

Unlike full-fledged semi-automatic modeling pipelines [Arikan et al., 2013], the number of primitives affected by manual refinements is assumed to be negligible compared to the model complexity. For instance, to produce the reconstruction of 'Building D' shown in Section 4.7 only 21 rectangles were altered, representing only 4% of the total number of rectangles discovered in the model.

Throughout this process, all bounding rectangles are shown in color-coding according to their current label (permanent structures are visualized in green, clut-



(a) (b) (c)

Figure 4.5: *Results of the permanent component detection on dataset 'Office 3' (a) using the approach based on structural paths in 3D (c) and the 2.5D technique described in the previous chapter (b). The result shown in (c) was obtained without any user refinement.*

ter is shown in red). The user can intervene using the following three operations (shown in Figure 4.6).

(a) Label flip. Selecting a rectangle and inverting its label. A rectangle marked as permanent becomes clutter and vice-versa.

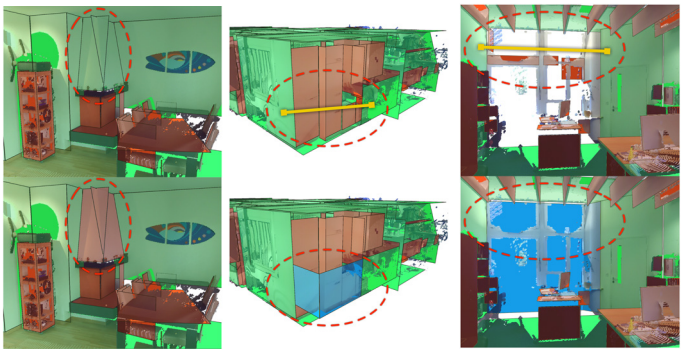
(b) Extension. Sketching a line that starts from one permanent rectangle and ends on another green rectangle. The two rectangles are extended to the intersection of their planes.

(c) Orthogonal extension. Sketching a line that starts from one permanent rectangle r_i and ends on another green rectangle r_{i+1}, crossing a boundary line segment s_i' of r_i. A new rectangle orthogonal to the plane of r_i and extending from s_i' until the plane of r_{i+1} is added as a new permanent component, while r_{i+1} is extended until reaching such extension.

4.4 Construction of 3D Space Partitioning

Differently from the 2.5D pipeline of Chapter 3, the space partitioning structure used in this approach is fully three-dimensional and consists in a 3D BSP tree, similar to that used in related methods [Chauve et al., 2010; Boulch et al., 2014].

We use as splitting planes for the construction of the BSP the *dominant planes* of the environment, that is, the unbounded planes that describe its main architectural shape. The procedure used to compute the dominant planes follows the one presented in the 2.5D case 3.4.1: we use the *mean-shift* algorithm [Comaniciu and Meer, 2002] to cluster the permanent rectangles according to their normal



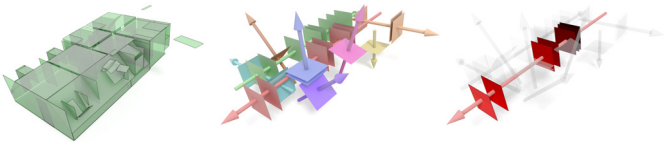
(a) (b) (c)

Figure 4.6: *Examples of interactive refinement of the automatic results: label flip (a); extension (b); orthogonal extension (c).*

direction and to their offset along that direction (see Figure 4.7). To avoid averaging effects inherent to the mean-shift procedure, we additionally perform for each cluster obtained a *least-median of squares* (LMS) fit [Rousseeuw and Leroy, 1987] on the points associated to its rectangles.

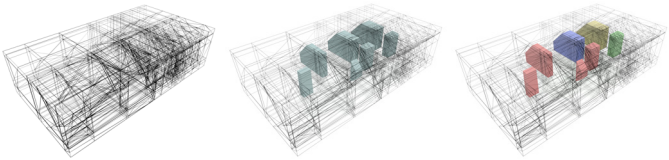
A BSP tree of the scene is then built by intersecting its bounding box (expanded by a small factor) with all the dominant planes extracted. It is well known [Wang, 2011] that every internal node of this tree corresponds to a split of the input volume into two half-spaces, and that every node in the tree is associated to a convex region obtained by the intersection of all half-spaces from that node to the root of the tree.

The set of polyhedral cells $\mathcal{C}$ (shown in Figure 4.8(a)) associated to the leaves of this tree corresponds to the partitioning of the space induced by the dominant planes. We assemble such cells according to their adjacency relationships $\mathcal{N} = \{(c, c') | c, c' \in \mathcal{C} \wedge c, c' \text{are adjacent}\}$ into a *3D cell complex* $(\mathcal{C}, \mathcal{N})$. An adjacency $(c, c') \in \mathcal{N}$ corresponds to a polygonal facet $f_{c,c'}$ shared by the polyhedra corresponding to c and c'. During the construction of the complex we associate to each such facet the ID of the dominant plane that generates it and the *coverage* of the facet (denoted by $\text{cov}(f_{c,c'})$), which corresponds to the fraction of its area covered by scanned points. It is worth mentioning that the coverage of a facet can be used as a measure of *dissimilarity* between its two adjacent cells, since a facet with high coverage is likely to lie on the surface of a permanent structure (e.g. a wall) and thus separate cells that belong to different environments.



(a) (b) (c)

Figure 4.7: *Extraction of dominant planes by clustering of the structural rectangles. The rectangles output by the structural region growing (a) are first clustered according to their direction (b); the rectangles in each cluster obtained are further clustered based on their offset along their direction (c).*



(a) (b) (c)

Figure 4.8: *Wireframe visualization of the polyhedral cells of the 3D cell complex (a); the positions of the acquisition device during the scanning fall inside the viewpoint cells (b), which can be clustered according to their visible surface overlap to obtain a set of clusters corresponding to the rooms in the environment (c).*

As in the case of the 2D cell complex described in Section 3.4.2, the use of *unbounded* primitives in the construction of the space partitioning contributes to making the overall pipeline robust to occlusions. This is because even if a part of a permanent structure is missing due to a viewpoint occlusion, the region of space surrounding that missing part will still be cut along the actual wall surface. In that region, the facets of the polyhedral cells will adhere to the real-world boundary corresponding to the wall, thus allowing the optimization-based reconstruction (Section 4.6) to extract the correct room shape.

4.5 Room Detection

In accordance with the general pipeline described in Section 2.1, the shape of each room is obtained as the volumetric union of a subset of regions of the complex $(\mathcal{C}, \mathcal{N})$. In this full-3D approach, the reconstruction is cast as a *multi-label optimization* problem [Boykov and Kolmogorov, 2004], in which each cell is as-

signed one out of $n_{\text{labels}} = n_{\text{rooms}} + 1$ labels, that is, n_{rooms} labels for the rooms, plus an additional one for the outer space and the space occupied by walls.

To reliably estimate n_{rooms}, we adapted the technique initially proposed by DiBenedetto and colleagues. [Di Benedetto et al., 2014]. In their method, aimed at generating panoramas for image-based rendering, multiple view probes are clustered using a *Markov cluster algorithm* (MCL) [Van Dongen, 2008] driven by the amount of visible surface overlap. We adapt their approach by considering as view probes the polyhedral cells $\mathcal{C}^{vp} = \{ c_1^{vp}, \ldots, c_{n_{vp}}^{vp} \}$ of the complex that contain a scan position (Figure 4.8(b)). As stated in Section 4.2, we assume that every room contains at least one scan position, which implies that the number of rooms in the environment can be obtained by correctly clustering the set $\mathcal{C}^{vp}$.

In the absence of an exact mathematical definition of a room [Turner and Zakhor, 2014], we define it as a sub-space of an environment mostly separated from the rest of the space by permanent components. From this it follows that two locations placed in different rooms should see very different parts of the scene, while the visibility from positions within the same room is very similar. We define the amount of *visible surface overlap* between two cells in terms of the visible structural bounding rectangles $\mathcal{R}_{\text{struct}}$ (see Section 4.3). In particular, let vis_i^r denote the visible area of $r \in \mathcal{R}_{\text{struct}}$ as seen from cell c_i and let $\text{overlap}^r(c_i, c_j)$ be the visibility overlap between cells c_i and c_j relative to rectangle r, defined as:

$$\text{overlap}^r(c_i, c_j) = 1 - \frac{\max(vis_i^r, vis_j^r) - \min(vis_i^r, vis_j^r)}{\max(vis_i^r, vis_j^r)} \tag{4.1}$$

We then define the visibility overlap $\text{overlap}(c_i, c_j)$ between the viewpoint cells c_i and c_j as follows:

$$\text{overlap}(c_i, c_j) = \frac{\sum_{r \in \mathcal{R}_{\text{struct}}} w_{i,j}^r \cdot \text{overlap}^r(c_i, c_j)}{\sum_{r \in \mathcal{R}_{\text{struct}}} w_{i,j}^r} \tag{4.2}$$

Here $w_{i,j}^r$ is a term that balances the importance of each rectangle taking into account all rectangles as seen from every viewpoint cell:

$$w_{i,j}^r = \frac{\max(vis_i^r, vis_j^r)}{\max_{r \in \mathcal{R}_{\text{struct}}, i=1 \ldots n_{vp}} vis_i^r} \tag{4.3}$$

Given this definition of visibility overlap, we construct a weighted graph G_{vp} that contains one node n_i for every viewpoint cell c_i^{vp}. This graph is undirected and complete; each edge (n_i, n_j) is assigned the weight $g(\,\text{overlap}(c_i^{vp}, c_j^{vp})\,)$, where $g(x)$ is a monotonically increasing concave function that suppresses the contribution of low values. We defined it as $g(x) = 1 - e^{-(x/0.5)^2}$, as this function proved to work well in all our test cases.

Applying the Markov clustering [Van Dongen, 2008] to G_{vp} yields a set of clusters $\Gamma = \{\Gamma_1, \ldots, \Gamma_{n_{\text{rooms}}}\}$ (shown in Figure 4.8(c)), grouping the viewpoint cells according to their visibility overlap. The algorithm automatically selects the number of room clusters, which, thanks to the similarity measure chosen, corresponds to the number of rooms n_{rooms} of the environment. In all our experiments, the MCL clustering was used with inflation 1.1 and thresholding the links with weight < 0.1; setting these two parameters and using the default values for all the others allowed to obtain consistently good results.

It is worth stressing that this clustering step is only applied to the cells that contain a viewpoint (the viewpoint cells). While this process could be applied to all cells in $\mathcal{C}$ to obtain the shapes of the rooms, the resulting reconstruction would only be based on visibility information and lack geometric and structural regularity. For this reason, we extract the final room models using a more expressive multi-label energy minimization approach.

4.6 Room Reconstruction

The previous steps of the pipeline yield a polyhedral cell complex $(\mathcal{C}, \mathcal{N})$ and n_{rooms} clusters $\Gamma_1, \ldots, \Gamma_{n_{\text{rooms}}}$, each corresponding to a room of the environment and containing the viewpoint cells located inside that room. In this final step, we assign each polyhedral cell of the complex one label from the set $\mathcal{L} = \{l_1, \ldots, l_{n_{\text{rooms}}}, l_{\text{out}}\}$, which includes one label for each room plus the additional label l_{out} for the outer space. The problem can be modeled naturally as a multi-label *Markov random field* (MRF) [Boykov et al., 2001]; in particular, we seek the optimal label assignment $L^* = \{\ L_c^* \mid L_c^* \in \mathcal{L}\ ,\ c \in \mathcal{C}\ \}$ that minimizes an energy function of this kind:

$$E_{\text{label}}(L) = E_{\text{data}}(L) + E_{\text{smooth}}(L)\ . \tag{4.4}$$

These two terms (denoted in the literature as *data* and *smoothness* terms) correspond to the energy associated, respectively, to an initial, coarse labeling of the cells and to the coherency of the label assignments to pairs of adjacent cells.

4.6.1 Data Term

E_{data} consists of a sum of unary functions, each representing a penalty for assigning label $L_c \in \mathcal{L}$ to a cell $c \in \mathcal{C}$:

$$E_{\text{data}}(L) = \sum_{c \in \mathcal{C}} D_c(L_c) \tag{4.5}$$

To derive the data terms, we treat the viewpoint cells $c' \in \Gamma_i$ of a room cluster Γ_i as representatives of the visibility for that room; we then compute the penalty

for assigning a label l_i to any cell c in terms of the visibility overlap with the viewpoint cells in Γ_i (using Equation 4.2). Since the notion of viewpoint cell does not apply to the cluster of outer space, the penalty for labeling a cell c as l_{out} is defined as the area $O(c)$ of the rendered scene (as seen from the center of c) that corresponds to empty space (i.e., not occupied by any structural rectangle). The term $D_c(L_c)$ can therefore be expressed as follows:

$$D_c(L_c) = \begin{cases} 1 - \max_{c' \in \Gamma_{L_c}} \text{overlap}(c, c') & \text{if } L_c \neq l_{\text{out}} \\ O(c) & \text{otherwise} \end{cases} \tag{4.6}$$

4.6.2 Smoothness Term

The effect of E_{data} is balanced by the smoothness energy E_{smooth}, which aims at regularizing the labeling by penalizing the assignment of different labels to adjacent cells. We express this term as a sum of four sub-terms, which enforce not only fidelity to the measured data, but also geometric simplicity of the resulting model, structural coherency of the rooms and separation between rooms:

$$E_{\text{smooth}}(L) = \lambda_{\text{cov}} E_{\text{cov}} + \lambda_A E_A + \lambda_G E_G + \lambda_{\text{sep}} E_{\text{sep}} \tag{4.7}$$

Here the dependency of the energy sub-terms on the labeling L has been omitted for brevity. Each of these energies is defined as a sum of pairwise potentials involving the pairs $(c, c') \in \mathcal{N}$ of adjacent cells in the complex and has the following general form:

$$E_{\dots}(L) = \sum_{(c,c') \in \mathcal{N}} V^{\dots}_{c,c'}(L_c, L_{c'}) \tag{4.8}$$

The individual terms are defined as follows:

- Coverage term E_{cov}: penalizes assigning the same label to a pair (c, c') if the facet $f_{c,c'}$ is densely covered by scanned points; its potential is defined as follows:

  $$V^{\text{cov}}_{c,c'}(L_c, L_{c'}) = \mathbf{1}(L_c \neq L_{c'}) \cdot \text{cov}(f_{c,c'}) \tag{4.9}$$

  where $\text{cov}(f_{c,c'})$ is the coverage defined in Section 4.4;

- Area term E_A: favors geometric simplicity by penalizing the total area of the interface surface between two rooms; its potential is defined as:

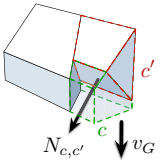
  $$V^{A}_{c,c'}(L_c, L_{c'}) = \mathbf{1}(L_c \neq L_{c'}) \cdot \text{area}(f_{c,c'}) \tag{4.10}$$

  where $\text{area}(f_{c,c'})$ is the surface area of the facet $f_{c,c'}$ between cells c and c';

- Gravity term E_G: penalizes label assignments to a pair (c, c') that leave one cell "floating", i.e. without a cell below to sustain its weight (see Figure 4.9); this condition is formulated with the following potential:

$$V^G_{c,c'}(L_c, L_{c'}) = \mathbf{1}(L_c \neq L_{c'}) \cdot (v_G \cdot N_{c,c'}) \cdot u(f_{c,c'}) \cdot s(c, c', L_c, L_{c'}) \quad (4.11)$$

where v_G is the vector $(0, 0, -1)$, $N_{c,c'}$ is the normal of $f_{c,c'}$, $u(f_{c,c'})$ is a function that takes value 0 if the facet $f_{c,c'}$ lies on the floor (and 1 otherwise) and $s(c, c', L_c, L_{c'})$ is a function that takes value 1 if the bottom cell in the pair (c, c') is labeled l_{out} and the upper cell has a label $\neq l_{\text{out}}$, and 0 otherwise (see inset, which describes the case $s(c, c', L_c, L_{c'}) = 1$);



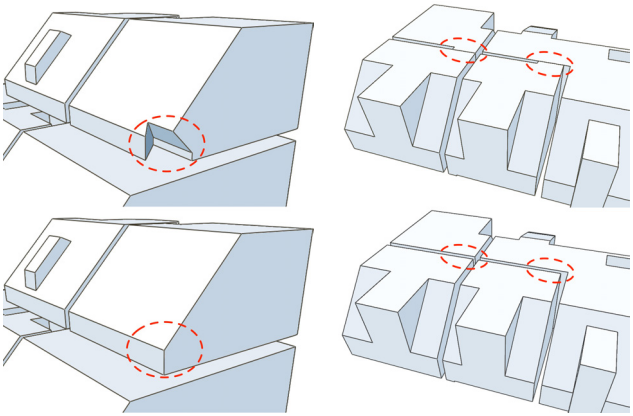
- Room separation term E_{sep}: enforces the presence of thick walls between the clusters of any two rooms, by penalizing the assignment of different labels to adjacent cells unless one of the two labels is L_{outer}; the corresponding potential is as follows:

$$V^{\text{sep}}_{c,c'}(L_c, L_{c'}) = \begin{cases} 1 & \text{if } L_c = L_{\text{outer}} \vee L_{c'} = L_{\text{outer}} \\ 0 & \text{otherwise} \end{cases} \quad (4.12)$$

The relative importance of the data term and of the smoothness sub-terms can be adjusted by varying the values of λ_{cov}, λ_A, λ_G, λ_{sep} in Equation 4.7 in the range $[0 \dots 1]$. In all our tests, we set $\lambda_{\text{cov}} = 0.2$, $\lambda_A = 0.05$, $\lambda_G = 0.1$, $\lambda_{\text{sep}} = 0.2$. Since all the pairwise potentials used satisfy by definition the submodularity property and are *semi-metrics*, the energy function of Equation 4.4 can be minimized using the $\alpha - \beta$ *swap* algorithm [Boykov et al., 2001]. The minimum energy is associated to the optimal labeling L^* and the shape of the individual rooms can be reconstructed by volumetric union of the cells with the same label.

