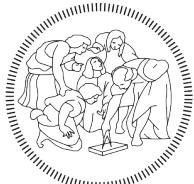


**POLITECNICO DI MILANO**  
Corso di Laurea Magistrale in Ingegneria Informatica  
Dipartimento di Elettronica, Informazione e Bioingegneria



**POLITECNICO**  
**MILANO 1863**

**COSTRUZIONE DI MAPPE  
MULTILIVELLO PER ROBOT MOBILI  
AUTONOMI**

Relatore: prof. Francesco Amigoni  
Correlatore: ing. Matteo Luperto

Tesi di Laurea Magistrale di:  
Matteo Calabrese, matricola 818046

Anno Accademico 2015-2016



# Sommario

In questo lavoro di tesi si propone un sistema di mapping multilivello per ambienti indoor, nell’ambito della robotica autonoma. Questo sistema è in grado di rappresentare diversi livelli di conoscenza relativi all’ambiente tramite l’integrazione di diverse mappe, rappresentando i legami tra le diverse rappresentazioni.

Le mappe, organizzate sequenzialmente secondo un livello di astrazione crescente, sono: mappa metrica, layout delle stanze, grafo topologico e mappa semantica. Le mappe sono ottenute da diversi algoritmi strutturati come layer, ognuno dei quali elabora l’output del layer inferiore allo scopo di estrarne ulteriori informazioni.

Il metodo è stato sperimentato su 56 diversi ambienti. I risultati delle sperimentazioni, mostrati in modo esaustivo in Appendice A, sono stati valutati utilizzando delle metriche, e risultano simili a quelli ottenuti da altri metodi presi dallo stato dell’arte, dimostrando un buon livello di accuratezza.



# Ringraziamenti

Si conclude un periodo importante della mia vita, un'esperienza fatta di impegno, sacrifici, ma anche soddisfazioni e gratificazioni. Sono consapevole che da solo non sarei mai stato in grado di superare le difficoltà in cui mi sono imbattuto lungo il percorso universitario, dunque mi sembra doveroso ringraziare chi mi ha supportato ed accompagnato lungo questo cammino.

Il primo ringraziamento va alla mia famiglia, senza la quale questa esperienza non sarebbe stata possibile. Mi avete sostenuto economicamente e moralmente, spronandomi a dare il meglio e mantenendo sempre il giusto equilibrio. Questa laurea è anche vostra.

Voglio poi ringraziare i miei amici, in particolare Pietro e Davide (alias Mario), che con la loro preziosa compagnia mi hanno restituito il sorriso ogniqualvolta le cose non andavano nel verso giusto. La vostra sincera amicizia mi ha protetto dall'insicurezza e dalla paura di non farcela.

Un ringraziamento va anche a Luca, Fede, Ale, Save (con le sue stupende pizzate) e tutti gli altri compagni di università che hanno alleggerito ore di lezioni ed esercitazioni con la loro simpatia. Studiare con voi è stato davvero bello.

Infine il mio più grande ringraziamento va a te, Silvia. Sei stata il mio indispensabile sostegno, hai accolto le mie preoccupazioni trasformandole in fiducia e motivazione. Il tuo amore mi ha spronato a superare ogni nuovo ostacolo e ha curato ogni mia inquietudine. Mi hai sempre supportato (e sopportato), valorizzando il mio impegno. Grazie, grazie, grazie.



# Indice

<b>Sommario</b>	<b>I</b>
<b>Ringraziamenti</b>	<b>III</b>
<b>1 Introduzione</b>	<b>1</b>
<b>2 Stato dell'arte</b>	<b>5</b>
2.1 Il mapping nella robotica mobile autonoma . . . . .	5
2.2 Mappe multilivello . . . . .	6
2.3 Mappa metrica . . . . .	10
2.4 Layout delle stanze . . . . .	11
2.4.1 Approcci basati su point-cloud . . . . .	12
2.4.2 Approcci basati su immagini RGBD . . . . .	15
2.4.3 Approcci basati su Voronoi graph . . . . .	17
2.4.4 Approcci basati su graph partitioning . . . . .	19
2.4.5 Approcci basati su feature . . . . .	20
2.4.6 Approcci basati su operatori morfologici . . . . .	22
2.4.7 Approcci basati su distance transform . . . . .	23
2.4.8 Approcci basati su interpretazione di planimetrie . . . . .	24
2.4.9 Approcci basati su preprocessing dell'ambiente . . . . .	25
2.5 Mappa topologica . . . . .	29
2.6 Mappa semantica . . . . .	32
<b>3 Progetto logico della costruzione del sistema multilivello</b>	<b>37</b>
3.1 Il sistema multilivello . . . . .	37
3.2 Mappa metrica . . . . .	38
3.2.1 Obiettivo . . . . .	38
3.2.2 Mappa metrica non nota a priori . . . . .	38
3.2.3 Mappa metrica nota a priori . . . . .	41
3.3 Layout delle stanze . . . . .	42
3.3.1 Obiettivo . . . . .	42

3.3.2	Preprocessing della planimetria . . . . .	43
3.3.3	Individuazione dei segmenti corrispondenti ai muri . . . . .	51
3.3.4	Calcolo delle rette rappresentative dei muri . . . . .	54
3.3.5	Partizionamento in celle della mappa metrica . . . . .	57
3.3.6	Raggruppamento delle celle interne in stanze . . . . .	61
3.4	Grafo topologico . . . . .	64
3.4.1	Obiettivo . . . . .	64
3.4.2	Voronoi graph . . . . .	66
3.4.3	Medial axis . . . . .	69
3.4.4	Analisi dei templates delle porte . . . . .	70
3.5	Mappa semantica . . . . .	71
3.5.1	Obiettivo . . . . .	71
3.5.2	Le feature che descrivono le stanze . . . . .	72
3.5.3	Schema di labeling . . . . .	76
3.5.4	Extremely Randomized Trees . . . . .	77
3.5.5	Dati di training . . . . .	80
3.5.6	Classificazione . . . . .	82
<b>4</b>	<b>Architettura del sistema</b>	<b>85</b>
4.1	Il modulo MappaMetrica . . . . .	86
4.1.1	Mappa metrica non nota a priori . . . . .	86
4.1.2	Mappa metrica nota a priori . . . . .	88
4.2	Il modulo Preprocessing . . . . .	89
4.2.1	Individuazione dei template . . . . .	89
4.2.2	Eliminazione dei template . . . . .	89
4.2.3	Individuazione dell'informazione testuale . . . . .	90
4.2.4	Eliminazione dell'informazione testuale . . . . .	91
4.2.5	Dati di input/output . . . . .	91
4.3	Il modulo LayoutStanze . . . . .	93
4.3.1	Identificazione dei segmenti corrispondenti ai muri . . . . .	93
4.3.2	Clustering dei segmenti . . . . .	95
4.3.3	Rette rappresentative e celle . . . . .	96
4.3.4	Calcolo del peso dei contorni delle celle . . . . .	97
4.3.5	Classificazione delle celle . . . . .	97
4.3.6	Matrice di affinità . . . . .	98
4.3.7	Clustering delle celle in stanze . . . . .	98
4.3.8	Dati di input/output . . . . .	99
4.4	Il modulo GrafoTopologico . . . . .	100
4.4.1	Medial axis . . . . .	100
4.4.2	Analisi dei template delle porte . . . . .	102

4.4.3	Creazione del grafo topologico . . . . .	102
4.4.4	Dati di input/output . . . . .	103
4.5	Il modulo MappaSemantica . . . . .	103
4.5.1	Creazione del file XML . . . . .	103
4.5.2	Creazione del classificatore extra-trees . . . . .	105
4.5.3	Addestramento del classificatore . . . . .	105
4.5.4	Classificazione . . . . .	106
4.5.5	Creazione della mappa semantica . . . . .	107
4.5.6	Dati di input/output . . . . .	107
<b>5</b>	<b>Realizzazioni sperimentali e valutazione</b>	<b>109</b>
5.1	Gli step di preprocessing . . . . .	110
5.1.1	Template matching . . . . .	111
5.1.2	Individuazione dell'informazione testuale . . . . .	113
5.1.3	Eliminazione delle feature individuate . . . . .	114
5.2	Gli step di costruzione del layout delle stanze . . . . .	115
5.2.1	Canny edge detection . . . . .	116
5.2.2	Hough line transform . . . . .	117
5.2.3	Clustering dei segmenti . . . . .	118
5.2.4	Suddivisione in celle . . . . .	120
5.2.5	Clustering delle celle . . . . .	121
5.3	Performance globali di segmentazione . . . . .	124
5.3.1	Valutazione qualitativa . . . . .	125
5.3.2	Valutazione quantitativa . . . . .	128
5.3.3	Mappe con mobilia . . . . .	135
5.4	Performance globali di classificazione . . . . .	136
5.4.1	Valutazione qualitativa . . . . .	137
5.4.2	Valutazione quantitativa . . . . .	139
<b>6</b>	<b>Conclusioni e direzioni future di ricerca</b>	<b>149</b>
<b>Bibliografia</b>		<b>155</b>
<b>A Risultati completi</b>		<b>167</b>



# Capitolo 1

## Introduzione

Uno dei principali obiettivi della ricerca nel campo della robotica mobile è dotare i robot di autonomia, ovvero renderli in grado di prendere decisioni e compiere azioni senza il continuo intervento umano. Il raggiungimento di un alto livello di autonomia per un robot mobile richiede l’interazione con l’ambiente circostante. Tale interazione necessita spesso di una rappresentazione dell’ambiente tramite una *mappa* che sia d’aiuto al robot per portare a termine le attività da svolgere. Di conseguenza, una condizione necessaria per il raggiungimento dell’autonomia da parte di un robot mobile è il possesso di un’efficace rappresentazione dell’ambiente circostante, sotto forma di mappa. Le mappe sono tipicamente ricavate da dati acquisiti tramite sensori montati sui robot, quali laser scanner e telecamere o kinect.

Una mappa può essere rappresentata in molteplici modi, distinti in base al grado di astrazione e al tipo di informazione offerta. Questa distinzione risulta evidente considerando le differenze tra una mappa metrica e una mappa semantica. La mappa metrica fornisce al robot una conoscenza di basso livello relativa alla struttura e all’occupazione (spazio libero o ostacoli) dell’ambiente circostante il robot, mentre la mappa semantica eleva il grado di astrazione attribuendo un significato ad ogni regione di spazio mediante un’etichetta semantica, o *label*, e operando una categorizzazione dell’ambiente simile a quanto farebbe una persona. Esempi di label possono essere ‘cucina’, ‘corridoio’, ‘ufficio’, ecc.

Un robot può utilizzare contemporaneamente diversi tipi di mappe,legate tra loro, corrispondenti a diversi livelli di astrazione. In questo caso si parla di sistemi di *mapping multilivello*. Questi sistemi hanno la capacità di rappresentare i diversi livelli di conoscenza relativi all’ambiente tramite l’integrazione di diverse mappe. I sistemi di questo tipo sono in grado di rappresentare le informazioni sull’ambiente e i legami tra le diverse rap-

presentazioni. Le mappe incluse in un sistema multilivello sono tra loro correlate, e modellano lo spazio a diversi gradi di astrazione in base al proprio livello.

In questo lavoro di tesi è stato sviluppato un sistema di mapping multilivello per edifici. Il sistema integra in un unico schema diversi tipi di mappe. Queste, ottenute sequenzialmente secondo livelli di astrazione crescenti, sono: mappa metrica, layout delle stanze, grafo topologico e mappa semantica. I diversi algoritmi sono strutturati come layer, ognuno dei quali elabora l'output del layer sottostante allo scopo di estrarne ulteriori informazioni.

Il punto di partenza consiste nella mappa metrica, che offre il più basso grado di astrazione. Essa fornisce informazioni di carattere metrico riguardanti l'occupazione delle aree di un ambiente indoor, come mostrato in [65]. Può essere nota a priori oppure ottenuta tramite esplorazione dell'edificio ed elaborazione dei dati sensoriali raccolti.

Il livello di conoscenza della mappa metrica viene raffinato dal layer successivo, corrispondente al layout delle stanze. Questo approccio di rappresentazione elabora le informazioni metriche in modo da ottenere un modello che mostri la disposizione e suddivisione delle stanze che compongono l'edificio. La costruzione di tale modello si ispira all'approccio presentato in [68], il cui primo passo consiste nell'analisi di scansioni laser tridimensionali. In questo lavoro di tesi invece il punto di partenza del layer consiste nell'identificare gli elementi geometrici che rappresentano le pareti dell'edificio, analizzando una mappa metrica bidimensionale. A partire da questi segmenti viene realizzata una sequenza di operazioni il cui risultato finale consiste in una segmentazione della mappa metrica in stanze.

Il grado di astrazione del layout delle stanze viene ulteriormente incrementato dal grafo topologico, il cui obiettivo risiede nel mostrare la connettività delle regioni di spazio individuate dal layer precedente, ovvero come le stanze sono connesse tra loro. L'individuazione delle connessioni segue l'approccio presentato in [117], in cui viene utilizzata la tecnica del medial axis [19]. Le informazioni sull'ambiente assumono la struttura di un grafo in cui i nodi rappresentano le stanze e gli archi le connessioni dirette tra di esse.

Infine l'ultimo layer esegue una classificazione semantica delle stanze, associando a ciascun nodo del grafo topologico una label che ne indica la funzione (ad esempio ‘ufficio’, ‘cucina’, ‘corridoio’). Come classificatore viene utilizzato extra-trees [32], un metodo di classificazione supervisionato che viene addestrato su un dataset di edifici reali. Dopo aver indotto una funzione che mette in relazione alcune caratteristiche geometriche delle stan-

ze e la relativa etichetta, il classificatore la utilizza per predire una label relativamente ad ogni stanza del layout.

Il lavoro di tesi si ispira al lavoro di [118], anch'esso consistente in un sistema di mapping multilivello, in cui però le rappresentazioni impiegate risultano in parte differenti. In [118] infatti, oltre a mappa metrica e grafo topologico, sono sviluppati navigation graph e mappa concettuale, sostituiti in questo lavoro di tesi da layout delle stanze e mappa semantica.

La valutazione dei risultati viene fatta sia da un punto di vista qualitativo (tramite degli esempi visuali di segmentazione e classificazione semantica), sia da un punto di vista quantitativo (tramite alcune metriche sulle performance globali di segmentazione e classificazione). L'intero sistema è stato implementato in Python.

La tesi è strutturata nel modo seguente.

Nel Capitolo 2 viene presentato lo stato dell'arte relativo ai diversi approcci di mapping integrati nel sistema multilivello sviluppato in questo lavoro di tesi: mappa metrica, layout delle stanze, grafo topologico e mappa semantica.

Nel Capitolo 3 viene descritto l'approccio logico allo sviluppo del sistema di rappresentazione multilivello. Per ciascuna delle rappresentazioni integrate nel sistema viene discussa la finalità e il livello di conoscenza offerto al robot. Sono inoltre presentate le tecniche che permettono di ottenere tali rappresentazioni.

Nel Capitolo 4 viene mostrato il progetto dell'architettura del sistema, implementato in Python. Sono presentati e descritti i vari moduli che lo compongono, incentrandosi sugli aspetti tecnologici e implementativi e ponendo particolare attenzione alle modalità con cui i moduli interagiscono tra loro.

Nel Capitolo 5 viene descritta la valutazione sperimentale, eseguita sia in modo qualitativo (tramite esempi visuali) sia in modo quantitativo (tramite metriche) sulle performance di segmentazione dell'ambiente in stanze e classificazione semantica.

Nel Capitolo 6 vengono tratte alcune conclusioni riguardanti il lavoro svolto e vengono elencati alcuni spunti per sviluppi futuri.

In Appendice A sono mostrati in modo esaustivo tutti i risultati ottenuti dalle sperimentazioni svolte.



## Capitolo 2

# Stato dell'arte

In questo capitolo viene fornita una descrizione riguardante lo stato dell'arte del mapping, nell'ambito della robotica mobile autonoma. Il mapping consiste nell'abilità, da parte di un robot mobile, di costruire un modello spaziale dell'ambiente fisico circostante, tipicamente rappresentato sotto forma di mappa. Per generare una mappa, è necessario che il robot possieda sensori che gli permettano di percepire il mondo esterno. La costruzione della mappa viene compiuta attraverso un'analisi dei dati sensoriali raccolti. Gli approcci di mapping discussi sono inclusi nel sistema multilivello sviluppato nel lavoro di tesi. Seguendo un grado di astrazione crescente, un ambiente può essere rappresentato tramite: mappa metrica, layout delle stanze, grafo topologico e mappa semantica. Questi metodi di rappresentazione possono essere dipendenti tra loro, costruiti l'uno sopra all'altro, e possono essere integrati in un'unica mappa multilivello. Viene fornita un'introduzione ai vari metodi di rappresentazione e mapping dell'ambiente, riportando esempi presenti in letteratura, e confrontandone alcuni con il lavoro di tesi.

### 2.1 Il mapping nella robotica mobile autonoma

Nell'ambito della robotica è spesso necessario o utile utilizzare mappe per rappresentare lo spazio in cui opera un robot. Questa pratica acquista considerevole rilevanza nel caso in cui il robot in questione sia un *robot mobile autonomo*. Questo poichè il raggiungimento di un alto livello di autonomia per un robot richiede interazione con l'ambiente circostante. L'interazione a sua volta spesso richiede una mappa, attraverso la quale il robot possa avere una rappresentazione dello spazio entro cui può muoversi. Di conseguenza, una condizione necessaria per il raggiungimento dell'autonomia da parte di un robot mobile è il possesso di un'efficace rappresentazione dell'ambiente

circostante, sotto forma di mappa. Nel lavoro di tesi, come ambienti di lavoro per il robot sono stati considerati esclusivamente ambienti *indoor*, ovvero edifici. Parallelamente al conseguimento della proprietà di autonomia, la mappa permette o facilita lo svolgimento di alcuni task, come *localizzazione*, *esplorazione* e *path planning*, di seguito discussi.

- **Localizzazione:** consiste nell'abilità del robot di stabilire la propria posizione ed il proprio orientamento nel sistema di riferimento. Diversi approcci per la localizzazione sono descritti in [55], [8] e [108]. Nel caso in cui la mappa non sia nota a priori, può essere ottenuta parallelamente al processo di localizzazione, attraverso la tecnica di *SLAM* (*Simultaneous Localization and Mapping*). Per mezzo di questa tecnica un robot si muove in un ambiente sconosciuto, costruisce la mappa di tale ambiente ed allo stesso tempo è capace di localizzarsi all'interno della mappa. I lavori in [30], [109] e [20] descrivono alcuni metodi per lo SLAM, spiegandone il funzionamento e l'utilità. In questo lavoro di tesi si farà l'assunzione di non conoscere inizialmente la mappa. Per tale motivo saranno utilizzate tecniche di SLAM.
- **Esplorazione:** è l'insieme degli approcci che permettono al robot di decidere come esplorare l'ambiente incognito. Nel caso in cui sia richiesto il coordinamento tra più robot, si possono migliorare le performance di esplorazione di ambienti ignoti, applicando approcci descritti in lavori come [2], [95], e [97]. Ad esempio, dovendo esplorare un ambiente indoor in un tempo limitato, è preferibile percorrere prima i corridoi piuttosto che esplorare tutte le stanze singolarmente.
- **Path planning:** è la capacità del robot di pianificare un percorso, dalla posizione corrente del robot a una posizione di goal. Approcci di path planning sono descritti in [97] e [5]. L'obiettivo del task è quello di individuare il percorso migliore, per esempio il percorso più breve, riducendo il rischio di collisioni con gli altri oggetti presenti nell'ambiente.

Per soddisfare questi task ed ottenere autonomia, il robot necessita di una rappresentazione del mondo circostante. Per svolgere quest'operazione possono essere adoperati diversi paradigmi.

## 2.2 Mappe multilivello

Nel lavoro di tesi è stato realizzato un sistema basato su una rappresentazione multilivello dell'ambiente, che incorpora una serie di diverse mappe,

correlate e connesse tra loro. I diversi approcci di rappresentazione sono stati sviluppati come *layer*, costruiti l’uno sopra all’altro ed integrati in una rappresentazione basata su mappe multiple. Lo spazio è modellato a diversi livelli di astrazione, che variano da mappe metriche di basso livello per la localizzazione e la navigazione del robot, a mappe che forniscono invece una categorizzazione concettuale dello spazio simile a quella umana. In questi casi, come in [118], si parla di *sistemi multilivello*. Attraverso questo sistema è possibile ottenere in maniera semplice un alto numero di informazioni concernenti l’ambiente in cui si muove un robot mobile autonomo. I layer integrati in un sistema multilivello possono essere di diversi tipi.

Nel lavoro presentato in [118], gli approcci di rappresentazione sono i seguenti:

- **Mappa metrica:** è il livello più basso del modello spaziale. La mappa metrica supporta la localizzazione del robot, e viene ottenuta analizzando i dati sensoriali relativi all’ambiente, raccolti dal laser range scanner montato sul robot. A tal scopo viene utilizzato il modulo SLAM, il quale estrae primitive geometriche dalle scansioni laser in modo da allineare gli scan acquisiti e costruire una mappa metrica dell’ambiente. Le primitive geometriche usate nell’approccio consistono in linee, che corrispondono tipicamente a muri ed altri ostacoli presenti nell’ambiente indoor in cui si muove il robot. Poichè i muri sono statici, queste feature geometriche sono utilizzate per mantenere il robot localizzato. Le linee sono salvate in una mappa metrica globale con un sistema di riferimento assoluto, di cui la Figura 2.1 mostra un esempio.
- **Navigation graph:** viene costruito in cima alla mappa metrica. Stabilisce un modello dello spazio libero basato sulla sua connettività e raggiungibilità. Il robot, navigando attraverso l’ambiente, genera dei marcatori dello spazio libero, o *nodi di navigazione*. Questi sono dei nodi posti entro lo spazio libero, e rappresentano la traiettoria percorsa dal robot durante la navigazione. Ogni marcitore viene creato non appena il robot ha percorso una certa distanza dal marcitore esistente più vicino. I marcatori sono poi connessi seguendo l’ordine in cui sono stati generati.

In questo layer la rappresentazione spaziale viene ulteriormente arricchita con informazione semantica relativa all’ambiente. Questa classificazione semantica associa una porzione di spazio ad un termine che ne indica la funzionalità o altre caratteristiche. Dai laser range da-

ta vengono derivate altre feature, oltre alle linee rappresentate nella mappa metrica. Queste feature sono utilizzate per classificare semanticamente le *pose* del robot (combinazione di posizione e orientamento) nel corso della navigazione. Alle pose del robot vengono associate tre classi semantiche, che si considerano essere presenti in ogni ambiente indoor: stanza, corridoio e porta. La classe porta indica la transizione tra due diverse regioni spaziali, ed è individuata quando il robot attraversa un passaggio relativamente stretto. Le classi stanza e corridoio sono invece individuate estraendo l'aspetto geometrico degli ambienti e la loro approssimazione poligonale. La classificazione dei nodi di navigazione viene derivata da quella delle pose del robot. Per ogni nodo viene salvata in una memoria a breve termine la classificazione delle ultime  $N$  pose. In base alla maggioranza di queste classificazioni, il nodo di navigazione viene etichettato semanticamente. La Figura 2.2 mostra un esempio di navigation graph generato attraverso questo approccio, sovrapposto alla corrispondente mappa metrica. Il navigation graph è visivamente rappresentato dai nodi di navigazione, ovvero dai punti posti entro lo spazio libero, collegati tramite segmenti. Colori diversi rappresentano aree diverse, separate da porte, le quali sono simboleggiate da stelle rosse.

- **Mappa topologica:** divide l'insieme dei nodi di navigazione in aree, ognuna delle quali consiste in un insieme di nodi interconnessi. Le aree sono separate da un nodo di navigazione classificato come porta. Questo layer di astrazione corrisponde a una segmentazione qualitativa di un ambiente indoor in regioni differenti, classificate come stanze o corridoi. La categoria semantica di un'area viene determinata in base alla maggioranza delle classificazioni dei nodi di navigazione appartenenti all'area in questione. Le aree topologiche sono passate alla mappa concettuale, in cui sono rappresentate come istanze delle rispettive categorie.
- **Mappa concettuale:** contiene un'ontologia concettuale che definisce categorie astratte per stanze ed oggetti, e mostra come queste categorie sono collegate. Inoltre l'informazione estratta dai dati sensoriali viene rappresentata come token che instanziano i concetti astratti. Un esempio di mappa concettuale è mostrato nella Figura 2.3.

Il sistema sviluppato in questo lavoro di tesi mostra analogie con quello appena descritto (presentato in [118]), in quanto entrambi i sistemi si basano su di una rappresentazione multilivello dell'ambiente. Tuttavia i layer



Figura 2.1: Il modulo SLAM in [118] crea una mappa metrica in cui le linee rappresentano muri e altre superfici dell'ambiente.

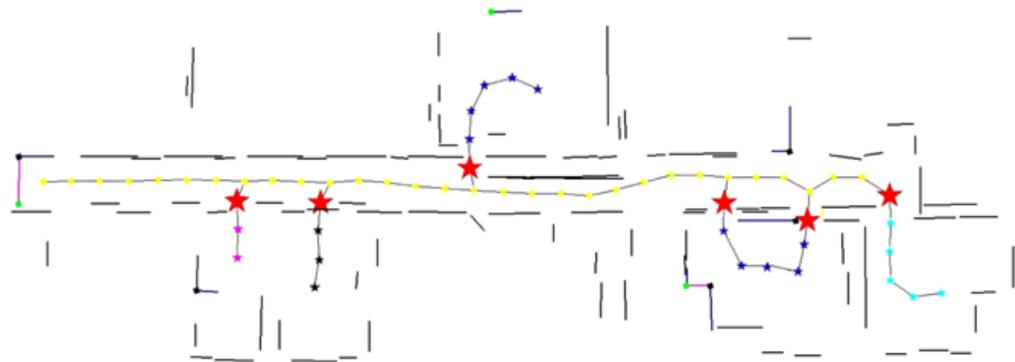


Figura 2.2: Navigation graph in [118], sovrapposto alla corrispondente mappa metrica.

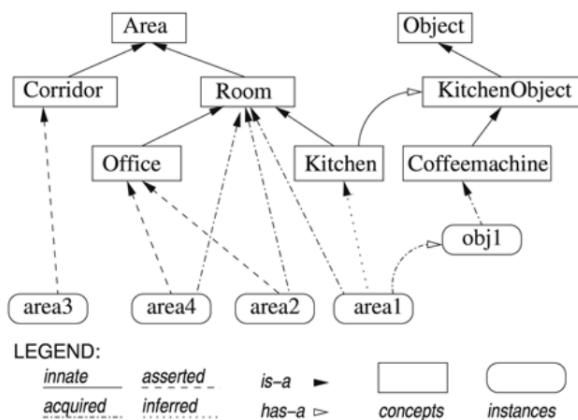


Figura 2.3: Esempio di mappa concettuale in [118]. La mappa combina diversi tipi di conoscenza.

di rappresentazione integrati nel sistema multilivello sviluppato in questo lavoro di tesi sono differenti da quelli presentati in [118].

Nel lavoro di tesi gli approcci di mapping che costituiscono i layer sono, secondo un livello di astrazione crescente: mappa metrica, layout delle stanze, grafo topologico e mappa semantica.

In breve: partendo da una rappresentazione metrica di basso livello, viene costruito un layer che riporta la suddivisione dell'ambiente in stanze. Successivamente il grado di astrazione viene incrementato attraverso la costruzione di un grafo topologico, in cui i nodi rappresentano le stanze e gli archi rappresentano le connessioni tra di esse. Infine l'ultimo layer classifica semanticamente le stanze, associando ai nodi del grafo topologico una label semantica che indica la tipologia della regione di spazio (stanza di servizio, corridoio, ecc.).

Viene di seguito fornita una descrizione dello stato dell'arte di ciascuno di questi approcci di rappresentazione, riportando esempi presenti in letteratura e confrontandone alcuni con il lavoro di tesi.

### 2.3 Mappa metrica

La *mappa metrica* offre una rappresentazione dello spazio attraverso cui le zone occupate sono distinte da quelle libere. La mappa codifica l'occupazione e la non occupazione dell'ambiente, offrendo un basso grado di astrazione. È possibile rappresentare una mappa metrica in svariati modi.

Una comune tipologia di mappa metrica è la *mappa a griglia*, chiamata anche *occupancy grid map*, in cui lo spazio viene rappresentato come una matrice bidimensionale. Ad ogni cella viene assegnato un valore compreso tra zero e uno. Questo valore indica la probabilità che la cella sia occupata o meno da un oggetto. La Figura 2.4 mostra un esempio di mappa a griglia. Le celle identificate come occupate sono colorate di nero; al contrario, a quelle libere è associato il colore bianco.

In [25], [65], [84] e [105] sono descritti approcci per generare mappe a griglia.

Una rappresentazione metrica alternativa può essere ottenuta rilevando feature primitive dello spazio, come punti o segmenti, tramite sensori. Lavori come [44], [50], [71] e [118] offrono riferimenti sulla creazione di segmenti a partire dai punti ottenuti da scansioni laser. Queste linee corrispondono a muri ed altre strutture piane dell'ambiente. Un esempio di questa tipologia di mappa metrica è mostrato nella Figura 2.1.



Figura 2.4: Esempio di occupancy grid map tratta da [65]

Le informazioni sull'occupazione dell'ambiente possono essere utilizzate per la localizzazione del robot o per pianificare una traiettoria, evitando gli ostacoli. Tuttavia, non offrono alcun tipo di informazione su ciò che una porzione di spazio rappresenta. L'unica informazione memorizzata indica se una determinata regione sia libera o meno, a prescindere dalla natura dell'elemento che la occupa.

## 2.4 Layout delle stanze

Il layout delle stanze è una rappresentazione dell'ambiente che mostra la disposizione e la suddivisione delle varie stanze. In questa rappresentazione, le stanze sono individuate come elementi separati l'uno dall'altro. Il modello risultante identifica la suddivisione dello spazio in varie regioni, esplicitando le varie stanze che compongono l'edificio, come mostrato in Figura 2.5.

Il processo che sta alla base di questa rappresentazione è la *segmentazione*. Questo processo consiste nel partizionamento di una mappa in più segmenti, con lo scopo di estrarre da essa informazioni relative alla suddivisione in stanze. La segmentazione assegna una label a ogni pixel, in modo che pixel con stessa label condividano certe caratteristiche. L'output della fase di segmentazione consiste in una suddivisione dei dati in ingresso, corrispondente al partizionamento dello spazio interno in regioni distinte. La segmentazione può basarsi su svariati approcci; vengono di seguito presentati i più rilevanti.

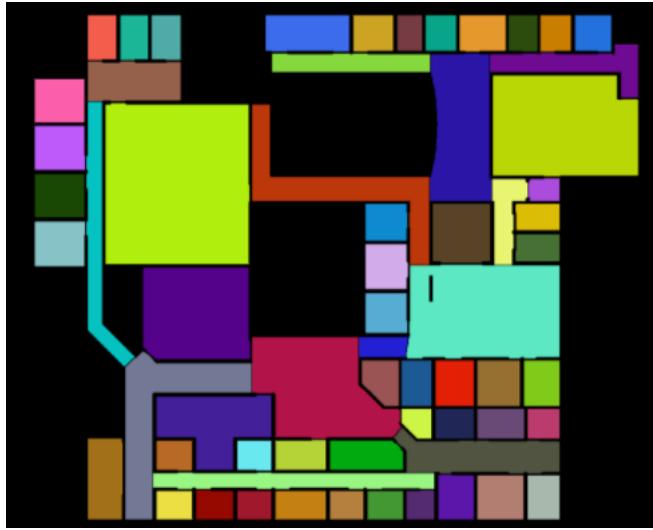


Figura 2.5: Esempio di layout delle stanze tratto da [9].

#### 2.4.1 Approcci basati su point-cloud

Esistono dei metodi di segmentazione che trovano le proprie fondamenta nell'analisi dei *point-cloud*. I point-cloud corrispondono ai dati sensoriali raccolti; consistono infatti negli insiemi di punti scansionati dell'ambiente, ottenuti tramite uno scanner laser.

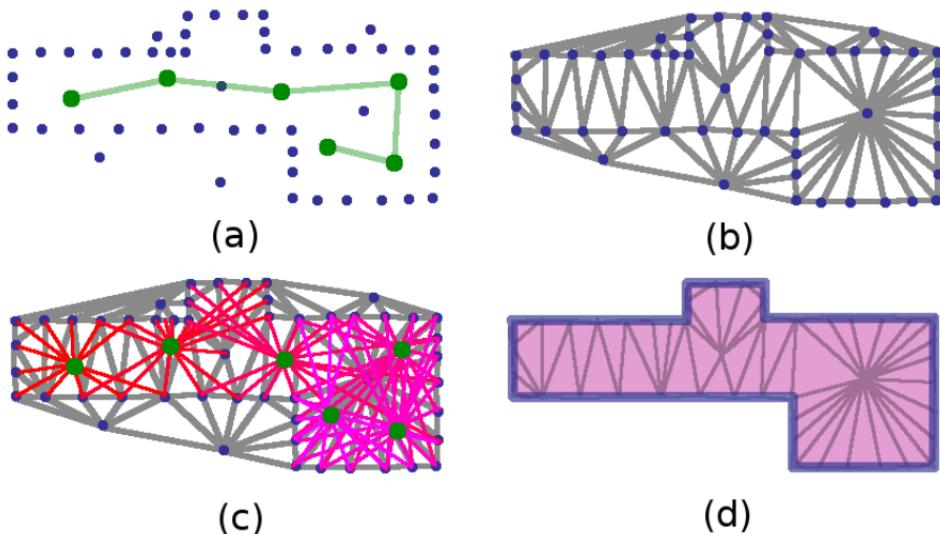
In [110] e [111] viene presentato un approccio di questo tipo. Partendo da 3D point-cloud catturati da sistemi di mobile mapping, si ottengono modelli 2.5D di interni di edifici. Col termine 2.5D si intende una grafica bidimensionale in cui l'asse  $Z$  è fisso, creando di conseguenza l'illusione della tridimensionalità. I modelli ricostruiti raffigurano gli elementi architettonici dell'edificio, come pavimenti, muri e soffitti, e mostrano il layout delle stanze dell'edificio. In entrambi i metodi l'input dell'algoritmo è un insieme di campionamenti dei punti appartenenti ai muri, in cui ogni campione è associato alla posizione dello scanner che lo ha osservato, e alla stima dell'altezza del soffitto. I point-cloud sono proiettati da uno spazio tridimensionale a uno bidimensionale, e l'ambiente viene partizionato in dominio interno ed esterno. Come mostrato nella Figura 2.6, il dominio interno rappresenta l'open space dell'ambiente, come stanze e corridoi, mentre il dominio esterno rappresenta lo spazio al di fuori dell'edificio, lo spazio occupato da oggetti solidi e lo spazio non osservabile.

Il partizionamento può essere eseguito utilizzando i campioni di punti appartenenti a muri per generare una *triangolazione di Delaunay* sul piano,

procedimento spiegato in [54]. La triangolazione di Delaunay, dato un insieme di punti, suddivide lo spazio in una serie di triangoli, il cui insieme di vertici corrisponde all'insieme dei punti in ingresso. Garantisce inoltre che nessuno dei punti dati sia incluso nella circonferenza circoscritta di alcun triangolo.

Inizialmente ogni triangolo è etichettato come esterno. Per ogni scansione nel tempo, vengono considerati i segmenti che vanno dalla posizione della scansione, ad ogni campione di muro associato a tale scansione. Se un triangolo è intersecato da uno di questi segmenti, allora viene etichettato come interno, poiché la scansione non è stata occlusa da alcun oggetto solido. Il risultato di questo partizionamento è una planimetria, ovvero una rappresentazione in piano dell'edificio, in cui il confine tra dominio interno ed esterno corrisponde ai muri.

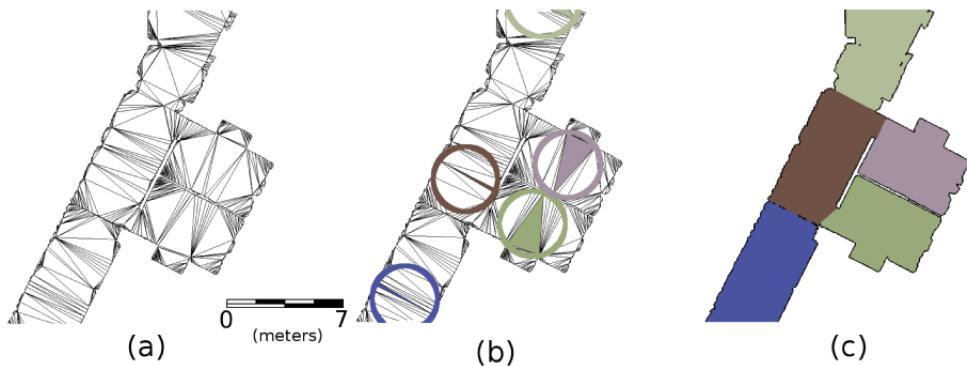
A questo punto vengono modellate le singole stanze. La suddivisione in



*Figura 2.6: Individuazione dello spazio interno in [110]: (a) Campioni di muri (in blu) e percorso dello scanner (in verde); (b) Triangolazione di Delaunay dei campioni di muri; (c) Scansioni laser da ciascuna posizione (in rosso); (d) Triangoli che intersecano le scansioni laser (in rosa), identificati come triangoli interni, e perimetro del modello dell'edificio (in blu).*

stanze è trattata come un problema di *graph partitioning*, di cui in [26] vengono descritti e confrontati vari approcci. Il graph partitioning consiste in una partizione dei vertici di un grafo in sottoinsiemi disgiunti. Come mostrato in Figura 2.7, inizialmente vengono individuati i *seed triangle*, ossia triangoli la cui circonferenza circoscritta è un massimo locale. Ogni seed

triangle rappresenta una stanza. Successivamente ogni triangolo viene considerato come nodo di un grafo. Il peso dell'arco tra due triangoli adiacenti consiste nella lunghezza del lato condiviso. Allo scopo di suddividere il dominio interno in stanze, viene operato un *minimum cut* sul grafo, processo descritto in [38] e [100]. Il minimum cut partiziona i nodi del grafo, minimizzando il peso degli archi tagliati. Questo processo coincide con una minimizzazione della lunghezza dei confini tra le stanze, che sono quindi definite in modo da minimizzare le dimensioni delle porte.



*Figura 2.7: Partizionamento delle stanze in [110]: (a) Triangolazione interna; (b) Seed triangle e corrispondenti circonferenze circoscritte; (c) Label delle stanze propagate a tutti gli altri triangoli.*

Anche il metodo presentato in [78] si basa su point-cloud e mira alla costruzione di un modello dell'ambiente usando scansioni indoor. Vengono derivate entità architettoniche di alto livello come stanze e porte, ottenendo dunque un layout dell'edificio con stanze separate, come mostrato in Figura 2.8. Partendo da un point-cloud 3D ottenuto in precedenza, viene modellata in modo probabilistico l'affiliazione di ogni punto a una determinata stanza nell'edificio. A questo scopo viene sviluppato un algoritmo di clustering probabilistico, che sfrutta la conoscenza dell'appartenenza di ogni punto misurato ad una particolare scansione. Ogni punto è inizialmente etichettato con l'indice della scansione da cui proviene. Il problema del partizionamento viene risolto usando un algoritmo iterativo, che fa affidamento sulle visibilità stimate tra ogni coppia di pose entro il point-cloud. Questo è dovuto all'assunzione che la maggior parte dei punti appartenenti a una stanza possano essere visti da un punto arbitrario appartenente alla medesima stanza. Ciò significa che due punti verosimilmente appartengono alla stessa stanza se sono stati individuati dalla medesima scansione. Tuttavia quest'assunzione non sempre è valida, e infatti il metodo produce risultati

inesatti nel caso in cui vi siano stanze fortemente non convesse, come un corridoio a forma di ‘T’. In questa circostanza, molti punti del corridoio sono erroneamente assegnati a stanze adiacenti.

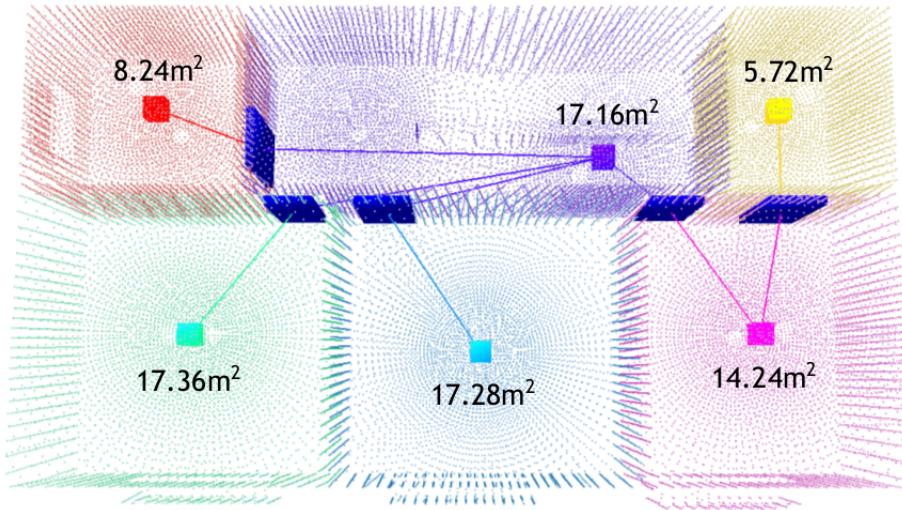


Figura 2.8: Esempio di layout delle stanze ricostruito da [78].

Il metodo presentato in [79] consiste in un'estensione di [78]. L'approccio si basa su point-cloud indoor, e mira alla ricostruzione di modelli 3D di edifici. Analogamente a [78], per ottenere una clusterizzazione iniziale dei punti scansionati, ogni punto è etichettato con l'indice della scansione in cui è generato. Allo stesso modo si basa sull'assunzione che la maggior parte dei punti appartenenti a una stanza possano essere visti da un punto arbitrario della stessa stanza. Tuttavia questa segmentazione iniziale viene ulteriormente estesa. Prima di tutto, vengono rese possibili operazioni di modifica di alto livello, in termini di rimozioni di muri o di rimodellamento di stanze. Queste operazioni portano a una rappresentazione topologica consistente. In secondo luogo, per mezzo di questo metodo è consentito considerare facilmente misure come spessore dei muri o area delle stanze.

#### 2.4.2 Approcci basati su immagini RGBD

Altri approcci per ottenere il layout delle stanze si basano su *immagini RGBD*. Le camere RGBD sono sistemi sensoriali in grado di catturare immagini RGB (Red Green Blue), insieme ad informazioni riguardanti la profondità di ogni pixel. In [80] viene descritta questa tipologia di dato visivo,

e vengono discusse possibili applicazioni.

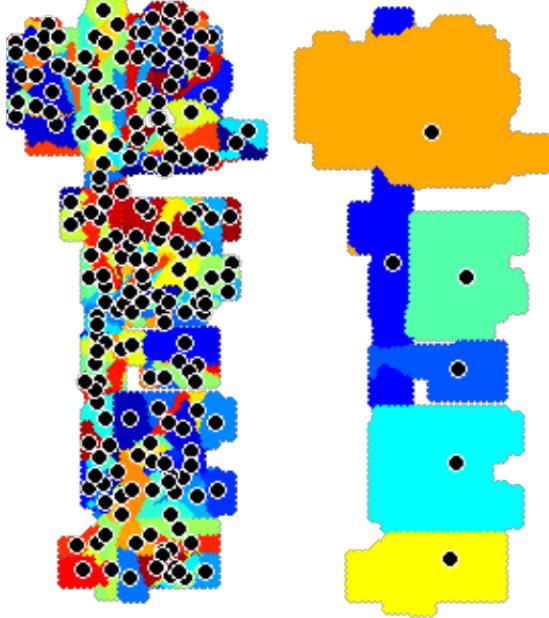
Un esempio di approccio basato su immagini RGBD si può trovare in [42], in cui la geometria di un ambiente è rappresentata come un grafo, nel quale i nodi corrispondono a elementi strutturali come stanze, muri e oggetti, e gli archi rappresentano la loro relazione geometrica. L'approccio di [42] si avvale di una grammatica strutturale, che definisce una lista di possibili trasformazioni del grafo. La grammatica guida un algoritmo di ricostruzione, in cui le regole grammaticali vengono applicate sequenzialmente, per generare un modello strutturato. Le regole grammaticali le cui precondizioni sono soddisfatte sono applicate, finché nessuna regola può essere più applicata. Una di queste regole grammaticali consiste nella segmentazione delle stanze, ed è l'unica regola applicabile inizialmente. La segmentazione viene formulata come un problema di clustering, ovvero come raggruppamento di elementi omogenei in un insieme di dati. Il clustering viene conseguito utilizzando l'algoritmo di *k-medoids*, come mostrato in Figura 2.9. Questo è un algoritmo di clustering partizionale che prevede in input un insieme di  $n$  oggetti, ed un numero  $k$  che determina quanti cluster si vogliono in output. L'algoritmo cerca di minimizzare *l'errore quadratico medio*, ossia la distanza tra i punti di un cluster ed il punto designato per esserne il centro. Un medoid può essere definito come un oggetto di un cluster, la cui dissimilarità media rispetto a tutti gli altri oggetti nel cluster è minima. In questo modo esso sarà il punto più centrale di un dato dataset.

Un'altra importante regola grammaticale è la ricostruzione delle stanze, che ottimizza la forma delle stanze diminuendo il numero di vertici.

Nel lavoro di tesi ci si discosta sia dagli approcci basati su point-cloud delle scansioni laser, sia dagli approcci basati su immagini RGBD. Non vengono infatti utilizzati tali elementi come base per operare la segmentazione ed ottenere il layout delle stanze.

L'approccio del lavoro di tesi mostra invece analogie con metodi basati su mappa metrica per operare una segmentazione dello spazio. Il punto di partenza consiste in una occupancy grid map o in una planimetria.

In [9] sono elencati una serie di metodi funzionali alla segmentazione basati su mappa metrica, confrontandoli e valutandoli. I metodi sono distinti in base all'approccio su cui poggiano, ovvero segmentazione basata sul *Voronoi graph*, sul *graph partitioning*, sulle *feature*, segmentazione *morfologica*, segmentazione basata sulla *distance transform* e interpretazione di



*Figura 2.9: Segmentazione delle stanze in [42]. Da sinistra a destra: 200 cluster iniziali dell'algoritmo  $k$ -medoids; cluster finali, in cui ogni pixel è colorato in base al cluster center più vicino.*

planimetrie.

#### 2.4.3 Approcci basati su Voronoi graph

L'approccio più utilizzato per segmentare una mappa metrica si basa sull'uso del Voronoi graph generalizzato. Il Voronoi graph è una partizione spaziale di una mappa, tipicamente applicata sulla occupancy grid map. È costituito dall'insieme dei punti dello spazio libero aventi più di un ostacolo alla stessa distanza minima (*basis point*).

I lavori in [106], [105] e [49] utilizzano il Voronoi graph allo scopo di trovare i suoi *punti critici*, ossia i punti corrispondenti a porte e passaggi stretti. I punti critici sono poi collegati ai propri basis point, generando le *linee critiche*. Come mostrato in Figura 2.10, l'occupancy grid map viene così suddivisa in regioni, ottenendo la segmentazione dello spazio.

In modo analogo, quest'algoritmo viene applicato al metodo di segmentazione della mappa e di mapping multi-robot incrementale presentato in [117]. Qui il Voronoi graph viene costruito sulla *distance map*, una mappa rappresentata come una matrice, che contiene per ogni cella la distanza dal-

l'ostacolo più vicino.

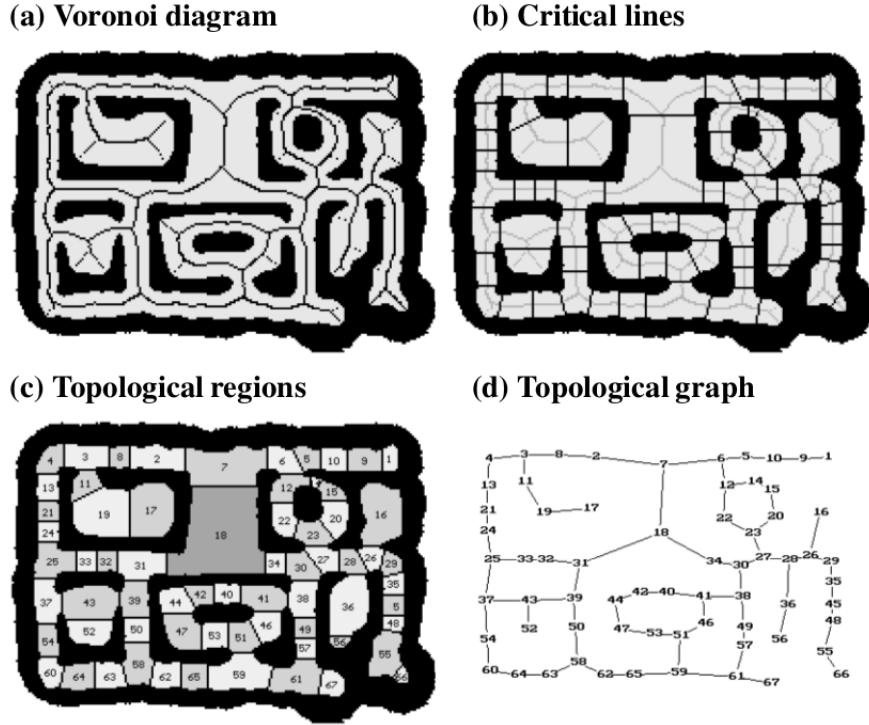


Figura 2.10: Esempio di segmentazione eseguita in [106]: (a) Voronoi graph; (b) Punti e linee critiche; (c) Regioni separate; (d) Grafo topologico estratto.

