POLITECNICO DI MILANO
DIPARTIMENTO DI ELETTRONICA, INFORMAZIONE E BIOINGEGNERIA
DOCTORAL PROGRAM IN INFORMATION TECHNOLOGY

SEMANTIC MODELING OF THE GLOBAL STRUCTURE OF BUILDINGS

Doctoral Dissertation of:
Matteo Luperto

Supervisor:
Prof. Francesco Amigoni

Tutor:
Prof. Luciano Baresi

The Chair of the Doctoral Program:
Prof. Andrea Bonarini

2016 – XXIX

Ringraziamenti

Questa tesi conclude idealmente un percorso lungo, iniziato oramai qualche anno fa e portato avanti tra tanti momenti, circostanze, luoghi e situazioni diverse. E soprattutto, in mezzo ad un sacco di gente.

E siccome è quesa la cosa che importa, quella che conta davvero, è giusto e doveroso iniziare questo manoscritto ringraziando tutti quelli che, direttamente o indirettamente, ne hanno costruita una parte.

Voglio innanzitutto ringraziare il Prof. Francesco Amigoni, sotto la cui guida ho compiuto questo percorso. Sono stati tre anni lunghi, cinque contando anche la tesi magistrale, RoboCup, il Messico e tutto il resto. Anni dove ho imparato molto sia a livello scientifico che a livello umano, che mi hanno lasciato tanto e che spero continueranno ancora.

Voglio poi ringraziare tutti quelli che mi hanno accompagnato (quasi) ogni giorno in questa vita politecnica: per primo Jacopo, che ha avuto la sfortuna di capitarmi in ufficio assieme e che si è rivelato un prezioso amico; poi Alberto, Francesco e Marco, che mi hanno sopportato nell'oramai riconvertito openspace durante il primo anno e che sono rimasti vicini anche quando non erano più nella scrivania accanto. Non dimentico però Alessandro, Davide e Carlo e tutti gli altri "giovani" con cui ho passato questi ultimi due bei anni, e con cui ho avuto il piacere di scambiare idee, opinioni e stupidità. Rimane poi Nicola, il cui percorso si è intrecciato ripetutamente con il mio e con cui spero di collaborare per molto tempo ancora.

Ringrazio poi Diego e tutto HocLab, con cui ho iniziato questo dottorato e che danno un tocco di colore al dipartimento, e tutti i colleghi con cui

ho avuto il piacere di parlare, discutere, pranzare e collaborare. Desidero poi ringraziare Cristiana Bolchini e Vittorio Zaccaria, perché ritengo che fare attività di laboratorio con loro sia stata una cosa sia interessante che formativa, che ha arricchito il mio percorso.

Concludo questa sezione "accademica" ringraziando tutti gli studenti con cui ho collaborato in questi anni: Leone, Michele, Nino, Mattia, Matteo, Alessandro, Valerio, Matteo, David e Nicola. È stato un piacere seguirvi, spero vi sia stato utile.

Ringrazio poi i miei ex compagni di corso, Mladen, Michele, Alessandro, Alain, Riccardo ed Eros, con cui supportiamo la ristorazione etnica lecchese oramai dal 2007, che è un sacco di tempo fa.

Volevo poi ringraziare i miei amici, che, nonostante la vita che passa, sono rimasti vicini dopo tanti anni che ci conosciamo, e con i quali ho passato tante tante cose. Non vi nomino, che siete tanti e finirei per lasciare fuori qualcuno o riempire la pagina di nomi: sapete bene tanto chi siete. Tra di voi c'è Elton, coinquilino e compagno di disavventure musicali sui palchi, più o meno popolati, di mezza Italia. A lui e a tutti i musicisti con cui ho avuto il piacere di condividere un palco, un grazie: parte di questo lavoro è stata scritta sui sedili posteriori di qualche furgone diretto ad un improbabile check in ritardo in qualche paese in Toscana. È stato bello, e non termina qui.

Concludo ringraziando la mia famiglia: mia madre, mio padre, mia sorella, i miei nonni anche se non ci sono più, le mie cugine e il piccolo Jacopo. E tutti gli animali, ovviamente, che sono un sacco, sono per me molto importanti, e anche se non sanno leggere è giusto che abbiano un posticino qui sopra.

Infine, una grazie infinito a Giulia che, per motivi a me ignoti, continua a rimanermi accanto da più di dieci anni, e che mi supporta, sopporta, incoraggia e aiuta. Senza di lei non ce l'avrei fatta a portare a termine questo percorso. Letteralmente, e qui è tempo di una confessione: se vi piaceranno tutte le immagini colorate che vedrete in questo elaborato, sappiate che buona parte sono state fatte da lei, con pazienza infinita. È stata e sarà la mia più valida alleata.

Abstract

Autonomous mobile robots can perform many different tasks to help humans during their activities or to replace them in hazardous environments and in simple routine operations.

When we consider indoor tasks, robots have to interact with environments that are specifically designed for human activities and for interaction between humans, *buildings*. Buildings are strongly structured environments that are organized in regular patterns. For instance, *rooms* typically have a geometrical structure that is characterized by features, such as walls perpendicular to the floor and to the ceiling, and by a layout that can be, in most cases, approximated by a box-like model. In order to increase their ability to autonomously operate in indoor environments, robots must have a good understanding of buildings, similarly to that human beings exploit during their everyday activities. If we consider how people and robots interact with indoor environments, it can be said that people naturally understand and "*read*" buildings as human-made environments (and act in them accordingly), and that this is hardly the case for autonomous mobile robots.

One of the most important tools that researchers have developed to address a robot's needs for interacting with an indoor environment are *semantic maps*. Semantic maps are abstract representations which aim to represent the *meaning* of parts of the perceived environment in order to provide robots a human-like understanding of their surroundings. Semantic maps can be used for describing heterogeneous concepts that can be useful for robots, such as objects and rooms. In this dissertation, we focus on a particular type of semantic maps, which identifies rooms, represents

how rooms are connected, and assigns to each room a *semantic label* indicating its function, such as 'corridor', 'classroom', 'office', or 'bathroom'. Semantic maps are usually built on metric maps, that represent the space occupation and are particularly useful for tasks such as path planning and localization.

Typically, the interaction between a robot and its environment is heavily based on data acquired with perception. Mapping methods usually provide reliable knowledge only on parts of the environment that have been already visited. This approach often implies that what has not been seen by the robot does not exist, adopting, in a sense, a closed world assumption on the environment. This statement is true considering both semantic and metric maps. This form of interaction with the environment is radically different from that of humans, who can easily navigate and comprehend the structure of buildings even without having seen them before.

The contribution of this thesis moves from the consideration that the global structure of buildings, which is often neglected when building semantic maps, could be exploited to increase the autonomous abilities of robots when operating in indoor environments. Our proposed framework aims at identifying and at overcoming the limitations in standard semantic mapping methods by starting from two insights on indoor environments. In first place, we consider an entire floor of a building as a single object, by identifying relations between different (and potentially unconnected) parts of the building. This can be done both considering the metric map of the environment, for example by identifying that rooms in different parts of the building share one or more walls, and by considering the topology of the environment, namely how rooms are connected with each other, and for example observing that parts of the building with a similar function have a similar structure.

Moreover, we consider each building in relation with other buildings with the same function. The function of a building is represented by the main activity that each building is designed for and is captured by the concept of *building type*. Examples of building types are schools, offices, hospitals, university, shopping malls, and others. The function of a building imposes its structure, its floor plan, and the structure of its rooms. Each building, having a precise function, shares some structural features with all other buildings with the same purpose.

Exploiting these two observations, we provide an analytical model of the *structure* of a building that considers altogether all the relations between its single parts and that considers the features shared by the set of buildings belonging to the same type. We start from reconstructing the layout of an

indoor environment from its metric map. The layout can then be used for obtaining a graph representation the building. In our approach, data from the metric maps are used in combination with data representing floor plans of buildings belonging to the same type. Using graph kernels and Monte Carlo Markov Chains, we provide a method able to generate new instances of building structures from a set of examples. We outline some possible applications of our approach, involving reasoning on unknown parts of buildings and labelling entire floors of buildings accordingly to their function.

Sommario

In un prossimo futuro, i robot mobili autonomi potranno aiutarci nelle attività quotidiane, sollevarci da compiti svolti in ambienti pericolosi, e svolgere per noi semplici attività ripetitive.

In particolare, già ora, i robot che operano in ambienti indoor devono sapersi muovere e interagire con delle strutture appositamente progettate per l'attività umana e per un uso specifico, gli *edifici*. Gli edifici sono ambienti fortemente strutturati e sono organizzati in una serie di pattern regolari. Le *stanze*, ad esempio, hanno tipicamente una struttura geometrica caratterizzante, con pareti perpendicolari al pavimento e al soffitto ed un aspetto che, nella maggior parte dei casi, può essere approssimato da un box-model, un parallelepipedo. Un robot autonomo deve essere in grado di *capire* in maniera efficace l'ambiente in cui si trova, cercando di avere un comportamento quanto più simile possibile a quello della propria controparte umana. Se consideriamo come le persone vivono all'interno degli edifici, si può vedere come esse naturalmente riescano a comprendere e *leggere* gli edifici come costruiti attorno a loro; una considerazione simile non è invece fattibile quando si parla di robot.

Uno dei metodi più efficaci che un robot ha a disposizione per comprendere l'ambiente che lo circonda è la *mappa semantica*. Una mappa semantica è una rappresentazione astratta dell'ambiente in cui viene attribuito un *significato* alle parti dell'ambiente percepite dal robot. Il fine di una mappa semantica è di fornire al robot la possibilità di comprendere l'ambiente che lo circonda in maniera simile ad un essere umano. Le informazioni contenute in una mappa semantica possono essere di diverso tipo, e

variano in base all'ambiente in cui il robot deve operare (interno o esterno) e ai compiti che il robot deve svolgere. Una mappa semantica è dunque una descrizione eterogenea dell'ambiente funzionale all'attività del robot e contenente indicazioni quali la presenza di oggetti e la tipologia delle singole stanze. In questo lavoro di tesi, ci concentreremo su una particolare tipologia di mappa semantica finalizzata ad identificare le stanze presenti nell'edificio, a capire come sono tra loro connesse e ad assegnare loro una *etichetta semantica* indicante la loro funzione, come 'corridoio', 'classe', 'ufficio' o 'bagno'.' Le mappe semantiche sono tipicamente ricavate a partire da mappe metriche, le quali rappresentano lo spazio come occupato o libero, e che sono particolarmente utili per attività come il path planning e la localizzazione.

L'interazione tra il robot e l'ambiente in cui opera è solitamente caratterizzata dai dati acquisiti mediante i sensori. Di conseguenza, i metodi di mapping forniscono al robot dei dati e delle informazioni solo su quelle porzioni dell'ambiente che sono già state visitate dal robot stesso. Questo approccio implica il fatto che tutto ciò che non è stato già visto dal robot, di solito non venga considerato affatto. Per un robot, quindi, ciò che non ha visto è come se non esistesse, adottando, in un certo senso, una closed-world assumption. Questa considerazione, ugualmente valida per mappe metriche e mappe semantiche, fa sì che la percezione dell'ambiente da parte di un robot sia radicalmente diversa da quella di un essere umano, che riesce facilmente a orientarsi e comprendere la struttura di un edificio anche senza averlo mai visto prima.

Il contributo di questo lavoro di tesi parte dalla considerazione che la struttura di un edificio, che spesso non viene considerata come una fonte di informazioni per il mapping semantico, può essere utilizzata per aumentare l'autonomia e le capacità percettive dei robot quando questi operano in ambienti indoor. In particolare, vogliamo proporre di identificare e superare alcune limitazioni nel corrente stato dell'arte del mapping semantico sfruttando due considerazioni sulla natura stessa degli edifici.

In primo luogo, consideriamo un intero piano di un edificio come un oggetto unico identificando delle relazioni tra le varie parti che lo compongono, siano anche non direttamente connesse. Questo può essere fatto attraverso la mappa metrica, ad esempio identificando come due stanze poste in parti differenti dello stesso piano condividano una parete, o considerando la topologia dell'ambiente, ovvero come le stanze sono connesse tra loro, ad esempio osservando che parti dell'edificio che sono deputate alla stessa funzione hanno una simile struttura.

In secondo luogo, consideriamo ogni edificio in relazione con l'insieme

degli edifici aventi la stessa funzione. La funzione di un edificio è l'attività principale che deve essere svolta al suo interno, per cui è stato concepito e progettato, e per cui viene usato. Per funzione di un edificio facciamo riferimento al concetto di *tipo edilizio*, discusso ed analizzato in architettura. Esempi di tipi edilizi sono l'edilizia scolastica, gli ospedali, le università, i centri commerciali, o le abitazioni. La funzione di un edificio impone quella che è la sua struttura, la sua planimetria, la forma e la disposizione delle stanze che lo compongono. Ogni edificio, avendo una precisa funzione, condivide delle proprietà strutturali con tutti quegli altri edifici appartenenti allo stesso tipo edilizio.

Sfruttando queste due considerazioni, proponiamo un modello analitico della *struttura* di un edificio che consideri allo stesso tempo le relazioni tra le sue singole componenti e che consideri le proprietà che condivide con l'insieme degli edifici appartenenti allo stesso tipo edilizio. A tale fine, il primo passo da compiere è la ricostruzione della planimetria dell'edificio a partire dalla sua mappa metrica. La planimetria può essere usata per ricavare una rappresentazione a grafo dell'edificio. Questa rappresentazione a grafo viene così sfruttata in combinazione con la mappa metrica e con i dati ottenuti da altri ambienti appartenenti alla stessa tipologia. Mediante l'uso di kernel su grafi e di Monte Carlo Markov Chains, proponiamo un metodo capace di generare nuove istanze di possibili strutture di edifici a partire da un insieme di esempi. Concludiamo così con un insieme di possibili esempi applicativi del nostro metodo, finalizzati soprattutto ad ottenere informazioni su porzioni dell'ambiente che non sono state ancora esplorate dal robot.

Contents

CHAPTER 1

Introduction

Understanding the environments in which they operate is an important capability for autonomous mobile robots. These robots can perform many different tasks to help humans during their activities or to replace them in hazardous environments and in routine operations. Examples of these applications includes house cleaning, patrolling buildings in order to detect anomalies, surveillance, finding victims in search and rescue missions, helping patients in hospitals, guiding people in museums or in other large social spaces like malls, and many others. In several of these tasks, robots are required to operate without any human supervision, especially in environments that human beings cannot access, as in rescue settings.

When we consider indoor tasks, robots have to interact with environments that are specifically designed for human activities and for interaction between humans, *buildings*. Buildings are strongly structured environments that are organized in regular patterns. For instance, *rooms* typically have a geometrical structure that is characterized by features, such as walls perpendicular to the floor and to the ceiling, and by a layout that can be, in most cases, approximated by a box-like model. In order to increase their ability to autonomously operate in indoor environments, robots must have a good understanding of buildings, similarly to that human beings exploit

during their everyday activities. This is even more important when the robot have not any previous knowledge on the environment in which it operates, but incrementally acquires data from the environment while performing its task. In such a scenario, the robot has to cope with several problems, mostly due to the inherent uncertainty of perceiving and moving through the world. Missing and noisy data, faulty sensors, and perception errors can undermine its functionality. If we consider how people and robots interact and move within indoor environments, it can be said that people naturally understand and "*read*" buildings as human-made environments (and act in them accordingly), and that this is hardly the case for autonomous mobile robots.

The standard way the robot use for representing the environments are *metric maps*. Metric maps, like grid maps [97], represent the space occupation and are particularly useful for tasks such as path planning. However, metric maps do not explicitly contain any knowledge about the building structure, but are limited to a representation of the space occupation. For example, there usually is no distinction between a cell of the grid occupied by a wall and a cell occupied by a chair that can be easily moved. Metric maps can be incrementally built by robots, using SLAM techniques [97].

Semantic maps have been proposed as an abstraction of metric maps which aim to represent the *meaning* of parts of the perceived environment. Semantic maps can be used for describing heterogeneous concepts that can be useful for robots, such as the nature of objects within the environment and the kind of activity performed in a room [54]. In this thesis, we focus on a particular type of semantic map. Specifically, we consider a semantic map as a representation of the environment that identifies rooms by partitioning the metric map, represents how rooms are connected, and assigns to each room a *semantic label* indicating its function, such as 'corridor', 'classroom', 'office', or 'bathroom'. This form of knowledge can be used for improving numerous task, from human-robot interaction [22, 71], autonomous cleaning [14], search [8], and (multi)robot exploration [79, 92].

Semantic mapping is usually performed starting from a metric map. As a consequence, a semantic map represents and provides to the robot only information about the subset of the environment that has been already visited by the robot. The main contribution of this work starts from the consideration that the structure of buildings, which is often neglected when building semantic maps, could be exploited to increase the autonomous abilities of robots when operating in indoor environments. Following this direction, we analyse, throughout the elaborate, the concept of structure of a building under different perspectives, while explorative examples of applications of such concept to increase the robot's understandings of its environment are

eventually shown in Chapter 8.

1.1 Understanding indoor environments

Typically, the interaction between a robot and its environment is heavily based on data acquired with perception. Robots incrementally build maps by aggregating perceptual data and rely on such maps for the completion of their tasks. This approach often implies that what has not been seen by the robot does not exist, adopting, in a sense, a closed world assumption on the environment. This form of interaction with the environment is radically different from that of humans. As an example, consider how people interact with a work place such as an office building. Usually, people do not need to enter each room of the building to understand which kind of activity is performed in it; instead, people know and have entered only a subset of the rooms of a building. In a different setting, such as a multi-level residential building, people usually can infer that all apartments are similar. People usually are proactive towards their environments, trying to comprehend their characteristics, understand their features, and *read* their *image* [61].

In this work we provide a framework that allows autonomous mobile robots to analyse and reason on the global structure of buildings and their common features, in order to make a step towards filling the gap between how humans and robots interact with the environment they inhabit.

The basic representation we consider is the *semantic map* of a building. A semantic map identifies rooms from a partition of the metric map, represents how rooms are connected, and assigns to each room a *semantic label* indicating its function, such as 'corridor', 'classroom', 'office' or 'bathroom'. Our contributions aims at improving standard semantic mapping techniques and at overcoming some of their actual limitations. More precisely, we identify three main limits that affect most of the mainstream semantic mapping approaches:

- Semantic mapping methods are usually trained and tested on a limited set of environments. This can potentially undermine the generalization of such approaches when applied to different contexts.

- Usually semantic mapping methods provide reliable knowledge only on parts of the environment that have been already visited. This can undermine the potential exploitation of semantic knowledge. For example, semantic information on the environment has been shown to

speed up multirobot exploration [79], but semantic information on the environment is obtained only after the robot has explored it.

- Semantic labels are assigned to parts of the environment often considering only information acquired directly there or, at most, on a local neighbour of those parts of environment. No information about the global structure of the environment is usually considered.

Our proposed framework tries to overcome the above limitations by starting from two insights on indoor environments that are discussed extensively in the following chapters.

The first insight is that we consider an entire building as a single object, by identifying relations between different (and potentially unconnected) parts of the building. This can be done both by considering the metric map of the environment, for example by identifying that rooms in different parts of the building share one or more walls, and by considering the topology of the environment, namely how rooms are connected with each other, and for example observing that parts of the building with a similar function have a similar structure. As an example, if we consider two different corridors of the same office building, it is probable that they will have a similar layout and that they will be connected with rooms with similar functions (e.g., offices) and similar shapes.

The second insight regards the fact that we consider each building in relation with other buildings with the same function. The function of a building is represented by the main activity that each building is designed for and is captured by the concept of *building type* [84]. Examples of building types are schools, offices, hospitals, university, shopping malls, and others. We analyse data sets of buildings belonging to the same building type, such as schools or offices, in order to identify the features that are shared between all the buildings of the same type.

Under these two assumptions, we model the *structure* of a building in order to consider altogether the relations between the single parts of the building. Using an analogy, we consider rooms within a building as musical notes in a score. While each single note has it own meaning by itself (its pitch), it is only by considering their progression and the relation between different chords that one can identify the melody and the tonality of a music score. Similarly, for each room we can consider its function, represented by its semantic label, and its geometry. But only considering how it is connected to all the other rooms of the same building we can fully define its role and function.

We provide an analytical model of the structure of a building by identi-

fying some of its *features*, which are measurable quantities. As an example we can identify that some rooms are aligned on a same wall, or that two different parts of a building have the same structure. These features are intended to describe *global* characteristics of the environment, that cannot be determined by applying standard *local* semantic mapping approaches.

The decision of considering the global structure of a building as the principal target of our analysis leads us to focus on the 2D structure of the environment. Our general pattern is to derive semantic maps from the segmentation of the environments into rooms using directly input from metric maps, for example built from data collected by laser range scanners. For this reason, we do not consider other possible sensors sources that are usually exploited for semantic categorization of buildings, as the room *appearance* and the presence of significant objects [77, 78]. This choice, although radical, allow us to focus on the main goals of this thesis.

Each element of our framework is presented separately in a chapter.At first we extract automatically the structure and the connectivity of rooms of a building from metric maps obtained by a robot. We evaluate this extracted structure against those of similar buildings obtained from data sets. In this way, we can find relations between parts of the target building and those of the buildings in the data sets. This knowledge is used for performing classification tasks on the target building, for semantic mapping, and for enabling other reasoning tasks such as categorizing an entire building according to its function (for example, determining if a building in a university campus is a dorm, a research facility or an administrative office). Finally, we develop an analytical model of the structure of the buildings, which is used for predicting the structure of unvisited parts of a building.

1.2 Original contributions

The original contributions of this dissertation are listed below.

- We develop a method that extracts the structure of a building from a metric map acquired by a robot. This task is performed by extracting information about the straight walls that compose the environment and using this knowledge for segmenting the metric map into a set of separate rooms. Our setting performs simultaneously *layout reconstruction* and *room segmentation*.

- We introduce the concept of *building type* from the field of architecture and we show how it can be fruitfully applied to semantic mapping in robotics in order to increase the generalizability of semantic map-

ping methods to different heterogeneous sets of environments. We introduce the concept of structure of buildings and we show how the structure of a building is a direct consequence of its functionality and of its building type. Part of this contribution has been presented in [57] and [59].

- We apply a Statistical Relational Learning tool, kLog [34], for performing semantic mapping by considering all the rooms that form a semantic map altogether, thus promoting a global approach to semantic mapping. We use this framework for performing novel tasks, namely *building classification*, in which we apply a semantic label to an entire environment representing its function, its building type. Part of this contribution is presented in [60].

- We develop a model for sampling new instances of buildings given a set of examples. This model is based on a graph representation of the buildings that is processed using graph kernels [40]. We show how sampled graphs have a structure similar to that of real world environments. Our model is generative and can be adapted for predicting the possible structure of a partially explored building. Parts of this contribution have been presented in [58].

- We show how reasoning on the structure of buildings can be used for assess the validity of simulated environments used in robotics. We present a method for deciding if a simulated environment is similar (or not) to real world environments. Part of this contribution is presented in [4].

- We propose two applications of our approach for inferring new knowledge on partially explored environments. This is done by predicting the layout of a partially explored room and by providing a prediction of the structure of the corridors of the unseen portion of a partially explored environments.

All the code and data sets developed for this thesis are available upon request.

1.3 Document structure

This document is structured as follows.

Chapter 2 introduces and explains relevant approaches for reasoning on indoor environments that have been presented in the literature on semantic

mapping, with a particular attention to those methods that perform *place classification*. Then, we focus on techniques that perform *room segmentation*, to underline the fact that the process of semantic mapping is usually intertwined with the task of partitioning a map into a set of rooms.

Chapter 3 shows how the structure of a building can be extracted from its metric map by performing room segmentation and layout reconstruction. Our method can be applied to input sources that include partially explored maps, blue prints, and evacuation maps.

In Chapter 4 we introduce the concept of building type and we discuss how the labeling process usually takes place in the state-of-the-art of semantic mapping. Finally, we discuss the concept of *model floor plan* as possible source of knowledge for autonomous mobile robots.

In Chapter 5 we describe the graph representation of environments which is used in the following chapters. We leverage on the method presented in Chapter 3 to extract a graph representation of a building, which is given to a classifier to obtain a semantic map.

In Chapter 6 we use the graph representation of buildings as input for Statistical Relational Learning techniques. We apply this method for performing semantic mapping and for categorizing buildings according to their type.

Chapter 7 proposes a method for developing an analytical model of a building type. This model uses the graph representation of Chapter 5 and generate (samples) new instances of graphs of buildings belonging to a building type.

Chapter 8 proposes three applications of the methods introduced in the previous chapters, two of which involves reasoning on partially explored environments. The proposed applications are, namely, validation of simulation worlds, reasoning on the layout of partially explored rooms, and prediction of the structure of partially explored buildings.

Chapter 9 concludes this dissertation and suggests further directions of research.

CHAPTER 2

State of the art

Semantic mapping for autonomous mobile robots is the process of building a representation of the environment which associates spatial concepts with spatial entities [76]. The goal of semantic mapping is to endow robots with the capabilities to understand their surroundings in human-centric terms, by identifying features and symbols that contain meaningful concepts for humans. This broad definition embraces a great variety of different methods that a robot could adopt, from using vision to detect the objects placed on a table to identifying roads, cars, and pedestrian for an autonomous vehicle driving in a city.

In this thesis we focus on a specific type of semantic mapping task called *place classification*. With place classification we intend the task of identifying and labelling all the rooms that compose a building. To each room it is assigned a *semantic label* which indicates its function, such as 'office', 'corridor', or 'kitchen'. Ideally, each semantic label should name a room similarly to what a human being would done in the same context. Under these premises, we focus only on indoor environments, or buildings. In place classification the map of the building can either be considered as initially unknown or known to the robot. In the former case, which is called *place categorization*, the semantic map is constructed during the exploration of

the environment. When then the map of the environment is considered as initially known (the model of the environment is trained on the same environment on which is tested) the task is called *place recognition*, [76].

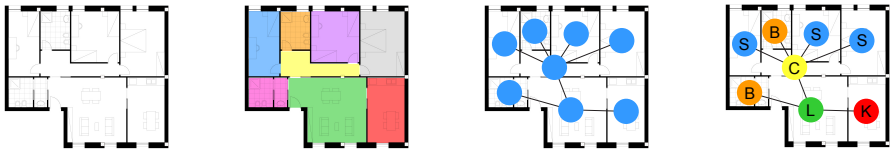


Figure 2.1: *An example of the various steps of semantic mapping, starting from a metric representation of the environment (on the left), which is segmented into different rooms. A topological map is a graph which indicates how rooms are connected between them. A semantic map is obtained by associating each node of the topological map to a semantic label (on the right), with* B *indicating a bathroom,* C *a corridor,* S *a bedroom/studio,* L *a living room, and* K *a kitchen.*

In this work, if not explicitly stated, we refer to a *semantic map* as the result of place classification. This form of representation of the environment usually abstracts from the metric map obtained by (or given to) the robot and can be useful in several tasks, like when the robot is asked to interact with humans or to perform tasks which require a proper understanding of the environment, such as human-robot communication [22,71], autonomous cleaning of the environment [14], task planning and allocation [15]. Semantic knowledge can be used for speeding up search, both when the environment is considered as known *a priori* [55,100] and when the environment is previously unknown [8,46]. The task of searching a specific object by proactively interact within the environment exploiting semantic information is known as *active search*. In [45], it is shown how semantic maps can be used for task planning under uncertain worlds and with possible task failures. In [23] it is shown how semantic knowledge could be exploited by robots to assist human activities by gathering and reporting up-to-date knowledge about the environment by autonomous navigation towards a waypoint and while dynamically adapting to changes and new information.

Semantic categorization of places acquired during the exploration of an unknown building has been proven to improve the exploration performance for a team of robot [91,92]. In a different setup, a guess on the semantic categorization of unknown rooms according to their possible size (as 'smalls', 'medium', or 'big' rooms) is shown to further improve the multi-robot exploration [24,79].

The construction of a semantic map intrinsically involves two steps, namely:

- *segmentation* of the metric map into rooms,
- semantic *classification* of rooms assigning them a label.

Semantic mapping is thus highly related to the mapping process, as the semantic map is usually built on top on a metric map representing the occupancy of the environment. A *metric map* is a globally consistent map of the robot's operating environment [97]. If the map is not previously known to the robot, the robot can use its sensor to acquire the map of the environment while localizing itself with respect to this map, thus, to determine its global pose with remarkable accuracy. This problem is called Simultaneously Localization and Mapping, or SLAM, and has received a wide interest in the last decade. While metric maps can be represented either in 2D or in 3D, in this thesis we focus mainly on the two dimensional case. A typical representation of a 2D metric map of the environment is a grid map, where the environment is divided into cells, similarly to a matrix [97]. Each cell contains a value which indicates if the cell is free or occupied (or the corresponding probability).

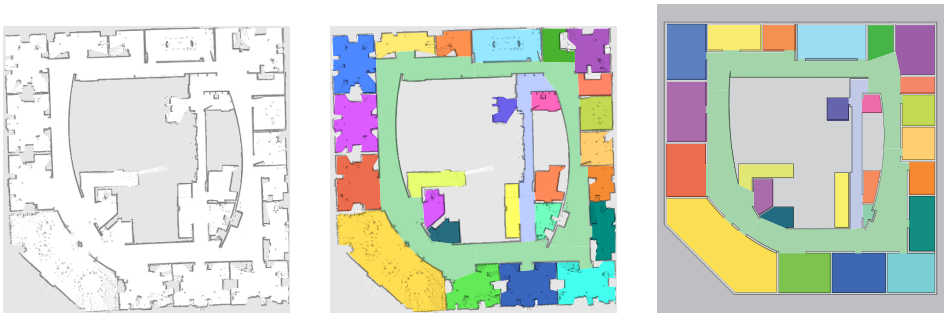
Segmentation and classification highly depend on how the metric map is obtained and are intertwined between them. Several approaches have been proposed in literature, that can be roughly divided into three main categories, depending on which step is performed first. If segmentation is performed before classification, as for instance in [32, 38, 39], the metric map of the robot is divided into separate rooms by identifying narrow passages such as doors. Each room is then classified using information about its structure or other sources of knowledge acquired by the robot.

A different approach, that is used in [64, 66], classifies each cell of the grid map according to the semantic label of the room. This task is usually performed by directly classifying the sensorial input obtained by the robot, as laser range scans. Segmentation in this case is obtained later by merging cells that are classified using the same label.

A third different approach, such the one of [77, 80], segmentation and classification are performed simultaneously during the mapping process, for example by identifying doors within the environment as separation between different rooms.

In this work, we derive semantic maps from the segmentation of the environment, following the first approach presented. Differently from most state-of-the-art approaches, we do not perform the segmentation of the environment directly on the metric map but we perform a further step, called

layout reconstruction. By doing layout reconstruction we do not partition the environment into rooms using the cells of the metric map as primary source of information but, because they may contain inaccuracies or errors, we try to reconstruct the actual layout of the individual rooms and the floor plan of the building. The reconstruction of the layout allows us to introduce a further degree of abstraction on the metric map, where single rooms are represented as geometrical concepts.



(a) *Initial metric map obtained by a robot.* **(b)** *Map segmented into rooms.* **(c)** *The original layout.*

Figure 2.2: *An example of the differences between the room segmentation from a metric map (Figure 2.2b) and the original layout of the building which was used for exploration (Figure 2.2c). Layout reconstruction tries to retrieve the layout such as Figure 2.2c from a metric map as Figure 2.2a.*

Another form of representation of the environment strictly related to the semantic map is the *topological map*. A topological map abstracts the structure of the environment from its corresponding metric map, representing only the connectivity between different parts of the environment. A topological map is typically a graph, where nodes can be poses of the robot or, as in our case, different rooms. In our case we consider as edges the fact that two rooms are connected by a doorway.

The rough outline of the process we use for obtain a semantic map through this work is thus the following:

- we obtain the metric (grid) map from the robot,
- we reconstruct the layout the building and identify the rooms,
- we construct a topological map of the environment,
- we classify each room according to its function.

An example of the segmentation of a floor plan into different rooms, of a topological map and of a semantic map can be seen in Figure2.1.

In the rest of this chapter we provide a significant (although not exhaustive) sample of semantic mapping systems presented in the literature. The outline of the chapter is the following. At first we describe more in detail some state-of-the-art approaches for performing place categorization, following the survey on semantic mapping of [54]. Since mapping is highly correlated to the identification of rooms, we then continue by analyzing techniques for performing room segmentation and layout reconstruction, following the recent survey of [15]. Note that, as segmentation and place categorization are highly connected tasks, some works are discussed in both sections under two different perspectives.

2.1 Place classification

This section surveys a significant (although not exhaustive) sample of place classification systems presented in the literature. For a more exhaustive review of semantic mapping systems the reader is referred to the survey of [54], where a detailed taxonomy of all kinds of semantic mapping techniques is proposed. The typical approach performs place classification by classifying data coming from sensors mounted on robots, like laser range scanners and cameras. The labeling of an environment is built by incrementally combining information obtained from place classification at each time step, possibly using a probabilistic framework. Multiple sensors can be used simultaneously to assign a semantic label to a room; in this case, their outputs are processed together in a multimodal setting.

The system of [66] classifies a single laser range scan as belonging to a 'room', to a 'corridor', or to a 'hallway'. The classifier is based on AdaBoost and considers both features that are extracted from raw sensor data (e.g., average beam length) and features that are calculated after approximating raw data with a polygon (e.g., area of the approximating polygon). Experimental results show that the system is able to correctly classify scans taken in environments different of those used for learning the classifier. A extension of such approach, which uses also features extracted from cameras and obtain a topological map of the environment by grouping together all adjacent cells in the grid map which share the same label, is presented in [67]. Another extension of these two methods for classifying trajectory is introduced in [64].

In [103], a general framework which describes the characteristics of a semantic mapping robot is presented. The main contribution of the paper is a multi-layered spatial representation of the environment, in which different input data and classifiers can be easily integrated. Semantic classification

of places is performed using laser range scans within an approach similar to that [66]. A navigation graph and a topological map are built on top of a line segment-based metric map (in turn, built using data from laser range scanners and obtained from SLAM). From these maps, using camera data, object recognition, and an ontology representing knowledge about indoor environments (e.g., the fact that fridges are usually located in kitchens), a higher-level map of the environment is constructed, that contains information about places and objects.

A similar system which develops a hierarchical semantic map from laser range scanner and camera data is presented by [39], where the metric map is segmented into rooms using a fuzzy morphological operator. The segmentation is then refined using a watershed segmentation, from which the environment topology is extracted. Each room is then classified by anchoring a spatial hierarchy of all the concepts found in the environment within an ontology-like conceptual hierarchy.

In [38], laser range scans and metric maps are used to describe the occupancy of the environment, while data from cameras are used for describing its appearance, in order to obtain a semantic map. The grid map is segmented in rooms. The spatial information gathered by the robot sensors is anchored to a multi-layer ontological semantic model which represents concepts in the domain and relations between them. The semantic model and the anchoring are used to make inferences on other parts of the observed environment.

In [78], the authors present a system which exploits multiple sensors (laser range scanners and cameras) in order to associate semantic labels to parts of the environment. Different features (like SIFT or CRFH for camera images) are extracted from sensory input collected in an environment and semantic labels obtained using geometrical approximations of laser range scans using an approach similar to [66]. Three labels describing the environment are obtained independently using SVMs to classify the features with three different methods. A final label is calculated by merging the three labels with a multi-modal approach.

The system presented in [77] uses a probabilistic framework that applies semantic labels starting from six kinds of environment features (objects, doorways, room shapes, room size, appearance, and associated spaces). Data are acquired using both data from laser range scanners (for classifying the shape of each scan, using an approach similar to [66](and visual data, for detecting the presence of objects or doorways or for checking each room appearance). Environmental features, or cues, are classified independently using SVMs following the approach of [78]. The semantic map is

represented as a probabilistic chain graph model that generalizes Bayesian Networks and Markov Random Fields. The use of a chain graph allows to account for the uncertainties of the sensory models, to classify a newly perceived room also according to the label of the rooms that are connected to it, and to predict the existence of a feature of a certain category (like a room and its label) in nearby unvisited space, extending the semantic map accordingly.

In [80], the authors propose a technique for place recognition and classification from image streams, called PLISS (Place Labeling through Image Sequence Segmentation). Differently from many supervised systems, PLISS can learn and update place models online and is able to operate even in the absence of training data.

Another approach, proposed by [49], integrates metric, topological, and semantic representations with information derived from natural language descriptors obtained from human users. This is done using a factor graph formulation of the semantic properties and inferring these properties by combining natural language descriptions and image- and laser-based scene classifications. The result is a method that, for instance, can infer from the sentence "the exit is next to the cafeteria down the hall" the existence and location of an exit and a cafeteria.

The system in [18] extracts a topological map by partitioning the environment into a set of open spaces connected by narrow passages. The partitioning is done using a fuzzy grid map of the emptiness of the environment, from which the rooms (assumed to be rectangular) are extracted using a fuzzy morphological technique.

Authors in [98] describe a method for integrating the robot metric map with a human representation of the environment. This goal is reached using a semantic map based on data obtained from a laser range scanner and using the concept of region and locations, statistically derived from sensor data. Every room is characterized by features, such as its area, its major and minor axes, and its eccentricity. These features are used to affix a semantic label ('small', 'medium size', or 'large') to rooms.

In [36], a topological map of the environment is extracted using Voronoi Random Fields (VRF). To build this map, a Voronoi graph is extracted from an occupancy grid map. Each point of the Voronoi graph is a node of a conditional random field, and the resulting VRF estimates the label ('room', 'doorway', 'hallway', 'junction', 'other') of each node using features from both the grid map and the Voronoi graph.

In [75] a dynamic Bayesian mixture model (DBMM) is used for performing multiclass semantic place classification. A DBMM composed of a

mixture of heterogeneous base classifiers, using geometrical features computed from 2D laser scanner data to classify each scan, thus assigning a semantic label to the robot trajectory in the environment.

[89] proposes a system for the automatic segmentation of two-dimensional indoor metric maps into semantic units, evaluating spatial functions based on features, such as connectivity (number of paths between rooms which cross a certain area) and functional properties (e.g., the room is a work place). These spatial functions are represented as energy functions, which are evaluated over the points of the metric map. A semantic label is then assigned to each point according to the energy functions. A refinement step adjusts the semantic labels by evaluating their spatial patterns.

[88] uses logistic regression to classify laser range scan as rooms or corridors. The system, differently from [66], is able to classify a single laser scan with different semantic labels. At first, doors are detected using an heuristic method which identifies gaps within walls. Then, a list of 150 geometrical features is extracted from each laser range scan. Authors shows how the use of only three selected features can be effective for performing place classification with a minor degradation in performances, and identify a trade off between classification accuracy and computational complexity of the classification.

In all the above approaches, the task of performing place recognition starts from the environment geometry and shape, i.e., from considering data coming from 2D laser range scans or from the metric map. This fact is particularly evident considering how classifiers are trained using 2D data: datasets obtained directly from robot scans from previous runs are manually classified by a user, and such classified data are then used for training the classifiers. A schema summarizing the typical approach to place categorization using laser range scanners can be seen in Figure 2.3. Different approaches are used in computer vision, for example, for training classifiers for object recognition or for interpreting the appearance of the rooms, where the input images can be obtained through other means that the direct use of a robot (e.g., in [65]).

A recent work that follows this direction is [93], where convolution networks (ConvNets) are used to build a semantic map using 2.5 million $227 \times 227 \times 3$ pixel images from 205 room categories. Those images are taken from sources like Google Images, Bing, or Flickr and labeled by human workers in the dataset PLACES205 from [104]. A semantic category is assigned to each laser range scan according to the output of the ConvNets that classifies the image obtained in the same time. If a grid cell of the metric map is covered by more laser range scans, a probability distribution

on the category is assigned by considering all the classified laser scans that cover a single cell and by propagating the class probability.

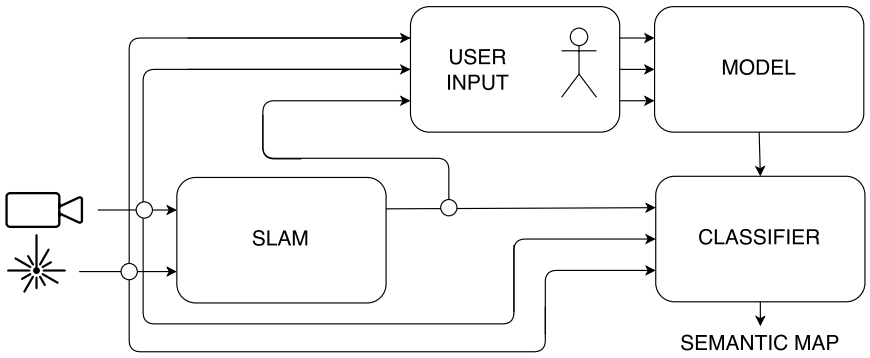


Figure 2.3: *This schema summarizes most semantic mapping approaches. Data from laser range scanner and from camera are acquired in previous runs of the robot (or from data sets) and manually classified. A model of the environment is trained on these data. The map obtained from SLAM is classified using this model.*

Moreover, in most of the methods proposed in the literature the association of semantic labels to perceived spaces is performed incrementally as sensor data are gathered and is often independent of the semantic labels associated to previously perceived spaces (with some relevant exceptions, like [77], that consider adjacent spaces). In this sense, we say that these methods are *local*, since each space is classified using only local features associated to the space itself, largely disregarding the structure of the whole map.

More generally, we can say that the mainstream semantic mapping approach presents a number of limitations, which can possibly undermine its application in real-world applications. The first limitation regards the use of a limited number of (real world) datasets (manually classified by the users) for training and testing the methods. In most of the proposed methods, experimental evaluation is conducted only on a limited number of examples, considering few rooms and using as set of labels only the type of rooms which are present in the dataset. This can potentially reduce the generalizability of place classification methods to different and more heterogeneous settings. We discuss in detail this limitation in Chapter 4, where we present a deeper analysis of some of the place classification methods introduced earlier in this chapter.

A second limitation involves the already discussed fact that semantic

classification of a room mostly relies on features collected within the same room or in its immediate neighborhood, using a local approach. As a consequence, it is not possible to recognize, using standard place classification methods, that two different parts of the same building are similar (both in terms of shape and functionality of the rooms, as in the case of two corridors in two symmetrical wings of the same floor of a building). In Chapters 5 and 6 we discuss how we can use the knowledge obtained on the entire floor of a building in order to perform place classification and how we can adapt a relational framework which considers all the rooms of a building altogether as a tool for semantic mapping.

A third limitation of current place classification approaches is that semantic maps represent knowledge obtained only on the parts of the buildings that have already been visited by the robot. Semantic maps can be used for speeding up multi-robot exploration [79, 91], but almost all of the methods developed in the state of the art only focus on the parts of the environment that have already been explored and perceived by the robot, thus ignoring and not providing any information on the parts of the building that have not been explored yet. In Chapters 7 and 8 this limitation is discussed more in detail together with a proposal for going beyond it.

In [7], a different approach is introduced, which is focused on predicting the structure of a building. The authors consider a knowledge base of $38,000$ rooms (representing the MIT and KTH university campuses). Each floor of the buildings is represented as a graph, where nodes are rooms labeled according to their function (classrooms, offices, ...). The most frequent patterns of rooms are then extracted using gSpan, a well-known algorithm for finding the most frequent subgraphs in a set of graphs. Using this information and given a partial map (graph) of a known portion of a building, the method predicts both the topology and the labels of unvisited rooms by identifying the most common subgraphs that partially overlap the partial map. Differently from the previous methods, that of [7] operates at an higher level of abstraction, reasoning directly on the rooms contained in the knowledge base, obtained from blueprints independently from the use of robots. The task of the proposed method is not to classify the explored rooms but to infer new knowledge on the nearby unexplored spaces. This approach is similar to those we present in Chapter 7 However, while [7] make predictions that consist in identifying the subgraphs (portions of environment) in the knowledge base that match the part of the environment already visited and reason on the next room that can be added on the environment locally, our approach considers the whole structure of a building, which are not necessarily limited to parts present in the initial knowledge

base. We will discuss in more detail this point in Section 8.3.

Another interesting approach, where the use of Statistical Relational Learning (SRL) techniques in semantic mapping has been pioneered, is the one by [56], which proposes a method for obtaining semantic maps of indoor environments using a framework based on a MLN and data-driven Markov Chain MonteCarlo (MCMC) sampling. Using MCMC, the system samples many possible semantic worlds and selects the one that best fits the sensor data. A semantic world contains not only the segmentation of the environment into rooms, but also tries to reconstruct a box-model for each room, thus performing layout reconstruction, and assigns to each room a label like 'room', or 'corridor'. The MLN evaluates the plausibility of each sampled semantic world, considering all the rooms (represented as atoms) at the same time, thus adopting a global approach.