4.7 Results

The effectiveness of our modeling pipeline was evaluated on a test suite composed of 8 datasets. Three of these correspond to 2.5D environments and were used to compare our novel pipeline against more constrained state-of-the-art approaches, including the one presented in Chapter 3; the remaining models clearly violate the 2.5D and Manhattan-World assumptions and highlight the capabilities of the proposed full-3D approach. All the relevant information about the reconstruction



(a) (b)

Figure 4.9: *Effects of gravity and room separation terms. Our gravity term helps obtain plausible reconstructions by favoring stable cells configurations (a), while our room separation term ensures that rooms are separated by empty space (b).*

results are provided in Table 4.1; a description of the datasets used can be found in Appendix A.

Implementation Our prototype implementation of the approach is written in C++ and uses the publicly available implementations of the MCL [Van Dongen, 2008] and $\alpha - \beta$ *swap* [Boykov and Kolmogorov, 2004] algorithms. All tests were performed on a MacBook Pro with an Intel Core i7 (2.5GHz), 16GB DDR3 RAM and an NVIDIA GeForce GT 750M. The processing times (listed in column 'Time' of Table 4.1) vary from about 20 seconds (for dataset 'Cottage') to about 5 minutes (for 'House'). As in the case of our 2.5D method, the timings include the initial planar segmentation and construction of the adjacency graph of bounding rectangles.

To allow for a re-implementation of the approach, we provide here some details on the computation of the coverage of a facet and of the visible area of a rectangle. The coverage $\mathrm{cov}(f_{c,c'})$ of a facet $f_{c,c'}$ (introduced in Section 4.4) is estimated by rasterizing the projection of the corresponding point splats onto the facet and by evaluating the difference between the area of the facet and the area

Dataset	#Rooms	#Rect. (perm./clut.)	#Alt.	#Planes	#BSP cells	Time
Office 2	6	629 (121/508)	25	53	4958	79.9s
Apartment 1	5	377 (143/234)	27	71	9634	177.5s
Building D	27	501 (252/249)	21	68	8317	304.0s
Penthouse	5	332 (93/239)	17	51	7188	66.5s
Maisonnette	5	296 (115/181)	8	63	12202	112.9s
Cottage	7	159 (51/108)	6	29	836	19.3s
House	9	519 (153/336)	13	66	11784	317.7s
Modern	3	266 (57/209)	0	49	5717	70.6s

Table 4.1: *Relevant statistics on the reconstruction process. From left to right: number of rooms detected; number of bounding rectangles extracted automatically (in brackets: structural rectangles/clutter rectangles); number of rectangles altered during interactive refinement; number of dominant planes; size of the BSP complex (i.e., number of cells); overall computation time.*

covered by the splats. The visible area vis_i^r of rectangle r as seen from cell c_i (defined in Section 4.5) is estimated by rendering r from the center of mass of c_i and by then evaluating the area of its rasterization. In both cases, the screen-space areas are computed using occlusion queries. We have empirically found that the viewport size used has little or no influence on the outcome of the reconstruction. Finally, we note here that input adjacency graph is obtained using $\theta_{\text{off}} = 0.5$cm and $\theta_{\text{ang}} = 0.975$ in the planar decomposition and by considering two bounding rectangles as adjacent if their minimum distance θ_{adj} is less than 20cm; note that we pruned the rectangles with a diagonal < 20cm and a ratio between smaller and greater side < 0.001.

Correctness of reconstruction (2.5D case) The tests on the models of 2.5D interiors (Figure 4.10) confirm that the full-3D method proposed, although designed to capture more general 3D structures, is able to process this type of environments with results comparable to those of more constraint methods. In particular, the reconstructions of 'Office 2' and 'Apartment 1' show that ceilings of different heights can be detected without requiring any ad-hoc steps, which is not possible with the method proposed in Chapter 3. Although less rich in fine-grained geometric detail, the reconstruction of 'Apartment 1' obtained with our 3D pipeline is similar to that output by the method of Ikehata and colleagues [Ikehata et al., 2015]. Moreover, even for environments with a high room count such as 'Building D' (27 rooms) the proposed approach can produce a correct recon-

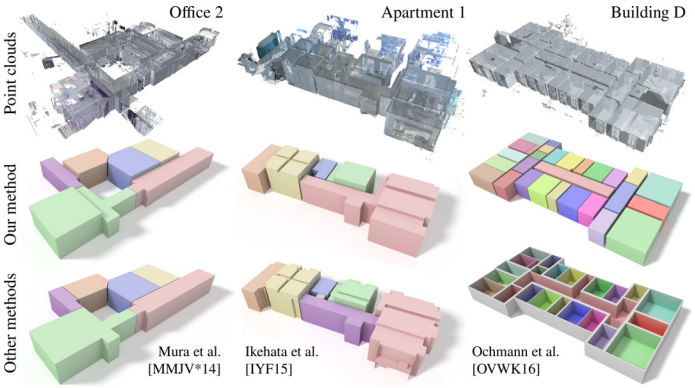


Figure 4.10: *Reconstruction results for 2.5D datasets and comparison with state-of-the-art. From top to bottom: input 2.5D models (TOP ROW), reconstruction obtained with our proposed full-3D pipeline (MIDDLE) and reconstruction obtained with state-of-the-art 2.5D methods (BOTTOM).*

struction. The output of the visibility-based room detection algorithm is consistent with that of other 2.5D approaches [Ikehata et al., 2015; Ochmann et al., 2016]; only minor differences occur in ambiguous cases, such as in the presence of wide passages between rooms and corridors.

In Figure 4.11 we show a direct comparison with the approach introduced in Chapter 3 on the first floor of 'Maisonnette', which exhibits several features that can not be described in 2.5D. While the 2.5D method can reconstruct an approximate floorplan, many large wall structures, as well as entire sub-spaces (e.g. the alcove in the green room) are lost. On the other hand, the 3D approach achieves a relatively faithful reconstruction fully automatically, which can be further improved with a few operations of the proposed interactive refinement. Note that the complete dataset with fine details highlighted is shown in Figure 4.12. In addition to this, we run the 2.5D method on building models with more irregularities like 'Modern', but the results produced barely reflected the actual shapes of the rooms. This is due to errors in the occlusion-aware detection of permanent components, which is meant to work on environments with vertical walls and with a clearly identifiable ceiling height.

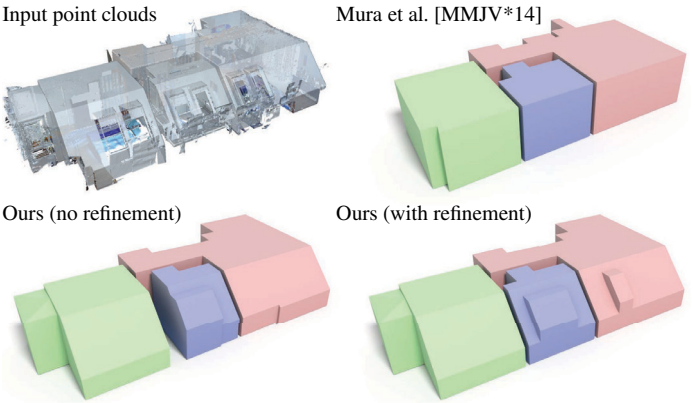


Figure 4.11: *Comparison between the results obtained with our 2.5D modeling pipeline (Chapter 3) and those produced by our full-3D method without and with user intervention for the first floor of 'Maisonnette'.*

Correctness of reconstruction (full-3D case) Reconstruction results for a set of more general models are shown in Figure 4.12. All environments are composed of multiple rooms and exhibit many slanted ceilings and wall surfaces, which makes them well-suited to evaluate the capabilities of our pipeline. 'Cottage' represents an environment with a traditional gable roof and consisting of two large rooms (each with a bedroom, a bathroom and a small passage) and a corridor in relatively regular arrangements. 'Penthouse' is a more challenging model, with less regular room shapes, more structural detail (e.g. window alcoves) and high levels of clutter. The reconstruction results show that our method can correctly capture the architectural shape of this kind of environments. The proposed method can also handle more complex environments with multiple stories such as 'Maisonnette' and 'House'. 'House' (synthetic dataset) corresponds to a 3-story house containing many rooms and interior details (as shown in the inset). 'Maisonnette' contains many structures that violate both the Manhattan-World and the 2.5D assumption, exhibits irregular room shapes and is rich in geometric detail. In particular, the large window alcove in the green room has a roof with several different orientations; as shown in the inset in Figure 4.12, these are preserved in the final reconstruction. The ability to capture fine-grained orientations is also demonstrated by 'Modern', a synthetic dataset containing many complex wall arrangements and configurations, all captured correctly in our reconstruction.

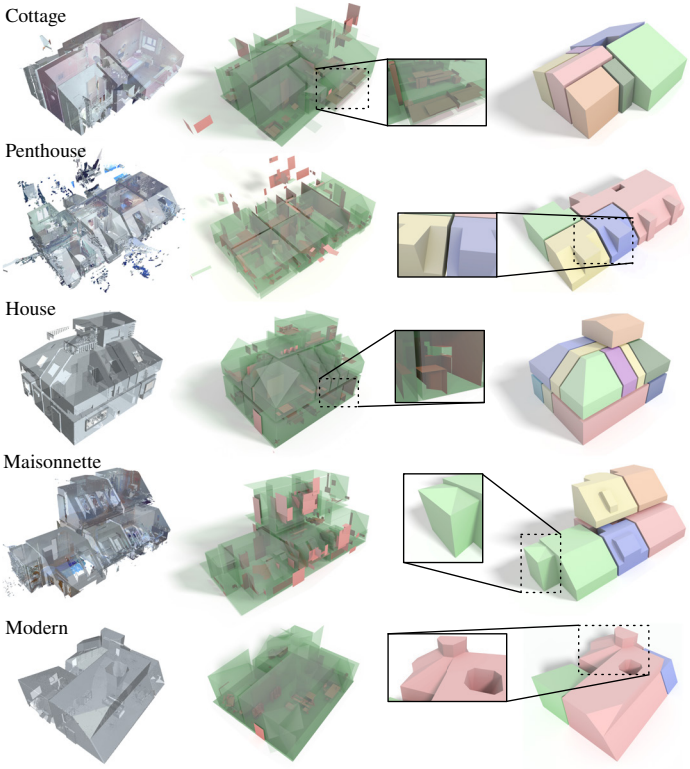


Figure 4.12: *Reconstruction results for general 3D datasets. For each dataset, we show the input point cloud (LEFT), the classification into structural (green) and clutter (red) rectangles (MIDDLE) and the reconstructed room polyhedra (RIGHT). Insets show either details of the complexity of the planar components or of the final reconstruction.*

Interactive refinement We evaluated the number of operations and the time needed in the interactive refinement to complete a typical model with high levels of clutter, using the first floor of 'Maisonnette' as shown in Figure 4.11. In this case, the user intervention helps reconstruct the correct boundaries for the blue room in almost complete absence of scanned evidence for a wall surface. We

asked 1 expert and 2 novice users to complete the task. The expert user completed the refinement in 43 seconds performing 6 operations, while the 2 novice users (who received a complete introduction and training to the system) performed, respectively, 12 and 11 operations in 360 and 339 seconds. Note that the inexperienced users produced results comparable to those obtained by the expert and shown in Figure 4.11. In addition to this evaluation, we report in Table 4.1 the number of operations performed on each test model.

Rectangles classification The first floor of 'Maisonnette' was also used to evaluate the detection step (Section 4.3) in terms of precision and recall. The values obtained, respectively 97% and 93%, show that the approach is effective in selecting the main architectural features of the environment. Note that the relatively lower recall value is due to the almost complete lack of input data on a specific wall surface: the few rectangles on that wall could not be reached by any structural path and were therefore classified as non-permanent. As explained in the previous paragraph, a few interactive operations were sufficient to fix these issues and obtain an optimal reconstruction.

We also quantitatively compared our classification step with that of the 2.5D approach of Chapter 3 using 'Office 3', thus complementing the visual comparison in Figure 4.5. The full-3D method obtained a precision of 85% and a recall of 77%, whereas the 2.5D approach scored, respectively, 90% and 48%. The lower precision value of the structural region growing is due to false positives corresponding to spurious ceiling rectangles caused by window reflection artifacts; as these are scattered and isolated by nature, they do not affect the reconstruction. Both methods attain fairly low recall values on this dataset. In the case of the full-3D approach, this is due to an issue similar to the one already described for the first floor of 'Maisonnette': due to occlusions, some isolated rectangles corresponding to the lower parts of many walls are not included in any structural path, which significantly lowers the recall. However, the larger, unoccluded upper parts of the same walls are correctly classified as structural, thus ensuring that the correct room boundaries appear in the final reconstruction.

Robustness We analyzed the robustness of our approach by corrupting the synthetic model 'Modern' with increasing levels of noise. In our first test, we varied the noise level while keeping all parameters of our pipeline fixed; the input planar decomposition was obtained by adapting the threshold θ_{off} of the region growing to the noise level of the input dataset. As shown in Figure 4.13, our method is fairly robust for noise levels corresponding to $\sigma_{\text{noise}} \leq 5$cm.

We also evaluated the robustness with respect to the adjacency threshold θ_{adj} used to construct the adjacency graph G_{adj}. We did this for every noise level con-

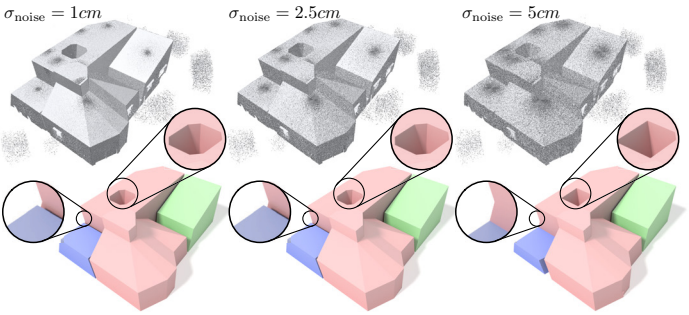


Figure 4.13: *Robustness with respect to increased measurement noise. The progressive decrease in reconstruction accuracy is shown by the details in the insets.*

sidered, but for $\sigma_{\text{noise}} = 1cm, 2.5cm$ we did not detect meaningful differences with the results in Figure 4.12; we therefore show in Figure 4.14 only the results obtained at the highest noise level ($\sigma_{\text{noise}} = 5cm$). Decreasing θ_{adj} from the default value ($20cm$) to $10cm$ leads to an incorrect classification of some structural rectangles and eventually to an erroneous reconstruction of the blue room. For $\theta_{\text{adj}} = 5cm$ even more rectangles are misclassified, resulting in the green room being attached to the central space. Notice, however, that in this case the adjacency threshold θ_{adj} corresponds to the noise level considered and that the boundaries of the patches produced in the pre-processing (Section 2.2.2) can exhibit inaccuracies proportional to the noise level in the input point cloud. Under these conditions, a failure of the classification step is bound to happen.

Besides studying the influence of σ_{noise} and θ_{adj} on the final reconstruction, we analyzed how they influence the precision and recall values for the detection of the permanent components. For $\sigma_{\text{noise}} = 1cm, 2.5cm$ both measures attain at 100%, whereas for $\sigma_{\text{noise}} = 5cm$ we obtained a precision of 94% and a recall of 88%. The influence of θ_{adj} is stronger: for $\theta_{\text{adj}} = 20cm, 10cm, 5cm$, the precision/recall pairs are, respectively, $94\%/88\%$, $96\%/85\%$, $94\%/60\%$. These numbers show that our pipeline is relatively robust to measurement noise, but is more sensitive to variations in the adjacency threshold; in particular, the recall values show that the number of correctly classified rectangles steadily decreases when making the requirements for adjacency between rectangles stricter. When dealing with data of real-world scenes, which are heavily affected by viewpoint occlusions and thus missing regions, such a behavior is to be expected.