Una significativa estensione del metodo di segmentazione basata sul Voronoi graph viene introdotta in [7]. Qui viene impiegato l'*extended Voronoi graph* (EVG), un Voronoi graph che rimane in prossimità dei muri, come mostrato in Figura 2.11. L'ambito in cui questo approccio apporta il vantaggio più rilevante è l'esplorazione di aree le cui dimensioni superano l'orizzonte sensoriale del robot.

In questi scenari il Voronoi graph generalizzato non produce risultati ottimali, poiché invece di definire percorsi paralleli ai muri, definisce un percorso che volge al centro dell'area. Seguendo questo percorso, il robot si ritroverebbe in una posizione da cui, a causa del suo limitato orizzonte sensoriale, non sarebbe visibile alcun ostacolo. Di conseguenza, da questa posizione sarebbe impossibile per il robot calcolare ulteriormente il Voronoi graph a partire dai dati sensoriali locali.

Utilizzando l'extended Voronoi graph viene invece offerta al robot una configurazione spaziale caratterizzata da percorsi prossimi ai muri nelle aree di grandi dimensioni. In questo modo viene garantita la possibilità di acquisire

dati sensoriali locali, funzionali all'orientamento.

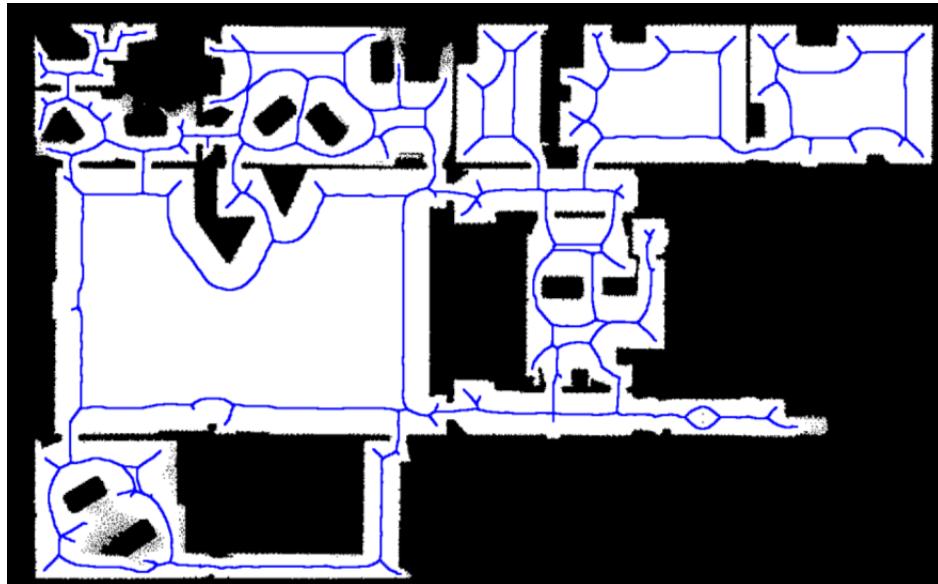


Figura 2.11: Esempio di extended Voronoi graph utilizzato in [7].

Un approccio basato sul concetto di Voronoi graph viene utilizzato anche in questo lavoro di tesi. La differenza sostanziale rispetto ai metodi citati sopra risiede nello scopo del suo utilizzo. L'obiettivo del suo impiego non è infatti quello di ricostruire il layout, bensì quello di trovare le connessioni tra le stanze, dopo che queste sono già state individuate come elementi separati. Nel lavoro di tesi questo approccio non apporta dunque alcun contributo alla fase di ricostruzione del layout delle stanze, ma permette invece di individuare le connessioni tra le stanze, rappresentate poi dagli archi del grafo topologico.

#### 2.4.4 Approcci basati su graph partitioning

Un altro possibile approccio per segmentare l'ambiente di lavoro di un robot consiste nell'utilizzo del graph partitioning. Il graph partitioning è un processo attraverso il quale si cerca di partizionare un grafo in componenti più piccole, dotate di proprietà specifiche.

In [120] viene applicato il graph partitioning per ottenere una gerarchia di mappe topologiche. Viene preso in considerazione un grafo di navigazione delle aree accessibili, con le relative connessioni, e viene suddiviso tale grafo

per mezzo di graph partitioning.

In [12] viene proposto un sistema per creazione incrementale online di mappe topologiche, e viene calcolato il graph partitioning con l'uso di tecniche di spectral clustering, il cui risultato è mostrato in Figura 2.12. Lo spectral clustering, presentato in [76], è una tecnica di partizionamento che utilizza gli autovalori della matrice di similarità dei dati per trovare una partizione del grafo, in cui il numero di nodi di ogni partizione è simile e bilanciato.

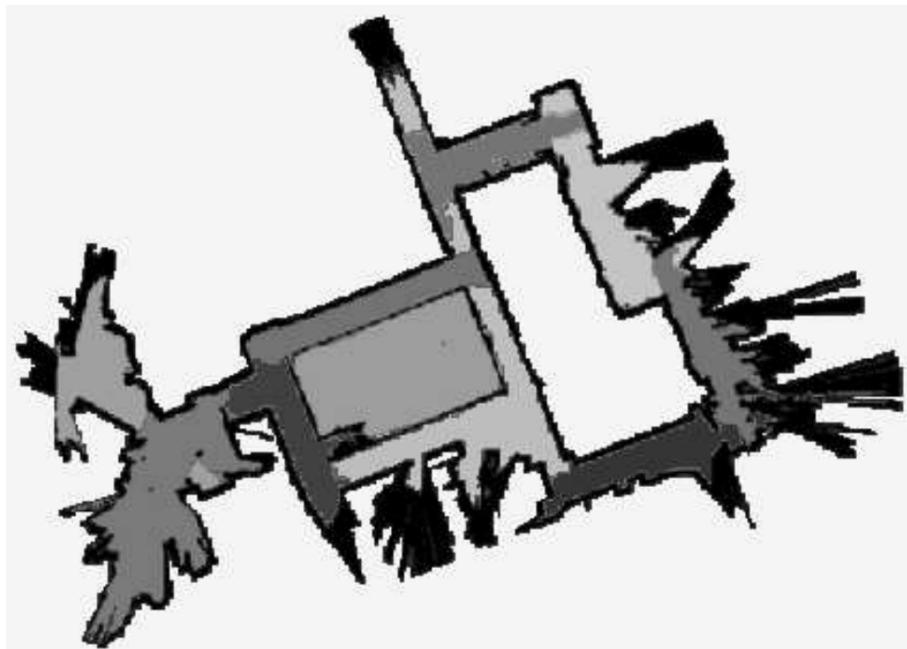


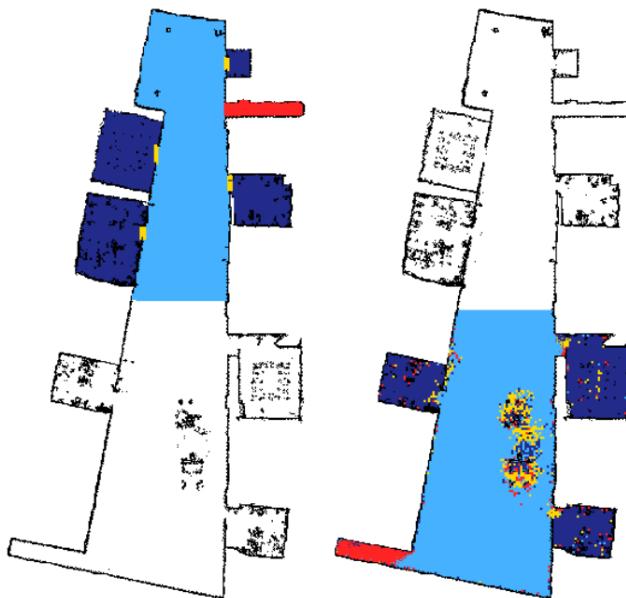
Figura 2.12: Esempio di mappa segmentata in diverse sottomappe (diversi gradi di grigio) tramite un algoritmo di spectral clustering, tratto da [12].

#### 2.4.5 Approcci basati su feature

Gli algoritmi di segmentazione delle stanze basati sulle feature fanno tipicamente affidamento sull'apprendimento di caratteristiche estratte dall'aspetto locale della mappa a griglia.

I lavori descritti in [66] e [65] propongono l'utilizzo di un *classificatore AdaBoost*, il quale opera una mappatura da uno spazio di feature a un insieme di etichette. L'algoritmo di AdaBoost prende come input un *training set*

di esempi, ognuno con una label associata. Poichè l'algoritmo è designato per la classificazione binaria, la label associata ad ogni esempio indica se l'esempio è positivo (1) o negativo (0). Utilizzando un ampio insieme di semplici feature, calcolate direttamente sulle misure del laser scanner, ad ogni punto della mappa viene associata una label spaziale come stanza, corridoio o porta, come mostrato in Figura 2.13



*Figura 2.13: Esempio di classificazione con AdaBoost in [65]. Da sinistra a destra: dati di training (colorati) e di test (bianchi) dall'ambiente con tre classi; risultato della classificazione dei dati di test.*

In [28] viene integrato l'approccio di [66] e [65] con una segmentazione tramite Voronoi. Lo scopo è quello di etichettare ogni posizione nella mappa metrica dell'ambiente con il tipo di regione a cui appartiene. Queste label forniscono una segmentazione intrinseca della mappa, che rappresenta la struttura topologica dell'ambiente.

Il sistema di partizionamento di [22] cerca di separare le stanze apprendendo l'aspetto delle porte per mezzo di feature lineari e puntuali. Ogni qualvolta il robot attraversa un passaggio stretto, viene ipotizzato l'attraversamento di una porta, la quale separa due stanze. Per mezzo di questa strategia viene ottenuto un partizionamento della mappa.

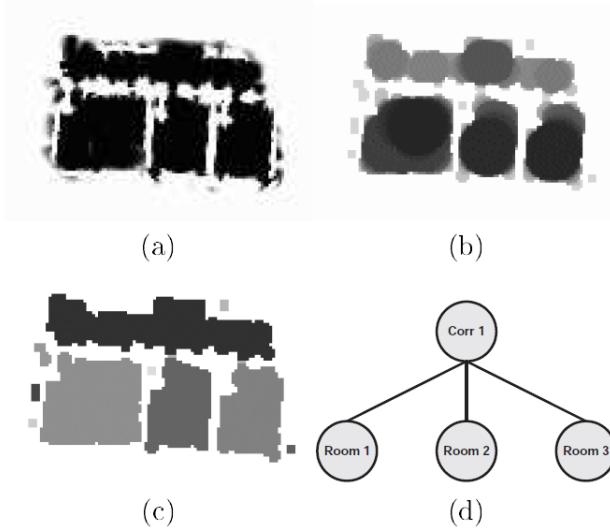
Sviluppando un approccio analogo, in [77] vengono usate forme geometriche più complesse, come rettangoli ruotati, per segmentare una data mappa e catturare strutture regolari delle stanze come uffici e corridoi.

Il metodo presentato in [96] descrive la segmentazione automatica in unità semantiche di una mappa indoor bidimensionale. L'assunzione fondamentale adottata è che le proprietà funzionali siano la chiave per suddividere il mondo in regioni separate. Ad esempio ciò che caratterizza una cucina non è il suo aspetto, bensì il fatto che sia uno spazio in cui il cibo viene preparato e consumato. Un modello basato sull'aspetto potrebbe fallire nel riconoscimento di una cucina nel caso in cui questa abbia un design non convenzionale. Ma se un robot fosse in grado di valutare una funzione "cucina", migliorerebbe la sua abilità di generalizzare e di comunicare efficientemente con l'uomo. Il metodo cerca di utilizzare aspetti funzionali dello spazio per ottenere una suddivisione delle aree di mappe 2D. L'algoritmo assegna label funzionali come stanza, corridoio, cucina e ufficio, ottimizzandole attraverso un framework di *energy maximization*. Questo framework associa ad ogni label un livello di energia, che rappresenta quanto correttamente quella label descrive il relativo spazio. Un più alto livello di energia corrisponde ad un migliore labeling.

#### 2.4.6 Approcci basati su operatori morfologici

In [88] e [89] viene utilizzata una mappa a griglia *fuzzy*, di cui la Figura 2.14 (a) mostra un esempio. Essa rappresenta l'occupazione delle celle da parte di ostacoli. Dettagli su come costruire mappe a griglia fuzzy partendo dai dati sensoriali possono essere trovati in [24] e [82]. Questa tipologia di mappa è definita come un array bidimensionale di celle, a cui è associato un valore reale nell'intervallo  $[0, 1]$  che rappresenta il grado di occupazione di quella porzione di ambiente. Vengono in principio applicati in sequenza *dilation* ed *erosion*, degli operatori morfologici descritti in [41]. Questi sono in grado di estrarre dalla mappa a griglia fuzzy le forme degli open space di grandi dimensioni. Il risultato è una nuova mappa a griglia, mostrata in Figura 2.14 (b), in cui il valore di ciascuna cella rappresenta quanto verosimilmente quella cella appartenga ad un ampio open space. L'informazione morfologica rappresentata da questa nuova mappa è ancora suddivisa in piccole porzioni. Per catturarne la struttura topologica viene utilizzato il *watershed algorithm*, un altro operatore morfologico descritto in [114] che segmenta la mappa in un insieme di componenti connessi tramite stretti passaggi come porte. Partendo da un punto dentro ad ogni area, l'algoritmo determina quali celle

appartengono alla stessa stanza. Il risultato è mostrato in Figura 2.14 (c), e la relativa mappa topologica in Figura 2.14 (d).



*Figura 2.14: Esempio di segmentazione tratto da [88]: (a) Mappa a griglia fuzzy; (b) Ristultato dell'applicazione di dilation ed erosion; (c) Segmentazione ottenuta utilizzando il watershed algorithm; (d) Corrispondente mappa topologica.*

#### 2.4.7 Approcci basati su distance transform

In [20] viene descritto un metodo semi-automatico di segmentazione delle mappe basato sul *distance transform*. Il distance transform è una rappresentazione che associa ad ogni cella la distanza dall'ostacolo più vicino. La Figura 2.15 mostra un esempio di distance transform, in cui più una cella è lontana dagli ostacoli più il suo colore è chiaro.

Si suppone che il robot segua l'utente, il quale setta label spaziali in diverse posizioni. Dopo il completamento del mapping, il distance transform viene utilizzato per raggruppare celle della mappa in stanze. Vengono individuate le celle del distance transform il cui valore di distanza rappresenta un massimo locale, tipicamente in prossimità del centro della stanza. Tutte le altre celle non occupate sono associate al massimo locale più vicino, ottenendo la segmentazione della mappa in stanze.

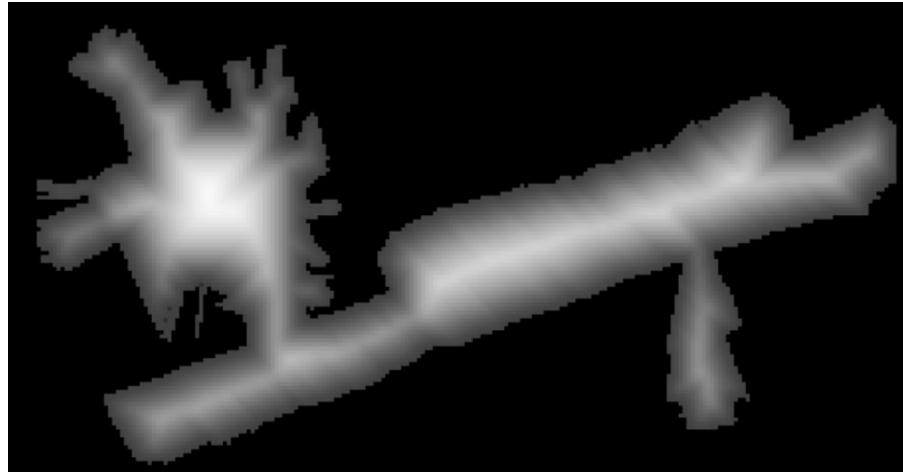


Figura 2.15: Esempio di distance transform tratto da [20].

#### 2.4.8 Approcci basati su interpretazione di planimetrie

Esistono metodi che operano segmentazione basandosi sull'interpretazione di planimetrie. Una planimetria è un disegno in scala che mostra una vista dall'alto delle relazioni tra le stanze, spazi ed altre caratteristiche fisiche di un edificio.

In [1] e [39] è descritto un sistema che prende in input planimetrie con simboli specifici e annotazioni testuali. Il metodo per dividere la mappa in stanze individuali è basato sull'individuazione delle linee, effettuata per trovare i muri, come mostrato in Figura 2.16. I muri sono poi usati per costruire una mappa a griglia. La partizione finale è operata in seguito all'identificazione delle porte nella planimetria.

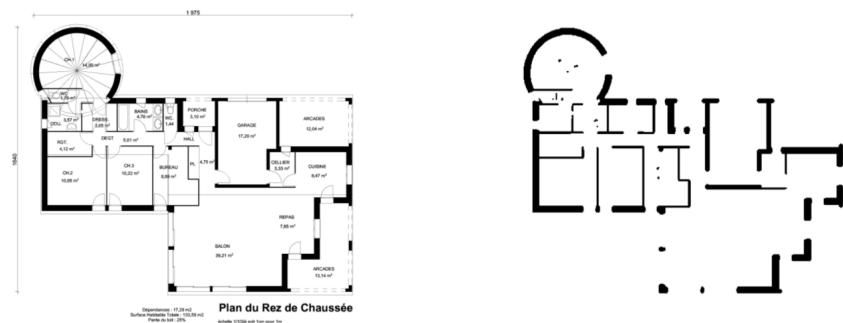


Figura 2.16: Esempio di estrazione di muri tratto da [39]: (sinistra) planimetria in input; (destra) muri estratti.

In [14] viene discusso un sistema simile, che per trovare muri e dividere la planimetria in stanze utilizza in sequenza *Canny edge detection* [13] e *Hough line transform* [43]. L'algoritmo di Canny edge detection è un operatore per il riconoscimento dei contorni in un'immagine, mentre l'Hough line transform è una trasformazione che individua linee rette in un'immagine.

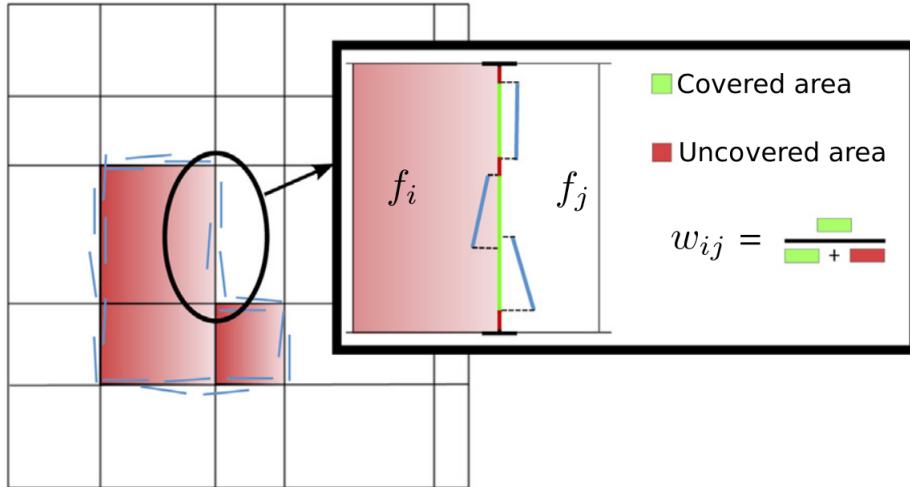
Il lavoro di questa tesi mostra analogie con questi metodi, nel caso in cui l'input sia una planimetria. Nella fase di pulitura di tale mappa vengono infatti eseguiti dei passaggi simili, come estrazione del testo e template matching per trovare le porte. Tuttavia in questi lavori tali operazioni sono finalizzate alla segmentazione, suddividendo le stanze in base ai template delle porte o al testo estratto. Invece nel lavoro di tesi, template e testo non ricoprono alcun ruolo nella fase di segmentazione dello spazio, bensì costituiscono esclusivamente un processo di pulitura preliminare.

#### 2.4.9 Approcci basati su preprocessing dell'ambiente

Un'importante caratteristica che contraddistingue il lavoro di tesi è il fatto di non eseguire una segmentazione sulla mappa metrica così come essa si presenta originariamente. Viene infatti operato un *preprocessing* dell'ambiente. Di seguito sono riportati alcuni metodi che compiono un preprocessing analogo, da cui il lavoro di questa tesi ha tratto ispirazione.

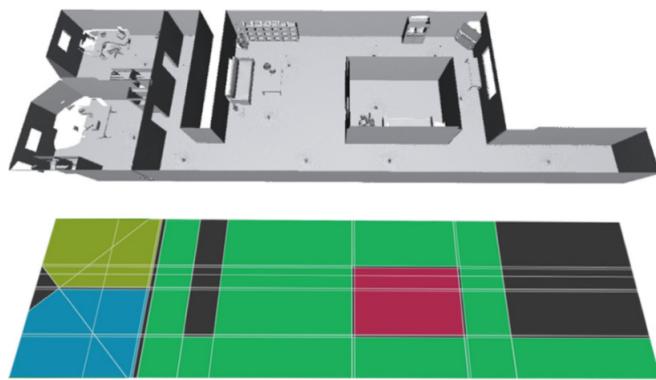
In [68] e [69] viene presentato un approccio robusto per ricostruire la principale struttura architettonica di ambienti indoor complessi. L'input è un insieme di scansioni tridimensionali non ordinate, di cui si assumono come note le pose da cui sono tratte. Il metodo presentato estrae i muri candidati, separandoli dai dati disordinati e riportandoli su di una planimetria bidimensionale sotto forma di segmenti. I muri candidati sono raggruppati in cluster per ottenere un numero limitato di *linee rappresentative* dei muri. Le linee rappresentative tracciate sulla planimetria generano, per mezzo delle loro intersezioni, un insieme di celle. A questo punto la planimetria è dunque costituita da un complesso di celle, generato dalle intersezioni delle linee rappresentative dei muri. L'obiettivo dell'approccio è quello di raggruppare celle adiacenti in stanze. Il primo passo per raggiungere tale obiettivo è pesare i lati (*edge*) delle celle in accordo con la probabilità di essere muri realmente presenti nella mappa. Lo svolgimento di questa operazione è mostrato in Figura 2.17.

Dopo aver pesato i lati delle celle, viene usato un processo di diffusione



*Figura 2.17: Calcolo del peso di un edge del complesso di celle 2D in [68]. I segmenti dei muri candidati (blu) sono proiettati sull'edge, e il rapporto tra la porzione occupata e l'intera lunghezza viene assegnato come peso dell'edge.*

per propagare similitudini tra celle adiacenti che appartengono alla stessa stanza. Il processo di diffusione vede nella propagazione del calore la sua interpretazione naturale. L'intero insieme di celle viene raggruppato in cluster di celle adiacenti in modo da identificare le stanze usando un algoritmo iterativo di suddivisione binaria, il k-medoids. Viene trovato automaticamente il numero esatto delle stanze sfruttando la conoscenza delle posizioni delle scansioni. Nel risultato finale, ogni cluster di celle rappresenta una stanza dell'ambiente, come mostrato in Figura 2.18.



*Figura 2.18: Risultato finale del raggruppamento di celle in stanze in [68]. Ogni insieme di celle adiacenti raffigurate con uno stesso colore, rappresenta una stanza.*

Il lavoro di tesi utilizza le stesse tecniche descritte in questi paper per ottenere una segmentazione dello spazio in diverse stanze. Viene analogamente analizzata la partizione indotta dai muri candidati, propagando similitudini tra celle che appartengono alla stessa stanza, attraverso un processo di diffusione. Tuttavia per eseguire il clustering non viene utilizzata una versione binaria del k-medoids, bensì l'algoritmo di *DBSCAN*, dimostratosi come l'algoritmo di clustering in grado di produrre i risultati più accurati.

Un altro lavoro che opera preprocessing per effettuare la segmentazione ed ottenere il layout delle stanze è [57]. L'obiettivo è offrire una mappa dell'ambiente indoor astratta, annotata semanticamente, ma probabilistica. Questo modello astratto ha la forma di un grafo costituito da stanze e porte, e può essere visualizzato come una classica planimetria. Tale grafo è definito da un vettore  $W$  di parametri che specifica lo stato del mondo che ha generato la occupancy grid map  $M$ . Lo stato del mondo consiste nella struttura dell'ambiente, ovvero il numero di stanze, la loro dimensione e connettività, la posizione delle porte. Viene utilizzato il *maximum posterior approach* [31] per derivare lo stato più probabile  $W^* \in \Omega$  dallo spazio  $\Omega$  dei mondi probabili, data la mappa  $M$ :

$$W^* = \arg \max_{W \in \Omega} p(W|M) \quad (2.1)$$

$p(W|M)$  è la posterior distribution di  $W$  data la mappa nota  $M$ .

Per trovare una soluzione approssimata all'equazione 2.1 si effettua una ricerca nel complesso solution space  $\Omega$ . A questo scopo viene impiegato un *data driven Markov Chain Monte Carlo method (MCMC)*, descritto in [119]. I metodi MCMC sono una classe di algoritmi per il campionamento da distribuzioni di probabilità. Attraverso questa tecnica è possibile catturare la distribuzione di mondi probabili che il robot potrebbe incontrare.

Il modello viene costruito eseguendo in sequenza transizioni di stato da un dato stato del mondo  $W$  a un altro stato  $W'$ , in accordo con una distribuzione di transizione dei kernel della Markov Chain Monte Carlo. I kernel modificano la struttura del mondo e sono organizzati come coppie reversibili, e gli effetti della loro applicazione è mostrata in Figura 2.19.

Le stanze esplorate parzialmente non vengono considerate nel modello finale di layout prodotto da questo metodo. Come si può osservare in Figura 2.20, gli ambienti parziali, simboleggiati da un cerchio rosso, vengono rimossi, e lo spazio in questione viene escluso dalla segmentazione. In questo lavoro di tesi sono invece considerate anche stanze il cui spazio interno non è stato interamente scansionato dal robot. Questi ambienti non vengono ignorati,

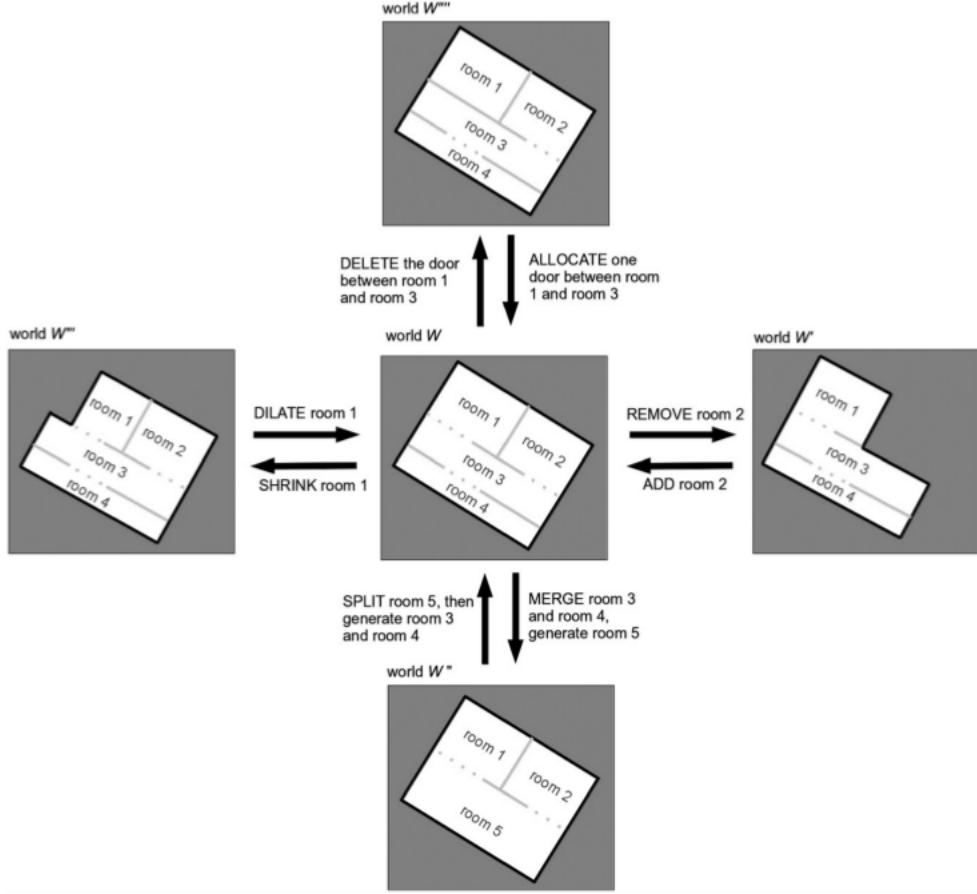


Figura 2.19: Coppie di kernel MCMC reversibili in [57]: ADD/REMOVE, SPLIT/MERGE, SHRINK/DILATE e ALLOCATE/DELETE.

bensì etichettati come parziali, comparendo nel modello finale di layout delle stanze.

Il lavoro in [56] consiste in un'estensione di [57]. Viene proposto un framework per *data abstraction*, ovvero per trovare un modello astratto compatto per i dati di input, usando termini astratti predefiniti. Basandosi su questi termini astratti, il robot acquisisce l'abilità di fare inferenza secondo una specifica *knowledge base*, in modo da gestire meglio la complessità e l'incertezza del mondo reale. Questo framework viene realizzato combinando *Markov logic networks*, descritte in [86], e campionamento con data driven MCMC. Le prime sono un potente strumento per modellare conoscenza incerta, sotto forma di soft rule (formule logiche con associato un peso probabilistico). Queste regole permettono l'esistenza di contraddizioni nella



*Figura 2.20: Esempio di modello ricostruito che esclude stanze parziali, tratto da [57]. Le stanze parziali sono simboleggiate da un cerchio rosso.*

knowledge base e conservano flessibilità. Il secondo metodo offre un modo efficace per estrarre campioni da distribuzioni complesse e sconosciute.

## 2.5 Mappa topologica

La mappa topologica rappresenta le diverse regioni di spazio individuate dalla segmentazione, mostrando addizionalmente le connessioni tra queste regioni. Riporta quindi la nozione aggiuntiva di *connettività* tra gli spazi. La connessione tra due spazi viene rappresentata dalla mappa topologica se queste due aree sono direttamente collegate (ad esempio tramite una porta). Una particolare tipologia di mappa topologica consiste nel grafo topologico. Un esempio di grafo topologico viene mostrato in Figura 2.21. I diversi spazi dell’ambiente sono rappresentati come nodi, e le connessioni tra di essi sono rappresentate come archi.

Questa rappresentazione offre un più alto grado di astrazione rispetto al lay-

out delle stanze, ed arricchisce il contenuto informativo del modello. Viene infatti esposta in maniera intuitiva ed immediata, attraverso l'uso di nodi ed archi, la nozione di connettività tra le diverse stanze.

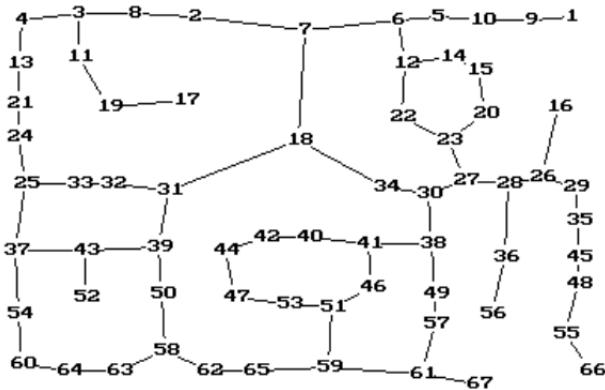
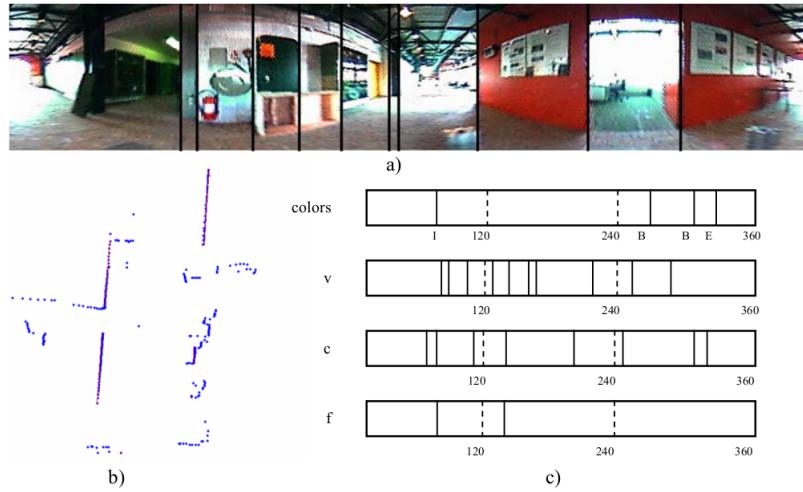


Figura 2.21: Esempio di grafo topologico tratto da [106].

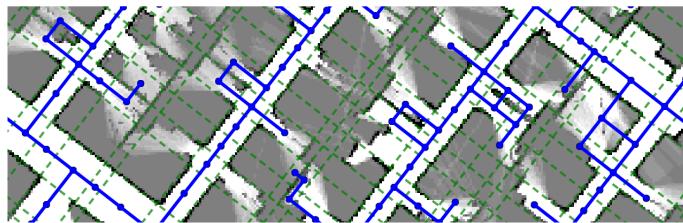
In [102] e [101] viene presentata una tecnica per mapping topologico usando *fingerprint* degli spazi che caratterizzano l'ambiente. Lavori come [51] e [103] descrivono approcci per costruire fingerprint. Una fingerprint è una lista di feature, in cui l'ordine rispecchia quello delle feature attorno al robot, rilevate tramite camera e laser scanner. La sequenza è denotata da caratteri, ognuno dei quali rappresenta un'istanza di uno specifico tipo di feature. Le feature vengono estratte dai sensori, e ordinate in sequenza in base alla posizione angolare, come mostrato in Figura 2.22. L'approccio si appoggia su un'euristica che individua se la posizione corrente del robot è simile o meno a una già mappata. Il sistema introduce un nuovo nodo nella mappa topologica ogniqualvolta riconosce un cambiamento significativo nell'ambiente, basandosi sulle differenze tra le feature percepite. Ciò viene reso possibile utilizzando le fingerprint. Il metodo traccia una mappa topologica e allo stesso tempo segmenta lo spazio in stanze diverse, basandosi sulle feature raccolte nelle fingerprint.

In [93] viene costruita una mappa piana metrico-topologica basata su di una mappa a griglia. Il metodo individua gli orientamenti dominanti degli ostacoli rilevati e adatta le feature lineari a questi orientamenti. Per mezzo degli orientamenti dominanti, i quali rappresentano la struttura dell'ambiente, viene ottenuta una decomposizione in celle della mappa metrica. Questa decomposizione consiste in facce poligonali con diverse forme e dimensioni, e viene salvata come un grafo di adiacenza. Il grafo formula la connettività tra



*Figura 2.22: Creazione delle fingerprint in [102]: (a) Immagine panoramica con edge verticali e colori individuati; (b) Scansioni laser con spigoli estratti; (c) Le prime tre immagini indicano la posizione (da 0 a 360 gradi) dei colori, degli edge verticali e degli spigoli, rispettivamente. La quarta immagine descrive la corrispondenza tra edge verticali e spigoli.*

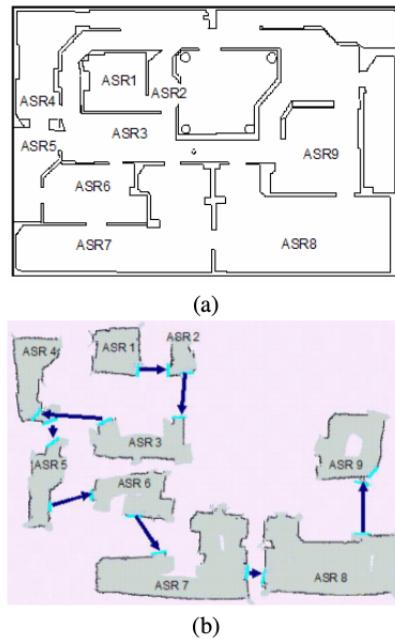
le facce della decomposizione e rappresenta la struttura metrico-topologica dell’ambiente, come mostrato in Figura 2.23. Ogni nodo del grafo rappresenta una cella ed è localizzato al centro di essa.



*Figura 2.23: Grafo topologico ottenuto in [93]. La suddivisione è rappresentata in verde, i link in blu. I nodi del grafo rappresentano le celle in cui è decomposta la mappa metrica, mentre i link rappresentano la loro connettività.*

In [53] e [45] viene utilizzato il *cognitive mapping*, un approccio di mapping topologico descritto in [48] che rappresenta le stanze e le loro connessioni spaziali. Nello specifico viene impiegata la *absolute space representation (ASR)*, una tecnica di cognitive mapping usata per costruire modelli di stanze o spazi visitati. Questo metodo, piuttosto che salvare la mappa come un’occupancy grid, vede il mondo come una serie di spazi connessi. Questi spazi sono mappati inizialmente come un’occupancy grid stanza per

stanza. Ogniqualvolta il robot lascia la stanza, ovvero quando attraversa una porta, la griglia è convertita in una rappresentazione poligonale. Queste rappresentazioni vengono salvate e collegate tra loro per mezzo di link che rappresentano le connessioni tra le stanze. L'ASR così ottenuta consiste in una rete topologica che mostra la struttura dell'ambiente, ed un esempio è mostrato in Figura 2.24.



*Figura 2.24: Absolute space representation ottenuta in [53]. Le rappresentazioni poligonali relative alle stanze sono collegate tra loro tramite link (in blu) che rappresentano le connessioni tra le stanze.*

Un'altra tecnica di cui avvalersi per la costruzione di una rappresentazione topologica è l'utilizzo del Voronoi graph, già discusso in precedenza in riferimento al layer di layout delle stanze. Un approccio di questo tipo viene utilizzato in [106], [105], [49], [7] e [117]. In questi metodi sono generate mappe topologiche a partire dalla mappa metrica, dividendola in regioni coerenti tramite l'utilizzo del Voronoi graph, sia esso nella sua versione generale, extended o ottenuto dalla distance map.

## 2.6 Mappa semantica

La *mappa semantica* descrive gli spazi dell'ambiente attraverso l'attribuzione di label che assegnano un significato semantico a porzioni dell'ambiente.

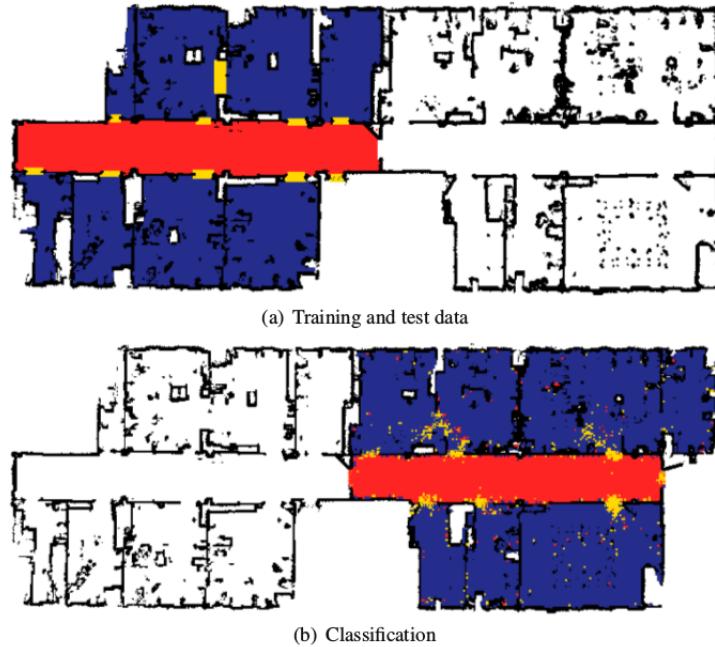
Lo scopo dell’attribuzione di label semantiche è quello di associare alla stanza una parola che ne descriva la funzione o altre peculiarità. Queste label semantiche consistono in etichette che possono essere utilizzate per indicare il significato e la funzione della porzione di ambiente cui sono associate. Tipiche label semantiche possono essere stanza, corridoio, porta, ufficio. Esistono numerosi metodi che eseguono un mapping semantico dell’ambiente di lavoro di un robot.

In [66], [67] e [65] per classificare semanticamente mappe geometriche viene proposto un approccio basato sul *supervised learning*, descritto in [64], [104] e [116]. Il supervised learning è un processo attraverso il quale si ricava una funzione partendo da dati di training. Ogni dato di training è una coppia che consiste in un oggetto di input (tipicamente un vettore) e in un valore di output desiderato. Un algoritmo di supervised learning analizza i dati di training e produce una funzione, che può essere usata per mappare nuovi esempi. In questi lavori vengono annotate semanticamente mappe geometriche ottenute con robot mobili in un ambiente indoor usando laser range scanner. Si assume che il robot conosca la mappa di un ambiente sotto forma di occupancy grid. L’approccio determina la classe semantica di ogni cella non occupata della griglia, tramite l’utilizzo di un classificatore AdaBoost, con un ampio insieme di semplici feature calcolate direttamente sulle misure del laser scanner. Un esempio è mostrato in Figura 2.25.

In [28] viene integrato l’approccio di questi tre lavori, basato sulle feature, con una segmentazione tramite Voronoi graph. L’obiettivo è etichettare ogni posizione nella mappa metrica dell’ambiente con il tipo di regione a cui appartiene, come mostrato in Figura 2.26. Queste etichette forniscono una segmentazione intrinseca della mappa, che rappresenta la struttura topologica dell’ambiente. I tipi di spazi sono appresi con AdaBoost, applicato sia alle feature della mappa che alle feature della topologia del Voronoi graph, come distanza dagli ostacoli o numero di vicini nel Voronoi graph.

In [84] e [65] viene presentato un metodo per semantic labeling che si compone di più layer, analogo al lavoro in [118] descritto in precedenza. I layer sono integrati in una rappresentazione basata su mappe multiple a diversi livelli di astrazione. Partendo dalla mappa metrica, che costituisce il layer più basso, si aggiungono rappresentazioni più astratte sotto forma di nuovi layer, costruiti sui layer precedenti.

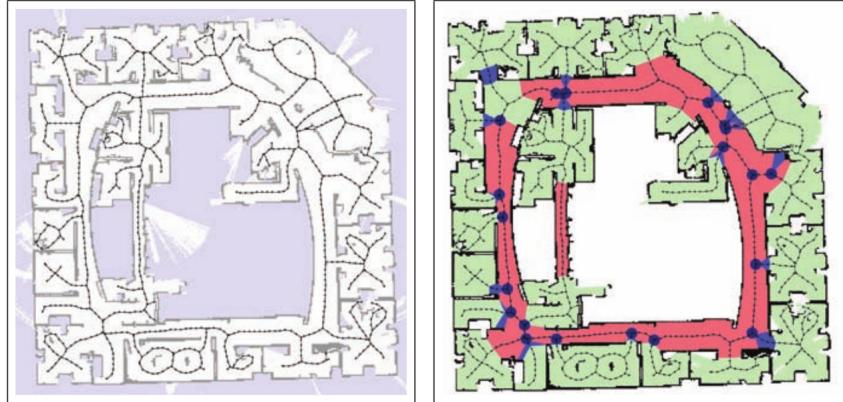
In cima alla mappa metrica viene costruito un grafo di navigazione. I nodi



*Figura 2.25: Esempio di mapping semantico in [65]: (a) Dati di training (colorati) e dati di test (bianchi); (b) Risultato della classificazione.*

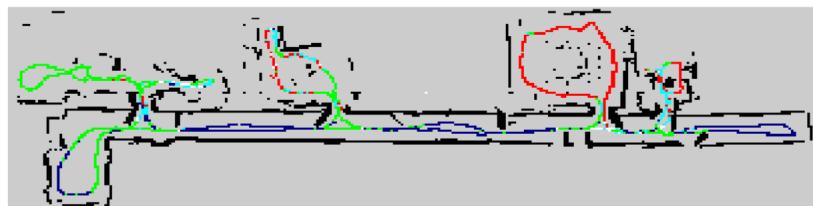
del grafo vengono annotati semanticamente in tre classi: stanza, corridoio e porta. Il layer successivo consiste nella mappa topologica. La mappa topologica divide i nodi in aree, ognuna delle quali consiste in un insieme di nodi interconnessi, separati da un nodo classificato come porta. Le aree vengono classificate come stanze o corridoi in base alla maggioranza delle classificazioni dei nodi dell'area in questione. L'ultimo layer consiste nella mappa concettuale. In essa vengono rappresentate categorie astratte per stanze e oggetti, con relazioni corrispondenti. Le aree estratte nei layer precedenti vengono rappresentate come token che instanziano concetti astratti. L'approccio è stato integrato in un sistema in esecuzione su di un robot mobile, dotato di un sensore laser e di una camera.

In [34] viene presentato un framework in grado di costruire mappe annotate semanticamente partendo da misurazioni laser di un robot mobile. Per ogni istante di tempo il laser range scanner raccoglie dati sensoriali mentre il robot si muove entro l'ambiente. I valori misurati sono salvati in un distance vector, uno per ogni istante di tempo. Utilizzando distance vector successivi e i corrispondenti movimenti del robot, l'algoritmo di SLAM Gmapping (descritto in Sezione 3.2.2) produce una mappa a griglia. Dai distance vector



*Figura 2.26: Esempio di mapping semantico in [28]: (sinistra) mappa a griglia costruita con SLAM, e Voronoi graph; (destra) Voronoi graph etichettato che definisce un tipo di spazio per ogni punto della mappa. I corridoi sono colorati in rosso, le stanze in verde e le porte in blu.*

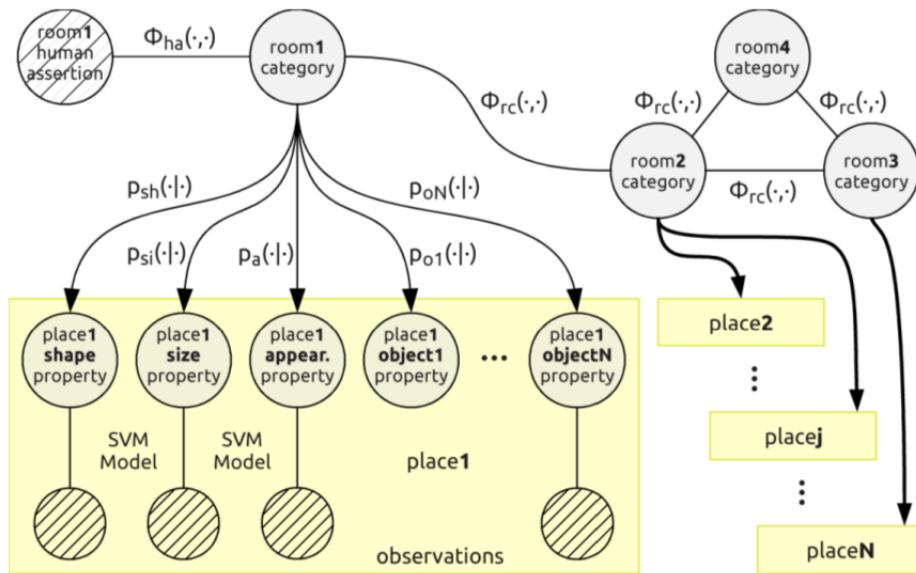
viene estratto un insieme di feature come base per la classificazione. Questo insieme di feature, sotto forma di vettore, viene passato a un classificatore AdaBoost, per mezzo del quale viene assegnata un'etichetta semantica alle celle della occupancy grid map. Le possibili classi semantiche sono porta, corridoio, spazio libero, stanza e sconosciuto. In Figura 2.27 viene mostrata l'annotazione semantica del percorso di un robot entro un ambiente indoor.



*Figura 2.27: Percorso di un robot reale con annotazioni semantiche tratto da [34].*

In [83] viene presentato un framework probabilistico che, per eseguire mapping semantico, combina informazioni eterogenee ed incerte. Queste informazioni possono essere forma, dimensione e aspetto di oggetti e stanze. Il framework astrae informazioni sensoriali e le integra con conoscenza concettuale attraverso un approccio probabilistico. Per rappresentare l'informazione concettuale ed effettuare ragionamento spaziale viene utilizzato un *chain graph*, descritto in [52]. Il chain graph è un modello grafico probabilistico che consiste in un grafo privo di cicli, come mostrato in Figura 2.28. In questo grafo i vertici rappresentano variabili casuali, gli archi rappresen-

tano le relazioni probabilistiche tra le variabili casuali e i vertici colorati indicano osservazioni che corrispondono a prove percepite. L'abilità del sistema consiste nella deduzione di categorie semantiche delle stanze, nella predizione dell'esistenza di oggetti, nella valutazione di altre proprietà spaziali e nel ragionamento sullo spazio inesplorato.



*Figura 2.28: Struttura del chain graph della mappa concettuale in [83]. I vertici rappresentano variabili casuali. Gli archi rappresentano le relazioni probabilistiche tra le variabili casuali. I vertici colorati indicano osservazioni che corrispondono a prove percepite.*

## **Capitolo 3**

# **Progetto logico della costruzione del sistema multilivello**

In questo capitolo viene descritto l'approccio logico allo sviluppo del sistema di rappresentazione multilivello, oggetto di questo lavoro di tesi. Viene inizialmente descritta la finalità di un sistema di questo tipo, incentrandosi poi il discorso sui diversi approcci di mapping che compongono il sistema. Per ciascuno di essi viene in prima istanza spiegato l'obiettivo e il livello di conoscenza offerto al robot. Successivamente sono presentate le tecniche che permettono di ottenere tale rappresentazione.

### **3.1 Il sistema multilivello**

Un robot mobile, al fine di essere autonomo, deve possedere la conoscenza relativa all'ambiente circostante. Questa viene acquisita dall'ambiente raccogliendo dati sensoriali ed elaborandoli al fine di ottenere una rappresentazione dello spazio entro cui il robot può muoversi. Tale rappresentazione consiste tipicamente in una mappa, che fornisce al robot informazioni utili per comprendere lo spazio circostante. Le informazioni riportate dalla mappa possono essere utilizzate per estrarne di nuove, estendendo la conoscenza che il robot possiede sull'ambiente e suddividendola in livelli di astrazione, cui corrispondono diverse rappresentazioni. Queste rappresentazioni caratterizzano diversi approcci di mapping.