2.2 Room segmentation and layout reconstruction

Automatic analysis of representations of floors of buildings in order to extract structural information is an interdisciplinary topic addressed in different research fields, such as robotics, architecture, computer vision and image analysis. While an exhaustive survey is out of the scope of this section, we cover a significant sample of methods that perform room segmentation and layout reconstruction, with a particular focus on methods developed for mobile robots. Both tasks partition a metric map of a building into a set of rooms. While room segmentation techniques directly segment the metric map into rooms, layout reconstruction methods try to retrieve the blueprint of the building from its map. A comparative example of room segmentation and layout reconstruction on the same metric map obtained from SLAM is shown in Figure 2.2 (this example is obtained by hand). The *layout* of a building is a geometrical representation of the building's walls, rooms, and doorways. Each room is represented either by a polygon (in 2D), a box model or a set of planes (in 3D). A layout of a building thus represents rooms that compose the building structure without considering furniture and without noise, missing data, and occlusions which are often present in a metric map. Layout reconstruction is the task of retrieving the layout from a metric representation of the environment and can be performed at room or at floor level, starting from 2D grid maps [56] and from perfectly-aligned 3D point clouds [6, 68].

Room segmentation can be performed either automatically or interactively, exploiting human indications. In this work, we focus on automatic approaches, which can be online or offline. Online methods incrementally

partition a (possibly incomplete) metric map while it is built. Offline methods consider a previously acquired complete metric map of the environment.

Recently, the authors of [15] presented a survey of room segmentation techniques for mobile robots. They identify four main room segmentation approaches, namely Voronoi-based partitioning, graph partitioning, feature-based segmentation, and morphological segmentation. According to the survey, the most popular approach for room segmentation is the *Voronoi-based partitioning*. Given a metric map, the corresponding Voronoi diagram is composed of points with maximal distance to at least two points belonging to the closest obstacles. As an example, in [96], authors segment the map using *critical points*. Critical points are the points of the Voronoi diagram that are closer to the obstacles than their neighbouring points, and can be used to identify narrow passages such as doorways. Each critical point is used to segment the environment into a number of regions, which are later merged with their adjacent regions according to different heuristics.

Segmentation using *graph partitioning* techniques is used in [16], where a topological map is built incrementally using spectral clustering techniques. A similar approach, using Normalized Cut [87] for partitioning, is used in [12, 49].

Feature-based segmentation techniques perform segmentation by identifying local features of the metric map directly from sensor data, by labeling their locations, and by propagating the labels to other locations with a smoothing process. An example is [64], where an AdaBoost classifier is applied to a feature vector that describes the shape of a single laser scan. The labeled scans are used to determine the labels of the cells of a grid map using an associative Markov network. Adjacent cells with the same label are considered as a single room. This framework can be combined with Voronoi segmentation [36]. In [89], a 2D map of an environment is divided into a set of small units called places. Segmentation and labeling is performed using an energy maximization framework which finds clusters of places and their labels such that each label appropriately describes the functional features of the associated place cluster.

An example of *morphological segmentation* techniques is that of [18], where fuzzy-morphological operator is applied to divide areas that are connected by a narrow passages in a fuzzy grid map.

Methods representing the four above approaches for room segmentation are evaluated and compared in [15]. No method clearly outperforms the others, although Voronoi-based segmentation techniques appear to be the

most accurate.

The approach of [20] uses Canny edge transform and Hough line transform are used to extract lines from a grid map. This larger set of lines is then used for obtaining a scalable grid representation of the environment, similar to an octo-map and called *A-grid*, which is eventually used to perform segmentation using a technique similar to [18] with or without human supervision.

Differently from the room segmentation methods that we have presented so far and that are developed for mobile robots, layout reconstruction methods are typically developed using perfectly aligned point cloud obtained by calibrated camera or laser range scanner. We present a list of such methods by explaining for each one of them what type of representation of the environment they require as input and underlining if they have been developed for robotics

Authors of [6] present a method that performs room segmentation from a 3D point cloud perfectly aligned to a reference system. Points are projected on the floor plan, in order to find walls as sets of points whose projections are close to each other. Differently from other state-of-the-art approaches, walls are identified not considering the occupied space but are induced by a gap area (an area without points) between two walls (two areas with peak of points in the projected space). The entire floor is over-segmented using walls and their projections; each part is used for create a neighbouring graph. Splitting points between two rooms in the neighbouring graph that are not induced by a 'gap-peak-gap' in the point distribution are merged obtaining the final segmentation. A similar solution, where the area is segmented in triangular regions that are later merged together can be found in [99].

In [68], a perfectly aligned 3D point cloud taken in an indoor environment is projected on a 2D plane for obtaining a representation of each wall of the environment. Walls are then clustered together using mean-shift, and each clustered is represented as a single line in the floor plan. A set of polygons, called *faces*, is identified as intersections between lines. Rooms are identified by clustering together adjacent faces using Affinity Propagation. An heuristic is then applied to merge together over-segmented rooms. The number and the exact location of each room is assumed to be known and is used for performing clustering.

In [2, 3] a method for performing layout reconstruction from 2D architectural floor plans is presented. The method recognizes thick and thin walls and builds a geometrical model of the floor plan. This is done by recognizing wordings and standard symbols (such as door openings, windows,

...) that are used in architecture for drawing blueprints. The recognized symbols are then used to reconstruct the layout.

The approach presented in [56] segments a metric map of an indoor environment while reconstructing its layout. It uses a framework based on a Markov Logic Network and data-driven Markov Chain MonteCarlo (MCMC) sampling. Using MCMC, the system samples many possible semantic worlds (a layout of the environment) and selects the one that best fits the sensor data. Differently from our contribution of Chapter 3, which identifies the building structure by analyzing the entire metric map, each transition from a state to another state in the MCMC is based on local edit operations on the layout of a single room, ignoring other parts of the building.

Most of the above methods for reconstructing the layout of a building use a perfectly-aligned collection of 2D (or 3D) point data, as for example acquired by laser range scanners. There are also methods for *scene understanding* that use visual data to reconstruct the layout of a single room. In [37] the layout of a room is extracted from a video sequence obtained using a monocular camera. A set features from the video sequence is used for monocular V-SLAM, which results into a 3D sparse representation of the environment, namely a 3D point cloud is extracted. A large set of planes is fitted to the points, representing all possible candidates for the room layout. A particle filter updates all the layout candidates and it is used to extract the final layout of the room.

In [48], the layout of a cluttered room is recovered from extracting straight lines from images and grouping them according to three mutually orthogonal vanishing points. Vanishing points are used to generate candidates for the box layout. The more robust box layout estimation is then selected according to edge-based image features. The box layout represent the position and shape of each room's walls, floors and ceiling as a set of planes and can be retrieved from a single image of a cluttered room. However, the method focuses on single rooms and it has not been generalized to consider multiple rooms altogether.

CHAPTER 3

Reconstructing the structure of buildings

In this chapter we propose a method that reconstructs the layout of buildings starting from 2D metric maps. The method derives from the metric map a more abstract representation of the structure of the building in order to reason on a "clean" and stable knowledge. In this sense, our approach differs from room segmentation methods presented in Chapter 2, which are typically applied directly on the metric map, for example by identifying doorways or narrow passages as the features that separate and connect two rooms. Although often accurate, metric maps are built from data perceived by robots and can be affected by different sources of uncertainty, such as noisy sensor readings, motion errors, partial or missing data, and occlusions. Our method approximates the metric map and obtains room segmentation performance similar to that of the state-of-the-art methods, while showing more robustness to incomplete data.

The proposed method identifies the *representative lines* along which the walls of the building are aligned and uses these lines to segment the area in smaller parts, called *faces*, which are finally clustered in rooms. The representative lines allow to find regularities between parts of the same building; for instance, two rooms placed at the opposite sides of the building can have aligned walls or a long corridor can connect rooms with the same

shape and lying on the same wall. Our method and can be applied to different input metric maps, either obtained by a robot or received from an external source: complete grid maps, partial grid maps, evacuation maps, and blueprints. (An example of an application of our method to partially explored maps is presented in Chapter 8.)

The outline of the chapter is the following: at first, we illustrate our method for layout reconstruction, focusing our attention on metric maps obtained by a robot as input. Then, we evaluate the performance of our method also by comparing our results to those of standard techniques for room segmentation. Finally, we illustrate how our method can be extended with a pre-processing phase to reconstruct the building layout from evacuation maps and floor plans.

3.1 Our method for layout reconstruction

The method we propose in this chapter starts from a metric map, identifies the walls in it, and uses the walls for segmenting the map into rooms and for reconstructing the layout of the environment. The method is based on a number of steps executed sequentially. A sketch of the algorithm is reported as Algorithm 1. Some of the steps are inspired by the approach of [68] and detailed in the following with the help of a running example (Figure 3.1). More precisely, we use the method developed in [68] for dividing the metric map in smaller parts (which are used to identify rooms) using a set of lines. However, differently from our approach, [68] reconstructs the 3D layout of a building from 3D complete point clouds aligned and registered in the same coordinate system and knowing the exact number and location of rooms.

The starting point is a metric map M representing an indoor environment, typically a floor of a building. We assume that M is a 2D grid map, as the one in Figure 4.4a, namely a two-dimensional matrix of cells (pixels), each one representing the probability that the corresponding area is occupied by an obstacle. The metric map M can be obtained from different sources, such as map building (SLAM) algorithms, floor plans, and evacuation maps, as described in the following. Differently from [68], our method can be used with partially explored maps (an example of such use is presented in Chapter 8) and can cope with misaligned laser scans (maps that are not perfectly aligned are adjusted by our method, as shown in Section 3.2).

The first operation to M is the detection of significative edges using the Canny edge detection algorithm [19], which partitions the cells of M

Input: a grid map M
Output: the reconstructed layout $\mathcal{L}$
/* Compute features from the map */
$M' \leftarrow$ *CannyEdgeDetection*(M)
$S \leftarrow$ *pHoughLineTransform*(M')

/* Obtain contour of the map */
$M'' \leftarrow$ *thresholdMap*(M, q)
Inner $\leftarrow$ *computeMapContour*(M'')

/* Create clusters of collinear segments */
$\mathcal{C} \leftarrow$ *meanShiftClustering*(S)
$\mathcal{C}' \leftarrow$ *spatialClustering*$(\mathcal{C}, line\_distance)$

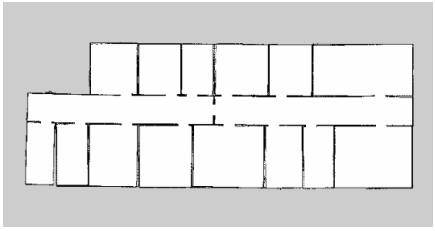
/* Find faces from representative lines */
$lines \leftarrow$ *getsRepresentativeLines*$(\mathcal{C}')$
$F \leftarrow$ *findFaces*$(lines)$

/* Compute spatial affinity between faces */
$L \leftarrow$ *computeAffinityMatrix*(F)
$D \leftarrow$ *diag*$(\sum_{j=1}^{n} L_{i,j})$
$A \leftarrow D^{-1}L$

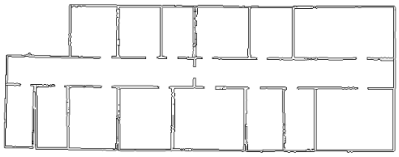
/* Remove faces outside border */
for $f \in F$ **do**
  if $f \cap$ *Inner* $= \emptyset$ **then**
    $F \leftarrow F \setminus f$
  end
end

/* Cluster together faces into rooms */
$\mathcal{L} \leftarrow$ *DBSCAN*$(F, A, \epsilon, minPoints)$
return $\mathcal{L}$

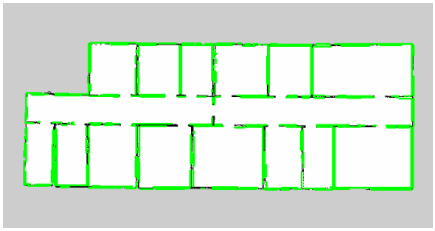
Algorithm 1: Our method for layout reconstruction.



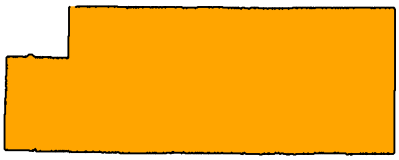
(a) *Metric map.*



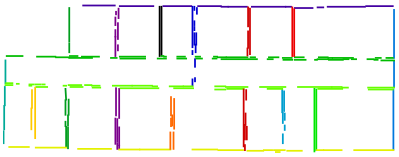
(b) *Canny edge detection.*



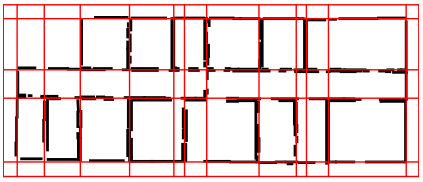
(c) *Hough line transform.*



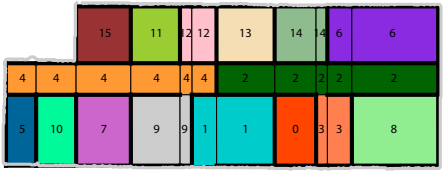
(d) *Map contour.*



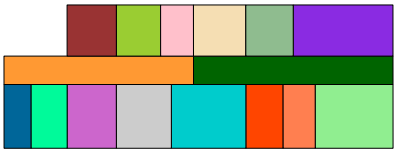
(e) *Clustering of line segments.*



(f) *Representative lines and faces.*



(g) *Clustering of faces*



(h) *Reconstructed layout.*

Figure 3.1: *An example run of our method for layout reconstruction.*

into free space or obstacles. More precisely, for each cell $c \in M$, c is considered as free space its occupancy probability value is less than *minVal* and as an obstacle if its occupancy probability value is more than *maxVal*. If the occupancy probability value of a cell c is between *minVal* and *maxVal*, then c is considered as an obstacle only if it is adjacent to another cell already classified as obstacle (otherwise, c is considered as free space). The binary metric map resulting from the application of the Canny edge detection algorithm is called M' and an example is shown in Figure3.1b.

The set of edges (obstacle cells) is then processed by a standard probabilistic Hough line transform algorithm [52] to detect the set of line segments S that approximate the edges in M'. This well-know algorithm operates by considering lines in the form $\rho = x\cos(\theta) + y\sin(\theta)$ and by summing up the number of obstacle cells of M' that lie on ("hit") the line corresponding to each combination (ρ_i, θ_i). If a combination (ρ_i, θ_i) has a number of hits larger than a threshold, then the line $\rho_i = x\cos(\theta_i) + y\sin(\theta_i)$ is projected on M' and the line segments corresponding to obstacles are identified. The output of the application of the Hough line transform algorithm is thus a set of line segments S representing the "walls" of the environment. Figure 3.1c shows such line segments in green.

Then, the contour of the map is obtained using the contour detection algorithm of [94]. The contour is obtained from the original map M after the application a threshold q that divides the cells in free or occupied. Figure3.1d shows in yellow the area inside the map border.

The line segments S identified by the Hough line transform are then clustered together according to the angular coefficients of their supporting lines using the mean shift clustering algorithm [25]. The mean shift algorithm iteratively shifts data points $P \subseteq \mathbb{R}^n$ to their mean within a neighborhood. At step 0, the cluster centers T^0 coincide with the initial points P. At step i, the cluster centers $T^i = \{t_i\} \subseteq \mathbb{R}^n$ are calculated as follows. The sample mean $m(t_i)$ is computed with kernel function K at $t_i \in \mathbb{R}^n$:

$$m(t_i) = \frac{\sum_{p \in P} K(p - t_i)p}{\sum_{p \in P} K(p - t_i)} \tag{3.1}$$

The cluster center for the next step is changed from t_i to $m(t_i)$ for all $t_i \in T^i$. This process continues until $\max(m(t_i) - t_i)$ is sufficiently small. In our case, the set of data points P is the set of line segments S and the kernel function used in Equation (3.1) is:

$$K_1(\theta) = \begin{cases} \left(1 - \frac{1-\cos(\alpha-\theta)}{h}\right)^2 & \text{if } \frac{1-\cos(\alpha-\theta)}{h} \leq 1 \\ 0 & \text{otherwise} \end{cases} \tag{3.2}$$

where θ is the angle of the line segment (data point), α is the angle corresponding to the cluster with center t_i, and h is a parameter (set, after some preliminary tests, to 0.023 in our experiments).

At the end of the angular clustering, we obtain a set of clusters $\mathcal{C} = \{C_1, C_2, \ldots\}$ such that each $C_j \subseteq S$ and $C_1 \cup C_2 \cup \ldots = S$. Each cluster represents the set of line segments with similar angular coefficient α.

Next, the line segments belonging to the same angular cluster C_j are further clustered according to their spatial separation. Consider two line segments s and s' belonging to an angular C_j with angular coefficient α and call l and l' the lines passing through the middle points of s and s' and with angular coefficient α. If the distance between the parallel lines l and l' is less than a threshold (set, after some initial tests, to 90 cm in our experiments), then s and s' are in the same spatial cluster. At the end of this step, we have a set of clusters $\mathcal{C}' = \{C_{1,1}, C_{1,2}, \ldots, C_{2,1}, C_{2,2}, \ldots\}$, such that $C_1 = C_{1,1} \cup C_{1,2} \cup \ldots$ and $C_2 = C_{2,1} \cup C_{2,2} \cup \ldots$, and so on. Figure3.1d shows the results of the spatial clustering where line segments belonging to the same cluster in $\mathcal{C}'$ are displayed with the same color.

At this point, for each cluster $C_{j,k}$, a *representative line* $l_{j,k}$ that represents all line segments in $C_{j,k}$ is determined. The representative line is computed as the line with angular coefficient equal to that of C_j and that passes through the median of the set of middle points of the line segments in $C_{j,k}$. Representative lines are used to segment the area of map M' in cells, as shown in Figure3.1f. Each representative line, in red, indicates the direction of a wall within the building. The intersections between all lines divide the map into different areas, called *faces*. Call F the set of faces.

Rooms are determined by grouping together faces. Adjacent faces that are divided by a wall should belong to different rooms, while adjacent faces that are divided by edges not corresponding to walls should be grouped together in the same room. For each pair of faces (f,f') that share a common edge we compute a weight $w(f, f')$. Specifically, given an edge $e_{f,f'}$ shared by two faces f and f' (and belonging to the representative line $l_{j,k}$ of a spatial cluster $C_{j,k}$), its weight is calculated (as described in [68]) as:

$$w_{f,f'} = \frac{cov(e_{f,f'})}{len(e_{f,f'})}$$

where $len(e_{f,f'})$ is the length of $e_{f,f'}$ and $cov(e_{f,f'})$ is the length of the

projections, on $e_{f,f'}$, of the line segments in $C_{j,k}$. The larger the weight $w_{f,f'}$, the stronger the hypothesis that there is a wall (obstacle) along $e_{f,f'}$ (namely, between faces f and f'). If an edge is completely covered by projections of line segments in $C_{j,k}$, then its weight is 1. Following the definition of [68], weighted edges are used to compute an affinity measure L between all pairs of faces. L is similar to a Laplacian, and its entries $L_{f,f'}$ are defined as follows:

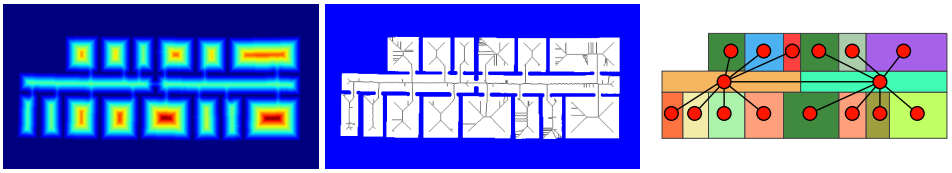
$$L_{f,f'} = \begin{cases} e^{-w_{f,f'}/\sigma} & \text{if } f \neq f' \text{ and } f \text{ and } f' \text{ are adjacent} \\ 1 & \text{if } f = f' \\ 0 & \text{otherwise} \end{cases}$$

From matrix L, a local affinity matrix A is defined as $A = D^{-1}L$, with $D = diag(\sum_{j=1}^{n} L_{i,j})$ where n is the number of faces in F. Each element $A_{f,f'}$ indicates an affinity value considering the local connectivity between faces f and f'. The matrix A is used as input for DBSCAN [31], which clusters faces. DBSCAN groups together faces that are close together in a dense portion of the feature space. For each point (face), it checks if in its ϵ-neigbourhood there are at least *minPoints* points (faces). In our experiments, we set ϵ to 0.85 and *minPoints* to 1.

Before applying DBSCAN, some faces are discarded. Specifically, discarded faces are those called *external* and such that the area of their intersection with the inner area of M (obtained from the contour) is smaller than a threshold δ. Remaining faces are called *partial* if they are adjacent to an external face via an edge whose weight is less than a threshold (0.2 in our experiments) and *internal* otherwise. Partial faces cover the area of rooms that are not fully known according to the data collected in M, as for example during the exploration of a building. An example of use of partial faces is provided in Section 8.2.

Then, DBSCAN is applied to internal faces and a set of clusters $R = \{F_1, F_2, \ldots\}$ of F is obtained. A room r_i is a polygon obtained by merging together all the faces belonging to the cluster F_i. Figure 3.1g shows the results of DBSCAN applied to our example map. The number in each face indicates its cluster F_i. Different clusters are displayed with different colors. The final reconstructed layout $\mathcal{L} = \{r_1, r_2, \ldots\}$ is finally displayed in Figure3.1h. Note that it is a "clean" representation of the original grid map of Figure4.4a.

$\mathcal{L}$ represents the building structure but does not convey any information about the connectivity between rooms. Thus, we integrate our approach with a method that finds the connectivity between rooms of $\mathcal{L}$. We apply



(a) *Distance Map.* **(b)** *Voronoi Diagram.* **(c)** *Topological Map.*

Figure 3.2: *Computation of the topological map from layout*

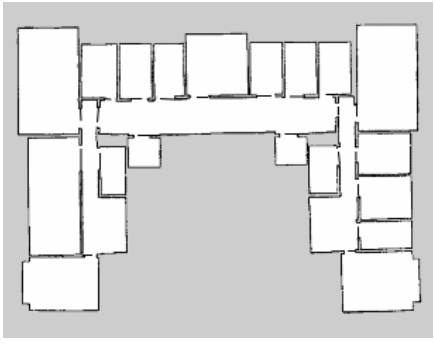
to the original map M a distance transform. Then, we compute a Voronoi diagram, which is superimposed to the layout $\mathcal{L}$. Connections are stored in an undirected graph called *topological graph*. A node in the topological graph is created for each room of $\mathcal{L}$. If there exists an edge in the Voronoi diagram between two adjacent faces f and f' such that $f \in F_i$ and $f' \in F_j$, $F_i \neq F_j$, then we add an edge between the two nodes representing r_i and r_j. The point where the Voronoi diagram intercepts the edge between f and f' is considered as a doorway.

An example of the method for steps used for obtaining a topological can be found in Figure3.2.

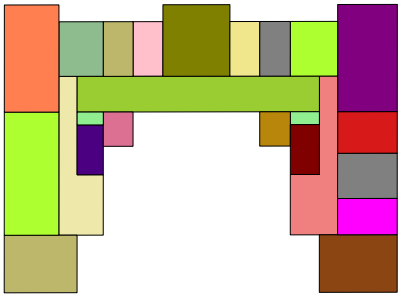
3.2 Experimental results

In this section we evaluate our method for reconstructing the layout of a building from a metric map. Our method is implemented in Python and experiments are run on a single core of an Intel Core i7-3610QM@2.30 GHz CPU with 8 GB RAM. For a single metric map, on average, layout reconstruction takes approximatively 12 seconds.

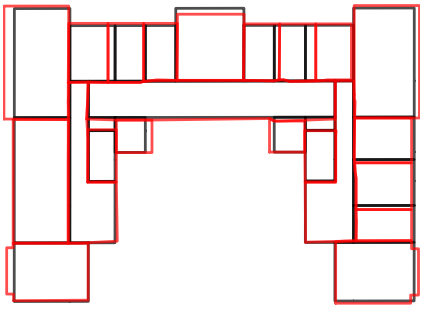
A reconstructed layout is expected to present three main characteristics: (1) all the rooms of the real building (and only them) should be in the layout; (2) the shape of each reconstructed room should match that of its real counterpart; (3) the connectivity between rooms in the layout should match the connectivity of rooms in the real building. Evaluation is performed both visually and quantitatively, comparing the reconstructed layout $\mathcal{L}$ and a ground truth layout Gt. Following the approach of [15] we introduce two matching functions between rooms in $\mathcal{L}$ and in Gt, namely *forward coverage* FC and *backward coverage* BC. FC represents how well the reconstructed layout $\mathcal{L}$ is described by the ground truth layout Gt, while BC represents how well the ground truth layout Gt is described by the



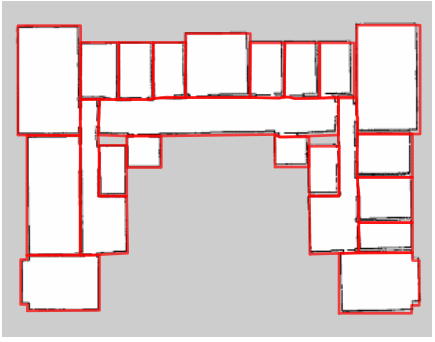
(a) *Grid map.*



(b) *Reconstructed layout.*



(c) *Comparison of reconstructed layout (black) and ground truth (red).*



(d) *Comparison of grid map and ground truth (red).*

Figure 3.3: *An example of a layout reconstruction.*

Layout reconstruction	
A_{BC}	$87.8\% \pm 4.5\%$
A_{FC}	$87.6\% \pm 5.7\%$

(a) *Layout reconstruction accuracy on* 20 *grid maps obtained from gmapping.*

	no furniture		furniture	
	precision	recall	precision	recall
morph	88.5% ± 9.2%	**98.1%±2.4%**	90.5%±8.1%	84.6% ± 7.2%
distance	88.4 % ± 9.3%	96.9% ± 2.8%	88.4%±8.5%	76.1%±12.3%
Voronoi	**94.8% ± 5.0%**	95.0% ± 2.3%	**94.5%±5.1%**	86.6%±5.2%
feature-based	90.2% ± 8.0%	89.2%±11.8%	87.0%±14.5%	85.5%±7.2%
our method	90.1% ± 6.3%	90.0%±4.1%	94.0% ± 2.2%	**89.5% ± 6.2%**

(b) *Room segmentation accuracy obtained by four state-of-the-art methods (first four data rows, from [15]) and by our method (last data row).*

Table 3.1: *Accuracy of layout reconstruction (a) and of room segmentation (b).*

reconstructed layout $\mathcal{L}$:

$$FC : r \in \mathcal{L} \mapsto r' \in Gt$$

$$BC : r' \in Gt \mapsto r \in \mathcal{L}$$

For each room $r \in \mathcal{L}$, FC finds the room $r' \in Gt$ that maximally overlaps r; while for each room $r' \in Gt$, BC finds the room $r \in \mathcal{L}$ with the maximum overlap with r'. Calling *area*() a function that computes the area of a polygon, overlap between a room $r \in \mathcal{L}$ and a room $r' \in Gt$ is defined as *area*$(r \cap r')$. Matching functions FC and BC are then used for computing two measures of accuracy called *forward accuracy* A_{FC} and *backward accuracy* A_{BC}:

$$A_{FC} = \frac{\sum_{r \in \mathcal{L}} \textit{area}(r \cap FC(r))}{\sum_{r \in \mathcal{L}} \textit{area}(r)}$$

$$A_{BC} = \frac{\sum_{r' \in Gt} \textit{area}(r' \cap BC(r'))}{\sum_{r' \in Gt} \textit{area}(r')}$$

The number of rooms in $\mathcal{L}$ and Gt can be different, due to over- or under-segmentation. Over-segmentation results in high A_{FC} and low A_{BC}, while under-segmentation results into high A_{BC} and low A_{FC}.

We consider 20 grid maps obtained using the gmapping[1] algorithm of

[1] `http://wiki.ros.org/gmapping`

ROS[2] on data collected by a robot equipped with a laser range scanner moving in 20 school buildings simulated in Stage[3]. It is important to point out that the evaluation is performed by comparing the layouts reconstructed from the grid maps built by the robot and the real layouts of the simulated buildings fed to Stage (i.e., the ground truth). This allows us to evaluate if our layout reconstruction approach is able to cope with noise and errors introduced in the mapping process. The ground truth represents the actual blueprint of the environment which is provided to Stage in the form of a bitmap image of the map. Layout reconstruction accuracy is reported in Table 3.1a. Our method is able to reconstruct successfully and with good accuracy the layout of the original buildings. Figure 3.3 presents a grid map obtained by the robot (Figure 3.3a) in one of the worlds and the corresponding layout reconstructed using our method (Figure 3.3b). For this particular example, we have $A_{FC} = 90.1\%$ and $A_{BC} = 93.3\%$. In Figure 3.3c the reconstructed layout (in black) and the ground truth building layout (in red) are superimposed. Although the reconstructed layout is generally accurate, there are small differences in the geometry of rooms, due to approximations introduced by our method, which result in a slight performance degradation. In Figure 3.3d the grid map and the ground truth building layout (in red) are superimposed. While the grid map provides a good representation of the environment, some inaccuracies, such as irregular gaps between walls, are present. Our method tries to reduce and filter those inaccuracies in the reconstructed layout.

Figure 3.4 and Figure 3.5 show eight other examples of reconstructed layouts from grid maps. In all these cases, our method is able to correctly reconstruct the layout of the environment, coping well with some rotation errors and gaps between rooms in the metric maps. Few inaccuracies are introduced, like long corridors split into smaller units, thus producing oversegmentation. Another error is made sometimes when a there is a small gap within the building (e.g., due to large wall or a pillar), which is misclassified as occupied face. Small rooms, like small closets or deposits, are sometimes filtered out due to approximation. Overall, our method is able to successfully reconstruct the environment introducing only a limited number of small artifacts.

In Table 3.1b we compare our method against four standard room segmentation methods implemented and evaluated in [15] using precision and recall metrics, which are defined similarly to A_{FC} and A_{BC}:

[2] http://wiki.ros.org
[3] http://wiki.ros.org/stage

$$Precision = \frac{\sum_{r \in \mathcal{L}} area(r \cap FC(r))/area(r)}{|\mathcal{L}|}$$

$$Recall = \frac{\sum_{r' \in Gt} area(r' \cap BC(r'))/area(r')}{|Gt|}$$

The evaluation is performed on two datasets of 20 metric maps each originally used in [15], with and without furniture, respectively. When the dataset with furniture is used, our method uses a threshold-based preprocessing step to remove from the metric maps small groups of occupied cells, likely representing furniture and not structural elements of the building (see next section for further details). We stress that that all four approaches of [15] perform room segmentation directly on the metric map, while our method uses representative lines and faces, thus introducing some approximations. An example of the approximations introduced by our layout reconstructions is displayed in Figure 3.6. Figure 3.6a shows the Voronoi-based segmentation of the metric map of Figure 4.4a, as reported in [15]. It can be observed that Voronoi-based methods tend to produce over-segmentation. In Figure 3.6b we compare our reconstructed layout (in black) with the real structure of the building (in red). Despite being able to correctly capture the building layout, for this map we have $A_{FC} = 93.6$ and $A_{BC} = 94.6$, due to some visible approximations. In general, the performance of our room segmentation expressed as precision and recall is upper-bounded by A_{FC} and A_{BC}. However, looking at Table 3.1b, our method performs comparably with the others, with a slightly better performance in segmenting metric maps from the dataset with furniture. In addition, our method computes an high level representation of the metric map by obtaining the reconstructed layout.

The results presented in this chapter are obtained on complete maps with a limited amount of inaccuracies and clutter introduced by occlusions, presence of objects, and partially explored rooms. Regarding maps with occlusions and objects, however, our method can be adapted by introducing some preprocessing steps for identifying and filtering out frontiers, objects, and outliers from laser range scans. Knowledge on the shape and position of walls can be retrieved even from maps of particularly cluttered environments using vision, with techniques as [37, 48]. Walls that are partially covered by furniture can be identified using 3D data and then projected in 2D map, as done in [6, 68] for perfectly aligned 3D point clouds. However, such improvements are beyond the scope of this chapter and are left for further improvements.

Partially explored maps can be reconstructed with our method. Examples of the use of the proposed method with maps where some parts of the rooms are yet not perceived are shown in Chapter 8.

Since our method does not assume a Manhattan world it can be potentially used in non-Manhattan environments, such as those with round walls or with diagonal walls. However, since in our method walls are approximated by straight lines, round walls are approximated by a polyline. An example of this behaviour can be seen in Figure 3.7, where is shown the reconstruction of a map from [15].

Maps that present a (relatively small level of) unalignment are adjusted by our method, as can be seen in the last two examples of Figure 3.4 and in the third example of Figure 3.5. A trade-off exists between the accuracy of alignment that can be reached by our method and its ability to approximate round walls with polylines. This trade-off can be set by modifying the parameters of the line segment clustering step, when collinear walls are clustered together and representative lines are identified. If alignment is not required, round walls can be effectively approximated by fine-grained polylines, but this results in the side effect that line segments that are diagonal due to alignment problems are not straightened up (and are wrongly recognized as diagonal walls). Similarly, strong regularization of unaligned maps results into a coarse approximation of round walls. In all the examples presented in this chapter we preferred strong alignment over good approximation of round walls. Note that, since we consider 2D maps, we make the implicit assumption of having walls perpendicular to floors and ceilings.

3.3 Extracting the layout from floor plans

In this section we explain how the method presented in this chapter can be adapted to analyze floor plans of buildings. More precisely, we introduce a series of preprocessing steps on a floor plan for identifying all the possible information that can be found within it. These steps are illustrated using the example of the floor plan of Figure 3.8.

Floor plans are highly codified forms of representation showing a scale view from above of the relationships between rooms, spaces, and other physical features of a structure. A floor plan can describe a building using different levels of detail. The simplest type of floor plan, similarly to a sketch of an environment, represents only walls (and doors as opening within the the walls). Other features that may be found in a floor plan and that are usually represented with codified symbols are:

- doors,

- windows,
- walls' size,
- stairs or elevators,
- restrooms,
- furniture,
- location of different type of objects,
- evacuation plans,
- the function of all rooms, typically indicated with a label or with a brief text description.

The reconstructed layout of a building from a floor plan can be a useful source of knowledge in robotics, especially when the environment in which the robot operates is initially unknown to the robot but some knowledge can be acquired, as in a situation where a team of robot performs search and rescue operations and an evacuation map of the environment can be seen on a wall. One of the possible uses of such knowledge is *localization*, which usually depends on knowledge of the environment. An example of a method for such application is that of [102], where a robot can localize itself using a floor plan instead of a metric map. A reconstructed layout may introduce some approximations compared to the real shape of the environment, as explained in the previous section. In [10, 13], it is explained how partially inaccurate hand-drawn sketch maps can be effectively used by robots for localization. In principle, using similar approaches, a reconstructed layout of a building can be used for localization even if slightly inaccurate.

Another possible application of our method is to automatically obtain knowledge about indoor environments by automatically segment and reconstruct the layout of many buildings from the set of their floor plans, in a similar way to what is done in computer vision by collecting large data sets of classified images. More details of this application are illustrated in the following chapter.

Finally, a floor plan, if correctly analysed, can provide a series of interesting information about the environment, as shown in the evacuation maps of Figure 3.8 where the position of every door of the environment, the paths used for evacuation, and the semantic label of each room are reported.

The preprocessing steps are executed sequentially, and an example of them is shown in Figure 3.8. In the first step, we identify symbols within

the map using the Template Match algorithm from [85], which finds the presence of a specific template image, in a target image. In our case we use as templates a set of architectural symbols representing doors, windows, fire extinguishers, emergency exits, and others, which have been taken from several sources like [69, 95]. These symbols are rather codified and standardized in CAD drawing software programs used to draw blueprints, such as ArchiCAD.

At first, we identify doors from templates, as shown in Figure 3.9a. The position of each door is saved for helping during the construction of the topological map once the layout has been reconstructed (see Section 3.1). The instances of the door templates in the floor plan are then removed, and the door-gaps are filled with a black line in order to facilitate segmentation as shown in Figure 3.9b. Other objects present in the map are then identified and removed using the same algorithm for template matching, as illustrated in Figure 3.9c, where a hydrant template is used. If the removal of objects results in gaps along the the representative lines (obtained as explained in Section 3.1), the gaps are filled with line segments oriented as the representative lines. Once each symbol is identified and filtered out of the image, we identify the text using a standard OCR technique from [90]. The identified text can be fed to a reasoner in order to obtain a prior on the semantic label of each room. This can be done, as an example, by transforming text data into a logic relational representation, similarly to what is done in [38]. A logical relational representation of semantic maps is presented in Chapter 6. However, this application is beyond the scope of this chapter.

A threshold is then applied to the image of the floor plan, converting it from gray scale or colors one into a black and white image. An erosion operator is used to filter out some inaccuracies that, may results after removing the identified templates. An example of these steps, with the identified text highlighted in blue, can be found in Figure 3.9d.

Text is then removed from the floor plan, obtaining a 2Ds map as the one of Figure 3.9e, which is then given as input to our method of Section 3.1 for performing layout reconstruction. The final result of the reconstructed layout of the example map can be seen in Figure3.9f.

The identification of doors and their replacement with black segments to fill the door gaps highly improve the performance of our method in layout reconstruction, since it helps in separating cells which belong to different rooms.

Some results obtained using this framework are presented in Figure 3.10, in Figure 3.11, and in Figure 3.12. Our method is able to reconstruct the layout successfully, only adding some minor inaccuracies such as the seg-

mentation of long corridors into smaller pieces.

3.4 Discussion

In this chapter we have presented a method for reconstructing the layout of a building given its metric map or its floor plan. Our method identifies lines that represent walls of the building and uses them to approximate the shape of the rooms. Experimental results show that our method performs room segmentation comparably with state-of-the-art methods, but it is able to cope with metric maps and with evacuation maps provided as input.

In Section 3.3 we presented an extension of our method which allows the processing floor plans. Standard techniques for image feature detections can be used to extract and filter out from the floor plan relevant features, such as doors, objects, and text fragments that indicate rooms functions. An example of a standard method for feature extraction applied to doors and text detection is shown in Section 3.3.