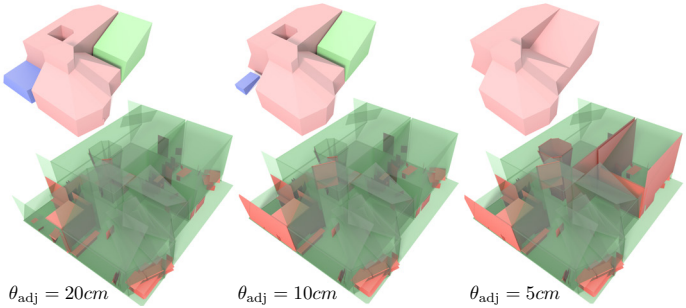
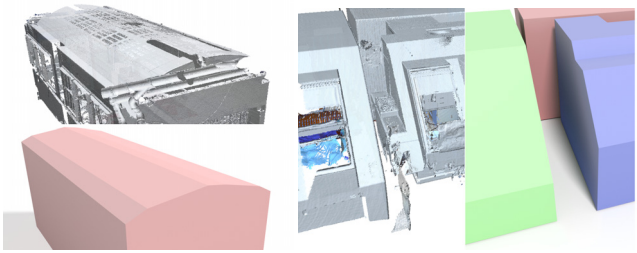


Figure 4.14: *Robustness to varying adjacency threshold in the graph. The model shown is 'Modern', corrupted with Gaussian noise of $\sigma_{\text{noise}} = 5cm$ (see Figure 4.13).*

4.8 Discussion and Outlook

The approach presented in this chapter represents the first pipeline for architectural reconstruction of multi-room building interiors that goes beyond the 2.5D and Manhattan-World assumptions. It allows to reconstruct the individual room polyhedra in fairly complex 3D environments with arbitrary wall orientations; nevertheless, it is robust to clutter and missing data and is capable of detecting the separate rooms of the environment.

Our evaluation has proven the method to work very well on a wide array of real-world cases. Nevertheless, the approach has some limitations. First of all, while most round surfaces in practice are approximated well by our approach, in presence of occlusions the proposed piecewise-planar approach can not guarantee that non-flat surfaces are recovered in a uniformly smooth manner (see Figure 4.15(a)). Secondly, very high levels of clutter can make the reconstruction problem ill-posed. Due to the presence of built-in bookshelves or cabinets, some boundary walls are almost entirely missing in the scanned model (see detail of 'Maisonnette' in Figure 4.15(b)) and can only be recovered with user interaction. In addition to this, due to the generality of building shapes targeted, some reconstruction solutions are intrinsically ambiguous without additional prior information. For instance, the chimney in Figure 4.6(a) could be considered not to be part of the main architectural shape of the environment, but its rectangles are detected as structural by the automatic classification. User input is needed to disambiguate these cases. Moreover, similar to what already mentioned for the 2.5D method in Section 3.7, the use of the scan positions in the room detection process



(a) (b)

Figure 4.15: *Some limitations of our* full-3D *method. Using a piecewise-planar approach, curved surfaces may not be smoothly approximated (a). If a surface is almost entirely missing in the input model, the automatic reconstruction can be incomplete (b).*

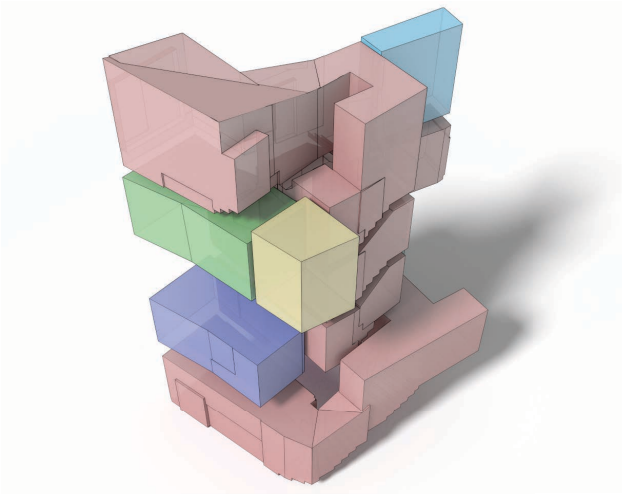
results in two restrictions. First, a room is detected only if it was acquired from at least one location inside it. Second, if two scans corresponding to the same (very large) room do not have enough overlap, the visibility-based clustering may assign them to different clusters, leading to over-segmentation. However, these issues only occur if the input scans do not cover the scene in an adequate way; the problem did not happen in any of the real-world datasets.

Thanks to its increased expressiveness, which allows to model a significantly larger array of real-world building interiors compared to other approaches, our method significantly advances the state-of-the-art in the field and represents a step forward towards the creation of a flexible and effective pipeline for the *scan-to-BIM* problem. There are however various aspects that can be further improved. Among the other, it would be interesting to explore a data-driven approach to extract a larger set of configurations for the structural components. In this context, the addition of higher-level primitives would be a straightforward extension. An even more important question is how to extend this work to process efficiently very large-scale and multi-story datasets. In particular, it would be useful to reduce the complexity of the 3D space partitioning structure created to support the reconstruction.

The approach presented in Chapter 5 focuses on this aspect, proposing a scalable solution to this problem based on detecting the individual rooms early on in the pipeline and on performing the reconstruction separately for each of them.

C H A P T E R 5

ROOM-BASED MODELING OF LARGE-SCALE ENVIRONMENTS



The previous chapter introduced a *full-3D* method capable of modeling environments with arbitrary orientations of its (planar) architectural surfaces. While capable of describing a significantly larger array of environments compared to the approach of Chapter 3, it relies on a global 3D space partitioning whose construction becomes prohibitively expensive when dealing with large environments. To cope with this problem, this chapter introduces a scalable pipeline that exploits the room structure of the environment to reconstruct each room separately, which allows to drastically reduce the complexity of the space partitioning built in the modeling.

5.1 Motivation and Background

The recent improvements in 3D scanning technologies have resulted in acquisition devices that are not only more accurate, but also extremely fast and capable of acquiring hundreds of thousands of samples per second. This makes it possible to digitize entire floors of buildings in a matter of hours, as we recently reported in the context of object classification [Mattausch et al., 2014]. As a result, practitioners working in the Architecture, Engineering and Construction (AEC) domain are faced more and more often with the problem of processing large-scale models of buildings. In this context, the sheer number of digital samples that compose the model is neither the only nor the biggest challenge to be addressed. In fact, much of the complexity of state-of-the-art *scan-to-BIM* pipelines does not arise from the number of samples in the input model, but rather from the architectural complexity of the environment to be reconstructed. This is strongly connected to the number and arrangement of the permanent structures that make up the building, as well as to the amount of furniture and other movable objects in the environment.

In particular, many modern modeling pipelines (including those previously presented in this thesis) rely on constructing a space partitioning structure around the input model. This allows to represent the space surrounding the target environment as a set of convex cells and to reconstruct it by detecting and merging the cells that represent the inner space. In the state-of-the-art, a single, global space partitioning structure is built for the whole environment, using the surfaces of the geometric primitives detected in the entire input model. When the input model represents a large building with dozens of rooms, the construction of this structure becomes prohibitively expensive and can represent a significant bottleneck for the whole modeling pipeline [Verdie et al., 2015]. This is especially true in the case of large buildings with many different wall orientations and irregular room arrangements, which leads to the detection of many distinct primitives, thus further increasing the cost of creating the space subdivision. Furthermore, an increased complexity in the space partitioning structure also makes the subsequent

steps of a reconstruction pipeline less efficient.

It is therefore necessary not only to have a scalable and efficient implementation for the construction of the space subdivision, but also to devise a pipeline that scales well with respect to the complexity of the input environment.

State-of-the-art So far, none of the indoor modeling approaches proposed has focused on scalability to large environments. Many pipelines incorporate specific steps to reduce the amount of input data to be considered; for instance, methods that work on 2D projections of the input model often discard samples whose neighborhood does not have one clear dominant direction, as they can not belong to the (vertical) wall structures of the environment [Turner and Zakhor, 2012; Oesau et al., 2014]. However, the problem of scalable processing is particularly pressing for the methods that consider the full 3D nature of the problem. Some researchers who propose using three-dimensional space partitionings for the reconstruction explicitly limit the applicability of their approach to small environments [Boulch and Marlet, 2014]. A number of pipelines adopt clever strategies to reduce the complexity of the data structures built for the reconstruction. These include techniques for discarding primitives that can not belong to the permanent structures to be recovered (as done also in this thesis, see Section 4.3) or specific heuristics that reduce the complexity of the data structures by restricting the spatial influence of the primitives used to build it [Chauve et al., 2010]. In the context of urban reconstruction, some researchers have proposed building a coarse approximation of the exact partitioning, deferring much of the complexity of the construction process to a later reconstruction step [Verdie et al., 2015]. Such strategies allow to greatly reduce the complexity of many large buildings. Nevertheless, since they are combined with the use of *global* space partitionings that cover the entire input model, their effectiveness eventually vanishes when its complexity exceeds a certain level.

Contribution This chapter describes an approach that exploits the room-based structure of building interiors to reduce the complexity of the computations and make the modeling process scalable – and parallelizable – with respect to the number of rooms in the environment. Like the approaches of Chapters 3 and 4, this method is robust to artifacts and occlusions and delivers an output reconstruction that is focused on the permanent components of the environments. However, differently from the two previous pipelines, the detection of the individual rooms is performed early on in the pipeline: this allows to construct a separate space partitioning structure for each room, using the sole geometric primitives that are relevant to the reconstruction of that room. As a result, large indoor environments can be processed efficiently and in a scalable manner. Moreover, since each space

partitioning is built using a limited set of primitives, it is possible to apply more conservative strategies for the detection of the architectural structures and for reconstructing the shape of each room from its associated space partitioning, thus leading to a more fine-grained final reconstruction.

5.2 Method Overview

The room-based modeling pipeline described in this chapter (illustrated in Figure 5.1) is based on the general scheme presented in Section 2.2.2, although there are some minor yet notable differences that should be explicitly highlighted.

First, while the input still consists in the adjacency graph of bounding rectangles G_{adj}, the underlying point-based model of the environment is represented as an *unorganized* point cloud (Equation 2.1). With respect to the formats used in the approaches of the previous chapters (Chapters 3 and 4), this representation is more generic, as it does not require that any information about the position and orientation of the scanning device at acquisition time (e.g. subdivision into individual scans, positions of the device) be preserved in the final model. Second, the order of the four main computation steps is altered by moving the detection of the individual rooms earlier in the pipeline. This allows to reconstruct each sub-environment separately and in parallel, thus greatly reducing the complexity of the space partitioning and allowing to cast the surface reconstruction task as a simpler *binary* segmentation.

The individual steps of the pipeline are shortly introduced in the remaining of this paragraph and more thoroughly described in the next sections.

Room detection by clustering of view probes A set of *view probes* are generated in the region of space surrounding the input model. Using the parts of the scene visible from each view probe, the probes are first classified as being either inside or outside the environment; then, those that are inside are clustered according to their visible surface overlap, so that probes that are in the same room are assigned to the same cluster. Finally, the rectangles of G_{adj} (i.e. the planar regions in the environment) are assigned to the clusters of view probes in a fuzzy manner.

Detection of candidate permanent components For each cluster of rectangles obtained by fuzzy assignment, the rectangles that are good candidates for representing permanent structures of the environment are selected. The approach follows the idea of the structural paths introduced in Section 4.3, but is more conservative in the pruning performed. Additionally, the rectangles corresponding to ceiling and floor regions are detected.

Construction of 3D space partitioning The permanent rectangles in each cluster are grouped based on their normal direction and on their offset along that di-

rection, using a variant of the PEARL algorithm [Isack and Boykov, 2012]. This procedure yields, for each cluster of permanent rectangles, the set of dominant planes of the corresponding room. Each set of dominant planes is used to construct a separate 3D space partitioning, one for each detected room.

Room reconstruction The regions of each space partitioning structure are classified into inner and outer space; the shape of each room is obtained by volumetric union of the inner regions. Since the large-scale environments targeted by this method can contain very large rooms, the initial clustering of the view probes can result in over-segmented rooms. For this reason, after the separate reconstruction of each room, a further step is applied to iteratively merge the rooms that should belong to the same environment.

It is important to notice that the last three of these steps (with the exception of the room merging computation during room reconstruction) can be performed independently and in parallel for each cluster discovered in the first step.

5.3 Room Detection by Clustering of View Probes

The goal of this step is to obtain a grouping of the rectangles of G_{adj} according to the room to which they belong. In order to achieve this, the detection of the rooms is performed early on in the pipeline. We follow the idea, already presented in Section 4.5, that clustering a set of locations inside the environment according to their visible surface overlap yields an indication of the spatial extent of the individual rooms. This approach does not assume that a set of inner positions (the scan positions) is given as part of the input data; instead, the very first step of the pipeline consists in extracting a set of locations that are likely to be inside the environment considered.

5.3.1 View Probes Generation

The positions at which the scanning device was placed during the acquisition process, often included in the input model (see Section 2.2), represent some samples of the space that is with certainty inside the environment. In fact, they are often optimal locations, chosen empirically when producing the input model to maximize the surfaces visible by the camera while ensuring the necessary overlap between different scans. When such locations can not be obtained from the input model (as in the case considered in this chapter), it is possible to generate a set of sample positions – denoted in the following as *view probes* – around the input model and to determine which are inside the environment by analyzing the properties of the surrounding scene.

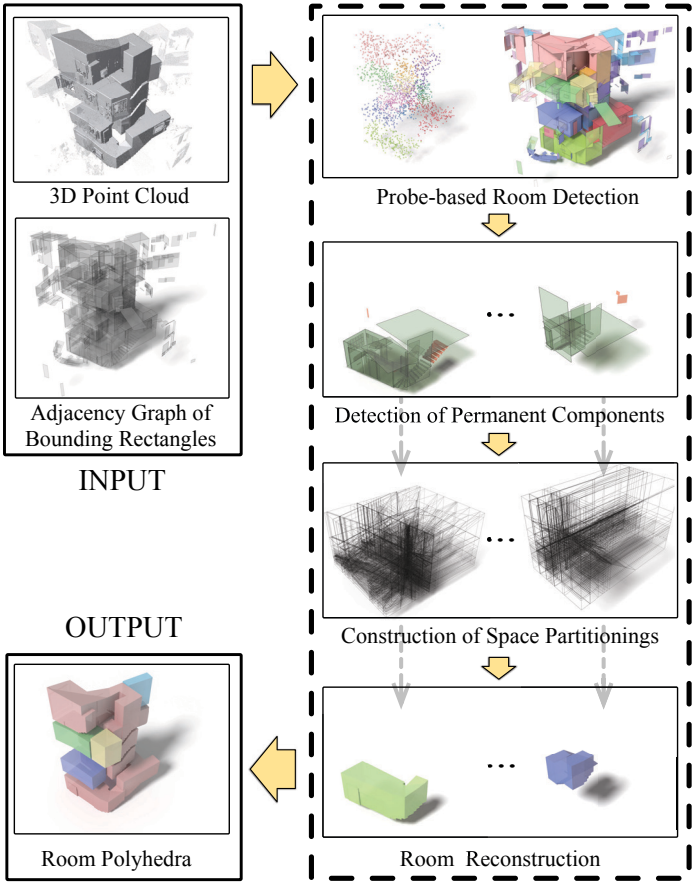


Figure 5.1: *Overview of our room-based modeling pipeline. Compared to the general pipeline of Figure 2.1, the room detection is performed first and the remaining steps are perform separately and in parallel for each room detected.*

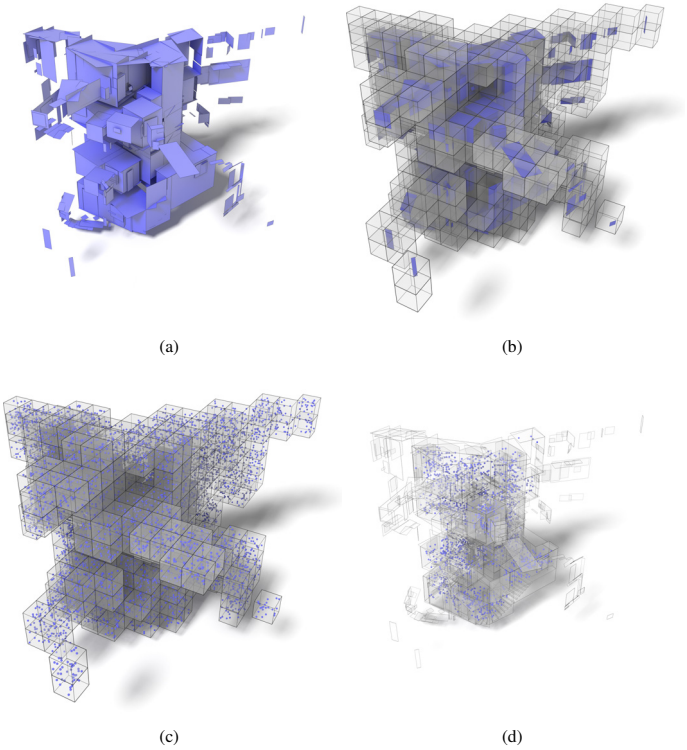
Many techniques for optimal viewpoint placement have been proposed throughout the years [Fleishman et al., 2000; Scott et al., 2003; Wenhardt et al., 2006]. For the approach proposed in this thesis, it is only important to ensure that the probes generated cover each sub-space of the environment and that there is sufficient visibility overlap between the probes. A straightforward approach to generate said probes is to build an axis-oriented regular grid on the bounding box of the input model and to use as view-probes the centers of the cells generated. The only parameter in this construction is the cell size s_{cell}, which controls the density of the view probes and could be adjusted depending on the expected minimum size of the rooms.