Un sistema di mapping multilivello ha la capacità di rappresentare i diversi livelli di conoscenza relativi all'ambiente tramite l'integrazione di diverse mappe. I sistemi di questo tipo sono in grado di rappresentare

tutte le informazioni sull’ambiente e i legami tra le diverse rappresentazioni. Le mappe sono tra loro correlate, e modellano lo spazio a diversi gradi di astrazione in base al proprio livello di conoscenza.

In questo lavoro di tesi è stato sviluppato un sistema di mapping multilivello che integra in un unico schema onnicomprensivo diversi tipi di mappe: mappa metrica, layout delle stanze, grafo topologico e mappa semantica. Vengono di seguito discussi gli obiettivi di ciascuno di questi metodi di mapping e l’approccio logico alla loro costruzione.

## 3.2 Mappa metrica

### 3.2.1 Obiettivo

La mappa metrica è il primo layer di rappresentazione integrato nel sistema multilivello. L’obiettivo di questo approccio di mapping consiste nel fornire al robot una conoscenza di basso livello relativa alla struttura e all’occupazione dell’ambiente. La rappresentazione si limita ad un modello metrico dello spazio circostante il robot, come mostrato e discusso nella Sezione 2.3.

Il livello di conoscenza estratto dalla mappa è utilizzato dal robot al fine di soddisfare task di navigazione e localizzazione. Utilizzandola il robot acquisisce l’abilità di muoversi autonomamente entro l’ambiente circostante, allo stesso tempo localizzandosi in esso. La mappa metrica inoltre conferisce al robot la capacità di pianificare percorsi dalla posizione corrente a quella di un goal, garantendo che il percorso sia per quanto possibile libero da ostacoli e scongiurando il pericolo di collisioni.

La mappa metrica rappresenta il punto di partenza del sistema multilivello, la base su cui vengono costruiti i layer successivi. Partendo dall’informazione offerta dalla rappresentazione metrica, è infatti possibile estrarre altri livelli di conoscenza, caratterizzati da un grado di astrazione più elevato.

La mappa metrica può essere nota o non nota a priori; sono di seguito discussi i due casi.

### 3.2.2 Mappa metrica non nota a priori

Nel caso la mappa metrica non sia nota inizialmente, si rende necessario l’utilizzo di un metodo tramite il quale ottenerla, integrando in modo incrementale le scansioni acquisite in punti diversi dell’ambiente. Questi metodi prendono il nome di *Simultaneous Localization and Mapping (SLAM)* [107], che rappresenta l’insieme delle attività compiute dal robot per costruire una mappa metrica dell’ambiente e al contempo stabilire la propria posizione

all'interno di essa. In particolare in questo lavoro di tesi viene utilizzato il modulo di SLAM *GMapping*, descritto in [35] e [36].

## GMapping

La complessità del problema dello SLAM consiste nella dipendenza reciproca tra la *posa* del robot (combinazione di posizione e orientamento) e la definizione della mappa. Infatti il robot per localizzarsi necessita di una mappa consistente e allo stesso tempo per costruire una mappa richiede una buona stima della sua posizione.

In [21] e [70] vengono introdotti dei filtri di particelle di tipo *Rao-Blackwellized* come strumento per risolvere tale problema. L'idea chiave di questo approccio di risoluzione dello SLAM risiede nella stima della *joint posterior*  $p(x_{1:t}, m|z_{1:t}, u_{1:t-1})$  sulla mappa  $m$  e sulla traiettoria  $x_{1:t} = x_1, \dots, x_t$  del robot. Questa stima viene eseguita date le osservazioni  $z_{1:t} = z_1, \dots, z_t$  e le misure di odometria  $u_{1:t-1} = u_1, \dots, u_{t-1}$  ottenute dal robot mobile. L'appuccio utilizza la seguente fattorizzazione:

$$p(x_{1:t}, m|z_{1:t}, u_{1:t-1}) = p(m|x_{1:t}, z_{1:t}) * p(x_{1:t}|z_{1:t}, u_{1:t-1})$$

Per stimare la posterior  $p(x_{1:t}|z_{1:t}, u_{1:t-1})$  sulle traiettorie potenziali, viene applicato un filtro di particelle Rao-Blackwellized, in cui ogni particella rappresenta una potenziale traiettoria del robot, associando inoltre una singola mappa ad ogni campione. Il filtro elabora in modo incrementale le osservazioni sensoriali e le misure di odometria non appena disponibili, aggiornando l'insieme di campioni che rappresentano la posterior riguardante mappa e traiettoria del robot. Ad ogni step, l'insieme successivo di particelle  $x_t^{(i)}$  è ottenuto dall'insieme  $x_{t-1}^{(i)}$ , effettuando un campionamento dalla proposal distribution  $\pi$ . Come proposal distribution  $\pi$  viene utilizzato un modello probabilistico odometrico. Tuttavia questa proposal distribution non è ottimale, specialmente quando le informazioni sensoriali sono significativamente più precise del movimento stimato del robot basato sull'odometria. Ciò causa la necessità di un alto numero di particelle, rendendo l'algoritmo poco efficiente.

Il problema principale degli approcci Rao-Blackwellized consiste dunque nella loro complessità misurata in termini di numero di particelle richieste per costruire una mappa accurata. Ridurre questa quantità è il principale risultato conseguito tramite l'utilizzo del GMapping.

Il GMapping è un approccio che incrementa le prestazioni dei filtri di particelle Rao-Blackwellized applicati per risolvere il problema dello SLAM. Esso utilizza una proposal distribution che considera l'accuratezza dei sen-

sori del robot e permette di definire particelle in una maniera accurata. La proposal distribution  $\pi$ , a differenza dei lavori in [21] e [70], è calcolata prendendo in considerazione non solo il movimento del robot, ma anche le osservazioni più recenti, integrando le informazioni dell'odometria con quelle relative alle scansioni sensoriali. In questo modo, l'osservazione sensoriale più recente è utilizzata per creare il successivo insieme di particelle. Questo permette di stimare lo stato del robot per mezzo di un modello maggiormente aggiornato ed accurato rispetto a quello ottenuto basandosi esclusivamente sulle informazioni dell'odometria. La mappa è più accurata poichè l'osservazione corrente è incorporata nelle singole mappe dopo aver considerato il suo effetto sulla posa del robot. Questo riduce significativamente l'errore di stima, rendendo necessario un minor numero di particelle per rappresentare la posterior.

Tramite GMapping viene risolto il problema dello SLAM, apprendendo una mappa a griglia a partire da dati di laser range e di odometria, simultaneamente al processo di localizzazione.

Per mezzo della mappa a griglia, di cui un esempio è riportato in Figura 3.4 (a), il robot ricava informazioni che codificano l'occupazione dello spazio, distinguendo le zone occupate da quelle libere. Ad ogni cella della griglia costituente la mappa è associata l'informazione riguardante la sua occupazione da parte di un muro o un ostacolo.

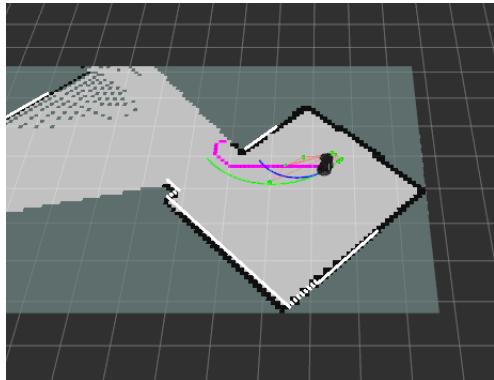
La costruzione della mappa richiede l'esplorazione dell'ambiente da parte di un robot dotato di scanner laser. L'esplorazione indica l'insieme degli approcci che permettono al robot di decidere come esplorare l'ambiente incognito. A tale scopo possono essere impiegate diverse tecniche. In questo lavoro di tesi è stata utilizzata la tecnica di esplorazione *Nearest Frontier*, discussa in [46] e di seguito descritta.

### **Nearest frontier**

Nell'ambito dell'esplorazione, assume grande rilevanza la nozione di *frontiera*. Questa corrisponde al confine che separa lo spazio conosciuto da quello non conosciuto. Lo spazio conosciuto è la parte di ambiente di cui sono state raccolte informazioni sensoriali; nella mappa a griglia corrisponde a celle descritte come libere o occupate con una probabilità compresa tra 0 e 1. Lo spazio non conosciuto consiste invece nella porzione di ambiente di cui non è disponibile alcuna informazione sensoriale.

La tecnica di esplorazione *Nearest Frontier* si basa sul concetto di frontiera. La prima operazione eseguita da questa tecnica consiste nell'individuare le

frontiere accessibili presenti nella mappa a griglia. Una frontiera accessibile è una frontiera per cui esiste almeno un path non occluso da ostacoli che la collega al robot. Il robot calcola le distanze che lo separano da tutte le frontiere accessibili individuate. La frontiera accessibile più vicina alla posizione corrente del robot viene scelta come destinazione successiva. La navigazione verso tale frontiera viene eseguita pianificando un percorso, garantito come libero da ostacoli. La Figura 3.1 mostra un esempio di navigazione del robot per mezzo di questa tecnica. La traiettoria in rosa indica il percorso privo di ostacoli individuato dal robot per poter raggiungere la frontiera più vicina. Una volta giunto alla destinazione, il robot ricalcola le distanze dalle frontiere accessibili rimanenti, e la tecnica di esplorazione si ripete in modo analogo, finché non vi sono più frontiere accessibili inesplorate.

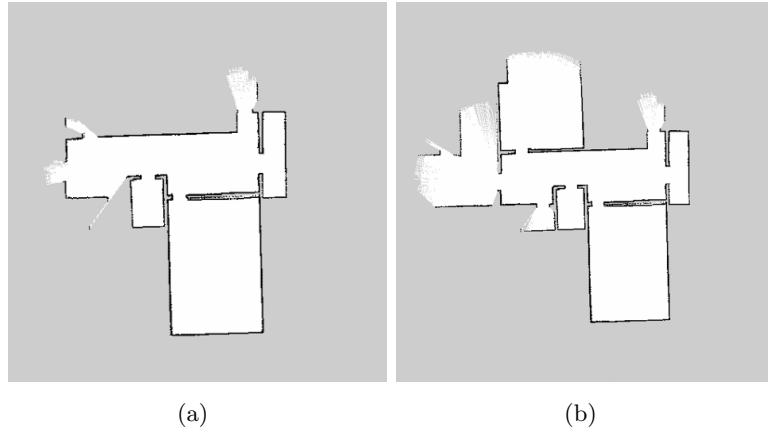


*Figura 3.1: Approccio di esplorazione Nearest frontier. In rosa è rappresentato il percorso calcolato dal robot tramite cui raggiungere senza collisioni la frontiera più vicina.*

La Figura 3.2 mostra un esempio di creazione incrementale della mappa a griglia tramite esplorazione da parte del robot, seguendo un approccio di esplorazione Nearest frontier e utilizzando il modulo di SLAM GMapping.

### 3.2.3 Mappa metrica nota a priori

Nel caso in cui la mappa metrica sia nota, possono essere identificate almeno due opzioni: la mappa nota al robot deriva da un'esplorazione precedente del robot (Sezione 3.2.2), oppure è fornita come una planimetria, ovvero un disegno bidimensionale in scala che mostra una vista dall'alto dell'ambiente, similmente all'utilizzo di una cartina da parte di una persona. La Figura 3.3 (a) riporta un esempio di planimetria. Questa rappresentazione può includere elementi aggiuntivi relativi all'ambito dell'evacuazione, come



*Figura 3.2: Mappa a griglia creata in modo incrementale. Da (a) a (b) il robot tramite esplorazione ha acquisito informazioni sensoriali su regioni di spazio non conosciute.*

simboli di estintori, idranti, uscite di sicurezza o altre indicazioni testuali sull’ambiente.

### 3.3 Layout delle stanze

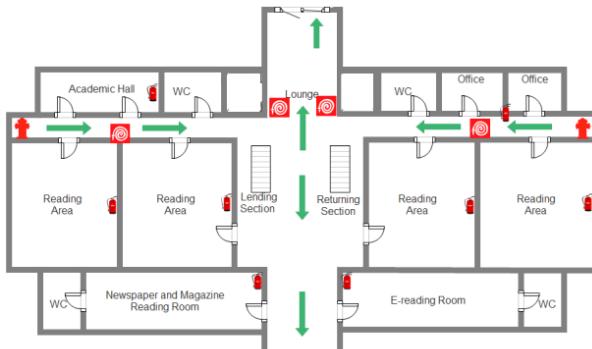
#### 3.3.1 Obiettivo

Le informazioni metriche riportate dal layer precedente vengono elaborate in modo da ottenere un nuovo modello che arricchisce il contenuto informativo relativo all’ambiente. Questo modello di rappresentazione è caratterizzato da un grado di astrazione più elevato, dove la conoscenza non è più esclusivamente di carattere metrico.

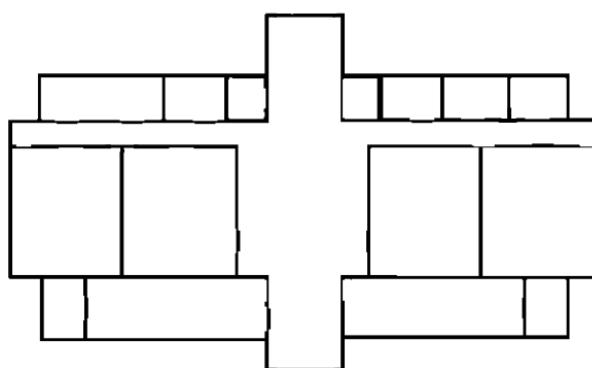
L’obiettivo dell’identificazione del layout delle stanze è offrire una rappresentazione che mostri la disposizione e suddivisione delle stanze che compongono l’edificio, come mostrato in Figura 3.4 (b). Tramite questa mappa viene estesa la conoscenza che il robot possiede sull’ambiente circostante, in quanto alle informazioni metriche si aggiunge una separazione dello spazio in diverse aree.

I possibili input alla fase di costruzione del layout delle stanze sono:

- **Occupancy grid map:** una mappa a griglia ottenuta tramite esplorazione dell’ambiente o come immagine nota a priori, di cui un esempio è riportato in Figura 3.4 (a). Una mappa di questo tipo rappresenta esclusivamente elementi corrispondenti a pareti ed ostacoli.



(a)



(b)

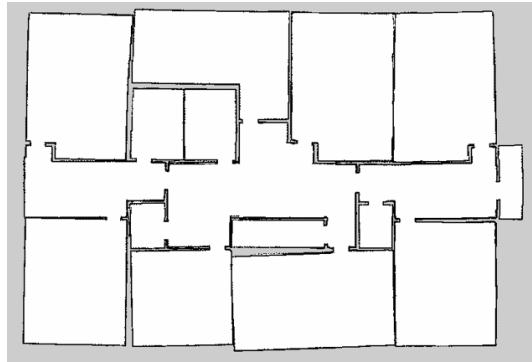
*Figura 3.3: (a) Esempio di planimetria di una scuola, tratta da [27]; (b) Risultato finale della fase di pulitura. Sono stati eliminati tutti i simboli non corrispondenti a muri, compresa l'informazione testuale.*

- **Planimetria:** un disegno bidimensionale che può contenere simboli e testo oltre alla rappresentazione delle pareti, come ad esempio mostrato in Figura 3.3 (a).

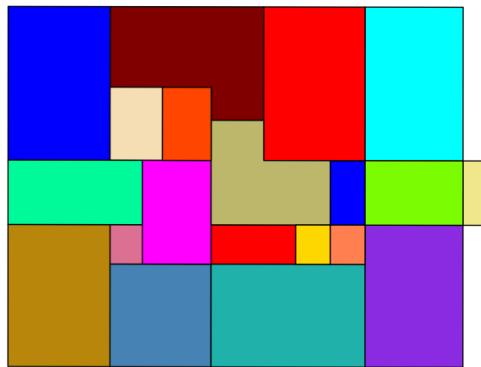
### 3.3.2 Preprocessing della planimetria

Nel caso in cui la mappa metrica consista in una planimetria, è spesso possibile identificare in essa altri simboli oltre ai muri. In questa situazione si rende necessaria una preliminare fase di preprocessing della mappa. Esempi di feature che è possibile identificare sono: simboli di porte, estintori, idranti, kit di pronto soccorso, uscite di sicurezza e informazione testuale.

La fase preliminare di preprocessing della planimetria, il cui risultato può essere osservato in Figura 3.3, ha un duplice scopo:



(a)



(b)

*Figura 3.4: (a) Esempio di occupancy grid map. Il livello di conoscenza offerto consiste in una codifica dell'occupazione dell'ambiente. Le celle nere sono occupate, le celle bianche sono libere, le celle grigie corrispondono all'ambiente di cui il robot non ha raccolto dati sensoriali; (b) Layout delle stanze costruito dalla occupancy grid map. Diversi colori rappresentano stanze diverse.*

- Ottimizzare il partizionamento della mappa metrica in stanze, attraverso la rimozione di tutti gli elementi non corrispondenti a muri.
- Salvare la posizione degli elementi estratti dalla planimetria. Nei layer successivi, analizzando tali posizioni, sarà possibile comprendere la collocazione di questi simboli nelle stanze, deducendo ad esempio quali stanze siano dotate di estintore, idrante o di altri oggetti rappresentati dai simboli estratti. Inoltre l'analisi delle posizioni dei simboli delle porte permetterà di derivare la connettività delle stanze.

Per identificare dei simboli o dei pattern in un'immagine vi sono in letteratura svariate tecniche, come *scale-invariant feature transform (SIFT)*

[58], *speeded up robust features (SURF)* [6], *gradient location and orientation histogram (GLOH)* [63], *histogram of oriented gradients (HOG)* [16]. In questo lavoro di tesi si è deciso di utilizzare una tecnica dello stato dell'arte chiamata *template matching* [90].

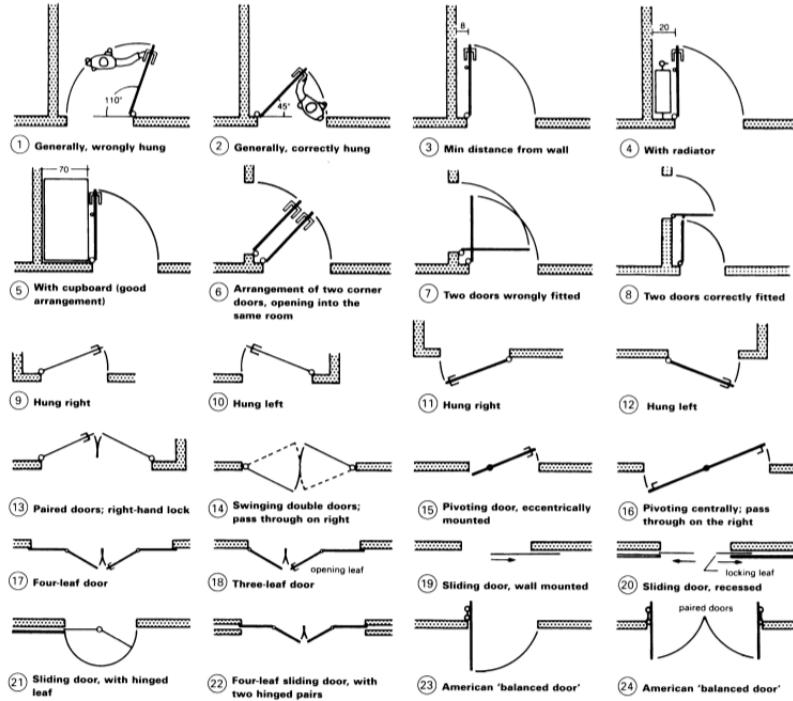
### Template matching

Il template matching è una tecnica di elaborazione digitale delle immagini, finalizzata all'identificazione di piccole porzioni di un'immagine che corrispondano ad una immagine-modello (template). Tramite questa tecnica vengono cercate e individuate le posizioni dei template all'interno di un'immagine più grande. La tecnica utilizza l'intera immagine-modello per determinare la miglior posizione di un eventuale riscontro individuato. L'approccio è sensibile a rotazione e scaling, di conseguenza ciascun template viene ruotato e scalato in diversi modi prima di procedere alla ricerca, in modo da ottimizzare i risultati conseguiti.

In questo lavoro di tesi il template matching è applicato alla planimetria con lo scopo di individuare i simboli non corrispondenti a muri (porte, estintori, idranti, ecc.). A questo scopo viene utilizzata una lista di template standard di tali oggetti tipicamente presenti nelle planimetrie, tratta da [75]. Al fine di migliorare l'accuratezza del matching, la lista include diversi template per ciascuna tipologia di oggetto. In Figura 3.5 è riportata una parte di questi template, relativa alle porte. In Figura 3.6 viene mostrato un esempio di individuazione di simboli di estintori (a) e idranti (b) in una planimetria, per mezzo di questa tecnica. I simboli identificati sono circondati da un rettangolo blu.

Gli elementi individuati, se non corrispondono a porte, vengono eliminati dalla planimetria colorando del colore dello sfondo, in questo caso bianco, tutti i pixel che vi appartengono.

Nel caso in cui il template individuato corrisponda a una porta, il processo di eliminazione è differente. Le porte vengono infatti sostituite con altri template, istanziati nella medesima posizione delle porte e raffiguranti una porzione di muro larga quanto la porta in questione. Il procedimento viene mostrato in Figura 3.7: i simboli delle porte individuati tramite template matching, indicati da un rettangolo blu (a), sono sostituiti da template di muri (b). Lo scopo è generare una planimetria in cui le porte risultino chiuse, accentuando la separazione tra stanze adiacenti. In questo modo viene incrementata l'accuratezza del modello di layout delle stanze prodotto dalla segmentazione. Viene infatti ridotto il rischio di un erroneo raggruppamento

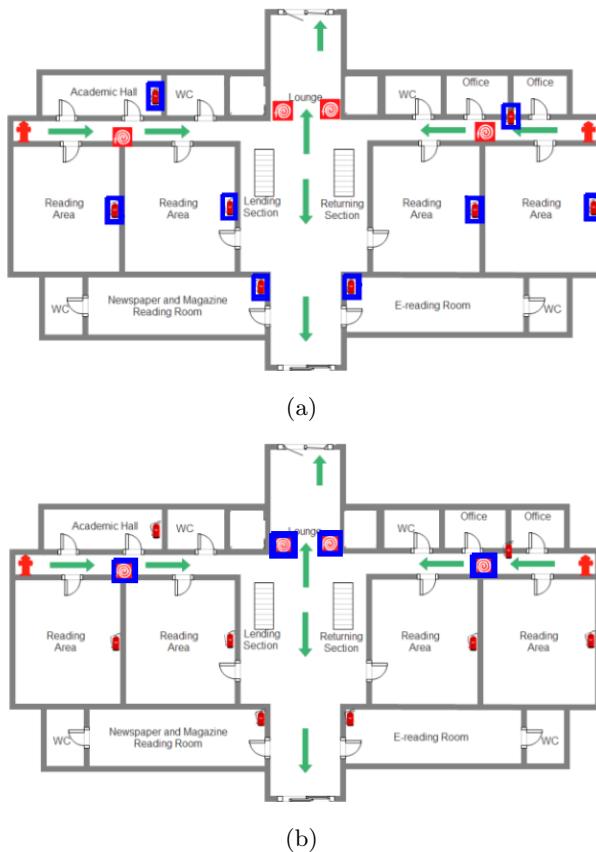


*Figura 3.5: Lista di template standard di porte, tipicamente presenti in una planimetria. Il template matching viene eseguito ricercando questi simboli all'interno della planimetria. L'immagine è tratta da [75].*

nello stesso cluster di stanze adiacenti collegate da una porta, come mostrato in Figura 3.8. Eliminando le porte in modo analogo agli altri simboli (a), ovvero colorando del colore dello sfondo (bianco) tutti i pixel che vi appartengono, il layout delle stanze può presentare imprecisioni (b). La stanza in alto a destra (evidenziata da un cerchio rosso) non è riconosciuta come una regione di spazio separata dalle altre. La chiusura delle porte ottenuta tramite sostituzione con template di muri (c) genera invece un layout delle stanze corretto (d), in cui la stanza in alto a destra (cerchio rosso) risulta disgiunta dalle altre.

### Motivo dell'uso del template matching

Template matching non è il più raffinato tra i metodi in grado di identificare simboli all'interno di un'immagine. Esso infatti ricerca un template per volta ed è sensibile a rotazione e scaling. Di conseguenza per ottimizzare i risultati della ricerca è necessario ruotare e scalare i vari template. Ciò può influire negativamente sulla complessità computazionale, in quanto il



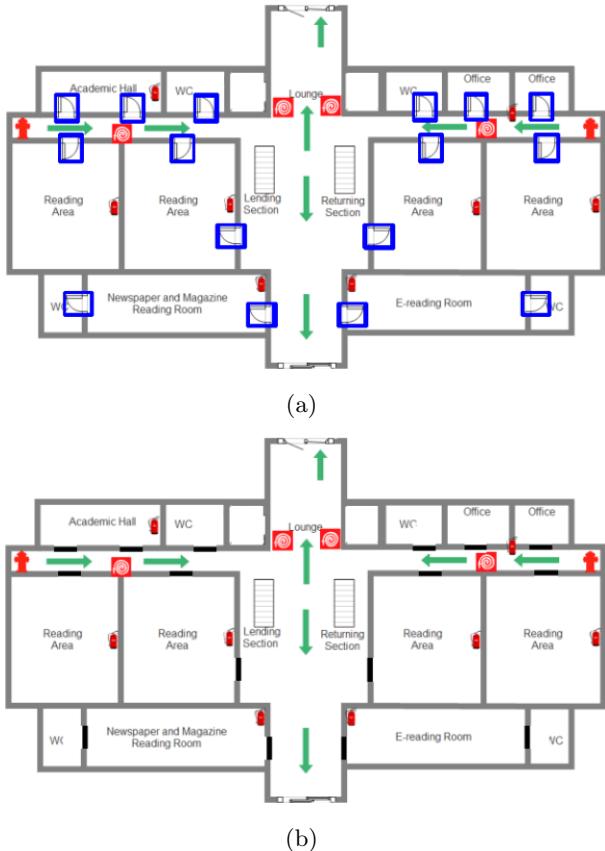
*Figura 3.6: Esempio di simboli individuati nella planimetria per mezzo della tecnica di template matching. I simboli individuati sono indicati da un rettangolo blu: (a) Estintori; (b) Idranti.*

numero di operazioni richieste cresce linearmente con il numero di template ricercati.

Ciononostante si è deciso di utilizzare il template matching in questo lavoro di tesi poiché l’obiettivo principale del lavoro non consiste in un’analisi raffinata ed approfondita delle immagini. L’identificazione di semplici simboli all’interno di una planimetria è un processo secondario e funzionale alle fasi successive, di conseguenza non si rende necessario l’utilizzo di un approccio più raffinato. Inoltre i risultati ottenuti per mezzo del template matching si dimostrano corretti.

### Informazione testuale

Spesso nelle planimetrie sono presenti delle informazioni testuali che tipicamente identificano le aree dell’edificio, come mostrato in Figura 3.3 (a). Que-

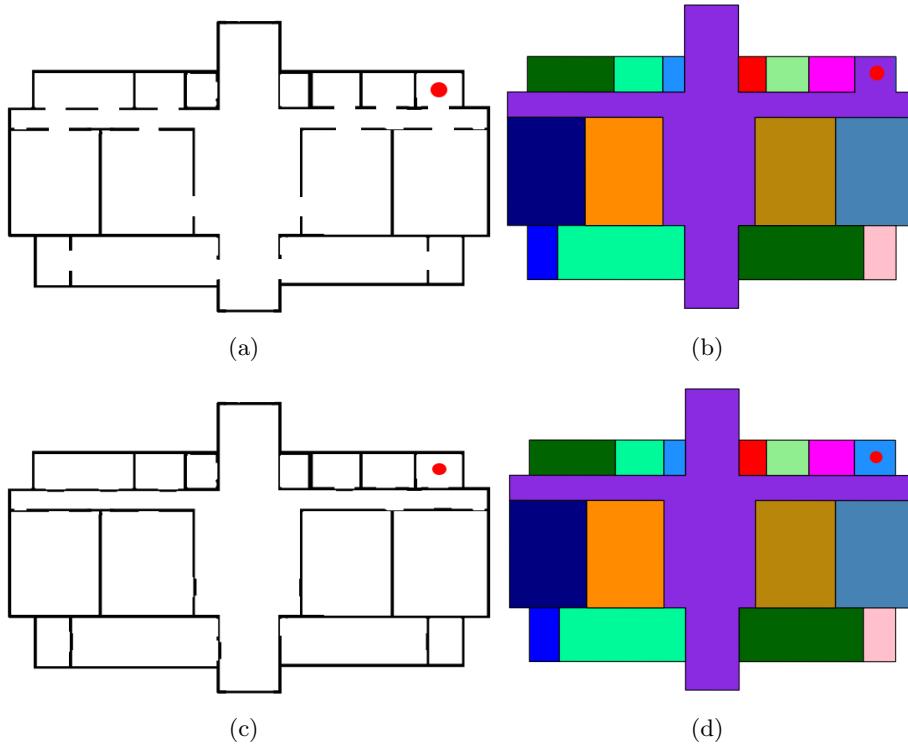


*Figura 3.7: Sostituzione dei simboli delle porte con template di muri: (a) Simboli delle porte individuati per mezzo del template matching, indicati da un rettangolo blu; (b) Risultato della sostituzione di ciascuna porta con un template di muro.*

ste informazioni, sebbene utili, devono essere rimosse dalla planimetria al fine di ottenere una mappa metrica in cui siano rappresentate esclusivamente le pareti, necessaria per il partizionamento in stanze.

L'eliminazione dell'informazione testuale richiede un'operazione differente dall'eliminazione degli altri simboli. Il testo viene individuato e salvato utilizzando un generico sistema di *riconoscimento ottico dei caratteri* (*OCR*) (in questo lavoro di tesi è stato utilizzato Tesseract [85]). Un *OCR* è un sistema dedicato alla conversione di un'immagine contenente testo in testo digitale, modificabile con un normale editor.

Questo strumento riconosce e salva in un file il testo presente nella planimetria. Una volta salvato, il testo può essere eliminato. L'eliminazione non è effettuata dall'*OCR*, ma viene eseguita individuando gli oggetti nella planimetria il cui contorno esterno ha dimensioni inferiori rispetto ad una



*Figura 3.8: (a) Risultato del preprocessing della planimetria eseguito eliminando i simboli delle porte in modo analogo agli altri simboli, ovvero colorando del colore dello sfondo (bianco) tutti i pixel che vi appartengono; (b) Relativo layout delle stanze, con un'imprecisione causata dall'apertura delle porte. La stanza in alto a destra (evidenziata da un cerchio rosso al centro di essa) non è stata individuata come regione di spazio separata dalle altre. (c) Planimetria ottenuta sostituendo le porte con template di muri; (d) Relativo layout delle stanze, in cui la stanza in alto a destra (cerchio rosso) risulta correttamente disgiunta dalle altre.*

soglia  $\eta$  (settata a  $3 \text{ m}^2$ ). Questa operazione si basa sull'assunzione che gli oggetti relativamente piccoli corrispondano alle parole da eliminare, mentre i muri siano caratterizzati da dimensioni maggiori del contorno esterno, come mostrato in Figura 3.9.

Dopo questi passaggi, la fase di preprocessing della planimetria è conclusa e può prendere il via la fase di segmentazione. Il metodo utilizzato per ottenere la segmentazione della mappa metrica in stanze prende ispirazione dall'approccio presentato in [68]. Vi sono tuttavia alcune differenze. Di seguito vengono presentate le operazioni svolte dal lavoro di [68], mostrate in precedenza nella Sezione 2.4.9. Sono poi discusse le differenze con l'approccio sviluppato in questo lavoro di tesi.

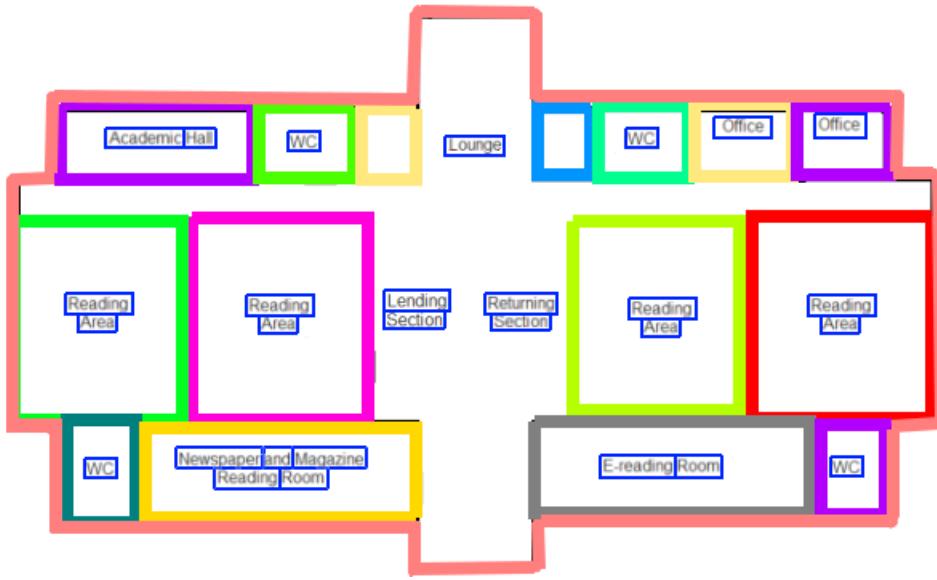


Figura 3.9: Sono rappresentati i contorni identificati nella planimetria. I contorni delle parole (blu) risultano di dimensione inferiore rispetto ai contorni dei muri (altri colori).

## Approccio guida

Il lavoro descritto in [68] si compone dei seguenti passi:

- **Selezione delle superfici dei muri in 3D:** analizzando i punti relativi alle scansioni laser tridimensionali vengono individuate le superfici verticali corrispondenti alle pareti dell'edificio.
- **Proiezione delle superfici dei muri in 2D:** le superfici identificate vengono proiettate sul piano del pavimento, ottenendo dei segmenti corrispondenti ai muri.
- **Calcolo delle rette rappresentative:** viene applicato l'algoritmo di clustering *mean-shift* [15] sui segmenti in modo da raggrupparli in un numero limitato di rette rappresentative, ciascuna corrispondente ad un cluster.
- **Costruzione dell'insieme di celle:** le rette rappresentative generano per mezzo delle loro intersezioni un insieme di celle. Gli edge (lati) di ciascuna cella vengono pesati in base alla loro probabilità di corrispondere a muri realmente presenti nell'edificio.
- **Clustering delle celle in stanze:** viene creata una matrice di affinità locale delle celle utilizzando i pesi degli edge. Le celle vengono

raggruppate in stanze tramite l'algoritmo *k-medoids*, descritto in [37], applicato sulla matrice creata.

### Differenze con questo lavoro di tesi

Il metodo implementato in questo lavoro di tesi segue a grandi linee l'approccio appena descritto per ottenere il layout delle stanze, apportando le seguenti modifiche:

- **Dati di partenza:** in [68] i segmenti corrispondenti ai muri sono ottenuti a partire dalle scansioni laser tridimensionali, analizzate e riportate in 2D. In questo lavoro di tesi invece i segmenti corrispondenti ai muri vengono estratti analizzando una mappa metrica bidimensionale.
- **Clustering delle celle in stanze:** l'algoritmo utilizzato per raggruppare le celle in stanze non è il *k-medoids* bensì il *DBSCAN*, presentato in [23] e dimostratosi il più accurato tra gli algoritmi di clustering sperimentati.

Tutti gli altri passaggi restano invece invariati.

Ora illustriamo nel dettaglio le operazioni di cui si compone il processo di costruzione del layout delle stanze, discutendone l'approccio logico.

La prima di queste operazioni ha lo scopo di identificare i segmenti corrispondenti ai muri.

#### 3.3.3 Individuazione dei segmenti corrispondenti ai muri

Il primo problema che intercorre nella fase di costruzione del layout delle stanze consiste nell'estrazione di feature su cui basare il processo di segmentazione. Non disponendo direttamente delle scansioni prese da GMapping, ma disponendo della mappa metrica derivata da tali scansioni, quest'ultima costituisce la sorgente da cui estrarre le feature geometriche fondamentali per poter sviluppare la segmentazione. Nel nostro caso queste feature consistono nei segmenti corrispondenti ai muri.

Una soluzione al processo di estrazione dei segmenti dalla mappa metrica è l'utilizzo di tecniche di feature extraction usate per task simili. A questo scopo vengono applicati alla mappa metrica, in sequenza, gli algoritmi di *Canny edge detection* [13] e *Hough line transform* [43]. Il *Canny edge detector* è uno strumento in grado di estrarre i contorni presenti nella mappa metrica. Il suo risultato viene preso in input dall'*Hough line transform* il

quale analizza i contorni identificati dal *Canny edge detector* con lo scopo di estrarne dei segmenti.

La giustificazione dell'utilizzo di questi algoritmi risiede nel fatto che la mappa metrica sia considerata come una matrice, composta da celle nel caso di una occupancy grid map, o da pixel nel caso di una planimetria.

I due metodi sono di seguito descritti.

### Canny edge detection

L'algoritmo di *Canny edge detection* [13] è un operatore per il riconoscimento dei contorni in un'immagine.

Il riconoscimento dei contorni ha lo scopo di marcare i punti di un'immagine in cui l'intensità luminosa cambia bruscamente. Esso genera un'immagine contenente molte meno informazioni rispetto a quella originale, poiché elimina la maggior parte dei dettagli non rilevanti al fine dell'individuazione dei contorni, conservando invece le informazioni essenziali per descrivere la forma e le caratteristiche strutturali e geometriche degli oggetti rappresentati.

L'obiettivo dell'algoritmo di *Canny edge detection* è quello di individuare i contorni reali, ovvero insiemi di celle o pixel a seconda della natura della mappa metrica di input (per comodità in seguito si parlerà solo di celle). L'algoritmo marca un dato contorno una sola volta, evitando che il rumore presente nell'immagine provochi il riconoscimento di falsi contorni.

Prima di iniziare l'elaborazione, all'immagine viene applicato un *filtro gaussiano*, con lo scopo di ridurne il rumore. Il filtro gaussiano è un filtro la cui risposta impulsiva è una funzione gaussiana, ovvero una funzione della forma

$$f(x) = a e^{-(x-b)^2/c^2}$$

per qualunque costante reale  $a > 0$ ,  $b$  e  $c$ . Per mezzo di questo filtro ad ogni cella della mappa metrica viene assegnata la media pesata dei valori di luminosità delle celle in un suo intorno.

Un contorno di un'immagine può avere un'inclinazione qualsiasi, quindi l'algoritmo di Canny usa quattro filtri differenti per individuare i contorni orizzontale, verticale e diagonali dell'immagine, a cui è stato precedentemente applicato il filtro gaussiano. Per ciascuna cella viene assunta come valida l'inclinazione relativa al filtro che dà il maggior gradiente di luminosità.

L'estrazione dei contorni dalla mappa dei gradienti generata dallo step precedente si esegue definendo due soglie,  $\alpha$  e  $\beta$  (poste rispettivamente a 90 e 100 in questo lavoro di tesi), che vengono confrontate con il gradiente in ciascuna cella. Se il valore del gradiente è:

- inferiore alla soglia  $\alpha$ , il punto è scartato;
- superiore alla soglia  $\beta$ , il punto è accettato come parte di un contorno;
- compreso fra le due soglie, il punto è accettato solamente se contiguo ad un punto già precedentemente accettato.

In Figura 3.10 (b) sono mostrati i contorni ottenuti applicando il Canny edge detector ad una mappa metrica (a).

### Hough line transform

I contorni individuati dal *Canny edge detector* sono processati dall'*Hough line transform*. Questa è una tecnica di estrazione di feature, il cui scopo è identificare segmenti di retta in un'immagine, caratterizzati dalle coordinate degli estremi. L'importanza dell'estrazione dei segmenti corrispondenti ai muri sta nel fatto che essi costituiscono la base su cui sviluppare la segmentazione dello spazio bidimensionale allo scopo di costruire il layout delle stanze.

Le rette vengono tipicamente parametrizzate nella forma  $y = mx + c$ . Tuttavia, per poter ovviare al problema del valore di  $m$  tendente all'infinito nel caso di rette verticali, si considerano le rette nella *Hesse normal form*

$$r = x \cos\theta + y \sin\theta \quad (3.1)$$

in cui  $r$  indica la distanza tra la retta e l'origine del sistema di riferimento, mentre  $\theta$  indica l'angolo tra l'asse  $x$  e la retta perpendicolare alla retta passante per l'origine.

L'Hough line transform costruisce un array bidimensionale che rappresenta lo spazio dei parametri, con tante righe e colonne quanti sono rispettivamente i diversi valori di  $r$  e  $\theta$ . Per ogni combinazione dei parametri  $r$  e  $\theta$  l'algoritmo individua il numero di celle dell'immagine che appartengono alla retta identificata da quella coppia di parametri.

I massimi locali nell'array indicano quali sono i parametri delle rette la cui presenza all'interno dell'immagine è più probabile. L'algoritmo infine confronta ulteriormente l'immagine con le rette individuate in modo da poterne estrarre dei segmenti caratterizzati dalle coordinate degli estremi.

In questo lavoro di tesi, i segmenti restituiti dall'algoritmo corrispondono ai muri rappresentati nella mappa metrica. In Figura 3.10 (c) vengono mostrati i segmenti estratti dall'Hough line transform, applicato sui contorni ricavati dal Canny edge detector (b). Si nota la corrispondenza tra questi segmenti ed i muri della mappa metrica (a).



*Figura 3.10: (a) Mappa metrica; (b) Contorni ricavati dalla mappa metrica per mezzo del Canny edge detector; (c) Segmenti ottenuti applicando l’Hough line transform sui contorni estratti in (b). I segmenti corrispondono ai muri di (a).*

Una volta terminata questa fase, sono stati ottenuti i segmenti corrispondenti ai muri. Queste feature geometriche sono la base su cui poter sviluppare la segmentazione che porta all’ottenimento del layout delle stanze. L’obiettivo finale consiste nell’eseguire clustering su delle celle in modo da raggrupparle in stanze. Il problema dunque ora consiste nell’ottenere un insieme di rette, tramite le cui intersezioni possa essere generato un insieme di celle. Una possibile soluzione è generare l’insieme di rette in modo che ognuna di esse rappresenti un gruppo di segmenti affini tra loro. L’affinità tra i segmenti può essere valutata in base al loro orientamento e distanza. Vengono dunque clusterizzati i segmenti secondo i due criteri di orientamento e distanza laterale, come di seguito descritto.

### 3.3.4 Calcolo delle rette rappresentative dei muri

Viene eseguito un primo clustering angolare in base all’orientamento dei segmenti. Per mezzo delle coordinate degli estremi viene calcolato l’*orientamento*

di ciascun segmento, ovvero l'angolo tra la retta su cui giace il segmento e l'asse  $x$  del sistema di riferimento. I cluster angolari possono essere individuati utilizzando un algoritmo di clustering che prenda in considerazione gli orientamenti dei segmenti. Analogamente al lavoro in [68], per eseguire quest'operazione è stato scelto *mean-shift*, descritto in [15].

### Clustering angolare con mean-shift

Il mean-shift è una tecnica di clustering non parametrica, la quale non richiede conoscenza preliminare del numero di cluster. L'algoritmo viene applicato iterativamente sui dati in ingresso, un insieme  $P$  che in questo lavoro di tesi consiste negli orientamenti dei segmenti. Allo step 0 i *cluster center*  $T^0$  coincidono con tutti gli orientamenti dei segmenti. Ad ogni step  $i$  i nuovi cluster center  $T^i = t_i$  sono calcolati come segue. La *sample mean*  $m(t_i)$  viene calcolata usando la *kernel function*  $K$ :

$$m(t_i) = \frac{\sum_{p \in P} K(p - t_i)p}{\sum_{p \in P} K(p - t_i)} \quad (3.2)$$

La kernel function usata in (3.2) per ottenere i cluster angolari è:

$$K(\theta) = \begin{cases} [1 - \frac{1-\cos(\alpha-\theta)}{h}]^2 & \text{se } \frac{1-\cos(\alpha-\theta)}{h} \leq 1 \\ 0 & \text{se } \frac{1-\cos(\alpha-\theta)}{h} > 1 \end{cases} \quad (3.3)$$

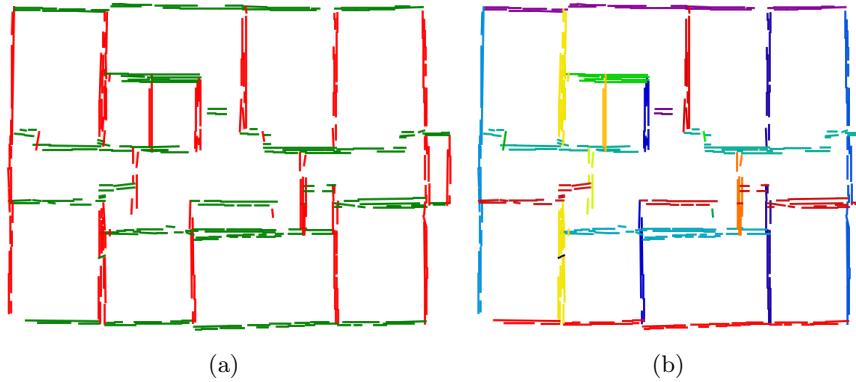
Nella formula (3.3)  $\alpha$  corrisponde al cluster center corrente,  $\theta$  corrisponde all'orientamento del segmento e  $h$  è una bandwidth arbitraria (settata a 0.023 in questo lavoro di tesi).

Applicando le formule (3.2) e (3.3) i cluster center per lo step successivo vengono modificati da  $t_i$  a  $m(t_i)$  per ogni  $t_i \in T^i$ .

Questo processo continua finché  $\max(m(t_i) - t_i)$  è sufficientemente piccolo (inferiore ad una soglia arbitraria, nel nostro caso settata a 0.00001). Una volta terminato l'algoritmo, i segmenti corrispondenti ai muri sono stati raggruppati in cluster angolari, basandosi sul loro orientamento. In Figura 3.11 (a) è mostrato un esempio di clustering angolare operato sui segmenti rappresentati in Figura 3.10 (c). I muri sono colorati in base al cluster angolare a cui appartengono. In questo caso sono stati individuati due cluster angolari (rosso e verde).

### Clustering spaziale

In questo step i segmenti appartenenti ad uno stesso cluster angolare vengono ulteriormente raggruppati in base alla prossimità spaziale. Per svolgere

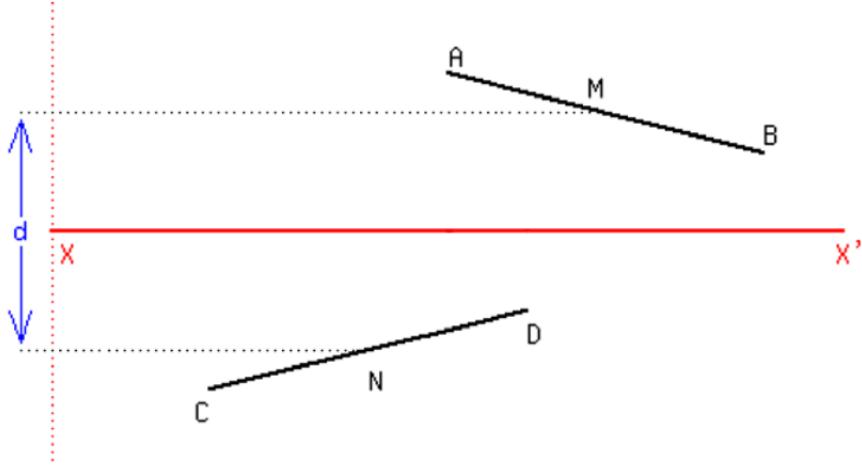


*Figura 3.11: (a) Clustering angolare dei segmenti in Figura 3.10 (c). Segmenti dello stesso colore appartengono allo stesso cluster angolare; (b) Clustering spaziale eseguito sui segmenti appartenenti ad uno stesso cluster angolare, in base alla loro distanza laterale. Ad ogni cluster spaziale è associato un diverso colore.*

questa operazione viene considerata la misura relativa alla *distanza laterale*  $d$ : dati due segmenti, la loro distanza laterale  $d$  corrisponde alla distanza tra i loro punti medi proiettati sulla retta perpendicolare all'orientamento del cluster angolare a cui entrambi i segmenti appartengono. La Figura 3.12 mostra la nozione di distanza laterale. I segmenti  $AB$  e  $CD$  appartengono allo stesso cluster angolare in quanto hanno orientamento simile (sono approssimabili a segmenti orizzontali). Vengono calcolati i loro punti medi, rispettivamente  $M$  ed  $N$ , e proiettati sulla retta perpendicolare alla retta  $XX'$  che ha lo stesso orientamento del cluster angolare di  $AB$  e  $CD$ . La distanza  $d$  tra le due proiezioni corrisponde alla distanza laterale tra i due segmenti. Se  $d$  è inferiore a una soglia arbitraria ( $90cm$ ), ai segmenti è assegnato lo stesso cluster spaziale.

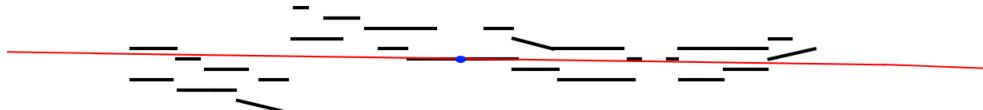
Per mezzo di questo processo viene ottenuto un insieme di cluster spaziali dei segmenti corrispondenti ai muri. Si noti che segmenti appartenenti allo stesso cluster spaziale appartengono necessariamente anche al medesimo cluster angolare, poiché il clustering spaziale viene applicato su segmenti precedentemente raggruppati in uno stesso cluster angolare. In Figura 3.11 (b) viene mostrato un esempio di clustering spaziale. Segmenti dello stesso colore appartengono al medesimo cluster spaziale.