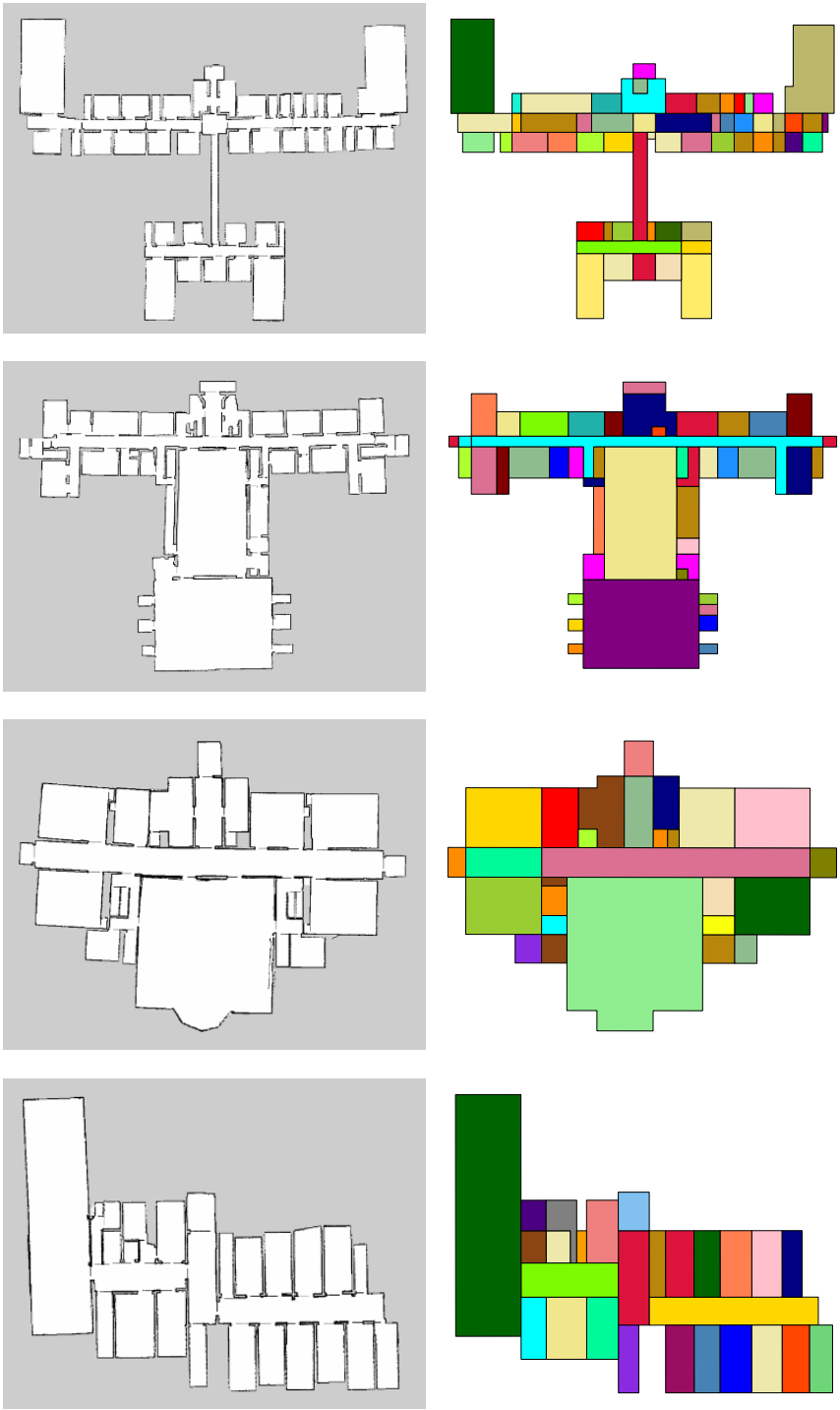


Figure 3.4: *Examples of layout reconstructions from metric maps.*

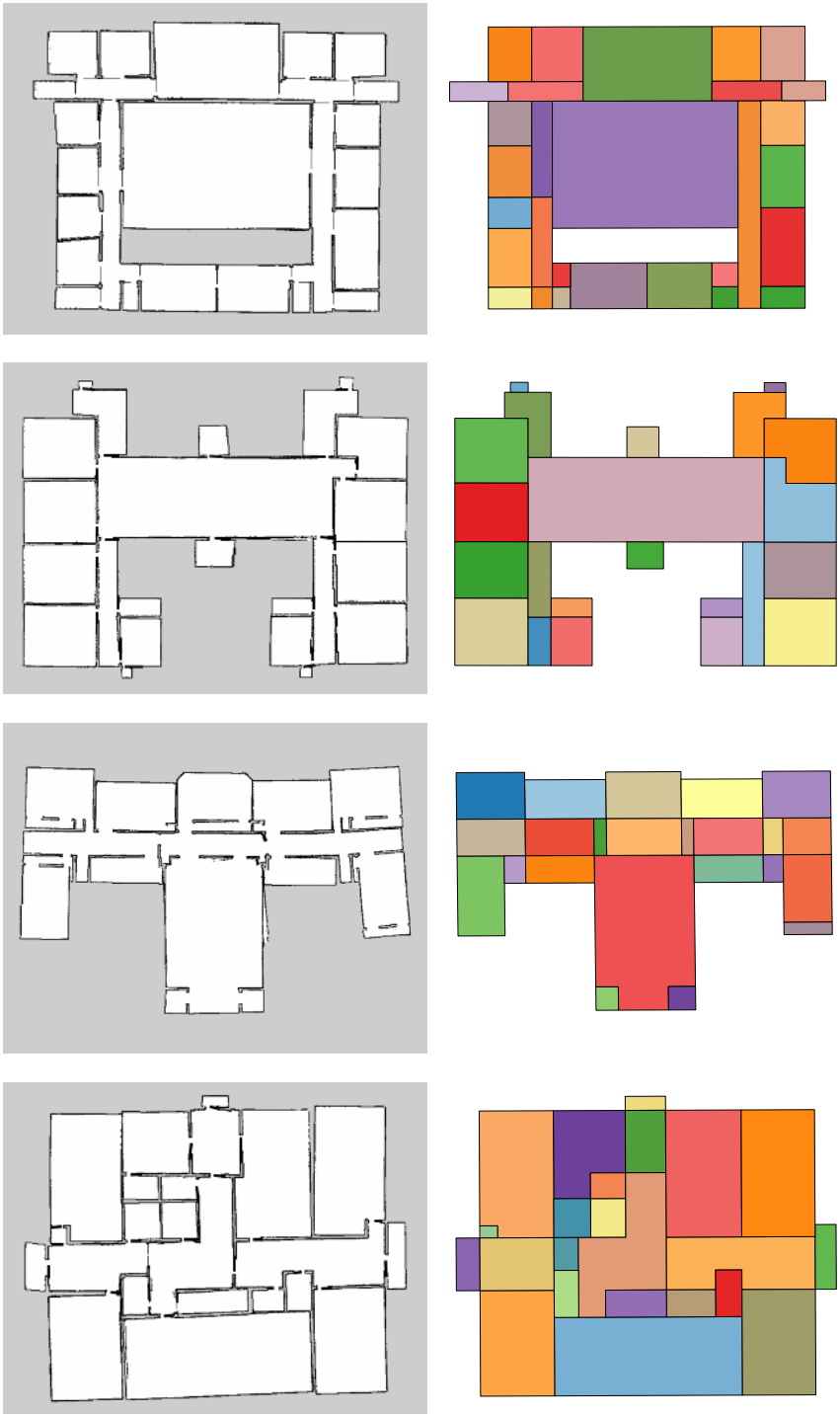
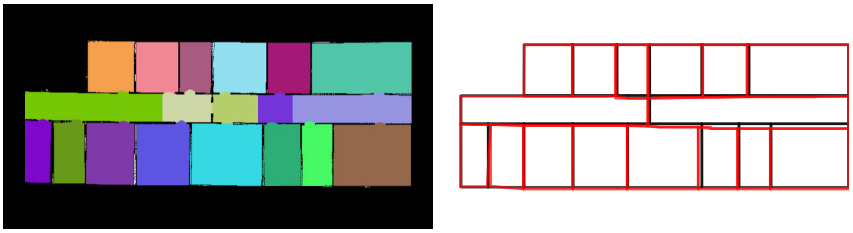
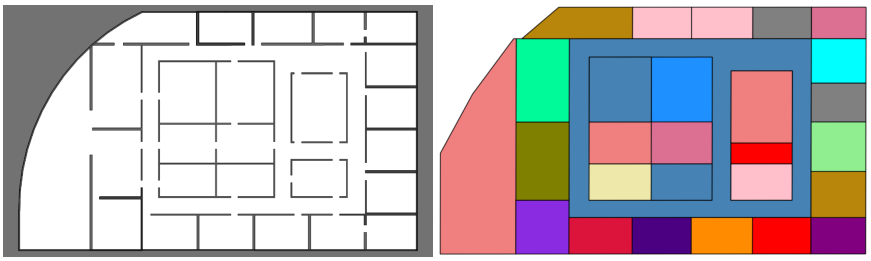


Figure 3.5: *Examples of layout reconstructions from metric maps.*



(a) (b)

Figure 3.6: *Segmentation by a Voronoi-based approach, from [15] (a) and comparison between our reconstructed layout (black) and the ground truth layout (red) (b) for the metric map of Figure 4.4a.*



(a) (b)

Figure 3.7: *An example run of our method on an environment with round walls taken from a map used in [15].*

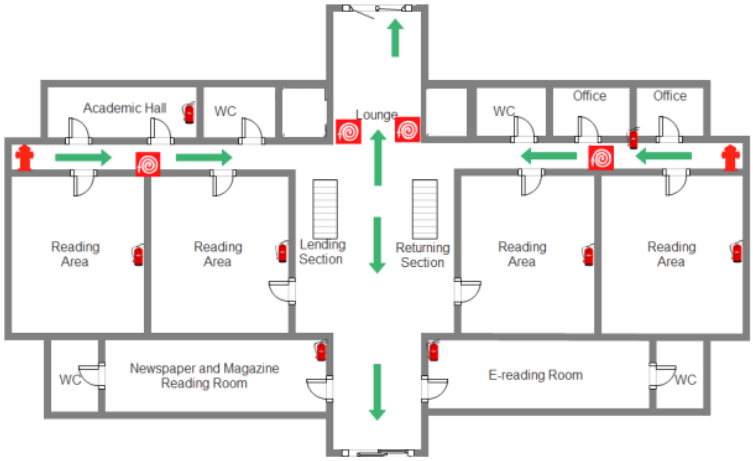
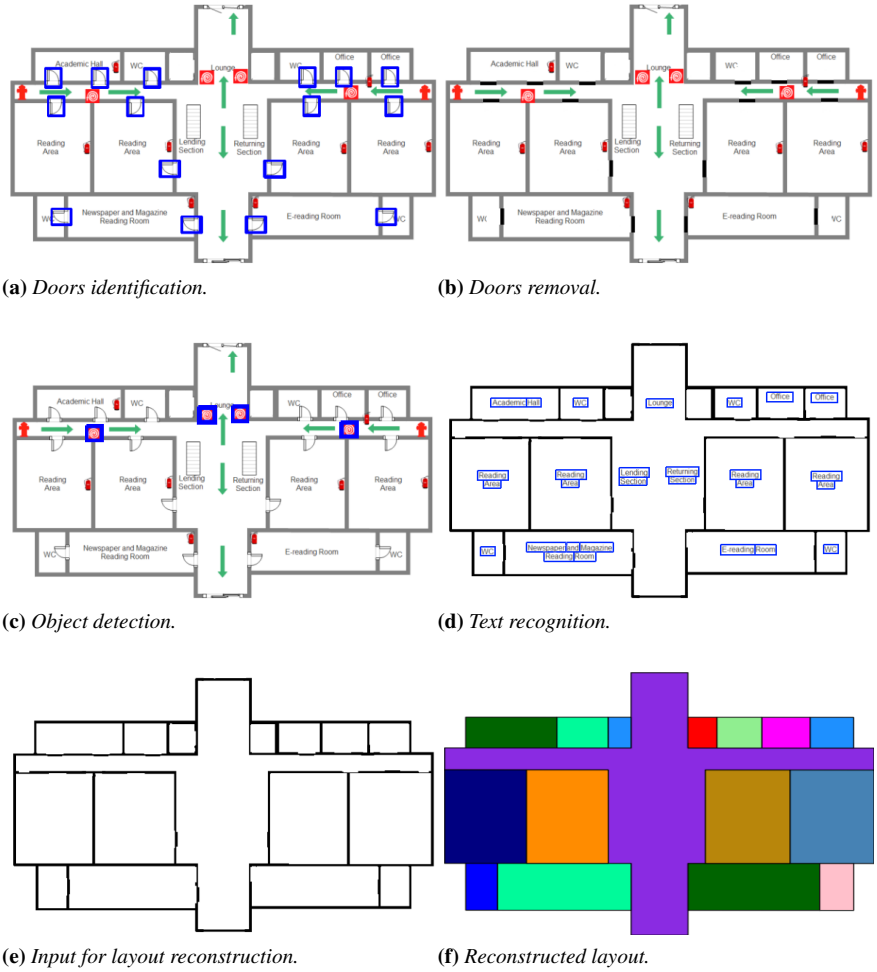


Figure 3.8: *An floor plan of a school used as evacuation map.*



(a) *Doors identification.* **(b)** *Doors removal.*

(c) *Object detection.* **(d)** *Text recognition.*

(e) *Input for layout reconstruction.* **(f)** *Reconstructed layout.*

Figure 3.9: *Example of preprocessing steps for identifying and filtering symbols from an building floor plan. The initial floor plan is that of Figure 3.8.*

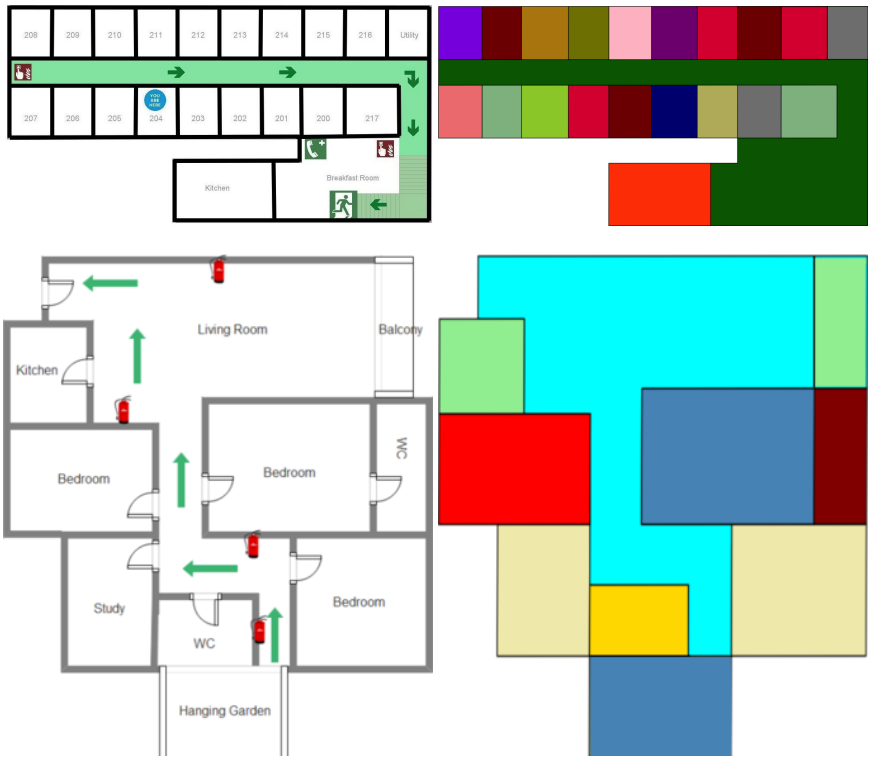


Figure 3.10: *Examples of layout reconstructions from floor plans.*

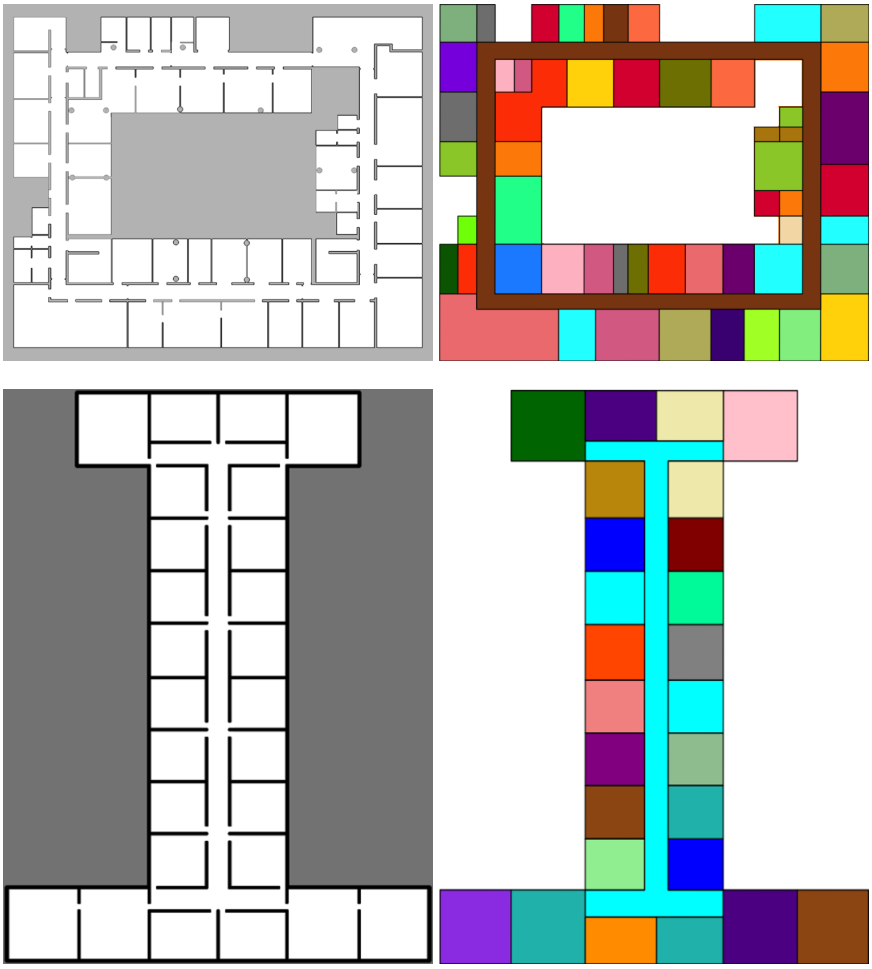


Figure 3.11: *Examples of layout reconstructions from floor plans.*

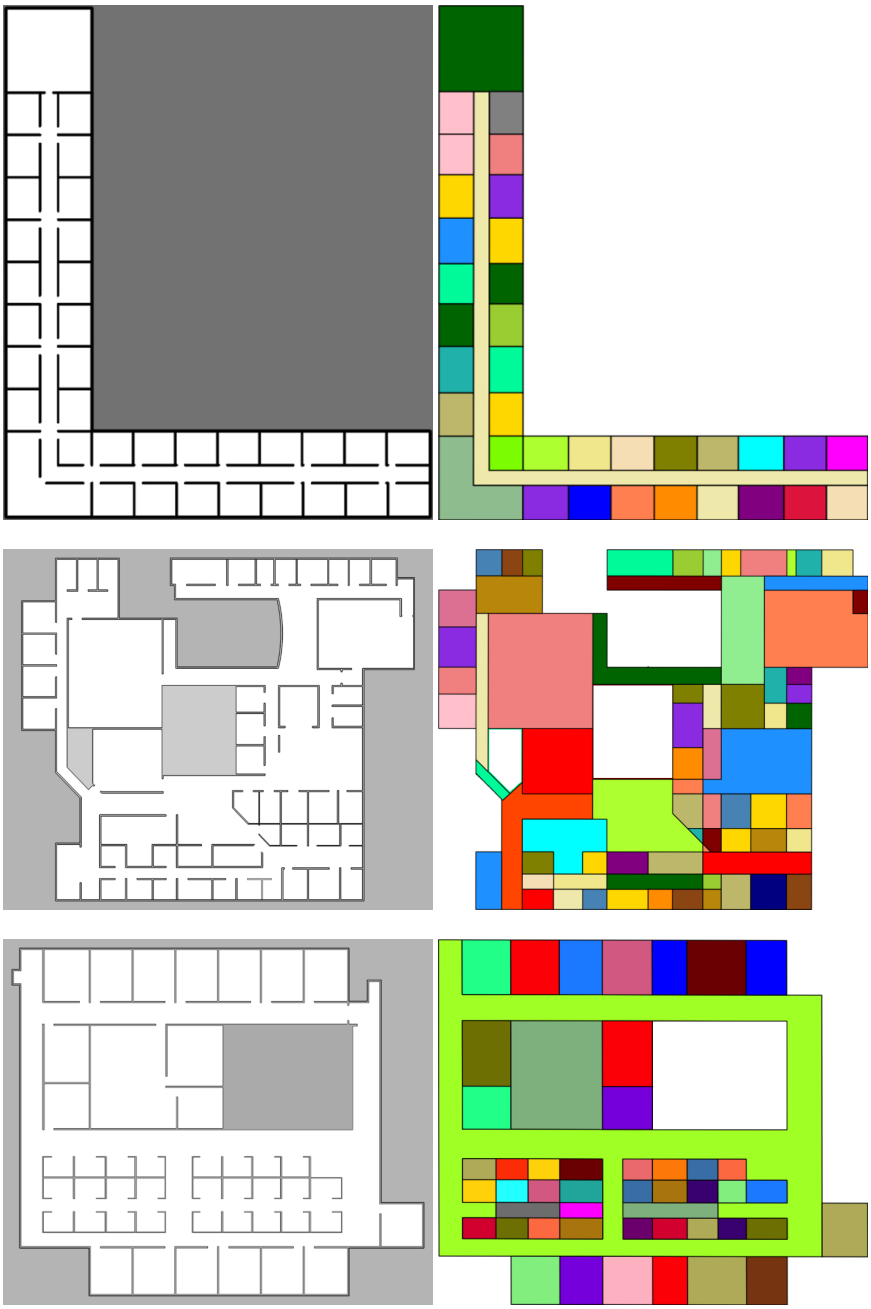


Figure 3.12: *Examples of layout reconstructions from floor plans.*

CHAPTER 4

Building types

In Chapter 3 we have introduced a method for extracting abstract representations of environments from metric maps. In this chapter we discuss how we can reason on these representations by using data sets of floor plans of buildings categorized accordingly to their function.

Broadly speaking, semantic mapping of indoor environments is a process which involves three actors: a *robot* which categorizes an environment, an *environment* to be categorized, and a categorization relation imposed on the environment by the labels (the actual *mapping method*). Other factors involved in this process are the type of *sensors* the robot is equipped with the type of *features* extracted from sensors, how the map is represented and how the semantic categories (*labels*) are matched to areas of the metric map.

Typically, when authors present a new method for performing semantic mapping, they focus their attention on a subset of these factors, maily on the mapping method, on the robot, on its sensors, and on the features that are extracted from the sensors. In this chapter we want to focus on another factor that characterizes a semantic mapping method, the environment.

Focusing on the environment, we aim to discuss some limitations in current semantic mapping systems that are hindering their general appli-

cability and by proposing some solutions for overcoming them. The first limitation is related to the small number and the large degree of homogeneity of the buildings usually employed to train semantic mapping systems. As a consequence, these systems are at risk of overfitting over specific kinds of buildings. The second limitation is about the lack of a principled way to define semantic labels for rooms. We propose an abstract framework for semantic mapping that can move towards a generalization of the application of state-of-the-art semantic mapping approaches to different contexts and environments. Attempts to prove a more general semantic mapping framework have been proposed. These consideration are later used in Chapter 6 and Chapter 7 for developing techniques that model indoor environments.

Generalizing with respect to the robot is a topic previously discussed (e.g., in [77, 103]), and multilevel probabilistic modular frameworks, designed to be easily extendable to different sensorial data and different semantic classifiers, were proposed. In this thesis, we provide a framework for generalizing with respect to two other elements, the environment and the labels.

Our framework addresses the first limitation by adopting the ideas of *building type* and *model floor plan*, both taken from architecture, to develop more heterogeneous data sets that help to avoid overfitting and thus increase the generality of the semantic mapping systems. The second limitation is addressed by proposing a hierarchical labeling approach in which the top level contains only two labels, 'ROOM' and 'CORRIDOR', and the lower levels contain semantic labels that specialize those at the top level and that are specific for the single buildings. We assess the validity and the significance of our framework by applying density-based spatial clustering techniques [43] to some large publicly-available data sets. Our results show that our framework can obtain more heterogeneous data sets and that the labels 'ROOM' and 'CORRIDOR' "naturally" emerge from the features used to describe rooms of different kinds of buildings. We explicitly note that the original contribution of this chapter is to provide the seed of a framework in which general semantic mapping systems can be fruitfully developed.

The outline of the chapter is the following. We overview briefly how buildings are categorized (considering which kind of labels are used to describe them in literature). Then, we analyse a data set of buildings in order to identify a context for developing generalizable semantic mapping methods. We finally propose an abstract labeling schema to describe different kinds of environments. The content of this chapter is based on the works of [57, 59]. The proposed labeling schema and the analysis performed on data sets (which is continued in the next chapter) are used in the following

	SENSORS		LABEL				
	LASER	CAMERA	SET	ONTOLOGY	SPECIFIC	NUMBER	NOTES
[38]	✓	✓		✓			
[78]	✓	✓	✓		✓	4	
[92]	✓		✓			2	corridor / room
[39]	✓	✓		✓	✓		commonsense ontology
[103]	✓	✓		✓			generic framework
[66]	✓		✓		✓	4	room / corridor / hallway / doorway
[67]	✓	✓	✓		✓	6	extension of [66]
[89]	✓		✓		✓	4	room / corridor + others
[77]	✓	✓		✓		11	50-room data set

Table 4.1: *Comparison between semantic mapping approaches.*

chapters for developing several applications..

4.1 Labeling in semantic mapping

As explained in Chapter 2, the creation of a semantic map is usually based on an underlying metric map obtained from SLAM. The data contained in the metric map and acquired by robot sensors inform the construction of the semantic map. In this context, the most commonly used sensors are laser range scanners and cameras. This section considers a significant sample of semantic mapping systems found in literature, trying to dwell on the methodologies, the sensors, and the labels involved. Table 4.1 summarizes the main features of these systems. We focus on the sensorial data used as input and on the semantic labels applied to rooms of buildings. Labels can be taken from a fixed set/list (column SET) or from a more structured ontology (column ONTOLOGY). Sometimes, the semantic labels are specific for the rooms in the training and testing sets (column SPECIFIC). We also report the total number of semantic labels used (column NUMBER). For a detailed description of methods listed in Table 4.1 the reader is referred to Chapter 2.

All the listed methods use laser range scanners, and most of them rely also on visual data. In most of the cases, the labels used for semantic mapping are those present within the training data set (column SPECIFIC), i.e.,

no list of general semantic labels is specified in advance

All the works discussed above and listed in Table 4.1 share the fact that classifiers for labeling rooms are developed, trained, and tested with the use of data sets collected with runs of robots in real contexts. Given the difficulty of obtaining these data and the fact that many systems are developed within research laboratories, only a limited number of buildings are used as initial source of data. In some cases, the systems are trained and tested in a single environment. In other situations, only a small number of buildings (often less than 5) is used. This leads to one of the major current limitations of semantic mapping systems, namely their inability to generalize over environments. Given these premises, it is difficult to understand and assess how these systems will perform in different buildings. This drawback has been recently recognized, and attention has been given to generalize semantic mapping systems [29, 77].

Generalization has been attempted along three lines:

1. test the systems on multiple (public) data sets built from real robot runs [66, 77];
2. use transfer learning techniques, trying to exploit knowledge obtained in one context to recognize the features of a new, initially unknown, building [29];
3. increase the size of training data (tens of thousands of rooms rather than only dozens) by using sources different from data sets collected with robot runs in real world, for instance data sets built from floor plans [7].

Although these proposals are contributing effectively to move towards more general semantic mapping systems, they have not yet reached the full goal. Using 50-room data sets (like in [77]) instead of $5/10$-room data sets is an improvement, but still a limited one, if the goal is to have a working system for the countless rooms in the world. The same can be said of using more data sets: the number of available data sets acquired by real robots is relatively small, and the total number of rooms per data set is often less than one hundred. Furthermore, there is an underlying bias: almost all the data sets are acquired directly by research groups and represent university buildings or research laboratories. The consequence is that the systems trained on these data sets have only knowledge of a rather peculiar kind of environment, university campuses, and little or nothing about the rest of the world. Using thousands of floor plans, as in [7], can be a useful tool for generalization, as it allows to collect a huge amount of data with a limited

effort. In [7], $38,000$ rooms are analysed, a number not comparable to the other approaches. However, also these data come from two university campuses (MIT and KTH) that, as all large building complexes, have a high degree of symmetry, resulting in a redundancy of data and overfitting of semantic mapping systems that use them for training.

4.2 Generalizing with respect to environments

In this section we discuss how to develop a semantic mapping framework that is general enough to be used potentially in all indoor environments. Among other things, this requires data sets covering all kinds of environments. The main difficulties that emerge for solving the problem of acquiring data sets of all possible kinds of environments can be summarized into two main topics:

- acquiring data sets with robot runs is difficult and expensive, and
- the types of buildings are countless.

In order to tackle these issues, we developed a data set representing 6000 rooms of indoor environments, belonging to an heterogeneous set of 160 buildings. As source we used blueprints and floor plans of real world buildings. Each room is associated to a feature vector that describes its geometry and to a semantic label indicating its function. The set of features stored in the vector is chosen to mimic the data on the geometry on each room that a robot exploring an environment could potentially collect using, as an example, a laser range scanner. The analysis of these data allows us to try identify some pattern or regularities that can be used for developing a general semantic mapping systems, without having the overhead of directly collecting with a robots such data. Specifically, we try to determine wether or not rooms belonging to building with the same function are similar among them.

4.2.1 Building types

Our analysis starts with the following insight: at a first glance, it may seem that every building has its own identity, as it is different in its structure and conformation from every other one. However, this fact is in disagreement with our daily experience: people can easily orientate themselves and navigate inside a previously unseen building and they fast adapt to and understand almost every new indoor environment they enter.

A building is an artifact created for a specific function, its purpose, because it is built by people for people who inhabit it. This simple observation is often neglected when designing mobile robots that operate in indoor environments. Indeed, these environments are usually treated as natural, fixed, immutable entities, rather than as cultural products, the result and summary of centuries of social evolution, as the modern buildings are. As a consequence, robot designers usually consider the environments in which the robots move as structured, without fully exploiting the implications of their structure.

The function of a building imposes its structure, its floor plan, and the structure of its rooms. Each building, having a precise function, shares some structural features with all other buildings with the same purpose. A *building type* is a set of buildings that have the same function [84]. A building type can be associated to a model that represents the structural features shared by buildings belonging to the type.

In architecture, there is a substantial amount of literature enquiring which are the features of this model well known (as cultural facts) to humans, who use them for localization and orientation, as done in [61].

An example of how the function of a building imposes its structure can be seen by observing that multifloor office buildings are usually structured in the following way: at the ground floor, there are the rooms for public relations and social spaces, and possibly a canteen and conference rooms; at the upper floors, there are office rooms for the back-end activities. Note that, when comparing two school buildings, they are, at an initial sight, different from each other. However, they share a common model that, for example, represents the fact that rooms are typically connected in certain ways. Recognizing this common model helps humans in acting properly in schools they never entered before.

Let now discuss how data sets relative to a building type can be obtained. Beyond actual collection of data in real buildings, it is also possible to use another source of data for training semantic mapping systems. Studies about building types are the analytical moment of architecture [84] and their actual impact on designing buildings are described in several architecture type manuals. Although this architectural knowledge is often not structured, it is discussed with texts, graphics, and sample floor plans of particularly significant buildings. In the rest of this dissertation we use as source of knowledge data sets of floor plans of buildings, each one representing a specific building type.

More specifically, we consider three building typologies: HOUSE (residential buildings and houses), SCHOOL (school buildings), and OFFICE

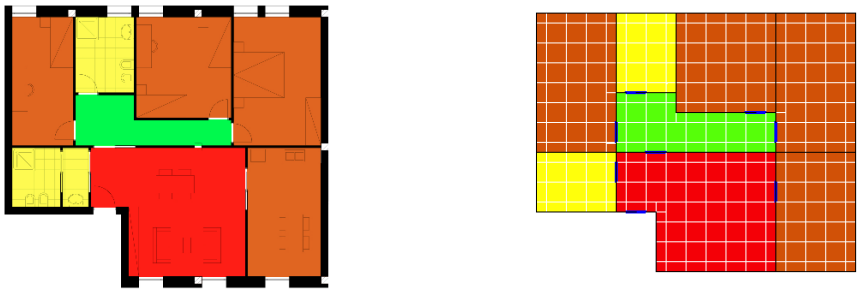


Figure 4.1: *An example of a floor plan of a house drawn with AutoCAD (left) and with our CAD-like software (right). Colors indicate semantic labels (green is corridor, yellow is small room, red is big room, brown is medium room).*

(office buildings and open spaces). These three building types have been chosen because they are particularly significant in the real world and because they are usually considered for experimental activities of autonomous mobile robots.

The data sets have been built as follows. We selected the floor plans of dozens of buildings belonging to the types HOUSE, SCHOOL, and OFFICE from eleven monographic books used for the design and analysis of buildings in architecture (for example, [74] is one of the books used for the School building type). In these books, rooms are labeled according to their function and these labels are assigned by the architects who designed the buildings. Monographic books on building types contain a selection made by experts of particularly representative buildings and are used as reference and as models by architects when developing new buildings. We manually create CAD-like graphical representations of several floor plans (a general standard format for such kind of data is required, as the one used in [101]), extracting information from architectural books and manuals. We then converted these representations in data sets containing rooms labeled according to their functions. With this method we can build data sets composed of thousands of rooms in a relatively quick and easy way. An example of an entry of a data set obtained with this method is shown in Figure 4.1.

Then, each labeled floor plan is fed into another software program that automatically extracts the geometry of each room, which is used for computing a set of features representing each room and which are described in Section 4.2.3. These features and the corresponding label for are used to create entries to populate the data sets H, S, and O (for HOUSE, SCHOOLS

and OFFICE respectively). (All the above steps have been manually performed. In practice, a method like the one presented in Chapter 3 can be used for automatically extract data sets from floor plans without a human supervision. The main limitation of using a manual approach is that the automatically reconstructed floor plans have to be classified semantically by a human supervisor.)

In order to identify if the concept of building type emerges from these data sets, we assigned to such rooms a set of general semantic labels common to all three types. These semantic labels contain only a limited amount of information about the function of the room and about their size. Those labels indicate either the function of connecting rooms (corridors (C) and halls (H)) or the size of other rooms (small (S), medium (M) and big room (B)). We refer to this set of labels as $\mathcal{L}_{general}$:

$$\mathcal{L}_{general} = \{\text{C}, \text{H}, \text{S}, \text{M}, \text{B}\}$$

Note that the selected labels are similar to those typically used in state-of-the-art semantic mapping approaches using only laser range scanners (e.g., [66] uses 'corridor', 'hall', and 'room' and [98] uses 'small', 'medium size', and 'large' room).

4.2.2 Model floor plan

Using the concept of building type to create data sets for training and for testing semantic mapping systems is also interesting from another point of view. Multifloor buildings present, in almost all cases, strong structural symmetries at two levels: intra-building symmetry, namely a correlation between the structure of every floor, and intra-floor plan symmetry, namely each floor plan is composed of the repetition of a room pattern, as can be seen in the highly-symmetrical floor plan of Figure 4.2. These recurrent patterns are a direct consequence of how buildings are designed, because the structure of a building is studied to be consistent to its functionality. A standard pattern is a structure that represents the function of the building and that is replicated across the floors, possibly with some modifications and adjustments [69]. We call *model floor plan* this standard pattern, which represents the implementation of the concept of the building type within a specific building. Within the field of robotics, and of semantic mapping in particular, the analysis of a model floor plan can be interesting, since it can reveal valuable knowledge of the structure of a whole floor and, consequently, of all the other floors of a building. Building data sets composed of model floor plans can easily improve the variety of data, since each model floor plan refers to (and is representative of) a different building.

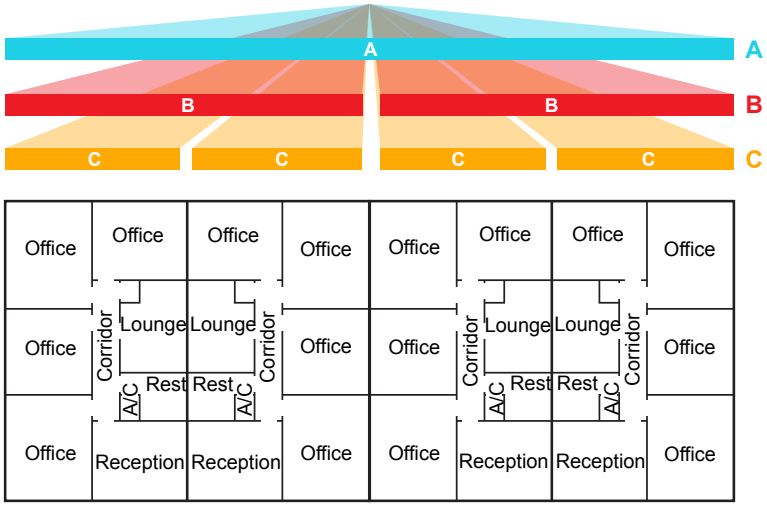


Figure 4.2: *An example of a highly-symmetrical office building. The pattern 'C' is an example of* model floor plan *that is repeated twice in each 'B'. In turn, 'B' is repeated in 'A', the entire floor plan.*

4.2.3 Building type data sets analysis

In this section we want to assess the significance of the concept of building type as context for developing general semantic mapping methods. In order to do so we evaluate if a clear difference emerge while considering rooms belonging to different building type. This is done with the analysis of five data sets containing approximatively $38,000$ rooms.

The first three data sets are collected as explained in Section 4.2.1 and represent rooms belonging to a specific building type and using the concept of model floor plan (all rooms within a data sets come from building of the same type and come from different model floor plans). The fifth and much larger data set is obtained from [101] and represents the entire MIT campus, without any reference to different possible function of buildings (e.g., dorms or research labs) and without considering the concept of model floor plan. The MIT data set does not use the concept of modern floor plan since it is composed by repeated entries, as (a) it contains different floors of the same building which have an almost identical layout and (b) it contains floors of different buildings that have been designed in a similar way, by the same architects, and at the same time. In this sense, we can say that the MIT data set includes the MIT model floor plans repeated several times.

More precisely the five data sets are:

- H, S, O data sets represent the House, School, and Office building

types respectively[1]. These data sets are composed by 1300, 1400, and 2600 rooms, respectively. These data sets are composed of model floor plans covering about 160 buildings.

- H+S+O, the fourth data set, is the union of the first three data sets.
- MIT data set is composed of the $32,000$ rooms of all the MIT campus buildings. This data set represents a collection of several building types that can be find in an university campus, such as dorms, research facilities, offices, and teaching facilities. Moreover it includes the MIT model floor plans repeated several times. The number of buildings in it (~ 160) is the same as the H+S+O data set, but the MIT data set has $5\times$ the number of rooms.

These data sets are significantly different from those typically used for reasoning about the semantic of an environment, since they are not directly collected by robots and they are substantially larger. However, rooms are represented using features similar to those that can be extracted, as instance, from laser range scanners. In the H, S, and O data sets, every room is represented by its label $L \in \mathcal{L}_{general}$, a set of geometric features A, d, rt (area, number of doorways, and the axes ratio $rt = M/m$ of the major axis M and minor axis m of the room bounding box), and the list of the labels of the rooms connected to it (for each one of the d doorways, the label of the connected room). Labels have been chosen to cover rooms of different types (see Section 4.2.1) and are used accordingly to each type: in the S data set, a M (medium) room is a classroom, while, in the O data set, it is an office. In the MIT data set every room is represented by its label (from a set of about 90 labels, e.g., SLPBTH is a dorm room with a private bathroom) and by the previously described geometric features A, d, rt.

Table 4.2 presents data regarding the geometric features A, d, rt of rooms in the data sets. The structure of the rooms is different for different building types (e.g., the number of doors d in corridors varies between 4.0 and 6.7, and the area of the M rooms is more than twice in schools S, 54.8 m^2, than in offices O, 18.5 m^2). Unsurprisingly, the standard deviation increases in H+S+O. The difference between considering data sets H, S, and O, composed of model floor plans, and a data set that is an instance of a building type is evident by looking at the MIT data set. Despite having $5\times$ rooms and $20\times$ labels with respect to H, S, and O data sets combined, the MIT data set is much more uniform than H, S, and O data set, as clearly evident by looking at the standard deviation of d and rt. This is a con-

[1] http://home.deib.polimi.it/luperto/datasets/floorplans

sequence of the redundancy of the MIT data set, which replicates several times instances of the same model floor plans.

In order to discover if the concept of building type emerges from the data, we classify each room in H, S, O according to the label set $\mathcal{L}_{general}$. We consider four well-known classification algorithms: rule induction (RI), multi-layer perceptron (MLP), decision tree (DT), and k-nearest neighbor (K-NN) [11], in order to compare their performance. These algorithms have been chosen because they are general and not tailored for the task of classifying places. In this way, we do not introduce any bias in the evaluation. All tests are performed using 10-fold cross-validation on the data set.

With this classification test, we want to discover if we obtain better performance on semantic classification of rooms by considering each room's building type or by considering a large knowledge base composed of several building types together. A total of 16 classifier are trained, four for every algorithm using as training set H, S, O and H+S+O. We refer to the classifiers trained using the building type data sets as 'type classifier' and to the classifiers trained using H+S+O as 'standard classifiers'.

Figure 6.2 shows the accuracy of classification of the 12 type classifiers compared with the classification accuracy of the 4 standard classifiers and Table 4.3 shows the confusion matrices of each building type. The average (over algorithms) accuracy of the type classifiers is 87.8% (4.9 is the standard deviation) for building type House, $88.3\%\,(4.8)$ for building type School, and $81.6\%\,(4.6)$ for building type Office. The lower performance on the Office building type with respect to classifiers trained and tested on the other two building typologies may be due to a further division that can be made internally in this category, namely between small offices, located in mixed office/residential buildings, and large open spaces. By contrast, standard classifiers have a lower average accuracy of $73.6\%\,(4.7)$. These results seem to suggest the effectiveness of using building types in semantic mapping. The performance of the standard classifiers is rather good because they exploit the several rooms labeled as S and C, whose features (large number of doors and small area) are similar in all the building typologies. For all building typologies, the results of the algorithms are similar, with the RI algorithm performing slightly better than the others over all the typologies.

To further assess the utility of the idea of building types, we trained a classifier on a data set and tested it on another data set using different building types, S and H. We call TRAIN(H) (TRAIN(S)) a classifier trained on H (S) and TEST(H) (TEST(S)) a classifier tested on H (S). As shown in

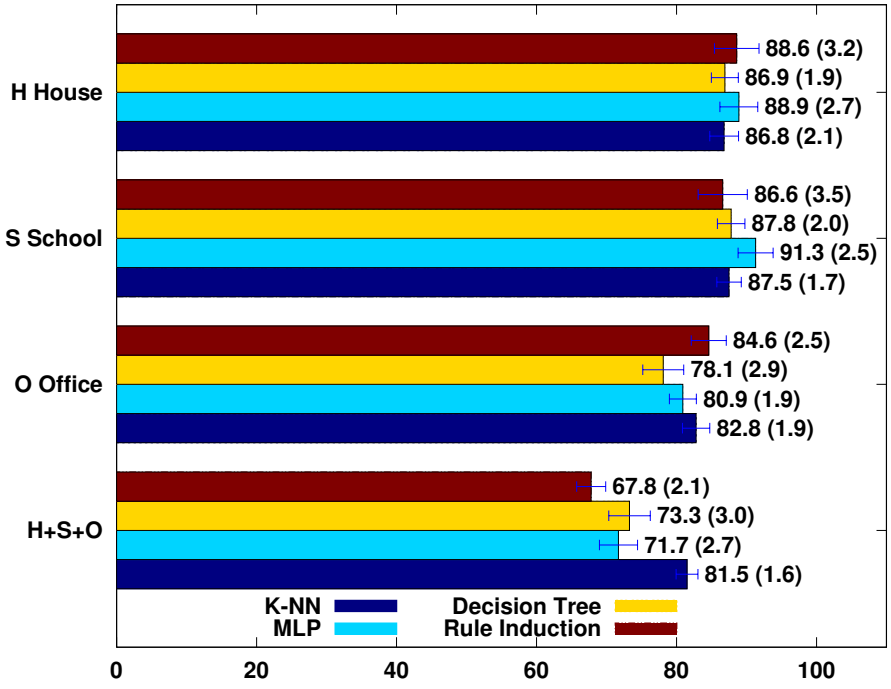


Figure 4.3: *Accuracy of classification (average and standard deviation) of the type and standard classifiers.*

the Table 4.4, the classification accuracy is reduced consistently by mixing the building types in training and testing. This implies that the models embedded in the types classifiers are actually specific to the corresponding building type. Classic semantic mapping approaches, not using building typologies, are implicitly developed for a unique type (e.g., university campus). Their use for a different type (e.g., houses) may result in a deterioration of their performance that in our results could reach 50%.

4.2.4 Model floor plan analysis

In order to evaluate the impact of the combined use of the model floor plan and building type, we now consider the characteristics of rooms in the MIT data set agains those of H, S, and O. Unfortunately, a direct application on the MIT data set of the four classifiers used in the previous section to evaluate H, S, O is not possible, because some aspects of MIT. Indeed, MIT contains 90 labels, and throughout the entire data set different symbols are used to describe a similar concept (e.g., 'kitchen' and 'kitchenette') and the same label is used to indicate a different concept (e.g., a bathroom in a dorm, with a shower, is very different from a bathroom in a research lab which usually contains only a toilet and a sink); these differences make it unfeasible to define a mapping from the labeling schema of MIT to $\mathcal{L}_{general}$.

For this reason, we developed a different unsupervised test which is independent from the set of labels. First, we remove all the labels from the five data sets described in Section 4.2.3, then, we use use a cluster algorithm (on the same feature vectors representing rooms described in the previous sections) in order to find out which rooms are considered similar and grouped together without any bias given by their semantic label. Due to the nature of the data (multi-dimensionality, many outliers, and complex cluster shapes), we use density-based spatial clustering, or DBSCAN [43], as clustering algorithm. We use parameters $\mathit{MinPts} = 6$ and $\epsilon = 3$; ϵ is selected using a K-distance plot. To display the clustering results, PCA (Principal Component Analysis) is then performed for displaying data in a 2D plot. Cluster evaluation is performed both visually and using *homogeneity* and *completeness* [83] as metrics. Homogeneity is a real-valued measure (belonging to $[0, 1]$) of how clusters present a similar distribution of semantic labels. It is 1 when each cluster contains only elements with a single label. Completeness is a real-valued measure (belonging to $[0, 1]$) of how much the rooms with an identical label are similar between them. It is 1 when all elements with a given label are assigned to the same cluster.

Figure 4.4(a)-(e) shows the result of DBSCAN clustering on the data sets. A building type data set presents, approximatively, between 20 and 30 clusters (Figure 4.4(a)-(c)), while merging together the three building types data set increases the number of clusters (Figure 4.4(d)). This indicates that rooms of a building type are similar with each other, but different of rooms of other building types. The number of clusters for the MIT data set (Figure 4.4(e)) confirms this finding. An interesting observation can be done on the number of rooms considered as outliers, represented as black dots in Figure 4.4(a)-(e). In the building type data sets, approximatively 17% of the rooms are considered as outliers. In the MIT data set only the 0.2% of rooms are outliers. However, the distribution of the labels in the outliers is the same: 75% of the outliers are C (or CORR for the MIT data set). Moreover, the MIT data set (despite having 90 labels) has a higher completeness (which, as said, measures intra-rooms similarity) than the building type data sets (Figure 4.4(f)). The difference in completeness and in the number of outliers is again a consequence of the fact that the building type data sets are composed of different model floor plans, thus providing more variety than the MIT data set.

4.3 Generalizing with respect to labels

As seen in Section 4.1, little attention has been given to define the sets of labels used in semantic mapping systems. Generally, these systems focus more on *how* to label the environments and not much less on the meaning of these labels. This can be a limit when generalizing beyond single-building contexts. Semantic mapping approaches often use from 2 to 20 labels (see Table 4.1). Increasing the amount of data, as in [7], increases the number of labels. The MIT and KTH campus floor plan data sets used in [7] have 90 and 160 semantic labels, respectively. However, despite both data sets derive from university campuses, there is not any definite mapping between their sets of labels. Intuitively, as often argued in papers, labels are intended to represent the name that human beings give to rooms: but is it always necessary, for a robot, to know the difference between a corridor and a vestibule?

We argue for more general semantic labels. While building types deal with the concept of function of a building, labels concern the function of rooms, rather independently of the buildings to which they belong. For example, a corridor is a sort of wildcard room, since it is present in all environments. However, the structure of a school hallway is very different from that of an house corridor. Since the function of the two rooms is

the same, namely to connect other rooms together, in a general semantic mapping system the label 'corridor' should apply in both cases, despite their major structural differences.

Our proposal is a hierarchical labeling approach structured as following:

- A general, yet minimal, set of labels that are common to all buildings, regardless of their type: ROOM and CORRIDOR, and
- a schema for specializing with more descriptive labels the concept of ROOM and CORRIDOR and that depends on the building type.

4.3.1 ROOMs and CORRIDORS

We analyse the data sets of Section 4.2.3 in order to verify whether the distinction between ROOMs and CORRIDORs is reflected in the data sets. We consider the H, S, and O data sets and we map C and H rooms onto CORRIDOR, while S, M, and B rooms are mapped onto ROOM. From the number of doors d, reported in Table 4.2, it can be seen that almost all ROOMs have one or two doors, while CORRIDORs have usually more than four doors. As an example, in the O data set the 98% of the office rooms (which are the 35% of the total number of rooms) are connected to at least one corridor and 92% of offices have only one door, confirming the intuition that there is a distinction between a connecting room like a corridor and a functional room like an office. When considering the two labels ROOM and CORRIDOR in the clusters found with DBSCAN for H, S, and O data sets (Figure 4.4), cluster homogeneity is, on average, 0.71 (which, as mentioned above, indicates good intra-cluster similarity). High cluster homogeneity implies that our top-level labels emerge as a characteristic of the data.

Clusters $\mathcal{C}$ found by DBSCAN (Figure 4.4) can also be used for semantic classification of rooms with the following naïve method. We consider a room R; we assign R to a cluster $C_i \in \mathcal{C}$ according to the geometrical features of R using DBSCAN; we then count the number of ROOMs and CORRIDORs in each cluster C_i and assign to the room R the most frequent label (either ROOM and CORRIDOR) in C_i. With this simple method the room classification accuracy using clusters for the H, S, and O data sets is, on average, 93.5%. This confirms once again that the two top-level labels ROOM and CORRIDOR can be accurately assigned to rooms only on the basis of simple geometric features of the rooms themselves.

Recognizing these two different types of rooms can be useful for a robot, for example for efficient exploration [24,92]. A generic framework capable of dividing environments in ROOMs and CORRIDORs can be a step towards semantic mapping generalization.