However, to avoid generating an overly-large number of probes in regions that correspond to outer space, we build instead an adaptive octree, locally refining it in the proximity of the bounding rectangles of G_{adj} (as shown in Figures 5.2(a), 5.2(b)). In particular, we consider two cell size values s_{cell} and $m \cdot s_{\text{cell}}$ (with $m \in \mathbb{N}$) that correspond respectively to levels l and l' of the octree, with $l' < l$. We refine the octree until level l, but extract the view probes only in the regions of space corresponding to the nodes $\mathcal{N}^{l'} = \{N_1^{l'}, \ldots, N_m^{l'}\}$ at level l' that contain at least one bounding rectangle. This ensures that the only regions considered for the generation of the probes are near the scanned surfaces of the environment.

To generate the final set of view probes, we consider the centers $\mathcal{C}$ of the children at level l of the nodes in $\mathcal{N}^{l'}$. The children nodes that actually contain a rectangle (that is, the actual leaf nodes of the adaptive octree) are discarded in this procedure, so as to avoid generating view probes that are close to a surface and therefore are not good representatives of the visible parts of a room. The view probes are extracted as a subset of $\mathcal{C}$ by using a sampling procedure, in which the nodes $N_1^{l'}, \ldots, N_m^{l'}$ are used as buckets. One sample is drawn from each bucket in a round-robin fashion, until the number of positions extracted matches a user-defined target number of probes N_{probes}. The results of this procedure for one of the test models used are shown in Figure 5.2(c). In our implementation, N_{probes} is set by default to 5% of the number of nodes at level l that are children of a node in $\mathcal{N}^{l'}$ and not empty.

5.3.2 Selection of Interior Probes

The probes generated in the previous step are placed in the vicinity of the bounding rectangles of G_{adj}. To select the probes that lie inside the environment we move from the fact that, in the ideal case, any location within a closed, man-made environment is bound completely by walls and other structures. In other words, such structures block the visibility from an inner location and are therefore the only elements that are visible.



(a) (b)

(c) (d)

Figure 5.2: *Generation of interior view probes for an input model. The input bounding rectangles (a) are embedded in an octree and the non-empty cells at a prescribed level (derived from the octree resolution) are selected (b). A set of view probes is extracted from each cell (c) by sampling the center positions of its leaf nodes. Finally, the view probes that are inside the environment (as represented by the bounding rectangles) are selected (d).*

To determine if a view probe lies inside the environment we therefore render the bounding rectangles (acting as proxies for the structures of the scene) around it and compute the fraction of the field-of-view that is occupied by the projections

of such rectangles. Due to occlusions and missing regions in the input data, some real-world bounding structures might not be represented by any rectangles, leading to gaps in the rendered scene that result in empty space or that make structures in other rooms visible. To ensure that only the relevant surfaces are considered in this process, only the projections of the rectangles that are front-facing with respect to the view probe are rendered. In this process, the back-facing rectangles are only used to ensure that the projected areas of the front-facing ones are blocked or reduced according to real-world occlusions; however, their projections do not appear in the renderings of the scene.

Using mathematical notation, denoting as $A_{\mathrm{FF}}^{p_{\mathrm{view}}}$ the visible area of the front-facing rectangles seen by a probe p_{view}, we classify p_{view} as an interior probe if $A_{\mathrm{FF}}^{p_{\mathrm{view}}} > \theta_{\mathrm{FF}} \cdot A_{\mathrm{vis}}^{p_{\mathrm{view}}}$, where $A_{\mathrm{vis}}^{p_{\mathrm{view}}}$ is the total area of the field-of-view of p_{view} and θ_{FF} is a user-defined threshold that we conservatively set to 0.75. Applying this criterion to the set of view probes yields a set of locations that lie inside the environment and cover all of its sub-spaces (see Figure 5.2(d)).

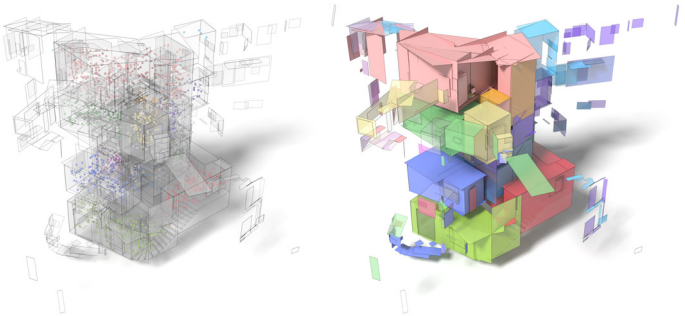
5.3.3 Room Detection using Interior Probes

The view probes classified as interior can be used to compute the approximate locations of the rooms in the environment. This can be achieved by clustering the interior probes according to their *visible surface overlap*, using the technique already introduced in Section 4.5. Note that, in this context, the visible overlap $\mathrm{overlap}(p_i, p_j)$ (see Equation 4.2) between two probes p_i and p_j is defined in terms of *all* bounding rectangles $\mathcal{R}$ of G_{adj}, thus without restriction to those corresponding to permanent structures.

This visibility-based clustering produces a set of clusters of view probes $\Gamma^{\mathrm{probes}} = \{\Gamma_1^{\mathrm{probes}}, \ldots, \Gamma_{n'_{\mathrm{rooms}}}^{\mathrm{probes}}\}$ (see Figure 5.3(a)). Depending on the amount of outliers and artifacts in the input data, some clusters might not correspond to real rooms, but rather to dense groups of points located outside the actual environment. For this reason, the number of view probes clusters n'_{rooms} is a provisional estimate of the number of rooms. Spurious rooms will be pruned at the reconstruction stage (see Section 5.6), resulting in the correct number of rooms n_{rooms}.

5.3.4 Fuzzy Assignment of Bounding Rectangles

The visible area of the bounding rectangles has been used in the previous step to cluster the view probes based on the common parts of the environment they see. Conversely, the visible area values vis_i^r for a given $r \in \mathcal{R}$ can be used to determine how likely it is for r to be relevant to the reconstruction of each room. In particular, let us define the visible area of a rectangle from a cluster of view probes $\Gamma_k^{\mathrm{probes}}$ as $vis_{\Gamma_k^{\mathrm{probes}}}^r = \max_{p_i \in \Gamma_k^{\mathrm{probes}}} vis_i^r$, that is, as the maximum visible



(a) (b)

Figure 5.3: *Room detection by clustering of interior view probes. The view probes that are selected as being inside the environment (see Figure 5.2(d)) are clustered according to their visibility overlap (a), which highlights the location of the individual rooms. Then, the bounding rectangles are assigned to the clusters of view probes in a fuzzy manner (b).*

area from all its view probes. Then, we define the probability $P_r(\Gamma_k^{\text{probes}})$ that a rectangle r belongs to a cluster of view probes Γ_k^{probes} as follows:

$$P_r(\Gamma_k^{\text{probes}}) = \frac{vis^r_{\Gamma_k^{\text{probes}}}}{\sum_{i=1}^{n'_{\text{rooms}}} vis^r_{\Gamma_i^{\text{probes}}}} \tag{5.1}$$

Based on such probability values it is possible to assign each bounding rectangle to one or more room clusters. Since the same rectangle can represent a permanent structure that spans several rooms, we perform a *fuzzy* assignment, allowing a rectangle r to be associated to more than one room. In particular, r is assigned to all clusters Γ_j^{probes} such that $P_r(\Gamma_j^{\text{probes}}) > \theta_{\text{fuzzy}}$. The value of θ_{fuzzy} controls the number of rectangles that are potentially used to reconstruct each room and represents a tradeoff between computational efficiency and safety. We used a value of 0.25 in our tests, thus considering a rectangle in the reconstruction of a room if the probability of it being relevant is higher than 25%.

The fuzzy assignment described above results in n'_{rooms} non-disjoint sets of bounding rectangles $\mathcal{R}^1, \ldots, \mathcal{R}^{n'_{\text{rooms}}}$, each contributing to the reconstruction of a room of the environment. An exemplar result of the fuzzy assignment can be seen in Figure 5.3(b).

5.4 Detection of Candidate Permanent Components

In this stage of the pipeline, each set $\mathcal{R}^i$ is analyzed to discard the rectangles that do not belong to permanent components of the environments. This follows the principle (introduced in Section 4.3) that the structural elements of building interiors are arranged in a consistent top-to-bottom arrangement, with ceiling structures unloading their weight onto the floor, transitioning through walls. However, with respect to the method based on structural paths (Chapter 4), we adopt here a more conservative approach that discards fewer rectangles.

As a first step, we detect in each cluster $\mathcal{R}^i$ the set of rectangles $\mathcal{R}^i_{\text{ceil}}$ that correspond to regions on the ceiling. This is done by first selecting the rectangles whose normals face downwards (up to an angular tolerance of $45°$) and then extracting the connected components among the selected rectangles. In this process, it is useful to discard connected regions whose area is too small, as they typically correspond to artifacts in the input data. Once the ceiling rectangles have been identified, we apply a simple region growing to the subgraph of G_{adj} corresponding to room i (denoted in the following as G^i_{adj}) that starts from the rectangles in $\mathcal{R}^i_{\text{ceil}}$ and only transitions from one node to another if the two nodes are adjacent in G^i_{adj}. The set of rectangles $\mathcal{R}^i_{\text{struct}}$ that are found in this process are regarded as good candidates for permanent components and are brought forward into the pipeline, while the others are discarded.

Compared to the more aggressive approach of Section 4.3, the number of primitives that are brought forward into the pipeline is significantly higher. However, thanks to the fact that each room is reconstructed separately, the computational complexity of the subsequent steps remains limited, thus allowing for a more conservative detection strategy. This reduces the chances of incurring in a misclassification error, which would prevent a structural surface to be represented in the spatial decomposition, and defers to the later global optimization step the selection of the true architectural surfaces of the environment.

Detection of floor levels Although only the ceiling rectangles are used to detect the candidate permanent components, the location of the floors can be used to guide the optimization-based reconstruction of the room shapes (Section 5.6). The set $\mathcal{R}^i_{\text{floor}}$ of the floor rectangles of room i can be detected by applying a strategy similar to that adopted for the ceiling rectangles; in this case, only the horizontal and up-facing rectangles are considered in the region growing.

Many of the regions found in this way do not correspond to real floor regions, but originate from pieces of furniture that have large horizontal surfaces, such as tables, beds or sofas. At this stage, we do not discard such rectangles and rather

opt for a soft detection of the floor levels. We move from the rationale that furniture always lies on the floor; for this reason, the rectangles in a horizontal region corresponding to an object, when orthogonally projected downwards, will overlap with the rectangles of an actual floor region. We therefore sort the horizontal regions detected according to their increasing height level; then, the rectangles in each region are projected onto the floor levels of the regions below. The regions that exhibit a significant projected area (at least 10% of their surface area) are considered as originating from furniture, while the others are marked as floor regions and their rectangles are added to $\mathcal{R}^i_{\text{floor}}$.

Detection of stairs In the same room, floor regions of different height are normally connected by stairs. This is obviously the case for staircases connecting separate rooms across different stories, but also happens for large halls. We recover the steps between the detected floor levels by performing a constraint region growing in G^i_{adj}, starting from each connected component in $\mathcal{R}^i_{\text{floor}}$ and only expanding towards valid adjacent rectangles. The rules for growing the region mimic those defined by Boulch and colleagues in their grammar-based semantic analysis of buildings [Boulch et al., 2013]. In particular, we consider a step as composed by a vertical (the *riser*) and a horizontal (the *tread*) rectangle, arranged in a sequence that spatially expands upwards. Since stairs are typically empty and not cluttered with objects, the rectangles that are considered in this process can only belong to actual steps, to walls, or to other floor levels. We terminate the growing when another floor level is reached and we distinguish between a riser and a vertical wall by means of their height (which for a riser can not exceed a threshold of $h_{\text{riser}} = 0.25m$ [Boulch et al., 2013]).

Due to missing data, some steps in a staircase might be missing, thus breaking the expected sequence of steps. To add robustness with respect to this issue, we modify the plain scheme described above in two ways. First, we consider for each level not only the ascending, but also the descending sequences of steps (discarding possible duplicates). Second, we relax the requirement that a step must be composed of a riser and a tread and continue the growing even in the presence of only one of these elements. Since we heuristically discovered that the regions corresponding to risers are more often missing in real-world point clouds (in particular, in those obtained by static laser scanning), for each tread detected we explicitly look for a corresponding adjacent vertical riser. If this is not found in G^i_{adj}, we generate one by extruding downwards the edge of the tread rectangle that is spatially furthest from the previous steps in the detected sequence. Such *ghost* primitives, which represent hypotheses about plausible structures not detected during the acquisition, are used also in previous work [Chauve et al., 2010] and help obtain a plausible reconstruction in the presence of missing data.

The rectangles corresponding to treads and risers are added to $\mathcal{R}^i_{\text{struct}}$, thus contributing to the construction of the space partitioning for room i. Moreover, the tread rectangles are added to $\mathcal{R}^i_{\text{floor}}$ and contribute to defining the levels of the floors later used in the room reconstruction.

5.5 Construction of Space Partitionings

Each cluster of rectangles $\mathcal{R}^i_{\text{struct}}$ extracted in the previous step contains planar primitives that are likely to lie on the permanent structures of a room. In this stage of the modeling pipeline, a set of *dominant planes* is extracted from each cluster and is used to build a subdivision of the space into volumetric cells. This mimics the approach described in Section 4.4. However, since each set of rectangles is relative to a single room of the environment, the number of primitives in each cluster $\mathcal{R}^i_{\text{struct}}$ is significantly lower than the total number of bounding rectangles $\mathcal{R} \in G_{\text{adj}}$. This allows to apply a less aggressive strategy for the computation of the dominant planes, thus allowing to recover even more fine-grained orientations of the architectural surfaces of the building.

5.5.1 Computation of Dominant Planes

The goal of the extraction of the dominant planes is to group together the rectangles in the adjacency graph G_{adj} that originate from the same architectural surfaces, corresponding to unbounded 3D planes. As explained in Sections 3.4.1 and 4.4, they can be computed from the structural rectangles in each cluster $\mathcal{R}^i_{\text{struct}}$ by first clustering them based on their orientation (i.e., on their normal direction) and then clustering each cluster obtained according to the offset along that direction.

Compared to the technique used in the previous two chapters, we propose here a more conservative approach in which both clustering steps are cast as optimal labeling problems, following the general scheme of the PEARL algorithm [Isack and Boykov, 2012]. In particular, we seek to assign each rectangle the label of a model that is optimal with respect to a data term E_{data} (which measures how well the model fits the rectangle) and to a regularization term E_{reg} based on label costs (which aims to reduce the number of models used to explain the set of input rectangles). After the optimal labeling has been computed, the clusters are composed of the groups of rectangles that have been assigned the same label.

In our formulation, we consider a set of input bounding rectangles $\mathcal{R}_{\text{in}} = \{r^i_1, \ldots, r^i_{n^{\text{input}}}\}$ and seek to assign each of them the label of the model that best describes it. The set of models $\mathcal{M} = \{M_1, \ldots, M_{n^{\text{input}}}\}$ is initialized with the models corresponding to the input rectangles $\mathcal{R}_{\text{in}}$; moreover, each rectangle r_j

is initially assigned a label l_{M_j} corresponding to its model M_j. The optimal assignment of models to the rectangles is obtained with an optimization-based procedure, in which two steps are repeated until convergence: computation of the optimal label assignment; re-estimation of the set of models $\mathcal{M}$.

At each step, the optimal labeling $X^* = \{ x_1^*, \ldots, x_{n^{\text{input}}}^* \mid x_i^* \in \{l_{M_j} | M_j \in \mathcal{M}\} \}$ is obtained using the alpha-expansion algorithm with label costs [Delong et al., 2012] to minimize the following energy function:

$$E(X) = E_{\text{data}}(X) + \lambda_{\text{reg}} \cdot E_{\text{reg}}(X) \; . \tag{5.2}$$

The data term $E_{\text{data}}(X)$ is expressed as a sum of individual sub-terms, each measuring the error of fitting a model to a rectangle:

$$E^{\text{data}}(X) = \sum_{r_i \in \mathcal{R}_{\text{in}}} D_{\text{err}}(r_i, x_i) \; . \tag{5.3}$$

The regularization term $E_{\text{reg}}(X)$ is also defined as a sum of sub-terms:

$$E_{\text{reg}}(X) = \sum_{l \in \{l_{M_j} | M_j \in \mathcal{M}\}} c_l \cdot \delta_l(X) \; . \tag{5.4}$$

Here, $\delta_l(X)$ is an indicator function that equals 1 if l appears in the label assignment X and 0 otherwise, while c_l is a penalty factor for using a label $l \in \{l_{M_j} | M_j \in \mathcal{M}\}$. We defined $c_{l_{M_j}}$ as the inverse of the number of points in the input point cloud that are explained by model M_j, that is, the number of points that represent the inliers of the rectangles associated with label l_{M_j}. This ensures that the labels corresponding to models that explain many of the input points are not penalized and appear in the assignment X.