A ciascun cluster spaziale viene associata una retta rappresentativa, come mostrato in Figura 3.13. La figura rappresenta il risultato del clustering spaziale su un gruppo di segmenti di Figura 3.10 (c). La retta rappresentativa è una retta (in rosso) identificata da un punto (in blu) e da un orientamento. L'orientamento è dato dal cluster angolare comune ai seg-



*Figura 3.12: AB e CD sono due segmenti appartenenti allo stesso cluster angolare; M e N sono i punti medi dei segmenti; XX' è una retta con orientamento pari a quello del cluster angolare dei due segmenti. I punti medi sono proiettati sulla retta perpendicolare a XX', e la distanza d tra le proiezioni corrisponde alla distanza laterale tra i due segmenti. La figura è tratta da [50].*

menti, mentre il punto coincide con il mediano dei punti medi dei segmenti del cluster spaziale.

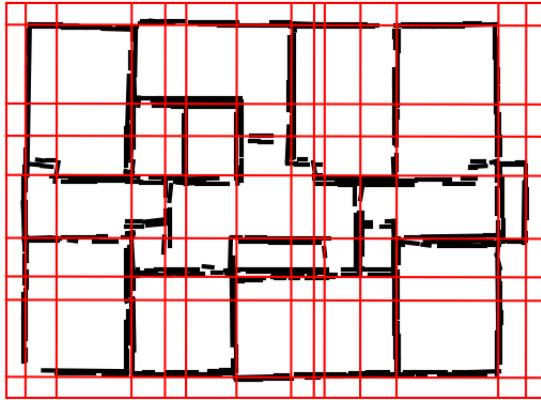


*Figura 3.13: Retta rappresentativa (in rosso) dei segmenti appartenenti ad uno stesso cluster spaziale. L'orientamento coincide con il cluster angolare dei segmenti, mentre il punto (in blu) è il mediano dei punti medi.*

### 3.3.5 Partizionamento in celle della mappa metrica

Le rette rappresentative sono delle rette che rappresentano i cluster spaziali in cui sono stati raggruppati i muri. Tali rette generano, per mezzo delle loro intersezioni, un insieme di celle, come mostrato in Figura 3.14.

L'obiettivo dell'identificazione delle celle è quello di raggruppare celle adiacenti in stanze, in modo da ottenere un modello che mostri la disposizione e separazione delle diverse stanze che compongono l'edificio. Analogamente



*Figura 3.14: Insieme di celle generato dalle intersezioni delle rette rappresentative (in rosso). Queste rette rappresentano i cluster spaziali dei segmenti (in nero).*

mente al clustering dei segmenti in base ad orientamento e distanza spaziale, anche per eseguire il clustering delle celle è necessario stabilire un criterio di affinità, in base al quale possa essere compiuto il raggruppamento di celle affini. Per valutare l'affinità tra due celle adiacenti una possibile idea consiste nel considerare quanto l'edge che le separa sia realmente corrispondente ad un muro. A questo scopo ad ogni edge viene assegnato un peso che indichi tale caratteristica.

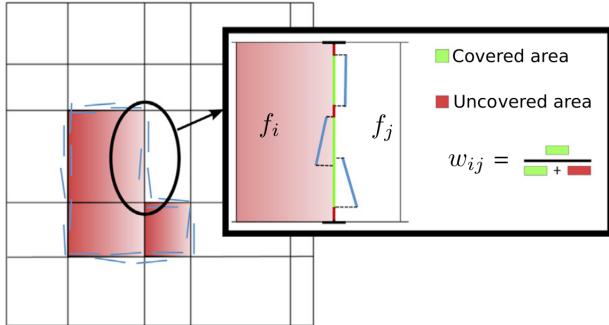
### Calcolo del peso degli edge

Per effettuare il raggruppamento delle celle in stanze, a ciascun edge (lato) delle celle viene associata la retta rappresentativa di cui l'edge fa parte e quindi il corrispondente cluster spaziale. Dato un edge  $e_{ij}$  che separa due celle adiacenti  $f_i$  ed  $f_j$ , gli viene assegnato un peso  $w_{ij}$  che indica la sua probabilità di corrispondere ad un muro realmente presente nella mappa metrica. Come mostrato in Figura 3.15, a questo scopo si considerano tutti i relativi muri candidati, ovvero tutti i segmenti (di colore blu) appartenenti allo stesso cluster spaziale di  $e_{ij}$ , e li si proiettano su  $e_{ij}$ , l'edge cerchiato e messo in evidenza nel riquadro nella figura. Si denota come  $cov(e_{ij})$  la porzione di  $e_{ij}$  coperta da queste proiezioni, rappresentata dalla parte verde dell'edge, mentre la parte rossa rappresenta la porzione di edge non coperta da alcuna proiezione.

Il peso dell'edge viene definito come segue:

$$w_{ij} = \frac{cov(e_{ij})}{length(e_{ij})} \quad (3.4)$$

dove  $length(e_{ij})$  è la lunghezza dell'edge  $e_{ij}$  e coincide con la somma delle porzioni verdi e rosse.



*Figura 3.15: Calcolo del peso di un edge. I segmenti (in blu) dello stesso cluster spaziale dell'edge sono proiettati su di esso, e il rapporto tra la porzione occupata e l'intera lunghezza dell'edge viene assegnato come peso dell'edge. La figura è tratta da [68].*

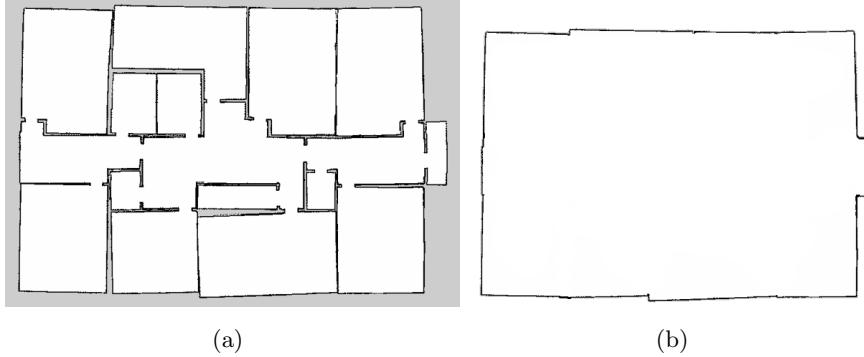
Una volta calcolati i pesi degli edge è possibile procedere con il raggruppamento delle celle in stanze. Per ottenere le stanze vogliamo che il processo di clustering venga eseguito sulle celle interne all'edificio e completamente percepite, ovvero sulle celle che corrispondono a regioni di spazio comprese entro le mura perimetrali e non appartenenti ad ambienti parziali. Poichè non tutte le celle possiedono questa caratteristica, è necessario classificarle in modo da distinguere le celle interne e completamente percepite da quelle esterne e parziali.

### Classificazione delle celle

Una volta calcolati i pesi degli edge tra celle adiacenti, l'insieme di celle viene suddiviso in celle *interne*, *esterne* e *parziali*. La suddivisione viene operata analizzando l'intersezione tra la superficie di ciascuna cella e la superficie della mappa metrica corrispondente all'ambiente interno dell'edificio. Quest'ultima può essere ottenuta individuando i contorni esterni della mappa metrica, estratti tramite l'utilizzo in sequenza degli algoritmi di *Canny edge detection* e *Hough line transform*.

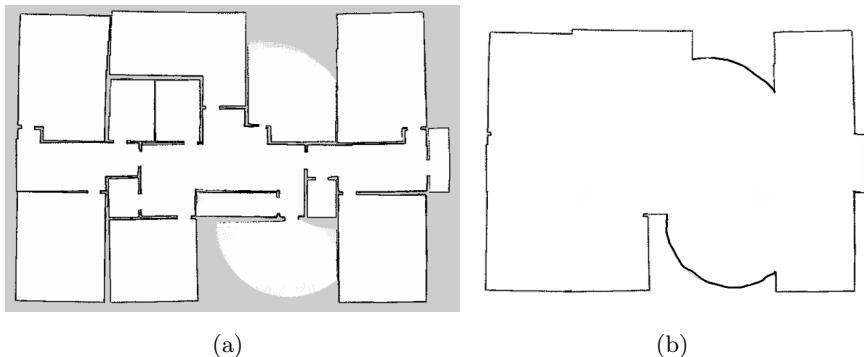
Nel caso in cui la mappa metrica consista in una planimetria o in una occupancy grid map completa (derivata da un esplorazione completa da parte del robot), i contorni esterni corrispondono alle mura perimetrali dell'edificio, come mostrato in Figura 3.16.

Diverso è invece il caso in cui la mappa metrica sia una occupancy grid map parziale, generata da un robot che ha esplorato l'ambiente in modo



*Figura 3.16: (a) Occupancy grid map completa; (b) Contorni esterni corrispondenti alle mura perimetrali.*

incompleto. In questo caso i contorni esterni della mappa metrica includono, oltre alle mura perimetrali, anche le frontiere tra spazio conosciuto e sconosciuto, come mostrato in Figura 3.17.



*Figura 3.17: (a) Occupancy grid map parziale; (b) Contorni esterni corrispondenti alle mura perimetrali e alle frontiere.*

Le celle la cui intersezione con l'ambiente interno è nulla o inferiore ad una soglia  $\delta$ , sono classificate come esterne. In questo lavoro di tesi,  $\delta$  è stato calcolato come metà dell'area della cella che si sta valutando. Le celle esterne corrispondono alle regioni di spazio di cui il robot non ha raccolto alcun dato sensoriale, perché al di fuori delle mura perimetrali dell'edificio oppure oltre la frontiera di una stanza esplorata parzialmente.

Le celle classificate come non esterne vengono analizzate in modo da operare una distinzione tra interne e parziali. A questo scopo ci si basa sull'assunzione che un edge  $e_{ij}$  con peso  $w_{ij}$  trascurabile difficilmente corrisponderà ad un muro realmente presente nella mappa metrica. Sono dunque classificate come parziali le celle  $f_i$  adiacenti ad una cella esterna  $f_j$  per mez-

zo di un edge  $e_{ij}$  il cui peso  $w_{ij}$  è inferiore a una soglia  $\mu$ , a cui si è deciso di associare il valore 0.2. In modo analogo, una volta classificata una cella  $f_i$  come parziale viene analizzata ciascuna cella  $f_j$  ad essa adiacente, classificando anch'essa come parziale se l'edge  $e_{ij}$  che le separa ha un peso inferiore a  $\mu$ . Le celle classificate come parziali corrispondono a porzioni di stanze della mappa metrica che non sono state interamente esplorate dal robot.

La classe parziale non viene propagata da una cella parziale  $f_i$  ad una cella adiacente  $f_j$  se l'edge  $e_{ij}$  che le separa ha un peso superiore a  $\mu$ . Un edge con questa caratteristica ha un'alta probabilità di corrispondere ad un muro realmente presente nella mappa metrica. Di conseguenza le due celle verosimilmente non appartengono alla stessa stanza. Una volta identificate le celle parziali, le celle non classificate come esterne né come parziali vengono classificate come interne. Queste, una volta raggruppate in base alla loro affinità, formeranno le stanze dell'edificio la cui esplorazione è avvenuta in modo completo.

### 3.3.6 Raggruppamento delle celle interne in stanze

Una volta conclusa la classificazione delle celle, viene stabilita una misura di affinità globale per ogni coppia di celle interne. A questo scopo vengono usati i pesi degli edge per derivare una matrice di affinità  $L$ , definita come:

$$L_{ij} = \begin{cases} e^{-w_{ij}/\sigma}, & \text{se } i \neq j \wedge f_i, f_j \text{ sono adiacenti,} \\ 1, & \text{se } i = j \\ 0 & \text{altrimenti} \end{cases} \quad (3.5)$$

Per mezzo della matrice  $L$  viene definita una matrice di transizione  $M = D^{-1}L$ . La matrice  $D = \text{diag}(\sum_{j=1}^n L_{ij})$  è una matrice diagonale, ovvero una matrice quadrata in cui solamente i valori della diagonale principale possono essere diversi da 0. Ciascun valore  $D_{ii}$  corrisponde alla somma dei valori contenuti nella  $i$ -esima riga di  $L$ .

Ogni elemento  $M_{ij}$  può essere visto come un valore di affinità locale tra le celle  $f_i$  e  $f_j$ . La matrice di affinità  $M$  ricopre un ruolo fondamentale nel raggruppamento delle celle in stanze. Questo processo si basa infatti sui valori di affinità contenuti in  $M$ , raggruppando nello stesso cluster quelle con alta affinità. Ogni cluster di celle individuato corrisponde a una stanza dell'ambiente. Per conseguire questo risultato, viene utilizzato l'algoritmo DBSCAN.

## DBSCAN

DBSCAN, presentato in [23], è un algoritmo di clustering basato sul concetto di *densità*, in quanto connette regioni di punti con densità sufficientemente alta. DBSCAN richiede due importanti parametri:

1.  $\varepsilon$  (eps): definisce il raggio di vicinanza attorno ad un punto  $x$ . In questo lavoro di tesi gli è stato assegnato il valore 0.85.
2.  $\text{min}Pts$ : indica il numero minimo di vicini compresi nell' $\varepsilon$ -vicinato di un punto  $x$ , ovvero la cui distanza da  $x$  è inferiore a  $\varepsilon$ . In questo lavoro di tesi è stato settato a 1.

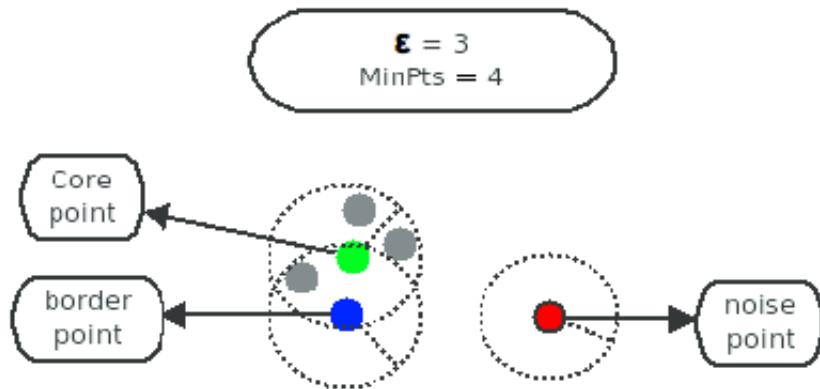
Ciascun punto  $x$  nel dataset che abbia un numero di  $\varepsilon$ -vicini uguale o maggiore a  $\text{min}Pts$  viene marcato come *core point*. Se invece un punto  $x$  possiede un numero di  $\varepsilon$ -vicini inferiore a  $\text{min}Pts$ , ma si trova nell' $\varepsilon$ -vicinato di un core point, allora viene marcato come *border point*. Infine se un punto  $x$  non è marcato come core point né come border point allora è considerato *outlier*. In Figura 3.18, tratta da [112], sono mostrati i diversi tipi di punti (core, border e outlier) usando  $\text{min}Pts = 4$  e  $\varepsilon = 3$ . Il punto di colore verde è un core point in quanto il numero di punti compresi nel suo  $\varepsilon$ -vicinato è uguale a 4 ( $\geq \text{min}Pts$ ). Il punto di colore blu è un border point poichè i suoi  $\varepsilon$ -vicini sono 2 ( $< \text{min}Pts$ ) ed appartiene all' $\varepsilon$ -vicinato di un core point. Il punto di colore rosso infine è un outlier in quanto il numero di suoi  $\varepsilon$ -vicini è 0 ( $< \text{min}Pts$ ) ed esso non appartiene all' $\varepsilon$ -vicinato di alcun core point.

Per comprendere l'algoritmo DBSCAN è necessario definire 3 proprietà:

1. **Density reachability diretta**: un punto  $a$  è *density reachable* da un altro punto  $b$  se  $a$  è nell' $\varepsilon$ -vicinato di  $b$  e  $b$  è un core point.
2. **Density reachability**: un punto  $a$  è *density reachable* da un altro punto  $b$  se esiste almeno una sequenza  $p_1, \dots, p_n$  di punti con  $p_1 = a$  e  $p_n = b$  in cui ogni  $p_{i+1}$  è direttamente density reachable da  $p_i$ .
3. **Density connection**: due punti  $a$  e  $b$  sono *density connected* se esiste un core point  $c$  tale che sia  $a$  che  $b$  sono density reachable da  $c$ .

Un cluster in DBSCAN è definito come un gruppo di punti *density connected*. Il funzionamento dell'algoritmo è di seguito descritto:

1. Per ogni punto  $x$  viene calcolata la sua distanza dagli altri punti del dataset. Ogni punto il cui  $\varepsilon$ -vicinato contenga un numero di punti maggiore o uguale a  $\text{min}Pts$  viene marcato come core point.



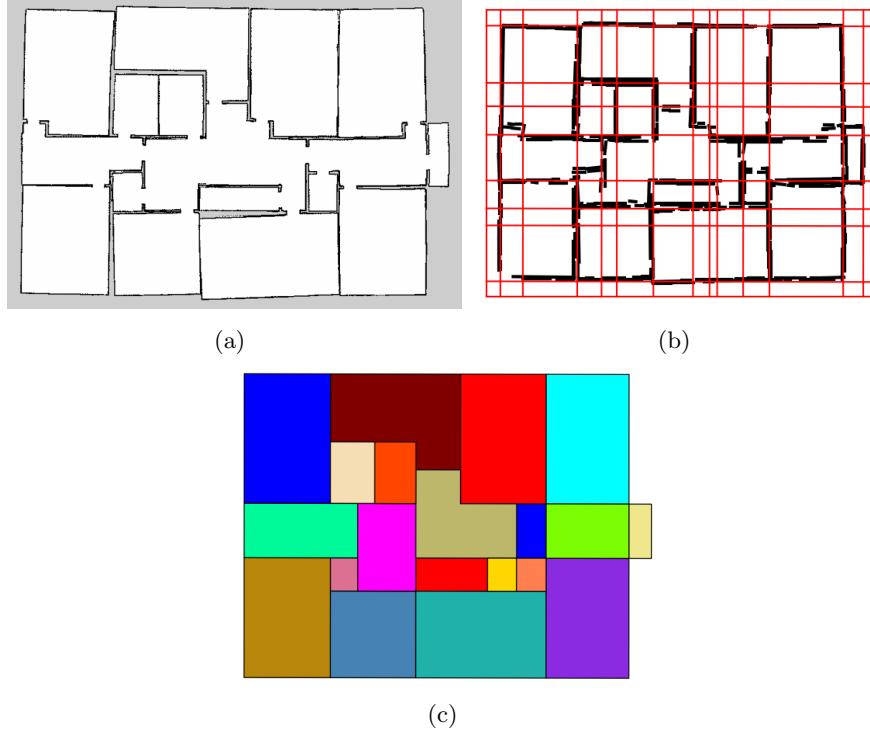
*Figura 3.18: Esempio di core point, border point e outlier, definiti in base ai valori di  $\epsilon$  e  $\text{minPts}$ , con  $\text{minPts} = 4$  e  $\epsilon = 3$ . Il punto di colore verde è un core point poichè possiede 4  $\epsilon$ -vicini ( $\geq \text{minPts}$ ). Il punto di colore blu è un border point in quanto i suoi  $\epsilon$ -vicini sono 2 ( $< \text{minPts}$ ) ed esso è nell' $\epsilon$ -vicinato di un core point. Il punto di colore rosso è un outlier. L'immagine è tratta da [112].*

2. Per ogni core point  $x$  viene creato un nuovo cluster se  $x$  non è stato ancora assegnato ad alcun cluster. Sono ricorsivamente assegnati allo stesso cluster del core point  $x$  tutti i punti density connected a  $x$ .
3. Dopo aver completato l'iterazione su tutti i punti del dataset non ancora visitati, sono marcati come outlier i punti che non appartengono ad alcun cluster.

In questo lavoro di tesi l'insieme di punti su cui l'algoritmo di DBSCAN deve eseguire il clustering corrisponde all'insieme delle celle in cui è stata partizionata la mappa. Per ogni coppia di celle  $f_i$  e  $f_j$ , DBSCAN analizza il valore  $d_{ij} = 1 - M_{ij}$  che indica la distanza tra le celle  $f_i$  e  $f_j$  in termini di affinità.

Una volta completato il clustering per mezzo di quest'algoritmo, si è ottenuto un raggruppamento di celle adiacenti; ciascun raggruppamento rappresenta una stanza, una regione di spazio costituita da celle affini, verosimilmente non separate da muri. Il modello così costruito costituisce il layout delle stanze, una suddivisione della mappa metrica in grado di mostrare la disposizione e separazione delle diverse stanze che compongono l'ambiente indoor dell'edificio, come mostrato in Figura 3.19 e Figura 3.20. Questa rappresentazione eleva il grado di astrazione della mappa metrica, estenden-

done il livello di conoscenza, e costituisce la base su cui costruire il layer successivo.



*Figura 3.19: (a) Mappa metrica; (b) Suddivisione in celle per mezzo delle rette rappresentative; (c) Layout delle stanze, ottenuto raggruppando celle interne adiacenti tramite DBSCAN.*

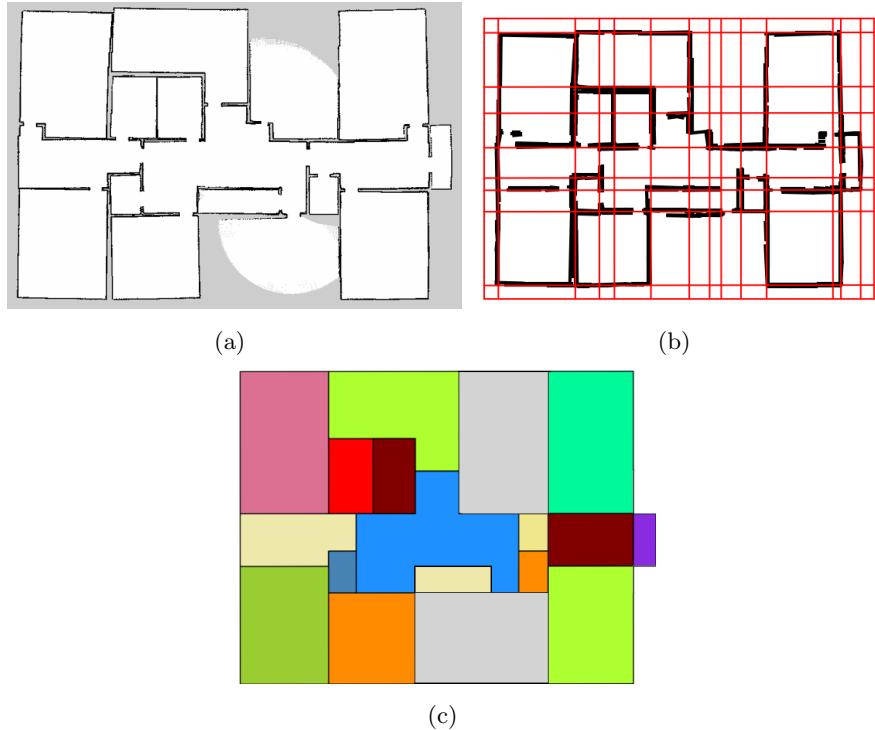
## 3.4 Grafo topologico

### 3.4.1 Obiettivo

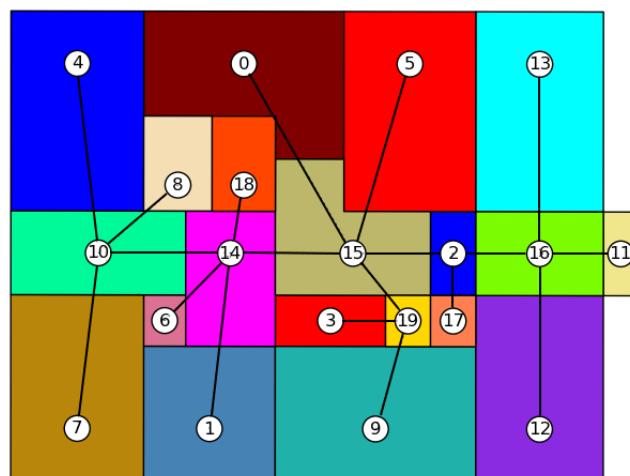
Il grafo topologico utilizza la conoscenza offerta dal layout delle stanze allo scopo di estrarre nuove informazioni e creare un ulteriore livello di conoscenza.

L'obiettivo di questa mappa consiste nel mostrare, attraverso un grafo  $G = (N, A)$ , la connettività fra le regioni di spazio, le stanze, che compongono l'edificio. Un esempio di grafo topologico, riportato sul corrispondente layout delle stanze, è mostrato in Figura 3.21.

Ciascun nodo  $n \in N$  del grafo  $G$  rappresenta le stanze individuate dalla segmentazione della mappa metrica, operata nella fase precedente. Per fare in modo che il layout del grafo  $G$  rispecchi il layout delle stanze, si



*Figura 3.20: (a) Mappa metrica parziale; (b) Suddivisione in celle per mezzo delle rette rappresentative; (c) Layout delle stanze in cui le stanze parziali, costituite dalle celle classificate come parziali, sono colorate di grigio.*



*Figura 3.21: Esempio di grafo topologico riportato sul rispettivo layout delle stanze. I nodi corrispondono alle stanze e contengono il numero del cluster individuato in precedenza da DBSCAN. Gli archi rappresentano le connessioni tra le stanze.*

è deciso di associare ad ogni nodo del grafo una posizione corrispondente ad un punto rappresentativo della stanza che esso rappresenta. Tale punto è ottenuto invocando un'apposita funzione sul poligono della stanza, come sarà mostrato nella Sezione 4.4.3. In questo modo il grafo  $G$  è in grado di offrire una rappresentazione intuitiva della suddivisione dell'ambiente. I numeri riportati all'interno di ogni nodo indicano il cluster individuato da DBSCAN nella fase di raggruppamento delle celle in stanze.

Gli archi  $a \in A$  rappresentano le connessioni tra le stanze. Due nodi del grafo topologico sono connessi da un arco se le corrispondenti aree nella mappa metrica sono direttamente collegate, cioè è possibile transitare direttamente da una all'altra, ad esempio tramite una porta o più genericamente attraverso un passaggio libero da ostacoli.

Allo scopo di ricavare l'informazione riguardante le connessioni tra le stanze, possono essere utilizzati metodi dallo stato dell'arte, fra i quali gli approcci basati su Voronoi graph [4].

### 3.4.2 Voronoi graph

Il Voronoi graph è una partizione spaziale di una mappa, tipicamente applicata sulla mappa metrica. Per calcolare il Voronoi graph  $G(m) = (V, E)$  di una mappa  $m$ , si considera l'insieme  $O_p(m)$  che contiene per ogni punto  $p$  nello spazio libero  $C$  di  $m$  l'insieme dei punti occupati più vicini. Il Voronoi graph è poi dato dall'insieme di punti in  $O_p(m)$  aventi più di un ostacolo alla stessa distanza minima (*basis point*):

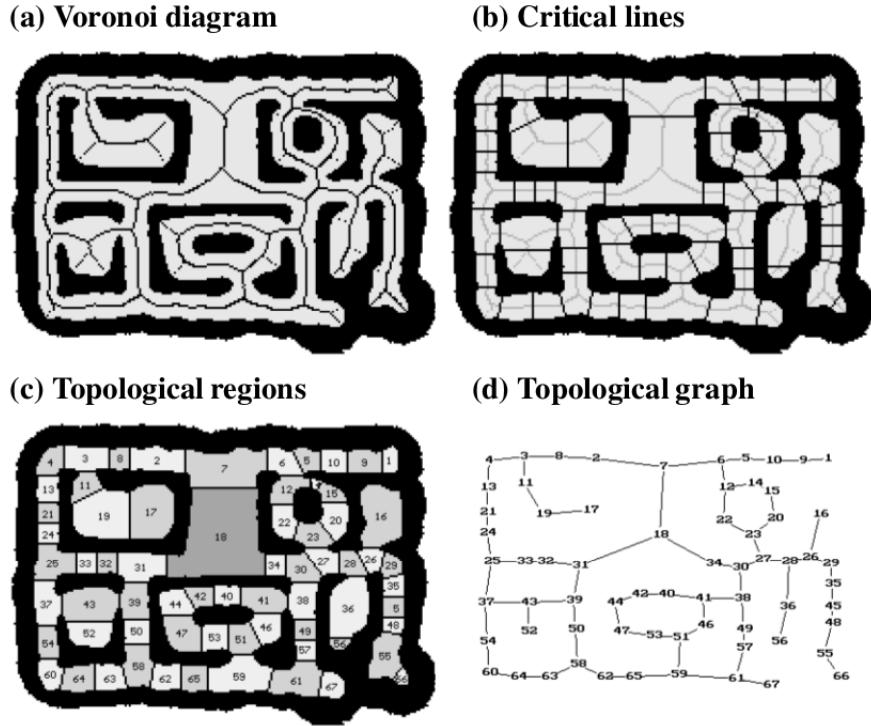
$$V = \{p \in C \mid |O_p(m)| \geq 2\}$$

$$E = \{(p, q) \mid p, q \in V, p \text{ adiacente a } q \text{ in } m\}$$

Per ogni coppia di nodi in  $G(m)$  viene aggiunto un arco se i punti corrispondenti in  $m$  sono adiacenti, ovvero collegabili con un segmento che non interseca alcun ostacolo.

I lavori in [106], [105] e [49] utilizzano il Voronoi graph allo scopo di trovare i suoi *punti critici*, ossia i punti corrispondenti a porte e passaggi stretti. I punti critici sono poi collegati ai propri basis point, generando le *linee critiche*, che segmentano lo spazio e ne identificano la connettività come mostrato in Figura 3.22.

In [7] viene impiegato l'*extended Voronoi graph* (EVG), un Voronoi graph composto da punti la cui distanza dal muro più vicino è sempre inferiore ad una soglia predefinita  $\kappa$ . I punti di questo Voronoi graph hanno dunque la caratteristica di rimanere in prossimità dei muri dell'edificio, come mostrato



*Figura 3.22: Esempio di utilizzo del Voronoi graph in [106]: (a) Voronoi graph; (b) Punti e linee critiche; (c) Regioni separate; (d) Grafo topologico estratto.*

in Figura 3.23. L’extended Voronoi graph può essere definito come l’unione di due insiemi di punti:

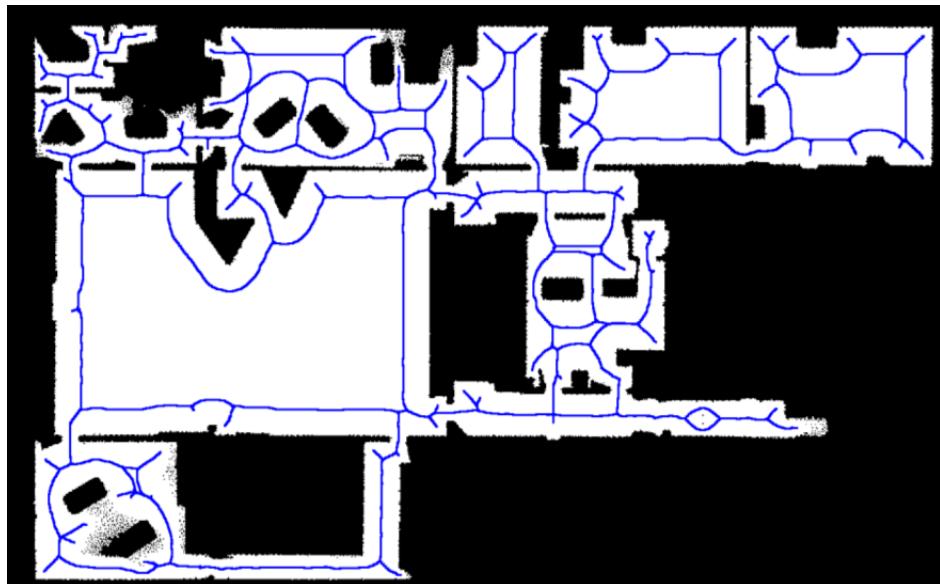
- Il sottoinsieme dei punti del Voronoi graph generalizzato la cui distanza dagli ostacoli più vicini è inferiore a  $\kappa$ .
- I punti la cui distanza dall’ostacolo più vicino equivale esattamente a  $\kappa$ .

Più formalmente, dati i punti  $p$  del Voronoi graph generalizzato e l’insieme  $O$  degli ostacoli, viene definito l’insieme  $P$  dei punti dell’EVG come segue:

$$\begin{aligned}
 P = & \{p : \exists o, o' \in O, \text{ dove } o \neq o' \wedge \\
 & \quad dist(p, o) = dist(p, o') \wedge dist(p, o) \leq \kappa \wedge \\
 & \quad \forall o'' \in O \ dist(p, o'') \geq dist(p, o)\} \\
 & \cup \\
 & \{q : \exists o \in O, \text{ dove } dist(q, o) = \kappa \wedge \\
 & \quad \forall o' \in O \ dist(q, o') \geq \kappa\}
 \end{aligned}$$

L'ambito in cui questo approccio apporta il vantaggio più rilevante è l'esplorazione di aree le cui dimensioni superano l'orizzonte sensoriale del robot.

In questi scenari il Voronoi graph generalizzato, utilizzato nel processo di definizione di percorsi, non produce risultati ottimali, poiché invece di definire percorsi paralleli ai muri, ne definisce uno che volge al centro dell'area. Seguendo questo percorso, il robot si ritroverebbe in una posizione da cui, a causa del suo limitato orizzonte sensoriale, non sarebbe visibile alcun ostacolo. Di conseguenza, da questa posizione sarebbe impossibile per il robot calcolare ulteriormente il Voronoi graph a partire dai dati sensoriali locali. Utilizzando l'extended Voronoi graph viene invece offerta al robot una configurazione spaziale caratterizzata da percorsi prossimi ai muri nelle aree di grandi dimensioni. In questo modo viene garantita la possibilità di acquisire dati sensoriali locali, funzionali all'orientamento.



*Figura 3.23: Esempio di extended Voronoi graph utilizzato in [7].*

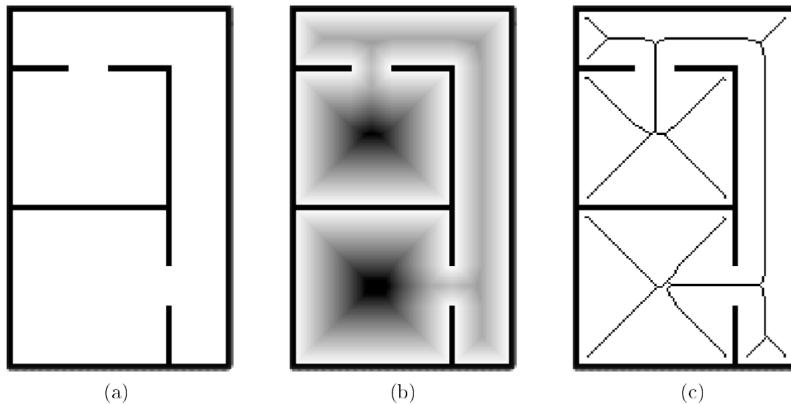
In questo lavoro di tesi, per soddisfare il task dell'identificazione delle connessioni tra le stanze, si è deciso di utilizzare un approccio differente da quelli appena citati. L'approccio non si basa sul Voronoi graph, bensì su uno strumento simile, il *medial axis*.

### 3.4.3 Medial axis

Il medial axis di una superficie, analogamente al Voronoi graph, viene definito in [19] come l'insieme dei punti aventi più di un ostacolo alla stessa distanza minima. In [19] viene anche descritta la relazione tra medial axis e Voronoi graph: nel caso di una superficie bidimensionale, i vertici del Voronoi graph convergono ai punti del medial axis.

Un lavoro che utilizza il medial axis allo scopo di individuare la connettività e la topologia di un ambiente è quello presentato in [117], il cui procedimento è mostrato in Figura 3.24. A partire dalla mappa metrica (a) viene ricavata in prima istanza la *distance map* (b), una mappa rappresentata come una matrice bidimensionale in cui ad ogni cella è associata la distanza dall'ostacolo più vicino. Il colore va dal bianco al nero allontanandosi dagli ostacoli. Da questa mappa viene in seguito ottenuto il medial axis (c), analizzando le distanze salvate nelle celle della matrice.

In questo lavoro di tesi il medial axis viene calcolato seguendo l'approccio



*Figura 3.24: Ottenimento del medial axis in [117]: (a) Mappa metrica; (b) Distance map; (c) Medial axis.*

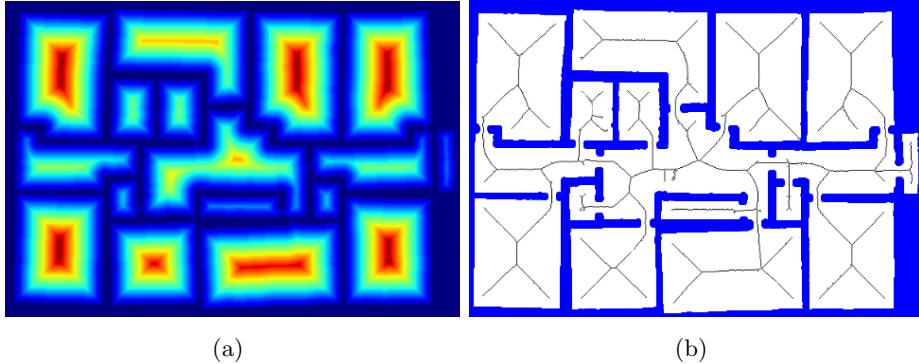
presentato in [117]:

1. Viene inizialmente applicata la *Euclidean distance transform* alla mappa metrica. Questa è una trasformazione che produce una distance map (Figura 3.25 (a)), calcolando per ogni cella dello spazio libero la *distanza euclidea* dalla cella occupata più vicina. Date due celle  $P = (p_x, p_y)$  e  $Q = (q_x, q_y)$ , il valore della loro distanza euclidea  $D_{eu}$  è

$$D_{eu} = \sqrt{(p_x - q_x)^2 + (p_y - q_y)^2}$$

Le celle della distance map il cui valore è 0 corrispondono alle celle occupate della mappa metrica.

- Il medial axis, mostrato in Figura 3.25 (b), viene ricavato dalla distance map tramite un'analisi delle distanze  $d$  salvate nelle celle della matrice. Appartengono al medial axis le celle  $c$  con valore  $d > 0$  per cui esiste più di una cella con valore 0 la cui distanza da  $c$  è pari a  $d$ .



*Figura 3.25: (a) Distance map della mappa metrica in Figura 3.19 (a). Il colore va dal blu al rosso allontanandosi dagli ostacoli; (b) Medial axis ottenuto dalla distance map.*

Per individuare le connessioni tra le stanze, vengono esaminati tutti i punti che compongono il medial axis. La posizione di ciascuno di questi punti viene confrontata con quella delle stanze individuate nel layer precedente. Se un punto del medial axis è localizzato in prossimità del confine tra due stanze, se ne deduce che le due stanze siano direttamente collegate tramite una porta o un passaggio privo di ostacoli. Le coppie di stanze così individuate vengono salvate come stanze tra loro collegate nel grafo topologico  $G$ .

#### 3.4.4 Analisi dei templates delle porte

Nel caso in cui la mappa metrica consista in una planimetria che contiene i simboli delle porte, possono essere ricavate informazioni aggiuntive riguardanti la connettività degli spazi. Se presenti nella planimetria, questi simboli sono estratti in una fase precedente, salvando la loro posizione all'interno della mappa metrica. Nel caso in cui la superficie del simbolo di una porta intersechi le superfici di due stanze, viene identificato un collegamento diretto tra le due stanze, aggiungendo la coppia di stanze a quelle già individuate applicando il medial axis.

Questo metodo non è un approccio sostitutivo del medial axis, bensì integrativo. Nel caso in cui nella mappa metrica siano presenti simboli di porte, vengono in prima istanza estratte informazioni sulla connettività per

mezzo del medial axis, integrandole in un secondo momento con le informazioni ricavate tramite questo approccio. Viene eseguita un'unione delle connessioni ricavate tramite i due approcci.

Una volta ottenute le coppie di stanze tra loro collegate, vengono tracciati i relativi archi nel grafo topologico, collegando i corrispondenti nodi come mostrato in Figura 3.21. Il grafo topologico così generato rappresenta il punto di partenza per lo sviluppo della mappa semantica.

## 3.5 Mappa semantica

### 3.5.1 Obiettivo

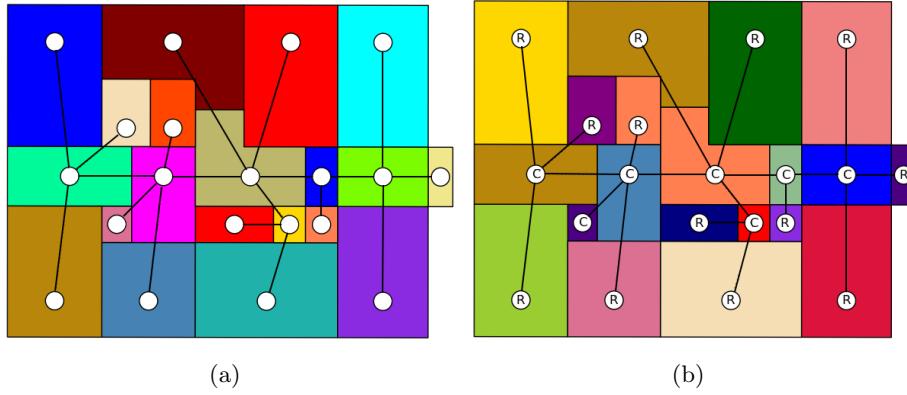
L'ultima rappresentazione del sistema multilivello è la mappa semantica. L'informazione semantica attribuisce un significato ad ogni stanza mediante una label con cui la stanza viene descritta; viene ottenuta attribuendo ad ogni stanza (nodo del grafo topologico) un'etichetta che ne descrive la funzione, come ad esempio *cucina*, *ufficio*, *corridoio*, ecc.

La finalità della mappa semantica è associare ciascuna porzione di spazio ad un'etichetta, operando una categorizzazione dell'ambiente simile a quanto farebbe una persona. Come mostrato in Figura 3.26 (b), la mappa semantica è costruita a partire dalla mappa topologica (a) arricchendone il contenuto informativo con un livello di conoscenza semantico. Le label *C* (corridoio) o *R* (stanza) vengono riportate all'interno dei nodi del grafo. In figura sia il grafo topologico che la mappa semantica sono riportati sul relativo layout delle stanze.

Questo livello di conoscenza può essere ulteriormente esteso integrando nella mappa semantica le informazioni relative a eventuali oggetti identificati nell'ambiente o indicati nella planimetria, come mostrato in Figura 3.27 (b). Il vantaggio di quest'operazione consiste nell'offrire al robot una conoscenza riguardante gli oggetti contenuti in ciascuna stanza.

Il task di classificazione semantica viene tipicamente eseguito tramite l'analisi, da parte di un classificatore, di un insieme di caratteristiche che descrivono ciascun oggetto da classificare. In base a quest'analisi può essere identificata la label che meglio descrive l'oggetto.

Il primo step della fase di classificazione consiste dunque nell'individuare una serie di feature tramite la cui analisi un classificatore possa predire ed associare una label a ciascuna stanza.



*Figura 3.26: (a) Grafo topologico riportato sul layout delle stanze. L'informazione relativa alle connessioni tra le stanze è rappresentata dagli archi che collegano i nodi del grafo; (b) Mappa semantica costruita sul layout delle stanze. L'informazione semantica consiste nelle etichette C o R applicate a ciascun nodo del grafo topologico, classificandolo rispettivamente come CORRIDOIO o STANZA.*

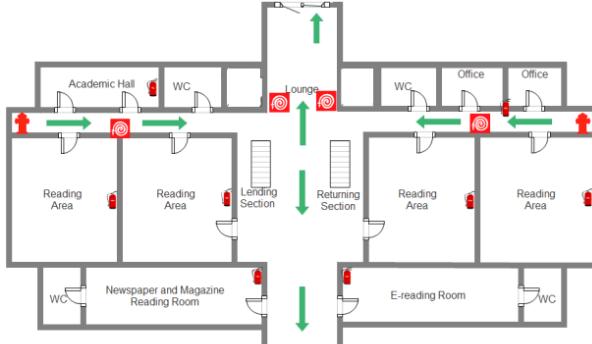
### 3.5.2 Le feature che descrivono le stanze

Per la scelta e il calcolo delle feature si è deciso di prendere spunto dal lavoro presentato in [65], in cui per classificare semanticamente mappe geometriche viene proposto un approccio basato sul *supervised learning*, descritto in [64], [104] e [116]. Il supervised learning è un processo attraverso il quale viene indotta una funzione partendo da dati di training etichettati. Ogni dato di training è una coppia che consiste in un oggetto di input (un vettore di feature) e in un valore di output desiderato (la label). L'algoritmo di supervised learning analizza i dati di training e ne ricava una funzione. Questa può essere usata per classificare nuovi esempi (dati di testing). Ciascun dato di testing deve essere passato al classificatore sotto forma di vettore di feature avente la stessa struttura del vettore che compone i dati di training. In [65] viene utilizzato un vettore consistente in un insieme di 16 feature in grado di descrivere l'oggetto a cui sono associate.

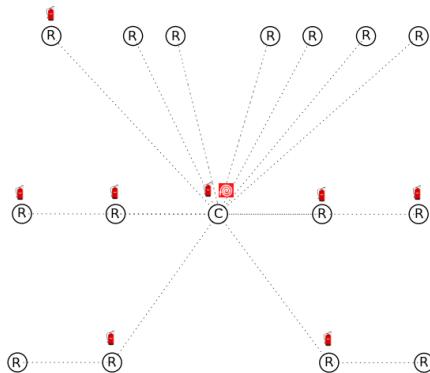
In questo lavoro di tesi si è scelto di utilizzare 8 delle 16 feature presentate in [65], selezionando le feature geometriche standard utilizzate in shape analysis. Queste vengono calcolate per ogni stanza  $r$ , ed una volta organizzate in un vettore  $V_r$ , vengono associate alla relativa stanza con lo scopo di descriverne la struttura geometrica.

Le feature che compongono il vettore  $V_r$  sono calcolate analizzando il poligono  $P(r)$  che approssima la forma della stanza  $r$ :

$$P(r) = (v_0, v_1, \dots, v_{N-1}, v_N \equiv v_0) \quad (3.6)$$



(a)



(b)

*Figura 3.27: (a) Esempio di planimetria di una scuola, tratta da [27]. Le informazioni riportate riguardano la struttura dell'edificio. Sono rappresentati i muri ed altri simboli come porte, estintori, idranti e testo; (b) Relativa mappa semantica in cui sono mostrate anche le informazioni riguardanti gli oggetti contenuti in ciascuna stanza, riportandone il simbolo in prossimità del nodo.*

Nell'equazione (3.6) i punti  $v_i = (x_i, y_i)$  rappresentano i vertici del poligono, mentre  $N$  è il loro numero totale.

Le feature, sottoinsieme di quelle utilizzate in [65], sono le seguenti:

**Area:** l'area del poligono  $P(r)$ , data da

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

**Perimeter:** il perimetro del poligono  $P(r)$ , dato da

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

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

**Rapporto tra area e perimetro:** calcolato come

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

**Distanza media tra il baricentro e il bordo del poligono:** calcolata come

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

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

date le coordinate  $c_x$  e  $c_y$  del baricentro  $c = (c_x, c_y)$  definito come

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

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

**Form factor:** il form factor del poligono  $P(r)$ , definito come

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

**Circolarità:** la circolarità del poligono  $P(r)$ , definita come

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

**Circolarità normalizzata:** la circolarità normalizzata di  $P(r)$ , definita come

$$f_{ncirc} = \frac{4\pi f_{area}}{f_{perimeter}^2}$$

**Distanza media normalizzata tra il baricentro e il bordo del poligono:** date le precedenti definizioni di baricentro e distanza dal baricentro, può essere calcolata come

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

$$\text{dove } \overline{dist}(v_i, c) = \frac{dist(v_i, c)}{\max_{\forall i} dist(v_i, c)}$$

Le feature appena elencate forniscono la base per un ragionamento *locale*, in quanto il vettore  $V_r$  si limita ad una descrizione geometrica della stanza  $r$ , ignorando il suo ruolo nell'ambiente in relazione alle altre stanze.

Al fine di testare una situazione in cui siano considerate anche informazioni *globali* oltre che *locali*, alle feature incluse nel vettore  $V_r$  se ne possono aggiungere altre due, le quali descrivono il ruolo della stanza nella struttura dell'edificio. Tali feature sono correlate alla nozione di *centralità*.

Le misure di centralità sono tipicamente utilizzate in network analysis e graph theory per valutare quantitativamente il ruolo di ogni nodo in un grafo: alti valori di centralità corrispondono ai più importanti nodi nella struttura di un grafo. Nel nostro contesto, la centralità è da intendersi come una misura dell'importanza di ogni stanza all'interno della struttura dell'edificio. Poichè il calcolo delle misure di centralità è fatto su un grafo, viene considerato il grafo topologico  $G$  ricavato nel layer precedente, in cui le stanze e le loro connessioni sono rappresentate rispettivamente da nodi  $n$  ed archi  $a$ .

Le due misure di centralità che sono integrate nel vettore di feature  $V_r$  sono *closeness centrality*, definita in [17], e *betweenness centrality*, presentata in [11].

**Closeness centrality:** la closeness centrality per un nodo  $n$  è definita in base alla minore distanza tra  $n$  e tutti gli altri nodi:

$$Closeness(n) = \sum_{t \in N \setminus \{n\}} 2^{-dG(n,t)}$$

dove  $dG(n, t)$  è la lunghezza del più breve percorso sul grafo  $G$  tra i nodi  $n$  e  $t$ , assegnando ad ogni arco un costo pari a 1.