4.3.2 Hierarchical labeling schema

While the difference between corridors and rooms can be found easily even using a small set of features describing each room and across different building types, it is difficult to obtain a finer categorization of rooms (that can be applied trough different building types) without considering a higher amount of data. We propose a framework for specializing the concept of ROOM and CORRIDOR that is based on each room functionality and that can be applied to all building types.

At the top level, the labeling schema is called $\mathcal{L}_{R/C}$ and contains only two general categories (labels):

- ROOM: a space in which an activity is performed;
- CORRIDOR: a space used to connect other spaces together.

As we have seen in the previous section, this categorization of rooms can be easily performed only with a limited amount of data.

In the second set of labels, called $\mathcal{L}_{F/C/E/S}$ = {FUNCTIONAL ROOM, CONNECTION, ENTRANCE, SERVICE ROOM}, CORRIDORs are specialized in:

- CONNECTION: a space that connects together different spaces within the same floor, such as corridors or hallways;
- ENTRANCE: a space that connects a floor to other floors or to the exterior of the building, such as an elevator or a staircase;

and ROOMs are specialized in:

- FUNCTIONAL ROOM: a space in which the core activity of the building is performed (e.g., a classroom in a school, an office or a meeting room in an office building);
- SERVICE ROOM: a space used to support the core activities of the building (e.g., restrooms or kitchens).

This set of labels can be further specialized using specific labels that indicate the activity performed in each room. We refer to this schema as $\mathcal{L}_{type}$. $\mathcal{L}_{type}$ is composed by a list of all the possible kinds of room that can by found in a building type. This set of labels is chosen for each building type in order to be descriptive enough to represent all possible kinds of rooms and using building type manuals from architecture as source. An example of the complete label schema for the building type School ($\mathcal{L}_{school}$)

and Office ($\mathcal{L}_{office}$) is provided in Figure 4.5 and is derived from [95]. This hierarchical labeling schema is used in the following chapters of this dissertation. When we use data belonging to different building types altogether we use $\mathcal{L}_{R/C}$ and $\mathcal{L}_{F/C/E/S}$, while when we focus on a specific building type we use $\mathcal{L}_{type}$.

4.4 Discussion

In this chapter we have proposed the seed of a framework for making semantic mapping systems for indoor environments more general. We addressed two issues that are currently limiting the application of semantic mapping methods to contexts different from those in which they have been trained. The first issue is about the large degree of homogeneity shown by the buildings usually employed to train semantic mapping systems. Starting from the ideas of building type and model floor plan, we propose a way to develop more heterogeneous data sets that help avoiding the risk of overfitting and thus increase the generality of the semantic mapping systems. The second issue is about the lack of a principled way to define semantic labels for rooms. We propose to use a hierarchical labeling scheme in which the top level contains only two labels, ROOM and CORRIDOR, which can be "naturally" related to the geometric features describing the rooms, and the lower levels contain semantic labels specific for the single building types. The concept of building type, model floor plan and the hierarchical labeling schema are extensively used in the following chapters.

			Area a		Doorways d		rt=M/m	
H House	Label	%	μ	σ	μ	σ	μ	σ
	C	23.9	14.0	11.4	4.0	1.8	3.4	3.6
	H	8.0	61.5	25.1	3.3	1.5	1.8	0.8
	S	28.0	6.0	2.6	1.0	0.2	1.6	0.6
	M	26.3	17.4	4.8	1.3	0.4	2.0	1.0
	B	13.7	33.7	7.8	1.8	1.0	1.5	0.4

			Area a		Doorways d		rt=M/m	
S School	Label	%	μ	σ	μ	σ	μ	σ
	C	15.4	36.1	42.2	5.7	3.8	7.4	7.6
	H	5.6	205.1	172.9	6.9	4.5	2.0	1.3
	S	55.6	10.0	9.7	1.1	0.4	1.6	0.9
	M	21.3	54.8	14.8	1.6	0.8	1.3	0.4
	B	2.0	102.9	22.3	2.1	0.7	1.7	0.7

			Area a		Doorways d		rt=M/m	
O Office	Label	%	μ	σ	μ	σ	μ	σ
	C	20.6	30.5	25.7	6.7	4.8	8.1	7.3
	H	6.6	106.9	84.5	6.5	3.6	2.4	1.6
	S	23.2	4.5	2.3	1.0	0.1	1.3	0.7
	M	34.5	18.5	14.4	1.2	0.3	1.9	0.8
	B	14.8	31.3	12.2	1.4	0.9	1.9	1.0

			Area a		Doorways d		rt=M/m	
H+S+O	Label	%	μ	σ	μ	σ	μ	σ
	C	20.3	26.4	27.9	5.7	4.1	6.5	6.8
	H	6.8	110.9	109.2	3.7	3.7	2.2	1.3
	S	32.3	7.1	6.9	1.0	0.4	1.5	0.8
	M	29.3	24.5	18.9	1.3	0.6	1.8	0.8
	B	11.5	35.0	18.0	1.6	0.9	1.8	0.9

			Area a		Doorways d		rt=M/m	
MIT	Label	%	μ	σ	μ	σ	μ	σ
	SLPBTH	3.15	28.9	17.8	0.8	0.5	1.3	0.4
	BATH	3.9	10.4	7.6	1.0	0.4	1.2	0.2
	SLEEP	3.5	15.6	6.3	1.0	0.4	1.2	0.1
	OFF	27.0	14.3	10.8	1.1	0.6	1.1	0.5
	STAIR	5.2	13.9	12.8	1.0	0.6	1.33	0.4
	RL LAB	10.2	22.8	35.5	1.0	0.6	1.3	0.5
	CORR	17.3	48.7	125.4	6.0	5.9	1.4	0.6
	U/M	3.2	29.2	57.2	1.0	0.5	1.4	0.4

Table 4.2: *Characteristics of the rooms in the data sets, where % is the percentage of semantic labels (refer to text for explanations) present in the data set and μ and σ are the mean and the standard deviation of the corresponding feature, respectively. For clarity, for the* MIT *data set, only most recurrent labels (1000+ occurrences, which cover more than 70% of the rooms) are presented.*

H House	Label	C	H	S	M	B	TP	FP
	C	92.3	1.2	1.7	3.5	1.3	92.3	7.7
	H	3.0	88.0	0.0	0.7	8.2	88.0	12.0
	S	3.8	0.3	91.4	4.5	0.0	91.4	8.5
	M	3.7	0.4	8.1	83.4	4.3	83.4	16.6
	B	1.0	7.1	0.0	5.4	86.5	86.5	13.5
	TN	75.7	98.5	96.3	96.0	97.5		
	FN	24.2	1.5	3.7	3.9	2.5		

S School	Label	C	H	S	M	B	TP	FP
	C	92.0	3.3	4.2	0.3	0.1	92.0	8.0
	H	7.4	80.0	1.3	5.2	6.1	80.0	20.0
	S	3.1	0.3	94.0	2.6	0.0	94.0	6.0
	M	0.6	3.5	6.5	85.5	3.9	85.5	14.5
	B	0.0	21.7	2.9	18.8	56.5	56.5	43.5
	TN	97.4	98.2	42.4	97.4	98.8		
	FN	2.6	1.8	57.6	2.6	1.2		

O Office	Label	C	H	S	M	B	TP	FP
	C	91.9	3.6	0.2	2.6	1.7	91.9	8.1
	H	7.8	80.9	0.0	6.5	4.8	80.9	19.1
	S	2.0	0.0	92.3	5.4	0.3	92.3	7.7
	M	2.6	1.2	5.6	70.9	19.6	70.9	29.1
	B	3.6	5.9	0.1	25.3	65.1	65.1	34.8
	TN	79.3	98.1	96.8	92.4	90.1		
	FN	20.7	1.9	3.2	7.6	9.9		

H+S+O	Label	C	H	S	M	B	TP	FP
	C	85.7	6.9	2.1	2.8	2.4	85.7	14.2
	H	7.9	75.8	1.0	7.1	8.0	75.8	24.1
	S	4.0	0.2	75.4	17.6	2.7	75.4	24.6
	M	2.3	3.0	11.9	63.3	19.6	63.3	36.7
	B	4.6	9.3	3.6	15.8	66.7	66.7	33.3
	TN	96.4	96.9	64.7	87.9	91.1		
	FN	3.6	3.1	35.3	12.1	8.9		

Table 4.3: *Confusion matrices of the type and the standard classifiers (averaged over the algorithms), where all numbers are in percentage, rows are true labels, columns are predicted labels (*C *'corridor';* H *'hall';* S *'small room';* M *'medium room';* B *'big room'), TP are true positives, FP false positives, TN true negatives, and FN false negatives.*

Algorithm	TRAIN(H)/TEST(S)	TRAIN(S)/TEST(H)
RI	40.0(3.2)	50.7(2.5)
MLP	35.7(1.1)	56.1(0.5)
DT	43.0(1.8)	53.0(2.2)
K-NN	41.2(0.4)	51.0(0.3)
average	40.0(3.25)	52.7(2.75)

Table 4.4: *Classification results obtained by mixing building types for training and testing.*

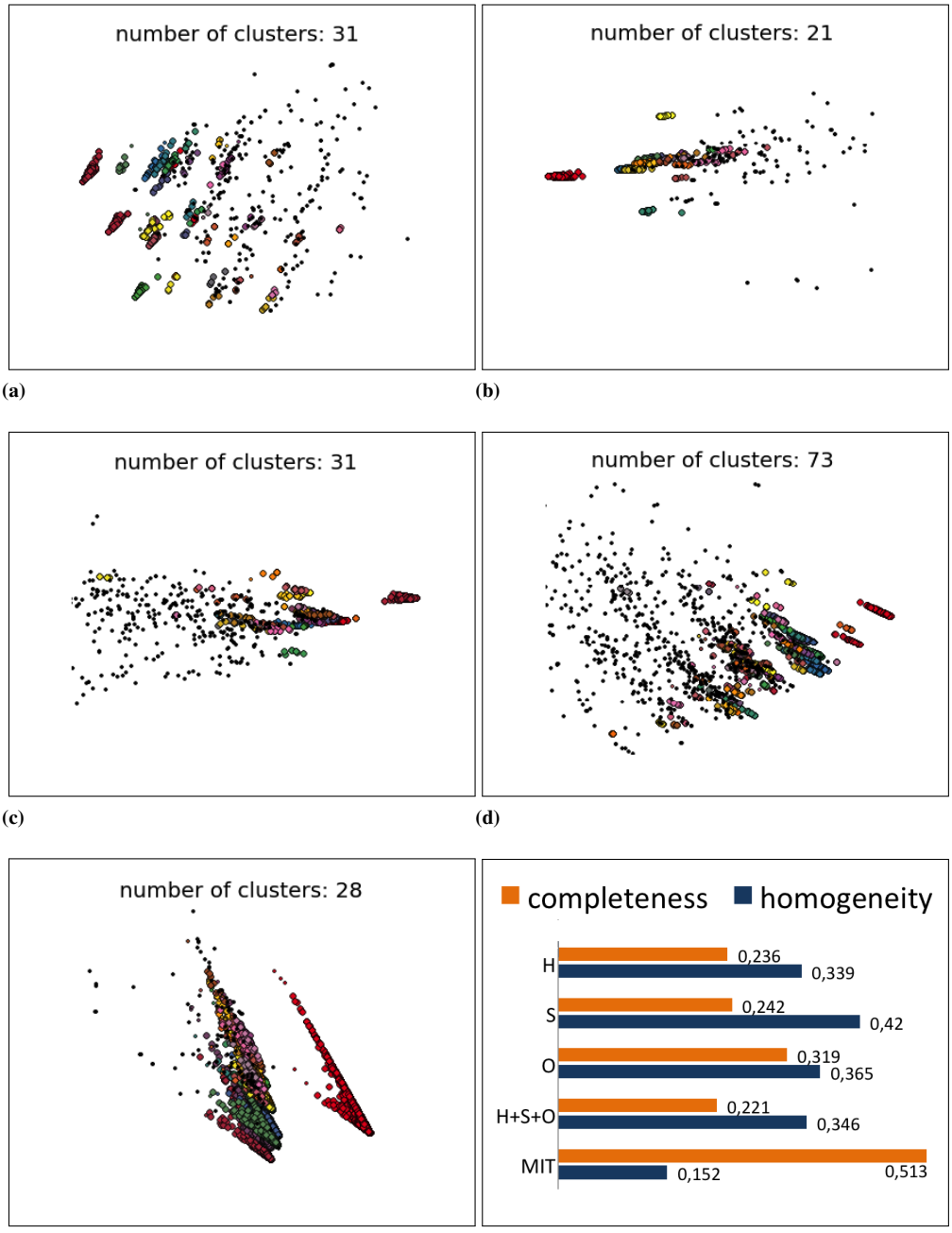


Figure 4.4: *DBSCAN clustering for the* H *(a),* S *(b),* O *(c),* H+S+O *(d), and* MIT *(e) data sets. Each cluster has a different color. Black dots are rooms not belonging to any cluster, namely outliers. Completeness and homogeneity of clusters for all data sets are presented in (f).*

ROOMS	FUNCTIONAL	classroom	support	teachers' room	laboratory	
	SERVICE	small admin. room	medium admin. room	big admin. room	gym	kitchen
		closet	conference room	medium service room	big service room	library
		bathroom	washroom	cafeteria canteen		
CORRIDORS	CONNECTION	corridor	lobby	hall		
	ENTRANCE	entrance	elevator	stairs		

(a) SCHOOL.

ROOMS	FUNCTIONAL	cubicule	conference room	executive office	openspace	conference hall
		office	shared office			
	SERVICE	small admin. room	medium admin. room	big admin. room	reception	kitchen
		closet	convenience store	medium service room	big service room	collective service room
		bathroom	washroom	cafeteria		
CORRIDORS	CONNECTION	corridor	lobby	hall		
	ENTRANCE	entrance	elevator	stairs		

(b) OFFICE.

Figure 4.5: *Labeling schema for* SCHOOL *and* OFFICE *building types.*

CHAPTER 5

Buildings as graphs

In Chapter 4 we have introduced the concept of building type for reasoning on specific classes of indoor environments. More precisely, we underlined how the *structure* of a building is strictly related to its function via the concept of building type. In this chapter we analyse how the concept of the structure of an indoor environment can be represented.

For this purpose we represent the semantic map obtained from floor plans of building as labeled topological maps. We extract the topological map using the method from Chapter 3. Labels are manually added using the labeling schema introduced in Section 4.3. This form of representation will be used in the following chapters of this thesis.

The outline of this chapter is the following: at first we introduce our graph representation and a set of metrics for characterize the building structure. Then we focus and discuss two building types, namely School S and Office O. We conclude the chapter by illustrating how the building structure permits to semantically classifying rooms into the basic label set introduce in Chapter 4 of ROOM or CORRIDOR using the reconstructed layouts of Chapter 3 as input.

5.1 Buildings as labeled graphs

Semantic maps are typically used in robotics for tasks requiring requires knowledge about the connectivity of rooms (i.e., what are the rooms connected to each other) and about the structure of the environment (i.e., how it is possible to go from room to another, given the fact that these rooms could be in completely different parts of the building). For these tasks it is particularly useful to represent the environment as a graph in the form of a labeled topological map, as explained in Chapter 2. Graphs are an abstract and compact form of representation of the environment, which is useful for reasoning, and can be used for representing effectively structured data [30].

Following these considerations, we represent (a floor of) an indoor environment as an undirected graph $G = (N, E)$, where each node $n \in N$ is a room and each edge $e = (n, n') \in E$ (with $n, n' \in N$) represents a physical connection between two rooms n and n' (e.g., a doorway). A semantic label $\mathcal{L}(n)$ taken from a finite set of labels $\mathcal{L}_{type}$ is assigned to each node n. Semantic labels are specific for the building type and follow the hierarchical schema explained in Chapter 4. An example of such a graph can be seen in Figure 5.1, where different colors are used to indicate different labels (see Figure 5.3c for a palette of the color used for indicating labels).

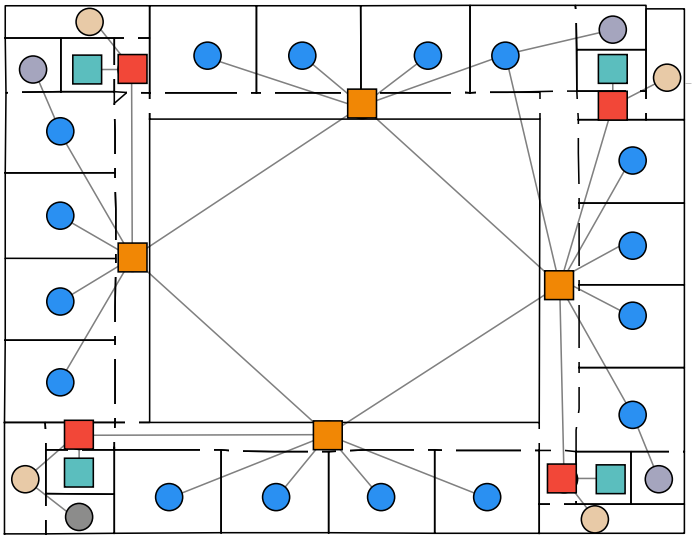


Figure 5.1: *A graph representing the topological structure of a school building superimposed to its floor plan. Different colors indicates different types of rooms.*

We use two datasets of floor plans of buildings created by hand from labelled floor plans of real buildings relative to two building types, SCHOOL (S) and OFFICE (O). These two building types are selected because they are examples of common, large scale, and structured buildings, and are highly codified by regulations and design guidelines (compared, as example, to residential buildings which usually present more variety of different possible structures). Specifically, we use from a graph database $\mathcal{G}$ built by hand from labeled floor plans of 50 schools ($\mathcal{G}_{\mathrm{S}}$) and 20 offices ($\mathcal{G}_{\mathrm{O}}$) and containing globally about 3500 rooms. Note that these data sets are different from those used in Chapter 4 for discussing about the concept of building types. More precisely we use the set of labels described in Section 4.3 and a superset of the features used in Chapter 4 for representing rooms (more precisely, an higher number of features is used for describing the room geometry). Note that data sets are collected manually since, although it is possible to use the method presented in Chapter 3 to automatically extracts the building layout, the tasks of labeling each room and of correcting possible errors in the reconstructed layout are difficult to be automatized.

We define a set of metrics representing structural properties of a graph. The set of metrics is presented in detail here and will be used in the following chapters.

Metrics can be divided in two groups. Those of the first group consider standard graph measures: the number of nodes (nodes), the average length of a shortest path between each pair of nodes in the graph (path-length), the maximum distance between two nodes along their shortest path (diameter), the number of articulation points (art-points; an articulation point is a node whose removal separates the graph into two distinct components), and the degree assortativity (assortativity) [70], which indicates whether nodes are connected to other nodes with a similar degree (1) or not (-1):

$$\text{assortativity} = \sum_{i,j} Dg_i Dg_j \frac{e_{Dg_i,Dg_j} - a_{Dg_i} a_{Dg_j}}{\sigma_a \sigma_b}$$

where Dg_i and Dg_j are the degree of nodes i and j, respectively, e_{Dg_i,Dg_j} is the fraction of edges between a node with degree Dg_i and a node with degree Dg_j, a_{Dg_i} is the fraction of the edges where at least one node has a degree of Dg_i (a_{Dg_j} is defined similarly), and σ_a and σ_b are the standard deviations of a_{Dg_i} and a_{Dg_j}, respectively.

The second group of metrics measure the *centrality* of each node in the graph. Centrality of a node indicates its *role* within the graph relatively to the connections between nodes. High centrality values indicate impor-

tant nodes, whose presence impact greatly in the graph layout, while low centrality values indicate peripheral nodes. Since centrality is a metric calculated for each node, we compute the average centrality μ and its corresponding standard deviation σ, by averaging over the number of nodes in a graph. Since in our domain CORRIDOR nodes are usually the most important (and thus should result in higher centrality values), while ROOM nodes are usually connected only to a corridor, the centrality measures are evaluated also separately for these two types of nodes. In the literature there exist several metrics for computing node's centrality. We select four centrality measures among the most used ones.

Betweenness centrality betw-cen for a node $n \in N$ of a graph $G = (N, E)$ is defined as the number of shortest paths between two nodes $u \neq t$ $(\neq n)$ that pass through n:

$$\text{betw-cen}(n) = \sum_{u,t \in N, u \neq n \neq t} \frac{\tau_{ut}(n)}{\tau_{ut}}$$

where τ_{ut} is the total number of shortest paths from u to t and $\tau_{ut}(n) \leq \tau_{ut}$ is the number of these paths that pass through n. Closeness centrality (closn-cen) for a node n is defined according to the shortest distance between n and all other nodes:

$$\text{closn-cen}(n) = \sum_{t \in N \setminus \{n\}} 2^{-d_G(n,t)}$$

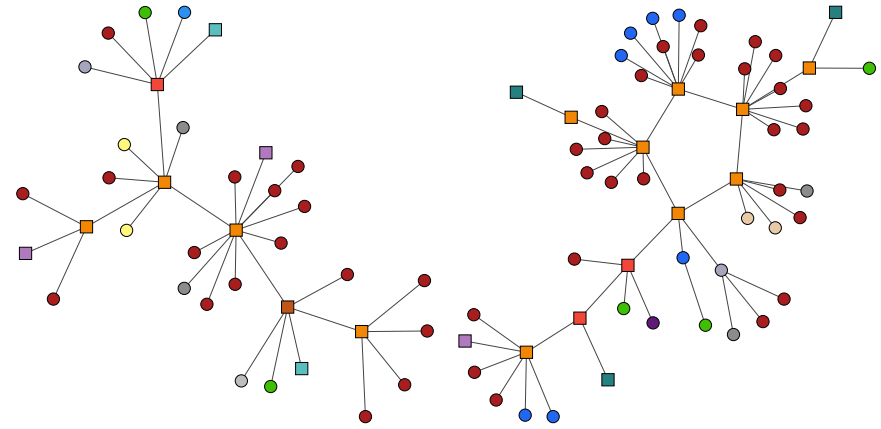
where $d_G(n, t)$ is the length of the shortest path on G between the nodes n and t. Eigenvector centrality (eig_cen) assigns a relative score to all nodes in the graph based on the concept that connections to high-scoring nodes contribute more to the score of a node than connections to low-scoring nodes. Is is calculated as:

$$\text{eig\_cen}(n) = \frac{1}{\lambda} \sum_{t \in N} m_{t,n}\, \text{eig\_cen}(k)$$

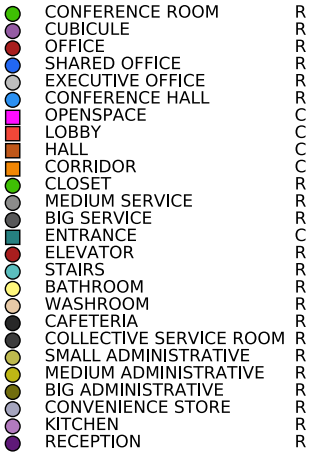
where $\lambda \neq 0$ is a constant and $m_{t,n}$ is the element (t, n) of the adjacency matrix M of the graph G. Katz centrality (katz) computes the centrality of a node n based on that of its adjacent nodes:

$$\text{katz}(n) = \alpha \sum_{n' \in N} m_{n',n} \text{katz}(n') + \beta$$

where α e β are constants and $m_{n',n}$ is the element corresponding to the (n', n)-entry in the adjacency matrix M representing G

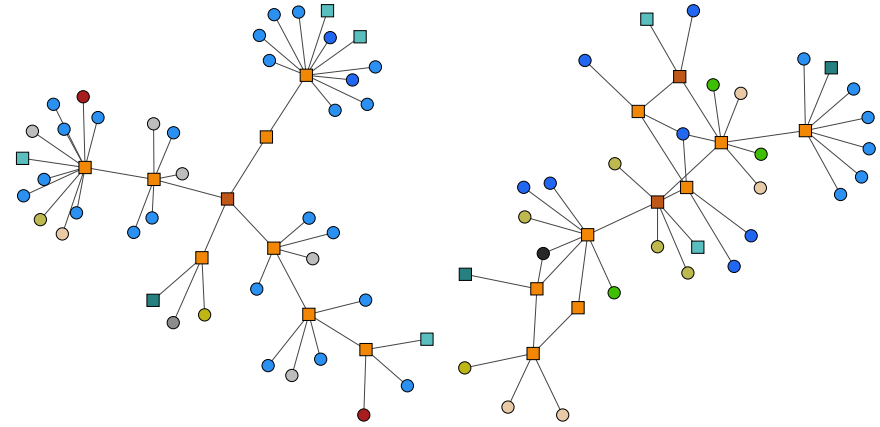


(a) (b)

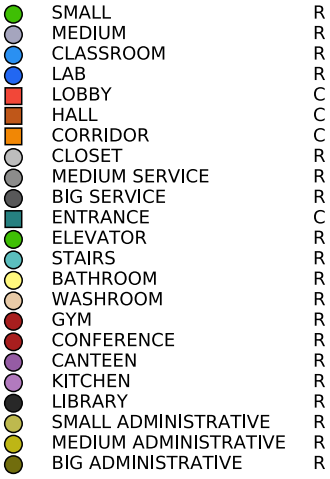


(c)

Figure 5.2: *An example of two graphs representing Office buildings and the palette of colors used for room labels. Note that* ROOMs *are indicated by small circles while* CORRIDORS *are indicated by squares.*



(a) (b)



(c)

Figure 5.3: *An example of two graphs representing School buildings and the palette of colors used for room labels. Note that* ROOMs *are indicated by small circles while* CORRIDORS *are indicated by squares.*

	$\mathcal{L}_S$	$\mathcal{L}_O$
nodes	35.24 (18.33)	45.15 (16.22)
nodes R	27.84 (15.05)	34.25 (12.94)
nodes C	7.42 (4.45)	10.90 (4.54)
path-length	3.33 (0.81)	3.51 (0.52)
diameter	6.34 (2.30)	6.9 (1.52)
art-points	8.8 (5.81)	10.65 (3.99)
assortativity	-0.51 (0.20)	-0.50 (0.13)
betw-cen	0.039 (0.011)	0.0302 (0.013)
betw-cen R	0.005 (0.006)	0.003 (0.003)
betw-cen C	0.181 (0.090)	0.117 (0.044)
closn-cen	0.328 (0.086)	0.245 (0.089)
closn-cen R	0.309 (0.077)	0.246 (0.083)
closn-cen C	0.418 (0.160)	0.311 (0.113)
eig-cen	0.256 (0.078)	0.219 (0.067)
eig-cen R	0.197 (0.062)	0.158 (0.046)
eig-cen C	0.498 (0.181)	0.401 (0.123)
katz-cen	0.181 (0.045)	0.152 (0.031)
katz-cen R	0.166 (0.045)	0.138 (0.032)
katz-cen C	0.241 (0.076)	0.198 (0.031)

Table 5.1: *Values of metrics for graphs representing* 50 *School buildings (*S*) and* 20 *Office buildings (*O*). Entries report average μ over the graphs and standard deviation σ (in parenthesis). R means* ROOM *and C means* CORRIDOR*.*

Table 5.1 shows that, globally, the metrics computed on S are similar to the ones computed on O. The average path-length is relatively small (on graph of 40 nodes, there is an average distance of about 3.5 nodes between two nodes in the graph). assortativity is computed considering only ROOMs and COORIDORs, and shows that graphs in $\mathcal{G}$ are highly disassortative: ROOMs are usually connected to CORRIDORs and not to nodes of the same category. Another interesting observation is that is the number of articulation points art-points which is relatively high considering the total number of nodes. This is because graphs G are planar-graphs (a graph is called planar if can be drawn without any edge intersection), with a relatively limited number of edges (compared, for example, to random graphs) and highly disassortative. Considering the centrality value, it can be seen how CORRIDORS are particularly important within the graph. If we consider the shape of our graphs, most of the buildings in our data sets can be represented as trees, where leaf node are ROOMs and non-leaf nodes are CORRIDORs. When graphs are more complex than trees, they can usually be transformed into trees by selectively removing one or two edges between two CORRIDORs. An example can be seen in Figure 5.2 and in Figure 5.3

with four examples of real world buildings. It can be easily seen that the role of corridors is particular important in all four buildings, and that their structure is similar to a tree.

5.2 Floor plan data sets

The data sets described in Section 5.1 are those that are used in the following of this dissertation. As we pointed out in the previous section, those data sets are different from the ones used in Chapter 4 for discussing the concept of building types. The data sets presented here have been obtained following the insights of Chapter 4 and Section 5.1.

More specifically, these data set (that we use in the following of this dissertation) are relative to two building types, SCHOOL (S) and OFFICE (O). The SCHOOL data set contains 50 buildings, while OFFICE data set contains 30 buildings; both data sets contain globally about 3500 rooms. (For some task, as in Section 8.1, we use only a subset of the data sets.) The two data sets follow the labeling schema of Section 4.3 and are obtained using the CAD-like tool presented in Section 4.2.1, starting from architecture books. The buildings whose floor plans are in the data sets have been selected using the concept of model floor plan (Section 4.2.2).

A vector of features V_n is associated to each room n, which describes its geometrical structure and some features representing the role of the room within the building structure. In our implementation, the vector V_n is composed of 8 geometrical features, and 3 features representing the structure of the building. Our geometrical features are a subset of those used in [64] and are calculated using the bounding polygon P that approximates each room:

$$P = \{v_0, v_1, ..., v_{N-1}, v_N \equiv v_0\}$$

where v_i is a vertex of the polygon having coordinates (x_i, y_i).

Area: the area of the polygon P, given by:

$$f_{area} = \frac{1}{2} \sum_{i=0}^{N-1} (x_i y_{i+1} - x_{i+1} y_i)$$

Perimeter: the perimeter of the polygon P, given by:

$$f_{perimeter} = \sum_{i=0}^{N-1} dist(v_i, v_{i+1})$$

where:

$$dist(v_i, v_{i+1}) = \sqrt{(x_i - x_{i+1})^2 + (y_i - y_{i+1})^2}$$

The ratio between area and perimeter: the area of P divided by its perimeter, defined as:

$$f_{AP} = \frac{f_{area}}{f_{perimeter}}$$

Mean distance between the centroid and the shape boundary:

$$f_{mean\_shape} = \frac{1}{N} \sum_{i=0}^{N-1} dist(v_i, c)$$

where

$$dist(v_i, c) = \sqrt{(x_i - c_x)^2 + (y_i - c_y)^2}$$

given that c_x and c_y are the coordinates of the centroid $c = (c_x, c_y)$ defined as:

$$c_x = \frac{1}{6f_{area}} \sum_{i=0}^{N-1} (x_i + x_{i+1})(x_i y_{i+1} - x_{i+1} y_i)$$

$$c_y = \frac{1}{6f_{area}} \sum_{i=0}^{N-1} (y_i + y_{i+1})(x_i y_{i+1} - x_{i+1} y_i)$$

Form factor: the form factor of polygon P, defined as:

$$f_{form\_factor} = \frac{4\pi f_{area}}{\sqrt{f_{perimeter}}}$$

Circularity: the circularity of polygon P, defined as:

$$f_{circularity} = \frac{f_{perimeter}^2}{f_{area}}$$

Normalized circularity: the normalized circularity of P, defined as:

$$f_{norm-circularity} = \frac{4\pi f_{area}}{f^2_{perimeter}}$$

Average normalized distance between the centroid and the shape boundary: given the previous definitions of centroid and distance from centroid, it is computed as:

$$f_{n\_dist\_shape} = \frac{1}{N}\sum_{i=0}^{N-1}\overline{dist}(v_i, c)$$

where:

$$\overline{dist}(v_i, c) = \frac{dist(v_i, c)}{\max\limits_{\forall i}(dist(v_i, c))}$$

In addition, three structural features are considered: the number of doors d and two centrality measures, closeness and betweenness centrality, introduced in the previous section. Note that this V_n, used in the rest of this thesis, contains a different set of features than those described in Chapter 4. This feature vector has been chose because it uses geometrical features from [64] that are broadly adopted in several state-of-the-art semantic mapping methods (e.g., [49, 77]) and because these geometrical features can be easily retrieved from our reconstructed layout of Chapter 3.

5.3 Classification of reconstructed layout

In this section we want to evaluate if it is possible to perform semantic mapping of a building using using only features relative to its structure. For this purpose, we implement a state-of-the-art ensemble classification method, Extremely Randomized Tress (Extra-Trees, ET) [42]. This application is intended to test the effectiveness of our layout reconstruction method (Chapter 3) as source of input for place categorization and to discuss the place categorization performances using some of the global features described in the previous section.

Extra-Trees algorithm builds an ensemble, or *forest*, of unpruned decision trees using a standard tree-based top-down classification procedures. Extra-Trees algorithm strongly randomizes on both attributes and cut-point choices while splitting a node of a tree. The two main differences with other tree-based ensemble methods are that (a) Extra-Trees algorithm splits nodes

by choosing cut-points fully at random and (b) it uses the whole learning sample (rather than a bootstrap replica) to grow the trees. This classification algorithm is known to be computationally efficient, accurate, and robust for different classifications tasks. The Extra-Trees algorithm has been selected since (a) it is a classification method that it is robust with respect to overfitting, (b) it performs well in heterogeneous classification tasks and (c) it provide good classification accuracy. In our setting, Extra-Trees provided a good balancing between classification accuracy and overfitting if compared with standard classification approaches such as multi-class SVMs, Decision Trees, k-NN or other ensemble methods as AdaBoost. Results are not reported here for the sake of brevity.

As training set for the classifier we use the 50 graphs of SCHOOLS analysed in the previous section. Room are represented by a the feature vector V_n as explained in Section 5.2. The classifier is then applied to reconstructed floor plan layouts of 20 other schools obtained using the method explained in Chapter 3. The set of features V_n describing each room are extracted from the geometrical approximation and from the topological map obtained from the reconstructed layout. Ground truth (the floor plan of the building manually classified with semantic labels) is used for evaluating the results. Classification is performed (separately) using the $\mathcal{L}_{R/C}$ and $\mathcal{L}_{F/C/E/S}$ labeling schema.

In order to evaluate match of rooms of the reconstructed layout to rooms of the ground truth layout, we rely on the concept of *forward coverage* FC and *backward coverage* BC introduced in Section 3.2. We call $r \in \mathcal{R}$ a room in the reconstructed layout and $r' \in Gt$ a room in the ground truth layout, we can define an indicator function that compares the labels of two rooms:

$$\mathbb{1}_{EQ}(r, r') = \begin{cases} 1, & \text{if room } r \text{ and room } r' \text{ have the same label} \\ 0, & \text{otherwise} \end{cases}$$

Using this definition, we can now compute two metrics for evaluating the accuracy of our classifier:

SemanticAccuracy$_{FC}$: measures how well the semantic map computed from the reconstructed layout and using ET matches Gt:

$$SemanticAccuracy_{FC} = \frac{\sum_{r \in \mathcal{R}} \mathbb{1}_{EQ}(r, FC(r))}{|\mathcal{R}|} \tag{5.1}$$

SemanticAccuracy$_{BC}$: measures how well the ground truth semantic map is described by the reconstructed layout semantic map.

$$SemanticAccuracy_{BC} = \frac{\sum_{r \in Gt} \mathbb{1}_{EQ}(r', BC(r'))}{|Gt|} \tag{5.2}$$

The two measures SemanticAccuracy$_{FC}$ and SemanticAccuracy$_{BC}$ are both necessaries since the reconstructed layout and the ground truth layout can be potentially different due to reconstruction inaccuracies as over- and under-segmentation. A room in the ground truth layout can be wrongly recognized as a set of different rooms in the reconstructed layout (oversegmentation) or a room in the reconstructed layout can be composed of several rooms in ground truth layout (undersegmentation). As an example, a long corridor in the ground truth layout can be (wrongly) recognized as a set of smaller corridors in the reconstructed layout; however if all these small corridors in the reconstructed layout are correctly classified, both SemanticAccuracy$_{FC}$ and SemanticAccuracy$_{BC}$ are unaffected by the segmentation error. Table 5.2 shows the results obtained using the two sets of labels ($\pm$ standard deviation) considering both SemanticAccuracy$_{FC}$ and SemanticAccuracy$_{BC}$. The structure of a building contains enough information to recognize whether a room is a ROOM or a CORRIDOR, while a more detailed classification results in a decrease of performances. In Table 5.3 and Table 5.4 it can be seen as, while Extra-Trees can distinguish with good accuracy between functional rooms F and corridors C using $\mathcal{L}_{F/C/E/S}$, a higher error rate is present when considering entrance and service rooms E and S. Using additional knowledge (e.g., data coming from laser scan or vision) in a multi-modal framework (as done in [77]) may increase the performance in distinguish between a higher number of classes of rooms, i.e. using a larger label set.

	SemanticAccuracy$_{FC}$	SemanticAccuracy$_{BC}$
$\mathcal{L}_{R/C}$	97.1% $\pm$ 2.1%	96.5% $\pm$ 2.5%
$\mathcal{L}_{F/C/E/S}$	76.1% $\pm$ 9.1%	77.5% $\pm$ 10.5%

Table 5.2: *Classification accuracy for School buildings of reconstructed layout ($\pm$ standard deviation) on a training data set of* 50 *schools and on a testing data set of* 20 *schools.*

	F	C	E	S
F	**82.3% ± 4.7%**	2.8% ± 3.8%	2.2% ± 2.9%	12.7% ± 3.3%
C	1.8% ± 3.1%	**94.4% ± 3.8%**	2.3% ± 2.5%	1.5% ± 3.2%
E	0.9% ± 2.8%	10.1% ± 15.5%	**72% ± 2.2%**	17% ± 14.5%
S	5.7% ± 8.1%	12% ± 10.2%	30.3% ± 18.1%	**52% ± 8.2%**

Table 5.3: *Confusion matrix relative to SemanticAccuracy$_{FC}$ with $\mathcal{L}_{F/C/E/S}$.*

	F	C	E	S
F	**96.6% ± 5.9%**	0.8% ± 1.5%	1.2% ± 1.3%	1.4% ± 1.7%
C	2.2% ± 4.1%	**90.6% ± 11.7%**	0.9% ± 1.3%	6.3% ± 8.7%
E	9.2% ± 11.5%	2.2% ± 3%	**64.3% ± 18.5%**	24.3% ± 17.4%
S	33.9% ± 22.3%	1.1% ± 1.2%	17.6% ± 12.5%	**47.4% ± 10.4%**

Table 5.4: *Confusion matrix relative to SemanticAccuracy$_{BC}$ with $\mathcal{L}_{F/C/E/S}$.*

5.4 Discussion

In this chapter we have presented a description of the structure of indoor buildings which involves the use of a graph (topological map) and metrics about the importance of each room within the entire environment (using centrality measures). Rooms obtained using the layout reconstruction method of Chapter 3 are successfully classified using these information. The representation of the structure of environment and the centrality measures described in the first and second section of this chapter are used in the rest of this dissertation.

CHAPTER 6

Reasoning on the building structure

In Chapter 5 we have discussed how the concept of *building structure* can be represented with a graph and used for semantic mapping. In this chapter we propose a method for reasoning on the structure of buildings by representing them as a set of logical relations (as example by considering the relation 'connect(r,r')' that holds true if there exists a door between two rooms r and r'). *Relations* are an expressive formalism that allows us to represent multiple associations between one or more objects overcoming the limitations of a propositional representation [30]. As we extensively discussed in Chapter 2, mainstream approaches for place classification are characterized by *local* reasoning, in which the input of the classifiers is a vector of features that refer to the space being classified or to its neighbourhood, following an attribute-value propositional schema. These approaches have proven effective in classifying single rooms, but they do not naturally apply to the classification of entire floors of buildings (i.e., assigning them labels like 'office' or 'school') since it is difficult to represent the structure of a building floor as a vector of features, being it better captured by a graph.

In order to address these limitations, in this thesis we propose an approach to semantic classification that reasons on the whole structure of

a (floor of a) building as a single entity by exploiting a relational representation of the building structure. Reasoning on such structured data is performed by exploiting *Statistical Relational Learning* (SRL) techniques. In recent years, the field of SRL has developed different techniques (like Markov Logic Networks [81]) for dealing and reasoning with complex multirelational data in different settings [41]. The use of these techniques for semantic classification can enable a form of *global* reasoning on the structure of buildings, by considering their recurrent patterns. In particular, we originally apply to the context of semantic classification a state-of-the-art SRL algorithm, *kLog* [34], which is a method based on graph kernels for logical Entity/Relational structured data. Using kLog, we are able to effectively perform classification tasks similar to those performed by classical local methods, like labeling rooms, and also classification tasks that are not naturally addressed by local methods, because they require reasoning on structured knowledge, such as classifying an entire building according to its function. In order to evaluate (also quantitatively) our approach, we compare results obtained with it against those obtained with a standard attribute-value classification method, namely Extremely Randomized Trees (Extra-Trees) [42].

Note that, for the sake of simplicity, we assume to consider only one floor of multi-stores buildings. In this sense, we do not consider explicitly possible relations between different floors of the same building. However, this limitation can be overcome by adding other floors of the building to the training set of the classifiers. If another floor of the same building on which the algorithm is being tested is used during the training phase of the classifiers, the similarity of the two structures is used automatically for classification since the two floors are likely to be highly correlated.

The use of SRL techniques in semantic mapping has been pioneered by [56]. This approach shares some similarities with that presented in this chapter, but in our case SRL techniques are used for classification purposes and not for evaluating a scoring function of a model. The content of this chapter is based on the work of [60].

6.1 A global reasoning approach

6.1.1 Problem formulation

In this chapter we use the graph representation of indoor environments that we have introduced in Section 5.1. More precisely, we represent (a floor of) an indoor environment as an undirected graph $G = (N, E)$, where each

node $n \in N$ is a room and each edge $e = (n, n') \in E$ (with $n, n' \in N$) represents a physical connection between two rooms n and n' (e.g., a doorway). A semantic label $\mathcal{L}(n)$ taken from a finite set of labels $\mathcal{L}$ is assigned to each node n. Each room n is also associated to a vector of features V_n which describe its geometrical structure. A list and description of the feature set can be found in Section 5.2.

We use two data sets representing two different building types. Dataset SCHOOL (S) contains 30 floor plans of school buildings, while data set OFFICE (O) contains 20 floor plans of office buildings. See Sections 5.1-5.2 for an analysis of the two data sets.

The semantic classification tasks we address in this thesis are defined as follows. The first task is place classification. Given a dataset $\mathcal{G}$ of labeled graphs representing buildings of a single building type, a function $f()$ that scores similarity between labeled graphs, and a query graph $\hat{G}$ whose nodes have associated vectors of features but not labels, the *place classification* task consists in finding the labels for the nodes in $\hat{G}$ such that $f(\hat{G}, \mathcal{G})$ is maximized.

Loosely speaking, local methods inductively build a model of the relation between the vectors of features V_n of nodes of graphs in $\mathcal{G}$ and the corresponding labels $\mathcal{L}(n)$ and, on the basis of this model, assigns a label $\mathcal{L}(\hat{n})$ to each node $\hat{n}$ of $\hat{G}$. Our global method, instead, inductively build a model of the labeled graphs in $\mathcal{G}$ and, on the basis of this model, assigns a label $\mathcal{L}(\hat{n})$ to each node $\hat{n}$ of $\hat{G}$.

The second task that we introduce in this thesis is building classification. Assume to have a dataset $\mathcal{G}$ of graphs whose nodes are labeled as before and such that each graph $G \in \mathcal{G}$ has also a graph label L_G. Given such $\mathcal{G}$, a function $f()$ as before, and a query graph $\hat{G}$ whose nodes are labeled but whose graph label $L_{\hat{G}}$ is unknown, the *building classification* task consists in finding $L_{\hat{G}}$ such that $f(\hat{G}, \mathcal{G})$ is maximized. It is intuitive that building classification tasks involves recognizing recurrent patterns in graphs, which can be more naturally performed following a global approach.