As a first step in the extraction of the dominant planes, the procedure sketched above is applied to the entire set of rectangles $\mathcal{R}^i_{\text{struct}}$ (i.e. $\mathcal{R}_{\text{in}} = \mathcal{R}^i_{\text{struct}}$). To cluster the rectangles according to their orientation, each model $M_j \in \mathcal{M}$ in this step corresponds to a three-dimensional vector $\mathbf{d}_{M_j} \in \mathbb{R}^3$ (representing the normal of a 3D plane). The set of models $\mathcal{M}$ is initialized with the normals of the rectangles $\mathcal{R}^i_{\text{struct}}$. To evaluate the error of assigning a model M_j to a rectangle r_i with normal $\mathbf{n}_{r_i}$, we define $D_{\text{err}}(r_i, x_i)$ as follows:

$$D^{\text{dir}}_{\text{err}}(r_i, x_i) = 1 - \mathbf{n}_{r_i} \cdot \mathbf{d}_{M_j} \qquad x_i = l_{M_j} \tag{5.5}$$

The directional clustering yields a set of clusters that contain rectangles with a coherent orientation. The items in each cluster undergo a second clustering step, to extract the offsets along the common direction at which the dominant planes are located. In this case, each model $M_j \in \mathcal{M}$ is a scalar value $o_{M_j} \in \mathbb{R}$ that corresponds to an offset from the origin of the reference system. The function

D_{err} is defined simply as the distance between o_{M_j} and the distance from the origin of the plane of r_i:

$$D^{\text{off}}_{\text{err}}(r_i, x_i) = |o_{M_j} - \text{offset}(r_i)| \qquad x_i = l_{M_j} \tag{5.6}$$

where $\text{offset}(r)$ denotes the distance between the plane of r and the origin.

In each iteration of the clustering, the computation of the optimal label assignment is followed by the update of the set of models. For each group of primitives assigned to a same label, the updated model is computed by performing an iterative weighted sum of the parameters of the primitives' models. An initial model is computed by using as weight the number of inlier points of each primitive; then, in each iteration, this weight is further scaled by the distance between the current model and the model of the primitive, using the distance definitions presented in Equations 5.5 and 5.6. To increase the robustness of the procedure, each distance is further weighted to reduce the influence of clear outliers. To this purpose, we used a simple monotonically decreasing function defined as $f(x) = 1 - x^4$.

Each group of rectangles computed with the offset clustering stems from a dominant plane of the environment. Its parameters (i.e. the normal vector and an anchor point) are obtained using the same procedure employed to compute the initial model during the models update step. We heuristically discovered that this basic approach does not result in significant averaging effects. This is because, in our implementation, we assign a low importance to the regularization term, resulting in final clusters that contain only rectangles lying on very similar planes. The set of 3D planes computed with this procedure form the dominant planes $\Pi^i = \{ \Pi^i_1, \ldots, \Pi^i_{n'_{\text{rooms}}} \}$ of room i and are used to compute the space partitioning for that room.

Note that, thanks to the early room detection and to the fuzzy, room-based assignment of the bounding rectangles, the computation of the dominant planes can be applied separately to each $\mathcal{R}^i_{\text{struct}}$, without having to consider the whole set of bounding rectangles $\mathcal{R}$ of G_{adj}. This makes the processing of even large environments tractable. Moreover, thanks to the limited number of primitives in each input set $\mathcal{R}^i_{\text{struct}}$, it is possible to perform a conservative grouping of the rectangles, which allows for a more fine-grained computation of the dominant planes. This means, for instance, that the weight λ_{reg} of the regularization term can be set to a low value (e.g. 0.05 in our experiments) to avoid merging distinct dominant planes into a single one.

5.5.2 Construction of Space Partitioning Structures

Each set of dominant planes $\Pi^i = \{ \Pi^i_1, \ldots, \Pi^i_{n'_{\text{rooms}}} \}$ is used to construct a separate 3D cell complex $(\mathcal{C}^i, \mathcal{N}^i)$ for each room of the environment. The type of

space partitioning structure, as well as the construction process, follow closely the scheme already presented in the previous chapter (see Section 4.4). It is important to stress here that each facet $f_{c,c'}$ that lies on the boundary of polyhedral cells $c, c' \in \mathcal{C}^i$ is associated with the ID of the dominant plane that generated it and with a measure of the *coverage* of the facet. This quantity, which corresponds to the fraction of the area of $f_{c,c'}$ that is covered by scanned points, is used as a regularizer in the subsequent optimization-based room reconstruction.

5.6 Room Reconstruction

In the previous computation stage, the space surrounding each detected room is partitioned into a set of convex polyhedral cells, arranged in a 3D cell complex. The goal of the room reconstruction step is to extract from each cell complex the (connected) set of cells that correspond to the volume inside the boundaries of the room. Thanks to the early room detection, in this approach the room reconstruction can be cast as a classical inside/outside partitioning problem (typical of urban reconstruction settings [Musialski et al., 2013]), for which fast and exact global optimization techniques exist. This differs from the multi-label formulation used in the *full-3D* modeling pipeline in Chapter 4, which is solved using an approximate optimization algorithm.

The binary segmentation of each room cell complex $(\mathcal{C}^i, \mathcal{N}^i)$ is performed by assigning each cell $c \in \mathcal{C}^i$ one of the two labels $\{0, 1\}$, which correspond, respectively, to the outer space and to the space inside the room. The optimal labels assignment $L^* = \{\ L_c^* \mid L_c^* \in \{0,1\}\ ,\ c \in \mathcal{C}^i\ \}$ is obtained by minimizing the following energy function:

$$E_{\text{label}}(L) = E_{\text{data}}(L) + E_{\text{smooth}}(L)\ . \tag{5.7}$$

Note that Equation 5.7 has the same general form of Equation 4.4, but is defined over binary variables and hence differs in the definition of the specific terms.

5.6.1 Data Term

The data term $E_{\text{data}}(L)$ consists in a sum of unary functions $D_c(L_c)$, one for each cell $c \in \mathcal{C}^i$:

$$E_{\text{data}}(L) = \sum_{c \in \mathcal{C}^i} D_c(L_c) \tag{5.8}$$

With respect to the formulation of Section 4.6, the difference lies in the definition of the potentials $D_c(L_c)$. Since in this context we perform a *binary* labeling, each potential can be defined simply in terms of the quantity $A_{\text{FF}}^{p_c}$, already introduced in the context of the selection of interior view probes (see Section 5.3.2) and denoting

the fraction of the scene (as seen from the center of c) that is occupied by the projection of front-facing structural rectangles:

$$D_c(L_c) = \begin{cases} A_{\mathrm{FF}}^{p_c} & \text{if } L_c = 0 \\ 1 - A_{\mathrm{FF}}^{p_c} & \text{if } L_c = 1 \end{cases} \tag{5.9}$$

This definition increases the energy when $A_{\mathrm{FF}}^{p_c}$ is close to 1 and the cell c is given the label 0, that is, it penalizes labeling a cell as outside if most of the space surrounding the center of c (denoted by p_c) is occupied by bounding rectangles. Conversely, when $A_{\mathrm{FF}}^{p_c}$ is high, assigning c the label 1 does not increase the energy significantly; this labeling is therefore favored. Similar arguments, with reversed label values, hold for low values of $A_{\mathrm{FF}}^{p_c}$.

5.6.2 Smoothness Term

The goal of this term is to regularize the initial guesses represented by the values in the data term, favoring room shapes that are geometrically simple and that adhere to common structural characteristics of real-world buildings. Similarly to the original *full-3D* approach of Chapter 4, this is achieved by including several regularizing sub-terms in the smoothness term $E_{\mathrm{smooth}}(L)$:

$$E_{\mathrm{smooth}}(L) = \lambda_{\mathrm{cov}}\, E_{\mathrm{cov}}(L) + \lambda_A\, E_A(L) + \lambda_G\, E_G(L) \tag{5.10}$$

Each term contributes to the regularization of the reconstruction in different ways. In particular, the coverage term E_{cov} and the area term E_A favor, respectively, room models with boundary walls densely covered by scanned point samples and that are geometrically simple, whereas the gravity term E_G penalizes label assignments that leave a cell without adequate support from an underlying cell. In our experiments, we balanced the contribution of these three regularizers by setting $\lambda_{\mathrm{cov}} = 0.2$, $\lambda_A = 0.05$ and $\lambda_G = 0.4$. It is worth noting that, compared to the formulation of Chapter 4, we increased the weight of λ_G; this has the effect of removing surfaces corresponding to many cluttering objects (e.g. beds, desks) within the optimization, thus compensating for the absence of the filtering step based on structural analysis (see Section 4.3).

More details on the definition of these terms are given in Section 4.6, where they were first introduced. Note, however, that in this formulation the function $u(f_{c,c'})$ (used in E_G to only restrict the action of the gravity term to facets that do not lie on a floor and are therefore potentially unstable), takes into account all the floor levels found using the approach of Section 5.4, including those corresponding to the treads of the steps.

The optimal inside/outside labeling of the cells $\mathcal{C}^i$ is obtained by minimizing the energy function described in Equation 5.7 using a formulation based on

graph-cuts [Kolmogorov and Zabin, 2004]. The 3D surface model of the room considered is then given by the facets that separate adjacent cells with different labels.

It is worth noticing here that the reconstruction (as well as the selection of candidate permanent rectangles and the construction of the space partitioning structure) is performed separately for each room detected in the first stage of the algorithm (i.e. room detection by clustering of view probes). As mentioned in Section 5.3.3), some of the clusters of view probes found at that stage can correspond to fake rooms and originate from spurious primitives. The reconstruction handles these cases indirectly, since it consistently produces an empty room model when applied to a candidate room that stems from a spurious cluster of view probes. By discarding such invalid reconstruction results we obtain the set of n_{rooms} actual sub-spaces of the input environment.

5.6.3 Merging of Reconstructed Rooms

Large-scale building interiors often contain sub-environments like corridors and staircases with complex shapes and often spanning multiple stories. Since the definition of room adopted in this work is based on the consistency of the visible scene, it can happen that such connecting sub-environments are split into multiple rooms. Given the n_{rooms} reconstructed room models, we apply a simple post-processing strategy to iteratively merge adjacent and overlapping room models.

We encode the possible adjacency relationships between the rooms using a complete undirected graph, in which the nodes represent the reconstructed room models and the edges connect spatially adjacent rooms. We sparsify this complete graph by considering the bounding boxes of the room models and by removing the edges that connect two rooms whose bounding boxes do not overlap. We sequentially consider each edge of the graph and verify if the corresponding room models intersect. If not, the edge is removed from the graph; otherwise, the edge is contracted and the new node is associated with the union of the two models corresponding to the nodes removed.

This process is repeated until all edges in the graph have been removed. The models corresponding to the remaining nodes represent the final result of the reconstruction.

5.7 Results

Since the main advantage of this pipeline over the one of Chapter 4 lies in the scalability to large and complex environments, the test suite used in the evaluation is restricted to 2 real-world models specifically chosen to highlight its main

capabilities. In particular, one model ('Building D') includes a high number of rooms in relatively regular arrangement, while the other ('Tower') consists of a long staircase attached to several rooms and spanning multiple floor levels with a highly irregular structure. Table 5.1 provides detailed information about the reconstruction process, while additional details on the datasets can be found in Appendix A.

Implementation The proposed pipeline was implemented in C++, using the threading API included in the Standard Template Library (STL) for the parallel computations. The software uses the publicly available implementations of the MCL algorithm [Van Dongen, 2008], of the multi-label optimization with label costs [Delong et al., 2012] and of the graph-cut-based energy minimization framework [Kolmogorov and Zabin, 2004]. The tests were performed on a MacBook Pro with an Intel Core i7 (2.5GHz), 16GB DDR3 RAM and an NVIDIA GeForce GT 750M. For the sake of practicality, we used the GPU for visibility-based computations (similarly to what described in Section 4.7.) and to compute the visible surface overlap between interior view probes (see Sections 5.3.3 and 4.5). As shown in column 'Time' of Table 5.1, the overall processing times for the two models 'Tower' and 'Building D' are about 2 and 6 minutes, respectively; these timings include the computation of the planar abstraction (see 2.2.2). It should be noted that in the case of 'Building D' the timings are dominated by the room detection step. This is due to the specific implementation used for the visibility-based computations and to the high number of interior view probes generated with our conservative and unoptimized approach; as mentioned in Section 5.3.1, more specialized state-of-the-art methods can be used to generate a reduced set of optimal view locations. This, however, lies outside the scope of this thesis.

Since the test machine used has 4 individual cores and supports the concurrent execution of up to 8 threads (with hyperthreading), exactly 8 threads were used in the steps involved in the reconstruction of the individual rooms (Sections 5.4, 5.4 and 5.6, except 5.6.3), which can be performed in parallel on each detected room. Note that, since the computation of the data terms (Section 5.6.1) is based on the graphics pipeline, we serialize the execution of this section.

Finally, we report the parameter values used in the planar abstraction, which were set to $\theta_{\text{off}} = 1.0$cm, $\theta_{\text{ang}} = 0.925$ and $\theta_{\text{adj}} = 20$cm.

Correctness of reconstruction The reconstruction results for the two test models are shown in Figure 5.4. The model 'Tower' represents a multi-story building composed of a single staircase that develops across all stories and connects several separate rooms. This building is a particularly interesting test case because of its architectural complexity: first of all, the vertical extent of the rooms

Dataset	#Rect	#VP Clusters	#Rooms	Time (abstr./detect./merg.)
Tower	827	14	7 (13)	125.6s (26.1s/45.2s/3.0s)
Building D	1180	26	25 (26)	367.8s (12.9s/299.0s/5.8s)

Table 5.1: *Main statistics for the room-based reconstruction. From left to right: no. of bounding rectangles in the adjacency graph; no. of view probes clusters; no. of rooms after and, in brackets, before merging; overall computation time and, in brackets, partial timings for planar abstraction, room detection (Section 5.3) and room merging (Section 5.6.3)*

is overlapping, which makes the detection of the individual floor levels ambiguous; secondly, the (planar) wall surfaces have irregular orientations, thus clearly violating the 2.5D assumption; finally, it includes a large subspace with a highly irregular structure, consisting of a staircase with attached small halls and extending vertically across several floors. Despite this complexity, all the rooms in the environment are detected and successfully reconstructed, also preserving geometric details such as individual steps and fine-grained ceiling orientations. It should be noted that the central space containing the staircase is first reconstructed as a set of 7 separate rooms and then correctly merged into a single one. This is coherent with the fact that a room is considered to be an environment with a similar visibility (see Section 4.5) and is functional to the goal of reducing the complexity of the reconstruction process. As a secondary detail, note that the cyan room on the top floor represents a partial reconstruction of the actual room of the building. In fact, that room was skipped during the acquisition and the only evidence of it in the input data consists in sparse measurements acquired through a door opening. In this case, the pipeline produces the best reconstruction that is coherent with the incomplete input data. Note that sparser groups of points (e.g. mirrorings of actual room structures) might be identified as a separate room cluster in the room detection step (Section 5.3), but originate an empty model at the reconstruction stage (Section 5.6) and are thus easily pruned.

The second model considered is an unorganized point cloud extracted from 'Building D' (already used to evaluate the *full-3D* pipeline); given the high number of rooms, this model is a good test case to evaluate the scalability of the proposed approach. As shown by the final result, the room-based pipeline produces a correct reconstruction of the environment. With respect to the model obtained with the standard *full-3D* pipeline (shown in Figure 4.10), it is worth considering the higher geometric detail obtained, noticeable in particular in the window alcoves and in the protrusions on the corners of some rooms. The recovery of such details is made possible by the more conservative filtering of the bounding rectangles performed in the room-based pipeline. A minor difference is represented by

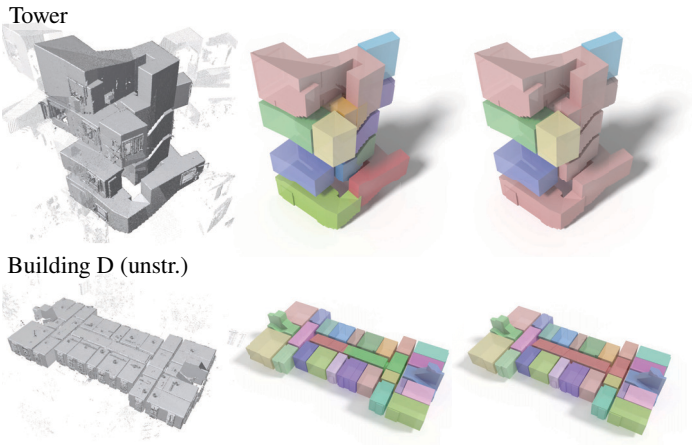


Figure 5.4: *Reconstruction results of our room-based modeling pipeline on datasets 'Tower' and 'Building D'. For each dataset, we show the input point cloud (LEFT) and the room reconstruction before (MIDDLE) and after the merging step (RIGHT).*

the central corridor: in the results produced by the room-based approach, a single room (shown in red in Figure 5.4) is obtained, corresponding to the union of two separate rooms produced by the standard *full-3D* pipeline. As already noted in Section 4.7, since the connections between this corridor and the adjacent rooms consist in very wide passages, both solutions can be considered correct.