**Betweenness centrality :** la betweenness centrality per un nodo  $n$  è definita come il numero di percorsi più brevi tra due nodi  $u \neq t (\neq n)$  che

passano per  $n$ :

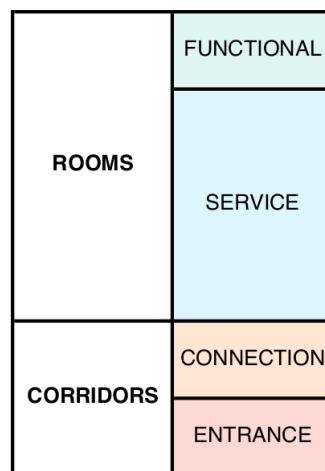
$$\text{Betweenness}(n) = \sum_{u,t \in N, u \neq t \neq n} \frac{T_{ut}(n)}{T_{ut}}$$

dove  $T_{ut}$  è il numero totale di percorsi più brevi da  $u$  a  $t$ ,  $T_{ut}(n)$  è il numero di questi percorsi passanti per  $n$ .

Una volta ottenute le feature che descrivono le stanze e raggruppate in un vettore, è necessario definire uno schema di labeling, in modo da stabilire l'insieme di label da cui può essere scelta quella da assegnare a una stanza.

### 3.5.3 Schema di labeling

In questo lavoro di tesi viene utilizzato uno schema gerarchico di labeling semantico, riportato in Figura 3.28.



*Figura 3.28: Schema di labeling utilizzato: ROOM e CORRIDOR costituiscono il livello di labeling superiore, mentre FUNCTIONAL, SERVICE, CONNECTION e ENTRANCE sono le label del livello inferiore.*

Il fine dell'utilizzo di questo schema è valutare l'impatto di diversi insiemi di label sulla classificazione. Al livello più alto, lo schema di labeling è  $L_{R/C}$  e contiene solo due etichette:

- **Room:** uno spazio in cui è svolta un'attività.
- **Corridor:** uno spazio usato per connettere altri spazi.

Viene definito un secondo insieme di label, chiamato  $L_{F/C/E/S} = \{\text{functional room, connection, entrance, service room}\}$ , in cui i corridor del livello precedente sono suddivisi in:

- **Connection:** uno spazio il cui scopo è connettere stanze interne ed appartenenti allo stesso piano.
- **Entrance:** uno spazio che costituisce una connessione con l'esterno dell'edificio o con stanze appartenenti ad un altro piano.

Mentre le room sono suddivise in:

- **Functional room:** uno spazio in cui è svolta l'attività principale dell'edificio (ad esempio una classe in una scuola).
- **Service room:** uno spazio usato per supportare le attività principali dell'edificio.

Questo generico schema di labeling è condiviso da tutte le tipologie di edifici, ovvero da tutte le classi di edifici aventi stessa funzione ed analoga struttura. In Figura 3.29 vengono mostrati gli schemi di labeling relativi alle tipologie *scuola* e *ufficio*. Si nota la comune composizione dei primi due livelli degli schemi (prime due colonne da sinistra), corrispondenti alla gerarchia mostrata in Figura 3.28. In questo lavoro di tesi viene preso a modello lo schema di labeling relativo alla tipologia scuola.

Una volta stabilito lo schema di labeling, possiamo definire il metodo di classificazione attraverso il quale associare una label ad ogni stanza. A tale scopo esistono numerosi approcci da stato dell'arte, come *support vector machine (SVM)* [10], *k-nearest neighbors (kNN)* [47] o *naive Bayes (NB)* [87]. In questo lavoro di tesi si è scelto di utilizzare *Extremely Randomized Trees (extra-trees)*, descritto in [32]. La scelta è motivata dall'efficienza computazionale e dall'accuratezza e robustezza dell'algoritmo.

### 3.5.4 Extremely Randomized Trees

L'algoritmo di extra-trees costruisce un insieme, o foresta, di alberi di decisione *unpruned* (non potati) usando una procedura di classificazione basata sugli alberi. Un albero di decisione è un modello in cui ogni nodo interno rappresenta una variabile ed ogni arco verso un nodo figlio rappresenta un possibile valore per tale variabile.

Gli extra-trees sono costruiti randomizzando fortemente sia gli *attributi* (variabili di input) che le scelte di *cut-point* (punto di uno spazio connesso la cui rimozione rende tale spazio disconnesso).

Extra-trees divide i nodi dell'albero scegliendo cut-point in modo completamente casuale ed utilizzando l'intero campione di apprendimento (osservazioni utili alla costruzione del modello) per formare gli alberi. Questo

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	cubicle	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.

Figura 3.29: Schema di labeling per le tipologie di scuola (a) ed ufficio (b).

algoritmo di classificazione è caratterizzato da efficienza computazionale e fornisce accuratezza e robustezza in diversi task di classificazione.

Lo pseudo-codice dell'algoritmo di extra-trees è tratto da [32] ed è mostrato in Figura 3.30. L'algoritmo è in grado di generare un insieme di  $M$  alberi  $T = \{t_1, \dots, t_M\}$  a partire da un training set  $S$ , costituito da un'insieme di *candidate attributes*  $a_i$ , ovvero da variabili che indicano osservazioni disponibili corrispondenti a feature. Il training set  $S$  è organizzato in una tabella in cui ogni riga corrisponde a una stanza  $r$  ed è costituita da un vettore di feature  $V_r$  e dalla relativa label. Per la costruzione di ciascun albero  $t$  si stabilisce in prima istanza se:

1. La cardinalità di  $S$  sia inferiore ad un numero  $n_{\min}$ .

Figura 3.30: Pseudo-codice dell'algoritmo di extra-trees, tratto da [32].

---

**Build\_an\_extra\_tree\_ensemble( $S$ ).**

*Input:* a training set  $S$ .

*Output:* a tree ensemble  $\mathcal{T} = \{t_1, \dots, t_M\}$ .

- For  $i=1$  to  $M$ 
  - Generate a tree:  $t_i = \text{Build\_an\_extra\_tree}(S)$ ;
- Return  $\mathcal{T}$ .

**Build\_an\_extra\_tree( $S$ ).**

*Input:* a training set  $S$ .

*Output:* a tree  $t$ .

- Return a leaf labeled by class frequencies (or average output, in regression) in  $S$  if
  - (i)  $|S| < n_{min}$ , or
  - (ii) all candidate attributes are constant in  $S$ , or
  - (iii) the output variable is constant in  $S$
- Otherwise:
  1. Select randomly  $K$  attributes,  $\{a_1, \dots, a_K\}$ , without replacement, among all (non constant in  $S$ ) candidate attributes;
  2. Generate  $K$  splits  $\{s_1, \dots, s_K\}$ , where  $s_i = \text{Pick\_a\_random\_split}(S, a_i), \forall i = 1, \dots, K$ ;
  3. Select a split  $s_*$  such that  $\text{Score}(s_*, S) = \max_{i=1, \dots, K} \text{Score}(s_i, S)$ ;
  4. Split  $S$  into subsets  $S_l$  and  $S_r$  according to the test  $s_*$ ;
  5. Build  $t_l = \text{Build\_an\_extra\_tree}(S_l)$  and  $t_r = \text{Build\_an\_extra\_tree}(S_r)$  from these subsets;
  6. Create a node with the split  $s_*$ , attach  $t_l$  and  $t_r$  as left and right subtrees of this node and return the resulting tree  $t$ .

**Pick\_a\_random\_split( $S, a$ )**

*Input:* a training set  $S$  and an attribute  $a$ .

*Output:* a split.

- If the attribute  $a$  is numerical:
    - Compute the maximal and minimal value of  $a$  in  $S$ , denoted respectively by  $a_{\min}^S$  and  $a_{\max}^S$ ;
    - Draw a cut-point  $a_c$  uniformly in  $[a_{\min}^S, a_{\max}^S]$ ;
    - Return the split  $[a < a_c]$ .
  - If the attribute  $a$  is categorical (denote by  $\mathcal{A}$  its set of possible values):
    - Compute  $\mathcal{A}_S$  the subset of  $\mathcal{A}$  of values of  $a$  that appear in  $S$ ;
    - Randomly draw a proper non empty subset  $\mathcal{A}_1$  of  $\mathcal{A}_S$  and a subset  $\mathcal{A}_2$  of  $\mathcal{A} \setminus \mathcal{A}_S$ ;
    - Return the split  $[a \in \mathcal{A}_1 \cup \mathcal{A}_2]$ .
- 

2. Tutti gli attributi (feature)  $a_i \in S$  siano costanti in  $S$ .

3. La variabile di output (label) sia costante in  $S$ .

Se almeno una delle tre condizioni è vera, allora viene ritornato un nodo foglia la cui label è decisa tramite majority vote sulle label in  $S$ .

In caso contrario vengono selezionati casualmente  $K$  attributi  $\{a_1, \dots, a_k\} \subset S$ , in base ai quali vengono generate  $K$  divisioni (split)  $\{s_1, \dots, s_k\}$ . Il procedimento tramite il quale viene ottenuta ciascuna divisione  $s_i$  dipende dalla natura dell'attributo  $a_i$ :

- Se  $a_i$  è un attributo numerico, viene definito un cut-point casuale  $a_c$  in  $[a_{\min}^S, a_{\max}^S]$ , dove  $a_{\min}^S$  e  $a_{\max}^S$  denotano rispettivamente il valore minimo e massimo di  $a_i$  in  $S$ .
- Se  $a_i$  è un attributo categorico, viene denotato come  $A$  il suo insieme di possibili valori e viene calcolato  $A_S \subseteq A$  come insieme dei valori di  $a_i$  che compaiono in  $S$ . Viene poi stabilito un sottoinsieme  $A_1$  di  $A_S$  ed un sottoinsieme  $A_2$  come  $A \setminus A_1$ .

Tra le  $K$  divisioni  $\{s_1, \dots, s_k\}$  così ottenute viene scelta la divisione  $s_*$  che massimizza una funzione *Score*, presentata in [115] e così definita:

$$Score(s, S) = \frac{2I_c^s(S)}{H_s(S) + H_c(S)} \quad (3.7)$$

dove  $H_c(S)$  è l'entropia della classificazione in  $S$ ,  $H_s(S)$  è l'entropia della divisione e  $I_c^s(S)$  è la mutua dipendenza tra il risultato della divisione e la classificazione.  $H_c(S)$ ,  $H_s(S)$  e  $I_c^s(S)$  sono anch'esse presentate e descritte in [115].

Una volta identificata la divisione  $s_*$ , in base ad essa l'insieme  $S$  viene suddiviso in  $S_l$  e  $S_r$ . A partire da questi due sottoinsiemi vengono costruiti due sottoalberi  $t_l$  e  $t_r$ , seguendo la medesima procedura. Viene poi creato un nodo in corrispondenza della divisione  $s_*$ , e a tale nodo vengono collegati  $t_l$  e  $t_r$  rispettivamente come sottoalbero di sinistra e destra. Terminata questa operazione si considera conclusa la costruzione di un albero  $t$ .

Il fondamento logico alla base del metodo di extra-trees è che la randomizzazione esplicita dei cut-point e degli attributi sia in grado di incrementare l'accuratezza dei risultati in modo più efficace rispetto a schemi di randomizzazione debole usati in altri metodi.

Dal punto di vista computazionale, la complessità della procedura è nell'ordine di  $N \log N$ , dove  $N$  indica la dimensione del training set  $S$ .

Extra-trees è un metodo di classificazione *supervisionata*. Esso è in grado di indurre una funzione a partire da dati di training etichettati, e tale funzione può essere utilizzata per classificare nuovi esempi con una label. Si rendono dunque necessari dei dati di training tramite i quali poter addestrare il classificatore.

### 3.5.5 Dati di training

In questo lavoro di tesi si è scelto di estrarre i dati di training da un dataset di file in formato *XML* (eXtensible Markup Language) che descrivono pla-

nimetrie di edifici reali di tipologia scuola, già utilizzati nei lavori presentati in [61], [59] e [60].

Ciascun file XML del dataset consiste in una descrizione testuale, suddivisa in campi, di tutti gli elementi che compongono l'ambiente. Per ogni stanza vengono riportate le coordinate del relativo bounding-polygon e le coordinate di muri e porte appartenenti a quella stanza. Un esempio di file XML di questo tipo è mostrato in Figura 3.31

```
<?xml version="1.0" encoding="UTF-8"?>
<building id="3f0b8784-33ee-4082-9e96-61faa44ba0e3">
  <scale>
    <represented_distance>
      <value></value>
      <um>pixel</um>
    </represented_distance>
    <real_distance>
      <value>90</value>
      <um>cm</um>
    </real_distance>
  </scale>
  <building_type>
    <main_type>SCHOOL</main_type>
  </building_type>
  <floor>
    <spaces>
      <space id="bfeb6869-3c8a-471d-8fee-6c4e51121457">
        <labels>
          <type>R</type>
          <label>CLASSROOM</label>
        </labels>
        <centroid>
          <point x="450" y="779"/>
        </centroid>
        <bounding_box>
          <maxx>
            <point x="516" y="769"/>
          </maxx>
          <maxy>
            <point x="450" y="844"/>
          </maxy>
          <minx>
            <point x="375" y="777"/>
          </minx>
          <miny>
            <point x="440" y="707"/>
          </miny>
        </bounding_box>
        <bounding_polygon>
          <point x="461" y="717"/>
          <point x="516" y="769"/>
          <point x="450" y="844"/>
          <point x="375" y="777"/>
          <point x="427" y="720"/>
          <point x="436" y="713"/>
          <point x="440" y="707"/>
          <point x="455" y="722"/>
          <point x="461" y="717"/>
        </bounding_polygon>
      </space>
    </spaces>
  </floor>
</building>
```

Figura 3.31: Esempio di file XML.

Inoltre questi file sono arricchiti associando a ciascuna stanza  $r$  una label  $l_r$  ed un vettore di feature  $V_r$ , calcolato sfruttando le informazioni riportate nel file.

Ciascun dato di training utilizzato in questo lavoro di tesi allo scopo di addestrare il classificatore extra-trees consiste in una stanza ground truth  $r$ , espressa come una coppia  $(V_r, l_r)$ .

Definiti i dati di training, può essere realizzata l'operazione di addestra-

mento del classificatore, per mezzo della quale viene indotta una funzione che può essere utilizzata per predire le etichette di altri esempi. Nel nostro caso gli oggetti da classificare corrispondono alle stanze segmentate, individuate nella precedente fase di costruzione del layout delle stanze.

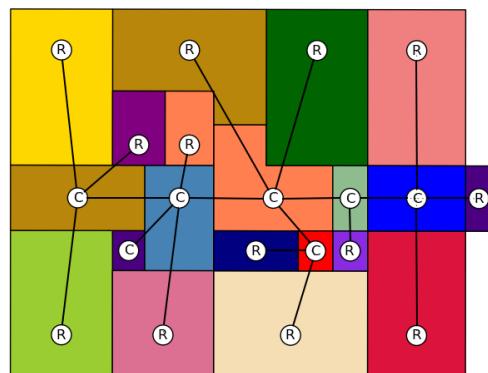
### 3.5.6 Classificazione

Una volta addestrato il classificatore, per ciascuna stanza segmentata  $r$  viene calcolata la corrispondente lista di feature  $V_r$ , e quest'ultima viene data in input al classificatore. Ricevendo in input l'insieme di feature relative alla stanza, l'algoritmo può predirne la label, fornita come output.

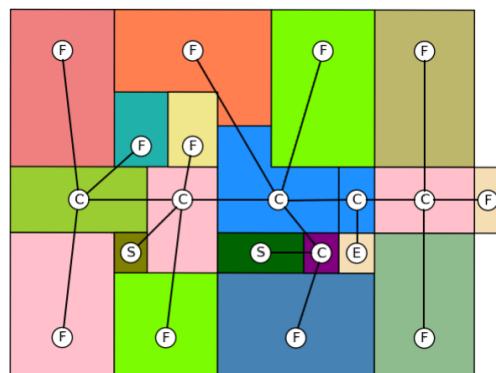
Allo scopo di valutare l'impatto di diversi insiemi di label sulla classificazione, si è deciso di eseguire due tipi di classificazioni semantiche:

- **RC:** per la classificazione viene considerato il livello superiore del labeling schema mostrato in Figura 3.28. La label predetta può essere  $R$  o  $C$ , etichettando la stanza come *room* o *corridor*, come mostrato in Figura 3.32 (a). Nella fase di training le label  $E$  (*entrance*) del dataset ground truth non vengono considerate come  $C$ , bensì come  $R$ , contravvenendo a quanto espresso nello schema gerarchico della Figura 3.28. Questo poichè gli spazi etichettati come *entrance*, ossia ingressi, le scale e gli ascensori, hanno lo scopo di connettere due stanze appartenenti a piani diversi, ma possiedono feature geometriche più simili a quelle di una *room* piuttosto che a quelle di un *corridor*. La scelta di interpretarle come  $R$  ha dunque lo scopo di ottimizzare la classificazione.
- **FCES:** viene considerato il livello inferiore dello schema di labeling, predicendo per ogni stanza un'etichetta semantica tra  $F$  (*functional room*),  $C$  (*corridor*),  $E$  (*entrance*),  $S$  (*service room*), come mostrato in Figura 3.32 (b).

Al fine di testare una situazione in cui siano considerate anche informazioni *globali*, oltre a quelle *locali* incluse in  $V_r$ , entrambi i tipi di classificazioni sono stati sperimentati considerando o meno le due feature relative alla centralità (*closeness* e *betweenness*).



(a)



(b)

*Figura 3.32: (a) Classificazione semantica RC. Le aree segmentate sono etichettate come ROOM (R) o CORRIDOR (C); (b) Classificazione semantica FCES. Le aree segmentate sono etichettate come FUNCTIONAL ROOM (F), CORRIDOR (C), ENTRANCE (E) o SERVICE ROOM (S).*



## Capitolo 4

# Architettura del sistema

In questo capitolo viene mostrata l'architettura del sistema sviluppato in questa tesi. Sono presentati e descritti i vari moduli che lo compongono, i quali permettono di ottenere una mappa multilivello che rappresenti un ambiente indoor, a partire da una mappa metrica.

La trattazione di questo capitolo tocca tutti gli aspetti tecnologici e implementativi legati al lavoro svolto in questa tesi. L'architettura del sistema viene descritta ponendo particolare attenzione al funzionamento dei moduli relativi ai diversi livelli di mapping e al modo in cui questi interagiscono tra loro. Vengono anche descritti alcuni aspetti significativi interni ai vari componenti.

L'intero modello è descritto dal punto di vista teorico nel Capitolo 3; ognuna delle sue fasi, dalla generazione della mappa metrica alla classificazione semantica, è stata implementata in `python`. Questo è un linguaggio di programmazione *object oriented*; permette progettazione e realizzazione di codice sotto forma di moduli e classi che interagiscono tra di loro.

Il sistema di rappresentazione multilivello oggetto di questo lavoro di tesi è stato progettato come un insieme di moduli distinti. I moduli sono eseguiti sequenzialmente, in quanto ognuno di essi sfrutta un livello di conoscenza estratto dal modulo che lo precede, con lo scopo di ottenere nuove informazioni. La suddivisione dei moduli rispecchia quella delle fasi del metodo proposto e presentato nel Capitolo 3:

1. **MappaMetrica:** ha il compito di fornire la mappa metrica dai dati raccolti dai sensori.
2. **Preprocessing:** esegue un preprocessing della mappa metrica nel caso essa consista in una planimetria contenente simboli non corrispon-

denti a muri.

3. **LayoutStanze**: genera il layout delle stanze segmentando la mappa metrica.
4. **GrafoTopologico**: crea il grafo topologico dell'ambiente, in cui ogni nodo corrisponde a una stanza e ogni arco a una connessione.
5. **MappaSemantica**: classifica semanticamente le stanze generando una mappa semantica che associa ad ogni stanza una label.

## 4.1 Il modulo MappaMetrica

In questa sezione viene discusso il modulo **MappaMetrica** che ha come obiettivo quello di fornire la mappa metrica ai layer di rappresentazione successivi. Le operazioni svolte da questo modulo si differenziano in base a due casi: mappa metrica nota e non nota a priori.

### 4.1.1 Mappa metrica non nota a priori

Nel caso in cui la mappa metrica non sia nota a priori, il modulo implementa un'esplorazione di un ambiente indoor da parte di un robot mobile attraverso il quale acquisire dati sensoriali ed elaborarli al fine di generare una mappa metrica. L'esplorazione viene fatta mediante l'uso di un simulatore, *Stage* [113]. In Stage un ambiente bidimensionale viene simulato a partire da un'immagine rappresentante lo spazio libero (spazio privo di ostacoli in cui il robot può muoversi liberamente) e gli ostacoli (spazio occupato da oggetti o pareti). Un robot mobile può essere istanziato all'interno di tale ambiente, e può essere dotato di sensori anch'essi simulati (nel nostro caso uno scanner laser) tramite cui raccogliere dati allo scopo di elaborarli e generare una mappa (nel nostro caso a griglia).

In questa tesi è stata utilizzata l'implementazione di Stage integrata nel framework *ROS (Robot Operating System)*, la cui repository è in [98]. ROS è un framework software per lo sviluppo e la programmazione per robot. Fornisce i servizi standard di un sistema operativo, come: astrazione dell'hardware, controllo dei dispositivi tramite driver, comunicazione tra processi, gestione delle applicazioni (package) e altre funzioni di uso comune. Un insieme di processi all'interno di ROS può essere rappresentato in un grafo come dei nodi che possono ricevere e inviare i messaggi provenienti da e verso altri nodi.

La comunicazione tra i nodi ROS avviene attraverso il paradigma *publish-subscribe*. I *topic* sono i canali di comunicazione utilizzati dai nodi per scambiare messaggi. Un nodo interessato alla ricezione di messaggi riguardanti un particolare tipo di dato, si iscrive (*subscribe*) al topic relativo allo scambio di tali messaggi. Un nodo che genera dati, invece pubblica (*publish*) messaggi sui topic corrispondenti.

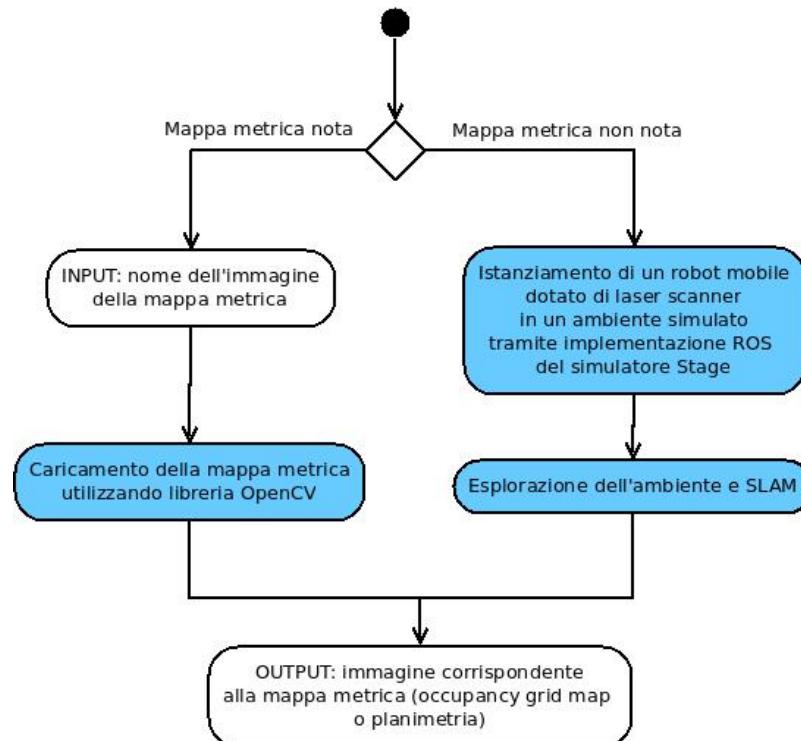
I principali nodi ROS utilizzati in questo lavoro di tesi sono:

- **Stage:** avvia il simulatore Stage con un dato ambiente, ricevuto in input come file in formato `world`. Questo file contiene informazioni sul robot mobile (sensori e posizione di istanziamento) e sull’ambiente simulato (dimensioni e nome dell’immagine da cui viene generato). In questo lavoro di tesi si è scelto di dotare il robot di uno scanner laser, simulando il sensore laser *Hokuyo*. Il nodo ROS `Stage` è incluso nel package `stage_ros` [99].
- **Navigator:** implementa le funzionalità di esplorazione autonoma e di navigazione verso un goal. Prende come dato di input un file di tipo `yaml`, un formato per la serializzazione di dati che sfrutta concetti di altri linguaggi come `C`, `python` e `XML`. Questo file contiene diversi parametri di navigazione, tra cui `exploration_strategy`, che indica la strategia di esplorazione da implementare. In questo lavoro di tesi si è scelto di utilizzare l’algoritmo di esplorazione *Nearest Frontier*, già discusso in Sezione 3.2.2 ed invocabile attribuendo al parametro `exploration_strategy` il valore `NearestFrontierPlanner` [73]. Tale algoritmo permette al robot di muoversi autonomamente evitando collisioni con ostacoli e dirigendosi iterativamente verso la frontiera più vicina. Il nodo ROS `Navigator` è incluso nel package `nav2d_navigator` [72].
- **Slam\_gmapping:** risolve il problema dello *SLAM (Simultaneous Localization and Mapping)* [107] implementando la tecnica *GMapping* [35], presentata in Sezione 3.2.2. Vengono integrate in modo incrementale le scansioni laser acquisite, allo scopo di costruire una occupancy grid map dell’ambiente e al contempo stabilire la posizione del robot all’interno di essa. Il nodo ROS `Slam_gmapping` è incluso nel package `gmapping` [33].

#### 4.1.2 Mappa metrica nota a priori

Nel caso in cui la mappa metrica sia nota a priori, possono essere identificate almeno due opzioni: la mappa nota al robot deriva da un'esplorazione precedente del robot (Sezione 3.2.2), oppure è fornita come una planimetria, ovvero un disegno bidimensionale in scala che mostra una vista dall'alto dell'ambiente, similmente all'utilizzo di una cartina da parte di una persona. L'operazione svolta dal modulo **MappaMetrica** in questo caso consiste in un semplice caricamento dell'immagine della mappa. Il caricamento è ottenuto invocando la funzione `imread` implementata all'interno della libreria **OpenCV** (Open Source Computer Vision Library) [81], una libreria software multipiattaforma di computer vision.

In Figura 4.1 è mostrato un diagramma delle attività relativo al modulo **MappaMetrica**. L'output del modulo consiste nell'immagine della mappa metrica, sotto forma di matrice le cui celle contengono i 3 valori RGB inclusi nell'intervallo [0,255]. L'immagine è offerta come input ai moduli successivi.



*Figura 4.1: Diagramma di attività del modulo **MappaMetrica**.*

## 4.2 Il modulo Preprocessing

Il modulo **Preprocessing** viene invocato nel caso in cui la mappa metrica proveniente dal modulo **MappaMetrica** consista in una planimetria avente al suo interno simboli non corrispondenti a muri (estintori, idranti, informazione testuale, ecc.) e quindi provenga da un'immagine. L'obiettivo del modulo è quello di estrarre tali oggetti, salvandone la posizione e producendo una mappa metrica che contenga esclusivamente una rappresentazione delle pareti.

### 4.2.1 Individuazione dei template

Al fine di individuare gli oggetti non corrispondenti a muri, in questo lavoro di tesi si è scelto di impiegare l'algoritmo di *template matching* [90], una tecnica di elaborazione digitale delle immagini finalizzata all'identificazione di piccole porzioni di un'immagine che corrispondano ad una immagine-modello (template).

Il modulo utilizza l'implementazione di questo algoritmo contenuta nella libreria **OpenCV**, invocando la funzione `matchTemplate`. La funzione prende come input l'immagine sorgente `metricMap` ed un template `T` appartenente alla lista `templateList` (inclusa nella directory corrente), da ricercare all'interno dell'immagine. La funzione è invocata iterativamente, ricevendo ad ogni iterazione come dato di input `T` un diverso template della lista `templateList`, sotto forma di matrice le cui celle contengono i 3 valori RGB.

L'output della funzione è una matrice `R` in cui ogni posizione  $(x, y)$  contiene un valore di corrispondenza  $c$  compreso tra 0 e 1. Ciascun valore  $c \geq \text{matchingThreshold}$  identifica l'individuazione del template `T` nella posizione  $(x, y)$  all'interno dell'immagine `metricMap`. Ciascun template individuato è caratterizzato da un rettangolo di dimensioni pari a quelle del template, con vertice superiore di sinistra posto nelle coordinate  $(x, y)$ .

I vertici dei rettangoli corrispondenti ai template individuati vengono salvati in due differenti liste in base alla natura del template:

- `doorsVertices`: lista dei template delle porte.
- `templateVertices`: lista dei template corrispondenti a tutti gli altri oggetti al di fuori delle porte (estintori, idranti, ecc.).

### 4.2.2 Eliminazione dei template

L'eliminazione degli oggetti inclusi in `templateVertices` e delle porte incluse in `doorsVertices` sono svolte in serie ed avvengono con due modalità

diverse.

Gli oggetti di `templatesVertices` vengono eliminati dalla planimetria sostituendo i pixel corrispondenti all'area coperta con pixel del colore dello sfondo, nel nostro caso bianco. La funzione che svolge questa operazione è `eliminaTemplates`, che prende come dati di input l'immagine `metricMap` e la lista di vertici `templatesVertices` che individuano i template da eliminare. L'output della funzione consiste nell'immagine `metricMap` in cui gli oggetti della lista `templatesVertices` non sono più visibili.

Il processo di eliminazione delle porte è eseguito dal modulo in maniera differente. Le porte vengono infatti sostituite con altri template, istanziati nella medesima posizione delle porte e raffiguranti una porzione di muro spessa quanto la porta in questione. L'operazione è eseguita invocando la funzione `sostituiscePorte` che prende come dati di input l'immagine `metricMap` restituita dalla funzione `eliminaTemplates`, la lista di vertici `doorsVertices` che individuano le porte da eliminare e la lista `wallTemplates` (generata manualmente) costituita dai template di muri da inserire nell'immagine al posto delle porte. Viene restituita come output della funzione l'immagine `metricMap` modificata in modo che le porte risultino come se fossero ‘chiuse’, come mostrato in Figura 3.7 (Sezione 3.3.2). I simboli delle porte individuati tramite template matching, indicati da un rettangolo blu (a), sono sostituiti da template di muri (b). Il fine di questa operazione è quello di ottimizzare la costruzione del layout delle stanze eseguita dal modulo successivo, come osservabile dai risultati mostrati in Figura 3.8. Eliminando le porte in modo analogo agli altri simboli (a), ovvero colorando del colore dello sfondo (bianco) tutti i pixel che vi appartengono, il layout delle stanze può presentare imprecisioni (b). La stanza in alto a destra (evidenziata da un cerchio rosso) non è riconosciuta come una regione di spazio separata dalle altre. La chiusura delle porte ottenuta tramite sostituzione con template di muri (c) genera invece un layout delle stanze corretto (d), in cui la stanza in alto a destra (cerchio rosso) risulta disgiunta dalle altre.

#### 4.2.3 Individuazione dell'informazione testuale

A questo punto il modulo può eseguire l'operazione di individuazione e salvataggio dell'informazione testuale nella mappa metrica. A questo scopo viene utilizzato un generico sistema di *riconoscimento ottico dei caratteri (OCR)*. Un OCR è un sistema dedicato alla conversione di un'immagine contenente

testo in testo digitale, modificabile con un normale editor. In questo lavoro di tesi è stato utilizzato il modulo python dell'OCR *Tesseract*, ovvero *pytesseract* [85]. Questo prende come dato di input l'immagine `metricMap` restituita da `sostituisciPorte` e riconosce e salva in un file `text.txt` nella directory corrente il testo presente nell'immagine.

#### 4.2.4 Eliminazione dell'informazione testuale

Dopo aver salvato l'informazione testuale il modulo procede con la sua eliminazione. Vengono ricavati i contorni esterni degli elementi presenti in `metricMap`, invocando la funzione `findContours` della libreria `OpenCV`. I dati di input della funzione sono l'immagine `metricMap` e la modalità `retrExternal` che indica alla funzione la ricerca di contorni esterni. L'output `contours` è una lista i cui elementi consistono a loro volta in liste, ciascuna delle quali è costituita dai punti che compongono il contorno esterno di un oggetto.

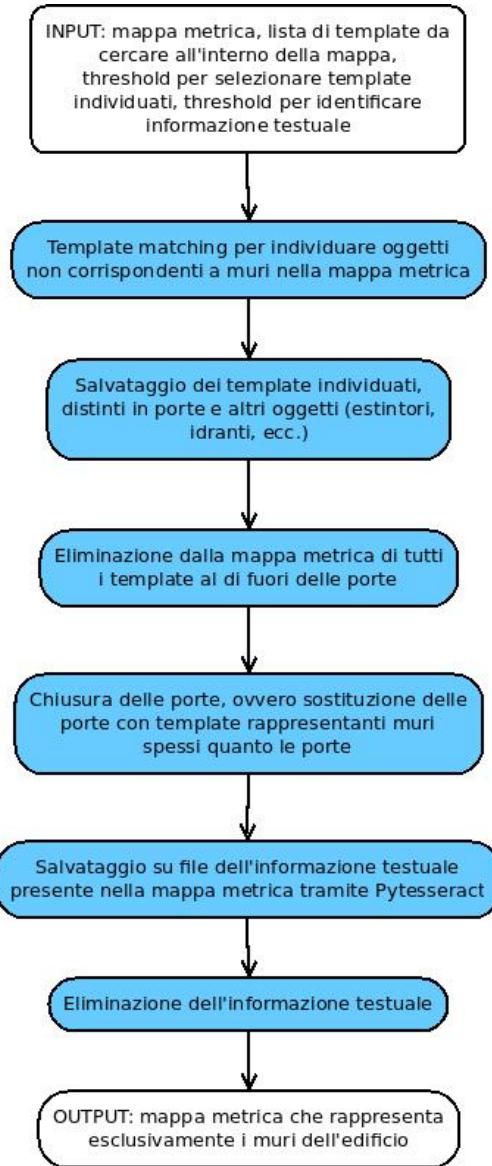
I contorni esterni individuati identificano due tipi di oggetti: le parole che costituiscono l'informazione testuale e i muri. La distinzione tra i due casi è eseguita analizzando il valore dell'area delimitata dai contorni. Ci si basa sull'assunzione che gli oggetti relativamente piccoli corrispondano alle parole da eliminare, mentre i muri siano caratterizzati da un'area di dimensioni maggiori. Il modulo invoca una funzione `eliminaTesto` che prende come dati di input `metricMap`, `contours` e una soglia `thresholdContours`. I contorni esterni che compongono la lista `contours` la cui area è inferiore a `thresholdContours` vengono identificati come informazione testuale e vengono eliminati in modo analogo ai template, ovvero colorando del colore dello sfondo i punti che vi appartengono. L'output della funzione è l'immagine `metricMap` che a questo punto rappresenta esclusivamente i muri dell'edificio.

Completata anche l'eliminazione del testo, il preprocessing può ritenersi concluso. In Figura 4.2 è mostrato il diagramma di attività del modulo, in cui vengono indicate le operazioni svolte.

#### 4.2.5 Dati di input/output

I dati di input iniziali sono:

- `metricMap`: la mappa metrica proveniente dal modulo precedente.



*Figura 4.2: Diagramma di attività del modulo Preprocessing.*

- **templateList:** lista di template standard tipicamente presenti all'interno delle planimetrie, tratta da [75]. La lista include diversi template per ciascuna tipologia di oggetto (estintori, porte, idranti, ecc.).
- **matchingThreshold:** soglia che decreta l'individuazione o meno di un template all'interno dell'immagine. Sperimentalmente è stata settata a 0.8.

- `thresholdContours`: soglia in base alla quale viene distinta l'informazione testuale dai muri, posta a  $3 \text{ m}^2$ .

L'output del modulo consiste in:

- `metricMap`: l'immagine della mappa metrica in cui sono rappresentati esclusivamente i muri.
- `templateVertices`: lista di vertici che individuano i template estratti, al di fuori delle porte.
- `doorsVertices`: lista di vertici che individuano le porte estratte.

## 4.3 Il modulo LayoutStanze

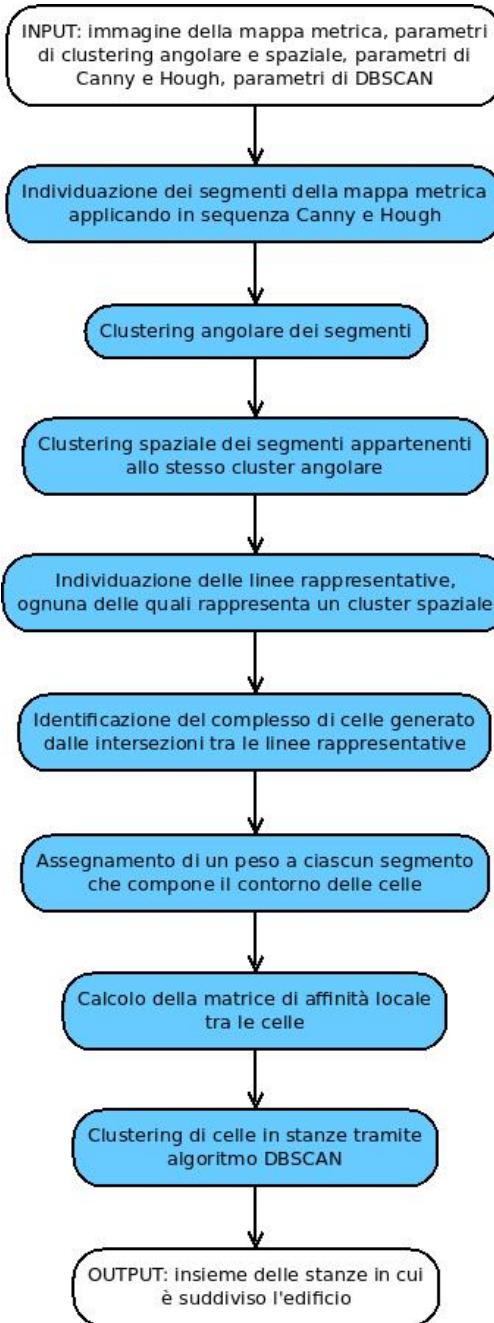
In questa sezione viene descritto il modulo `LayoutStanze` che ha come obiettivo principale quello di individuare la suddivisione dell'edificio in stanze separate l'una dall'altra. Il modulo è descritto dal punto di vista teorico nella Sezione 3.3. Il diagramma di attività in Figura 4.3 mostra il flusso di esecuzione della sequenza di operazioni svolte. Durante l'esecuzione i dati vengono riorganizzati in altre strutture dati in modo da favorirne la manipolazione.

### 4.3.1 Identificazione dei segmenti corrispondenti ai muri

Il primo passo del modulo è identificare i segmenti corrispondenti ai muri rappresentati nella mappa metrica. A questo scopo vengono applicati in sequenza *Canny edge detection* e *Hough line transform*.

L'algoritmo di Canny Edge Detection è un operatore per il riconoscimento dei contorni in un'immagine. In questo lavoro di tesi è stata utilizzata l'implementazione di tale algoritmo offerta dalla libreria OpenCV, invocando la funzione `Canny`. Questa prende come dati di input l'immagine `metricMap` proveniente dal modulo `Preprocessing`, una soglia minima `minVal` e una soglia massima `maxVal`. La funzione associa a ciascun punto dell'immagine `metricMap` un valore che ne indica la probabilità di appartenere ad un contorno. Tali valori vengono confrontati con le due soglie di input. Se il valore è:

- inferiore alla soglia `minVal`, il punto è scartato;
- superiore alla soglia `maxVal`, il punto è accettato come parte di un contorno;



*Figura 4.3: Diagramma di attività del modulo LayoutStanze.*

- compreso fra le due soglie, il punto è accettato solamente se contiguo ad un punto già precedentemente accettato.

L'output consiste in un'immagine `cannyEdges`, delle stesse dimensioni del-

l’immagine di input, in cui sono rappresentati i contorni individuati.

Il modulo a questo punto esegue l’algoritmo di Hough Line Transform, una tecnica di estrazione di feature il cui scopo è identificare segmenti in un’immagine, caratterizzati dalle coordinate degli estremi. Viene utilizzata l’implementazione presente nella libreria OpenCV, invocando la funzione `HoughLinesP`. I suoi dati di input sono l’output `cannyEdges` della funzione `Canny` e i parametri `rho`, `theta`, `thresholdHough` e `minLineLength`.

La funzione cerca all’interno dell’immagine `cannyEdges` delle rette nella forma  $\rho = x \cos(\theta) + y \sin(\theta)$  che rappresentino i contorni presenti nell’immagine. Viene costruito un array bidimensionale che rappresenta lo spazio di  $\rho$  e  $\theta$ , la cui risoluzione (step) è data rispettivamente da `rho` e `theta`. Il range di  $\rho$  va da 0 alla massima distanza possibile, equivalente alla lunghezza della diagonale dell’immagine, mentre il range di `theta` va da 0 a 180. Per ogni combinazione  $(\rho_i, \theta_i)$  l’algoritmo individua il numero di celle dell’immagine che appartengono alla retta descritta da tale coppia di valori. Se il numero di riscontri così individuati è superiore a `thresholdHough`, viene determinata la presenza in `cannyEdges` della retta  $\rho_i = x \cos(\theta_i) + y \sin(\theta_i)$ . Confrontando l’immagine di input con le rette individuate la funzione è in grado di identificare dei segmenti caratterizzati dalle coordinate degli estremi. Sono scartati i segmenti la cui lunghezza è inferiore a `minLineLength`.

L’output della funzione consiste in una lista `walls` di segmenti corrispondenti ai muri rappresentati nella mappa metrica. Ciascun segmento è un oggetto appartenente alla classe `Segmento`, i cui attributi sono:

- `punto1`: array contenente le coordinate del primo estremo,
- `punto2`: array contenente le coordinate del secondo estremo,
- `clusterAngolare`: il cluster angolare a cui appartiene il segmento,
- `clusterSpaziale`: il cluster spaziale a cui appartiene il segmento,
- `peso`: valore tra 0 e 1 che ne indica la probabilità del segmento di corrispondere ad un muro realmente presente nell’ambiente.

Giunti a questo punto, per ogni segmento della lista `walls` vengono settati esclusivamente gli attributi relativi alle coordinate dei due estremi. Gli altri attributi sono impostati successivamente.

### 4.3.2 Clustering dei segmenti

Ottenuti i segmenti, il modulo procede con il loro raggruppamento in cluster. Il clustering dei segmenti è suddiviso in due fasi: clustering angolare e

clustering spaziale.

Il clustering angolare viene eseguito basandosi sull'orientamento dei segmenti. A tale scopo il modulo invoca la funzione `meanShift`, che prende come dati di input la lista `walls` e i due parametri `h` e `minOffset`. La funzione in prima istanza calcola l'orientamento di ciascun segmento, ovvero l'angolo tra la retta su cui giace il segmento e l'asse  $x$  del sistema di riferimento. Viene poi implementato iterativamente l'algoritmo di clustering *mean-shift* [15] sugli orientamenti trovati. Allo step 0 i *cluster center* coincidono con tali orientamenti, mentre ad ogni iterazione successiva vengono calcolati i nuovi cluster center tramite le equazioni (3.2) e (3.3), presentate in Sezione 3.3.4, in cui viene utilizzata la bandwidth `h`. Il processo si conclude quando la massima differenza tra nuovi e vecchi cluster center è inferiore a `minOffset`.

Basandosi sul cluster angolare individuato, viene settato l'attributo `clusterAngolare` di ciascun segmento appartenente lista `walls`, completando così la fase di clustering angolare.

A questo punto i segmenti appartenenti ad uno stesso cluster angolare vengono ulteriormente raggruppati in base alla loro prossimità spaziale. Il clustering spaziale viene implementato dalla funzione `spatialClustering` che prende come dati di input la lista `walls` (aggiornata dopo il clustering angolare) e il parametro `minLateralSeparation`. La nozione che regola questa operazione è quella di *distanza laterale* tra due segmenti, definita come distanza tra i loro punti medi proiettati sulla retta perpendicolare al cluster angolare a cui entrambi i segmenti appartengono. La funzione associa allo stesso cluster spaziale segmenti del medesimo cluster angolare la cui distanza laterale  $d$  è inferiore a `minLateralSeparation`. Viene così settato l'attributo `clusterSpaziale` di ogni segmento di `walls`.

#### 4.3.3 Rette rappresentative e celle

Completato il clustering, il modulo procede con la creazione delle rette rappresentative di tali cluster, invocando la funzione `creaExtendedLines` il cui dato di input è la lista di segmenti `walls` (aggiornati dopo il clustering angolare e spaziale). L'output della funzione è una lista di rette `extendedLines` ognuna rappresentante un cluster spaziale.

Le intersezioni tra le `extendedLines` generano un insieme di celle (Sezione 3.3.5). Allo scopo di individuare tale insieme, il modulo invoca la

funzione `creaCelle`, che prende come dato di input la lista `extendedLines` e restituisce una lista `celle`. Ognuna delle celle appartenenti a tale lista consiste in un oggetto di classe `Cella`, che ha come attributo una lista `bordo` di segmenti che ne compongono il perimetro. Il cluster spaziale di ciascun segmento in `bordo` coincide con quello associato alla retta rappresentativa di cui fa parte.

#### 4.3.4 Calcolo del peso dei contorni delle celle

Il modulo provvede alla stima di un peso per ciascun segmento facente parte dei contorni delle celle, in base alla probabilità che esso corrisponda ad un muro realmente presente nella mappa metrica. L'operazione di calcolo e assegnamento dei pesi viene eseguita dalla funzione `setPeso`, i cui dati di input sono le liste `celle` e `walls`. Per ogni cella la funzione calcola il peso dei suoi segmenti perimetrali mediante l'equazione (3.4), presentata in Sezione 3.3.5. Il risultato è associato all'attributo `peso` del segmento in questione.

#### 4.3.5 Classificazione delle celle

A questo punto il modulo esegue la classificazione delle celle in *interne*, *esterne* e *parziali*, invocando la funzione `classificaCelle` che confronta la superficie di ciascuna cella con quella dell'ambiente indoor dell'edificio rappresentato nella mappa metrica.

Per ciascuna cella la funzione calcola un valore `delta`, corrispondente a metà della sua area. `delta` viene poi confrontato con l'area dell'intersezione tra la cella in questione e l'ambiente interno dell'edificio. Le celle la cui intersezione con l'ambiente interno è inferiore a `delta` sono classificate come esterne.

Le celle classificate come non esterne vengono analizzate in modo da operare una distinzione tra interne e parziali. Son classificate come parziali le celle  $f_i$  adiacenti ad una cella esterna  $f_j$  per mezzo di un segmento  $e_{ij}$  il cui peso  $w_{ij}$  è inferiore a `sogliaParziale`, sperimentalmente settato a 0.2. Le restanti celle sono classificate come interne.

I dati di input della funzione sono la lista `celle`, la mappa metrica `metricMap` e il parametro `sogliaParziale`. Vengono restituite le liste `celleInterne`, `celleEsterne` e `celleParziali` contenenti rispettivamente le celle classificate come interne, esterne e parziali.

#### 4.3.6 Matrice di affinità

Una volta conclusa la classificazione delle celle, il modulo stabilisce una misura di affinità globale per ogni coppia di celle della lista `celleInterne`, basandosi sul peso dei segmenti che ne costituiscono il contorno comune, come spiegato in Sezione 3.3.6. La misura di affinità viene espressa in forma matriciale invocando la funzione `creaMatriceAffinità` che prende come dato di input la lista `celleInterne`.

L'output della funzione consiste in una matrice  $M$  di dimensioni pari al numero di celle (righe) per il numero di celle (colonne), di cui ogni elemento  $M_{ij}$  può essere visto come un valore di affinità locale tra le celle  $f_i$  e  $f_j$ .

#### 4.3.7 Clustering delle celle in stanze

L'operazione successiva consiste nell'eseguire un clustering delle celle, basandosi sulla loro affinità espressa dalla matrice  $M$ . A tale scopo viene utilizzato l'algoritmo di *DBSCAN*, un algoritmo di clustering presentato in Sezione 3.3.6 e basato sul concetto di *densità*, in quanto in grado di connettere regioni di punti con densità sufficientemente alta. Ciascun punto  $x$  nel dataset che abbia un numero di  $\varepsilon$ -vicini (punti la cui distanza da  $x$  è inferiore a un valore `eps`) uguale o maggiore a `minPts` viene marcato come *core point*. Se invece un punto  $x$  possiede un numero di  $\varepsilon$ -vicini inferiore a `minPts`, ma si trova nell' $\varepsilon$ -vicinato di un core point, allora viene marcato come *border point*. Infine se un punto  $x$  non è marcato come core point né come border point allora è considerato *outlier* o rumore.

In questo lavoro di tesi è stata utilizzata l'implementazione di DBSCAN inclusa in `Scikit-learn` [92], una libreria python contenente numerosi algoritmi di classificazione e clustering.

L'insieme di punti su cui l'algoritmo di DBSCAN deve eseguire il clustering corrisponde all'insieme delle celle della lista `celleInterne`. Per ogni coppia di celle  $f_i$  e  $f_j$ , DBSCAN deve analizzare il valore  $d_{ij} = 1 - M_{ij}$  che indica la distanza tra le celle  $f_i$  e  $f_j$  in termini di affinità. La funzione `DBSCAN` che implementa l'algoritmo di clustering prende dunque come dato di input la matrice  $X = 1 - M$ , oltre ai parametri `eps` e `minPts` che guidano il processo di clustering. L'output della funzione è una lista `clustersCelle`, della stessa dimensione di `celleInterne`, contenente gli indici dei cluster. Più precisamente, l'indice del cluster a cui è stata assegnata la  $i$ -esima cella di `celleInterne` è contenuto nell' $i$ -esimo elemento di `clustersCelle`. Due celle in `celleInterne` appartengono allo stesso cluster se i corrispondenti valori in `clustersCelle` sono uguali.