We use two hierarchically organized semantic labeling schemas to evaluate the impact of different sets of labels on semantic classification, following the discussion of Section 4.3. At the top level, we use the labeling schema $\mathcal{L}_{R/C}$ and contains only two general categories, ROOM and CORRIDOR. The second set of labels, $\mathcal{L}_{F/C/E/S}$, specialize $\mathcal{L}_{R/C}$:

$$\mathcal{L}_{F/C/E/S} = \{\text{FUNCTIONAL ROOM}, \text{CONNECTION}, \text{ENTRANCE}, \text{SERVICE ROOM}\}$$

6.1.2 kLog

We use kLog [34] for performing semantic classification tasks by exploiting reasoning on the global structure of buildings. kLog input data are multiple relations between objects (possibly with attributes) represented in the form of an Entity/Relation knowledge base (E/R KB). This KB is expressed with a Prolog syntax and is composed of ground atoms under the closed world assumption. Entities and relations between them are expressed using *signatures* and roughly correspond to ground atoms listed explicitly (extensional signatures) and to ground atoms implicitly defined using Prolog definite clauses (intensional signatures), respectively. Listing 6.1 shows a subset of the signatures we use, whose meaning is intuitive. Learning and classification tasks and queries are also expressed in Prolog syntax in a declarative way.

```
signature buildingtype(buildingtype::property)::extensional.
signature space(space_id::self)::extensional.
signature connected(s1::space, s2::space)::extensional.
signature label(space_id::space, label::property)::extensional.
signature area(space_id::space, area::property)::extensional.
signature perimeter(space_id::space, perimeter::property)::extensional.
signature iscorr(room_id::room)::intensional.
```

Listing 6.1: *Examples of kLog signatures.*

The label associated to a space is selected from one of the two sets of labels ($\mathcal{L}_{R/C}$ or $\mathcal{L}_{F/C/E/S}$), according to the task. Classification is performed by learning the $label(\_,\_)$ relation or by learning the $buildingtype(\_)$ relation, according to the task. The former relation associates a label to a room object, while the latter relation associates a label (the building type) to the query graph (see Listing 6.1).

```
interpretation(floor01,buildingtype(school)).
interpretation(floor01,space(s001)).
interpretation(floor01,label(s001,classroom)).
interpretation(floor01,space(s002)).
interpretation(floor01,label(s002,corridor)).
interpretation(floor01,space(s003)).
interpretation(floor01,label(s003,classroom)).
interpretation(floor01,connected(s001,s002)).
interpretation(floor01,connected(s002,s003)).
interpretation(floor01,area(s001,100)).
interpretation(floor01,perimeter(s001,40)).
...
```

Listing 6.2: *Fragment of an interpretation.*

kLog learns from interpretations. In our system, a set of instantiated signatures representing a floor of a building is called an *interpretation*. A fragment of an interpretation (called *floor01*) representing a floor of a school

building composed of two classrooms connected by a corridor is reported in Listing 6.2.

Learning tasks, like binary or multi-class classifications, are performed in kLog using a Support Vector Machine (SVM) in a kernel space defined using graph kernels. *Graph kernels* allow to operate in representative high-dimensional feature spaces without suffering the high cost of computing the feature spaces explicitly. Graph kernels used in kLog belong to the family of R-convolution kernels [40], based on decomposing structured data in smaller parts. Given a decomposition relation R^{-1} that extracts from a graph G one of its possible decompositions in subgraphs $\{g\}$, the associated R-convolution kernel K compares all subgraphs of two graphs G and G', while a sub-kernel k_t compares all the features $t \in \{1, 2, \ldots, T\}$ extracted from two subgraphs g and g':

$$K(G, G') = \sum_{g \in R^{-1}(G),\, g' \in R^{-1}(G')} \sum_{t=1}^{T} k_t(g, g')$$

Specifically, in kLog, the Neighborhood Subgraph Pairwise Distance Kernel (NSPDK) is used. NSPDK is extensively described in [28, 34]; here we just introduce the idea. The relation R^{-1} used by NSPDK extracts from a graph G all the pairs of neighborhood subgraphs g_i and g_j of radius r and centered on all possible pairs of nodes n_i and n_j that are at a distance d in the graph (the same happens in G'). The sub-kernel k_t in this case depends on r and d, let call it $k_{r,d}$, and computes the similarity of all these pairs of subgraphs:

$$K(G, G') = \sum_{r} \sum_{d} \sum_{\substack{g_i, g_j \in R^{-1}(G) \\ g'_i, g'_j \in R^{-1}(G')}} k_{r,d}((g_i, g_j), (g'_i, g'_j))$$

For example, in an office building, the sub-kernel $k_{r,d}$ will assign large scores to symmetric portions of the building, since the r-neighborhoods of two symmetric nodes will be similar. In this sense, the use of graph kernels allow us to explicit capture the similarities between groups of nodes across a graph and between different graphs.

In kLog, the graph on which the graph kernel is applied is obtained from the E/R KB using a technique called *graphicalization*. Each entry found in the KB is converted into a node of a bipartite undirected graph whose nodes are ground atoms and edges connect an entity atom to a relationship atom if the identifier of the former appears as an argument in the latter. The graphicalization of the example of Listing 6.2 is shown in Figure6.1. Note

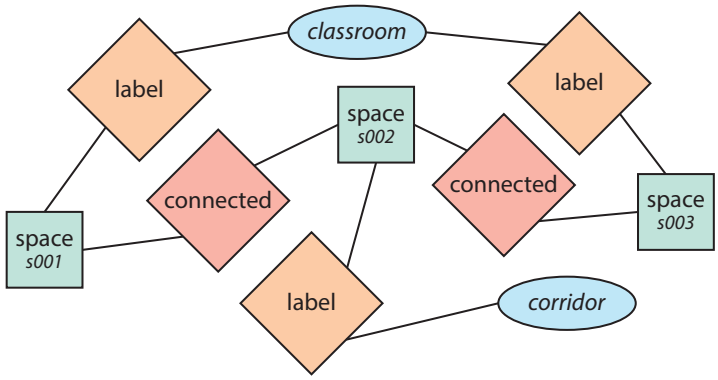


Figure 6.1: *Graphicalization of the Listing 6.2.*

that the generated graph embeds information about space connections (as the original graphs $\mathcal{G}$ of our dataset) plus other information (for example about labels: two spaces with the same label are now connected together via a label node). NSPDK is then applied to the graphicalized KB and the similarity results it produces are employed by a standard SVM algorithm from the LibSVM repository[1]. Please refer to [34] for further details.

Classic approaches to semantic mapping, as [77], usually represent the structure of the building using probabilistic graphical models like Bayesian Networks, Markov Random Fields (MRF) or Chain Graphs (which are a natural generalization of directed and undirected graphical models). In [77] a semantic map is represented using a Chain Graph using the following schema: rooms are represented as nodes in a Markov Random Field, doors between rooms are represented as connections (potentials) between two nodes, and features describing each room are represented as a Bayesian Network. Using kLog we are able to intuitively represent relations between different (and potentially not connected) parts of the structure of the building (as an example, that there exists two similar blocks of rooms into two different wings of the same building), and we are able to easily add or remove relations between objects without changing the learning task. Moreover, the conditional independence assumption of Markov Random Fields (and, consequently, of Chain Graphs) constitutes a structural limitation of such methods for representing relations between different rooms. As an example, using a Chain Graph like that of [77], two rooms that are connected to the same corridor are considered as independent given the fact that we know the exact label of the corridor. An explicit representation of all the

[1] https://www.csie.ntu.edu.tw/c̃jlin/libsvm/

possible dependencies between nodes using a MRF requires a much more dense and complex representation. For this reason, while Chain Graphs and MRFs are a powerful form of representation for representing the semantic map of an indoor environment, their generalization for representing the entire structure of a floor of a building as a single entity requires a further and complex analysis. For this reason, we choose to use a different formalization of our learning problem using kLog.

6.1.3 Extremely Randomized Trees

In order to compare the reasoning on the whole structure of buildings with a local approach, we implement a state-of-the-art ensemble classification method, Extremely Randomized Tress (Extra-Trees) [42], that we have introduced in Section 5.3.

Extra-Trees require an attribute-value representation of each room. Each room n is thus represented by its features V_n (introduced in Section 5.2), to which its label $\mathcal{L}(n)$ is associated. We also tried to add to V_n other information using the labels of the rooms that are directly connected to n, but experimental results are similar. The classification task is performed similarly as explained in Section 5.3, with the only difference that for this application we use only one data set of floor plans of buildings, while in Section 5.3 two data set are used (for ground truth and reconstructed layout respectively).

Extra-Trees have been selected since, as explained in Section 5.3, provided a good balancing between classification accuracy, overfitting and robustness to different classification tasks if compared with standard classification approaches such as multi-class SVMs, Decision Trees, Bayesian Networks, k-NN, or other ensemble methods as AdaBoost that have been used for place classification. More precisely, Extra-Trees are able to provide good accuracy in different classification tasks on the same domain with a little adaptation of the training process of the classifier.

In order to test a mixed situation, in which global information is considered in a local setting, we use a second version of the Extra-Trees algorithm in which two more features that describe the role of a room n in the graph representing the building are added to V_n. These two features are related to *centrality*, which evaluates the role of each node in the graph as introduced in Section 5.1: high values of centrality correspond to most important nodes in the structure of the graph. In our context, centrality can be considered as a measure of the importance of each room within its floor structure. We use closeness and betweenness centrality (see Section 5 for their defini-

tion). Classification tasks using Extra-Trees algorithm are evaluated both considering and not considering centrality among the features V_n of a room.

6.2 Experimental evaluation

	task	dataset	labels	centrality	error (%)
Ex-T	●	S	RC	○	3.74
Ex-T	●	S	RC	●	**2.44**
kLog	●	S	RC		6.82
Ex-T	●	S	$FCES$	○	35.56
Ex-T	●	S	$FCES$	●	**33.42**
kLog	●	S	$FCES$		46.89
Ex-T	●	O	RC	○	7.21
Ex-T	●	O	RC	●	**5.12**
kLog	●	O	RC		7.73
Ex-T	●	O	$FCES$	○	36.15
Ex-T	●	O	$FCES$	●	**32.80**
kLog	●	O	$FCES$		44.92
Ex-T	■	S + O	RC	○	40.00
Ex-T	■	S + O	RC	●	30.00
kLog	■	S + O	RC		**8.00**
Ex-T	■	S + O	$FCES$	○	**5.00**
Ex-T	■	S + O	$FCES$	●	**5.00**
kLog	■	S + O	$FCES$		8.00

Table 6.1: *Results on different learning tasks using kLog and Extra-Trees (Ex-T). Cyan circles stand for place classification tasks, while blue squares indicate building classification.* S *represents the* SCHOOL *dataset, and* O *is the* OFFICE *data set. The labels column represents the label set used, the centrality column indicates whether centrality is added (full circle) or not (empty circle) to V_n, while the error column indicates the classification error with the best performance in bold.*

In this section, we experimentally evaluate our global approach to semantic classification (based on kLog) and compare it against a classical local approach (Extra-Trees) on different tasks. All the results are obtained performing a leave-one-out cross validation over graphs of the two datasets relative to OFFICE and SCHOOL building types (see Section 5.2). OFFICE contains 20 graphs, while we used a subset of 30 graphs from SCHOOL for satisfying implementation constraints of kLog. Graphs from both data sets contains about 40 nodes each (see Section 5.1). On a single core of a Intel Core i7-3610QM@2.30 GHz CPU with 8 GB RAM, all the kLog learning

tasks discussed below run in less than 20 minutes (for all the datasets combined using cross validation), while Extra-Trees algorithm with a forest of 500 trees take approximatively slightly less then 1 hour for performing the same tasks.

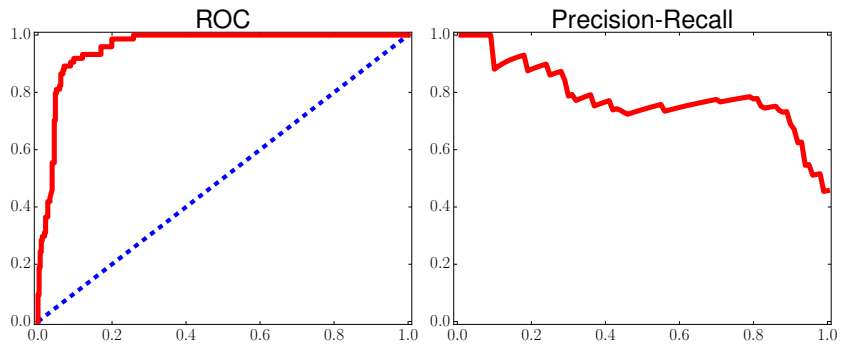
Evaluation is performed looking at the error rate of classification and at two curves. ROC (Receiver Operating Characteristic) curves plot the false positive rate on the x axis against the true positive rate on the y axis. Precision-recall curves plot the recall of classification on x axis against the precision of classification on y axis. Both axes goes from 0 to 1 and the quality of classification is given by the area underlying the curve. Area equal to 1 means a "perfect classifier" while the line with angle $\pi/4$ (area equal to 0.5) represents the performance of a random classifier.

6.2.1 Place classification

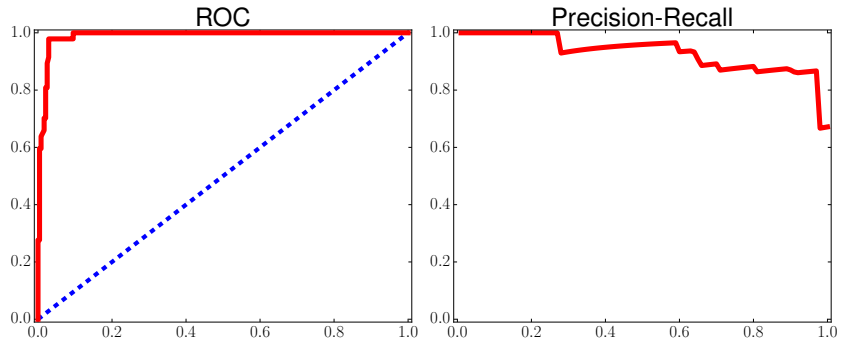
The first learning task mimics the classical task of place classification of rooms n of an environment, knowing the features V_n obtained from sensor readings, similarly to [64] (see Section 6.1.1). In this setting, we first consider the top level $\mathcal{L}_{R/C}$ labeling schema, asking kLog to classify each space as ROOM or CORRIDOR (note that discriminating between the two labels is of interest in tasks like exploration [92]). In other words, it performs a binary classification task for each space in the interpretation representing $\hat{G}$.

Figure. 6.2 shows the ROC and the precision-recall curves obtained by kLog for the building types OFFICE and SCHOOL. kLog is able to classify correctly ROOMs and CORRIDORs, as the large areas under the curves suggest. However, in both cases, Extra-Trees perform slightly better than kLog, as shown in Table 6.1. We can explain these results considering that it is easy to distinguish between ROOMs and CORRIDORs within a specific building type using only local features (e.g., the number of doors and the shape, usually squared for rooms and rectangular for corridors). This trend is confirmed when we use the larger set of labels $\mathcal{L}_{F/C/E/S}$, where the robustness (due to its randomness) of Extra-Trees outperforms the less flexible SVM of kLog. It is interesting to point out that the use of global knowledge in form of centrality features improves the results of Extra-Trees.

Overall, our results show that the proposed global approach based on kLog is able to perform place classification with performance comparably similar to that of a classical attribute-value local approach. Outcomes of a place classification task performed by kLog could provide a new and different perspective if integrated within a multi-modal standard semantic



(a) *Classification of spaces using* $\mathcal{L}_{R/C}$ *for* OFFICE*s.*



(b) *Classification of spaces using* $\mathcal{L}_{R/C}$ *for* SCHOOL*s.*

Figure 6.2: *Place classification for* OFFICE*s and* SCHOOL*s.*

mapping approach. For example, it is in principle easy to extend a (probabilistic) framework like that of [77], which assigns labels according to different features derived from camera data, object recognition, and geometry of the environment (as different potentials), to include also a new potential based on features returned after global reasoning, which could propose a new label not based on sensorial data, but on the global structure of the portion of a building which is currently known.

6.2.2 Building classification

Classifying entire buildings according to their function is a task that is much less explored in the literature, but that is intuitively expected to require reasoning about regularities and patterns of a specific building type. We refer to this task as *building classification*: we assume to know the structure, the geometry, and the room labels of a (floor of a) building, and we ask kLog and Extra-Trees to perform binary classification, categorizing the function

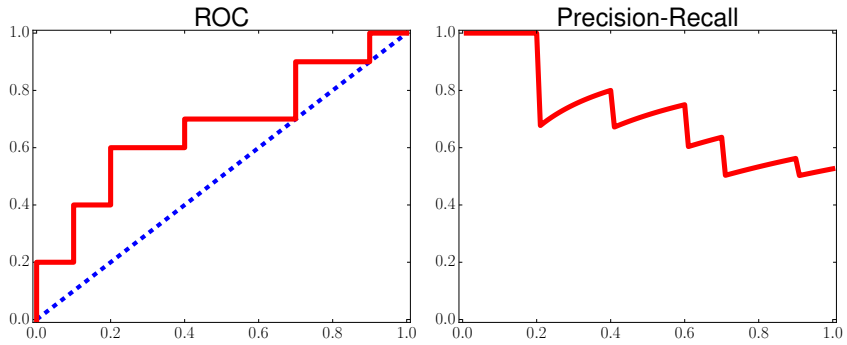


Figure 6.3: *Building classification in kLog using $\mathcal{L}_{R/C}$.*

of the building as OFFICE or SCHOOL (see Section 6.1.1). As input data we use the datasets (all the interpretations of) OFFICE and SCHOOL combined together. Since Extra-Trees reason only on local data and can classify only one room at a time, we ask the classifier to label each room according to a building type, and then we take the most frequent label to classify the whole floor. We perform tests using $\mathcal{L}_{R/C}$ and $\mathcal{L}_{F/C/E/S}$ schemas for labeling the rooms of the initial interpretations. Using $\mathcal{L}_{R/C}$ we assume that only a limited knowledge on the function of the rooms that a robot has viewed is available, while using $\mathcal{L}_{F/C/E/S}$ we assume to have more detailed knowledge on the function of the spaces.

Figure6.3 shows the ROC and the precision-recall curves for kLog when using $\mathcal{L}_{R/C}$. Good results are obtained, showing that, when the task is the classification of the whole building, reasoning on its global structure can compensate for a limited amount of information about the functions of the single spaces. If we compare the kLog results to those obtained using Extra-Trees (Table 6.1) we can see that kLog outperforms Extra-Trees. However, using $\mathcal{L}_{F/C/E/S}$, a more informative knowledge on the function of the spaces (which could be in principle more difficult to obtain and more error-prone) overshadows the global properties of the environment, allowing Extra-Trees to carry out more accurate classifications. kLog performs better with the $\mathcal{L}_{R/C}$ labeling of rooms, namely when coarse-grained information about rooms is available. This can happen, for instance, when a robot can only use data from laser range scanners (which can be enough to distinguish between rooms and corridors, as shown [66] and we discussed in Chapter 4) and not data from cameras (which could be used to distinguish between a more detailed set of labels like $\mathcal{L}_{F/C/E/S}$ as done, for example, in [77]), e.g., due to camera failure.

The ability of classifying whole buildings could be useful for a robot

moving in large heterogeneous structured environments, like a university campus, and needs to know whether the floor in which it is currently navigating hosts teaching activities (i.e., it is composed of classrooms and lecture halls) or the administrative section of the campus (i.e., it is composed of faculty's offices, reception, ...). We further test this ability by using the publicly available datasets of the MIT campus [7] that we have introduced in Section 4.2.3. We randomly select a sample of 33 floor plans belonging to 3 different types of buildings: DORMs, OFFICEs, and RESEARCH LABs, and we use kLog to perform a (leave-one-out cross validation) classification with the original labeling schema of the datasets, obtaining a $93,94\%$ accuracy on assigning the correct type to the floor plans, which corresponds to a very accurate classification.

6.3 Discussion

This chapter has discussed how SRL techniques can be used for reasoning on the whole structure of buildings, exploiting information on the connections and on the geometry of rooms, in the context of semantic classification of spaces. We compared the performances obtained by kLog with those obtained by a standard classifier that can be used for performing place classification, Extra-Trees. The outcomes show that the application of structured learning methods to semantic classification reasoning on the building structure is promising, especially when the task requires capturing and reasoning on the regularities of buildings and the available information about room labels is coarse-grained. While semantic classification usually requires knowledge about the environment geometry and appearance, the use of knowledge about the whole structure of the building could reduce the amount of such information that is required and that is obtained from sensors. So, our results on the use of kLog suggest a trade-off between knowledge of details of single rooms and knowledge of the whole structure of a building when performing semantic classification. Our structured learning method could potentially be integrated in a standard semantic mapping framework to provide a new perspective on the building structure that can be used to produce better semantic maps.

In this chapter we have investigated if the use of knowledge on the entire structure of environments can provide benefits to semantically understand indoor environments. The results and the insights obtained with this approach are used in the next chapter for providing an analytical model of the structure of indoor environments.

CHAPTER 7

Modeling building types

In Chapter 5, we have introduced a graph representation of indoor buildings which can be used to reason on the building structure. In this chapter, we use this representation to extract a model of a specific building type from a set of real floor plans. This model is used for sampling a new set of graphs representing possible instances of floor plans that have the same structural characteristics and node labels of a realistic floor plan of a building.

The aim of this framework is to obtain semantic knowledge on the *global* structure of a previously unknown (or partially visited) indoor environment (building). An application of this framework for predicting structural features of partially explored buildings is discussed in Chapter 8.

We propose to use a generative method that, following the general pattern of *Constructive Machine Learning* (CML) [27] is able, starting from a representative set of buildings, to model their topological structures and labeling schemas and to sample buildings belonging to the same type. In CML, the ultimate goal of learning is not to find good models of the data, but instead to find particular instances of the domain which are likely to exhibit some desired properties, while selectively sampling from an infinite or exponentially large domain. Our generative model is based on the graph representation of indoor environments introduced in Section 5.1.

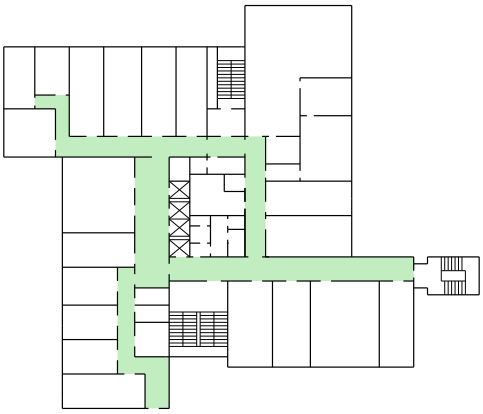
Given an initial set of buildings $\mathcal{G}$, our approach segments the graphs representing these buildings for finding significant subgraphs, which are then clustered according to their similarity. Similarity between graphs is calculated using *graph kernels*, which are the equivalent of kernel methods for structured data. Kernel methods are the basis of several machine learning techniques, like, among others, Support Vector Machines (SVM) [26]. In this chapter we use two graph kernels, namely the *Weisfeiler-Lehman Subtree Kernel* (K_{WL}) [86] and the *Graph Hopper Kernel* (K_{GH}) [33], which are introduced in Section 7.1.

Finally, a new graph representing a predicted building (or a predicted part of a building) is generated by sampling subgraphs from clusters and connecting them. The generated graph has similar topological structure and semantic labeling schema as the graphs representing initial buildings.

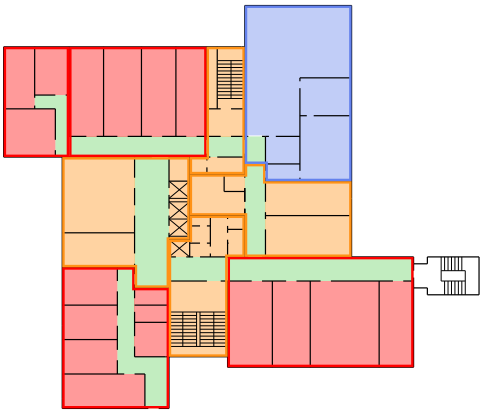
Under the CML framework we try to analytically model the concept of building type (see Chapter 4, Section 4.2.1); this is performed by using our model for sampling a set of new instances of graphs which are consistent with the features of a building type. Moreover, our approach, similarly to that by [7], can be thought as a move from the mainstream room-level perspective, modeling the semantic relations between perceived features and rooms, to a building-level perspective, modeling the connections between rooms and the structure of buildings. Our approach does not rely on sensor data acquired by robots, but on a knowledge base of graph-based topological structures and labeling schemas of buildings that can be obtained from previous semantic mapping efforts or, as in our case, from other sources (like collections of blueprints). It thus relies on *a priori* knowledge rather than on incrementally-acquired sensor-based knowledge.

A similar task addressing the problem of sampling new graphs which possess consistent structure with respect to a given set of input graphs, also following the Constructive Machine Learning paradigm, is addressed in a recent paper by [27]. The author develops a graph grammar to learn a distribution for MonteCarlo Markov chain sampling in a data-driven fashion. Probability estimators are implemented using a one-class Support Vector Machines graph kernel-based graphs classifier. Differently from our approach, [27] uses graph grammars to propose the edit operations which result in the sampling of new graphs, while our method exploits task-dependent knowledge for place classification on the role and the label of each node in the graph for guiding the sampling.

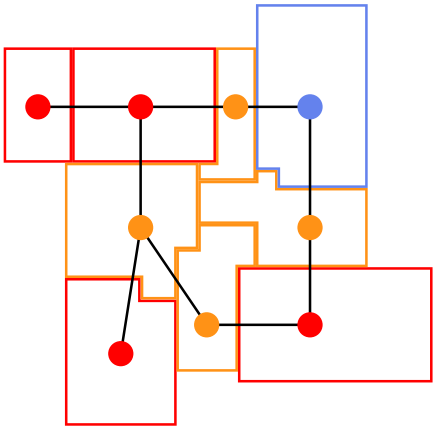
One of the main original ideas of this chapter is the use of a two-layered semantic representation for indoor environments. The first layer is provided by the graph, the semantic map, representing labeled rooms and their con-



(a) *Original floor plan.*



(b) *Functional areas.*



(c) *Functional graph.*

Figure 7.1: *Functional analysis of a research office building floor plan. In Figure 7.1a the corridors are highlighted in green. In Figure 7.1b the floor plan is segmented in* 9 *functional parts, one for each corridor. Similar rooms are grouped together with the corridors to which they are connected to. Each part is a* functional area *and is colored according to is function: red means office, blue means research labs, orange means service rooms such as stairs, elevators, toilets, storage rooms, and meeting rooms. From such a segmentation an abstract graph representation in the form of a* functional graph *is derived in Figure 7.1c.*

nections. As said, we segment the floor plans of large structured buildings in non-overlapping parts. Each one of these parts is expected to be formed by a group of neighbouring rooms with a similar function (e.g., a cluster of offices, an administrative section, a group of service rooms such as toilet, kitchen, and vending machines). We call these parts *functional areas*. An example of functional areas can be seen in Figure 7.1. If we observe the floor plan of the research office building displayed in Figure 7.1a, we can notice that rooms with a similar function are usually close to each other and connected to the same corridor (in green). It is thus possible to separate the floor plan into functional areas, shown with different colors in Figure 7.1b. In our example, we highlight in red the parts where offices are located, in blue the parts with research labs, and in orange the parts where support rooms, such as stairs, vending machines, elevators, or toilets, are located. This representation can be summarized in the graph structure of Figure 7.1c. Differently from Figure 5.1, in Figure 7.1c each node of the *functional graph* represents a *functional area* (namely a set of connected rooms that share the same function).The functional graph provides the second layer of our representation. The identification of functional areas introduces a new abstract perspective which can be used to reason on the structure of the buildings, enabling reasoning mechanisms able to predict the function of unknown parts of buildings and to generate new instances of buildings, as we show in ur experiments.

The structure of the chapter is the following. In Section 7.1 we overview the concept of graph kernels by discussing some notable examples that are used in the following of the chapter. In Section 7.2 we introduce the main assumptions and the goal of our approach. Section 7.3 describes how our models of building types are extracted from data. Section 7.4 presents our sampling mechanism, which employs the model to derive new data. Experimental results are reported in Section 7.5.

7.1 Graph kernels

The task of learning a model of structured data like graphs, as we do in this chapter, is often complex and requires *ad hoc* techniques since standard machine learning techniques cannot be naturally adopted due to the dimensionality of the data [30]. We make a substantial use of *graph kernel* methods, which use a decompositional approach to measure the similarity between two graphs. In this section we provide a general overview on the concept of graph kernel, as introduced by [47], and we describe the main graph kernels that we use in this chapter. Among all the possible

graph kernel families, the general class of *convolutional kernels* proposed by [47] represents the guiding principle in kernel design for structured objects [28, 40]. Convolutional kernels are based on the idea that the semantics of a graph can be captured by a relation R between the graph and its parts. A graph kernel is then defined as a composition on kernels defined on different parts of the graph.

Let $G = (N, E) \in \mathcal{G}$ be a graph and $\{g_1, \ldots, g_D\}$ one of its decompositions into (possibly overlapping) parts (subgraphs). Each part g_d in an element of a countable set G_d, for $d = 1, \ldots, D$, $D \geq 1$. Consider a relation R defined on $G_1 \times \ldots \times G_D \times \mathcal{G}$, where $R(g_1, \ldots, g_D, G)$ is true if the set $\{g_1, g_2, \ldots, g_D\}$ is one of the possible sets of the parts (subgraphs) of G (i.e., in which G can be decomposed). Given R, we can define the function R^{-1} as the decomposition function which, returns the sets of parts of G: $R^{-1}(G) = \{g_1, \ldots, g_D : R(g_1, \ldots, g_D, G)\}$. Consider now any positive definite kernel function K_d over $G_d \times G_d$, $d = 1, \ldots, D$. For two graphs $G, Q \in \mathcal{G}$, we can define a *convolutional* or *decomposition kernel* on graphs as the function:

$$K(G, Q) = \sum_{\substack{g_1,\ldots,g_D \in R^{-1}(G) \\ q_1,\ldots,q_D \in R^{-1}(Q)}} \prod_{d=1}^{D} K_d(g_d, q_d)$$

Given a generic graph kernel K, the similarity measure between two graphs $G, Q \in \mathcal{G}$ can be defined as the normalized version of the graph kernel:

$$K_{normalized}(G, Q) = \frac{K(G, Q)}{\sqrt{K(G, G)K(Q, Q)}}$$

In this work we use two instances of convolutional graph kernels: the *Weisfeiler-Lehman Subtree Kernel* (K_{WL}) [86] and the *Graph Hopper Kernel* (K_{GH}) [33]. Several other kernels, such as those by [28], [63], and [51] have been tested with less convincing results (numerical results are not reported here).

The K_{WL} belongs to the family of Weisfeiler-Lehman graph kernels [86], that exploit the Weisfeiler-Lehman test of isomorphism on graphs as a rapid feature extraction scheme. K_{WL} is implemented as an iterative procedure in which, at each iteration, the node labels are augmented by concatenating to the current label of each node the sorted list of labels of its adjacent nodes. The resulting augmented label list is then compressed into a new, shorter label list using an hash function. Each graph G is thus mapped to a sequence of graphs $\{G_0, G_1, \ldots, G_h\} = \{(N, E, L_0), (N, E, L_1),$

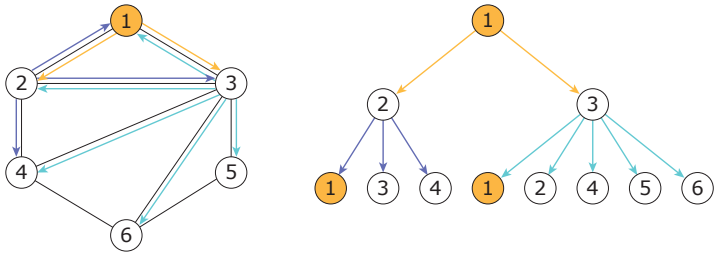


Figure 7.2: *An example of a* $d = 2$ *subtree rooted from node* 1.

$\ldots, (N, E, L_h)\}$, where G_i and L_i are the graph and the label set (one label per node) after i iterations of the algorithm, respectively, and h is the total number of iterations. Given any graph kernel K_b called *base kernel*, K_{WL} is then computed as:

$$K_{WL}(G, Q) = K_b(G_0, Q_0) + K_b(G_1, Q_1) + \ldots + K_b(G_h, Q_h)$$

In our setting we use as base kernel $K_b(G_i, Q_i)$ a subtree kernel. It computes all the rooted subtrees of $G = (N, E)$ (a rooted subtree is an acyclic sub-graph of G with a given depth d and rooted on a node $n \in N$, where nodes $n' \in N$ can be repeated in different branches of the tree; an example of a rooted subtree is shown in Figure 7.2). These rooted subtrees are the outcome of the decomposition R^{-1}. For each iteration i, $i = 1, \ldots, h$, K_b counts the number of common labels between all the subtrees of G_i and Q_i rooted in two nodes with the same label $l^* \in L_i$. The complexity of K_{WL} is $O(Nhm + N^2hn)$, where N is the total number of graphs, and n and m are the number of nodes and edges of the graphs respectively (assumed equal for all graphs for simplicity).

In K_{GH} [33], the decomposition relation R^{-1} is based on the shortest path between each pair of nodes:

$$K_{GH}(G, Q) = \sum_{\pi \in \mathcal{P}, \pi' \in \mathcal{P}'} k_p(\pi, \pi')$$

where k_p is a kernel defined on paths and $\mathcal{P}$ and $\mathcal{P}'$ are the sets of shortest paths between all pairs of nodes of G and Q, respectively. The path-kernel k_p is defined on two paths π and π' as:

$$k_p(\pi, \pi') = \begin{cases} \sum_{j=1}^{|\pi|} k_n(\pi(j), \pi'(j)) & \text{if } |\pi| = |\pi'|, \\ 0 & \text{else.} \end{cases}$$

where $\pi(i) = n_i$ indicates the i-th node in the path $\pi = (n_1, \ldots, n_{|\pi|})$. We use a linear node kernel $k_n(n_i, n_j)$ that returns 1 if n_i and n_j have the same label and 0 otherwise. Total complexity of K_{GH} is $O(N^2(n^2(m + \log n + d + \delta^2)))$ where N, n, and m are defined as in K_{WL}, δ is the length of the longest shortest path, and d is a constant.

7.2 Developing a building type model

As introduced and motivated in Chapter 4, we exploit the concept of *building type*, developing specific models for each building type. Reasoning on buildings of the same type allows to identify their common characteristics, that can be modeled effectively.

The problem that we address in this thesis is: given set of graphs $\mathcal{G}$ representing floor plans of a given building type, sample a semantic map $\hat{G}$ with a structure similar to that of graphs in $\mathcal{G}$.

We use a data set of semantic maps representing floorplans introduced in Section 5.1 considering the building type SCHOOL. (In general, our method can be straightforwardly applied to other building types. We developed and tested our approach for OFFICEs and the experimental results are similar to those obtained for SCHOOLs. However, for the sake of space and clarity, we do not discuss these results here.) Specifically, we start from a graph database $\mathcal{G}$ built by hand from labeled floor plans of 50 real schools (see Section 5.2) and containing about 2000 rooms.

We use two of the layers of the hierarchical semantic labeling schema presented in Section 4.3 At the top level, the labeling schema called $\mathcal{L}_{R/C}$ contains only the two general categories ROOM and CORRIDOR. For clarity, in the following figures, a square is used for CORRIDORs, while a circle is used for ROOMs, as in Figure 5.1.

In addition, we use a specific labeling schema for the SCHOOL building type $\mathcal{L}_{school}$. The complete set of labels and their color used for plotting regarding the SCHOOL building type and extracted from architectural sources [69, 95] and are reported in Figure 7.3.

7.3 Creation of the model

7.3.1 Segmentation

Each graph $G = (N, E) \in \mathcal{G}$ is segmented in smaller, non-overlapping subgraphs $S = \{s_1, s_2, \ldots\}$, such that, for each $s_i = (N_i, E_i), s_j = (N_j, E_j) \in S$, it holds $N_i \cap N_j = \emptyset$ and such that $\bigcup_{s_i \in S} N_i = N$. Segmentation is per-

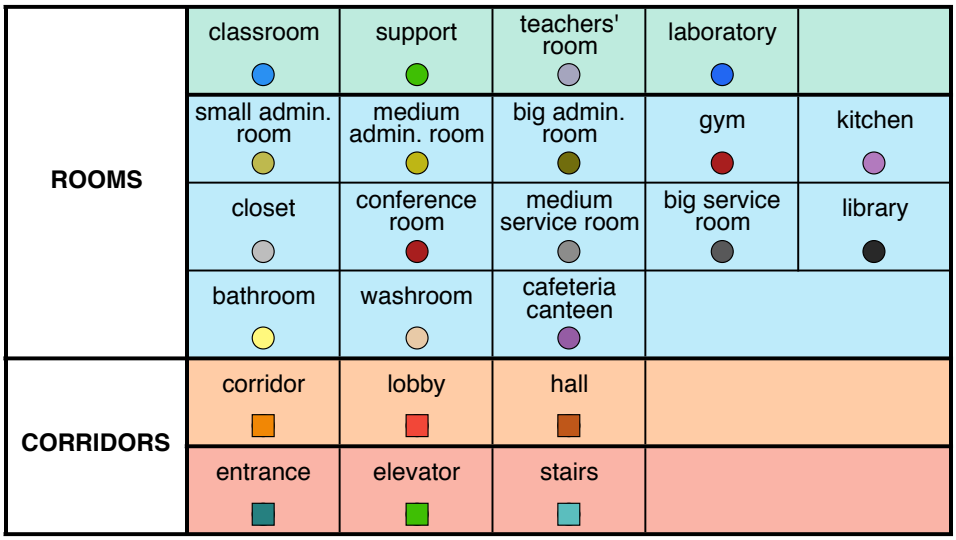


Figure 7.3: *Labeling schema for* SCHOOL *building type. Colors and shapes indicated for each label are used through the thesis.*

formed by removing some edges, or cuts, $c \in E$ from G and aims at identifying the *functional areas*. Functional areas, as introduced in the beginning of this chapter, and as exemplified in Figure 7.1, are groups of neighbouring rooms with a similar function within the buildings (i.e., a subgraph), which are connected to (nearby) corridors. The segmentation step is thus particularly important in our framework since it should divide each floor of a building according to the structure of the graph and the function of the rooms.

Two different unsupervised methods are used for segmentation. The first method is the Normalized Cut spectral method of [87], or `ncut`, as we used in [58]. `ncut` is a general and context independent segmentation method that partitions a graph into components of roughly the same size. Specifically, `ncut` recursively partitions a graph into two components $s_a = (N_a, E_a)$ and $s_b = (N_b, E_b)$ based on the magnitude of the eigenvalues of the Fiedler vector obtained after an *ad hoc* eigen-decomposition of the graph adjacency matrix. Each part is then recursively segmented until a threshold on the *normalized-cut* measure for each partition is obtained. The normalized-cut measure is computed as:

$$\textit{normalized-cut}(s_a, s_b) = \frac{Cut(s_a, s_b)}{Assoc(s_a, G)} + \frac{Cut(s_a, s_b)}{Assoc(s_b, G)}$$

where, called $w(u, v)$ the weight of the edge between the nodes u and v (in

our setting, $w(u, v) = 1$; note that $w(u, v) = 0$ if u and v are not connected by any edge):

$$Cut(s_a, s_b) = \sum_{u \in N_a, v \in N_b} w(u, v)$$

and

$$Assoc(s_a, G) = \sum_{u \in N_a, t \in N} w(u, t)$$

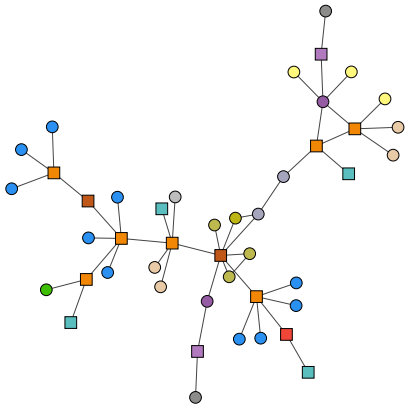
Similarly for $Assoc(s_b, G)$.

The second method (that we refer to as `corr`) depends on our particular domain and is based on the distinction between ROOMs and CORRIDORs of $\mathcal{L}_{R/C}$. It follows the intuition that CORRIDORs are particularly important nodes for an indoor environment and can be considered as the *hub*s of the corresponding graph. Under these premises, `corr` assigns each ROOM node to the nearest CORRIDOR node (if there is a tie, the CORRIDOR with the highest degree is selected). Each edge between two CORRIDOR nodes or between two ROOM nodes associated to two different CORRIDORs is considered as a cut c and removed by the segmentation process. This second segmentation method, unlike `ncut`, does not guarantee to produce a balanced segmentation of the graph; however, in our problem domain, both segmentation methods perform similarly in terms of number and size of subgraphs obtained after the segmentation, as shown in the example of Figure 7.4. A more detailed comparison between the two methods is performed in Section 7.5.

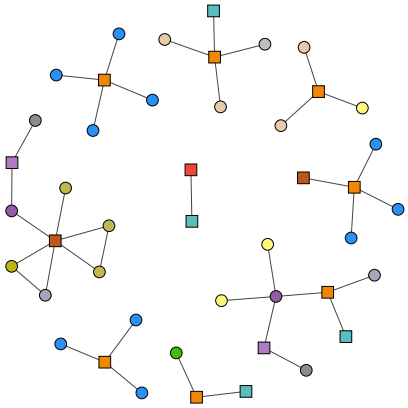
The results of the segmentation process performed over a graph (semantic map) G can be represented as a graph $H = (S, U)$, where every node in S is a subgraph $s \in S$ and an edge $u \in U$ exists between two nodes $s_i = (N_i, E_i)$, $s_j = (N_j, E_j) \in S$ if there is a cut $c = (n_i, n_j) \in E$ between two nodes $n_i \in N_i$ and $n_j \in N_j$. We refer to the graph H as *functional graph*. A pair of $G \in \mathcal{G}$ and of the associated $H \in \mathcal{H}$, where $\mathcal{H}$ is the set of all functional graphs, constitutes a representation of the same building at two different levels of abstraction , as discussed at the beginning of this chapter.

7.3.2 Clustering

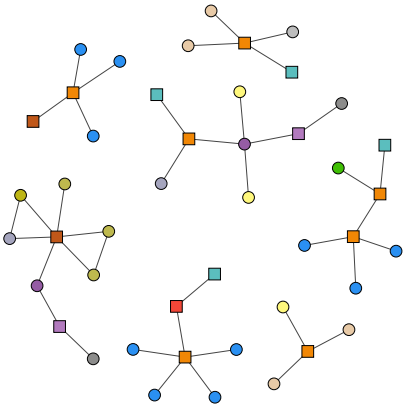
After all graphs $G \in \mathcal{G}$ have been segmented, the *clustering* step is responsible to group together all the subgraphs that have a similar structure, and



(a) *Initial graph G.*



(b) `ncut`.



(c) `corr`.

Figure 7.4: *An example of applications of different segmentation methods to a graph G.*

thus to identify the same functional areas over several buildings. All the subgraphs S_ℓ obtained from all the graphs G_ℓ in $\mathcal{G}$ are now considered altogether: $\mathcal{S} = S_1 \cup S_2 \cup \cdots \cup S_{|\mathcal{G}|}$. For each pair of subgraphs s_i, s_j an affinity value $\phi(s_i, s_j)$ is computed. Affinity $\phi(s_i, s_j)$ measures the similarity between subgraphs and is computed using one of the graph kernels K (K_{WL} or K_{GH}), exploiting the fact that each subgraph $s = (N_s, E_s)$ is actually a graph in which each node $n \in N_s$ is described by its label $\mathcal{L}_{school}$:

$$\phi(s_i, s_j) = K(s_i, s_j)$$

so ϕ depends on both the topological structure of s_i and s_j and on the labels of their nodes.

In order to cluster together similar subgraphs, we use the *affinity propagation* clustering algorithm [35]. Affinity propagation is a message passing clustering algorithm which selects a subset of the data as models for their relevance. Each data element exchanges two kinds of messages with other data elements indicating how much it consider itself suited to became a model for the other elements and how much each element is considered suited to be a model by the other, nearby, elements. These messages are exchanged recursively until stable clusters are formed or a maximum number of iterations is reached. Affinity propagation does not require the number of clusters as a parameter. A damping factor α between 0 and 1 is used for smoothing the message update, in order to prevent oscillations and increase convergence rate to stable clusters. The use of affinity propagation clustering is a difference what we did in [58] where clustering is performed using Normalized Cut `ncut` [87], which sometimes result in a high number of small clusters and/or in a big cluster containing almost all of the data and which is experimentally outperformed by affinity propagation (data are not reported here).

The clustering step outputs a set of clusters $\mathcal{C} = \{C_1, C_2, \ldots\}$ and a function $\psi : \mathcal{S} \to \mathcal{C}$ that maps each subgraph to its cluster. Given a subgraph s, we call $C = \psi(s)$ the cluster to which s belongs. A cluster C is expected to represent a *functional area* and the subgraphs in the cluster are parts of (possibly different) buildings that share the similar structure and labels. The clustering results are influenced by the way in which the segmentation is performed, by the cluster algorithm, and by the graph kernel used.

The semantic maps, the functional graphs, and the clusters constitute the model $\mathcal{M} = \{\mathcal{G}, \mathcal{H}, \mathcal{C}\}$ for a specific building type constructed starting from $\mathcal{G}$.