Scalability of space partitioning construction Since reducing the complexity of the space partitioning is a key feature of the proposed room-based method, we specifically evaluate this point under three aspects.

First, we consider the complexity of building the cell complexes using the proposed room-based approach. We use the number of cells as an indicator of the complexity of the structures and relate this to the number of dominant planes used to build the space partitioning. The bar plots in Figure 5.5 show, for the two models 'Tower' and 'Building D', the size (i.e. number of cells) of all the cell complexes built (Figures 5.5(a) and 5.5(c)), as well as the number of dominant planes and bounding rectangles involved in their construction (Figures 5.5(b) and 5.5(d)). It can be easily seen that reducing the number of primitives input to the cell complex construction process helps keep the size of the resulting struc-

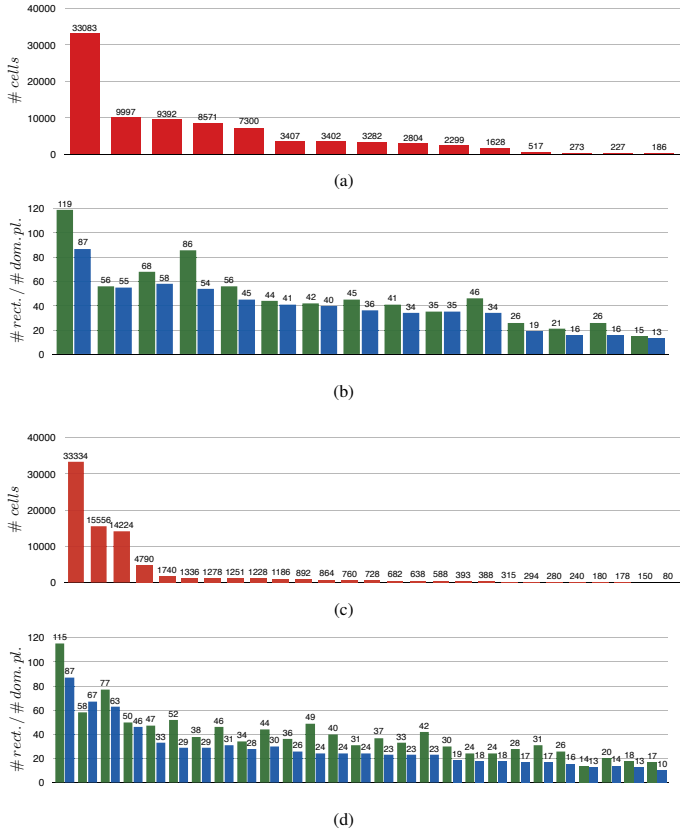


Figure 5.5: *Complexity of the cell complex construction in our room-based pipeline for the two models 'Tower' ((a) and (b)) and 'Building D' ((c) and (d)). In all figures, each vertical bar corresponds to a different cell complex. Plots (a) and (c) indicate the number of cells of the cell complexes; plots (b) and (d) describe the number of dominant planes (blue) used in their construction, together with the number of bounding rectangles from which the dominant planes are extracted (green).*

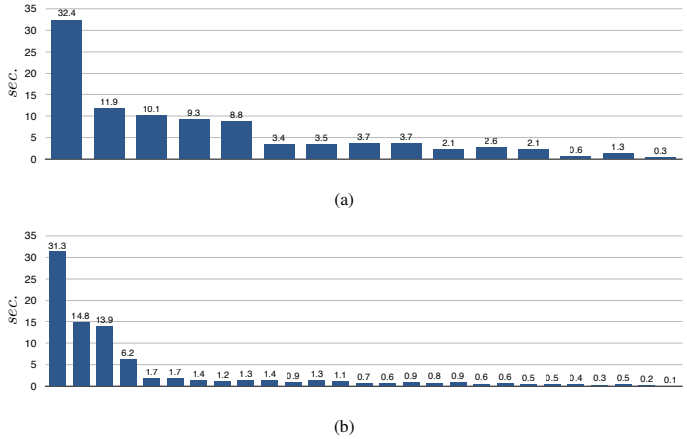
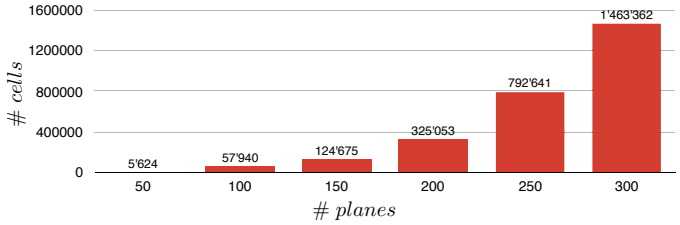


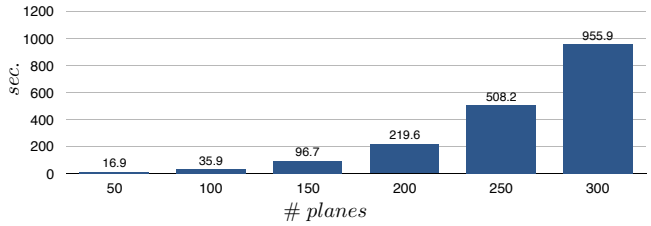
Figure 5.6: *Computation time for the construction of the cell complexes for the two models 'Tower' (a) and 'Building D' (b). Each vertical bar corresponds to a different cell complex; the order is the same as in Figure 5.5.*

ture to a manageable level. In particular, it should be noted that the size of the largest complexes is comparable to that of those built with the approach of Chapter 4, which employs a less conservative strategy to select the candidate permanent components. It would be interesting to compare the size of the cell complexes built using the room-based partitioning strategy with that of the global complex constructed using all dominant planes of the model; however, the construction of such a global structure exceeded the memory capabilities of our test machine. Nevertheless, we report the number of dominant planes input to the construction process, which amounts to 602 for 'Tower' and and 624 for 'Building D'. Note that, in the case of 'Tower', the construction was aborted after the insertion of the first 460 planes. An additional analysis of how the complexity of the cell complex grows when increasing the number of dominant planes is shown in Figure 5.7(a).

Second, in addition to considering the size of the cell complexes built, we measured the computing time needed for their construction. For these measurements, all computations were performed using a single thread. The detailed timings (shown in Figures 5.6(a) and 5.6(b)) demonstrate that the time complexity of the cell complex construction is aligned with that of the other steps of the pipeline (see Table 5.1) and that, thanks to the proposed room-based approach, it does not



(a)



(b)

Figure 5.7: *Analysis of the scalability of the cell complex construction. The plot shows the number of cells (a) and the computing time (b) for the construction of a mock cell complex from an increasingly large set of input 3D planes.*

represent a computational bottleneck. More importantly, these timings are drastically lower than those needed for constructing a global space partitioning using all dominant planes. On the user-level hardware used in our experiments, the partial (up to the first 460 planes) construction of the cell complex for 'Tower' took over 8 hours. Since the time complexity for the computation of the cell complex from a set of n 3D planes is $O(n^3)$ [Edelsbrunner et al., 1986], this strongly suggests the unfeasibility of a single space partitioning for the whole environment. To further confirm this indication, we constructed a mock cell complex by selecting an increasingly larger subset of the 602 dominant planes for the whole model 'Tower'. The results, presented in Figure 5.7(b), clearly highlight that the construction of the cell complex is prohibitively expensive even for a few hundreds planes, requiring 16 minutes for as few as 300 planes.

Third, we considered the steps involved in the room reconstruction (except the room merging) and evaluated the speed-up of the parallel execution over

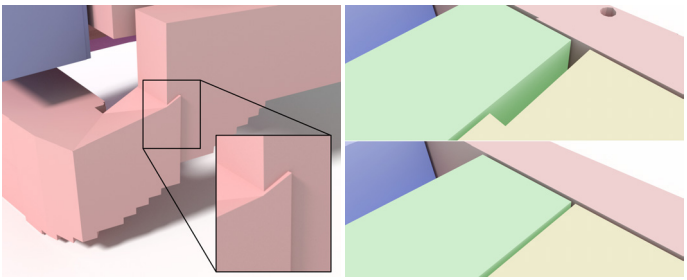
single-threaded computation. Using 8 threads on a 4-cores processor (see above), our parallel implementation takes 51.2 and 50.0 seconds for models 'Tower' and 'Building D', respectively. This corresponds to a speed-up of 2.1x and 1.75x over the timings of the serial execution (106.6 and 87.5 seconds).

5.8 Discussion and Outlook

This chapter described a scalable approach to modeling large-scale multi-room building interiors that exploits the subdivision into separate rooms to drastically reduce the complexity of the computations and to make the reconstruction of very large environments feasible. With respect to the method introduced in Chapter 4, this room-based approach does not rely on an aggressive pruning of the input primitives to ensure the feasibility of the reconstruction. On the contrary, it employs a conservative approach when discarding the parts of the environments that are not relevant to the reconstruction and when extracting the dominant planar surfaces that explain its architectural shape. The feasibility of the computation is achieved by exploiting a basic property of all buildings – that is, their subdivision into separate sub-environments.

While considering each room separately effectively solves the problem of handling large building interiors with many different rooms, this approach also has some limitations. First of all, since we rely on the subdivision into rooms to reduce the complexity of the reconstruction, the proposed strategy will not result in any improvement of performance for interiors consisting of a single, large and complex environment (identified by the room detection step as a single room). Secondly, since the optimization-based reconstruction is performed separately for each room, the action of the regularizing terms is restricted to the 3D model of that room and does not have any effect on the model resulting from the final merging (see Figure 5.8(a)). A further optimization step that uses the original point cloud could be added to the pipeline to improve the final output. Finally, the room merging approach proposed can fail if two reconstructed rooms that should be joined do not have enough overlap. These cases could be handled effectively and simply with an interactive, post-processing step, in which a user can trigger the fusion of two adjacent rooms by drawing a sketch that connects their boundaries. Finally, the pipeline proposed in this chapter employs a very conservative approach to remove non-permanent elements and relies on the action of the gravity term to ensure that they do not affect the final reconstruction. In some borderline cases (e.g. in the presence of very tall cabinets, as shown in Figure 5.8(b)), this approach might produce sub-optimal results and should be complemented by the pruning based on structural paths (Section 4.3).

Despite such disadvantages, the proposed pipeline represents the first solution



(a) (b)

Figure 5.8: *Some limitations of our room-based approach. Since the regularization acts on each reconstructed room separately, seams may arise after the room merging at the boundaries between rooms (a). Ambiguous furniture configurations such as the presence of a very tall cabinet leaning on a wall can affect the reconstruction ((b), TOP); in such cases, the selection of candidate permanent components based on structural paths (used in Chapter 4) allows to obtain the expected results ((b), BOTTOM).*

to the problem of handling in a scalable manner large-scale indoor environment and paves the way to further developments in the context of indoor modeling. First and foremost, since it drastically reduces the complexity of the space subdivision structures created, it allows to use more complex (and computationally expensive) optimization techniques to extract the final surface model. These include the regularization terms based on edges and corners, initially proposed by Boulch and colleagues [Boulch et al., 2014] and capable of delivering superior reconstruction quality. Moreover, the cell complexes generated are of a manageable size despite the fact that they include subdivisions that arise from non-permanent structures; this allows to explore new formulations that integrate the separation between permanent components and furniture directly in the final optimization step, potentially leading to improved reconstruction results. Another interesting avenue for future work is to combine the room-based approach presented in this chapter with optimized construction techniques for the space partitioning structure, along the lines of what already suggested in Section 4.8.

CHAPTER 6

CONCLUSIONS

6.1 Summary

Fostered by the widespread diffusion of 3D acquisition devices, the modeling of building interiors has become an increasingly researched problem, with many applications that promise to revolutionize the way we explore and organize indoor environments.

This thesis addresses some of the most pressing challenges that are left open by state-of-the-art methods. In particular, our work focuses on five research goals: achieving robustness to artifacts and occlusions in the raw input data; extracting the shape of an indoor environment as defined by the permanent components; detecting the individual rooms of the environment and integrating this computation in the reconstruction pipeline; correctly modeling environments with general wall orientations (that is, not conforming to the Manhattan-World and 2.5D assumptions); processing large-scale environments composed of multiple rooms in an efficient and scalable manner.

These specific research objectives are achieved with three research contributions, presented in this thesis in order of increasing complexity and generality. Each of them is focused on a processing pipeline that extracts an accurate, room-aware 3D description of an indoor environment from its input representation.

In the context of this thesis, 3D point clouds are considered as the initial input representation for the environments to be modeled, since this data type is widespread in the computer graphics and vision domains and because it is a common output format of the 3D acquisition systems commonly used by practitioners. As described in Chapter 2, we convert this low-level and highly redundant representation into a more compact and higher-level adjacency graph, which encodes the planar regions of the scene as bounding rectangles and arranges them according to their adjacency relations. Such a structure allows for efficient processing and simplifies reasoning on the architectural properties of the environment that are relevant to the modeling task. For this reason, it is provided as input alongside the original 3D point clouds to all the proposed pipelines.

As a first research contribution, this thesis introduces an indoor modeling approach that incorporates the recognition of individual rooms into the reconstruction. The approach (described in Chapter 3) lifts the Manhattan-World assumption, used by a large number of state-of-the-art methods, and only assumes that the target environment has a 2.5D structure (that is, that walls are vertical and that ceilings and floors are horizontal). Starting from the initial adjacency graph of bounding rectangles, the method first selects rectangles that are likely to belong to walls using an occlusion-aware computation, which is able to detect rectangles corresponding to wall surfaces even if a significant amount of their vertical extent is occluded by other objects. The candidate wall rectangles selected in this way are then projected vertically onto the plane of the ground and the room detection

problem is cast as the task of partitioning a 2D cell complex built on the floorplan of the environment. An iterative binary clustering driven by diffusion distances is used to simultaneously detect the individual rooms and to extract their floorplans, from which the 3D polyhedra describing the shapes of the rooms are finally reconstructed.

The 2.5D modeling is then extended to handle the more general case of environments with a piecewise-planar shape but with arbitrary orientations of the wall structures, thus performing a *full-3D* reconstruction (Chapter 4). This requires significant modifications to the previous pipeline and entails the use of more general algorithms and data structures, first and foremost of a three-dimensional space partitioning structure to represent the input scene. The initial detection of candidate permanent components – which in the 2.5D case could be performed by considering the unoccluded vertical extent of the rectangles – is replaced with a more general algorithm inspired by structural analysis that can detect permanent components with arbitrary orientations. A clustering of the planes of the permanent components yields the dominant planes of the environment, from which a three-dimensional cell complex is built. The complex represents a partitioning of the space surrounding the input scene into convex volumetric regions. The room shapes are extracted by appropriately grouping these regions with a multi-label energy minimization algorithm. Compared to the diffusion-based clustering used in the 2.5D case, this formulation includes several regularization terms that allow to cope more effectively with the increased geometrical and topological complexity arising from the 3D nature of the problem. To compute the labels required by the multi-label optimization, this thesis proposes the use of a visibility-based clustering procedure that detects the presence of the rooms prior to the final reconstruction of their shapes.

The visibility-based room detection algorithm is further extended to make the modeling process scalable to large environments composed of multiple rooms. In fact, the complexity of a three-dimensional space partitioning structure (necessary to capture permanent structures of arbitrary orientations) represents the biggest potential bottleneck in a reconstruction pipeline. In the case of the BSP-based structure used in this thesis and in other related approaches, the complexity of the structure grows cubically with the number of (dominant) planes used to construct it. Since each room is an enclosed space and does not intersect with other rooms in the environment, this thesis proposes performing an early visibility-based room detection that allows to extract clusters of dominant planes separately for each room and to build a separate 3D structure from each cluster of dominant planes. This decreases the overall complexity of the structure used and additionally allows to cast the multi-room reconstruction as a sequence of single-room reconstruction problems, for which simpler and more effective formulations can be employed.

6.2 General Discussion

Specific remarks on each of the three main approaches introduced in this thesis are provided in the concluding sections of the relevant chapters (Sections 3.7,4.8 and 5.8). We include in this section some final considerations on the intended application scenarios of each method and a discussion on the role of the parameters used in our pipelines.