Le celle di `celleInterne` corrispondenti ad uno stesso cluster sono unite in stanze dalla funzione `unisciCelle`, che prende come dati di input `celleInterne` e `clustersCelle`. L'output della funzione è una lista `stanze` composta dalle stanze in cui è suddiviso l'edificio. Ogni stanza della lista consiste in un oggetto di classe `Polygon` appartenente al package `Shapely`. Questo è un package python per manipolazione ed analisi di oggetti geometrici piani, la cui repository si trova in [94].

Ciascun elemento della lista `stanze`, ottenuto dall'unione di più celle, rappresenta una stanza, ovvero una regione di spazio che compone l'edificio. Con l'ottenimento di questa lista si considera dunque conclusa la definizione del layout delle stanze a partire dalla mappa metrica.

#### 4.3.8 Dati di input/output

I dati di input iniziali sono:

- `metricMap`: la mappa metrica proveniente dal modulo precedente. Essa rappresenta la base su cui viene costruito il layout delle stanze.
- `minVal`, `maxVal`: parametri di Canny Edge Detector che indicano soglia minima e massima utilizzate per determinare l'individuazione dei contorni all'interno della mappa metrica. Sono posti rispettivamente a 90 e 100.
- `rho`, `theta`: parametri di Hough Line Transform che indicano l'accuracy dei valori  $\rho$  e  $\theta$ , caratterizzanti una retta nella forma  $\rho = x \cos(\theta) + y \sin(\theta)$  da cercare all'interno di un'immagine. `rho` indica la risoluzione di  $\rho$  in pixel, mentre `theta` la risoluzione di  $\theta$  in radianti. Sono entrambi settati a 1.
- `thresholdHough`: parametro dell'Hough Line Transform che indica il numero minimo di riscontri relativi ad una retta per poterla considerare come appartenente ad un'immagine. Il suo valore viene posto a 20.
- `minLineLength`: parametro di Hough Line Transform che indica il numero minimo di punti che possono formare un segmento restituito. Viene settato a 7.
- `eps`, `minPts`: parametri di DBSCAN che guidano l'operazione di clustering che porta all'identificazione delle stanze. Il loro valore è posto rispettivamente a 0.85 e 1.

- **`h`, `minOffset`**: parametri di mean-shift che determinano il clustering angolare dei segmenti individuati all'interno mappa metrica. Il valore di `h` definisce una bandwidth utilizzata nella funzione kernel dell'algoritmo, mentre `minOffset` costituisce un valore di soglia che determina la conclusione dell'algoritmo. Il loro valore è posto rispettivamente a 0.023 e 0.00001.
- **`minLateralSeparation`**: indica la massima distanza laterale che due segmenti dello stesso cluster angolare devono possedere per poter essere considerati come appartenenti allo stesso cluster spaziale. Viene settato a 7.
- **`sogliaParziale`**: data una cella  $f_i$ , classificata come esterna, e una cella  $f_j$  non esterna e ad essa adiacente per mezzo di un segmento  $e_{ij}$ , se il peso  $w_{ij}$  del segmento è inferiore a `sogliaParziale` la cella  $f_j$  è classificata come parziale. Il parametro è sperimentalmente settato a 0.2.

La lista `stanze` costituisce l'output del modulo `LayoutStanze` e rappresenta la base su cui i moduli successivi possono operare, creando nuovi livelli di conoscenza.

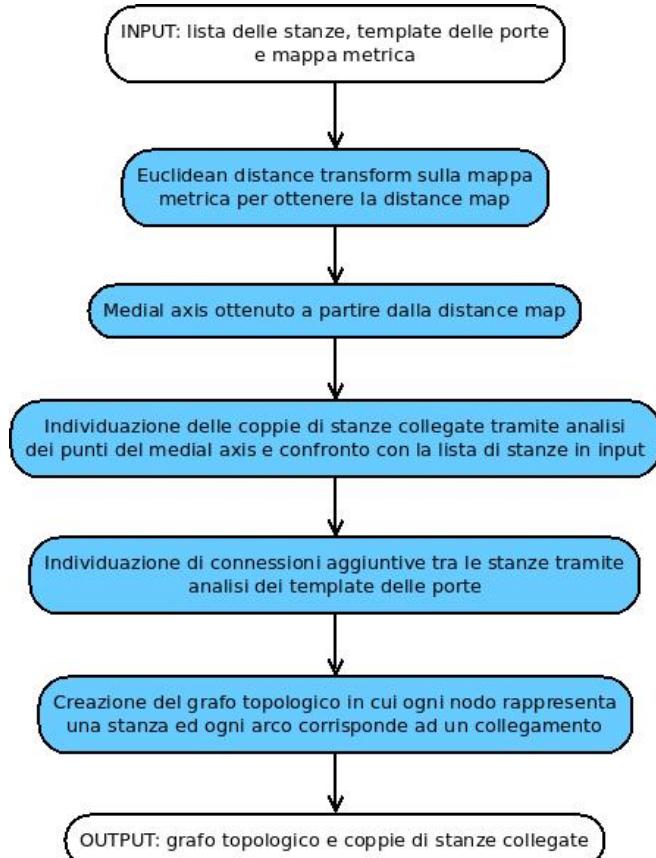
## 4.4 Il modulo GrafoTopologico

In questa sezione viene descritto il modulo `GrafoTopologico`, il cui obiettivo è generare un grafo topologico che rappresenti l'ambiente. Ogni nodo del grafo rappresenta una stanza, mentre ogni arco una connessione diretta tra stanze adiacenti. La sequenza di operazioni svolte è mostrata nel diagramma di attività in Figura 4.4.

### 4.4.1 Medial axis

La prima operazione svolta dal modulo consiste nell'identificare le connessioni tra le stanze, in modo da poterle poi rappresentare tramite gli archi del grafo. A tale scopo, la tecnica utilizzata in questo lavoro di tesi si basa sul *medial axis*, presentato in Sezione 3.4.3. Il medial axis di una superficie, definito come in [19], è l'insieme dei punti aventi più di un ostacolo alla stessa distanza minima.

Il modulo in prima istanza, a partire dalla mappa metrica, ottiene una *distance map*, ovvero una mappa rappresentata come una matrice bidimensionale in cui ad ogni cella è associata la distanza dall'ostacolo più vicino.



*Figura 4.4: Diagramma di attività del modulo GrafoTopologico.*

Questa operazione è eseguita applicando la *Euclidean distance transform* sulla mappa metrica, una trasformazione tramite la quale è possibile ricavare la distance map. Il modulo ne utilizza l'implementazione offerta da SciPy, una libreria open source di algoritmi e strumenti matematici per il linguaggio di programmazione python. Viene dunque invocata la funzione `distance_transform_edt`, che riceve come dato di input `metricMap` e restituisce una matrice bidimensionale `distanceMap` che contiene per ogni cella la distanza dall'ostacolo più vicino.

A questo punto il modulo ricava il medial axis a partire dalla distance map ottenuta. A tale scopo viene utilizzata l'implementazione offerta da `scikit-image (skimage)` [91], un package python che contiene un insieme di algoritmi dedicati all'elaborazione di immagini. Viene invocata la funzione `medial_axis` che come dato di input prende la distance map restituita da `distance_transform_edt`. L'output della funzione consiste nella lista

`medialAxis` costituita dai punti che compongono il medial axis.

Una volta ottenuto il medial axis, esso può essere analizzato per individuare le connessioni tra le stanze del layout. Questa operazione viene eseguita invocando la funzione `trovaCollegamenti`, che prende come dati di input la lista `stanze` restituita dal modulo precedente e la lista `medialAxis`. La funzione analizza i punti che compongono il medial axis ed individua quelli che confinano con due stanze. Queste sono identificate come stanze direttamente collegate tra loro, tramite una porta o un passaggio diretto privo di ostacoli. La funzione restituisce una lista `collegate` di coppie di stanze connesse tra loro.

#### 4.4.2 Analisi dei template delle porte

Nel caso in cui la mappa metrica consista in una planimetria contenente simboli delle porte, possono essere ricavate informazioni aggiuntive riguardanti la connettività degli spazi. Se presenti nella planimetria, questi simboli sono stati estratti dal modulo `Preprocessing`, salvandone i vertici dei contorni nella lista `doorsVertices`. Viene invocata la funzione `trovaCollegamentiPorte`, la quale prende come dato di input `stanze`, `doorsVertices` e `collegate`. Nel caso in cui la superficie del simbolo di una porta intersechi le superfici di due stanze, viene identificato un collegamento diretto tra le due stanze, aggiungendo la coppia di stanze a quelle già individuate applicando il medial axis. La funzione restituisce la lista `collegate` così aggiornata.

#### 4.4.3 Creazione del grafo topologico

Completata l'identificazione delle connessioni, il modulo si occupa della creazione del grafo topologico. A tale scopo viene utilizzato `NetworkX` [74], una libreria python dedicata alla creazione e manipolazione di grafi. All'oggetto `graph`, chiamato `G`, vengono aggiunti nodi e archi invocando rispettivamente le funzioni `add_node` (con dato di input `stanze`) e `add_edges_from` (a cui viene passata come input `collegate`). In questo modo ciascun nodo di `G` corrisponde ad una stanza del layout, ed ogni arco rappresenta un collegamento tra due stanze.

Il modulo provvede infine all'impostazione di un attributo `posizione` relativo a ciascun nodo di `G`. Come valore di quest'attributo vengono scelte le coordinate di un punto interno rappresentativo della stanza corrispondente al nodo in questione, ottenute invocando la funzione `representative_point` sul *Shapely Polygon* che descrive la stanza corrispondente.

#### 4.4.4 Dati di input/output

I dati di input del modulo sono:

- **stanze**: la lista di stanze restituita dal modulo `LayoutStanze`.
- **metricMap**: l'immagine della mappa metrica restituita dal modulo `Preprocessing`.
- **doorsVertices**: lista di vertici che individuano le porte estratte dal modulo `Preprocessing`.

Il graph `G` e la lista `collegate` di coppie di stanze connesse costituiscono l'output del modulo `GrafoTopologico`. Il grafo topologico ottenuto rappresenta per il modulo successivo il punto di partenza per lo sviluppo di un nuovo livello di rappresentazione.

### 4.5 Il modulo MappaSemantica

L'obiettivo di questo modulo è classificare semanticamente le stanze generando una mappa semantica che associa ad ogni stanza una label. In Figura 4.5 viene mostrato il diagramma delle attività del modulo.

#### 4.5.1 Creazione del file XML

La prima operazione svolta dal modulo consiste nella creazione di un file XML che descriva la struttura dell'ambiente. A tale scopo viene invocata la funzione `creaXML` che riceve come dati di input il nome del file `nomeXML`, le coppie di stanze connesse `collegate`, la lista di stanze `stanze` e la lista di porte `doorsVertices`. La funzione analizza le stanze e le loro connessioni e genera un file XML corrispondente a una descrizione testuale, suddivisa in campi, di tutti gli elementi che compongono l'ambiente. Per ciascuna stanza vengono riportate le coordinate del relativo bounding-polygon e le coordinate di muri e porte appartenenti a quella stanza. Un esempio di file XML di questo tipo è mostrato in Figura 3.31 (Sezione 3.5.5).

A questo punto il modulo procede con il calcolo delle feature relative ad ogni stanza dell'edificio, la cui descrizione è riportata in Sezione 3.5.2. Analizzando queste feature il classificatore sarà in grado di predire una label ed associarla alla stanza corrispondente. La lista di feature viene stimata invocando la funzione `parser` che prende come dato di input `nomeXML`, ovvero il nome del file XML generato allo step precedente. La funzione esegue

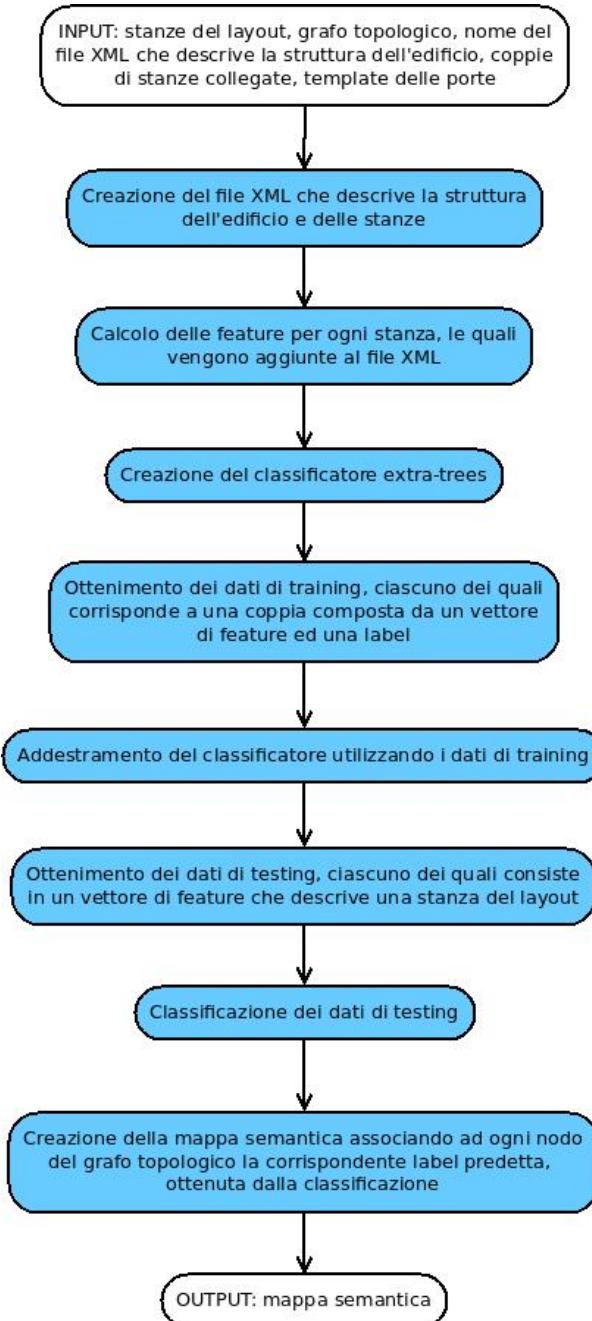


Figura 4.5: Diagramma di attività del modulo *MappaSemantica*.

un parsing di tale file e per mezzo dell’analisi dei suoi componenti calcola le feature relative alle stanze. Il file XML viene dunque arricchito di una lista di feature, posta all’interno di ogni campo *space* corrispondente a una

stanza.

#### 4.5.2 Creazione del classificatore extra-trees

Calcolate le feature, il modulo può eseguire la classificazione. In questo lavoro di tesi si è deciso di utilizzare l'algoritmo di classificazione *Extremely Randomized Trees*, introdotto in Sezione 3.5.4. Il modulo ne utilizza l'implementazione inclusa in **Scikit-learn** [92], corrispondente alla funzione `ExtraTreesClassifier` i cui dati di input sono:

- `n_estimators`: il numero di alberi nella foresta. Il suo valore è settato a 500.
- `criterion`: la funzione per misurare la qualità di uno split, posta uguale a `entropy`.
- `min_samples_split`: il numero minimo di campioni richiesti per dividere un nodo interno, il cui valore è stato posto a 1.

La funzione crea e restituisce un classificatore di tipo *extra-trees* chiamato `clf`.

#### 4.5.3 Addestramento del classificatore

Extra-trees è un metodo di classificazione *supervisionata*. Esso è in grado di indurre una funzione a partire da dati di training etichettati, e tale funzione può essere utilizzata per classificare nuovi esempi predicendo una label per ognuno di essi. Si rendono dunque necessari dei dati di training tramite i quali poter addestrare il classificatore. In questo lavoro di tesi si è scelto di estrarre i dati di training da un dataset di file in formato *XML*. Il dataset è composto da 30 file XML che descrivono altrettante planimetrie di edifici reali di tipo scuola, ottenuti per mezzo dei lavori presentati in [60], [61] e [59]. Le stanze descritte da questi file rappresentano il ground truth dell'ambiente esplorato dal robot nella fase iniziale. Gli XML del dataset indicano per ognuna di queste stanze ground truth una lista di feature ed una label.

Allo scopo di estrarre da questi file i dati di training viene invocata la funzione `getTrainingData`. I dati di input della funzione sono `labels_type` e `centrality`. Il primo indica la classe gerarchica di label che si vuole utilizzare per addestrare il classificatore. Questa può essere **RC** o **FCES**, come spiegato in Sezione 3.5.3:

- **RC**: le label si distinguono in *R* o *C*, identificando una stanza come *room* o *corridor*.

- **FCES**: viene considerato come dominio delle label *F* (*functional room*), *C* (*corridor*), *E* (*entrance*), *S* (*service room*).

Il dato di input `centrality` è invece un booleano che indica se debbano essere considerate anche le due feature relative alla centralità (*closeness* e *betweenness*). Se il suo valore è `False`, vengono considerate esclusivamente feature relative a proprietà locali e puramente geometriche. Se invece il suo valore è `True`, alle feature *locali* sono aggiunte anche le due feature *globali* relative alla nozione di centralità. Queste feature descrivono e valutano quantitativamente il ruolo della stanza nella struttura dell'edificio.

Al fine di testare una situazione in cui siano considerate anche informazioni *globali* oltre che *locali*, sia l'addestramento che la classificazione sono stati sperimentati considerando o meno le due feature relative alla centralità (*closeness* e *betweenness*).

I dati di output della funzione consistono in un array `training_rset`, contenente dei vettori di feature relativi alle stanze ground truth, e in un array `training_labels` composto dalle corrispondenti label ground truth.

Ciascun dato di training utilizzato in questo lavoro di tesi allo scopo di addestrare il classificatore extra-trees consiste in una stanza ground truth  $r_i$  etichettata, espressa come una coppia (`training_rset[i], training_labels[i]`).

Definiti i dati di training, può essere realizzata l'operazione di addestramento del classificatore, per mezzo della quale viene indotta una funzione che può essere utilizzata per predire altri esempi. L'addestramento viene eseguito dal modulo invocando la funzione `fit` sul classificatore `clf` precedentemente generato. La funzione prende come dati di input `training_rset` e `training_labels`.

#### 4.5.4 Classificazione

Conclusa la fase di training, il modulo può procedere con la classificazione delle stanze che compongono il layout. Vengono innanzitutto ricavate le feature relative alle stanze da classificare. Queste feature rappresentano i dati di testing tramite la cui analisi il classificatore può predire una label. Viene invocata la funzione `get_testing_data` che prende come dati di input `centrality`, che indica se considerare o meno le due feature di centralità nella predizione di tali label, e `nomeXML`, ovvero il nome del file XML generato in precedenza e contenente la descrizione dell'edificio e delle stanze che lo compongono. La funzione restituisce un array `testing_rset` composto dai vettori di feature che descrivono ciascuna stanza che si vuole classificare.

A questo punto il modulo invoca la funzione `classifica` che prende come dati di input il classificatore `clf`, i dati di testing `testing_rset` e la classe di label `labels_type` che si vuole predire. Per ciascun vettore di feature incluso in `testing_rset` il classificatore `clf` predice una label di tipo `labels_type`. L'output della funzione è una lista `predizioni` delle etichette semantiche così predette, una per ogni riga dei dati di testing.

#### 4.5.5 Creazione della mappa semantica

Conclusa la classificazione delle stanze, il modulo procede con la creazione della mappa semantica. Questa viene costruita a partire dal grafo topologico `G` creato dal modulo precedente, associando ad ogni nodo di tale grafo la predizione corrispondente inclusa in `predizioni`. Quest'operazione viene eseguita invocando la funzione `creaMappaSemantica` che prende come dati di input il grafo topologico `G` e la lista di label `predizioni`. L'output è un nuovo grafo `GSemantico` in cui ad ogni nodo viene attribuita la relativa label predetta dal classificatore, plottandola al centro del nodo.

#### 4.5.6 Dati di input/output

I dati di input del modulo sono:

- `stanze`: La lista di stanze in cui è suddiviso l'edificio.
- `G`: il grafo topologico generato dal modulo precedente, di tipo `networkx.graph`.
- `nomeXML`: nome del file XML in cui viene salvata la descrizione dell'ambiente.
- `collegate`: lista di coppie di stanze connesse.
- `doorsVertices`: template delle porte.

Il modulo restituisce la mappa semantica così creata sotto forma di grafo di tipo `networkx.graph` chiamata `GSemantico`, concludendo l'implementazione del sistema di rappresentazione multilivello.



## Capitolo 5

# Realizzazioni sperimentali e valutazione

In questo capitolo sono mostrate le valutazioni sperimentali relative al lavoro sviluppato in questa tesi. Vengono in prima istanza valutati qualitativamente i singoli passaggi di preprocessing e di segmentazione, esaminando tre casi distinti in base alla natura della mappa metrica di input: mappa a griglia completa, mappa a griglia parziale e planimetria, di cui è fornito un esempio in Figura 5.1.

1. **Mappa a griglia completa:** mappa a griglia (tipologia presentata in Sezione 2.3) derivata da un'esplorazione totale dell'ambiente da parte di un robot mobile, il quale ha raccolto dati sensoriali relativi a tutte le porzioni di spazio che compongono l'ambiente.
2. **Mappa a griglia parziale:** mappa a griglia derivata da un'esplorazione parziale dell'ambiente da parte di un robot mobile. Alcune porzioni di spazio risultano sconosciute, non essendo stati raccolti dati sensoriali a sufficienza.
3. **Planimetria:** disegno bidimensionale in scala che mostra una vista dall'alto dell'ambiente.

Successivamente vengono illustrati i risultati relativi alle performance globali di segmentazione e classificazione semantica, eseguendo una valutazione sia da un punto di vista qualitativo (tramite degli esempi visuali di segmentazione e classificazione), sia da un punto di vista quantitativo (tramite la definizione di alcune metriche).

Tutti i risultati discussi in questo capitolo sono stati ottenuti eseguendo l'approccio di mapping multilivello su di un singolo core di una Intel Core



*Figura 5.1: (a) Mappa a griglia completa; (b) Mappa a griglia parziale; (c) Planimetria.*

i5-2410M@2.30GHz CPU con 4GB RAM. L’approccio è stato sperimentato su 30 mappe a griglia (20 complete e 10 parziali) relative ad edifici scolastici reali ed ottenute per mezzo dei lavori presentati in [60], [61] e [59], e su 26 planimetrie, di cui 6 tratte da [27] e 20 tratte da [62] (con e senza mobilia).

In Appendice A sono mostrati in modo esaustivo tutti i risultati ottenuti dalle sperimentazioni.

## 5.1 Gli step di preprocessing

Nel caso in cui la mappa metrica consista in una planimetria, essa può includere elementi relativi all’ambito dell’evacuazione, come simboli di estintori, idranti, uscite di sicurezza o indicazioni testuali sull’ambiente. In questa situazione si rende necessaria una preliminare fase di preprocessing della mappa allo scopo di ottimizzare la costruzione del layout delle stanze, come spiegato in Sezione 3.3.2. Di seguito vengono valutati i singoli passaggi di preprocessing applicati sulla planimetria mostrata in Figura 5.1 (c). I risultati ottenuti elaborando altre planimetrie con simboli e testo (Figure A.31 - A.36 in Appendice A) sono qualitativamente simili.

### 5.1.1 Template matching

Per identificare simboli o pattern nella planimetria in questo lavoro di tesi si è deciso di utilizzare una tecnica dello stato dell'arte chiamata *template matching* e presentata in Sezione 3.3.2. I template ricercati appartengono ad una lista, tratta da [75], costituita da template standard di oggetti tipicamente presenti nelle planimetrie. Tale lista include diversi template per ciascuna tipologia di oggetto, ed in Figura 3.5 ne è riportata una parte.

Il template matching sembra funzionare in modo efficace a patto che i template inclusi nella planimetria non siano parzialmente coperti e che la ricerca venga eseguita ruotando e scalando iterativamente i template modello, in quanto la tecnica è sensibile a rotazione e scaling. In caso contrario alcuni simboli potrebbero non essere identificati. Inoltre se oggetti di natura diversa si assomigliassero troppo, potrebbero essere erroneamente identificati come la medesima tipologia di oggetto.

Allo scopo di mostrare il corretto funzionamento del template matching, sono di seguito illustrati degli esempi di risultati dell'impiego di tale tecnica sulla planimetria mostrata in Figura 5.1 (c), relativamente all'individuazione di estintori, idranti e porte.

#### Individuazione degli estintori

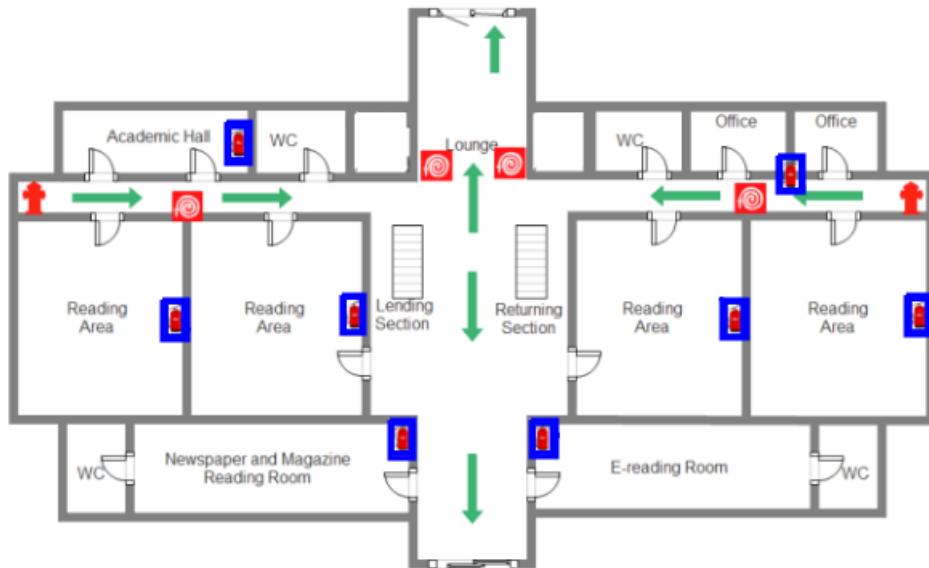
In Figura 5.2 sono mostrati gli estintori individuati all'interno di una planimetria, circondati da un rettangolo blu. Si nota come, in questo caso, l'algoritmo identifichi con precisione la totalità degli estintori presenti nell'immagine, evitando di marcare erroneamente oggetti differenti.

#### Individuazione degli idranti

La Figura 5.3 mostra i risultati del processo di identificazione degli idranti all'interno della planimetria. Gli idranti individuati sono evidenziati da un contorno rettangolare di colore blu. Anche in questo caso si nota l'accuratezza dei risultati, consistenti nell'identificazione del numero totale degli idranti senza erronei riconoscimenti di altri oggetti.

#### Individuazione delle porte

In Figura 5.4 è mostrata l'identificazione delle porte all'interno della planimetria. Analogamente a estintori e idranti, le porte sono marcate con un rettangolo blu. Il template matching sembra essere accurato anche in questa circostanza, in quanto si nota che tutte le porte presenti nella planimetria

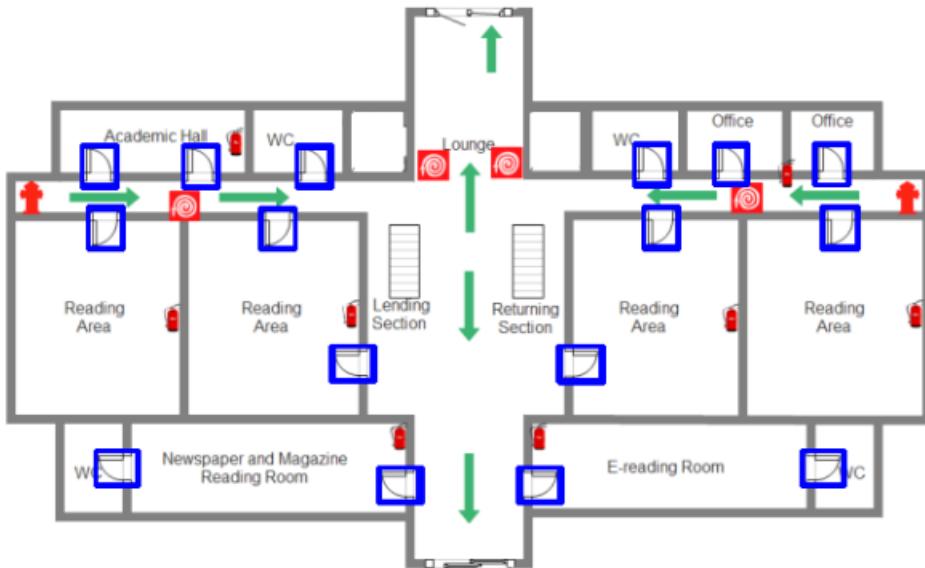


*Figura 5.2: Estintori individuati nella planimetria di Figura 5.1 (c) per mezzo della tecnica di template matching. Gli oggetti identificati sono indicati da un rettangolo blu.*



*Figura 5.3: Idranti individuati nella planimetria di Figura 5.1 (c) per mezzo della tecnica di template matching. Gli oggetti identificati sono indicati da un rettangolo blu.*

sono state individuate correttamente dall'algoritmo, evitando errori relativi ad identificazione di oggetti di altra natura.



*Figura 5.4: Porte individuate nella planimetria di Figura 5.1 (c) per mezzo della tecnica di template matching. Le porte identificate sono indicate da un rettangolo blu.*

### 5.1.2 Individuazione dell'informazione testuale

L'identificazione dell'informazione testuale presente all'interno della planimetria viene eseguita individuando gli oggetti nella planimetria il cui contorno esterno ha dimensioni inferiori rispetto ad una soglia  $\eta$  (settata a  $3 \text{ m}^2$ ), come descritto in Sezione 3.3.2. Questa operazione si basa sull'assunzione che gli oggetti relativamente piccoli corrispondano alle parole da eliminare, mentre i muri siano caratterizzati da dimensioni maggiori del contorno esterno. I risultati di tale operazione sono mostrati in Figura 5.5, in cui le parole individuate sono evidenziate da un rettangolo blu (si noti che nella planimetria raffigurata i template individuati nelle fasi precedenti sono già stati eliminati come descritto in Sezione 3.3.2). Tale assunzione, che sta alla base di questo processo, si dimostra in questo esempio corretta, in quanto dalla figura è possibile osservare che l'approccio identifica correttamente l'intera informazione testuale senza che vengano marcati elementi corrispondenti a pareti.

La tecnica sembra funzionare considerando anche gli altri esempi del dataset mostrati in Appendice A (Figure A.31 - A.36). Una sua utilità, oltre all'individuazione dell'informazione testuale, sembra essere l'individuazione parziale o completa del rumore presente nella mappa, tipicamente relativo a mobilia, come sarà mostrato in Sezione 5.3.3.

Tuttavia il metodo può introdurre degli errori nel caso in cui una stanza

abbia una superficie inferiore alla soglia  $\eta$ . Se ciò accade, la stanza viene erroneamente identificata come feature da eliminare, come mostrato in Figura 5.6 (a).

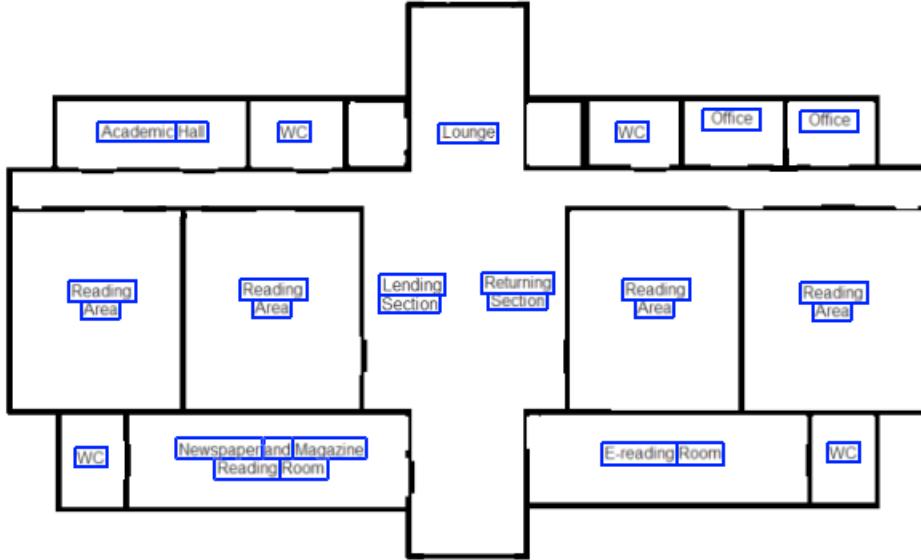
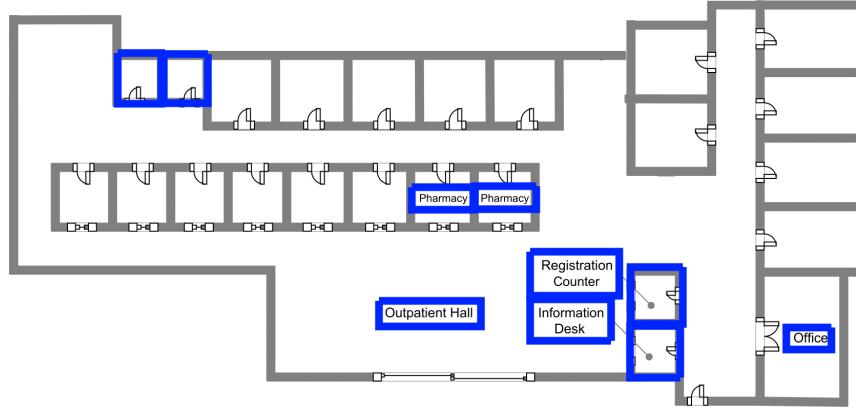


Figura 5.5: Testo individuato nella planimetria di Figura 5.1 (c) da cui sono stati eliminati tutti i simboli individuati in precedenza. L'informazione testuale è evidenziata da rettangoli blu.

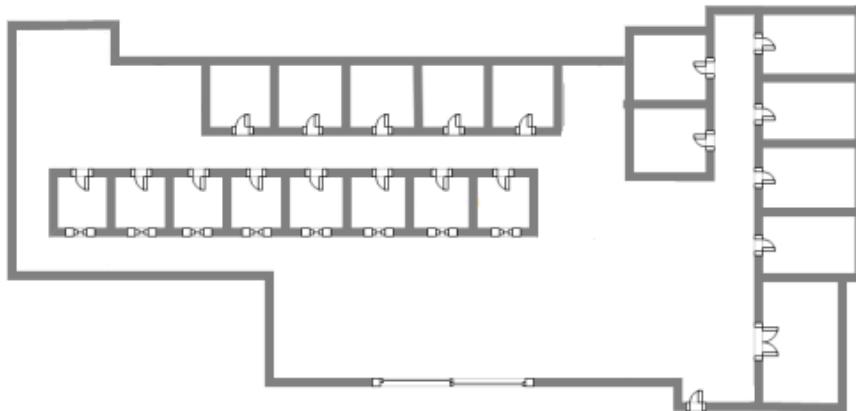
### 5.1.3 Eliminazione delle feature individuate

La fase di preprocessing viene completata attraverso l'eliminazione di tutte le feature individuate nei passi precedenti. La planimetria mostrata dunque in Figura 5.1 (c) al termine del preprocessing viene modificata così come mostrato in Figura 5.7. Nell'esempio in questione sono stati eliminati colorando del colore dello sfondo (bianco) estintori, idranti, informazione testuale e tutti gli altri simboli al di fuori delle porte, le quali sono state sostituite da porzioni di muro. Confrontando la planimetria iniziale con la mappa così generata, si nota l'accuratezza della fase di preprocessing in questo esempio, in quanto sono stati rimossi tutti gli elementi non corrispondenti a muri, producendo una mappa metrica in cui sono presenti esclusivamente le pareti dell'edificio.

I risultati finali della fase di preprocessing sembrano essere accurati anche considerando altri esempi mostrati in Appendice A (Figure A.31 - A.36). In tali esempi il metodo elimina correttamente i simboli e l'informazione testuale, lasciando invariate le pareti dell'edificio.



(a)



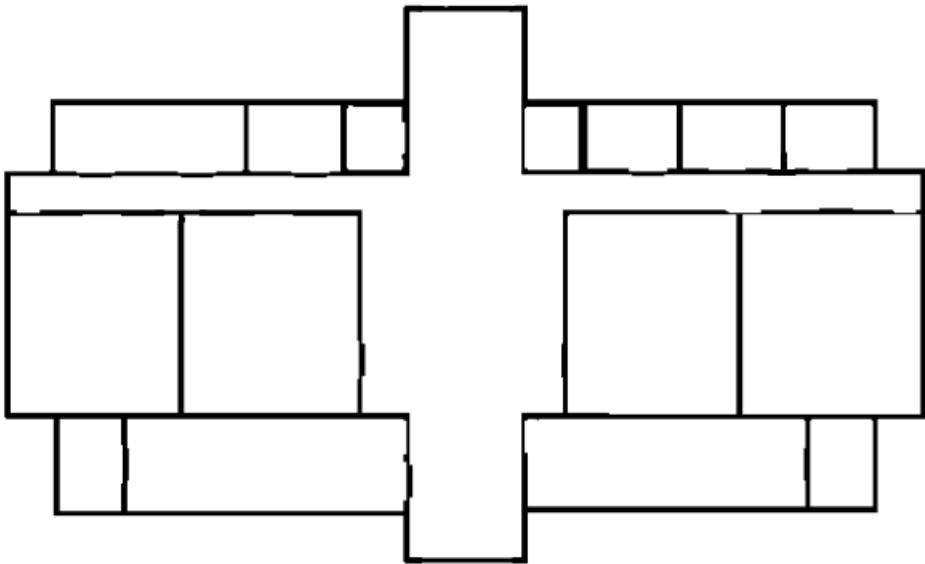
(b)

*Figura 5.6: (a) Erronea individuazione di stanze come elementi da eliminare, oltre all'informazione testuale; (b) Mappa ottenuta al termine del preprocessing, in cui sono state eliminate erroneamente alcune stanze.*

L'eliminazione delle feature individuate può non essere accurata nel caso in cui tra le feature da eliminare siano state identificate erroneamente delle stanze. Tali stanze verrebbero eliminate dalla planimetria, come mostrato in Figura 5.6 (b) influendo negativamente sulla precisione delle fasi successive.

## 5.2 Gli step di costruzione del layout delle stanze

In questa sezione vengono esaminate le valutazioni sperimentali relative ai singoli passaggi di costruzione del layout delle stanze. Per ciascuno di questi passaggi vengono mostrati come esempio i risultati riguardanti i 3 diversi



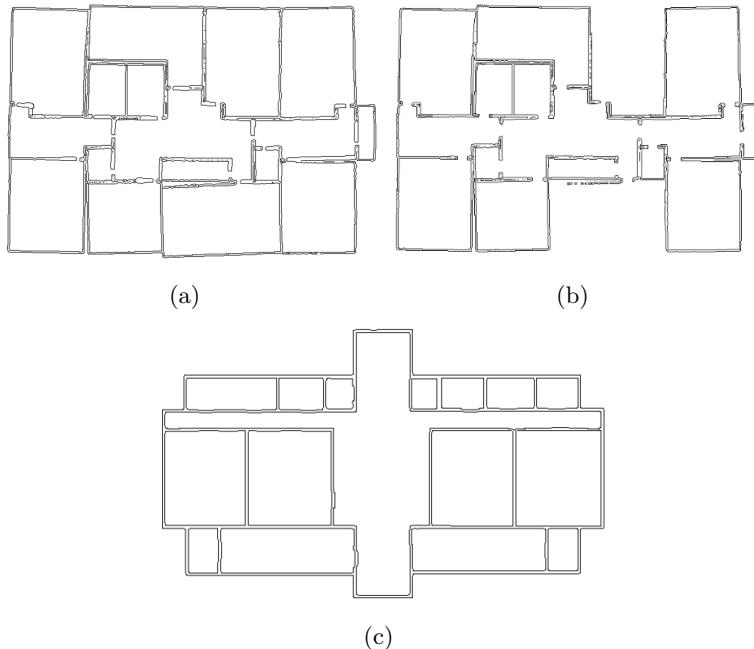
*Figura 5.7: Esempi finali della fase di preprocessing. Sono stati eliminati dalla planimetria di Figura 5.1 (c) tutti i simboli non corrispondenti a muri.*

esempi mostrati in Figura 5.1. I risultati relativi agli altri input mostrati in Appendice A sono qualitativamente simili.

### 5.2.1 Canny edge detection

Il primo passo nel processo di costruzione del layout delle stanze consiste nell'individuazione dei segmenti che corrispondono ai muri dell'edificio. Per identificare questi segmenti viene in prima istanza applicato l'algoritmo di Canny edge detection (già discusso in Sezione 3.3.3) allo scopo di individuare i contorni presenti nella mappa metrica. Il termine ‘contorno’ indica un insieme di punti di un’immagine in cui l’intensità luminosa cambia bruscamente.

Il risultato dell’applicazione dell’algoritmo sui 3 esempi è mostrato in Figura 5.8. Più precisamente, (a) si riferisce alla mappa a griglia completa in Figura 5.1 (a); (b) alla mappa a griglia parziale in Figura 5.1 (b); (c) alla planimetria in Figura 5.1 (c) (preprocessata in modo da ottenere la mappa in Figura 5.7). Confrontando i risultati con le mappe di partenza si osserva nei 3 casi l’accuratezza dell’algoritmo nell’individuare la totalità dei contorni interni alle mappe metriche. Viene così generata un’immagine contenente molte meno informazioni rispetto a quella originale. Sono infatti eliminati i dettagli non rilevanti, conservando invece le informazioni essen-



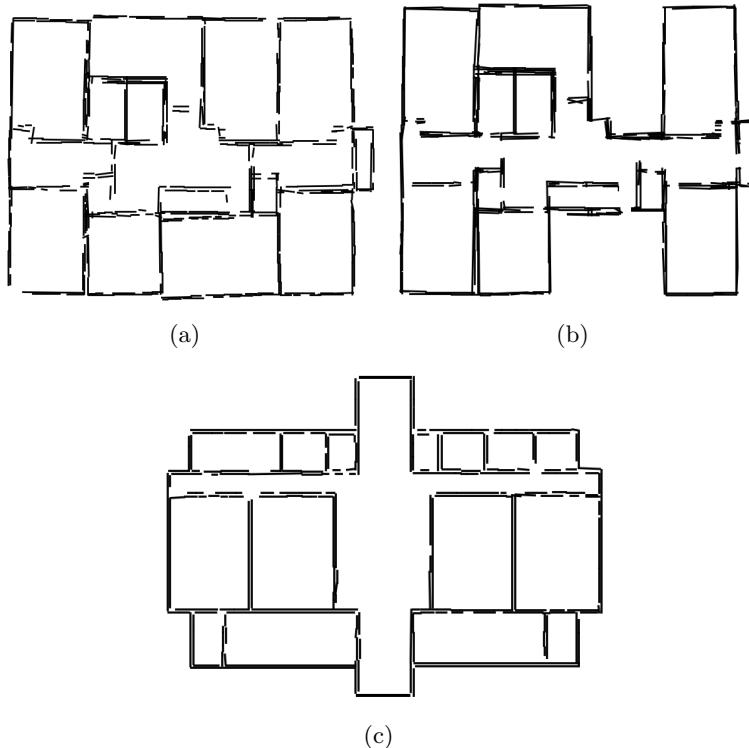
*Figura 5.8: Esempi di Canny edge detection relativi a: (a) Mappa a griglia completa; (b) Mappa a griglia parziale; (c) Planimetria.*

ziali per descrivere la forma e le caratteristiche strutturali e geometriche dell’ambiente.

### 5.2.2 Hough line transform

Il secondo e ultimo passaggio per ottenere i segmenti corrispondenti ai muri consiste nell’applicazione dell’algoritmo di Hough line transform (il cui funzionamento è spiegato in Sezione 3.3.3) sui risultati ottenuti dal Canny edge detector. Questo algoritmo è in grado di estrarre dei segmenti, ciascuno dei quali caratterizzato dai suoi due estremi, analizzando i contorni estratti dal Canny edge detector.

I risultati relativi alle mappe di esempio sono mostrati in Figura 5.9: (a) si riferisce alla mappa a griglia completa in Figura 5.1 (a); (b) alla mappa a griglia parziale in Figura 5.1 (b); (c) alla planimetria in Figura 5.1 (c). Analogamente al Canny edge detector, si osserva come negli esempi l’algoritmo riesca a riconoscere in modo efficace i segmenti presenti nelle mappe metriche di partenza. I segmenti individuati generano infatti una struttura quasi equivalente a quella riportata nella corrispondente mappa metrica. Si nota che per ogni muro presente nella mappa metrica l’algoritmo di Hough line transform potrebbe non estrarre un singolo segmento lungo



*Figura 5.9: Esempi di Hough line transform relativi a: (a) Mappa a griglia completa; (b) Mappa a griglia parziale; (c) Planimetria.*

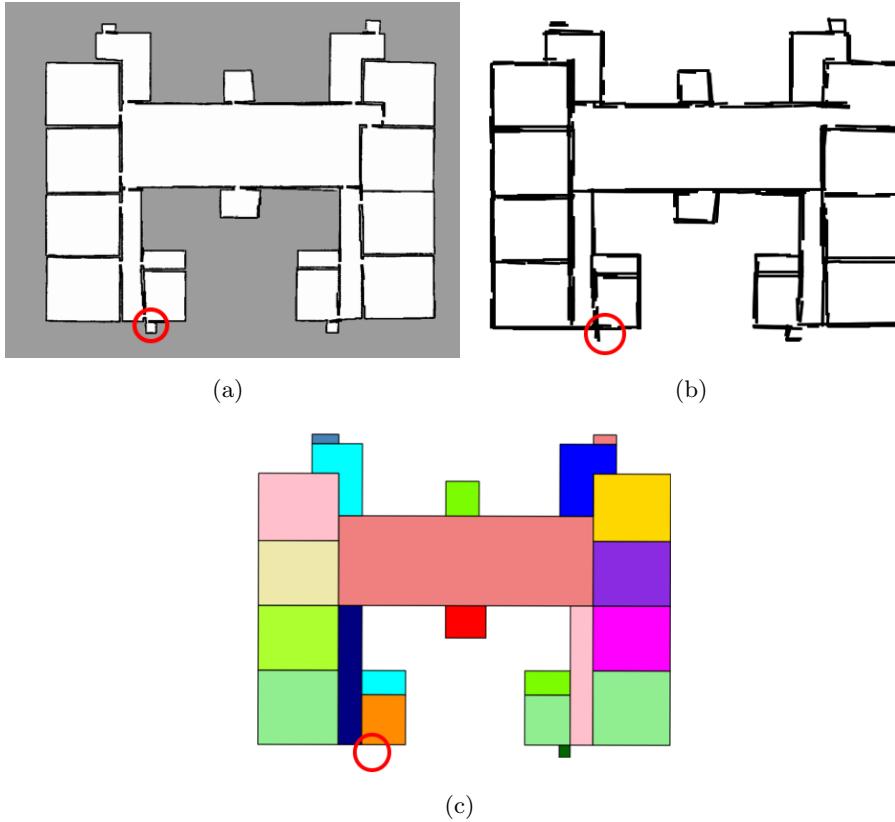
quanto il muro, bensì più segmenti di dimensione minore. Tuttavia essi sono disposti in modo tale che il loro insieme corrisponda all'intero muro in questione.

Anche considerando tutti gli altri esempi mostrati in Appendice A l'algoritmo di Hough line transform sembra essere accurato. In tali esempi viene individuata la quasi totalità dei segmenti corrispondenti ai muri dell'edificio.

Il metodo sembra non funzionare bene nel caso in cui alcuni dei muri da identificare abbiano un'estensione molto ridotta. L'algoritmo in questi casi può non riuscire ad individuare i segmenti corrispondenti, rendendo meno precisa la segmentazione successiva, come mostrato in Figura 5.10.

### 5.2.3 Clustering dei segmenti

Dopo aver ottenuto i segmenti corrispondenti ai muri, su di essi viene eseguito il processo di clustering. Come illustrato in Sezione 3.3.4, il clustering si compone di due diverse fasi: clustering angolare e spaziale. I segmenti vengono così raggruppati in base al loro orientamento (inclinazione) ed al-

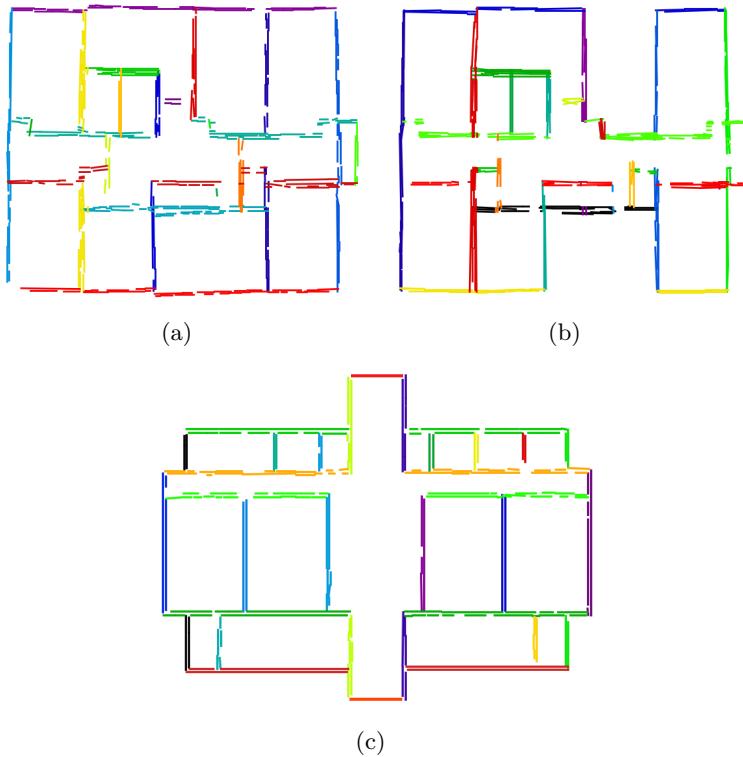


*Figura 5.10: (a) Mappa metrica in cui in basso a sinistra è messa in evidenza da un cerchio rosso una stanza caratterizzata da muri di piccola dimensione; (b) Segmenti individuati dall'algoritmo di Hough line transform. Il cerchio rosso in basso a sinistra mette in evidenza la mancata individuazione dei muri evidenziati in (a); (c) Layout delle stanze in cui, a causa dell'imprecisione dell'algoritmo di Hough line transform mostrata in (b), non è stata identificata la stanza marcata in (a).*

la loro distanza laterale. I risultati del processo di clustering sulle mappe di esempio sono mostrati in Figura 5.11, relativamente a: (a) la mappa a griglia completa, (b) la mappa a griglia parziale e (c) la planimetria. Ad ogni cluster è associato un diverso colore. I risultati mostrano la correttezza del processo di clustering nei casi di esempio; si osserva infatti che ciascun cluster risulta essere costituito da segmenti aventi un orientamento analogo ed una distanza laterale trascurabile.