7.3.3 Segmentation and cluster evaluation

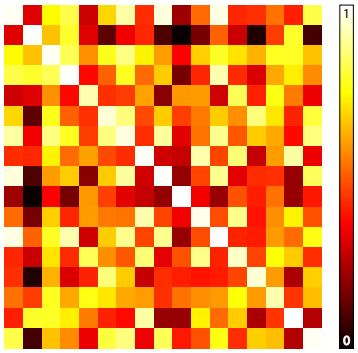
Our segmentation and clustering steps are implemented[1] in MATLAB R2015b. We present here a quantitative evaluation of the models built by our approach. Within our context, we consider as desirable the presence of the following features: a high intra-cluster similarity, a low inter-cluster similarity, a limited number of clusters, a number of clusters that is stable with respect to the increase of the number of graphs in the dataset $\mathcal{G}$, and a similar size for each cluster (i.e., the absence of many small clusters with less than 5 subgraphs).

We empirically find that a damping factor of $\alpha = 0.8$ for affinity propagation produces clusters with all the desired properties for our data set of schools. Remember that affinity ϕ is calculated according to K_{WL} or K_{GH}. Clusters obtained using K_{GH} present better features than those obtained with K_{WL}, but at the expense of a longer computing time. K_{GH} features, computed on shortest paths, capture the structure of the graphs of our domain better than the high-dimensional sparse features computed on subtrees by K_{WL}. Using K_{WL}, we obtain a higher number of clusters (27 and 12 for K_{WL} and K_{GH}, respectively, using 50 graphs in $\mathcal{G}$ that are segmented by `corr` into 287 subgraphs) and sometimes it is not able to correctly categorize all subgraphs (resulting in a highly different number of subgraphs per cluster). On a i7QuadCore Intel 820M 16GB computer, the entire segmentation and clustering steps take less than one minute using K_{GH} and K_{WL} on $\mathcal{G}$ composed of 50 graphs. Note that clustering is performed only once and clusters are then stored in the model $\mathcal{M}$ for being sampled later. From now on, we consider only K_{GH}, which performs better than K_{WL}.

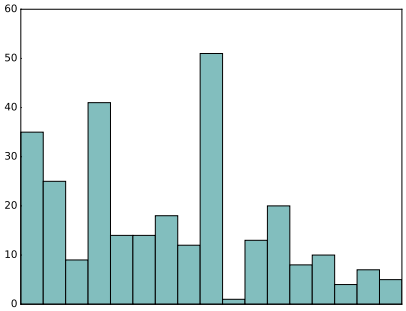
Both segmentation methods `ncut` and `corr` perform correctly for the task, producing a partition of each graph $G \in \mathcal{G}$ in components of approximatively the same size, by removing a limited number of edges. However, different segmentation methods result in slightly different clusters. `ncut` segments 50 graphs into 207 subgraphs, which are then clustered into 17 clusters. `corr` results in a higher number of subgraphs (287) but in a lower number of clusters (12).

These findings are shown in Figure 7.5. Figure 7.5a and Figure 7.5c show intra-cluster similarity and inter-cluster similarity for clusters obtained after segmentation performed with `ncut` and `corr`, respectively. Figures show *heat maps*. A heat map is a $|\mathcal{C}| \times |\mathcal{C}|$ matrix where cell (i, j) is light (dark) if subgraphs in C_i are similar to (different from) subgraphs in C_j. Similarities between subgraphs are computed using graph kernels.

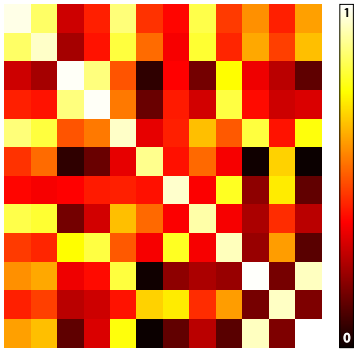
[1] Code is available upon request.



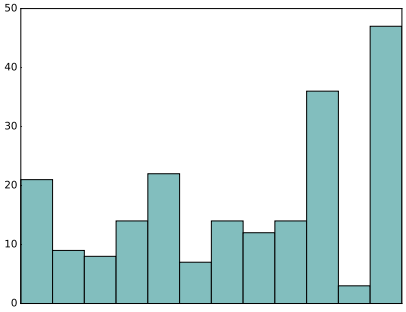
(a) *Heat map obtained using* `ncut`.



(b) *Number of subgraphs in clusters using* `ncut`.



(c) *Heat map obtained using* `corr`.



(d) *Number of subgraphs in clusters using* `corr`.

Figure 7.5: *Results of the clustering process for different segmentation algorithms. The color of a cell (i, j) in the heat map represents if subgraphs in the cluster C_i are, on average, similar to (light coloro) or different from (dark color) the subgraphs in C_j. Bar charts have a column for every cluster in $\mathcal{C}$. Heights of the column correspond to the number of subgraphs belonging to the clusters.*

High intra-cluster similarity results into light diagonal in the map, where low inter-cluster similarity results in dark cells outside the diagonal. Figure 7.5b and Figure 7.5d show the number of subgraphs in each cluster for `ncut` and `corr`, respectively. Every column indicates a separate cluster and the the height of each column represents the corresponding number of subgraphs.

Differently from `corr`, `ncut` guarantees by design that all subgraphs have a similar size, i.e., that there are no subgraphs containing only one or two nodes or, on the other side, subgraphs containing most part of the graph. Actually, using `corr`, we obtain such undesirable graph partitioning only in few cases and, due to the particular structure of the buildings, globally the partitioning results are similar using the two methods (as in the example of Figure 7.4).

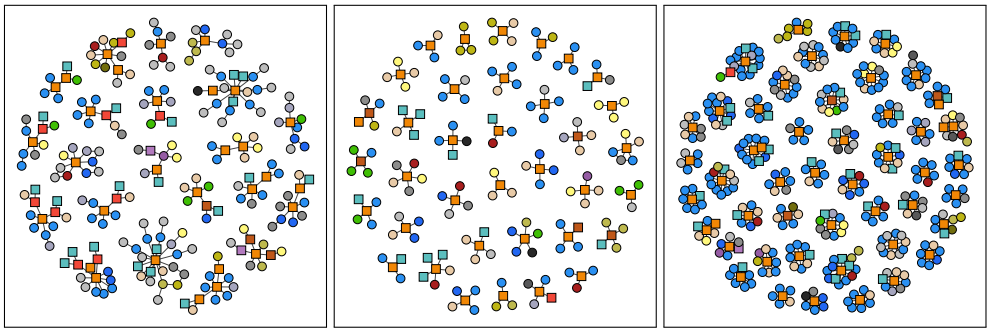


Figure 7.6: *An example of thee clusters obtained using* `corr` *for segmentation.*

Figure 7.6 shows an example of three clusters obtained using CORR. Similar subgraphs belong to the same cluster while different subgraphs are placed in different clusters. Figure 7.7 displays all clusters obtained using respectively `ncut` (Figure 7.7a) and `corr` (Figure 7.7b). Similarity is computed between all subgraphs using K_{GH} obtaining a similarity matrix of size $|\mathcal{S}| \times |\mathcal{S}|$. This high-dimensional feature space is then reduced to a two-dimensional one using a *t-distributed stochastic neighbor embedding* (t-SNE) [62]. t-SNE models each high-dimensional object with a two-dimensional point in such a way that similar objects are modeled by nearby points and dissimilar objects are modeled by distant points. Subgraphs belonging to the same cluster are shown with the same color. A legend which indicates the cluster color is shown at bottom left. It can be clearly seen that, even in a reduced 2-D dimensional space, similar subgraphs are effectively clustered together by our approach, while different

subgraphs are assigned to different clusters. The 2-D dimensional space is scaled to the range $[0, 1]$.

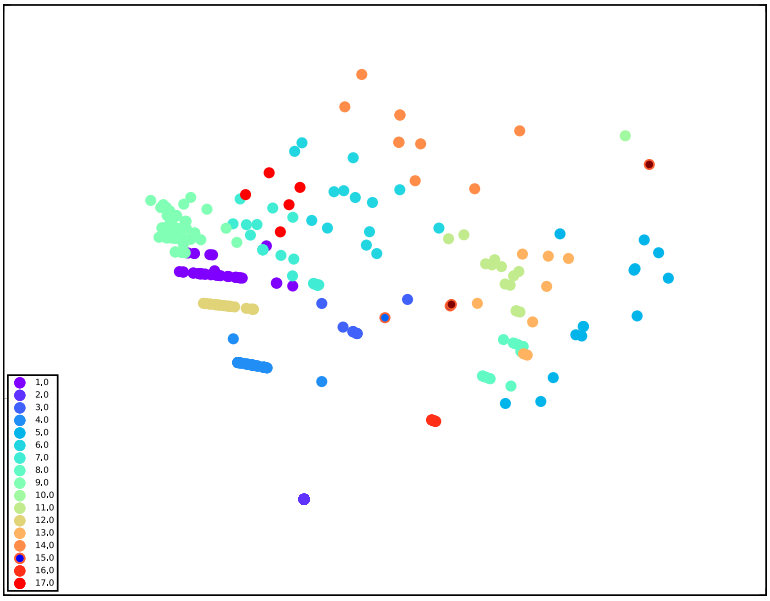
7.4 Sampling graphs from a building type model

The model $\mathcal{M} = \{\mathcal{G}, \mathcal{H}, \mathcal{C}\}$ obtained in the previous section can be used for generating new graph samples representing buildings or their parts. Formally, we assume that all graphs G in our dataset $\mathcal{G}$ follow the same (unknown) graph probability distribution P that is shared by all the buildings of the same type. This is reasonable since $\mathcal{G}$ refers to buildings of a single building type. Sampling a new graph from an unknown graph distribution P is a Constructive Learning Problem (CLP) [28], which can be seen as the task of sampling a graph from an empirical conditional probability distribution using an adaptive data-driven procedure. The model $\mathcal{M}$ is thus used to obtain an empirical probability distribution (which approximates P) for the set of graphs $\mathcal{G}$ using a hierarchical sampling process. The empirical distribution is modeled by a Markov Chain (MC) and the sampling process exploits a Monte Carlo Markov Chain (MCMC) [53]. In order to sample a new graph $\hat{G}$, the sampling process starts from generating its abstract structure, namely its functional graph $\hat{H}$, adding incrementally more details, following, in the reverse order, the segmentation and clustering processes of Section 7.3. Sampling is composed of several subsequent steps δ, each one modeling an empirical probability distribution P_δ by using a MCMC method.

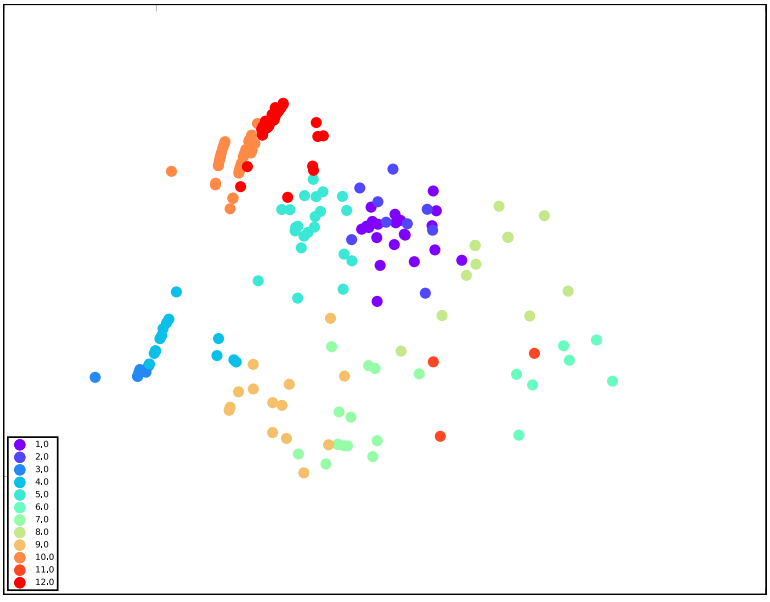
In the following of this section, first, a more formal definition of the sampling problem is given. Then, we introduce the MCMC method that approximates the empirical probability distributions P_δ for the sampling steps. Finally, we describe each sampling step.

7.4.1 Generating graphs from empirical distributions

Given a general domain of graphs $\mathbb{G}$ and a finite set of example graphs $\mathcal{G} \subseteq \mathbb{G}$, in this section we define the task of sampling a set of graphs $\hat{\mathcal{G}} \subseteq \mathbb{G}$ using an adaptive data-driven procedure. Let P be the (usually unknown) probability distribution associated to $\mathbb{G}$. Let P_θ be an approximation of P parametrized by θ. Following an approach similar to that by [27], define L as a non-negative convex function that, given two probability distributions over $\mathbb{G}$, returns zero if the two distributions are identical and positive values if the two distributions are different (the larger the values the more different



(a) `ncut`.



(b) `corr`.

Figure 7.7: *2D representation of clusters of subgraphs in $\mathcal{S}$ using t-SNE [62]. Points are scaled to the $[0, 1]$ range for plotting.*

the distributions). We can define an instance of the function L as:

$$L(P, P_\theta) = D_{KL}(P\|P_\theta) + D_{KL}(P_\theta\|P)$$

where D_{KL} is the Kullback-Leibler divergence:

$$D_{KL}(P\|P_\theta) = \sum_{G\in\mathbb{G}} P(G)\log\frac{P(G)}{P_\theta}$$

(Similarly for $D_{KL}(P_\theta\|P)$.)

The sampling problem, following the Constructive Machine Learning paradigm, is to find a parametrization θ^* that solves the following optimization:

$$\theta^* = \arg\min_\theta \; L(P, P_\theta) \tag{7.1}$$

Since P and P_θ are both unknown and $\mathbb{G}$ has a possibly infinite cardinality, we consider a finite set of examples $\mathcal{G} \subseteq \mathbb{G}$, that are drawn according a probability distribution $P_\mathcal{G} \sim P$. Let $\hat{\mathcal{G}}_\theta$ be a sample set of graphs randomly generated according to P_θ and let $f_\mathcal{G}$ (respectively, $f_{\hat{\mathcal{G}}_\theta}$) be the probability distribution estimator for the samples in $\mathcal{G}$ (respectively, $\hat{\mathcal{G}}_\theta$); we can assume:

$$f_\mathcal{G} \sim P_\mathcal{G} \sim P$$

$$f_{\hat{\mathcal{G}}_\theta} \sim P_\theta$$

Then, finally, we can approximate (7.1) with:

$$\theta^* = \arg\min_\theta L(f_\mathcal{G}, f_{\hat{\mathcal{G}}_\theta}) \tag{7.2}$$

Then, (7.2) returns the solution of our constructive learning problem.

The parametrization θ^* can be used to model P_{θ^*}, in order to sample from it a new set of graphs $\hat{\mathcal{G}}$. Since P_{θ^*} is unknown, in the following section we describe our proposed procedure to obtain f_{θ^*} starting from an initial set of data and we show how it can be used for sampling new data.

7.4.2 Monte Carlo Markov Chains

In our particular context, the set of graphs $\mathcal{G} \subseteq \mathbb{G}$ represents floors of buildings. In order to drawn samples from a distribution that approximates P we use a Metropolis Hastings algorithm, a particular type of MCMC method [53]. This section provides a general description of the method,

which is applied, with different parameters, in the four steps of our sampling method. The detailed implementation of the method for each step is described in the following.

Generally speaking, the Metropolis Hastings algorithm is defined by a transition model $\mathcal{T}(g \to g')$ that specifies, for each pair of states g, $g' \in \mathbb{G}$ the probability of going from g to g' according to a stationary distribution Π (note that, when graphs in $\mathbb{G}$ are intended as states for MCMC we use the lowercase symbol g):

$$\Pi(G = g') = \sum_{g \in \mathbb{G}} \Pi(G = g)\mathcal{T}(g \to g')$$

The transition model $\mathcal{T}$ is defined in terms of a proposal transition distribution T, that generates, using some edit function, successors of a state g and then selects randomly the next candidate g' from these successors. We can either accept the proposal (and move to g') or reject it (and stay at g), using a transition probability $\mathcal{A}(g \to g')$. Namely,

$$\mathcal{T}(g \to g') = T(g \to g')\mathcal{A}(g \to g') \;\; g \neq g'$$

$$\mathcal{T}(g \to g) = T(g \to g') + \sum_{g \neq g'} T(g \to g')(1 - A(g \to g'))$$

The acceptance probability $A(g \to g')$ reduces to:

$$A(g \to g') = \min\left(1, \frac{\Pi(g')T(g' \to g)}{\Pi(g)T(g \to g')}\right)$$

Since our proposal distributions are (in first approximation) symmetric, we have that $T(g \to g') = T(g' \to g)$.

Finally, we approximate $\Pi(g)$ with a data-driven adaptive Boltzmann-like function b:

$$\Pi(g) \sim b(g) = e^{-\alpha\Gamma(g,\mathcal{G})} \tag{7.3}$$

where $\mathcal{G}$ is our set of samples, Γ is a cost function that penalizes configurations of g that are different from those in $\mathcal{G}$, and α is a user-defined parameter (which is set to 1 if not indicated differently). For each step of the sampling process described in the following, a definition of the proposal transition distribution T (also called *transition kernel*) and of the cost function Γ will be provided.

The number of iterations of each MCMC is selected in order to have reasonable chances that the mixing time is reached (i.e., that the transition model reaches a stationary distribution). We use an heuristic based on

the observation that multiple chains sampling the same distribution should, upon convergence, yield similar estimates, as suggested by [53]. This method uses the observation that different sets of samples collected at different times should, if the mixing time has been reached, show similar variance in each of the chains. More formally, we run K separate chains for $\tau + M$ steps starting from different starting points. After discarding the first τ samples from each chain, the convergence is checked (see [53, p. 522-523] for details). If the mixing time has been reached, then τ is chosen as the minimum number of iterations, if not τ is increased by a fixed amount and the test repeated.

7.4.3 Hierarchical sampling

The hierarchical sampling process that creates a new graph $\hat{G}$ from a graph model $\mathcal{M} = (\mathcal{G}, \mathcal{H}, \mathcal{C})$ is divided in four main steps:

1. cluster configuration sampling;
2. functional graph sampling;
3. subgraphs sampling;
4. node connections sampling.

A running example of these four steps is shown in Fig. 7.8 and used as a reference in the following.

Cluster configuration sampling

The first step is the cluster configuration sampling that returns the composition $\hat{V}_c$ of the sampled graph $\hat{G}$ in terms of how many subgraphs for each cluster are present in $\hat{G}$. The cluster configuration of each original graph $G \in \mathcal{G}$ is a vector:

$$V_c^G = \{v_1^G, v_2^G, \ldots, v_{|\mathcal{C}|}^G\},\ v_i^G \in \mathbb{N}$$

where v_i^G represents the number of subgraphs of G (in which G has been segmented) belonging to the cluster $C_i \in \mathcal{C}$.

Using the Metropolis Hastings algorithm, we start from a random initialization of $\hat{V}_c = \{\hat{v}_1, \hat{v}_2, \ldots, \hat{v}_{|\mathcal{C}|}\}$ and employ a transition kernel T_1 where, at each step, one component of the vector $\hat{V}_c$ is selected at random and increased or decreased by one unit. Values of each $\hat{v}_i$ are bounded in the interval $[0, \max_{G \in \mathcal{G}} {v_i}^G]$.

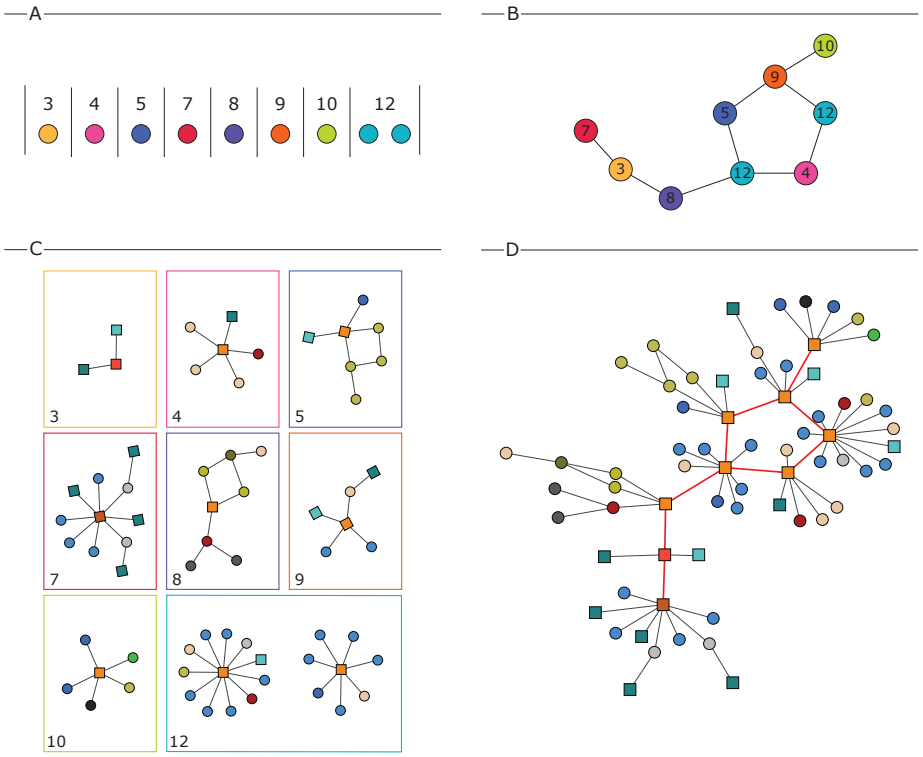


Figure 7.8: *An example of the sampling steps for generating a graph $\hat{G}$.*

The cost function Γ_1 is defined as:

$$\Gamma_1(\hat{V}_c) = \sum_{G \in \mathcal{G}} d_{pw}(\hat{V}_c, {V_c}^G)$$

where d_{pw} is the pairwise vector distance. In (7.3), α is set to 0.7 in case of `ncut` segmentation and to 0.55 in the case of `corr` segmentation. After $\tau_1 = 300$ iterations the sampled vector $\hat{V}_c$ is considered as final and the sampling continues with the next step.

In the example of Fig. 7.8, the sampled cluster configuration is:

$$\hat{V}_c = \{0, 0, 1, 1, 1, 0, 1, 1, 1, 1, 0, 2\}$$

where $|\mathcal{C}| = 12$, and is graphically represented in Fig. 7.8A.

Functional graph sampling

In the second sampling step, the sampled functional graph $\hat{H} = (\hat{S}, \hat{U})$ is computed from the cluster configuration $\hat{V}_c$. $\hat{H}$ is obtained creating a node $\hat{s}_i$ for each subgraph indicated by $\hat{v}_i$ and sampling an edge $\hat{u} = (\hat{s}_i, \hat{s}_j)$ between two subgraphs $\hat{s}_i$ and $\hat{s}_j$ that will be later connected.

This step is performed using the Metropolis Hasting algorithm introduced in Section 7.4.2 by selecting a random initialization for the edges of $\hat{H}$. The transition kernel T_2 is symmetric and consist in three moves: ADD an edge to the graph (randomly selecting the two nodes $\hat{s}_i$ and $\hat{s}_j$ to be connected), REMOVE an existing edge (selected randomly), and SWAP a node in an edge $\hat{u}_k$ (randomly select $\hat{u}_k = (\hat{s}_i, \hat{s}_j)$ and replace $\hat{s}_j$ with a different node $\hat{s}_t$, thus transitioning from $\hat{u}_k$ to $\hat{u}_k^{'} = (\hat{s}_i, \hat{s}_t)$). Each move is chosen using the following probabilities:

$$P_{\text{ADD}} = 1 - e^{-\lambda_1 \frac{|\hat{S}|}{|\hat{U}|}}$$

$$\epsilon = 1 - e^{\lambda_2 \frac{|\hat{U}|}{|\hat{S}|}}$$

$$P_{\text{REMOVE}} = (1 - P_{\text{ADD}})\epsilon$$

$$P_{\text{SWAP}} = (1 - P_{\text{ADD}})(1 - \epsilon)$$

where $|\hat{S}|$ and $|\hat{U}|$ are the number of nodes and edges in $\hat{H}$, respectively, and λ_1 and λ_2 are user defined parameters that are set empirically to reduce the mixing time of the MCMC. In our experiments, $\lambda_1 = \lambda_2 = 0.1$.

A transition is either accepted or rejected by evaluating a cost function Γ_2:

$$\Gamma_2(\hat{H}) = \sum_{H_i \in \mathcal{H}} K(\hat{H}, H_i)$$

where K is a graph kernel. For this step we use K_{WL} as graph kernel, since it guarantees similar performance but at a significant lower computational cost than K_{GH}, and the cost function Γ_2 should be evaluated once for every iteration of the MCMC. The total number of iterations τ_2 for this step is selected to be a random number higher than 2000.

In the example of Fig. 7.8, the generated functional graph $\hat{H}$ is shown in Fig. 7.8B, where the numbers indicate the cluster of each subgraph (node of the functional graph).

Subgraphs sampling

In the third sampling step, a specific subgraph $\hat{s}_i \in C_i$ for each node of $\hat{H}$ is selected. This is done by picking with uniform probability a subgraph $\hat{s}_i$ from cluster C_i until $\hat{v}_i$ subgraphs are sampled from C_i. Note that a cluster C can contain several repetition of the same subgraph s, due to its repeated presence in the graphs $\mathcal{G}$. The subgraphs sampling step is performed in a single step and does not use the MCMC framework.

In the example of Fig. 7.8, the sampled subgraphs are listed in Fig. 7.8C. Boxes are highlighted using the same colors for nodes in Fig. 7.8A and B, and the cluster number is shown at the bottom left of each box.

Node connections sampling

The last step is the node connections sampling, where the final semantic map $\hat{G} = (\hat{N}, \hat{E})$ is obtained from the functional graph $\hat{H} = (\hat{S}, \hat{U})$. $\hat{N}$ is the set of all the nodes of the subgraphs $\hat{s}_i$ sampled in the previous step, while $\hat{E}$ is initialized with all the subgraphs edges (and with no edges between nodes of different subgraphs). Then, for each edge $\hat{u} = (\hat{s}_i, \hat{s}_j) \in \hat{U}$ in the functional graph $\hat{H}$ an edge $e = \{\hat{n}_k, \hat{n}_l\} \in \hat{E}$ between two nodes $\hat{n}_k \in \hat{s}_i$ and $\hat{n}_l \in \hat{s}_j$ should be created. Ideally, this step is the opposite of the graph segmentation procedure described in Section 7.2, and a new connection e should be added where a cut c would have been performed during segmentation.

Adding a connection e is a local edit operation on the graph $\hat{G}$ which results in a global change in the graph's layout and in its structural properties. Given a connection $\hat{u} = (\hat{s}_i, \hat{s}_j)$ between two subgraphs $\hat{s}_i$ and $\hat{s}_j$ defined in the functional graph, call $\mathbf{S}_{\hat{u}}$ the set of all possible pairs of nodes belonging to the two subgraphs:

$$\mathbf{S}_{\hat{u}} = \{(n, n') : n \in \hat{s}_i, n' \in \hat{s}_j\}$$

Each pair of nodes in $\mathbf{S}_{\hat{u}}$ is a potential candidate for adding a connection between the two subgraphs connected by $\hat{u}$ in the functional graph $\hat{H}$. Given a pair of nodes $(n, n') \in \mathbf{S}_{\hat{u}}$ and the local neighbourhoods of n and n' in $\hat{G}$, the function $\Phi : \mathbf{S}_{\hat{u}} \rightarrow [0, 1]$ represents the probability that a connection exists between the two nodes. Φ is computed as a product of different potentials, exploiting the local structure of the neighbours of the nodes:

$$\Phi(n, n') = \Xi(n, n') \cdot \Lambda(n) \cdot \Lambda(n') \cdot \Upsilon(\mathcal{L}(n), \mathcal{L}(n'))$$

The first potential, $\Xi(n, n)$, is based on the local neighbourhood of the candidate edge (n, n'). We define as $\mathbf{N}_\epsilon(e)$ the ϵ-neighbourhood of an edge $e = (n, n')$, which is a subgraph composed of the edge e, its two nodes n and n', and all the nodes and edges that are at distance ϵ from n or n' (distance is the number of hops). Given the set $\mathbf{C}$ of all cuts c performed on the original graphs in $\mathcal{G}$, we compute the set $\mathbf{Q}$ of all their 1-neighbourhoods, $\mathbf{Q} = \{\mathbf{N}_1(c) : c \in \mathbf{C}\}$. The potential Ξ is then computed as:

$$\Xi(n, n') = \frac{\sum\limits_{x \in \mathbf{Q}} k(d(\mathbf{N}_1(n, n'), x), \bar{d}_x)}{\sum\limits_{x \in \mathbf{Q}} \sum\limits_{y \in \mathbf{Q}} k(d(y, x), \bar{d}_x)}$$

where $\bar{d}_x = \frac{1}{|\mathbf{Q}|} \sum\limits_{y \in \mathbf{Q}} d(x, y)$, $d(x, y)$ is the distance between two graphs x and y computed using a graph kernel (K_{WL} in our case), and k is a Gaussian kernel $k(d', d'') = e^{-\frac{\|d' - d''\|^2}{2\sigma^2}}$ ($\sigma = 0.5$ in our experiments).
The potential $\Xi(n, n')$ indicates how a connection between two nodes n and n' reconstructs a pattern of nodes (centered in n and n') similar to those of nodes close to a cut c performed during the segmentation process. In this sense we want to enforce that the connection of two nodes n and n' should add an edge which can be a good candidate for being *cut* during a graph segmentation process as those described in Section 7.3. Since in our graphs (almost) all of the cuts $\mathbf{C}$ are performed on edges connecting two CORRIDORs (which are connected to a set of leaf-node ROOMs), 1-neighbourhood subgraphs can be used to effectively describe a cut c. For graphs representing different data and with an higher number of connections between nodes, higher ϵ-neighbourhoods can be used for generalizing the method to other domains.

The second potential, $\Lambda(n)$, is based on the *preferential attachment* principle introduced by [9] (and empirically observed in graphs representing indoor buildings in [7]), which assigns to each node n a probability of having

a connection that is proportional to its degree $D(n)$:

$$\Lambda(n) = \frac{D(n)}{\sum_{n'' \in \bar{N}_c} D(n'')}$$

where $\bar{N}_c$ is the set of the nodes that appear in a cut $c \in \mathbf{C}$. $\Lambda(n')$ is computed similarly.

The third (and last) potential function, $\Upsilon(\mathcal{L}(n), \mathcal{L}')$, is evaluated considering all the cuts in $\mathbf{C}$. Considering the nodes n and n' and their labels $\mathcal{L}(n)$ and $\mathcal{L}(n')$, Υ calculates the frequency of a cut $c \in \mathbf{C}$ between two nodes with the same labels.

The probability $\Phi(n, n')$ is then used to select, for each edge $\hat{u}$ between two subgraphs in a functional graph $\hat{H}$, which pair of nodes n, n' belonging to the two subgraphs should be connected. This step is performed using the MCMC algorithm. A transition kernel T_4 selects randomly an edge $\hat{u}$ from the set $\hat{U}$ of the functional graph $\hat{H}$ and selects randomly a pair of nodes from $\mathbf{S}_{\hat{u}}$ to be connected according to the following probability distribution:

$$P_{conn}(n, n') = \frac{\Phi(n, n')}{\sum_{\dot{n}, \dot{n}' \in \mathbf{S}_{\hat{u}}} \Phi(\dot{n}, \dot{n}')}$$

The transition obtained with this edit operation is accepted or rejected according to a cost function Γ_4:

$$\Gamma_4(\hat{G}) = \sum_{G \in \mathcal{G}} K(\hat{G}, G) \tag{7.4}$$

where K is a graph kernel. For this last step, the MCMC is run for more than 100 iterations, after starting from a random initialization of the connections between two nodes in $\mathbf{S}_{\hat{u}}$ for each $\hat{u} \in \hat{U}$ computed using P_{conn}. We used K_{WL} as graph kernel in (7.4), since it guarantees similar performance, but at a significant lower computational cost, than K_{GH}, and the cost function Γ_4 should be evaluated once for each iteration.

In the example of Fig. 7.8, the final sampled semantic map is shown in Fig. 7.8D. The edges added in the last sampling step are highlighted in red.

7.5 Evaluation of the sampled graphs

In this section, a quantitative evaluation of the results obtained by our method in sampling new instances of graphs is presented. The proposed

	$\mathcal{G}$	$\hat{\mathcal{G}}_{\texttt{corr}}$	$\hat{\mathcal{G}}_{\texttt{ncut}}$
nodes	35.24 (18.33)	40.57 (17.73)	39.75 (16.10)
nodes R	27.84 (15.05)	32.3 (14.40)	31.68 (13.72)
nodes C	7.42 (4.45)	8.37 (4.27)	8.07 (3.16)
path-length	3.33 (0.81)	3.43 (0.59)	3.36 (0.55)
diameter	6.34 (2.30)	6.62 (1.73)	6.38 (1.55)
art-points	8.8 (5.81)	10.04 (5.07)	10.10 (4.30)
assortativity	-0.51 (0.20)	-0.47 (0.14)	-0.44 (0.13)
betw-cen	0.039 (0.011)	0.036 (0.013)	0.036 (0.012)
betw-cen R	0.005 (0.006)	0.005 (0.005)	0.006 (0.006)
betw-cen C	0.181 (0.090)	0.160 (0.065)	0.150 (0.051)
closn-cen	0.328 (0.086)	0.309 (0.054)	0.316 (0.057)
closn-cen R	0.309 (0.077)	0.290 (0.048)	0.296 (0.050)
closn-cen C	0.418 (0.160)	0.384 (0.087)	0.393 (0.084)
eig-cen	0.256 (0.078)	0.239 (0.072)	0.241 (0.069)
eig-cen R	0.197 (0.062)	0.177 (0.056)	0.241 (0.053)
eig-cen C	0.498 (0.181)	0.486 (0.159)	0.502 (0.142)
katz-cen	0.181 (0.045)	0.166 (0.043)	0.166 (0.042)
katz-cen R	0.166 (0.045)	0.152 (0.041)	0.159 (0.040)
katz-cen C	0.241 (0.076)	0.222 (0.056)	0.221 (0.051)

Table 7.1: *Values of metrics for original graphs $\mathcal{G}$ and graphs generated with the proposed approach using* `corr` *and* `ncut` *segmentation ($\hat{\mathcal{G}}_{corr}$ and $\hat{\mathcal{G}}_{ncut}$, respectively). Entries report average μ over the graphs and standard deviation σ (in parenthesis). R means* ROOM *and C means* CORRIDOR *(see Section 4.3 for details on the labeling schema).*

method has been implemented[2] in MATLAB R2015b and executed on a i7QuadCore Intel 820M 16GB computer.

We use $\mathcal{G}$ with 50 labeled graphs (representing schools and introduced in Section 5.2) and we generate 200 sampled graphs $\hat{\mathcal{G}}$. We use either `ncut` or `corr` for segmentation and affinity propagation for clustering, together with K_{GH}. ($\hat{\mathcal{G}}_{\texttt{corr}}$ and $\hat{\mathcal{G}}_{\texttt{ncut}}$ contain 200 graphs each.) Sampling is performed using K_{WL}. From preliminary tests we observed that the segmentation method used impacts most on sampling (indirectly, via formed clusters) and, for this reason, we consider both the segmentation methods in our tests.

To assess the similarity between the original graphs $\mathcal{G}$ and the sampled graphs $\hat{\mathcal{G}}$ we compare the average μ and standard deviation σ of the metrics introduced in Section 5.1 for evaluating the graph structure and computed on both sets. This allows us to check whether the structure of the generated graphs is consistent with that of the real buildings present in $\mathcal{G}$.

[2]Code is available upon request.

graph kernel	step 1	step 2	step 3	step 4
K_{WL}	10^{-1}	10^{-1}	10^{-3}	10
K_{GH}	10^{-1}	10	10^{-3}	10^{2}

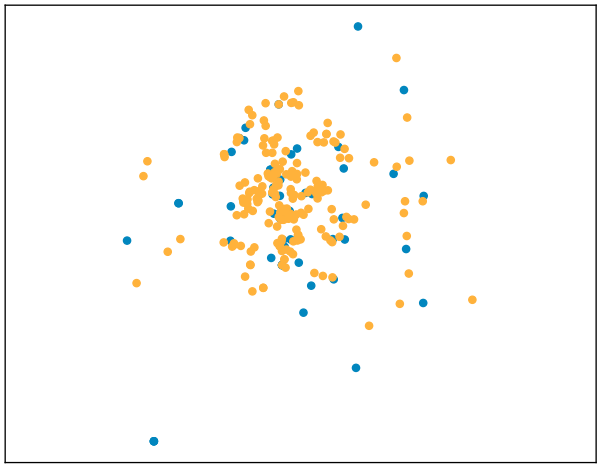
Table 7.2: *Computing time (order of magnitude, in seconds) for each sampling step. Step 1 indicates cluster configuration sampling, step 2 indicates functional graph sampling, step 3 indicates subgraphs sampling, and step 4 indicates node connections sampling. For steps 1, 2, and 4 the time is intended for a single iteration of the MCMC method explained in Section 7.4. Steps 1 and 3 do not use graph kernels.*

Table 7.1 shows that, for all the metrics, the generated graphs $\hat{\mathcal{G}}$ are coherent with the original ones in $\mathcal{G}$, in the sense that no statistically significant difference can be noticed for all the metrics. The results show that our method can sample new graphs with a similar structure as the original ones. Node importance in graphs $\hat{\mathcal{G}}$ is consistent with the node importance of the real graphs. It is important to point out that centrality measures are relative to the entire structure of the graph and are not dependent on any local pattern nor on single nodes. Similar centrality measures for $\mathcal{G}$ and $\hat{\mathcal{G}}$ mean that they actually contain similar graphs. Moreover, the different values of all four centrality metrics (although more evident for betw-cen) for nodes ROOM and CORRIDOR (with low and high values, respectively) indicates that also the roles of the nodes are distributed similarly over $\mathcal{G}$ and $\hat{\mathcal{G}}$.

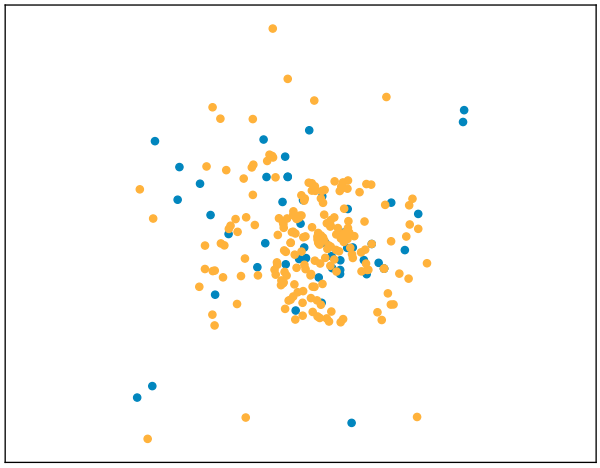
Figure 7.9 displays a visual representation of how graphs in $\mathcal{G}$ and $\hat{\mathcal{G}}$ are distributed, for both segmentation techniques `corr` and `ncut`. Using K_{WL} we compute the similarity between each graph in $\bar{\mathcal{G}} = \mathcal{G} \cup \hat{\mathcal{G}}$, obtaining a $\left|\bar{\mathcal{G}}\right| \times \left|\bar{\mathcal{G}}\right|$ similarity matrix. Data are reduced to a two-dimensional feature space using the same feature reduction algorithm illustrated in Section 7.2, t-SNE. In both Figure 7.9a and Figure 7.9b the sampled graphs (in orange) and the original ones (in blue) are similarly distributed, thus providing a visual confirmation that the two graph sets contain similar graphs.

Table 7.2 reports computing times for the sampling of graphs using our approach. It emerges that node connections sampling is the most expensive operation, since, for every iteration of the method and for every connection in the functional graph between two subgraphs $\hat{s}_i$, $\hat{s}_j$, it computes a score for each possible connections for every possible pair of nodes $\hat{n}_k \in \hat{s}_i$ and $\hat{n}_l \in \hat{s}_j$.

Sampling results obtained using `ncut` as segmentation method are sim-



(a) `corr`.



(b) `ncut`.

Figure 7.9: *2D representation of graph distributions for graph in $\mathcal{G}$ (blue) and in $\hat{\mathcal{G}}$ (orange) using t-SNE. Points are scaled to the $[0, 1]$ range for plotting.*

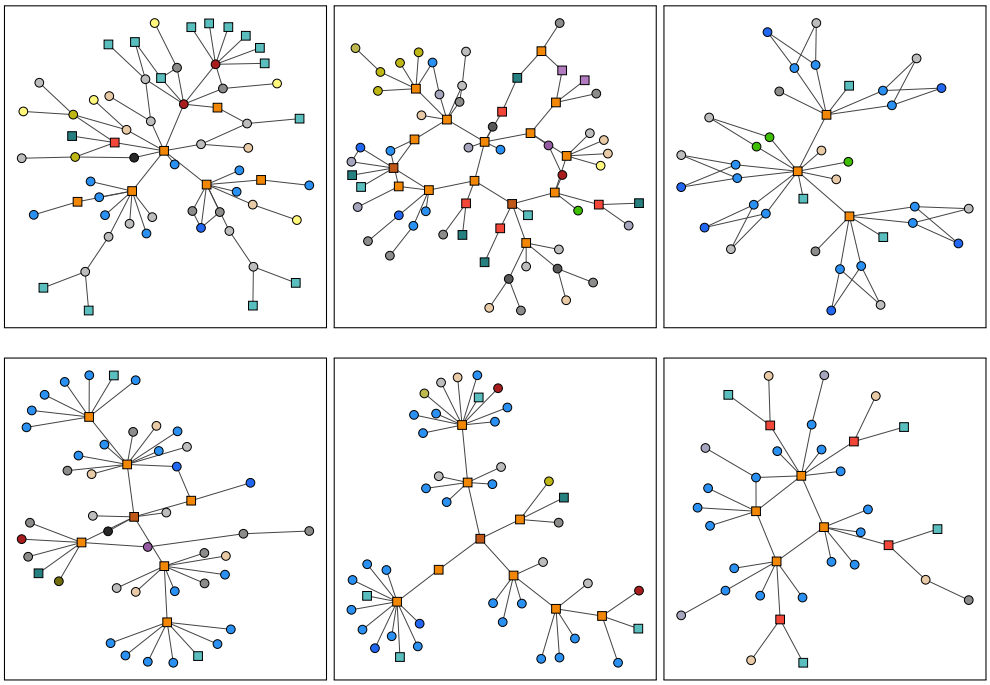


Figure 7.10: *Six graphs representing real world floor plans of schools from $\mathcal{G}$.*

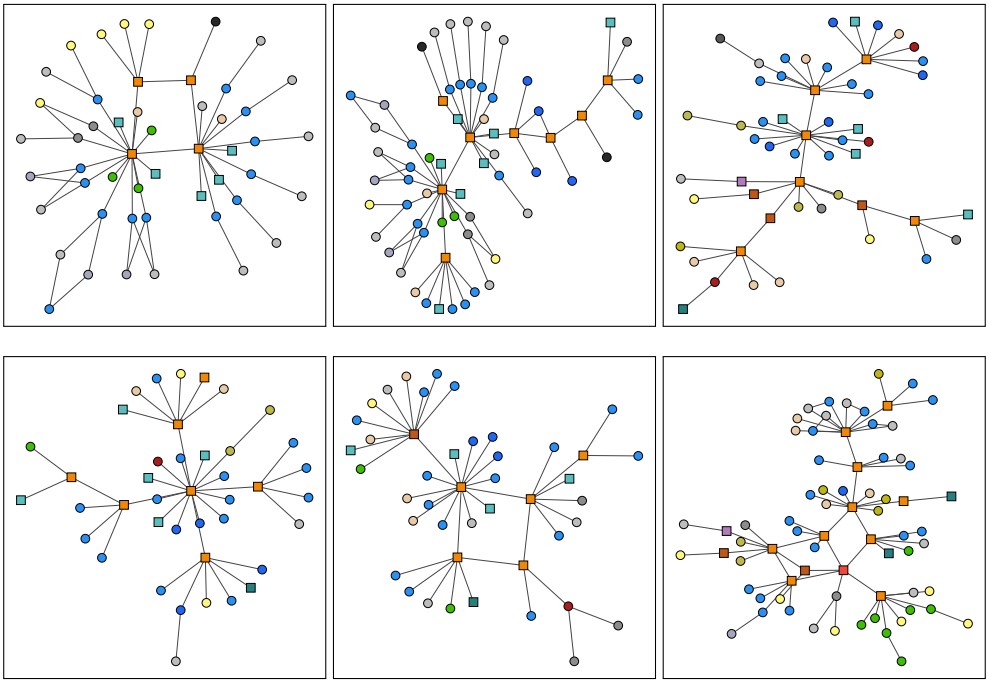


Figure 7.11: *Six graphs of $\hat{\mathcal{G}}_{\texttt{corr}}$ sampled with our method.*

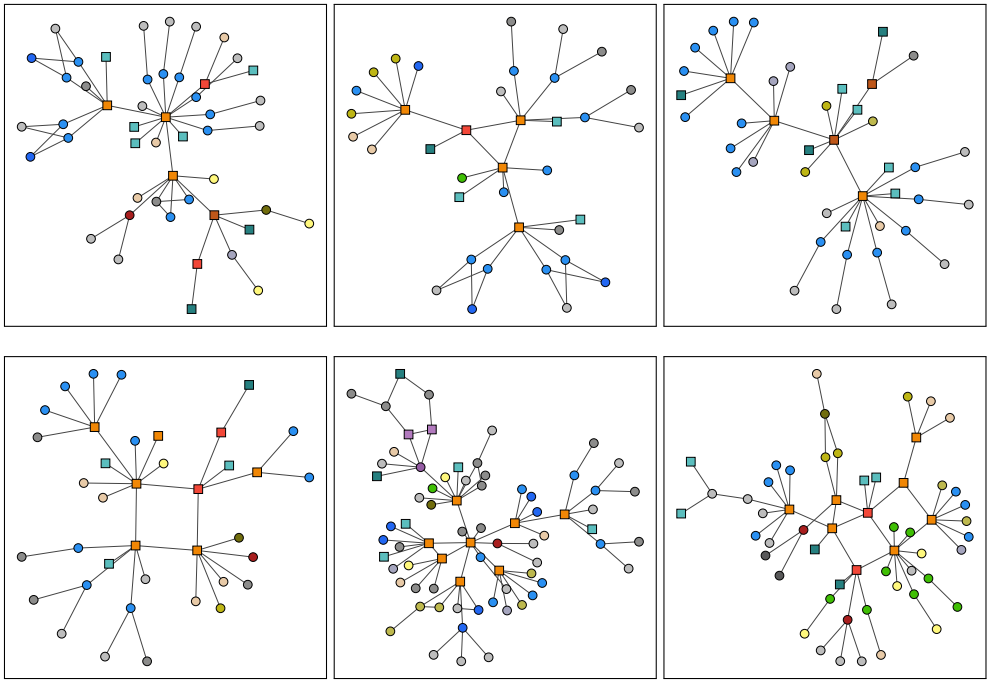


Figure 7.12: *Six graphs of $\hat{\mathcal{G}}_{ncut}$ sampled with our method.*

ilar to those obtained using `corr`. However, when the performed task is to sample new graphs from scratch, it is advisable to use `ncut`, since it guarantees to divide automatically the graph into subgraphs of similar size and perform cuts on different types of nodes independently of their semantics, thus providing more variety in the results when reconnecting the nodes. Segmentation using `corr` is designed to facilitate the recognition of a subgraph by a robot during the exploration of a building, and it should be selected when our method is used to predict the structure of a partially visited building given a partial semantic map. An example of this application is provided in the next chapter.