6.2.1 Scope of Application of the Proposed Pipelines

The three research contributions of this thesis (see Section 1.5) correspond to a sequence of pipelines of increasing complexity, each of which tackles a superset of the challenges addressed by the previous ones. Although in this sense the method of Chapter 4.6.2 is the most generic, there are some application scenarios for which it is advantageous to employ a less specialized solution.

In particular, the 2.5D approach of Chapter 3 is not entirely superseded by the full-3D method of Chapter 4: thanks to its tighter starting assumptions, the 2.5D pipeline is easier to implement and faster and can be used whenever efficiency is a priority and at the same time there is the certainty that the input environment has a 2.5D structure.

Similar arguments hold for the full-3D approach. While the room-based pipeline of Chapter 5 is more scalable and capable of recovering finer details, it lacks by design an optimization term that takes into account the interaction between different rooms (the room separation term E_{sep} described in Section 4.6.2). The room-based pipeline should be certainly used when the input environment is large, but the traditional full-3D approach includes a number of features (besides the room separation term, the detection of permanent components based on structural paths and the interactive refinement step) that make it the method of choice when dealing with medium-sized environment with few small-scale details. It should be noted, however, that some of the features mentioned above (with the exception of the room separation term) can be integrated into the room-based approach as well.

6.2.2 Discussion on Parameters

The methods proposed depend on several parameters. While the specific values chosen in our implementations are described in each chapter, we review here the role of the most important ones and discuss the sensitivity of the algorithms to their choice.

Planar Abstraction

One set of parameters shared by all pipelines is related to the planar abstraction performed as a pre-processing of the input point clouds (Section 2.1). These are the two thresholds θ_{ang} and θ_{off} used in the planar segmentation to steer the growing of the planar regions, as well as the adjacency threshold θ_{adj} used in the construction of the G_{adj} to determine whether two rectangles should be considered adjacent. The choice of θ_{ang} and θ_{off} is influenced by the level of noise in the input data; these values should be increased if high levels of noise values are expected, so as to allow a larger tolerance when extracting the extent of the planar regions. While the parameters of the underlying planes can be recovered in a fairly reliable manner by means of the robust fitting techniques used (in particular, the LMS-based fitting procedure), high levels of noise can result in shrunken planar regions close to the intersections with adjacent patches. We empirically discovered that setting a conservative value for θ_{adj} solves this problem for the methods of Chapters 4 and 5. However, the issue has a more direct impact on the occlusion-aware detection of permanent components used in the 2.5D pipeline (see Section 3.3): to ensure that an intersection occurs between a rectangle and the projection of its occluder, the scaling factor 1.05 used for such projection must be increased, possibly resulting in a higher number of false positives.

Extraction of Room Shapes

Regardless of the specific room reconstruction technique used, the parameters involved in this step have been set to values that ensure good reconstruction results independently of the properties of the specific input dataset used. This applies in particular to the weights λ_{cov}, λ_A, λ_G (used in Chapters 4 and 5) and λ_{sep} (used in Chapter 4), which are linked to the desired shape of the reconstruction rather than to the properties of the input raw data, but also to the parameters m (number of eigenvalues used) and t (diffusion time) used in the 2.5D approach (Chapter 3). With respect to t, it is worth noticing once more (as already done in Section 3.5.3) that large rooms like corridors can only be reconstructed as a single entity if a high enough diffusion time is allowed. Instead of performing an interactive adjustment of this parameter (a valid approach that we successfully applied in the context of object classification [Mattausch et al., 2014]), we opted for fixing the value of t and adding a post-processing step that automatically merges over-segmented rooms.

Method-specific Parameters

A number of other parameters are linked to the specific computations performed in each method and require individual comments.

In the 2.5D approach, a candidate permanent rectangle is expected to have an unoccluded vertical extent almost equal to the height of the environment h_{rooms}; a tolerance $\eta \cdot h_{\text{rooms}}$ is subtracted to account for small imprecisions in the estimation of the extent of the planar regions. We set $\eta = 0.05$, which works well for most environments and in the case of moderate noise. When dealing with very high levels of noise, it can become necessary to increase the value of η, causing objects with a significant vertical extent to be misclassified as permanent components. Note, however, that this only happens when jointly dealing with high measurement noise and ambiguous furniture configurations. One additional threshold is used in the post-processing step of Section 3.5.3, to evaluate if the boundary separating two adjacent room clusters corresponds to actual wall structures or if the two rooms should be merged into one. As discussed in the relevant section, all our tests showed that there is an unambiguous distinction between the two cases, making the choice of the threshold straightforward.

Both in Chapter 4 and in Chapter 5 the individual rooms are detected by applying the Markov cluster algorithm. We set all the parameters of this algorithm to the suggested default values, only modifying the inflation parameter and the lower bound on the edge weights (see Section 4.5). We discovered that our default values for these parameters work well and result in consistently good reconstructions. For the traditional full-3D pipeline, this can be explained with the low number of viewpoints that need to be clustered and with the fact that they correspond to optimal locations, chosen empirically at acquisition time. In the case of the room-based pipeline, in which the viewpoints are generated synthetically, we noticed that modifying the two parameters results in a different number of spurious clusters. However, as explained in Section 5.6, these are removed by the optimization-based reconstruction and therefore do not influence the final result.

The room-based pipeline of Chapter 5 introduces several additional parameters that are worth further discussion. The generation of the internal view probes (see Sections 5.3.1 and 5.3.2) is controlled by three main parameters: s_{cell} (the resolution of the octree); m (used to define $m \cdot s_{\text{cell}}$ and related to the granularity used to generate candidate view probes from the octree); N_{probes} (the number of final view probes to be generated). In general, finding *optimal* values for these parameters is a non-trivial problem. For this reason, we opted for a conservative choice that favors reconstruction quality at the cost of higher computation times (see Section 5.7). Similar arguments motivate our choices of the two other most important parameters of this pipeline, namely θ_{FF} (Section 5.3.2) and θ_{fuzzy} (Section 5.3.3): we empirically discovered that increasing the values used reduces, respectively, the number of probes classified as interior and the number of primitives used to reconstruct each room, but also increases the possibility of reconstruction errors.

6.3 Directions for Future Work

This thesis significantly advances the state-of-the-art in the field of modeling of building interiors and allows for a robust, room-based architectural modeling of large-scale environments with arbitrary wall orientations. Nevertheless, there are a number of important problems that remain open in this domain and that are worth investigating in future research.

- **Uniformly smooth reconstruction of curved surfaces**

  All methods proposed in this thesis perform a piecewise-linear reconstruction of the geometry of the rooms. Curved surfaces can be captured through their first-order approximation, but there is no guarantee that this is done in a reliable and uniform manner. It would be therefore useful to study an approach to model these surfaces more consistently, for instance by using RANSAC-based approaches [Schnabel et al., 2007] to fit models of curved primitives to the facets of the reconstructed 3D polyhedra.

- **Integration of openings detection into the room extraction**

  In this thesis, a room is detected as an enclosed space that is separated from the rest of the environment by permanent structures and such that a set of positions inside it see similar parts of the scene. While this leads to good results in normal settings, ambiguous cases (e.g. sub-environments separated by large, open passages) could be handled more consistently by detecting door openings in the scene and using these to explicitly mark a separation between two different rooms.

- **Semantic labeling of rooms based on their function**

  The subdivision into separate rooms represents fundamental semantic information about the environment. For a number of real-world application scenarios, such as automatic refurnishing of interiors, it is useful to associate each room with a semantic label that describes its functional type (e.g. office, living room, kitchen). This allows, for instance, to retrieve for each room coherent furniture from a pre-existing shape database and to generate appropriate furnishing solutions. The necessary semantic labels for the rooms could be obtained by means of supervised machine learning techniques. In this context, each room could be represented as a set of descriptors associated both to its architectural shape and to the features of the furniture contained in it. To this purpose, the modeling pipelines proposed in this thesis could be used to extract the required architectural model of the rooms, while a model of the furniture can be obtained using other indoor scene analysis approaches [Mattausch et al., 2014].

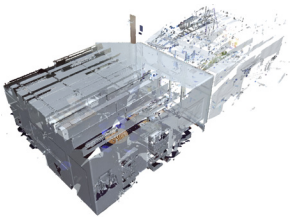
- **Multi-resolution modeling**

  State-of-the-art methods for indoor modeling (including those introduced in this thesis) focus on reconstructing a single model of the environment, optimizing it with respect to some predefined goals. Some approaches only consider larger architectural structures and filter out primitives that correspond to small details (e.g. recesses in the walls), while others include such features in the reconstructed model. However, no method has integrated the concept of level-of-detail in the reconstruction process, thus allowing to select a desired size of the architectural structures to be modeled. This capability, which has already been studied for more general urban environments [Verdie et al., 2015], would be a useful addition to the existing indoor modeling pipelines.

APPENDIX A

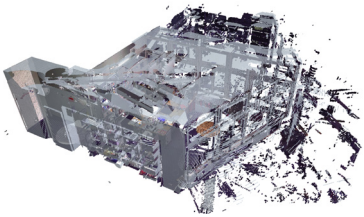
DESCRIPTION OF DATASETS

Room 1



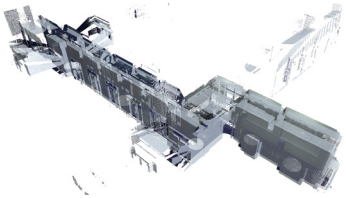
Description Single office room with convex shape and one large window surface
No. scans 2
No. rooms 1
No. points 5.5M
Acquisition modality High-end, terrestrial laser range-scanning
Format Range-grid set
Source Own

Room 2



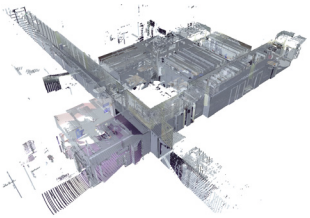
Description Single office room with concave shape and two large window surface
No. scans 3
No. rooms 1
No. points 8.3M
Acquisition modality High-end, terrestrial laser range-scanning
Format Range-grid set
Source Own

Office 1



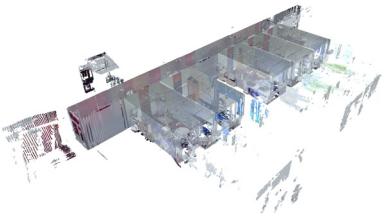
Description Long corridor inside an office building, composed of two sub-spaces
No. scans 5
No. rooms 2
No. points 13.8M
Acquisition modality High-end, terrestrial laser range-scanning
Format Range-grid set
Source Own

Office 2



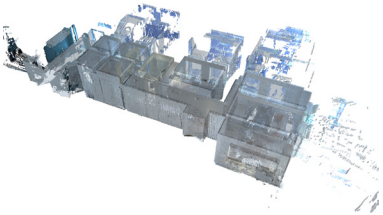
Description Office environment composed of multiple individual rooms
No. scans 10
No. rooms 6
No. points 27.7M
Acquisition modality High-end, terrestrial laser range-scanning
Format Range-grid set
Source Own

Office 3



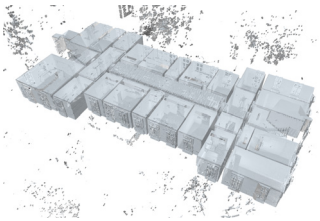
Description Office environment composed of multiple individual rooms
No. scans 14
No. rooms 6
No. points 38.8M
Acquisition modality High-end, terrestrial laser range-scanning
Format Range-grid set
Source Own

Apartment 1



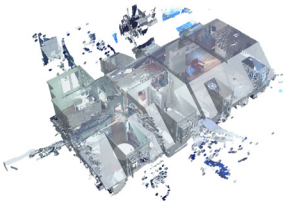
Description Residential environment composed of multiple individual rooms
No. scans 16
No. rooms 5
No. points 7.2M
Acquisition modality Proprietary acquisition system based on RGB-D cameras
Format Range-grid set
Source Floored Inc. (used in [Ikehata et al., 2015])

Building D



Description Office environment composed of multiple individual rooms
No. scans 36
No. rooms 27
No. points 63.1M
Acquisition modality High-end, terrestrial laser range-scanning
Format Range-grid set
Source DURAARK Project (used in [Ochmann et al., 2016])

Penthouse



Description Residential environment composed of multiple individual rooms

No. scans 7

No. rooms 4

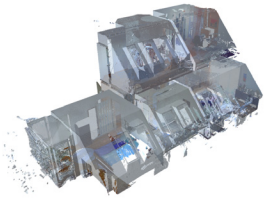
No. points 19.4M

Acquisition modality High-end, terrestrial laser range-scanning

Format Range-grid set

Source Own

Maisonnette



Description Residential environment composed of multiple individual rooms on two floor levels

No. scans 8

No. rooms 5

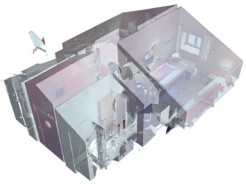
No. points 22.2M

Acquisition modality High-end, terrestrial laser range-scanning

Format Range-grid set

Source Own

Cottage



Description Residential environment composed of multiple individual rooms

No. scans 7

No. rooms 7

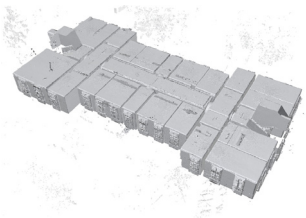
No. points 12.4M

Acquisition modality High-end, terrestrial laser range-scanning

Format Range-grid set

Source Own

Building D (unstr.)



Description Unstructured point cloud built by merging and resampling the individual scans of 'Building D'

No. scans 1

No. rooms 27

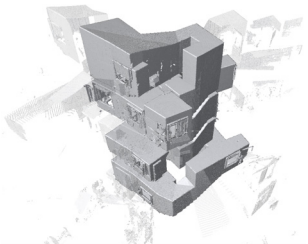
No. points 5.2M

Acquisition modality High-end, terrestrial laser range-scanning

Format Unstructured point cloud

Source DURAARK Project (used in [Ochmann et al., 2016])

Tower



Description Residential environment composed of a single staircase spanning multiple floor levels and connecting 5 separate rooms

No. scans 1

No. rooms 6

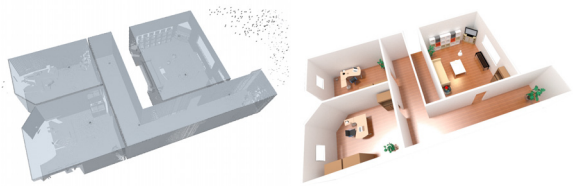
No. points 9.5M

Acquisition modality High-end, terrestrial laser range-scanning

Format Unstructured point cloud

Source Own

Synth 1



Description Office environment composed of multiple individual rooms

No. scans 7

No. rooms 4

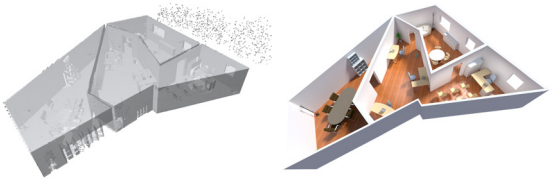
No. points 19.4M

Acquisition modality Synthetic dataset

Format Range-grid set

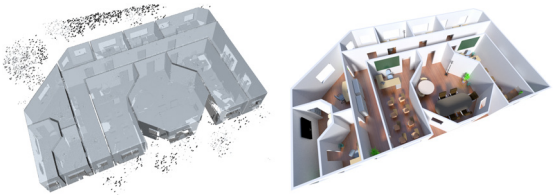
Source Own

Synth 2



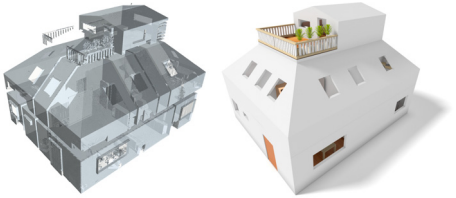
Description Office environment composed of multiple individual rooms
No. scans 7
No. rooms 3
No. points 19.4M
Acquisition modality Synthetic dataset
Format Range-grid set
Source Own

Synth 3



Description Office environment composed of multiple individual rooms
No. scans 25
No. rooms 11
No. points 69.4M
Acquisition modality Synthetic dataset
Format Range-grid set
Source Own

House



Description Residential environment composed of multiple individual rooms on three floor levels

No. scans 19

No. rooms 9

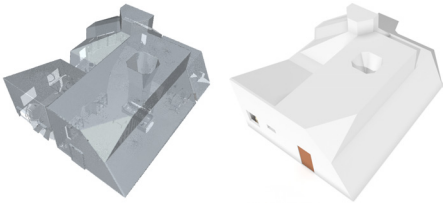
No. points 52.8M

Acquisition modality Synthetic dataset

Format Range-grid set

Source Own

Modern



Description Residential environment composed of multiple individual rooms

No. scans 8

No. rooms 3

No. points 22.2M

Acquisition modality Synthetic dataset

Format Range-grid set

Source Own

BIBLIOGRAPHY

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

[Arikan et al., 2013] Arikan, M., Schwärzler, M., Flöry, S., Wimmer, M., and Maierhofer, S. (2013). O-Snap: Optimization-based snapping for modeling architecture. *ACM Transactions on Graphics*, 32(1):6:1–15.