Il processo di clustering dei segmenti sembra essere accurato anche considerando gli altri casi di esempio mostrati in Appendice A. Sembra funzionare bene nel caso di muri poco spessi e con orientamento orizzontale o verticale.

Il metodo può invece non produrre risultati precisi nel caso di muri molto

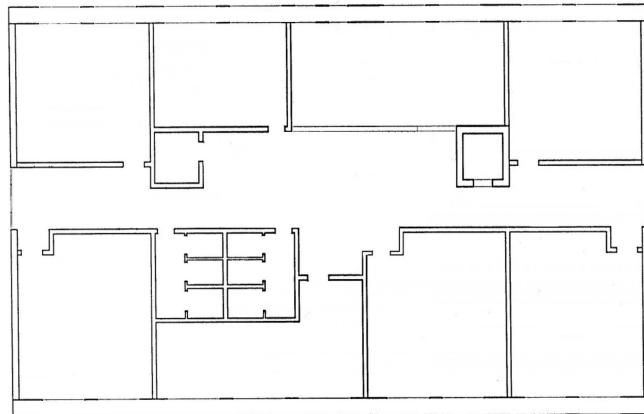


*Figura 5.11: Esempi di clustering dei segmenti corrispondenti a muri, relativi a: (a) Mappa a griglia completa; (b) Mappa a griglia parziale; (c) Planimetria.*

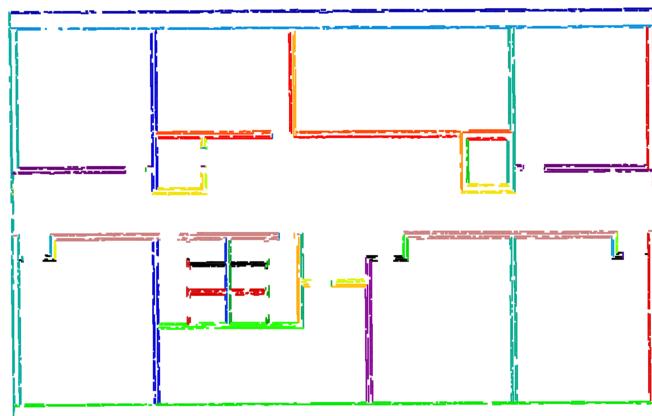
spessi o con orientamento obliqui, raggruppando in cluster diversi segmenti che in realtà corrispondono allo stesso muro, come mostrato in Figura 5.12.

#### 5.2.4 Suddivisione in celle

A partire dai segmenti rappresentanti i muri dell’edificio, estratti tramite l’algoritmo Hough line transform, viene costruito un insieme di celle, come spiegato in Sezione 3.3.5. Questa operazione ricopre grande importanza all’interno del processo di costruzione del layout delle stanze, poiché le stanze che costituiscono l’edificio saranno successivamente ottenute tramite raggruppamento delle celle definite in questa fase. In Figura 5.13 sono mostrati i partizionamenti in celle dei 3 esempi analizzati in questa sezione: (a) planimetria, (b) mappa a griglia completa, (c) mappa a griglia parziale. L’insieme delle celle è generato da rette rappresentative dei cluster individuati nella fase precedente. Ciascuna retta (disegnata in rosso) rappresenta un diverso cluster. Nelle immagini sono riportati in nero anche i segmenti, individuati in precedenza tramite Canny edge detection e Hough line transform.



(a)

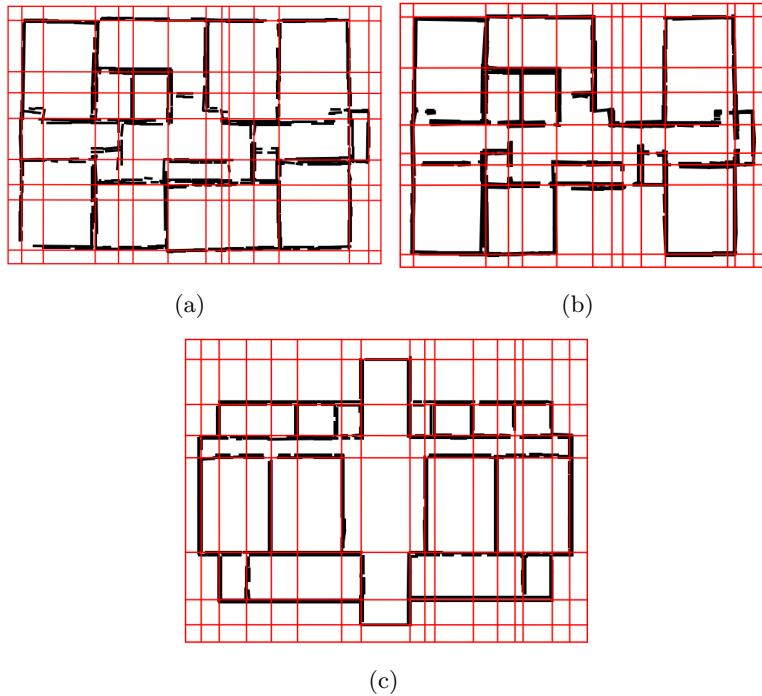


(b)

*Figura 5.12: (a) Mappa metrica i cui muri perimetrali sono caratterizzati da un ampio spessore; (b) Segmenti appartenenti allo stesso muro perimetrale (in alto e in basso) sono raggruppati erroneamente in cluster diversi (diverso colore), a causa della loro elevata distanza laterale.*

### 5.2.5 Clustering delle celle

Il clustering delle celle costituisce l'operazione finale del processo di costruzione del layout delle stanze. Attraverso questa operazione le celle vengono raggruppate in stanze, utilizzando l'algoritmo di DBSCAN, il cui funzionamento è spiegato in Sezione 3.3.6. Il clustering si basa sull'affinità tra celle adiacenti, valutata considerando la probabilità dei loro segmenti perimetrali di corrispondere a muri realmente presenti nell'ambiente. Tale probabilità è calcolata tramite la formula (3.4) in Sezione 3.3.5, ed assegnata come peso a

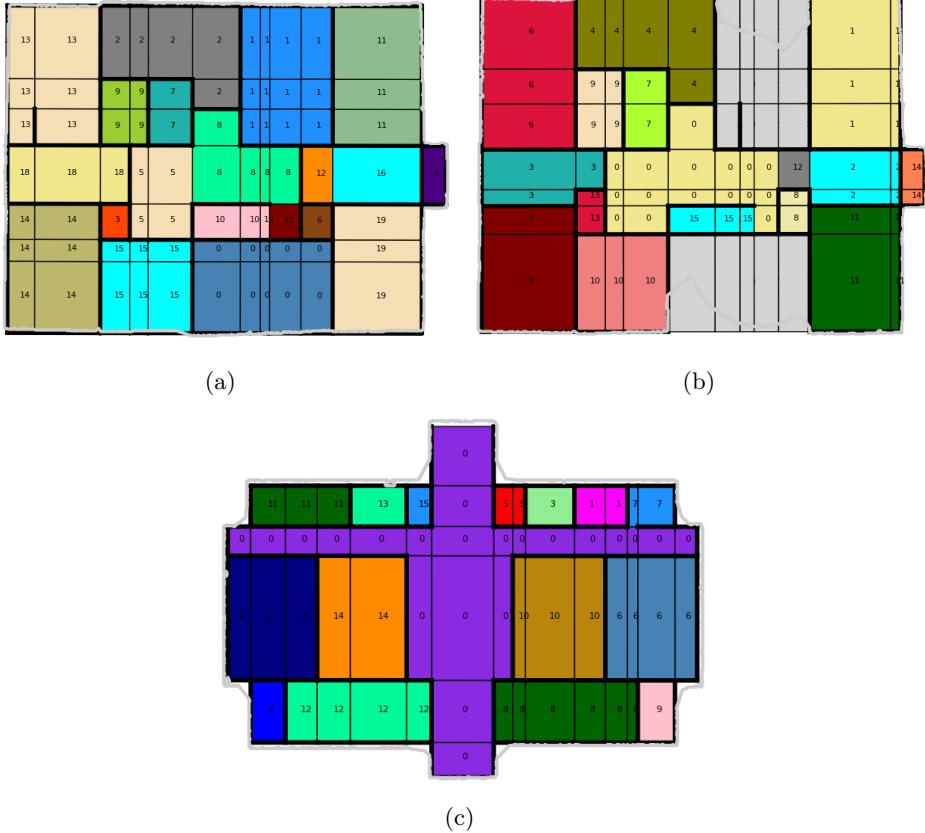


*Figura 5.13: Partizionamento in celle di: (a) Mappa a griglia completa; (b) Mappa a griglia parziale; (c) Planimetria.*

ciascun segmento. L'affinità tra due celle adiacenti è tanto più alta quanto più è basso il peso del segmento che le separa. I risultati del clustering eseguito tramite DBSCAN sulle 3 mappe di esempio sono mostrati in Figura 5.14.

Analogamente al clustering eseguito sui segmenti, anche in questo caso si è deciso di assegnare un colore diverso a ciascun raggruppamento di celle. Le celle appartenenti ad uno stesso cluster sono colorate allo stesso modo e contengono al loro interno un numero corrispondente all'indice del cluster a cui appartengono, identificato dall'algoritmo DBSCAN. Di particolare interesse è il risultato relativo alla mappa a griglia parziale (Figura 5.14 (b)). Si osserva la corretta identificazione delle due stanze parziali, ovvero delle porzioni di spazio di cui il robot ha effettuato un'esplorazione non completa. Le celle potenzialmente facenti parte di una stanza parziale sono colorate di grigio.

Nell'esempio in Figura 5.15 si osserva che diminuendo la soglia minima  $\delta$  di intersezione tra una cella ed il contorno esterno dell'edificio (soglia utilizzata per identificare le celle parziali, come spiegato in Sezione 3.3.5), la dimensione delle stanze parziali aumenta: in (c) è stata utilizzata una

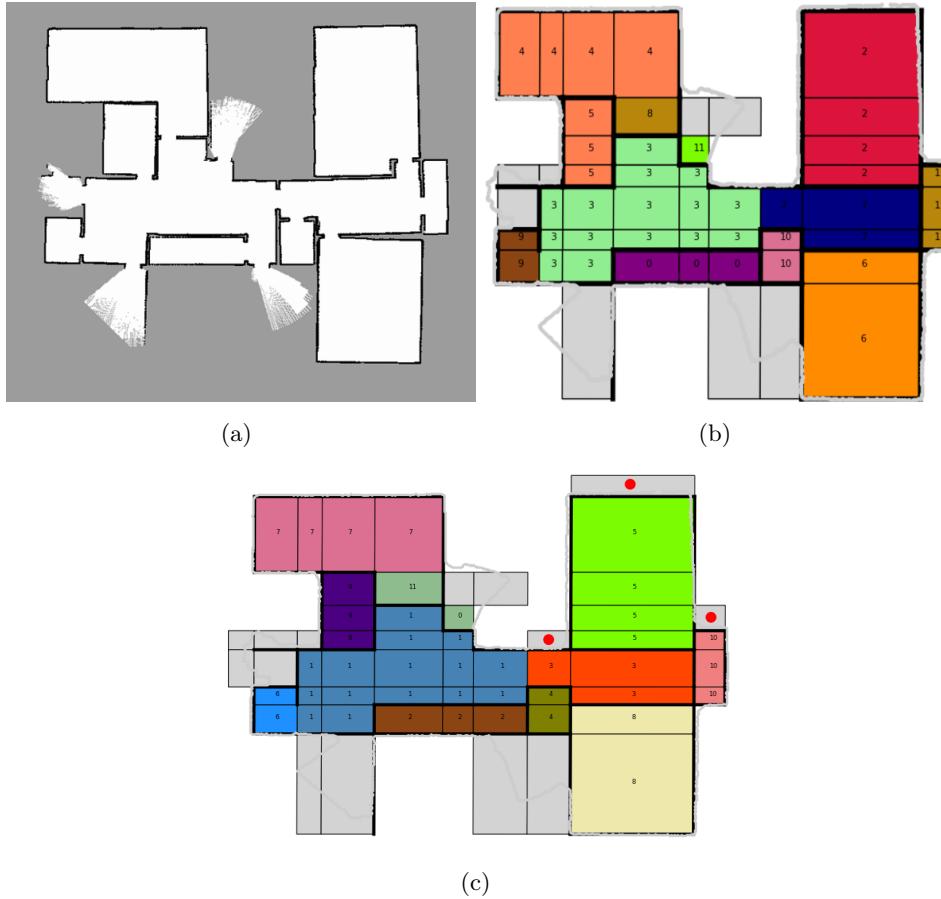


*Figura 5.14: Esempi di raggruppamento di celle in stanze tramite DBSCAN applicato su: (a) Mappa a griglia completa; (b) Mappa a griglia parziale; (c) Planimetria.*

soglia (un terzo dell'area della cella in questione) inferiore rispetto a quella utilizzata in (b) (metà dell'area della cella in questione), di conseguenza vi sono più celle parziali. Tuttavia tale diminuzione introduce degli errori poichè vengono classificate come parziali anche celle che in realtà dovrebbero essere esterne (indicate da un cerchio rosso in (c)). Questo perchè esse intersecano in modo lieve (ma superiore alla soglia  $\delta$ ) il contorno esterno dell'edificio.

Il clustering delle celle sembra funzionare correttamente anche negli altri esempi del dataset mostrati in Appendice A. Il metodo si dimostra accurato se l'insieme di celle non è troppo complesso, ovvero se esso è stato generato, nella fase precedente, da rette perpendicolari con due soli orientamenti possibili.

Il clustering sembra non essere preciso nel caso in cui la complessità dell'insieme di celle sia elevata. Ciò si verifica se vi sono più di due orientamenti

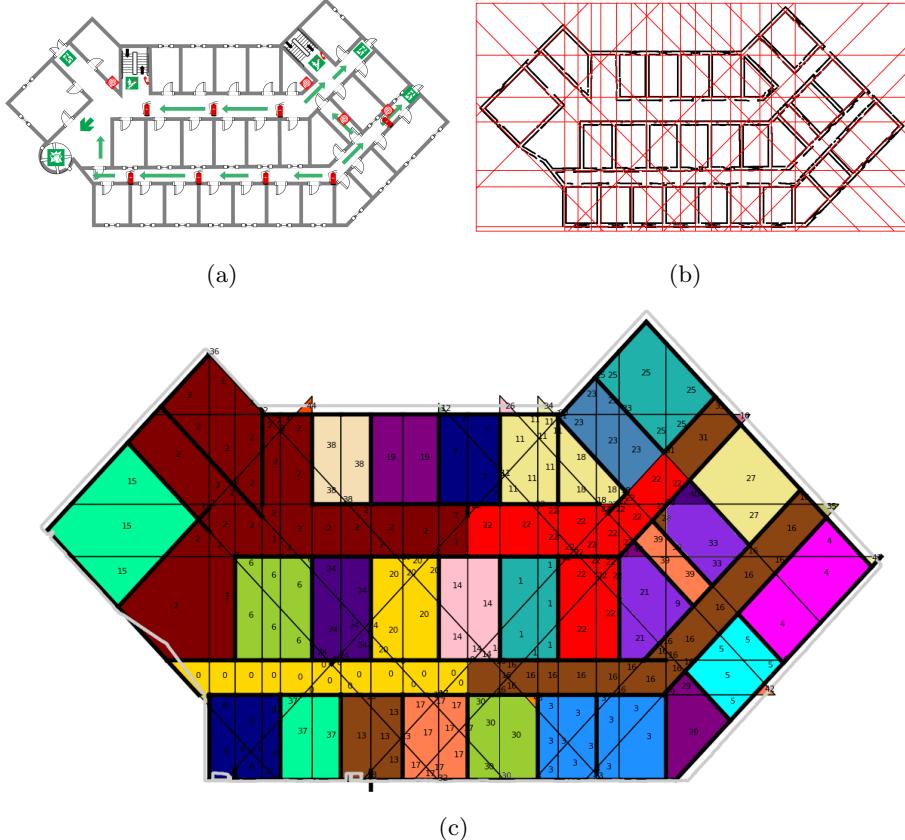


*Figura 5.15: (a) Mappa a griglia parziale; (b) Clustering delle celle, con celle parziali colorate di grigio; (c) Clustering delle celle con soglia di classificazione delle celle parziali diminuita rispetto a (b). Sono individuate di conseguenza più celle parziali, introducendo tuttavia degli errori: le celle parziali che in realtà dovrebbero essere esterne sono evidenziate da un cerchio rosso al loro interno.*

delle rette generatrici, con la conseguente presenza di rette diagonali e non perpendicolari, come mostrato in Figura 5.16.

### 5.3 Performance globali di segmentazione

In questa sezione vengono valutate le performance globali di segmentazione dell’ambiente in stanze, considerando la totalità degli esempi sperimentati in questo lavoro di tesi. Le valutazioni sono fatte sia in modo qualitativo (tramite esempi visuali di segmentazione), sia in modo quantitativo (tramite la definizione di metriche).

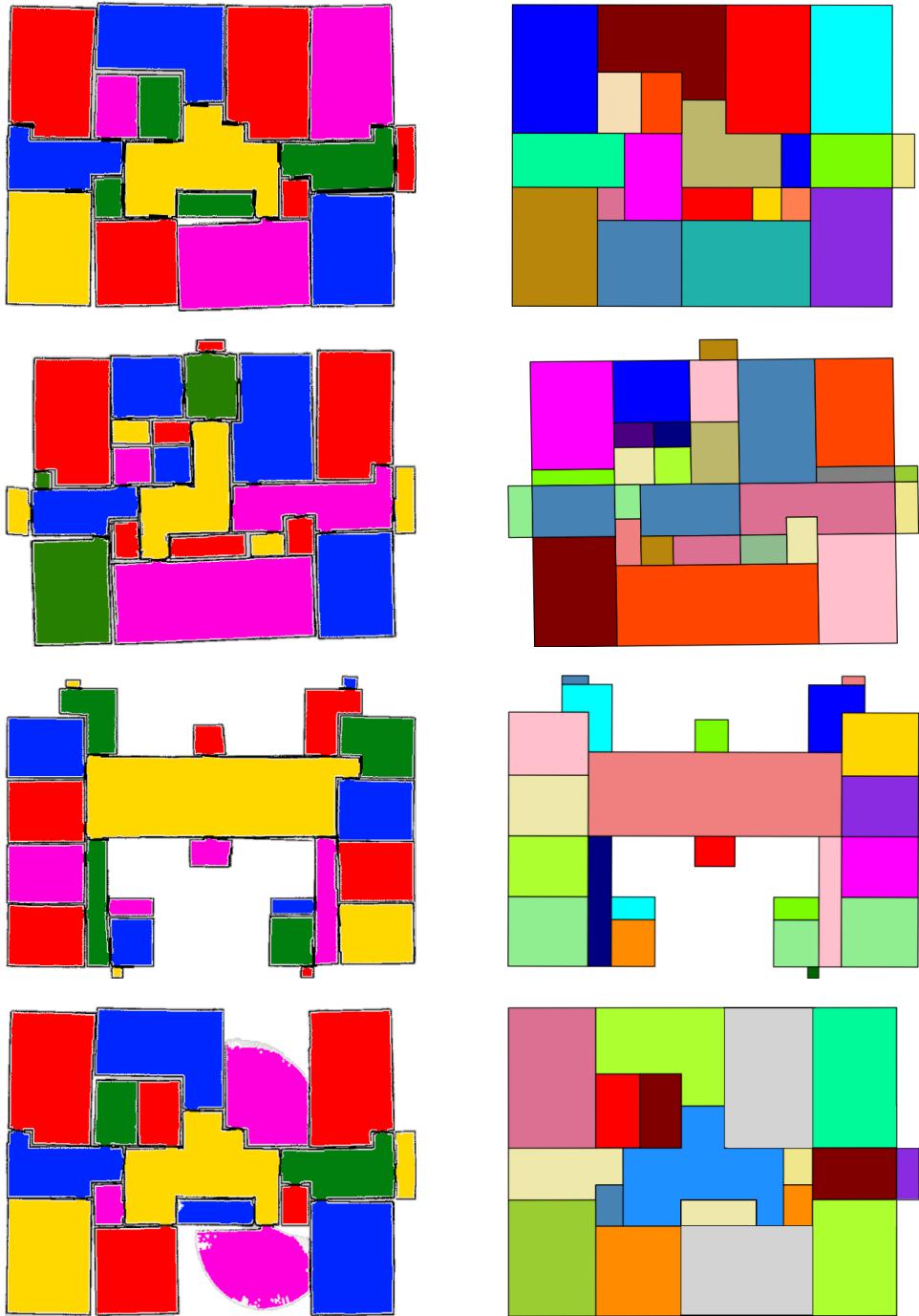


*Figura 5.16: (a) Mappa metrica; (b) Insieme di celle con elevata complessità; (c) Clustering delle celle caratterizzato da imprecisioni.*

### 5.3.1 Valutazione qualitativa

Il completamento del processo di costruzione del layout delle stanze viene conseguito tramite un'elaborazione dei risultati restituiti dall'algoritmo di clustering DBSCAN, mostrati in Figura 5.14. Le celle individuate come appartenenti allo stesso cluster vengono unite in un'unica porzione di spazio, rappresentante una stanza dell'edificio. In Figura 5.17 e Figura 5.18 sono mostrati alcuni esempi significativi di layout delle stanze ottenuti, con corrispondente mappa metrica di partenza (mappe a griglia di ambienti esplorati in Figura 5.17, planimetrie in Figura 5.18 rispettivamente). A ciascuna stanza è stato associato un diverso colore, sia nelle mappe metriche originali che nei layout prodotti.

Osservando i risultati nelle figure si nota una generica accuratezza del processo di segmentazione. Negli esempi mostrati infatti i layout delle stanze costruiti risultano caratterizzati da una buona somiglianza con le strutture



*Figura 5.17: Esempi di layout delle stanze (a destra) ottenuti partendo da mappe a griglia (a sinistra) di ambienti esplorati. Ad ogni stanza è associato un diverso colore.*

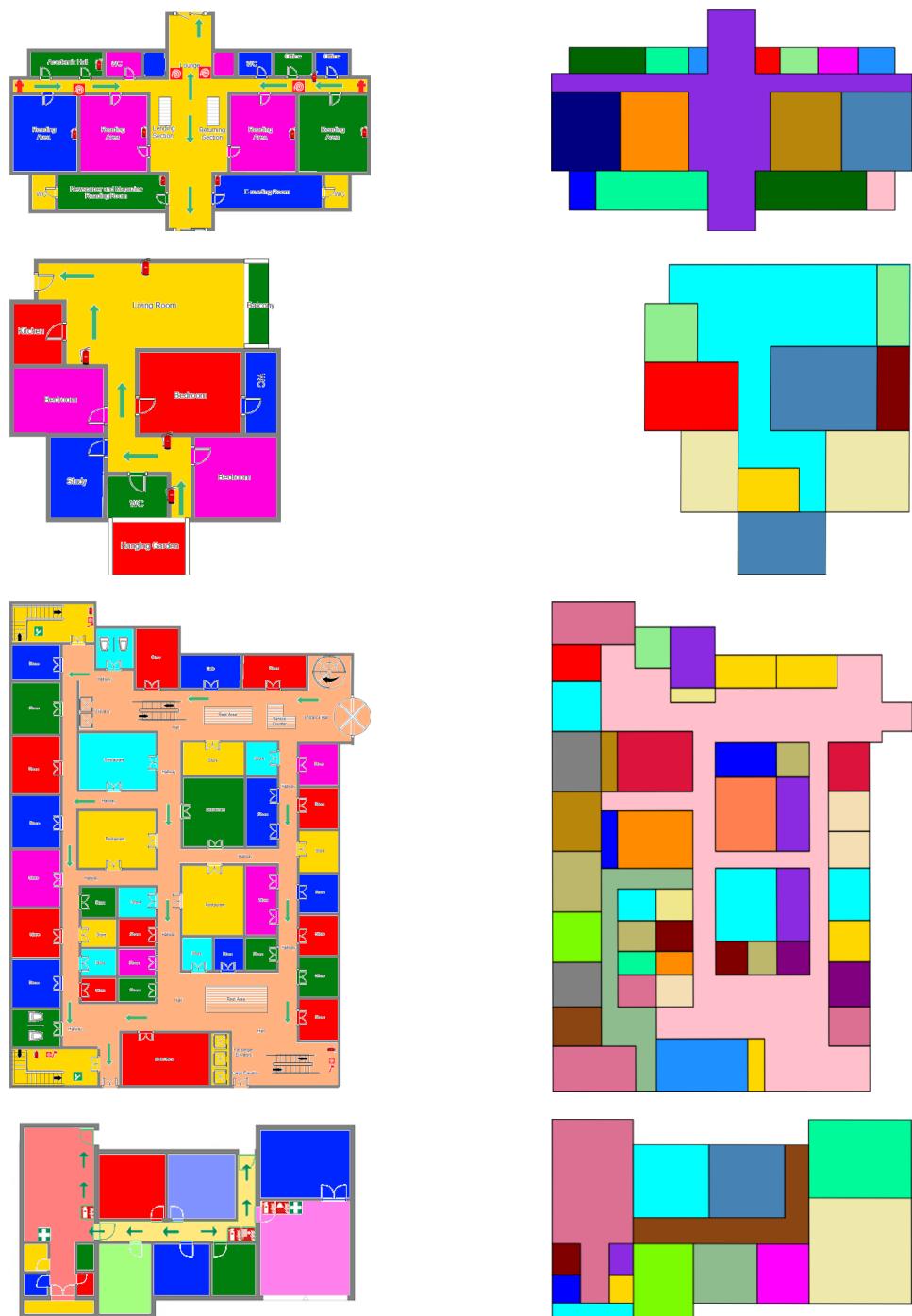


Figura 5.18: Esempi di layout delle stanze (a destra) ottenuti da planimetrie (a sinistra). Ad ogni stanza è associato un diverso colore.

rappresentate nelle mappe originali (le cui stanze verranno d'ora in poi chiamate stanze *ground truth*). Le poche discrepanze sono correlate alle nozioni di under-segmentazione e over-segmentazione.

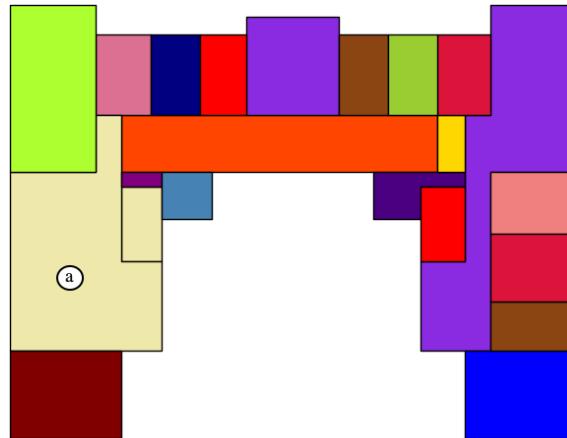
- **Under-segmentazione:** il risultato della segmentazione consiste in un numero di stanze inferiore rispetto al numero di stanze ground truth che coprono la stessa area. Questo è il caso mostrato in Figura 5.19, relativo alla Figura A.1 in Appendice A. Si nota il minor numero di stanze segmentate (Figura 5.19 (a)) rispetto a quelle ground truth (Figura 5.19 (b)). Nello specifico, le stanze ground truth indicate dai numeri 1 e 2 in Figura 5.19 (b) sono state unite in un'unica stanza *a* in Figura 5.19 (a).
- **Over-segmentazione:** il layout delle stanze ottenuto dalla segmentazione si compone di un numero di stanze superiore alla quantità di stanze ground truth che coprono la stessa area. Questo è il caso mostrato in Figura 5.20 (corrispondente al primo esempio in Figura 5.17), in cui si nota il maggior numero di stanze segmentate (Figura 5.20 (a)) rispetto a quelle ground truth (Figura 5.20 (b)). Nello specifico, la stanza ground truth indicata dal numero 1 in Figura 5.20 (b) è stata segmentata nelle tre diverse stanze *a*, *b* e *c* in Figura 5.20 (a). Analogamente la stanza ground truth indicata dal numero 2 corrisponde alle due stanze segmentate *d* ed *e*.

Tramite un'analisi qualitativa dei risultati di segmentazione mostrati in Figura 5.17 e 5.18 si osserva una lieve tendenza all'over-segmentazione, probabilmente causata da piccole imprecisioni da parte dell'algoritmo di Hough line transform nella fase di individuazione dei segmenti corrispondenti ai muri. Queste imprecisioni possono provocare l'erronea attribuzione di un peso non trascurabile ad alcuni lati dell'insieme di celle, anche senza una reale corrispondenza con i muri dell'edificio. Ciò può portare all'over-segmentazione.

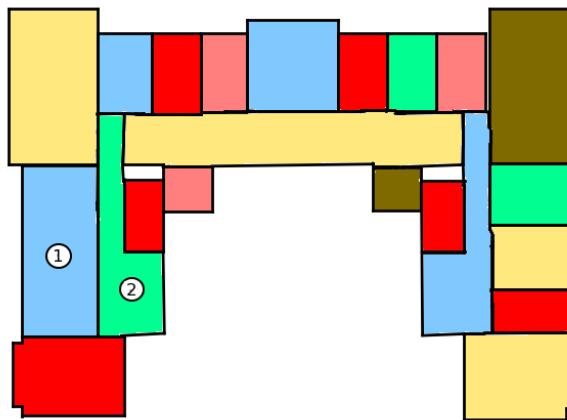
Tale tendenza è riscontrabile nel primo, secondo e quarto esempio in Figura 5.17 e nel terzo esempio in Figura 5.18.

### 5.3.2 Valutazione quantitativa

Allo scopo di valutare quantitativamente la correttezza dei risultati e le performance globali del processo di segmentazione, si è deciso di utilizzare due metriche che indichino la corrispondenza tra stanze segmentate dal nostro metodo e stanze della mappa metrica di riferimento (chiamate stanze ground



(a)



(b)

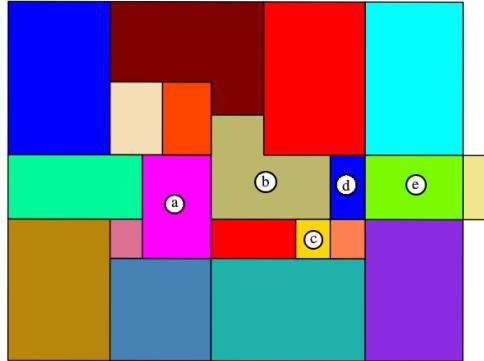
*Figura 5.19: Esempio di under-segmentazione: (a) Layout delle stanze segmentate; (b) Layout delle stanze ground truth.*

truth). Questa corrispondenza viene definita in termini di *copertura*, intesa come un numero che indica l'area sovrapposta tra due superfici.

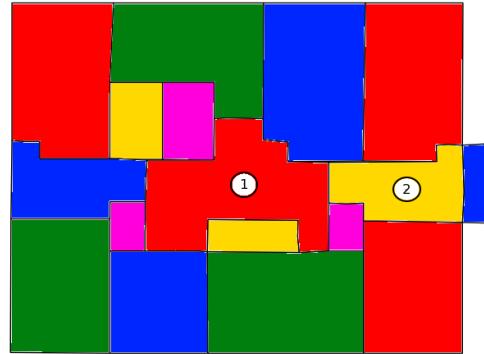
Dati due layout  $S$  e  $G$ , e due stanze  $x \in S$  e  $y \in G$ ,  $x$  è tanto più coperta da  $y$  quanto più la superficie di  $x$  è sovrapposta a quella di  $y$ .

Nel nostro caso i due layout su cui viene analizzata la proprietà di copertura corrispondono all'insieme di stanze segmentate e a quello di stanze ground truth.

In questo lavoro di tesi si è scelto di definire ed utilizzare due diversi tipi di coperture, introducendo le nozioni di *forward coverage* e *backward coverage*, ispirate da alcune metriche introdotte in [9] per confrontare diversi



(a)



(b)

Figura 5.20: Esempio di over-segmentazione: (a) Layout delle stanze segmentate; (b) Layout delle stanze ground truth.

metodi di segmentazione. La forward coverage è una funzione iniettiva, la backward coverage una funzione suriettiva. Se le segmentazioni fossero perfette sarebbero biunivoche.

- **Forward coverage:** definisce la corrispondenza delle stanze segmentate rispetto alle stanze ground truth. Data una stanza segmentata  $x$ , essa è *forward covered* da una stanza ground truth  $y$  se  $y$ , tra tutte le stanze ground truth, è quella che massimizza la copertura di  $x$ . Viene definita la funzione indicatrice  $\mathbb{1}_{FC}(x, y)$ :

$$\mathbb{1}_{FC}(x, y) = \begin{cases} 1, & \text{se } y \text{ massimizza la copertura di } x \\ 0, & \text{altrimenti} \end{cases}$$

Relativamente alla nozione di forward coverage viene anche definita la funzione  $Area_{FC}(x)$  che indica l'estensione dell'intersezione tra la

superficie della stanza segmentata  $x$  e della stanza ground truth da cui  $x$  è *forward covered*.

- **Backward coverage:** definisce la corrispondenza delle stanze ground truth rispetto alle stanze segmentate. Data una stanza ground truth  $y$ , essa è *backward covered* da una stanza segmentata  $x$  se  $x$ , tra tutte le stanze segmentate, è quella che massimizza la copertura di  $y$ . Viene definita la funzione indicatrice  $\mathbb{1}_{BC}(x, y)$ :

$$\mathbb{1}_{BC}(x, y) = \begin{cases} 1, & \text{se } x \text{ massimizza la copertura di } y \\ 0, & \text{altrimenti} \end{cases}$$

Relativamente alla nozione di backward coverage viene anche definita la funzione  $Area_{BC}(y)$  che indica l'estensione dell'intersezione tra la superficie della stanza ground truth  $y$  e della stanza segmentata da cui  $y$  è *backward covered*.

In base alle due nozioni di *forward coverage* e *backward coverage* tra le stanze segmentate e le stanze ground truth, possono essere calcolate due diverse stime dell'accuratezza della segmentazione per un edificio:

**Accuracy<sub>FC</sub>:** indica l'accuratezza della corrispondenza tra le stanze segmentate e le stanze ground truth da cui sono *forward covered*.

$$Accuracy_{FC} = \frac{\sum_{x \in S} \frac{Area_{FC}(x)}{Area(x)}}{|S|} \quad (5.1)$$

dove  $S$  è l'insieme delle stanze segmentate,  $Area(x)$  è la superficie della stanza segmentata  $x$  e  $|S|$  è il numero totale di stanze segmentate.

L'Accuracy<sub>FC</sub> è alta se le stanze segmentate sono interamente contenute all'interno delle stanze ground truth.

**Accuracy<sub>BC</sub>:** indica l'accuratezza della corrispondenza tra le stanze ground truth e le stanze segmentate da cui sono *backward covered*.

$$Accuracy_{BC} = \frac{\sum_{y \in G} \frac{Area_{BC}(y)}{Area(y)}}{|G|} \quad (5.2)$$

dove  $G$  è l'insieme delle stanze ground truth,  $Area(y)$  è la superficie della stanza ground truth  $y$  e  $|G|$  è il numero totale di stanze ground truth.

L'Accuracy<sub>BC</sub> è alta se le stanze ground truth sono interamente contenute all'interno delle stanze segmentate.

In generale una segmentazione è tanto più buona quanto più le due accuracy sono alte. E' perfetta se entrambe sono uguali a 1.

Le due metriche sopra definite sono tratte da [9], lavoro in cui esse fanno parte delle metriche utilizzate per confrontare quantitativamente tra loro i vari metodi di segmentazione analizzati. In [9] Accuracy<sub>BC</sub> ed Accuracy<sub>FC</sub> vengono chiamate rispettivamente *recall* e *precision*.

In Tabella 5.1 sono riportate le stime medie ( $\pm$  deviazione standard) di Accuracy<sub>FC</sub> ed Accuracy<sub>BC</sub> tratte da [9], a cui sono stati aggiunti nell'ultima colonna i valori relativi all'approccio di questo lavoro di tesi, sperimentato sulle medesime 20 mappe utilizzate in [9] (Figura A.37 - A.56 in Appendice A).

	morph	distance	voronoi	feature-based	nostro metodo
Accuracy <sub>BC</sub>	98.1% $\pm$ 2.4%	96.9% $\pm$ 2.8%	95.0% $\pm$ 2.3%	89.2% $\pm$ 11.8%	90.0% $\pm$ 4.1%
Accuracy <sub>FC</sub>	88.5% $\pm$ 9.2%	88.4% $\pm$ 9.3%	94.8% $\pm$ 5.0%	90.4% $\pm$ 8.0%	90.1% $\pm$ 6.3%

*Tabella 5.1: Valori medi ( $\pm$  deviazione standard) di Accuracy<sub>FC</sub> ed Accuracy<sub>BC</sub>. I risultati delle prime 4 colonne sono stati tratti da [9] e si riferiscono a metodi di segmentazione eseguiti su 20 mappe senza mobilia. I risultati dell'ultima colonna si riferiscono al metodo di segmentazione sviluppato in questo lavoro di tesi applicato sulle medesime 20 mappe.*

In Tabella 5.2 sono riportate le stime medie ( $\pm$  deviazione standard) di Accuracy<sub>FC</sub> ed Accuracy<sub>BC</sub> relative all'applicazione del nostro metodo sulla totalità delle mappe sperimentate in questo lavoro di tesi (30 mappe a griglia e 26 planimetrie mostrate in Appendice A).

	morph	distance	voronoi	feature-based	nostro metodo
Accuracy <sub>BC</sub>	98.1% $\pm$ 2.4%	96.9% $\pm$ 2.8%	95.0% $\pm$ 2.3%	89.2% $\pm$ 11.8%	89.5% $\pm$ 6.2%
Accuracy <sub>FC</sub>	88.5% $\pm$ 9.2%	88.4% $\pm$ 9.3%	94.8% $\pm$ 5.0%	90.4% $\pm$ 8.0%	94.0% $\pm$ 2.2%

*Tabella 5.2: Valori medi ( $\pm$  deviazione standard) di Accuracy<sub>FC</sub> ed Accuracy<sub>BC</sub>. I risultati delle prime 4 colonne sono stati tratti da [9] e si riferiscono a metodi di segmentazione eseguiti su 20 mappe senza mobilia. I risultati dell'ultima colonna si riferiscono invece al nostro metodo di segmentazione eseguito sulla totalità delle mappe sperimentate in questo lavoro di tesi.*

Dai risultati mostrati in Tabella 5.1 e in Tabella 5.2 si osserva che i valori delle metriche relative alla segmentazione sviluppata in questo lavoro di tesi (ultima colonna) sono generalmente alti; ciò indica che la segmentazione sembra adattarsi bene al ground truth. Di particolare interesse è il fatto che in entrambi i casi il valore di Accuracy<sub>FC</sub> sia superiore a quello

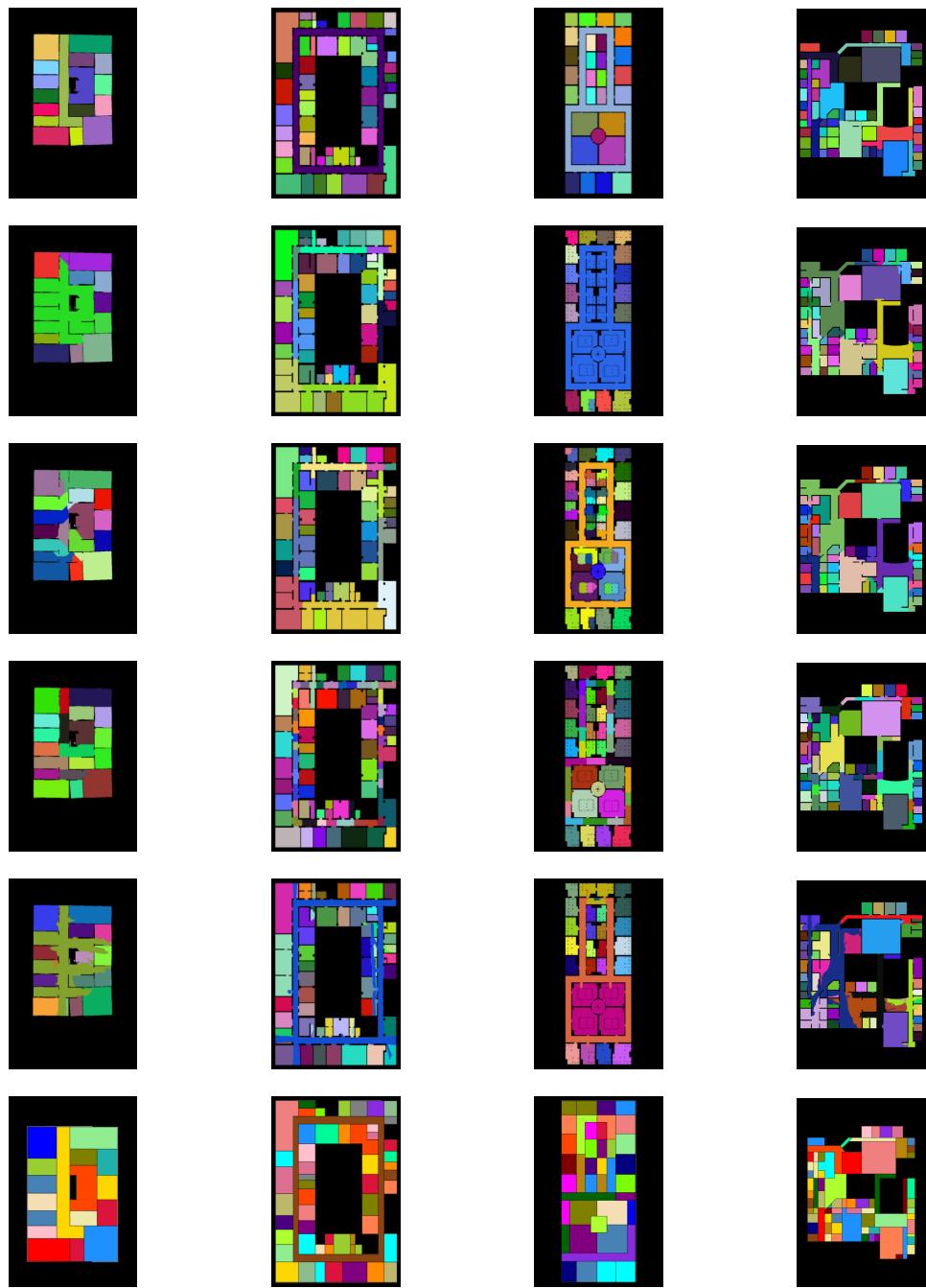
di Accuracy<sub>BC</sub>, cosa che invece non avviene per gli altri metodi di segmentazione (escluso il feature-based). Ciò indica che il metodo tende all'over-segmentazione, come già osservato tramite l'analisi qualitativa precedente (primo, secondo e quarto esempio in Figura 5.17; terzo esempio in Figura 5.18). Al contrario, i metodi di segmentazione morfologici, basati su distance transform e su Voronoi tendono all'under-segmentazione in quanto il loro valore di Accuracy<sub>FC</sub> è inferiore a quello di Accuracy<sub>BC</sub>.

In Figura 5.21 sono riportati alcuni esempi di segmentazione sulle 20 mappe senza mobilia utilizzate in [9]. La prima riga rappresenta l'immagine ground truth in cui ogni stanza è stata colorata manualmente con un colore diverso. Le altre righe dell'immagine consistono nei risultati (tratti da [9]) di segmentazione ottenuti applicando rispettivamente: segmentazione morfologica, basata su distance transform, basata su Voronoi graph e basata su feature. L'ultima riga si riferisce invece alla segmentazione ottenuta utilizzando l'approccio di questo lavoro di tesi.

Un'ulteriore analisi quantitativa di tali risultati è mostrata in Tabella 5.3, in cui sono riportate altre metriche di segmentazione valutate in [9]: tempo di esecuzione della segmentazione, numero di stanze ottenute, area delle stanze e perimetro delle stanze. Le prime 4 colonne sono state tratte da [9] e si riferiscono ai metodi di segmentazione esaminati in tale lavoro e sperimentati sulle 20 mappe senza mobilia. I risultati dell'ultima colonna sono invece relativi al metodo di segmentazione di questa tesi, sperimentato sulle medesime 20 mappe. Analizzando questi ultimi risultati, si osserva che il numero medio delle stanze individuate è superiore rispetto a quello ottenuto dagli altri metodi, mentre le misure medie di area e perimetro risultano inferiori. Questi dati sono coerenti con la tendenza all'over-segmentazione.

	morph	distance	voronoi	feature-based	tesi
runtime [s]	1.6 ± 2.6	1.8 ± 2.7	13.0 ± 15.3	269.3 ± 196.7	22.9 ± 18.2
number of segments	22.8 ± 12.3	24.7 ± 11.7	37.9 ± 20.3	32.6 ± 21.1	38.5 ± 19.2
segment area [m <sup>2</sup> ]	47.9 ± 54.0	43.7 ± 31.8	29.0 ± 24.1	36.6 ± 52.9	28.8 ± 22.7
segment perimeter [m]	36.2 ± 36.6	34.2 ± 21.6	22.7 ± 9.5	36.8 ± 57.4	22.6 ± 10.1

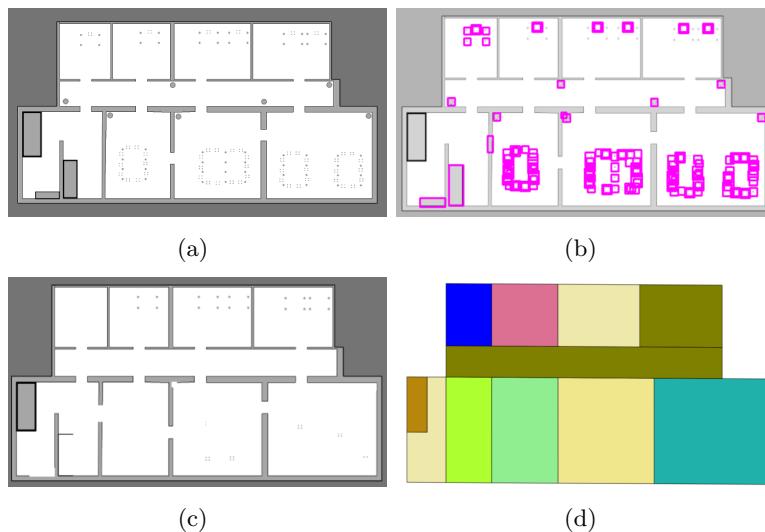
*Tabella 5.3: Valori medi (± deviazione standard) di: tempo di esecuzione della segmentazione, numero di stanze ottenute, area delle stanze, perimetro delle stanze. Le prime 4 colonne sono state tratte da [9] e si riferiscono a metodi di segmentazione eseguiti su 20 mappe senza mobilia. L'ultima colonna si riferisce al metodo di segmentazione sviluppato in questo lavoro di tesi sperimentato sulle medesime 20 mappe.*



*Figura 5.21: Esempi di segmentazioni eseguite su alcune delle 20 mappe senza mobilia utilizzate in [9]. Ad ogni stanza è associato un colore diverso. Le righe consistono in (dall'alto verso il basso): mappa metrica di partenza, segmentazione morfologica, segmentazione basata su distance transform, segmentazione basata su Voronoi graph, segmentazione basata su feature, segmentazione di questo lavoro di tesi.*

### 5.3.3 Mappe con mobilia

In [9], tra le mappe su cui vengono eseguiti i diversi metodi di segmentazione, vi sono 20 mappe caratterizzate da rumore ‘controllato’, corrispondente alla mobilia presente nell’ambiente. Il nostro metodo è stato sperimentato anche su queste mappe, ed un esempio di tale sperimentazione è mostrato in Figura 5.22. In (a) è mostrata la mappa di input, comprensiva della mobilia. In (b) è mostrato il processo di estrazione del rumore (marcato da un contorno viola); questo processo si avvale della stessa tecnica utilizzata in Sezione 3.3.2 per individuare l’informazione testuale. In (c) è mostrata la mappa originale dalla quale è stato eliminato il rumore identificato, colorandone i pixel dello stesso colore dello sfondo (bianco). Su questa sono poi eseguite le operazioni di segmentazione (Canny, Hough, ecc.) che producono il layout delle stanze mostrato in (d).



*Figura 5.22: (a) Mappa di input con mobilia, tratta da [9]; (b) Individuazione del rumore, marcato da contorni viola; (c) Mappa pulita; (d) Layout delle stanze.*

In tale esempio il metodo sembra essere abbastanza accurato, eliminando la quasi totalità del rumore presente nella mappa originale e generando una segmentazione quasi equivalente alla struttura dell’edificio.

In Tabella 5.4 sono riportate le stime medie ( $\pm$  deviazione standard) di Accuracy<sub>FC</sub> ed Accuracy<sub>BC</sub> relative all’applicazione del nostro metodo sulle 20 mappe con mobilia sperimentate in [9].