In order to provide also a qualitative evaluation of our sampled graphs, we report a set of example graphs extracted from data set $\mathcal{G}$, in Figure 7.10 and, for comparison, some sampled graphs randomly selected from $\hat{\mathcal{G}}_{\texttt{corr}}$ (Figure 7.11) and from $\hat{\mathcal{G}}_{\texttt{ncut}}$ (Figure 7.12). Real floor plan graphs of $\mathcal{G}$ have a similar structure and a similar label set when compared to those in $\hat{\mathcal{G}}_{\texttt{corr}}$ and $\hat{\mathcal{G}}_{\texttt{ncut}}$.

7.6 Discussion

This chapter has presented an approach that builds a generative model of graphs representing the topological structure and the semantic labeling of indoor environments and that uses the model to create new instances of (parts of) buildings. The three main steps of the proposed approach, namely segmentation, clustering, and sampling, are based on the use of graph kernels to assess similarity between graphs and on a hierarchical MCMC sampling algorithm. The experimental validation shows that the generated buildings, although not identical, share many features with the original buildings used to create the model and, significantly, have similar structures. The presented method allows us to sample semantic maps structurally similar to those of real world buildings belonging to a building type. In the next chapter we propose an application of this setting to partially visited environments.

CHAPTER 8

Applications of the proposed approach

In this thesis we have presented a framework for reasoning on buildings by considering their structure and by using as source of knowledge data sets of indoor environments extracted from floor plans of buildings. In Chapter 3 we show how it is possible to reconstruct the layout of indoor environments from a 2D grid map obtained by a robot. Chapter 4 and Chapter 5 introduce the basic of a framework for reasoning on indoor environments that considers each building in relation to those with the same function, its building type, and to that represents the structure of a building using a graph. In Chapter 6 we show how it is possible to reason on the entire structure of building by considering it as a single entity using Statistical Relational Learning techniques. In Chapter 7 we propose a generative model for building types, which is then used for sampling new graphs belonging to a building type.

In this chapter we try to put together all the above contributions and give some ideas about their practical employment. The applications described here are not exhaustive of the possibilities of our approach, but represent a significant sample of possible uses.

The first application is presented in Section 8.1, where we discuss how we can assess the validity of *simulation* tools by reasoning on the structure

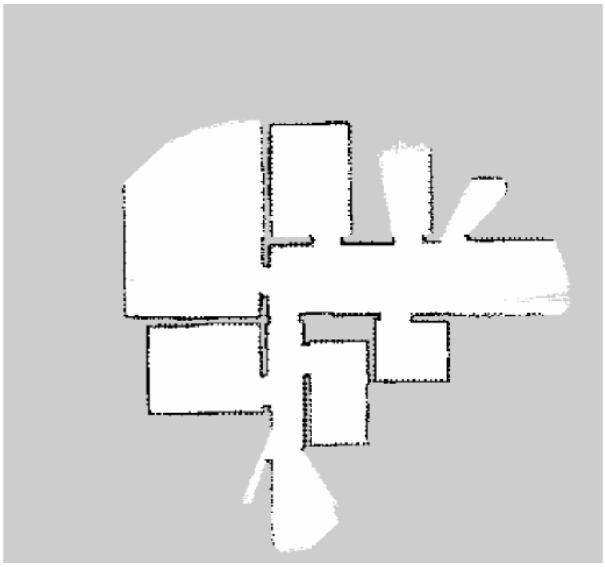


Figure 8.1: *A partially explored map. Pixel are colored accordingly to the probability of the occupancy of each cell in the grid map. Dark colors correspond to high probability of occupation. Frontiers are transition between free space (white) and unexplored space (gray).*

of the simulated environments, using the method from Section 6. We focus our attention on simulated indoor environments, considering whether or not the structure of a simulated floor plan of an environment is similar to those of real world environments of the same type.

The second and third applications regard reasoning about partially explored maps. Robots, during exploration, incrementally build the map of an initially unknown environment. A *partial map* is the map known at a given time to the robot. Exploration is usually performed by selecting and exploring the more interesting and near frontiers. An example can be seen in Figure 8.1.

In Section 8.2 we show how we can use our layout reconstruction method presented in Chapter 3 for roughly predicting the structure of partially explored rooms. This is done by identifying partially explored rooms and by using *faces* and *representative lines* identified from the maps (see Section 3.1).

Finally, in Section 8.3 we show how we can combine knowledge on the explored part of a building with knowledge on its building type by applying the generative model of Chapter 7 to predict the whole structure of a partially explored building from which we extract a semantic map repre-

senting the explored portion, which is used as seed for the sampling process described in Section 7.4.

As said, these three applications are intended to provide preliminary examples of the possible applications of the approach described in this thesis.

8.1 Evaluation of simulated environments

Simulations are a powerful tool for robotics because, potentially, they provide the designers of the experiments full control over all the variables of the settings [5]. This allows researches to tackle specific problems while reducing the sources of uncertainty that are usually present in the real world. The simulation tools could also be shared and used by many different authors, facilitating comparison and reproducibility of the results. Accurate modeling of the real-time interaction between robots and environments is difficult, but some simulation tools have been shown to do it reliably [21], avoiding the pitfalls of naïve robot simulations highlighted in [50], where carefully validated simulations with a proper treatment of noise are shown to overcome these problems.

A simulated experimental setting, in order to provide results that are generalizable to real world scenarios, has to posses some of the features of real environments; in other words a simulated experimental setting has to be *representative* of a real world experimental setting. We could say that a particular experimental setting is as *representative* as much as its *features* are close to those of the class of settings where robots can operate. A feature is a distinctive characteristic of an environment, such as the presence of a loop of corridors in an indoor environment. The identification of the features of experimental settings and of the metrics to measure their similarity, which can be used to precisely define representativeness, is a largely open issue that depends on the specific areas of autonomous robotics. An analysis of the concept of representativeness applied to robotics can be find in our work [4], from which part of this section is taken.

While in most simulations, the physical realism of the environment (e.g., the precise reproduction of the physical laws) is carefully taken into account, the structural plausibility (e.g., if the environment "looks like" its real counterpart) is not considered. This fact reduces the generalizability and the effectiveness of experimental results obtained with simulation, and is one of the facts that limit the diffusion of simulations as a valid experimental tool.

The above limitation can be addressed using an approach similar to that we have introduced in [4]: the design phase of experiments in simulation

must take in account *what* the simulated environment represents, *why* it is chosen, and *how* it relates to its real counterparts. Note that a simulated environment designed starting from real-world representative data sets can easily satisfy the representativeness requirements.

8.1.1 Simulation in robotics

In order to evaluate how simulations are used in robotics and what kind of simulated environments are used, we have considered 57 papers labeled with the keyword 'SLAM' in the proceedings of the IEEE International Conference on Robotics and Automation (ICRA) 2014[1].

ICRA is one of the main conferences on robotics and, as such, can be assumed to present a comprehensive picture of the current research on autonomous robots. Papers about SLAM (Simultaneous Localization And Mapping) are particularly fit to discuss the degree of representativeness of the environments used in experiments, because they show methods for building representations of physical spaces starting from sensorial data (both about the world and about the pose of the robots). Papers about SLAM are well represented at ICRA (as at the other main conferences on robotics), because the field has been investigated for years and is now considered mature, having developed a solid background of methods. At the same time, interesting research is still going on in SLAM, and new methods and experimental approaches are continuously proposed.

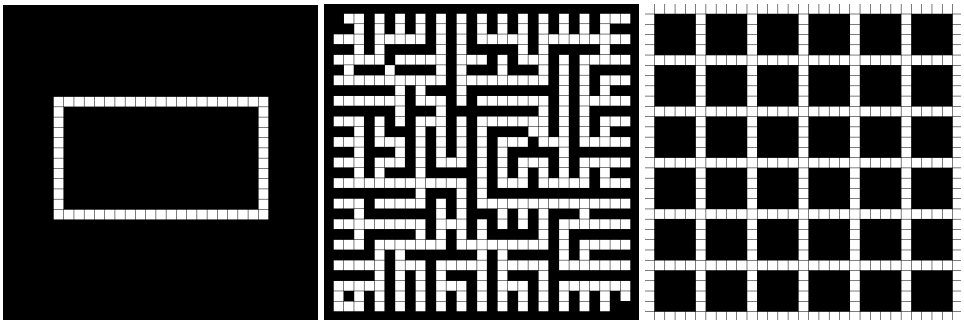
The 57 papers we analyzed are rather heterogeneous and employ different robot platforms (wheeled mobile robots, aerial robots, self-driving cars, ...) equipped with a wide range of sensors (RGBD and traditional cameras, lidar, GPS, ...). For our survey, we considered only 55 papers. The two excluded papers are about the proposal of a data set for benchmarking 3D SLAM methods based on RGBD cameras [44] and about the comparison between some SLAM methods and some topography approaches developed in the XIX Century [1], respectively, and are thus out of the scope of our investigation.

Table 8.1 reports some of the results we have collected from the 55 paper analysed. Between testing the methods *online* on real robot platforms that move in real environments (or on simulated platforms that move in virtual environments) and testing them *offline* using data previously collected, there is a clear preference for the second option, which is adopted by 83.6% of the papers we considered. Although the two are not mutually exclusive,

[1]Papers are available at `http://ieeexplore.ieee.org/xpl/mostRecentIssue.jsp?punumber=6895053`, while the keywords are shown at `https://ras.papercept.net/conferences/conferences/ICRA14/program/ICRA14_KeywordIndexWeb.html`.

		yes (%)		**no** (%)	
offline	public data sets	46(83.6)	26(47.2)	8(14.6)	28(50.9)
	own-collected data sets		35(63.4)		19(34.5)
online	real-world	22(40.0)	8(14.5)	33(60.0)	47(85.5)
	simulations		15(27.2)		40(72.7)

Table 8.1: *Types of experimental settings considered in our sample of ICRA papers: number of papers and, in parenthesis, the relative percentage (note that a single paper can consider many settings).*



(a) **(b)** **(c)**

Figure 8.2: *Three examples of non-realistic simulated environments: (a) a loop, (b) a labyrinth, and (c) a regular grid. White cells are free space, while black cells are obstacles.*

most of the papers (75%) we analyzed tend to adopt only one approach (either online or offline). In particular, the use of data sets specifically collected for the purpose of the paper is preferred by the surveyed papers over the use of publicly available data sets in offline evaluation.

In online evaluation, simulations are preferred over real-world experiments. Approximatively the 70% of papers with online evaluation presented results obtained via simulations.

By looking at the simulated environments used in the sample of papers we analyzed we can notice the following attitude of the authors towards simulation settings. Authors acquire and select data sets more on the basis of convenience than on a careful analysis of the main characteristics of the environments. When designing simulated environments, authors tend to select the *key* features they want to represent and to create simple simulated environments where these features are presented, keeping the rest as essential as possible.

Examples of this trend include the following situations.

- If the simulation is intended to test the ability of a SLAM algorithm to close loops, the environment is a rectangle of four connected corridors, as in Figure 8.2a.

- If the simulation is intended to test the ability of a robot to navigate in an intricate context, the environment is a labyrinth, as in Figure 8.2b.

- If the simulation is intended to test the behavior of a robot in a large-scale scenario, a huge Manhattan grid is created and used as environment, as in Figure 8.2c.

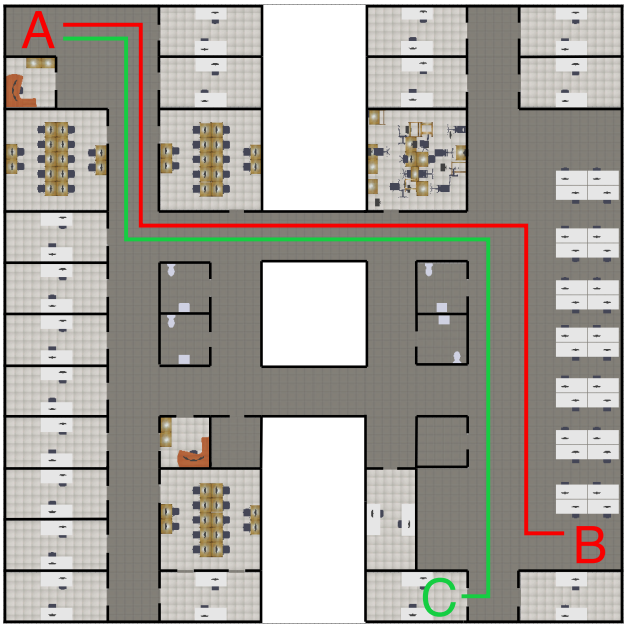
Only in the 21.5% of the papers we surveyed the used simulated environments actually represent real environments. In all the other cases, the simulated environments are synthetically generated, similarly to those presented in Figure 8.2. In practice, only some real environmental features are present in otherwise unrealistic contexts (like a table with a manipulator floating in an empty space, some boxes and wood walls in an empty room, ...). To sum up, from our analysis emerged how the realism of simulated environments is rarely taken into account during the set up of an experimental setting, where are often preferred synthetic environments with little resemblance with their real world counterparts.

8.1.2 Validation of simulated worlds

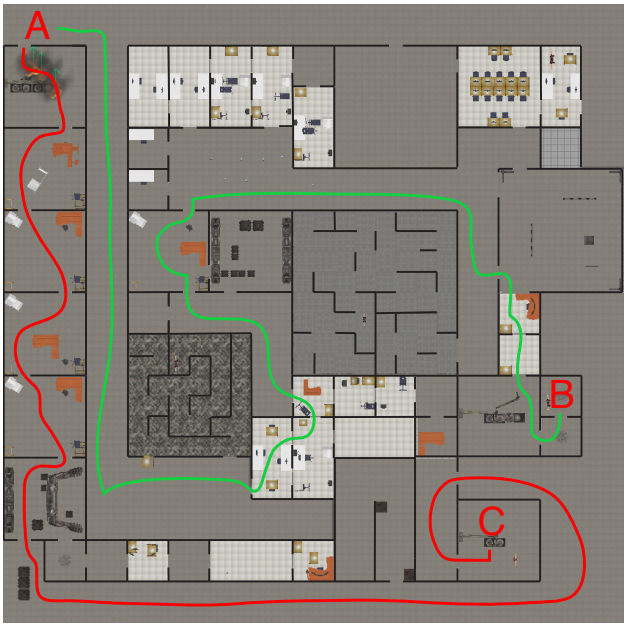
For some topics, including object-recognition, recognition and tracking of people, and recognition of human movement, simulation could be poorly representative and real-world data are needed for experimental assessment of methods. However, there are situations for which the structure of the environment is more important than the visual details (e.g., the presence of many rooms and of a large-scale environment is more important than the quality of textures used for the doors and of the 3D model of a chair). An example is coordinated multirobot exploration [17, 24].

Figure 8.3 shows two different experimental settings developed for testing multirobot exploration in a USAR (Urban Search and Rescue) scenario. These environments have been developed for the RoboCup Rescue Simulation Virtual Robot Competition and have been used in the 2012 and 2014 edition of the event, respectively[2]. The two maps share many features: both are developed for the same simulation tool (USARSim), both are designed

[2]Both maps are available, with some details about their usage, the performance obtained in these settings, and rules of the competition, at `http://www.robocuprescue.org/wiki/index.php?title=VRCompetitions`.



(a)



(b)

Figure 8.3: *A simulated office building generated using a real world floor plan as source (a) and a simulated office building generated without using real-world data as inspiration (b). Black line segments are walls, while traversable areas (offices, corridors, . . .) are represented with different colors according to the textures used for displaying them in the simulator.*

for the same purpose (the Virtual Robot Competition), and both represent the same kind of environment (a floor of an office building).

Nevertheless, the two maps are different, since one is structurally very similar to a real-world building (Figure 8.3a), while the other one is not (Figure 8.3b). To highlight this difference consider the shortest paths leading from one side to the other side of each map, starting from the top-left corner (A) to two close locations at the bottom-right corner (B) and (C). The paths in the realistic building of Figure 8.3a are more linear, since an office building is usually designed to facilitate the movement and the work of humans. Instead, in Figure 8.3b, the paths from A to B and from A to C are very complex, with several turns, and follow completely different routes than the path in Figure 8.3a. In this sense, the first environment (which is developed by us using real world buildings from OFFICE as source) is more structurally plausible than the second one (which is developed using an automatic procedure for creating a layout of a building from combination of a set of template rooms, as explained in [105]), at least considering modern-style buildings.

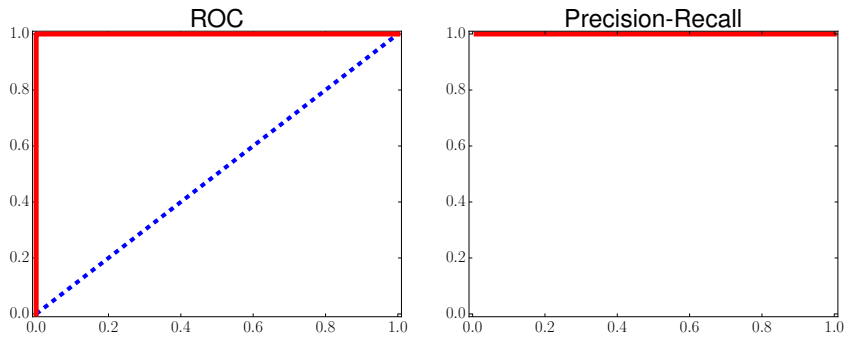


Figure 8.4: *Validation of structural realism of simulated worlds using kLog.*

In this section we propose a method, based on our SRL approach, to test if a simulated environment presents the same structural characteristics of a similar real-world environment. In the context of our approach, this can be done by evaluating if the topological and geometrical properties of spaces of a (floor of a) simulated building are coherent to those of (floors of a) real-world buildings of the same type. In this way, we can assess if a simulated environment is *structurally realistic*.

For this purpose we use the same experimental setting and methodology presented in Chapter 6 and explained in detail in Section 6.1. More pre-

cisely, we represent the structure of the building using a relational representation using a Prolog syntax, which is then provided to kLog [34], which performs classification considering the entire structure of the building as a single entity. Results are compared with those obtained by a standard attribute-value classifier which uses a local approach, Extra-Trees. Information on the structure of the environment is given to Extra-Trees using the two centrality measures. See Chapter 6 for more details of this setup.

We build a data set of all the office environments used during the finals of the 2012 and 2013 editions of the Virtual Robot Competition of the RoboCup Rescue Simulation League [82], which represent 6 simulated environments, designed to test the behavior of robots in an Urban Search and Rescue task. (Note that we removed from the set of RoboCup environments the one of Figure 8.3a, since it was developed by us.) In kLog and Extra-Trees, validation of such data set can be represented as a binary classification of interpretations (similar to a building classification), in which each interpretation is categorized as a positive instance of OFFICE or as negative example of an unrealistic simulation, using as input data all the interpretations of OFFICE and of RoboCup simulated offices combined together. Since Extra-Trees do not allow structured prediction, we consider the most frequent label, as for the building classification task (see Section 6.1). Figure 8.4 shows that kLog is always able to distinguish a real-world office building from an unrealistic office building simulated in the competition, outperforming Extra-Trees, as shown in Table 8.2. The results of Figure 8.4 are obtained using the label schema $\mathcal{L}_{R/C}$, but similar results are obtained using the more descriptive labeling schemas $\mathcal{L}_{F/C/E/S}$. Comparison of these results to the ones obtained by Extra-Trees (Table 8.2) highlights the limitation of using poor semantic information for different structural data but with similar local features.

These findings suggest that the structural realism of environments used in the Virtual Robot Competition could be improved in order to make the transfer of the results from simulation to the real-world easier. More generally, our system can be used in other situations to distinguish realistic from unrealistic simulated environments, thus providing a method for *validating* their suitability as experimental testbeds.

8.2 Predicting the layout of partially explored rooms

In this section we propose a method for estimating the layout of a partially explored room, given the rest of the environment.

Our method follows the approach used for reconstructing the layout of

	data set	labels	centrality	error (%)
Ex-T	R + O	RC	○	16.67
Ex-T	R + O	RC	●	8.33
kLog	R + O	RC		**0.00**
Ex-T	R + O	$FCES$	○	8.33
Ex-T	R + O	$FCES$	●	**0.00**
kLog	R + O	$FCES$		**0.00**

Table 8.2: *Results on simulated data set validation using kLog and Extra-Trees (Ex-T). R represents the* ROBOCUP *data set, and O is the* OFFICE *data set. The labels column represents the label set used, the centrality column indicates whether centrality is added (full circle) or not (empty circle) to V_n, while the error column indicates the classification error with the best performance in bold.*

an environment explained in Chapter 3 and exploits the insight that buildings present similarities and symmetries between different rooms. The structure of the building is identified by its set of walls. In Section 3.1 we describe how we can identify walls and display them as *representative lines*. Our method follows the one presented in Section 3.1 and it is described by using the running example in Figure 8.5.

We start from a metric map M representing a partial (grid) map of an indoor environment, as the one of Figure 8.5a. As explained in Section 3.1 we use Canny Edge detection and Hough Line transform to detect the line segments present in the map (Figure 8.5b). After the identification of the contour (Figure 8.5c), we apply our layout reconstruction method obtaining a set of representative lines and a set of faces F (Figure 8.5d-8.5e).

All faces F are classified as *internal*, *external* or *partial*. Specifically, faces called external are discarded. External faces are those with an intersection with the inner area of M (obtained from the contour as seen in Figure 8.5c) smaller than a threshold σ. Remaining faces are called partial if they are adjacent to an external face via an edge whose weight is less than a threshold (0.2 in our experiments) and internal otherwise. Partial faces cover the area of rooms that are not fully known according to the data collected in M, as for example during the exploration of a building.

After this classification step, faces are clustered together using DBSCAN , eventually obtaining the reconstructed layout R.

We classify as partial all the rooms that are composed by at least one partial face. Using this method we use faces to approximate the layout of partial rooms by considering all the faces that are assigned to that room as its layout. Although simple, this method can provide a initial guess of the

shape of partially explored rooms based both on the grid map and on the walls identified in other rooms of the environment.

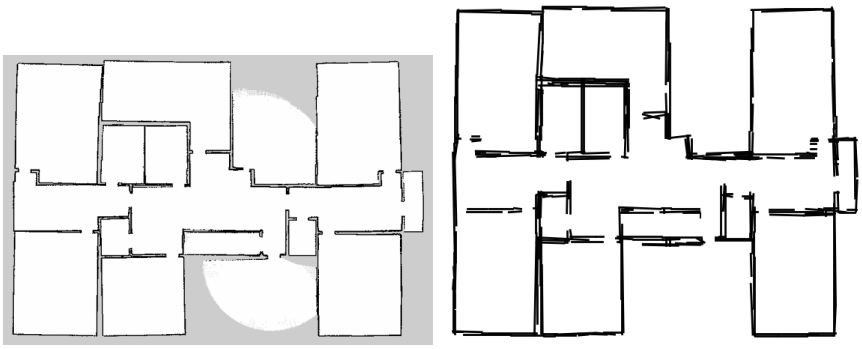
Figure 8.6 and Figure 8.7 shows six examples of layout reconstruction starting from partial grid maps. Partially explored rooms are identified correctly and marked in gray in the layout. The faces composing a partial explored rooms are added to the reconstructed layout, providing a guess of the unknown room appearance. These results suggest that our method can be used for predicting the shape of partially seen rooms, on the basis of the layout reconstructed from of the rest of the building. Information about the possible shape of partially seen rooms can be valuable for speeding up exploration and for providing knowledge to predict the semantic label of rooms before they have been fully explored.

8.3 Predicting the structure of partially explored buildings

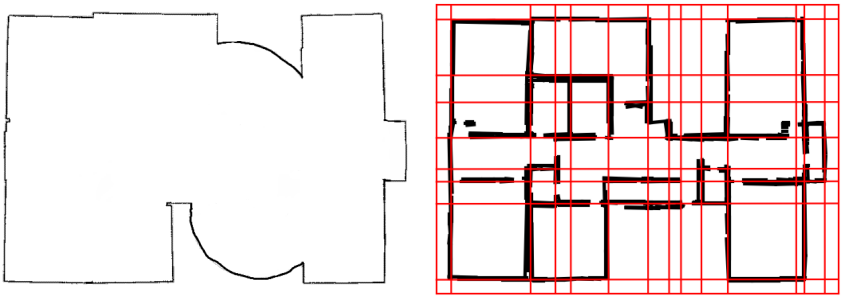
As discussed in Section 2.1 of the methods for place classification follow an approach that starts from the data perceived by the sensors mounted on-board mobile robots (e.g., laser range scanners and cameras), extracts some features from these data, and classifies the area from which the data have been acquired using (supervised) machine learning techniques. This approach has demonstrated to be very effective in labeling parts of environments already visited by the robots, but usually does not address the problem on inferring new knowledge on the labels and, more generally, on the structure of *unvisited* parts of environments (apart from some remarkable examples, as [77]), which are then considered as completely unknown until they are visited.

Availability of semantic knowledge about unvisited portions of environments can be useful for online planning tasks, like exploration and search. [8] shows how search methods could be improved using a probabilistic model of the search environment able to perform place classification and make local predictions on the neighbouring unexplored spaces. [72] shows how the knowledge of a rough topo-metric graph of the environment improves the exploration performance in an otherwise unknown environment. In [79] a correct prediction of the labeling of the unexplored parts of an environment is shown to improve the exploration performance of a team of robots. Finally, [73] shows that a prediction of the structure of the environment based on previously acquired maps can be used for better exploring challenging and repetitive environments by facilitating loop closures.

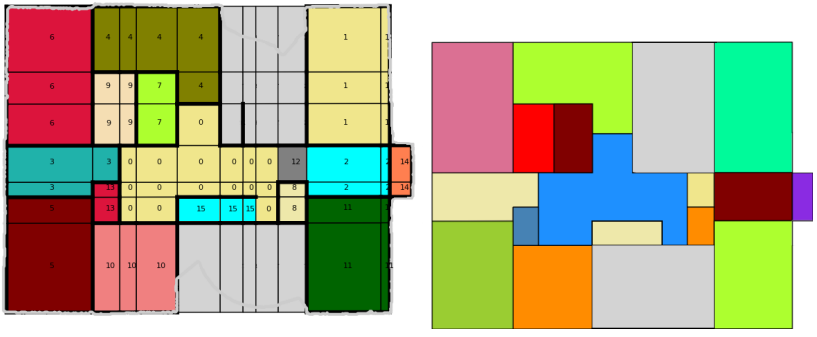
While the above studies show that an (approximate) knowledge on the unexplored space is useful for many online planning applications, meth-



(a) *Metric map.* **(b)** *Hough edge transform.*



(c) *Map Contour.* **(d)** *Representative lines and faces contour.*



(e) *Clustering faces using DBSCAN.* **(f)** *Reconstructed partial layout.*

Figure 8.5: *An example run of our method for predicting the layout of partially explored rooms. Partial rooms are represented in gray.*

Figure 8.6: *Examples of layout reconstruction from partial metric maps (partially explored rooms are in gray).*

Figure 8.7: *Examples of layout reconstruction from partial metric maps (partially explored rooms are in gray).*

ods for obtaining such knowledge are still largely missing. The approach proposed by [77] and by [8] consists in performing *local* predictions, i.e., probabilistic predictions of the labels of the unvisited rooms *directly connected* to the places already visited (and already semantically labeled) by a robot. The approach of [73], instead, proposes a method for predicting the structure of unknown parts of large environments by matching the explored part with previously acquired maps of similar environments.

In this section we adapt the generative model presented in Chapter 7 to the task of predicting the topology and semantic labels of an environment given those of a set of explored rooms. Although our approach may not find an *exact* prediction of the structure of an unknown environment, it is nevertheless able to capture some of its fundamental structural properties (that can be useful to perform several tasks, as discussed above).

We envisage a perspective applications of the proposed approach, in (multirobot) exploration, in which the proposed generative model can be used to predict the structure and the labeling of the unexplored parts of a partially known environment, in order to speed up the exploration, for example by better coordinating robots [79].

Consider a setting in which an initially unknown (floor of a) building is being explored by one or more robots. We assume the following setting for exploration:

1. exploration starts from one of the rooms labeled as ENTRANCE, ELEVATOR, or STAIRS;
2. the robots move to the first CORRIDOR room connected to the entrance;
3. each room connected to the corridor is explored using a breadth-first like exploration approach; the semantic map is expanded accordingly with the correct semantic labels for the explored rooms;
4. after all the rooms connected to the first corridor have been explored the robots predict the structure of the unexplored part of the building;
5. the robots move to the nearest corridor discovered in step 3, until the entire building has been explored.

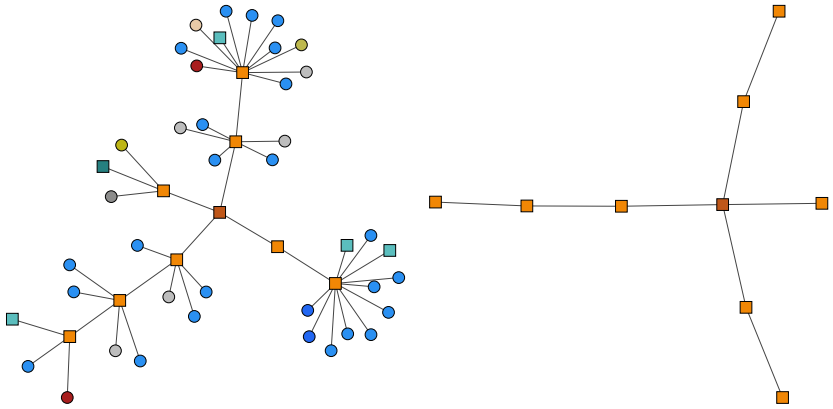
The sampling process depicted in Section 7.2 assumes that no *a priori* knowledge about the predicted building is available, but samples a completely new instance of a building from an empirical distribution. It is easy to adapt the process to the case in which some initial knowledge is available.

Specifically, assume that a portion of a building is known in form of some subgraphs $\{\bar{s}_1, \bar{s}_2, \ldots\}$ which are assigned to clusters $\{\bar{C}_1, \bar{C}_2, \ldots\}$. The cluster configuration sampling is thus performed as reported in Section 7.4.3, but considering the value of $\hat{v}_{\bar{C}_1}$ relative to $\bar{C}_1$ as bounded in the interval $[1, \max_{G \in \mathcal{G}}(v_i{}^G)]$. Similarly, for $\hat{v}_{\bar{C}_2}$ relative to $\bar{C}_2$ and so on. In the sampling of the functional graph, the known edges between $\{\bar{s}_1, \bar{s}_2, \ldots\}$ are not modified. Subgraphs $\{\bar{s}_1, \bar{s}_2, \ldots\}$ are considered fixed also during subgraphs sampling. Connection between known and unknown nodes are also preserved; for instance, if a corridor connected to five rooms has been explored and only three of them have been already explored, the two edges connected to the still unknown rooms are preserved and sampled rooms are connected to them. Finally, in sampling node connections, known connections between nodes of $\{\bar{s}_1, \bar{s}_2, \ldots\}$ are considered as fixed and are not modified during this last sampling step.

For this setup we use `corr` as segmentation method. This allows us to automatically detect the subgraphs during exploration, since every time step 3 above is executed the robots have explored a new subgraph (as explained in Section 7.2), and `corr` segments the graphs assuming that subgraphs are formed by a CORRIDOR and its neighbouring ROOMs. Under these premises, the structure of the building is predicted every time a new subgraph is explored. Prediction is tested on the same $\mathcal{G}$ composed of 50 graphs (representing school buildings) of Section 7.5 using leave-one-out (the explored graph used for prediction is removed from the database when training the clustering model).

In order to show how our approach can be applied for prediction in the above setting, we present in detail four examples. Given the importance of corridors for exploration (see, for instance, [79, 92]), we are particularly interested in predicting the structure of the corridors of the unvisited part of the environment. For each example, we display the original graph (representing the environment to be explored), its corridor structure, and the predicted graph and its corridor structure during an incremental exploration of the environment. The known (already explored) part of the original graph is highlighted with a grey overlay.

As a first example, consider the environment of Figure 8.8a, whose cross-shaped "skeleton" of corridors is shown in Figure 8.8b (refer to Figure 7.3 for the labels). Figs. 8.9a-8.9b show the predictions after exploration of the initial corridor and of the connected rooms. Figures 8.10-8.12 show three other predictions made after exploration of larger and larger parts of the environment. As expected, the more the initial knowledge used



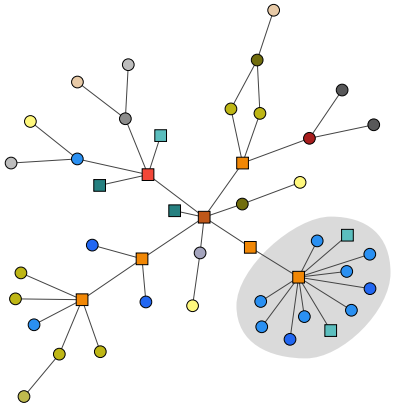
(a) *Real environment.* **(b)** *Corridor structure.*

Figure 8.8: *An environment being explored (first example).*

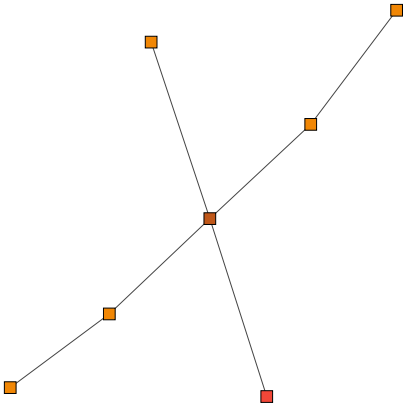
for sampling an environment, the more accurate the predictions when compared to the real environment of Figure 8.8. This holds especially for the corridor "skeleton" of the building, which is correctly predicted to be cross-shaped.

We also evaluate how well our approach can predict the labels distribution (i.e., the percentage of rooms with a specific label) in the unvisited part of the environment. Figs. 8.9c, 8.10c, 8.11c, and 8.12c show the labels distribution of the real graph of Figure 8.8 (in yellow), the average labels distribution for all the graphs in $\mathcal{G}$ (in green), and the predicted labels distribution at each step (in purple). Ideally, the purple bars should be more similar to the yellow ones than to the green ones, meaning that our approach is able to actually predict the distribution of labels for a given environment and does not simply return a "blind" prediction based on the average of labels distributions of the initial graphs. This is actually the case for the most relevant labels, especially when the exploration proceeds. For instance, see the corridors and the classrooms in Figure 8.11c and Figure 8.12c.

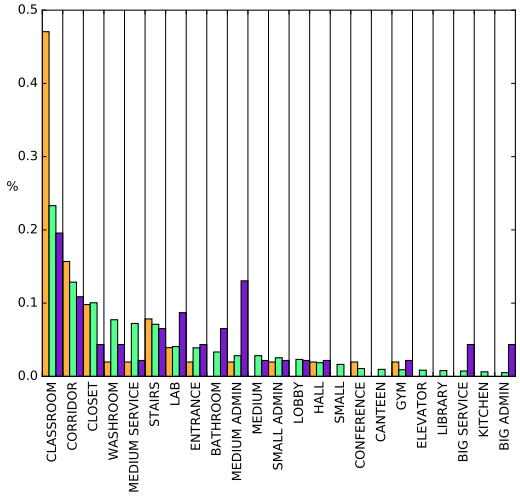
The second example, a building with several loops of corridors (two loops of corridors and a third one closed by a cafeteria), is shown in Figure 8.13 and in Figures 8.14-8.16. Also in this case, the loops of corridors that represents the "skeleton" of the environment are predicted correctly by our approach after step 1 of exploration. The other two examples can be seen in Figure 8.17 and Figures 8.18-8.19 (third example) and in Figure 8.20 and Figures 8.21-8.22 (fourth example). Note that all the above examples (and in several other tests that are not shown here), the prediction



(a) *Predicted semantic graph after step 1.*

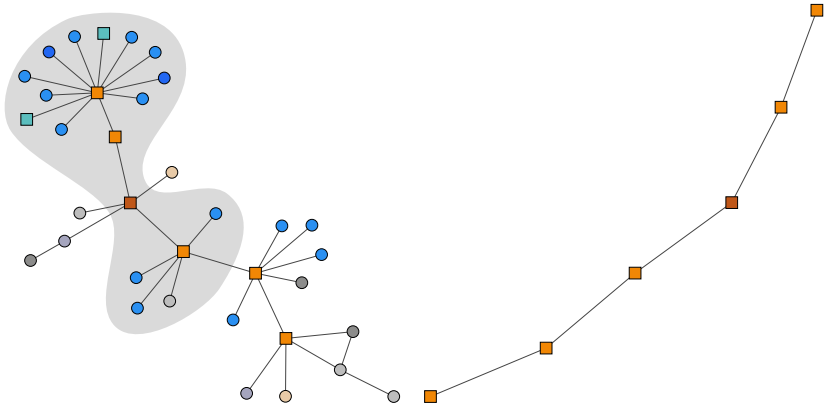


(b) *Predicted corridor structure after step 1.*

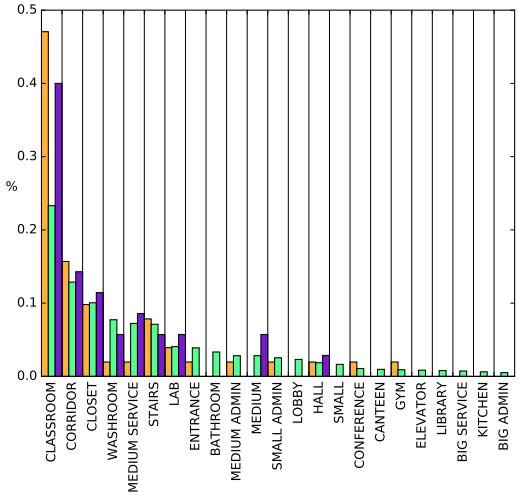


(c) *Labels distribution after step 1.*

Figure 8.9: *Predictions made after one of four steps of exploration for the environment of Figure 8.8.*

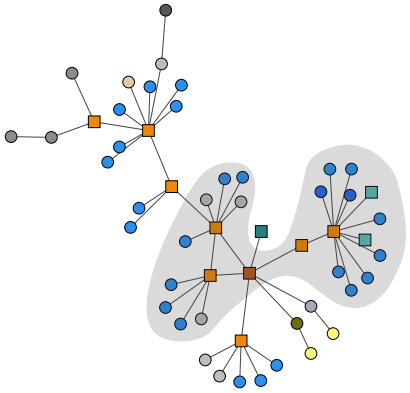


(a) *Predicted semantic graph after step 2.* **(b)** *Predicted corridor structure after step 2.*

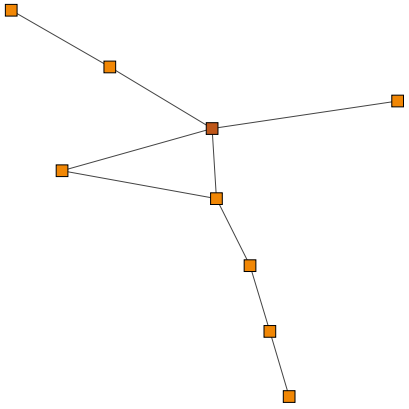


(c) *Labels distribution after step 2.*

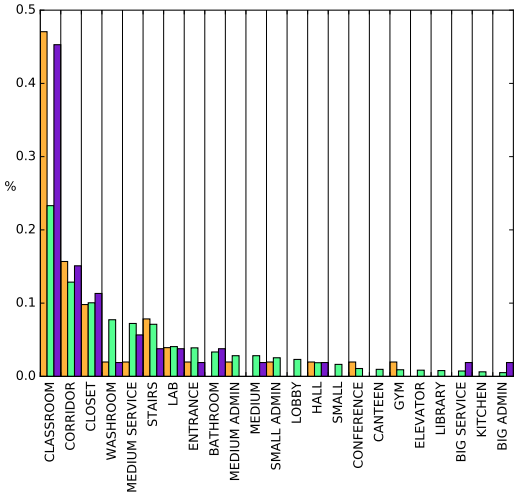
Figure 8.10: *Predictions made after two of four steps of exploration for the environment of Figure 8.8.*



(a) *Predicted semantic graph after step 3.*

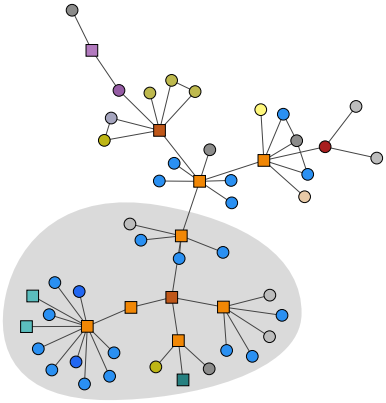


(b) *Predicted corridor structure after step 3.*

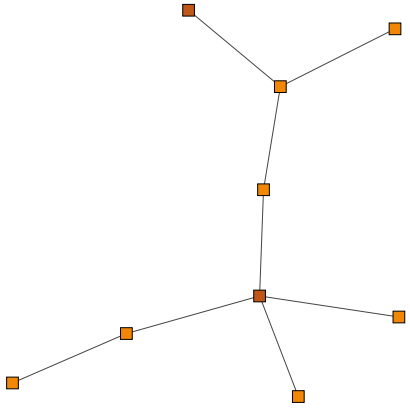


(c) *Labels distribution after step 3.*

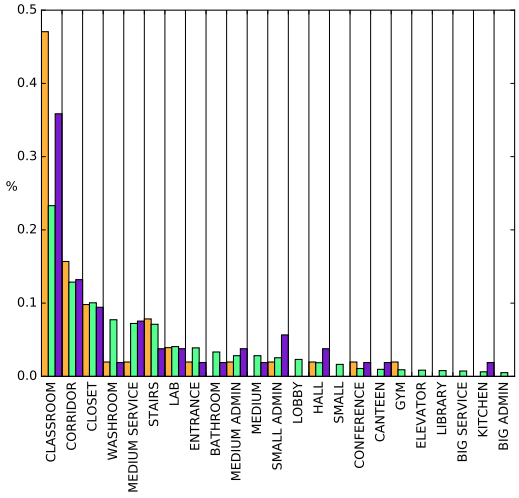
Figure 8.11: *Predictions made after three of four steps of exploration for the environment of Figure 8.8.*



(a) *Predicted semantic graph after step 4.*

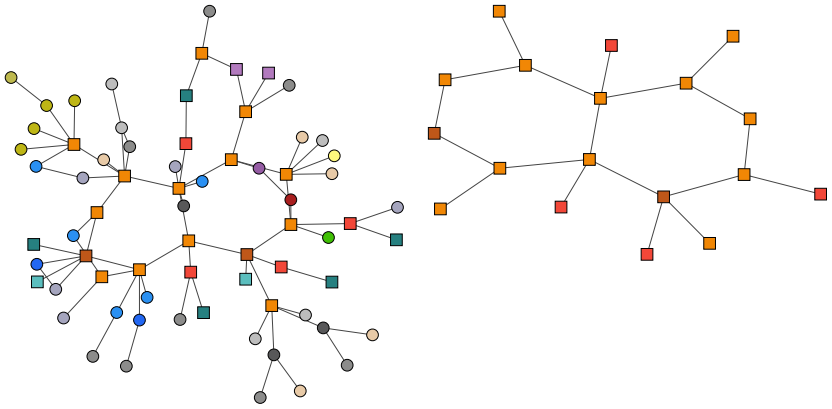


(b) *Predicted corridor structure after step 4.*



(c) *Labels distribution after step 4.*

Figure 8.12: *Predictions made after four steps of exploration for the environment of Figure 8.8.*



(a) *Real environment.* **(b)** *Corridor structure.*

Figure 8.13: *An environment being explored (second example).*

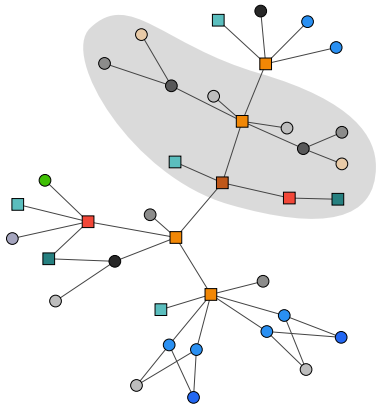
is not accurate with respect to the exact shape of the unvisited environment, but nevertheless it captures well the structure of the building. This is sound with the observation at the core of the idea of building type (see Chapter 4), namely that every school building is different but all school buildings share some similarities, which are actually captured by our model and approach.