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

[Blender Foundation, 2016] Blender Foundation (2016). Blender 2.78.

[Boulch et al., 2014] Boulch, A., de La Gorce, M., and Marlet, R. (2014). Piecewise-planar 3D reconstruction with edge and corner regularization. *Computer Graphics Forum*, 33(5):55–64.

[Boulch et al., 2013] Boulch, A., Houllier, S., Marlet, R., and Tournaire, O. (2013). Semantizing complex 3D scenes using constrained attribute grammars. In *Computer Graphics Forum*, volume 32, pages 33–42.

[Boulch and Marlet, 2014] Boulch, A. and Marlet, R. (2014). Statistical criteria for shape fusion and selection. In *Proceedings International Conference on Pattern Recognition (ICPR)*, pages 936–941.

[Boykov and Kolmogorov, 2004] Boykov, Y. and Kolmogorov, V. (2004). An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(9):1124–1137.

[Boykov et al., 2001] Boykov, Y., Veksler, O., and Zabih, R. (2001). Fast approximate energy minimization via graph cuts. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(11):1222–1239.

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

[Cabral and Furukawa, 2014] Cabral, R. and Furukawa, Y. (2014). Piecewise planar and compact floorplan reconstruction from images. In *Proceedings IEEE Conference on Computer Vision and Pattern Recognition*, pages 628–635.

[Ceylan et al., 2012] Ceylan, D., Mitra, N. J., Li, H., Weise, T., and Pauly, M. (2012). Factored facade acquisition using symmetric line arrangements. *Computer Graphics Forum*, 31(2):671–680.

[Chauve et al., 2010] Chauve, A.-L., Labatut, P., and Pons, J.-P. (2010). Robust piecewise-planar 3D reconstruction and completion from large-scale unstructured point data. In *Proceedings IEEE Conference on Computer Vision and Pattern Recognition*, pages 1261–1268.

[Choi et al., 2015] Choi, S., Zhou, Q.-Y., and Koltun, V. (2015). Robust reconstruction of indoor scenes. In *Proceedings Computer Vision and Pattern Recognition (CVPR)*, pages 5556–5565.

[Coifman and Lafon, 2006] Coifman, R. R. and Lafon, S. (2006). Diffusion maps. *Applied and Computational Harmonic Analysis*, 21(1):5–30.

[Comaniciu and Meer, 2002] Comaniciu, D. and Meer, P. (2002). Mean shift: A robust approach toward feature space analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 24(5):603–619.

[Coughlan and Yuille, 2000] Coughlan, J. M. and Yuille, A. L. (2000). The manhattan world assumption: Regularities in scene statistics which enable bayesian inference. In *Proceedings Neural Information Processing Systems*, pages 809–815.

[Crow, 1977] Crow, F. C. (1977). Shadow algorithms for computer graphics. *Proceedings ACM SIGGRAPH*, 11(2):242–248.

[Delong et al., 2012] Delong, A., Osokin, A., Isack, H. N., and Boykov, Y. (2012). Fast approximate energy minimization with label costs. *International Journal of Computer Vision*, 96(1):1–27.

[Di Benedetto et al., 2014] Di Benedetto, M., Ganovelli, F., Balsa R., M., Jaspe V., A., Scopigno, R., and Gobbetti, E. (2014). Exploremaps: Efficient construction and ubiquitous exploration of panoramic view graphs of complex 3D environments. *Computer Graphics Forum*, 33(2):459–468.

[Edelsbrunner et al., 1986] Edelsbrunner, H., O'Rourke, J., and Seidel, R. (1986). Constructing arrangements of lines and hyperplanes with applications. *SIAM Journal on Computing*, 15(2):341–363.

[Fleishman et al., 2000] Fleishman, S., Cohen-Or, D., and Lischinski, D. (2000). Automatic camera placement for image-based modeling. *Computer Graphics Forum*, 19(2):101–110.

[Furukawa et al., 2009] Furukawa, Y., Curless, B., Seitz, S. M., and Szeliski, R. (2009). Reconstructing building interiors from images. In *Proceedings International Conference on Computer Vision*, pages 80–87.

[Gregor and Whitaker, 2001] Gregor, J. and Whitaker, R. T. (2001). Indoor scene reconstruction from sets of noisy range images. *Graphical Models*, 63(5):304–332.

[Gross and Pfister, 2007] Gross, M. H. and Pfister, H., editors (2007). *Point-Based Graphics*. Series in Computer Graphics. Morgan Kaufmann Publishers.

[Hoppe et al., 1992] Hoppe, H., DeRose, T., Duchampt, T., McDonald, J., and Stuetzle, W. (1992). Surface reconstruction from unorganized points. In *Proceedings ACM SIGGRAPH*, pages 71–78. ACM Press.

[Huber and Ronchetti, 2009] Huber, P. J. and Ronchetti, E. M. (2009). *Robust Statistics*. Wiley Series in Probability and Statistics. Wiley.

[Ikehata et al., 2015] Ikehata, S., Yan, H., and Furukawa, Y. (2015). Structured indoor modeling. In *Proceedings IEEE International Conference on Computer Vision*, pages 1323–1331.

[Isack and Boykov, 2012] Isack, H. and Boykov, Y. (2012). Energy-based geometric multi-model fitting. *International Journal of Computer Vision*, 97(2):123–147.

[Kaufman and Rousseeuw, 1987] Kaufman, L. and Rousseeuw, P. (1987). Clustering by means of medoids. In Yadolah, D., editor, *Statistical Data Analysis Based on the L1-Norm and Related Methods*. North–Holland.

[Kim et al., 2012] Kim, Y. M., Dolson, J., Sokolsky, M., Koltun, V., and Thrun, S. (2012). Interactive acquisition of residential floor plans. In *Proceedings IEEE International Conference on Robotics and Automation*, pages 3055–3062.

[Kolmogorov and Zabin, 2004] Kolmogorov, V. and Zabin, R. (2004). What energy functions can be minimized via graph cuts? *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(2):147–159.

[Lafarge and Alliez, 2013] Lafarge, F. and Alliez, P. (2013). Surface reconstruction through point set structuring. *Computer Graphics Forum*, 32(2):225–234.

[Laga et al., 2013] Laga, H., Mortara, M., and Spagnuolo, M. (2013). Geometry and context for semantic correspondences and functionality recognition in manmade 3D shapes. *ACM Transactions on Graphics*, 32(5):150:1–16.

[Lipman et al., 2010] Lipman, Y., Chen, X., Daubechies, I., and Funkhouser, T. (2010). Symmetry factored embedding and distance. *ACM Transactions on Graphics*, 29(4):103:1–12.

[Mattausch et al., 2014] Mattausch, O., Panozzo, D., Mura, C., Sorkine-Hornung, O., and Pajarola, R. (2014). Object detection and classification from large-scale cluttered indoor scans. *Computer Graphics Forum*, 33(2):11–21.

[Monszpart et al., 2015] Monszpart, A., Mellado, N., Brostow, G. J., and Mitra, N. J. (2015). RAPter: Rebuilding man-made scenes with regular arrangements of planes. *ACM Transactions on Graphics*, 34(4):103:1–12.

[Mura et al., 2013] Mura, C., Mattausch, O., Jaspe Villanueva, A., Gobbetti, E., and Pajarola, R. (2013). Robust reconstruction of interior building structures with multiple rooms under clutter and occlusions. In *Proceedings IEEE Conference on Computer-Aided Design and Computer Graphics*, pages 52–59.

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

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

[Musialski et al., 2013] Musialski, P., Wonka, P., Aliaga, D., Wimmer, M., Van Gool, L., and Purgathofer, W. (2013). A survey of urban reconstruction. *Computer Graphics Forum*, 32(6):146–177.

[Nan et al., 2015] Nan, L., Jiang, C., Ghanem, B., and Wonka, P. (2015). Template assembly for detailed urban reconstruction. *Computer Graphics Forum*, 34(2):217–228.

[Nan et al., 2010] Nan, L., Sharf, A., Zhang, H., Cohen-Or, D., and Chen, B. (2010). SmartBoxes for interactive urban reconstruction. *ACM Transactions on Graphics*, 29(4):93:1–10.

[Newcombe et al., 2011] Newcombe, R. A., Izadi, S., Hilliges, O., Molyneaux, D., Kim, D., Davison, A. J., Kohi, P., Shotton, J., Hodges, S., and Fitzgibbon, A. (2011). Kinectfusion: Real-time dense surface mapping and tracking. In *Proceedings International Symposium on Mixed and Augmented Reality*, pages 127–136.

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

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

[Oesau et al., 2016] Oesau, S., Lafarge, F., and Alliez, P. (2016). Planar shape detection and regularization in tandem. *Computer Graphics Forum*, 35(1):203–215.

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

[Pintore and Gobbetti, 2014] Pintore, G. and Gobbetti, E. (2014). Effective mobile mapping of multi-room indoor structures. *The Visual Computer*, 30(6-8):707–716.

[Poppinga et al., 2008] Poppinga, J., Vaskevicius, N., Birk, A., and Pathak, K. (2008). Fast plane detection and polygonalization in noisy 3D range images. In *Proceedings International Conference on Intelligent Robots and Systems*, pages 3378–3383.

[Rabbani et al., 2006] Rabbani, T., van den Heuvel, F., and Vosselman, G. (2006). Segmentation of point clouds using smoothness constraint. *International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences*, 36(5):248–253.

[Rousseeuw, 1984] Rousseeuw, P. J. (1984). Least median of squares regression. *Journal of the American Statistical Association*, 79(388):871–880.

[Rousseeuw and Leroy, 1987] Rousseeuw, P. J. and Leroy, A. M. (1987). *Robust Regression and Outlier Detection*. John Wiley & Sons.

[Salas-Moreno et al., 2013] Salas-Moreno, R. F., Newcombe, R. A., Strasdat, H., Kelly, P. H., and Davison, A. J. (2013). Slam++: Simultaneous localisation and mapping at the level of objects. In *Proceedings Computer Vision and Pattern Recognition (CVPR)*, pages 1352–1359.

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

[Sankar and Seitz, 2012] Sankar, A. and Seitz, S. (2012). Capturing indoor scenes with smartphones. In *Proceedings ACM Symposium on User Interface Software and Technology*, pages 403–412.

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

[Scott et al., 2003] Scott, W. R., Roth, G., and Rivest, J.-F. (2003). View planning for automated three-dimensional object reconstruction and inspection. *ACM Computing Surveys*, 35(1):64–96.

[Steder et al., 2011] Steder, B., Rusu, R. B., Konolige, K., and Burgard, W. (2011). Point feature extraction on 3D range scans taking into account object boundaries. In *International Conference on Robotics and Automation (ICRA)*, pages 2601–2608.

[Surmann et al., 2003] Surmann, H., Nüchter, A., and Hertzberg, J. (2003). An autonomous mobile robot with a 3D laser range finder for 3D exploration and digitalization of indoor environments. *Robotics and Autonomous Systems*, 45(3–4):181–198.

[Tang et al., 2010] Tang, P., Huber, D., Akinci, B., Lipman, R., and Lytle, A. (2010). Automatic reconstruction of as-built building information models from

laser-scanned point clouds: A review of related techniques. *Elsevier Automation in Construction*, 19(7):829–843.

[Turner and Zakhor, 2012] Turner, E. and Zakhor, A. (2012). Watertight as-built architectural floor plans generated from laser range data. In *Proceedings Conference on 3D Imaging, Modeling, Processing, Visualization and Transmission (3DIMPVT)*, pages 316–323.

[Turner and Zakhor, 2013] Turner, E. and Zakhor, A. (2013). Watertight planar surface meshing of indoor point-clouds with voxel carving. In *Proceedings 3DV*, pages 41–48.

[Turner and Zakhor, 2014] Turner, E. and Zakhor, A. (2014). Floor plan generation and room labeling of indoor environments from laser range data. In *Proceedings International Conference on Computer Graphics Theory and Applications*, pages 29–44.

[Udeshi and Hansen, 1999] Udeshi, T. and Hansen, C. D. (1999). Towards interactive photorealistic rendering of indoor scenes: A hybrid approach. In *Rendering Techniques*, pages 63–76. Springer.

[Van Dongen, 2008] Van Dongen, S. (2008). Graph clustering via a discrete uncoupling process. *SIAM Journal on Matrix Analysis and Applications*, 30(1):121–141.

[Vanegas et al., 2012] Vanegas, C. A., Aliaga, D. G., and Benes, B. (2012). Automatic extraction of Manhattan-world building masses from 3D laser range scans. *IEEE Transactions on Visualization and Computer Graphics*, 18(10):1627–1637.

[Verdie et al., 2015] Verdie, Y., Lafarge, F., and Alliez, P. (2015). LOD generation of urban scenes. *ACM Transactions on Graphics*, 34(3):30:1–14.

[Volk et al., 2014] Volk, R., Stengel, J., and Schultmann, F. (2014). Building information modeling (BIM) for existing buildings — Literature review and future needs. *Automation in Construction*, 38:109–127.

[Wang, 2011] Wang, C. C. (2011). Approximate boolean operations on large polyhedral solids with partial mesh reconstruction. *IEEE Transactions on Visualization and Computer Graphics*, 17(6):836–849.

[Wenhardt et al., 2006] Wenhardt, S., Deutsch, B., Hornegger, J., Niemann, H., and Denzler, J. (2006). An information theoretic approach for next best view planning in 3D reconstruction. In *International Conference on Pattern Recognition (ICPR)*, pages 103–106.

[Zheng et al., 2010] Zheng, Q., Sharf, A., Wan, G., Li, Y., Mitra, N. J., Cohen-Or, D., and Chen, B. (2010). Non-local scan consolidation for 3D urban scenes. *ACM Transactions on Graphics*, 29(4):94:1–9.

[Zheng et al., 2014] Zheng, Y., Cohen-Or, D., Averkiou, M., and Mitra, N. (2014). Recurring part arrangements in shape collections. *Computer Graphics Forum*, 33(2):115–124.

CURRICULUM VITAE

Personal Information

Name	Claudio Mura
Date of birth	May 13th, 1987
Place of birth	Cagliari, Sardinia, Italy

Education

2012 - 2017	Doctoral studies in Computer Science, University of Zurich
2008 - 2010	Master of Science in Computer Science, University of Cagliari
2005 - 2008	Bachelor of Science in Computer Science, University of Cagliari

Professional Experience

since 11.2011	Research and Teaching Assistant, University of Zurich
12.2010 - 10.2011	Software Engineer and 3D Scanning Specialist, Centro di Ricerca, Sviluppo e Studi Superiori in Sardegna (CRS4)
10.2010 - 12.2010	Teaching Assistant, University of Cagliari
04.2010 - 10.2011	Software Engineer and 3D Scanning Specialist, Sardegna Ricerche

Scolarships and Awards

10.2016	Best Student Paper Award, Pacific Graphics
12.2011 - 11.2014	Marie Curie Fellowship, Initial Trainig Network 'DIVA'

Publications

Journal Articles

C. Mura, O. Mattausch and R. Pajarola, *Piecewise-planar Reconstruction of Multi-room Interiors with Arbitrary Wall Arrangements*, Computer Graphics Forum (Proceedings Pacific Graphics), 35(7):179-188, 2016.

C. Mura, O. Mattausch, A. Jaspe, E. Gobbetti and R. Pajarola, *Automatic Room Detection and Reconstruction in Cluttered Indoor Environments with Complex Room Layouts*, Computers & Graphics, 44:20-32, 2014.

O. Mattausch, D. Panozzo, C. Mura, O. Sorkine-Hornung and R. Pajarola, *Object Detection and Classification from Large-Scale Cluttered Indoor Scans*, Computer Graphics Forum (Proceedings Eurographics), 33(2):11-21, 2014.

Conference Publications

M. Agus, E. Gobbetti, A. Jaspe, C. Mura and R.Pajarola, *SOAR: Stochastic Optimization for Affine Global Point Set Registration*, International Workshop on Vision, Modeling and Visualization (VMV), 103-110, 2014.

C. Mura, A. Jaspe, O. Mattausch, E. Gobbetti and R. Pajarola, *Reconstructing Complex Indoor Environments with Arbitrary Wall Orientations*, Proceedings Eurographics (Posters), 2014.

C. Mura, O. Mattausch, A. Jaspe, E. Gobbetti and R. Pajarola, *Robust Reconstruction of Interior Building Structures with Multiple Rooms under Clutter and Occlusions*, IEEE Conference on Computer-Aided Design and Computer Graphics, 52-59, 2013.

S. Marras, C. Mura, E. Gobbetti, R. Scateni and R. Scopigno, *Two Examples of GPGPU Acceleration of Memory-intensive Algorithms*, Proceedings Eurographics Italian Chapter, 2010.

F. Guggeri, S. Marras, C. Mura and R. Scateni, *Topological Operations on Triangle Meshes Using the OpenMesh Library*, Proceedings Eurographics Italian Chapter, 2008.