Note finally that the kind of predictions made with our approach and discussed in this section cannot be attained with the methods performing local predictions (recall Chapter 2), because they can only predict the presence and the labels of nodes adjacent to the known nodes. Moreover, the predictions obtained with our approach are not limited to be equal to portions of the initial buildings in $\mathcal{G}$ (as in [7]), but are new compositions of the subgraphs of the initial buildings.

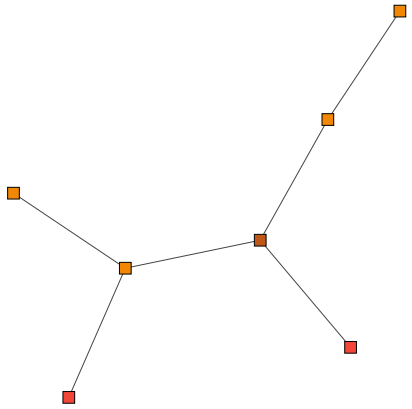
8.4 Discussion

In this section we have introduced three applications of our approach involving the validation of simulated environments and inference of new knowledge from partially explored maps.

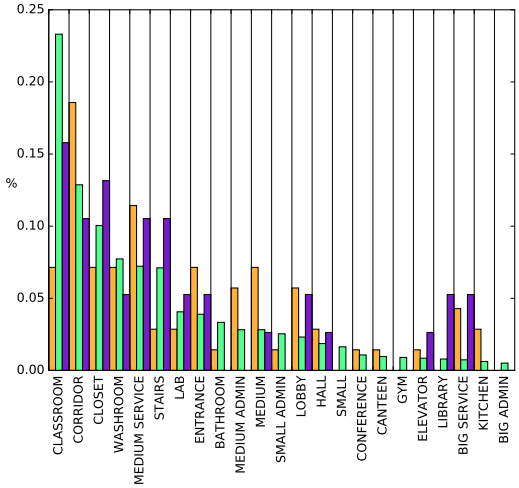
These three proposed approach are intended to be indicative examples of the possible applications of our method and are not an exhaustive discussion of what can be potentially done with it. This is particularly true when discussing the possible use of predicted knowledge about partially explored environments. Although preliminary results are promising, it is still untested how knowledge on partially visited environment could be used



(a) *Predicted semantic graph after step 1.*

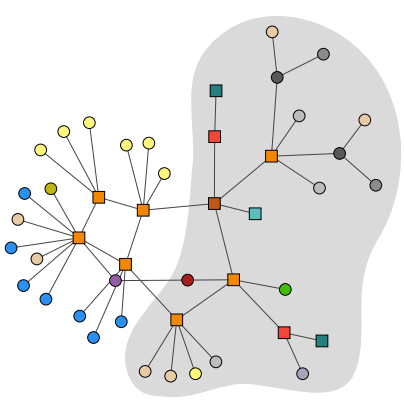


(b) *Predicted corridor structure after step 1.*

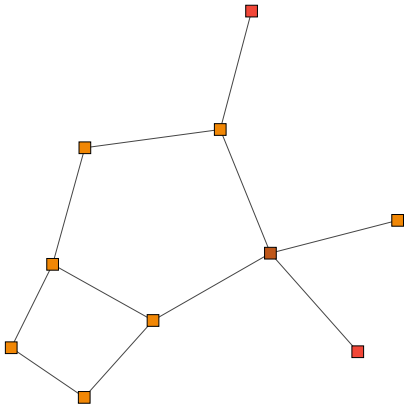


(c) *Labels distribution after step 1.*

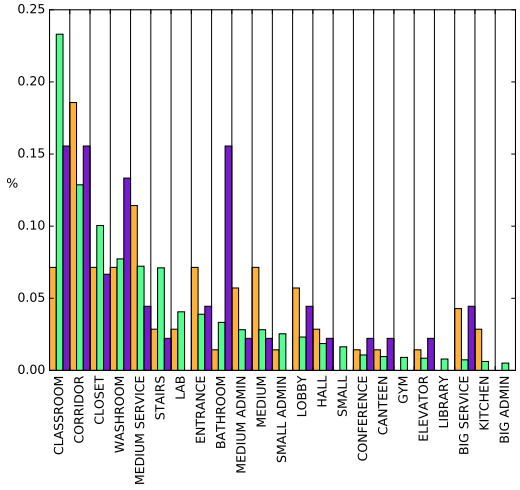
Figure 8.14: *Predictions made after one of three steps of exploration for the environment of Figure 8.13.*



(a) *Predicted semantic graph after step 2.*

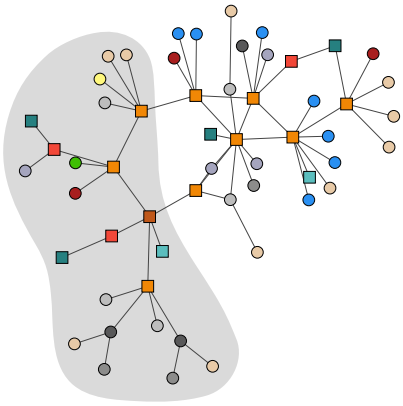


(b) *Predicted corridor structure after step 2.*

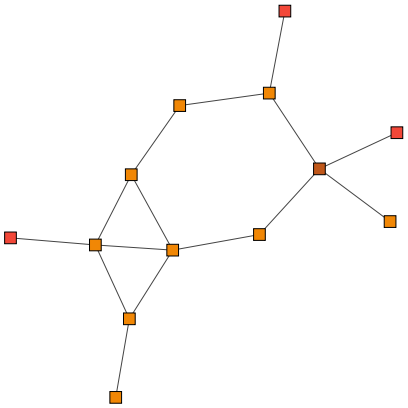


(c) *Labels distribution after step 2.*

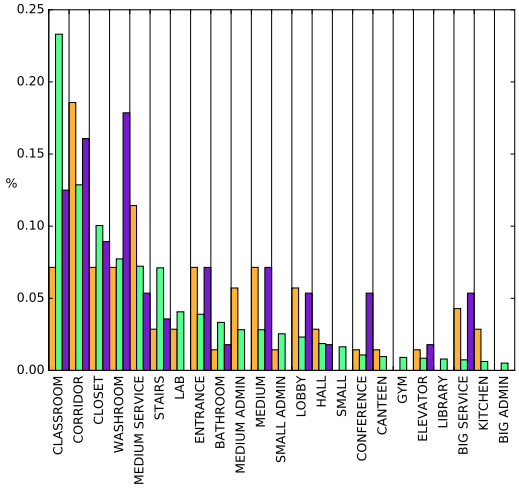
Figure 8.15: *Predictions made after two of three steps of exploration for the environment of Figure 8.13.*



(a) *Predicted semantic graph after step 3.*

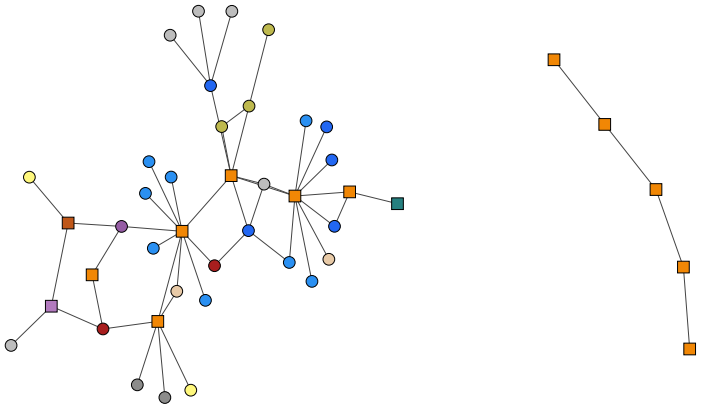


(b) *Predicted corridor structure after step 3.*



(c) *Labels distribution after step 3.*

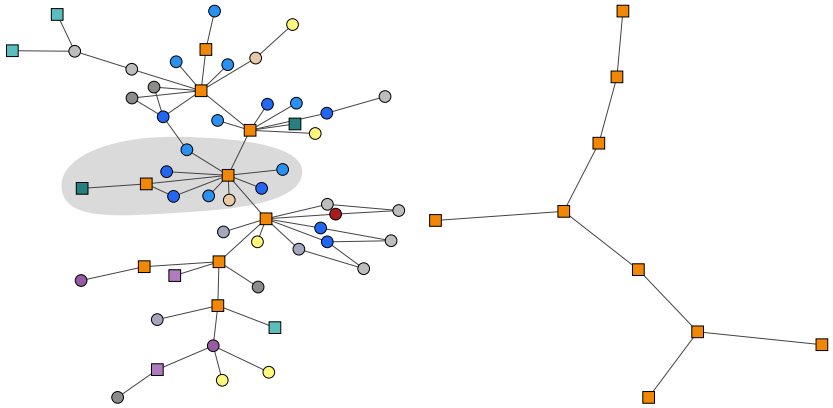
Figure 8.16: *Predictions made after three of exploration for the environment of Figure 8.13.*



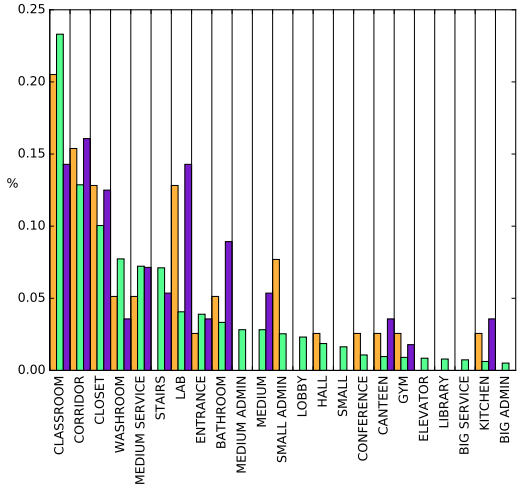
(a) *Real environment.* **(b)** *Corridor structure.*

Figure 8.17: *An environment being explored (third example)*

by autonomous mobile robots and what kind of improvements it could provide. Following this direction, an application of these examples to tasks such as search, (multi)robot exploration, and evacuation of buildings can be interesting in order to evaluate their impact. Future works involve testing the presented methods for predicting partially explored maps on real world robots and include identification and analysis of possibles trade-offs between prediction and exploration in online applications.

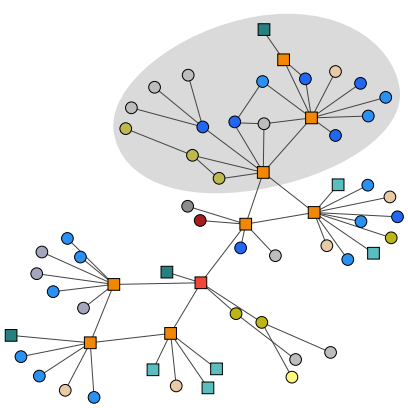


(a) *Predicted semantic graph after step 1.* **(b)** *Predicted corridor structure after step 1.*

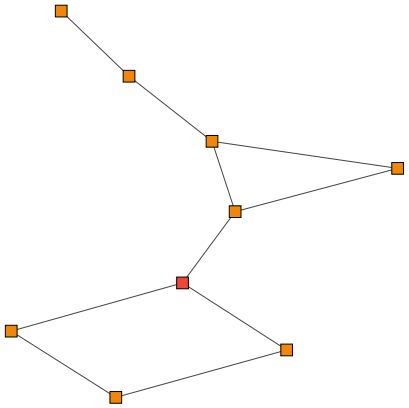


(c) *Labels distribution after step 1.*

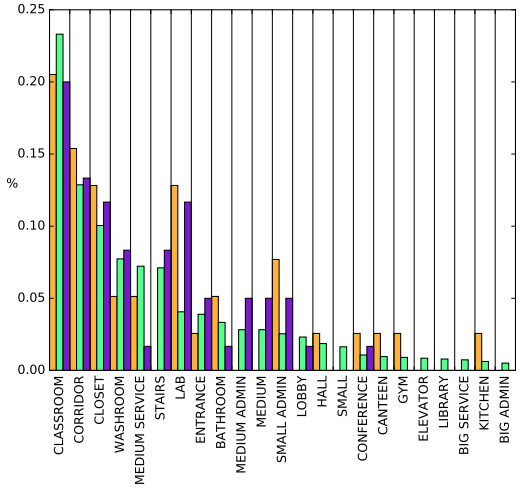
Figure 8.18: *Predictions made after one of two steps of exploration for the environment of Figure 8.17.*



(a) *Predicted semantic graph after step 2.*

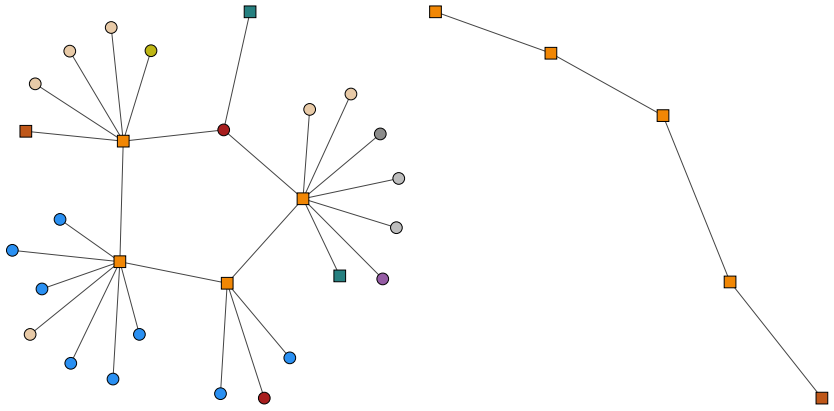


(b) *Predicted corridor structure after step 2.*



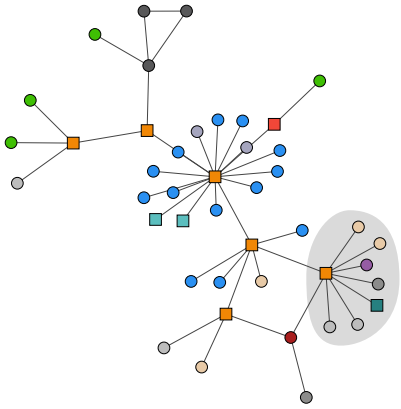
(c) *Labels distribution after step 2.*

Figure 8.19: *Predictions made after two steps of exploration for the environment of Figure 8.17.*

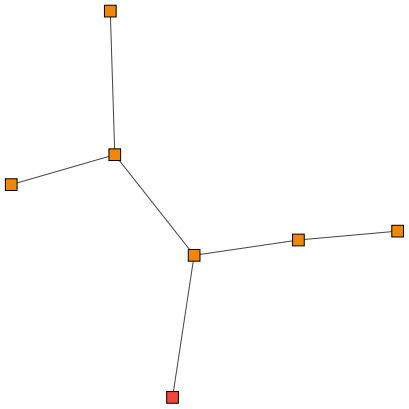


(a) *Real environment.* **(b)** *Corridor structure.*

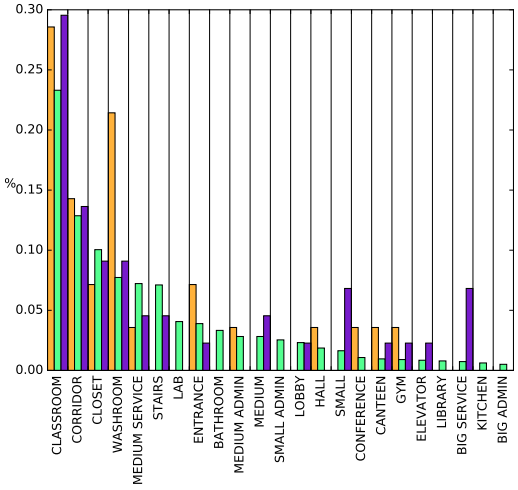
Figure 8.20: *An environment being explored (fourth example).*



(a) *Predicted semantic graph after step 1.*

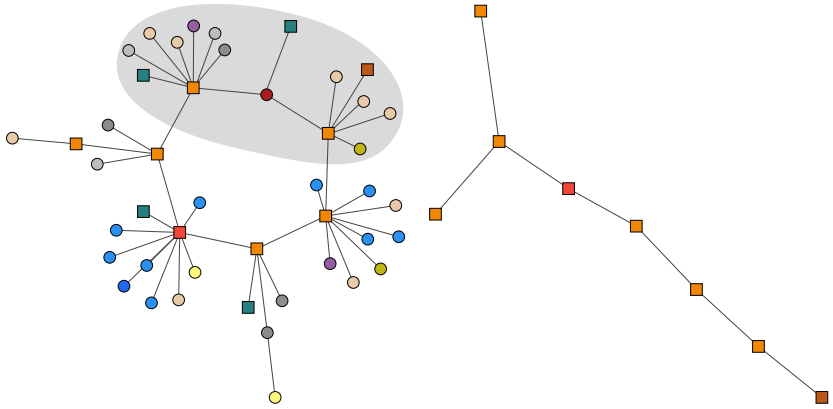


(b) *Predicted corridor structure after step 1.*

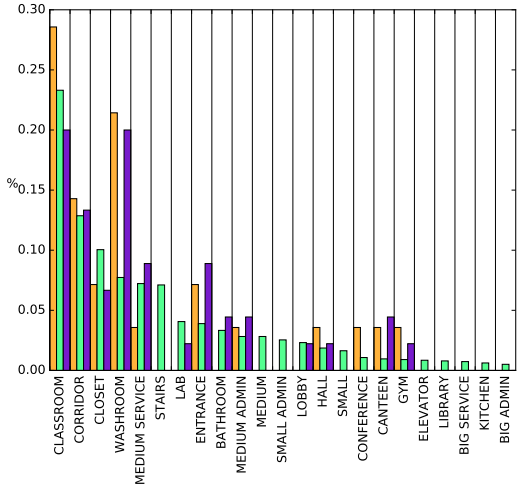


(c) *Labels distribution after step 1.*

Figure 8.21: *Predictions made after one of two steps of exploration for the environment of Figure 8.20.*



(a) *Predicted semantic graph after step 2.* **(b)** *Predicted corridor structure after step 2.*



(c) *Labels distribution after step 2.*

Figure 8.22: *Predictions made after two steps of exploration for the environment of Figure 8.20.*

CHAPTER 9

Conclusion and future works

Understanding the environments in which they operates is a fundamental ability for autonomous mobile robots.

When we consider indoor tasks, robots have to inhabit environments that are specifically designed for human activities, namely buildings. Buildings are strongly structured environments that are organized in regular patterns.

One the ways that have been developed in the last years to enable a robot to perceive and interact with an indoor environment is to represent it through a semantic map. In this dissertation, we dealt with semantic maps that identify rooms, represent how rooms are connected, and assigns to each room a semantic label indicating its function. Semantic maps are usually developed incrementally by integrating perceived data, also considering the metric map obtained from SLAM. This mainstream approach is centered on the robot perception and often implies that what has not been seen by the robot does not exists, adopting in a sense, a closed world assumption on the environment. This form of interaction with the environment is radically different from that of humans, who can easily navigate and comprehend the *structure* of buildings even without having seen them before.

The main contribution of this work aims at modeling and using knowledge about the structure of buildings, which is often neglected when build-

ing semantic maps, for increasing the capabilities of robots when interacting within indoor environments. By addressing some current limitations of the state-of-the-art methods in semantic mapping, our long-term goal is to make a step towards filling the gap between how humans and robots interact with the environment they inhabit.

Along the chapters of this thesis, we defined and modeled the concept of structure of a building following two insights: consider buildings as single entities, thus performing global reasoning on entire buildings, and consider the set of buildings with the same function together, in order to identify common features.

Starting from these premises, at first we have presented a method that extracts the structure of a building from a metric map acquired by a robots. This task is performed by extracting information about the straight walls that compose the environment and using this knowledge for segmenting the metric map into a set of separate rooms. This knowledge is used to reconstruct the layout of a building from its metric map. The layout of a building is an abstract geometrical representation of its floor plan which approximates the metric map, while reducing possible inaccuracies caused by SLAM. From the reconstructed layout, we are able to retrive the topological map of the environment and a feature vector representing the geometrical characteristics of each room.

We have then introduced the concept of building types as the set of buildings which have the same function (Chapter 4). We shown how the structure of buildings is strongly related to their type, and that rooms belonging to the same building type are similar between them, while room belonging to different building types can be very different. Using the concept of model floor plan we described how it is possible to retrieve useful knowledge for training semantic mapping algorithms from data sets of floor plans of buildings belonging to the same type.

By representing buildings as graphs, we underlined the difference in role and importance of ROOMs and CORRIDORs, and we applied such findings to the reconstructed layout (obtained from Chapter 3) for obtaining a semantic map of the environment.

Using a kLog, a SRL tool, we performed semantic mapping by considering all the rooms of a building altogether, thus promoting a global approach (Chapter 6). We used this framework for introducing and performing novel tasks, namely *building classification*, in which we apply a semantic label to an entire environment representing its function, its building type. We have also shown how reasoning on the structure of buildings can be used to assess the validity of simulated environments used in robotics and deciding

if a simulated environment is similar (or not) to real world environments.

Finally, using graph kernels, we developed a model for sampling new instances of floors of buildings given a set of examples (Chapter 7). Starting from a data set of graphs representing buildings of a building type, we are able to identify their common structure and sample new instances of similar buildings.

We concluded by proposing some applications of our approach for inferring new knowledge on partially explored environments (Chapter 8). This is done by predicting the layout of a partially explored room and by providing a prediction of the structure of the corridors of the unseen portion of partially explored environments.

Possible extensions of the proposed contributions regard a further investigation on the concept of building structure. The results presented here are intended to be a step towards building semantic mapping systems that can potentially be employed in every environment. To reach this ambitious goal, however, a number of further issues need to be addressed, starting from collecting data sets comprising even more building types (for example, HOSPITALS and MALLS) and to investigate thoroughly the application and impact of a large data sets representing building types as source of knowledge for semantic mapping. This could be done, for example, by integrating in a standard semantic mapping framework knowledge representing the building structure (as we presented in Chapter 6). Using the reconstructed layout of Chapter 3, we can derive new sets of relations between different parts of an environment (e.g., identifying symmetries between parts of building) and extending of the semantic representation of buildings to involve multiple relations between nodes, beyond adjacency. Objects identified using object recognition techniques can be used to enrich the relational representation of a building presented in Chapter 6 by better characterizing each room's function.

Similarly, it can be interesting to combine our layout reconstruction method of Chapter 3 with standard room segmentation approaches to investigate if better performance can be obtained by looking both at the global structural features and at the local features of the buildings. Moreover, we would like to carefully assess the performance of our layout segmentation method in presence of noise and large occlusions in metric maps and to extend it to 3D point clouds as input data, possibly integrating our representative lines in 2D with wall candidates identified as planes in 3D. Finally, we plan to integrate the reconstructed layouts with knowledge obtained from data sets representing building types and the generative models presented in Chapter 7 for performing semantic classification of rooms and for better

predicting the layout of partially explored rooms.

Reasoning about unexplored parts of the environment is still a largely unaddressed issue for autonomous robotics heretofore. In this work we have introduced a set of preliminary applications of our contributions in this direction. A comprehensive analysis of the benefits of obtaining (possibly inaccurate at level of single rooms, but accurate at level of structure) semantic knowledge about unvisited parts of the environment can provide useful insights on how to bridge the gap between our understanding of how human beings and robots interact with indoor environments.

Bibliography

[1] Pratik Agarwal, Wolfram Burgard, and Cyrill Stachniss. Helmert's and Bowie's geodetic mapping methods and their relation to graph-based SLAM. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 3619–3625, 2014.

[2] Sheraz Ahmed, Marcus Liwicki, Markus Weber, and Andreas Dengel. Improved automatic analysis of architectural floor plans. In *Proceedings of the International Conference on Document Analysis and Recognition (ICDAR)*, pages 864–869, 2011.

[3] Sheraz Ahmed, Markus Weber, Marcus Liwicki, Christoph Langenhan, Andreas Dengel, and Frank Petzold. Automatic analysis and sketch-based retrieval of architectural floor plans. *Pattern Recognition Letters*, 35:91–100, 2014.

[4] Francesco Amigoni, Matteo Luperto, and Viola Schiaffonati. Toward generalization of experimental results for autonomous robots. *Robotics and Autonomous Systems*, 2016.

[5] Francesco Amigoni and Viola Schiaffonati. Good experimental methodologies and simulation in autonomous mobile robotics. *Model-Based Reasoning in Science and Technology*, pages 315–332, 2010.

[6] Iro Armeni, Ozan Sener, Amir R Zamir, Helen Jiang, Ioannis Brilakis, Martin Fischer, and Silvio Savarese. 3d semantic parsing of large-scale indoor spaces. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[7] Alper Aydemir, Patric Jensfelt, and John Folkesson. What can we learn from 38,000 rooms? Reasoning about unexplored space in indoor environments. In *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4675–4682, 2012.

[8] Alper Aydemir, Andrzej Pronobis, Moritz Gobelbecker, and Patric Jensfelt. Active visual object search in unknown environments using uncertain semantics. *IEEE Transactions on Robotics*, 29(4):986–1002, 2013.

[9] Albert-László Barabási and Réka Albert. Emergence of scaling in random networks. *Science*, 286(5439):509–512, 1999.

[10] Bahram Behzadian, Pratik Agarwal, Wolfram Burgard, and Gian Diego Tipaldi. Monte carlo localization in hand-drawn maps. In *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4291–4296, 2015.

[11] Christopher M. Bishop. *Pattern recognition and machine learning*. Springer, 2006.

[12] Jose-Luis Blanco, Javier Gonzalez, and Juan-Antonio Fernandez-Madrigal. Consistent observation grouping for generating metric-topological maps that improves robot localization. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 818–823, 2006.

[13] Federico Boniardi, Bahram Behzadian, Wolfram Burgard, and Gian Diego Tipaldi. Robot navigation in hand-drawn sketched maps. In *Proceedings of the European Conference on Mobile Robots (ECMR)*, pages 1–6, 2015.

[14] Richard Bormann, Joshua Hampp, and Martin Hägele. New brooms sweep clean-an autonomous robotic cleaning assistant for professional office cleaning. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 4470–4477, 2015.

[15] Richard Bormann, Florian Jordan, Wenzhe Li, Joshua Hampp, and Martin Hỳgele. Room segmentation: Survey, implementation, and analysis. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 1019–1026, 2016.

[16] Emma Brunskill, Thomas Kollar, and Nicholas Roy. Topological mapping using spectral clustering and classification. In *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3491–3496, 2007.

[17] Wolfram Burgard, Mark Moors, Cyrill Stachniss, and Frank E. Schneider. Coordinated multi-robot exploration. *IEEE Transactions on Robotics*, 21(3):376–386, 2005.

[18] Pär Buschka and Alessandro Saffiotti. A virtual sensor for room detection. In *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 637–642, 2002.

[19] John Canny. A computational approach to edge detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 679–698, 1986.

[20] Roberto Capobianco, Guglielmo Gemignani, Domenico Daniele Bloisi, Daniele Nardi, and Luca Iocchi. Automatic extraction of structural representations of environments. In *Proceedings of the International Conference on Intelligent Autonomous Systems (IAS-13)*, pages 721–733, 2014.

[21] Stefano Carpin, Mike Lewis, Jijun Wang, Steve Balakirsky, and Chris Scrapper. Bridging the gap between simulation and reality in urban search and rescue. In *Proceedings of the RoboCup Symposium*, pages 1–12, 2007.

[22] Henrik Christensen and Elin AAnna Topp. Detecting region transitions for human-augmented mapping. *IEEE Transaction on Robotics*, 26, 2010.

[23] Michael Jae-Yoon Chung, Andrzej Pronobis, Maya Cakmak, Dieter Fox, and Rajesh PN Rao. Autonomous question answering with mobile robots in human-populated environments. In *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 823–830, 2016.

[24] Riccardo Cipolleschi, Michele Giusto, Alberto Quattrini Li, and Francesco Amigoni. Semantically-informed coordinated multirobot exploration of relevant areas in search and rescue settings. In *Proceedings of the European Conference on Mobile Robots (ECMR)*, pages 216–221, 2013.

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

[26] Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine learning*, 20(3):273–297, 1995.

[27] Fabrizio Costa. Learning an efficient constructive sampler for graphs. *Artificial Intelligence*, 2016. http://dx.doi.org/10.1016/j.artint.2016.01.006.

[28] Fabrizio Costa and Kurt De Grave. Fast neighborhood subgraph pairwise distance kernel. In *Proceedings of the International Conference on Machine Learning (ICML)*, pages 255–262, 2010.

[29] Gabriele Costante, Thomas Ciarfuglia, Paolo Valigi, and Elisa Ricci. A transfer learning approach for multi-cue semantic place recognition. In *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2122–2129, 2013.

[30] Luc De Raedt. *Logical and Relational Learning*. Springer, 2008.

[31] Martin Ester, Hans-Peter Kriegel, Jörg Sander, and Xiaowei Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Proceedings of the International Conference on Knowledge Discovery and Data Mining (KDD)*, pages 226–231, 1996.

[32] Elisabetta Fabrizi and Alessandro Saffiotti. Extracting topology-based maps from gridmaps. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 2972–2978. IEEE, 2000.

[33] Aasa Feragen, Niklas Kasenburg, Jens Petersen, Marleen de Bruijne, and Karsten Borgwardt. Scalable kernels for graphs with continuous attributes. In *Proceedings of the International Conference on Neural Information Processing Systems (NIPS)*, pages 216–224, 2013.

[34] Paolo Frasconi, Fabrizio Costa, Luc De Raedt, and Kurt De Grave. klog: A language for logical and relational learning with kernels. *Artificial Intelligence*, 217:117–143, 2014.

[35] Brendan J Frey and Delbert Dueck. Clustering by passing messages between data points. *Science*, 315(5814):972–976, 2007.

[36] Stephen Friedman, Hanna Pasula, and Dieter Fox. Voronoi random fields: Extracting the topological structure of indoor environments via place labeling. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, volume 35, pages 2109–2114, 2007.

[37] Axel Furlan, Stephen D Miller, Domenico G Sorrenti, Fei-Fei Li, and Silvio Savarese. Free your camera: 3d indoor scene understanding from arbitrary camera motion. In *Proceedings of the British Machine Vision Conference (BMVC)*, pages 24.1–24.12, 2013.

[38] Cipriano Galindo, Juan Antonio Fernández-Madrigal, Javier González, and A. Saffiotti. Robot task planning using semantic maps. *Robotics and Autonomous Systems*, 56(11):955–966, 2008.

[39] Cipriano Galindo, Alessandro Saffiotti, Silvia Coradeschi, Pär Buschka, Juan Antonio Fernandez-Madrigal, and Javier González. Multi-hierarchical semantic maps for mobile robotics. In *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2278–2283, 2005.

[40] Thomas Gärtner, John W Lloyd, and Peter A Flach. Kernels and distances for structured data. *Machine Learning*, 57(3):205–232, 2004.

[41] Lise Getoor and Ben Taskar. *Introduction to Statistical Relational Learning*. MIT press, 2007.

[42] Pierre Geurts, Damien Ernst, and Louis Wehenkel. Extremely randomized trees. *Machine Learning*, 63(1):3–42, 2006.

[43] Jiawei Han, Micheline Kamber, and Jian Pei. *Data mining: concepts and techniques*. Morgan Kaufmann, 2006.

[44] Ankur Handa, Thomas Whelan, John McDonald, and Andrew J Davison. A benchmark for RGB-D visual odometry, 3D reconstruction and SLAM. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 1524–1531, 2014.

[45] Marc Hanheide, Moritz Göbelbecker, Graham S. Horn, Andrzej Pronobis, Kristoffer Sjöö, Alper Aydemir, Patric Jensfelt, Charles Gretton, Richard Dearden, Miroslav Janicek, Hendrik Zender, Geert-Jan Kruijff, Nick Hawes, and Jeremy L. Wyatt. Robot task planning and explanation in open and uncertain worlds. *Artificial Intelligence*, in press:–, 2015.

[46] Marc Hanheide, Charles Gretton, Richard Dearden, Nick Hawes, Jeremy Wyatt, Andrzej Pronobis, Alper Aydemir, Moritz Göbelbecker, and Hendrik Zender. Exploiting probabilistic knowledge under uncertain sensing for efficient robot behaviour. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, volume 22, page 2442, 2011.

[47] David Haussler. Convolution kernels on discrete structures. Technical report, University of Santa Cruz, 1999.

[48] Varsha Hedau, Derek Hoiem, and David Forsyth. Recovering the spatial layout of cluttered rooms. In *Proceedings of the International Conference on Computer Vision (ICCV)*, pages 1849–1856, 2009.

[49] Sachithra Hemachandra, Matthew Walter, Stefanie Tellex, and Seth Teller. Learning spatial-semantic representations from natural language descriptions and scene classifications. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 2623–2630, 2014.

[50] Nick Jacobi, Phil Husbands, and Inman Harvey. Noise and the reality gap: The use of simulation in evolutionary robotics. In *Proceedings of the European Conference on Artificial Life (ECAL)*, pages 704–720, 1995.

[51] Hisashi Kashima, Koji Tsuda, and Akihiro Inokuchi. Marginalized kernels between labeled graphs. In *Proceedings of the International Conference on Machine Learning (ICML)*, volume 3, pages 321–328, 2003.

[52] Nahum Kiryati, Yuval Eldar, and Alfred M Bruckstein. A probabilistic hough transform. *Pattern Recognition*, 24(4):303–316, 1991.

[53] Daphne Koller and Nir Friedman. *Probabilistic graphical models: principles and techniques*. MIT press, 2009.

[54] Ioannis Kostavelis and Antonios Gasteratos. Semantic mapping for mobile robotics tasks: A survey. *Robotics and Autonomous Systems*, 66:86–103, 2015.

[55] Lars Kunze, Michael Beetz, Manabu Saito, Haseru Azuma, Kei Okada, and Masayuki Inaba. Searching objects in large-scale indoor environments: A decision-theoretic approach. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 4385–4390, 2012.

[56] Ziyuan Liu and Georg von Wichert. A generalizable knowledge framework for semantic indoor mapping based on Markov logic networks and data driven MCMC. *Future Generation Computer Systems*, 36:42–56, 2014.

[57] Matteo Luperto and Francesco Amigoni. Exploiting structural properties of buildings towards general semantic mapping systems. In *Proceedings of the International Conference on Intelligent Autonomous Systems (IAS-13)*, pages 375–387, 2014.

[58] Matteo Luperto, Leone D'Emilio, and Francesco Amigoni. A generative spectral model for semantic mapping of buildings. In *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4451–4458, 2015.

[59] Matteo Luperto, Alberto Quattrini Li, and Francesco Amigoni. A system for building semantic maps of indoor environments exploiting the concept of building typology. In *Proceedings of the RoboCup Symposium*, pages 504–515, 2013.

[60] Matteo Luperto, Alessandro Riva, and Francesco Amigoni. Semantic classification by reasoning on the whole structure of buildings using statistical relational learning techniques. In *Accepted at the IEEE International Conference on Robotics and Automation (ICRA)*, 2017.

[61] Kevin Lynch. *The Image of the City*. MIT press, 1960.

[62] Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-SNE. *Journal of Machine Learning Research*, 9:2579–2605, 2008.

[63] Sauro Menchetti, Fabrizio Costa, and Paolo Frasconi. Weighted decomposition kernels. In *Proceedings of the International Conference on Machine Learning (ICML)*, pages 585–592, 2005.

[64] Oscar M. Mozos. *Semantic labeling of places with mobile robots*, volume 61 of *Springer Tracts in Advanced Robotics*. Springer, 2010.

[65] Oscar M. Mozos, Hitoshi Mizutani, Ryo Kurazume, and Tsutomu Hasegawa. Categorization of indoor places using the kinect sensor. *Sensors*, 12(5):6695–6711, 2012.

[66] Oscar M. Mozos, Cyrill Stachniss, and Wolfram Burgard. Supervised learning of places from range data using AdaBoost. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 1730–1735, 2005.

[67] Oscar M. Mozos, Rudolph Triebel, Patrick Jensfelt, Axel Rottmann, and Wolfram Burgard. Supervised semantic labeling of places using information extracted from sensor data. *Robotics and Autonomous Systems*, 55(5):391–402, 2007.

[68] Claudio Mura, Oliver Mattausch, Alberto Jaspe Villanueva, Enrico Gobbetti, and Renato Pajarola. Automatic room detection and reconstruction in cluttered indoor environments with complex room layouts. *Computers & Graphics*, 44:20–32, 2014.

[69] Ernst Neufert and Peter Neufert. *Architects' data*. Wiley-Blackwell, 2012.

[70] Mark EJ Newman. Mixing patterns in networks. *Physical Review E*, 67(2):026126–1 – 026126–13, 2003.

[71] Carlos Nieto-Granda, John G Rogers, Alexander JB Trevor, and Henrik I Christensen. Semantic map partitioning in indoor environments using regional analysis. In *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1451–1456, 2010.

[72] Stefan Oßwald, Maren Bennewitz, Wolfram Burgard, and Cyrill Stachniss. Speeding-up robot exploration by exploiting background information. *IEEE Robotics and Automation Letters*, 1(2):716–723, 2016.

[73] Daniel Perea Strom, Fabrizio Nenci, and Cyrill Stachniss. Predictive exploration considering previously mapped environments. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 2761–2766, 2015.

[74] Bradford Perkins. *Building type basics for elementary and secondary schools*. Wiley, 2001.

[75] Cristiano Premebida, Diego R Faria, and Urbano Nunes. Dynamic bayesian network for semantic place classification in mobile robotics. *Autonomous Robots*, pages 1–12, 2016.

[76] Andrzej Pronobis. *Semantic mapping with mobile robots*. PhD thesis, KTH, 2011.

[77] Andrzej Pronobis and Patric Jensfelt. Large-scale semantic mapping and reasoning with heterogeneous modalities. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 3515–3522, 2012.

[78] Andrzej Pronobis, Oscar M. Mozos, Barbara Caputo, and Patrick. Jensfelt. Multi-modal semantic place classification. *International Journal of Robotics Research*, 29(2-3):298–320, 2010.

[79] Alberto Quattrini Li, Riccardo Cipolleschi, Michele Giusto, and Francesco Amigoni. A semantically-informed multirobot system for exploration of relevant areas in search and rescue settings. *Autonomous Robots*, 40(4):581–597, 2016.

[80] Ananth Ranganathan. Pliss: Detecting and labeling places using online change-point detection. In *Proceedings of Robotics: Science and Systems (RSS)*, 2010.

[81] Matthew Richardson and Pedro Domingos. Markov logic networks. *Machine Learning*, 62(1-2):107–136, 2006.

[82] RoboCup Rescue League. `http://www.robocuprescue.org/`, 2015. [Online; accessed 1-March-2016].

[83] Andrew Rosenberg and Julia Hirschberg. V-measure: A conditional entropy-based external cluster evaluation measure. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Processing (EMNLP-CoNLL)*, volume 7, pages 410–420, 2007.

[84] Aldo Rossi. *The architecture of the city*. MIT Press, 1984.

[85] Haim Schweitzer, James Wesley Bell, and Fu Wu. Very fast template matching. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 358–372, 2002.

[86] Nino Shervashidze, Pascal Schweitzer, Erik Jan Van Leeuwen, Kurt Mehlhorn, and Karsten M Borgwardt. Weisfeiler-lehman graph kernels. *The Journal of Machine Learning Research*, 12:2539–2561, 2011.

[87] Jianbo Shi and Jitendra Malik. Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):888–905, 2000.

[88] Lei Shi, Sarath Kodagoda, and Gamini Dissanayake. Laser range data based semantic labeling of places. In *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 5941–5946, 2010.

[89] Kristoffer Sjöö. Semantic map segmentation using function-based energy maximization. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 4066–4073, 2012.

[90] Ray Smith. An overview of the tesseract ocr engine. In *Proceedings of the International Conference on Document Analysis and Recognition (ICDAR)*, pages 629–633. IEEE Computer Society, 2007.

[91] C. Stachniss, O.M. Mozos, and W. Burgard. Efficient exploration of unknown indoor environments using a team of mobile robots. *Annals of Mathematics and Artificial Intelligence*, 52(2):205–227, 2008.

[92] Cyrill Stachniss, Oscar M. Mozos, and Wolfram Burgard. Speeding-up multi-robot exploration by considering semantic place information. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 1692–1697, 2006.

[93] Niko Sünderhauf, Feras Dayoub, Sean McMahon, Ben Talbot, Ruth Schulz, Peter Corke, Gordon Wyeth, Ben Upcroft, and Michael Milford. Place categorization and semantic mapping on a mobile robot. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2016.

[94] Satoshi Suzuki and KeiichiA be. Topological structural analysis of digitized binary images by border following. *Computer Vision, Graphics, and Image Processing*, 30(1):32–46, 1985.

[95] The Whole Building Design Guide. `https://www.wbdg.org`, 2015. [Online; accessed 1-March-2016].

[96] Sebastian Thrun. Learning metric-topological maps for indoor mobile robot navigation. *Artificial Intelligence*, 99(1):21–71, 1998.

[97] Sebastian Thrun, Wolfram Burgard, and Dieter Fox. *Probabilistic Robotics*. The MIT Press, 2005.

[98] Elin Anna Topp and Henrik Iskov Christensen. Topological modelling for human augmented mapping. In *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2257–2263, 2006.

[99] Eric Turner, Peter Cheng, and Avideh Zakhor. Fast, automated, scalable generation of textured 3d models of indoor environments. *IEEE Journal on Emerging and Selected Topics in Circuits and Systems*, 9(3):409–421, 2015.

[100] Pooja Viswanathan, David Meger, Tristram Southey, James J Little, and Alan K Mackworth. Automated spatial-semantic modeling with applications to place labeling and informed search. In *Proceedings of the Canadian Conference on Computer and Robot Vision (CRV)*, pages 284–291, 2009.

[101] Emily Whiting, Jonathan Battat, and Seth Teller. Topology of urban environments. In *Proceedings of the Computed-Aided Architectural Design Futures Conference (CAADFutures)*, pages 115–128, 2007.

[102] Wera Winterhalter, Freya Fleckenstein, Bastian Steder, Luciano Spinello, and Wolfram Burgard. Accurate indoor localization for rgb-d smartphones and tablets given 2d floor plans. In *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3138–3143, 2015.

[103] Hendrik Zender, Oscar M. Mozos, Patrick Jensfelt, Geert-Jan Kruijff, and W. Burgard. Conceptual spatial representations for indoor mobile robots. *Robotics and Autonomous Systems*, 56(6):493–502, 2008.

[104] Bolei Zhou, Agata Lapedriza, Jianxiong Xiao, Antonio Torralba, and Aude Oliva. Learning deep features for scene recognition using places database. In *Advances in neural information processing systems*, pages 487–495, 2014.

[105] Olaf Zwennes, Astrid Weiss, and Arnaud Visser. Adapting the mapping difficulty for the automatic generation of rescue challenges. In *Proceedings of RoboCup Iran Open*, pages 482–487, 2012.