Dai risultati mostrati in Tabella 5.4 si nota che i valori delle metriche relative al nostro metodo sono leggermente inferiori rispetto a quelli riportati nell’ultima colonna in Tabella 5.1 e Tabella 5.2. Tale minore precisione

	morph	distance	voronoi	feature-based	nostro metodo
Accuracy <sub>BC</sub>	84.6% $\pm$ 7.2%	76.1% $\pm$ 12.3%	86.6% $\pm$ 5.2%	85.1% $\pm$ 7.2%	86.0% $\pm$ 4.7%
Accuracy <sub>FC</sub>	90.5% $\pm$ 8.1%	88.4% $\pm$ 8.5%	94.5% $\pm$ 5.1%	87.1% $\pm$ 14.5%	87.8% $\pm$ 8.4%

*Tabella 5.4: Valori medi ( $\pm$  deviazione standard) di Accuracy<sub>FC</sub> ed Accuracy<sub>BC</sub>. I risultati delle prime 4 colonne sono stati tratti da [9] e si riferiscono a metodi di segmentazione eseguiti su 20 mappe con mobilia. I risultati dell'ultima colonna si riferiscono al metodo di segmentazione sviluppato in questo lavoro di tesi applicato sulle medesime 20 mappe.*

può essere spiegata considerando che il metodo a volte non è in grado di identificare in modo completo il rumore presente nella mappa. Il rumore restante può dunque influire negativamente sull'accuratezza.

Rimane invariata la lieve tendenza all'over-segmentazione (Accuracy<sub>FC</sub>  $>$  Accuracy<sub>BC</sub>), la cui probabile causa è anche in questo caso da imputare alla presenza di rumore residuo relativo alla mobilia dopo la fase di pulitura.

Come sarà discusso nel Capitolo 6, il metodo di segmentazione sviluppato in questo lavoro di tesi sembra diminuire la propria accuratezza se applicato ad ambienti con un elevato livello di rumore, maggiore di quello presente nelle planimetrie con mobilia utilizzate in [9]. Per ottenere anche in questi casi un buon livello di precisione andrebbero introdotti delle ulteriori operazioni all'interno del processo di segmentazione, al fine di sviluppare un metodo più raffinato di rimozione del rumore.

## 5.4 Performance globali di classificazione

Il classificatore utilizzato in questo lavoro di tesi per associare una label a ciascuna stanza del layout è un classificatore extra-trees, presentato in Sezione 3.5.4. Questo è un classificatore di tipo supervisionato, ciò vuol dire che necessita di una fase di addestramento in modo da indurre una funzione tramite la quale classificare nuovi esempi. Come già discusso in Sezione 3.5.5, si è scelto di estrarre i dati di training da un dataset di file in formato *XML* che descrivono planimetrie di edifici reali di tipologia scuola, già utilizzati nei lavori presentati in [61], [59] e [60]. Ciascun dato di training consiste in una stanza ground truth  $r$ , espressa come una coppia  $(V_r, l_r)$ , dove  $V_r$  è un vettore di feature che descrivono la stanza ed  $l_r$  è la sua label corretta. Lo schema di labeling utilizzato in questo lavoro di tesi è stato presentato in Sezione 3.5.3 ed è mostrato in Figura 3.28. Come già spiegato in Sezione 3.5.6, allo scopo di valutare l'impatto di diversi insiemi di label sulla classificazione, si è deciso di eseguire due tipi di classificazioni semantiche:

- **RC:** per la classificazione viene considerato il livello superiore del labeling schema mostrato in Figura 3.28. La label può essere *R* o *C*, etichettando la stanza come *room* o *corridor*.
- **FCES:** viene considerato il livello inferiore dello schema di labeling, associando a ogni stanza un’etichetta semantica tra *F* (*functional room*), *C* (*connection*), *E* (*entrance*), *S* (*service room*).

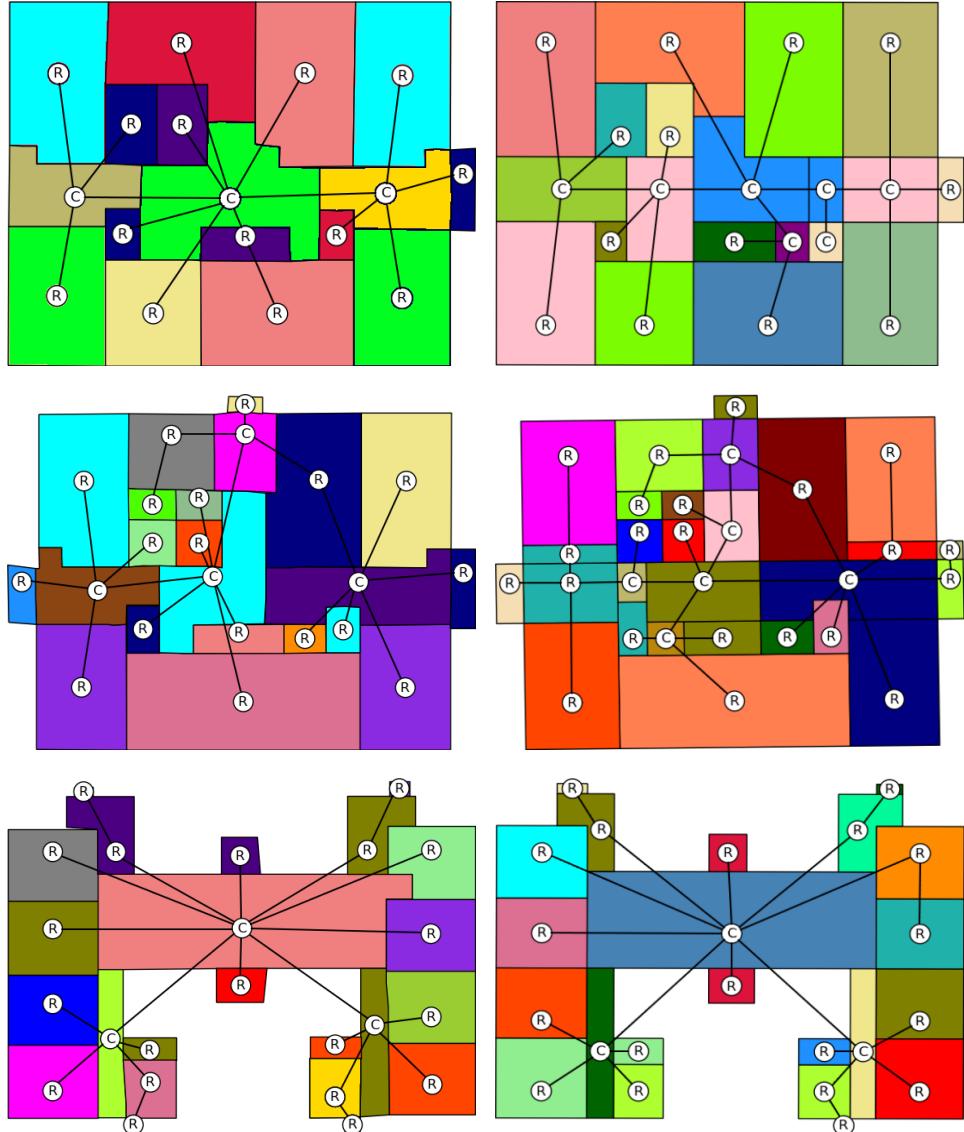
Inoltre, al fine di testare una situazione in cui siano considerate anche informazioni *globali*, oltre a quelle *locali* incluse in  $V_r$ , entrambi i tipi di classificazioni sono stati sperimentati considerando o meno le due feature relative alla centralità (*closeness* e *betweenness*), presentate in Sezione 3.5.2. Tali feature descrivono il ruolo della stanza nella struttura dell’edificio, distinguendosi dalle altre feature incluse in  $V_r$  che si limitano ad una descrizione geometrica e locale della stanza  $r$ , ignorando il suo ruolo nell’ambiente in relazione alle altre stanze.

#### 5.4.1 Valutazione qualitativa

Di seguito sono mostrati degli esempi significativi di mappe semantiche scelti tra la totalità di mappe generate in questo lavoro di tesi. Sono riportate le stanze ground truth e le corrispondenti stanze segmentate, rappresentando la label di ciascuna stanza all’interno di essa. I nodi e gli archi costituiscono il grafo topologico, presentato in Sezione 3.4. La classificazione di tipo RC è mostrata in Figura 5.23 senza centralità e in Figura 5.24 con centralità; la classificazione di tipo FCES è mostrata in Figura 5.25 senza centralità e in Figura 5.26 con centralità.

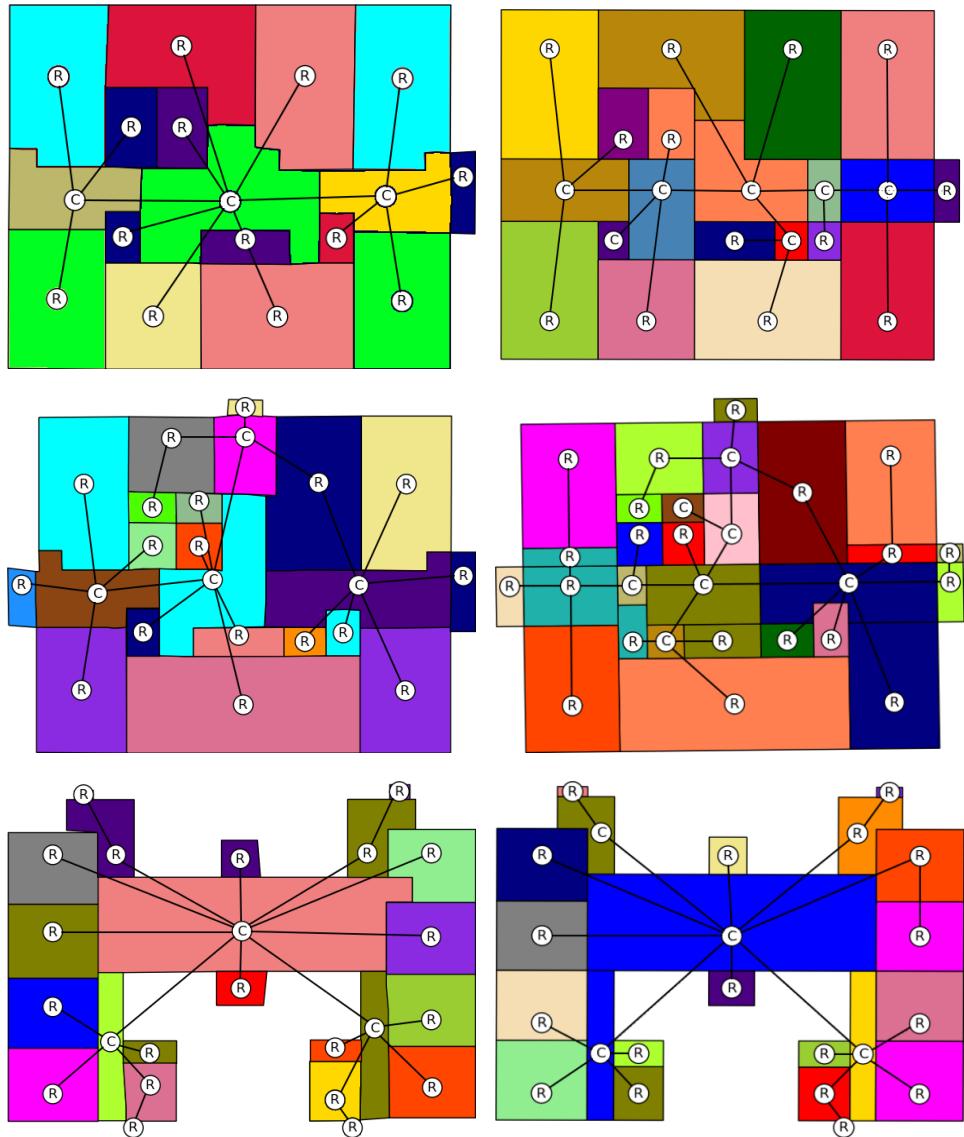
Valutando qualitativamente i risultati riportati nelle figure si osserva una maggiore accuratezza della classificazione semantica di tipo RC rispetto a quella di tipo FCES. Questo dato può essere spiegato considerando la maggiore facilità nell’esecuzione di una semplice distinzione tra stanza e corridoio (classificazione RC) piuttosto che nell’esecuzione di una classificazione più specifica distinguendo tra *connection*, *entrance*, *functional room* o *service room* (classificazione FCES).

Una seconda osservazione riguarda l’influenza della nozione di centralità sui risultati. La centralità è da intendersi come una misura dell’importanza di ogni stanza all’interno della struttura dell’edificio. Attraverso le due feature di centralità (*closeness* e *betweenness*) viene dunque aggiunta un’informazione globale a quelle locali apportate dalle altre feature, descrivendo il ruolo della stanza nella struttura dell’edificio. È interessante notare che, sia per la classificazione RC che per la classificazione FCES, l’utilizzo di co-



*Figura 5.23: Esempi di classificazione RC senza centralità. A sinistra sono riportate le stanze ground truth, a destra le stanze segmentate con il nostro metodo. Ad ogni stanza è assegnato un diverso colore. Viene riportato il corrispondente grafo topologico, includendo le label corrette (sinistra) o predette (destra) all'interno di ciascun nodo corrispondente ad una stanza.*

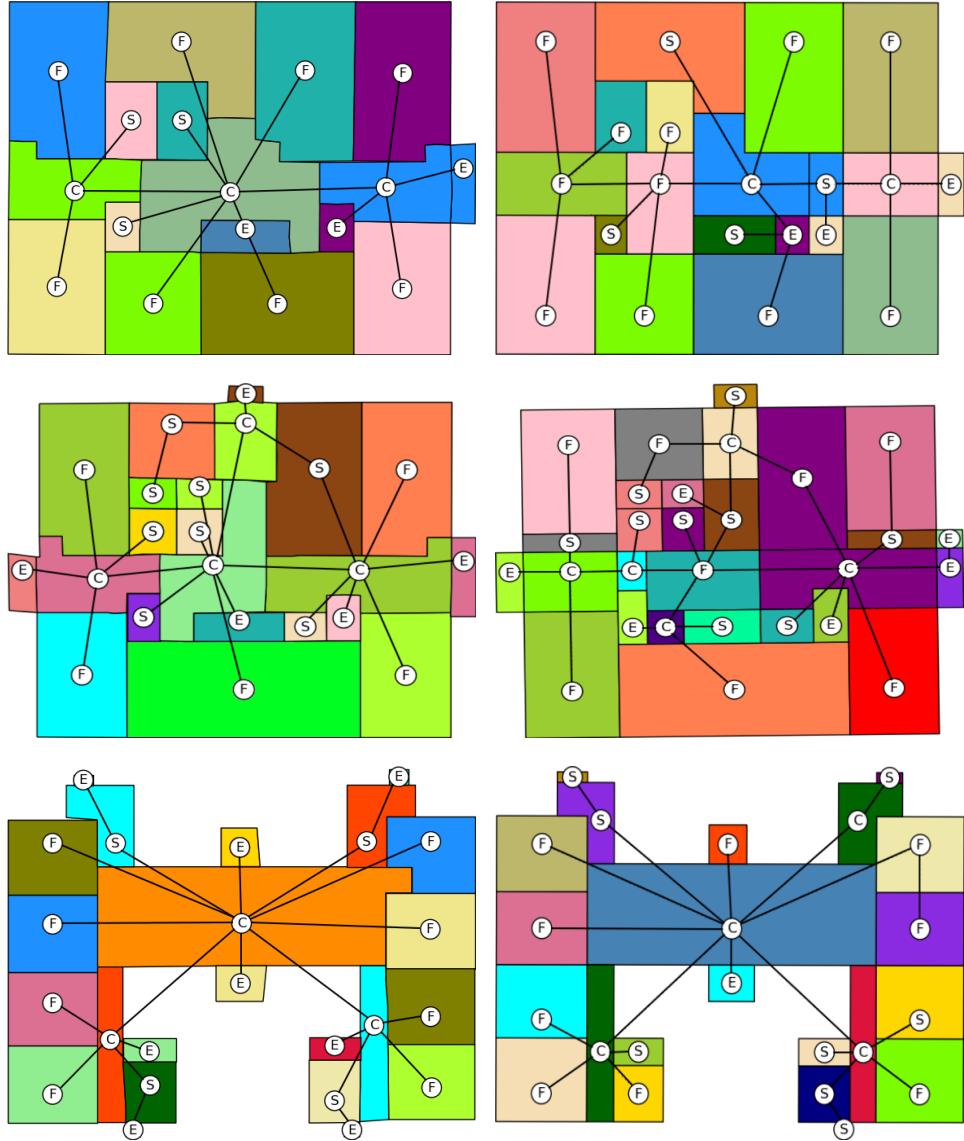
noscenza globale sotto forma di feature di centralità incrementa, seppur lievemente, l'accuratezza della classificazione, rispetto alla sola considerazione di feature locali per descrivere le stanze.



*Figura 5.24: Esempi di classificazione RC con centralità. A sinistra sono riportate le stanze ground truth, a destra le stanze segmentate con il nostro metodo. A ciascuna stanza è assegnato un diverso colore. Viene riportato il corrispondente grafo topologico, includendo le label corrette (sinistra) o predette (destra) all'interno di ciascun nodo rappresentante una stanza.*

#### 5.4.2 Valutazione quantitativa

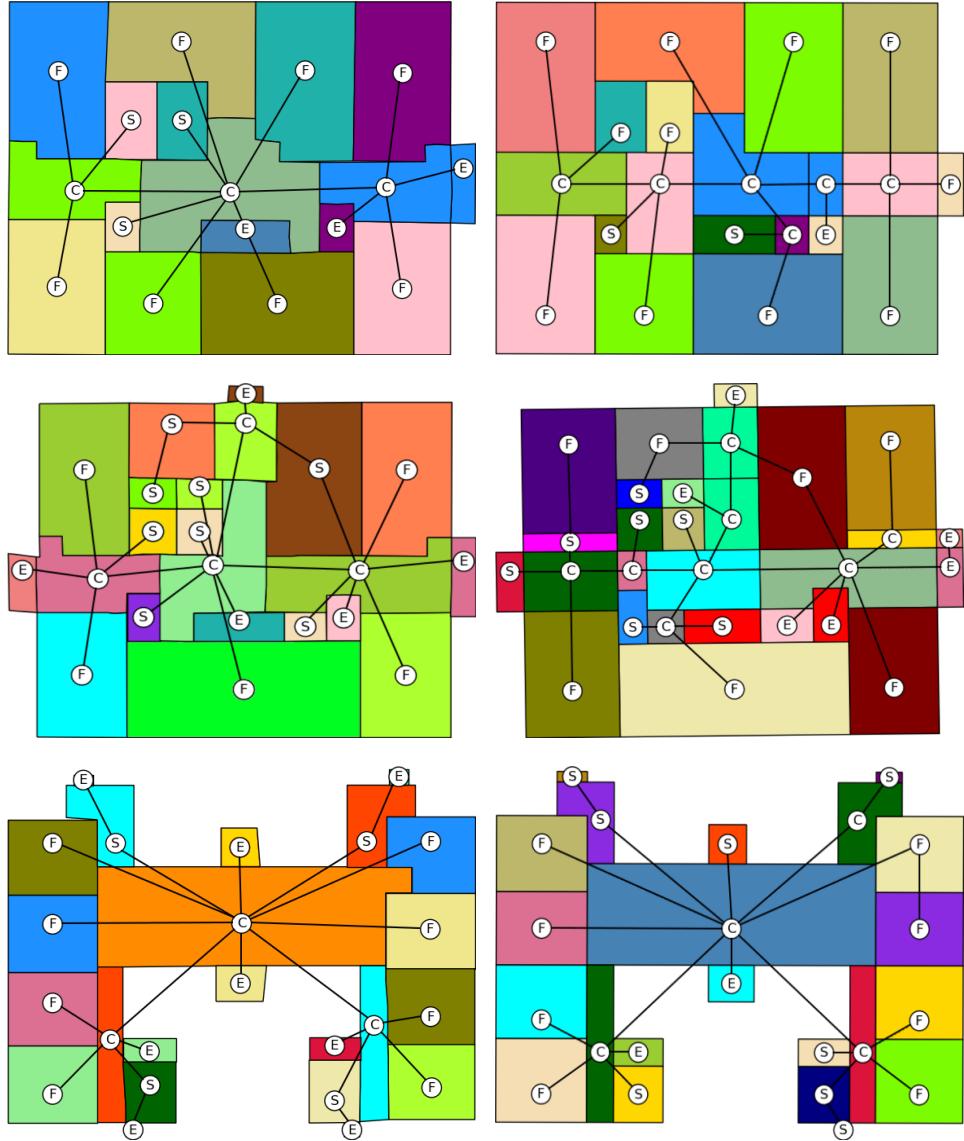
Allo scopo di stimare in modo quantitativo la correttezza della classificazione semantica può essere utilizzata la medesima nozione di corrispondenza tra stanze ground truth e stanze segmentate definita in precedenza nell'ambito



*Figura 5.25: Esempi di classificazione FCES senza centralità. A sinistra sono riportate le stanze ground truth, a destra le stanze segmentate con il nostro metodo. A ciascuna stanza è assegnato un diverso colore. Viene riportato il corrispondente grafo topologico, includendo le label corrette (sinistra) o predette (destra) all'interno di ciascun nodo rappresentante una stanza.*

della valutazione quantitativa delle performance globali di segmentazione.

La corrispondenza tra l'insieme di stanze segmentate ed etichettate dal classificatore e l'insieme di stanze ground truth etichettate correttamente viene quindi definita utilizzando le nozioni di forward e backward coverage,



*Figura 5.26: Esempi di classificazione FCES con centralità. A sinistra sono riportate le stanze ground truth, a destra le stanze segmentate con il nostro metodo. A ciascuna stanza è assegnato un diverso colore. Viene riportato il corrispondente grafo topologico, includendo le label corrette (sinistra) o predette (destra) all'interno di ciascun nodo rappresentante una stanza.*

congiuntamente alle relative funzioni indicatrici  $\mathbb{1}_{FC}(x, y)$  e  $\mathbb{1}_{BC}(x, y)$ .

Un'altra importante funzione indicatrice che può essere usata per valutare quantitativamente la correttezza della classificazione è la *label-equality*

$\mathbb{1}_{EQ}(x, y)$ , che date due stanze  $x$  e  $y$  indica se esse possiedono la stessa label:

$$\mathbb{1}_{EQ}(x, y) = \begin{cases} 1, & \text{se } x \text{ e } y \text{ hanno stessa label} \\ 0, & \text{altrimenti} \end{cases}$$

In base alle due nozioni di *forward coverage* e *backward coverage* tra le stanze segmentate e le stanze ground truth, possono essere calcolate due diverse stime dell'accuratezza della classificazione semantica:

**SemanticAccuracy<sub>FC</sub>**: indica l'accuratezza della corrispondenza semantica tra le stanze segmentate e le stanze ground truth da cui sono *forward covered*.

$$SemanticAccuracy_{FC} = \frac{\sum_{x \in S, y \in G} \mathbb{1}_{FC}(x, y) * \mathbb{1}_{EQ}(x, y)}{\sum_{x \in S, y \in G} \mathbb{1}_{FC}(x, y)} \quad (5.3)$$

dove  $S$  è l'insieme delle stanze segmentate e  $G$  è l'insieme delle stanze ground truth.

**SemanticAccuracy<sub>BC</sub>**: indica l'accuratezza della corrispondenza semantica tra le stanze ground truth e le stanze segmentate da cui sono *backward covered*.

$$SemanticAccuracy_{BC} = \frac{\sum_{x \in S, y \in G} \mathbb{1}_{BC}(x, y) * \mathbb{1}_{EQ}(x, y)}{\sum_{x \in S, y \in G} \mathbb{1}_{BC}(x, y)} \quad (5.4)$$

In Tabella 5.5 sono riportati i valori medi ( $\pm$  deviazione standard) di SemanticAccuracy<sub>FC</sub> e SemanticAccuracy<sub>BC</sub> stimati sulle classificazioni *RC* ed *FCES* eseguite considerando o meno le feature di *centralità*. I risultati derivano dalla sperimentazione sulla totalità delle mappe utilizzate in questo lavoro di tesi (30 mappe a griglia e 26 planimetrie mostrate in Appendice A).

	centralità	SemanticAccuracy <sub>FC</sub>	SemanticAccuracy <sub>BC</sub>
RC	no	94.4% $\pm$ 1.4%	93.5% $\pm$ 1.9%
RC	si	97.1% $\pm$ 2.1%	96.5% $\pm$ 2.5%
FCES	no	74.7% $\pm$ 8.7%	74.3% $\pm$ 12.5%
FCES	si	76.1% $\pm$ 9.1%	77.5% $\pm$ 10.5%

Tabella 5.5: Valori di SemanticAccuracy<sub>FC</sub> e SemanticAccuracy<sub>BC</sub> medie ( $\pm$  deviazione standard).

I dati riportati in tabella confermano le osservazioni ricavate dall’analisi qualitativa sui risultati nelle Figure 5.23-5.26. Si notano infatti una maggiore accuratezza della classificazione semantica di tipo RC rispetto a quella di tipo FCES ed un lieve incremento dell’accuratezza dovuto alla considerazione delle feature di centralità.

Tuttavia i valori di  $\text{SemanticAccuracy}_{FC}$  e  $\text{SemanticAccuracy}_{BC}$  offrono una visione parziale della correttezza della classificazione semantica, omettendo di riportare alcune informazioni come ad esempio quanto in media una label  $l_1$  sia erroneamente predetta come un’altra label  $l_2$ .

Al fine di riportare una più approfondita analisi quantitativa dell’accuratezza della classificazione, possono essere utilizzate le *matrici di confusione*. Queste sono delle matrici in grado di mostrare le relazioni che intercorrono tra le label predette e le label ground truth. Sono distinte in base alla corrispondenza rappresentata (forward coverage o backward coverage).

### Matrici di confusione forward coverage

Ogni riga  $i$  della matrice corrisponde a una label predetta  $l_i$ , associata alle stanze segmentate. Le colonne  $j$  corrispondono invece alle label ground truth  $l_j$ . Il valore della cella  $(i, j)$  indica quante stanze segmentate con label predetta  $l_i$  sono forward covered da stanze ground truth la cui label è  $l_j$ .

Sono di seguito mostrate le matrici di confusione forward coverage relative a:

- Classificazione RC ignorando le feature di centralità (Tabella 5.6).
- Classificazione RC considerando le feature di centralità (Tabella 5.7).
- Classificazione FCES ignorando le feature di centralità (Tabella 5.8).
- Classificazione FCES considerando le feature di centralità (Tabella 5.9).

In prima istanza i risultati evidenziano ciò che già è stato osservato attraverso l’analisi qualitativa sui risultati nelle Figure 5.23-5.26 e attraverso l’analisi quantitativa in Tabella 5.5: maggiore accuratezza della classificazione RC rispetto alla FCES ed incremento dell’accuratezza legato alla considerazione della centralità.

L’informazione aggiuntiva riportata dalle matrici di confusione forward coverage riguarda le relazioni che intercorrono tra le label predette e le label ground truth. Ad esempio si nota che per la classificazione FCES (con e

	R	C
R	98.7% $\pm$ 1.8%	1.3% $\pm$ 1.8%
C	9.3% $\pm$ 6.5%	90.7% $\pm$ 6.5%

Tabella 5.6: Matrice di confusione forward coverage relativa a classificazione RC senza centralità.

	R	C
R	98.2% $\pm$ 1.3%	1.8% $\pm$ 1.3%
C	4.7% $\pm$ 6.6%	95.3% $\pm$ 6.6%

Tabella 5.7: Matrice di confusione forward coverage relativa a classificazione RC con centralità.

	F	C	E	S
F	81.7% $\pm$ 12.9%	0.9% $\pm$ 2.5%	2.1% $\pm$ 4.2%	15.3% $\pm$ 11.5%
C	2.9% $\pm$ 5.65%	87.7% $\pm$ 10.2%	1.1% $\pm$ 4.8%	8.3% $\pm$ 11.7%
E	1.3% $\pm$ 2.7%	1.2% $\pm$ 3.1%	70.4% $\pm$ 15.5%	27.1% $\pm$ 13.5%
S	9.3% $\pm$ 6.6%	0.8% $\pm$ 1.1%	34.5% $\pm$ 10.3%	55.4% $\pm$ 3.7%

Tabella 5.8: Matrice di confusione forward coverage relativa a classificazione FCES senza centralità.

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

Tabella 5.9: Matrice di confusione forward coverage relativa a classificazione FCES con centralità.

senza feature di centralità) vi è una buona accuratezza di classificazione delle stanze *functional room* e *connection* (superiore all’80%), ovvero le stanze segmentate classificate come F o C corrispondono con buona probabilità a stanze ground truth con stessa label. Tale probabilità diminuisce per le stanze classificate come E o S. Per quanto riguarda la label E, si nota che la maggior parte degli errori sono dovuti ad una corrispondenza con stanze ground truth la cui label è S. In modo analogo, analizzando la label S si osserva che la maggior parte degli errori sono causati da corrispondenza con stanze ground truth con label E. Questo dato può essere spiegato analizzando la Figura 5.27, la quale mostra i diversi tipi di stanze associate alle label considerate in questo lavoro di tesi, relativamente ad edifici di tipo

scolastico. La causa degli errori di classificazione potrebbe infatti consistere nella somiglianza, relativamente a proprietà geometriche, tra alcune delle stanze di tipo *service* (ripostigli, bagni, piccole segreterie) e le stanze di tipo *entrance* (ascensori, scale, salette d'ingresso).

Una possibile soluzione a tale imprecisione potrebbe consistere nell'utilizzo di una telecamera tramite cui il robot possa ottenere informazioni visive relative alle porzioni di ambiente, come discusso nel Capitolo 6. Queste informazioni aggiuntive arricchirebbero la descrizione delle stanze, rendendo di conseguenza più accurata la classificazione.

	FUNCTIONAL	classroom	support	teachers' room	laboratory	
ROOMS	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		

Figura 5.27: Schema di labeling relativo a edifici scolastici utilizzato in questo lavoro di tesi.

### Matrici di confusione backward coverage

Ogni riga  $i$  della matrice corrisponde a una label ground truth  $l_i$ . Le colonne  $j$  corrispondono invece alle label predette  $l_j$ , associate alle stanze segmentate. Il valore della cella  $(i, j)$  indica quante stanze ground truth con label  $l_i$  sono backward covered da stanze segmentate la cui label predetta è  $l_j$ .

Sono di seguito mostrate le matrici di confusione backward coverage relative a:

- Classificazione RC ignorando le feature di centralità (Tabella 5.10).
- Classificazione RC considerando le feature di centralità (Tabella 5.11).
- Classificazione FCES ignorando le feature di centralità (Tabella 5.12).
- Classificazione FCES considerando le feature di centralità (Tabella 5.13).

	R	C
R	92.4% $\pm$ 1.8%	7.6% $\pm$ 1.8%
C	8.4% $\pm$ 11.7%	91.6% $\pm$ 11.7%

Tabella 5.10: Matrice di confusione backward coverage relativa a classificazione RC senza centralità.

	R	C
R	95.9% $\pm$ 2.9%	4.1% $\pm$ 2.9%
C	7.4% $\pm$ 11.7%	92.6% $\pm$ 11.7%

Tabella 5.11: Matrice di confusione backward coverage relativa a classificazione RC con centralità.

	F	C	E	S
F	95.8% $\pm$ 5.8%	1% $\pm$ 2.2%	0.8% $\pm$ 1.3%	2.4% $\pm$ 3.3%
C	17.6% $\pm$ 12.1%	80.3% $\pm$ 14.3%	0.9% $\pm$ 1.3%	1.2% $\pm$ 2.2%
E	5.2% $\pm$ 2.9%	4.2% $\pm$ 5.5%	50.3% $\pm$ 18.1%	40.3% $\pm$ 13.5%
S	30.3% $\pm$ 12.2%	2% $\pm$ 3.2%	15.1% $\pm$ 11.7%	52.6% $\pm$ 17.2%

Tabella 5.12: Matrice di confusione backward coverage relativa a classificazione FCES senza centralità.

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

Tabella 5.13: Matrice di confusione backward coverage relativa a classificazione FCES con centralità.

Anche in questo caso i risultati evidenziano ciò che già è stato osservato attraverso le precedenti valutazioni, ovvero maggiore accuratezza della classificazione RC rispetto alla FCES ed incremento dell'accuratezza legato alla considerazione della centralità.

Si osserva poi che per la classificazione FCES l'accuratezza di classificazione delle stanze *functional room* e *connection* è  $> 80\%$ , di conseguenza le stanze ground truth la cui label è F o C corrispondono con buona probabilità a stanze segmentate a cui il classificatore ha associato la stessa label. Analogamente alla matrice di confusione forward coverage, la probabilità di una corretta corrispondenza diminuisce radicalmente per le stanze ground truth la cui label è E o S. Per quanto riguarda la label E, si nota che la maggior

parte degli errori sono dovuti ad una corrispondenza con stanze segmentate che sono state classificate come S. Per quanto invece riguarda la label S, si osserva che la maggior parte degli errori sono causati da corrispondenza con stanze ground truth con label F. Anche in questo caso i dati possono essere spiegati analizzando lo schema di labeling in Figura 5.27 e considerando le somiglianze, relativamente a proprietà geometriche, che possono intercorrere tra:

- le stanze di tipo *entrance* (ascensori, scale, salette d'ingresso) ed alcune delle stanze di tipo *service* (ripostigli, bagni, piccole segreterie),
- altre stanze di tipo *service* (aule conferenza, librerie, palestre, mense) e le stanze di tipo *functional* (classi, laboratori)

Come già discusso relativamente alla matrice di confusione forward coverage, tali imprecisioni potrebbero essere risolte arricchendo la descrizione delle stanze con dell'informazione visiva, ottenuta per mezzo di una telecamera.



## Capitolo 6

# Conclusioni e direzioni future di ricerca

In questo lavoro di tesi è stato sviluppato un sistema di mapping multilivello per edifici. Il sistema integra in un unico schema diversi tipi di mappe: mappa metrica, layout delle stanze, grafo topologico e mappa semantica. Le mappe sono collegate tra loro, in quanto vengono ottenute sequenzialmente a livelli di astrazione crescenti, elaborando l'una l'output dell'altra al fine di estrarre ulteriori informazioni.

Il punto di partenza consiste nella mappa metrica, la quale fornisce informazioni di carattere metrico riguardanti l'occupazione dell'ambiente. Può provenire sia da un robot sia da una mappa di evacuazione.

Il livello di conoscenza della mappa metrica viene raffinato dal layer successivo, corrispondente al layout delle stanze. Questo approccio di rappresentazione elabora le informazioni metriche in modo da ottenere un modello che mostri la disposizione e suddivisione delle stanze che compongono l'edificio. A partire dagli elementi geometrici che rappresentano le pareti dell'edificio viene svolta una sequenza di operazioni il cui risultato finale consiste in una segmentazione della mappa metrica in stanze.

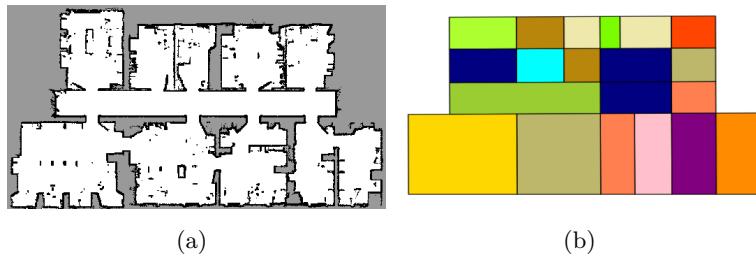
Da questa segmentazione viene ricavato il grafo topologico, che mostra come le stanze sono connesse tra loro tramite le porte. La rappresentazione dell'ambiente assume la struttura di un grafo in cui i nodi rappresentano le stanze e gli archi le connessioni dirette tra di esse.

Infine l'ultimo layer esegue una classificazione semantica delle stanze, associando a ciascun nodo del grafo topologico una label che ne indica la funzione. Come classificatore viene utilizzato extra-trees [32], un metodo di classificazione supervisionato che viene addestrato su un dataset di edifici reali.

Il sistema multilivello è stato testato utilizzando planimetrie note a priori e mappe a griglia ottenute simulando l'esplorazione di edifici. Nel Capitolo 5 è eseguita una valutazione qualitativa e quantitativa dei risultati ottenuti, riportati poi esaustivamente nell'Appendice A.

In conclusione, vengono presentati alcuni dei possibili sviluppi futuri del lavoro di tesi proposto:

- **Utilizzo di dati 2D da robot reali:** il sistema può essere perfezionato per l'utilizzo di dati provenienti da esplorazioni eseguite da robot reali. In tali situazioni la mappa a griglia ottenuta risulta essere più rumorosa ed imprecisa di quelle ottenute tramite esplorazioni simulate ed utilizzate in questo lavoro di tesi, a causa di mobili, altri oggetti presenti nell'ambiente reale ed errori nei sensori. Esempi di queste mappe sono comprese nel Radish Repository [40] e nel dataset di Oscar Martinez Mozos [18]. Osservando l'esempio mostrato in Figura 6.1, il nostro metodo sembra non essere sempre accurato nel caso in cui la mappa di input sia caratterizzata da un elevato livello di rumore.



*Figura 6.1: (a) Mappa metrica tratta da [18]; (b) Segmentazione ottenuta applicando il nostro metodo.*

Per poter applicare il nostro metodo a queste mappe sarebbe necessario filtrare le feature salienti in ogni scansione laser, in modo da eliminare il rumore. Questo è un problema comune a tutti i metodi di segmentazione.

- **Utilizzo di dati 3D come input:** possono essere presi come input del sistema dati tridimensionali acquisiti da un robot per mezzo di sensori come telecamere o scanner laser. Questi dati possono ad esempio consistere in point-cloud 3D da cui poi è possibile recuperare informazioni relative ai muri, al pavimento e al soffitto, rappresentati come piani geometrici, che delimitano l'ambiente.

In [3] viene presentato un lavoro il cui scopo è ricostruire la struttura di un edificio e la sua suddivisione in spazi separati partendo da un dataset di grandi dimensioni composto da punti con coordinate 3D e relativi valori RGB. Per identificare gli elementi che separano le stanze (i muri) non vengono generate delle superfici piane che approssimano il point-cloud, bensì vengono analizzati gli spazi vuoti. Un muro è quindi rappresentato come uno spazio vuoto compreso tra due margini paralleli del point-cloud.

Un altro lavoro il cui input consiste in dati tridimensionali è quello presentato in [29], il cui input è una sequenza di immagini acquisite da un robot dotato di camera monoculare. Tale sequenza è preprocessata in modo da ottenere un point-cloud 3D, sul quale è poi applicato un algoritmo che adatta delle superfici piane ai punti in modo da ottenere i piani corrispondenti agli elementi strutturali dell'edificio, come muri, pavimento e soffitto.

Un lavoro analogo è quello presentato in [68]. L'approccio è in grado di ricostruire la struttura di un edificio partendo da un insieme di scansioni laser tridimensionali. Il relativo point-cloud 3D viene elaborato allo scopo di estrarre un insieme di superfici piane corrispondenti a muri candidati. Vengono in prima istanza identificate regioni piane nel point-cloud 3D. Al fine di conservare solo superfici che corrispondano potenzialmente a muri candidati vengono selezionate esclusivamente le regioni classificate come verticali. Successivamente i muri sono proiettati sul piano del pavimento e la costruzione del modello dell'edificio procederà in uno spazio bidimensionale.

Integrare parti di questi metodi nel nostro lavoro migliorerebbe la segmentazione e il riconoscimento del layout, specialmente nel caso di mappe 2D molto rumorose.

- **Integrazione di informazioni semantiche più ricche:** possono essere raccolte ed utilizzate informazioni semantiche più raffinate relative all'ambiente, allo scopo di incrementare le performance di classificazione. Delle possibili soluzioni per questo task sono:

1. **Utilizzo di telecamere:** in questo modo possono essere raccolte informazioni riguardanti l'aspetto dell'ambiente, analogamente a quanto fatto in [83], dove vengono identificate 7 diverse tipologie di stanze in base al loro aspetto visivo: anticamera, bagno, corridoio, cucina, laboratorio, sala riunioni e ufficio.

2. **Riconoscimento di oggetti:** possono essere riconosciuti gli oggetti presenti nell'ambiente in modo da arricchire il contenuto informativo riguardante le stanze dell'edificio. Ad esempio in [83] ad ogni stanza è associata una descrizione degli oggetti individuati in essa.
3. **Utilizzo di ontologia a più livelli:** in lavori come [83] e [118], allo scopo di migliorare la comprensione dell'ambiente, viene utilizzata una ontologia a più livelli sotto forma di mappa concettuale. Essa definisce categorie astratte per stanze ed oggetti, e mostra come queste categorie sono collegate. L'informazione estratta dai dati sensoriali viene rappresentata come token che instanziano i concetti astratti. Un esempio di mappa concettuale tratta da [118] è mostrata in Figura 6.2. Le informazioni ricavate dal metodo proposto in questo lavoro di tesi (riconoscimento di porte, pareti, simboli e informazione testuale) possono essere integrate all'interno di una ontologia a più livelli, allo scopo di sfruttarle per meglio comprendere la struttura dell'edificio e le relazioni tra i vari elementi che lo compongono.

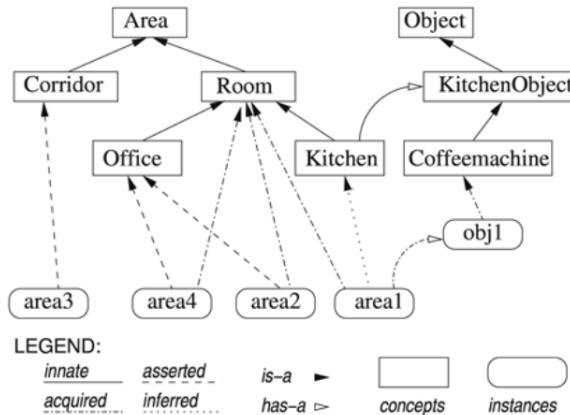
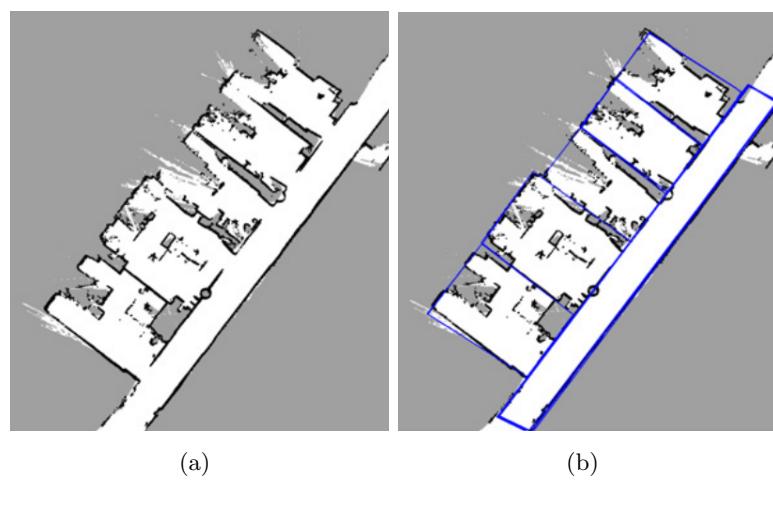


Figura 6.2: Esempio di mappa concettuale in [118]. La mappa combina diversi tipi di conoscenza sfruttando un'ontologia a più livelli.

- **Predizione di layout di porzioni parzialmente esplorate:** possono essere eseguite predizioni più accurate sui layout di porzioni di ambiente di cui il robot non ha effettuato un'esplorazione completa, come avviene ad esempio in [56], lavoro da cui è tratta la Figura 6.3: in (a) è mostrata la mappa di input in cui sono presenti stanze parziali; in (b) sono mostrati i layout predetti, evidenziati in blu.



*Figura 6.3: Esempio di predizione di layout per ambienti parziali, tratto da [56]: (a) Mappa in input; (b) Layout predetto (in blu).*



# Bibliografia

- [1] S. Ahmed, M. Liwicki, M. Weber e A. Dengel. «Automatic Room Detection and Room Labeling from Architectural Floor Plans». In: *Proceedings of the 10th IAPR International Workshop on Document Analysis Systems*. 2012, pp. 339–343.
- [2] P. Althaus e H. Christensen. «Behavior coordination in structured environments». In: *Advanced Robotics* 17.7 (2003), pp. 657–674.
- [3] I. Armeni, O. Sener, A. R. Zamir, H. Jiang, I. Brilakis, M Fischer e S. Savarese. «3D Semantic Parsing of Large-Scale Indoor Spaces». In: *Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition*. 2016, pp. 52–59.
- [4] F. Aurenhammer. «Voronoi Diagrams - a Survey of a Fundamental Geometric Data Structure». In: *Association for Computing Machinery Surveys* 23.3 (1991), pp. 345–405.
- [5] A. Aydemir, A. Pronobis, M. Gobelbecker e P. Jensfelt. «Active Visual Object Search in Unknown Environments Using Uncertain Semantics». In: *IEEE Transactions on Robotics* 29.4 (2013), pp. 986–1002.
- [6] H. Bay, T. Tuytelaars e L. Van Gool. «SURF: Speeded Up Robust Features». In: *Proceedings of the 9th European Conference on Computer Vision*. 2006, pp. 404–417.
- [7] P. Beeson, N. K. Jong e B. Kuipers. «Towards Autonomous Topological Place Detection Using the Extended Voronoi Graph». In: *Proceedings of the IEEE International Conference on Robotics and Automation*. 2005, pp. 4373–4379.
- [8] M. Betke e L. Gurvits. «Mobile robot localization using landmarks». In: *IEEE Transactions on Robotics and Automation* 13.2 (1997), pp. 251–263.

- [9] R. Bormann, F. Jordan, W. Li, J. Hampp e M. Hagele. «Room Segmentation: Survey, Implementation, and Analysis». In: *Proceedings of the IEEE International Conference on Robotics and Automation*. 2016, pp. 1019–1026.
- [10] B. E. Boser, I. M. Guyon e V. N. Vapnik. «A Training Algorithm for Optimal Margin Classifiers». In: *Proceedings of the Fifth Annual Workshop on Computational Learning Theory*. 1992, pp. 144–152.
- [11] U. Brandes. «A Faster Algorithm for Betweenness Centrality». In: *Journal of Mathematical Sociology* 25 (2001), pp. 163–177.
- [12] E. Brunskill, T. Kollar e N. Roy. «Topological Mapping Using Spectral Clustering and Classification». In: *Proceedings of the International Conference on Intelligent Robots and Systems*. 2007, pp. 3491–3496.
- [13] J. Canny. «A Computational Approach to Edge Detection». In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 8.6 (1986), pp. 679–698.
- [14] R. Capobianco, G. Gemignani, D. D. Bloisi, D. Nardi e L. Iocchi. «Automatic Extraction of Structural Representations of Environments». In: *Intelligent Autonomous Systems 13*. Vol. 302. 2015, pp. 721–733.
- [15] D. Comaniciu e P. Meer. «Mean shift: a robust approach toward feature space analysis». In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 24.5 (2002), pp. 603–619.
- [16] N. Dalal e B. Triggs. «Histograms of oriented gradients for human detection». In: *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. Vol. 1. 2005, pp. 886–893.
- [17] C. Dangalchev. «Residual closeness in networks». In: *Physica A: Statistical Mechanics and its Applications* 365.2 (2006), pp. 556–564.
- [18] Dataset di Oscar Martinez Mozos. [http://webpages.lincoln.ac.uk/omozos/place\\_data\\_sets.html](http://webpages.lincoln.ac.uk/omozos/place_data_sets.html). 2016.
- [19] K. T. Dey e W. Zhao. «Approximating the Medial Axis from the Voronoi Diagram with a ConvergenceGuarantee». In: *Algorithmica* 38.1 (2004), pp. 179–200.
- [20] A. Diosi, G. Taylor e L. Kleeman. «Interactive SLAM using Laser and Advanced Sonar». In: *Proceedings of the IEEE International Conference on Robotics and Automation*. 2005, pp. 1103–1108.

- [21] A. Doucet, N. de Freitas, K. Murphy e S. Russell. «Rao-blackwellised Particle Filtering for Dynamic Bayesian Networks». In: *Proceedings of the Sixteenth Conference on Uncertainty in Artificial Intelligence*. 2000, pp. 176–183.
- [22] S. Ekvall, D. Krägic e P. Jensfelt. «Object Detection and Mapping for Service Robot Tasks». In: *Robotica* 25.2 (2007), pp. 175–187.
- [23] M. Ester, H. P. Kriegel, J. Sander e X. 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*. 1996, pp. 226–231.
- [24] E. Fabrizi, G. Oriolo e G. Ulivi. «Accurate Map Building via Fusion of Laser and Ultrasonic Range Measures». In: *Fuzzy Logic Techniques for Autonomous Vehicle Navigation* (Physica-Verlag HD, 2001), pp. 257–279.
- [25] E. Fabrizi e A. Saffiotti. «Extracting topology-based maps from grid-maps». In: *Proceedings of the IEEE International Conference on Robotics and Automation*. 2000, pp. 2972–2978.
- [26] P. O. Fjällström. «Algorithms for Graph Partitioning: A Survey». In: *Linköping Electronic Articles in Computer and Information Science* 3.10 (1998), pp. 4–37.
- [27] *Free Floor Plans Templates*. <https://www.edrawsoft.com/share-floorplan.php>. [Online; accessed 1-February-2016]. 2016.
- [28] S. Friedman, H. Pasula e D. Fox. «Voronoi Random Fields: Extracting the Topological Structure of Indoor Environments via Place Labeling». In: *Proceedings of the International Joint Conference on Artificial Intelligence*. 2007, pp. 2109–2114.
- [29] A. Furlan, S. D. Miller e S. Savarese D. G. Sorrenti L. Fei-Fei. «Free your Camera: 3D Indoor Scene Understanding from Arbitrary Camera Motion». In: *Proceedings of the British Machine Vision Conference*. 2013, pp. 24.1–24.12.
- [30] A. Garulli, A. Giannitrapani, A. Rossi e A. Vicino. «Simultaneous localization and map building using linear features». In: *Proceedings of the European Conference on Mobile Robots*. 2005, pp. 44–49.
- [31] J. L. Gauvain. «Maximum a posteriori estimation for multivariate Gaussian mixture observations of Markov chains». In: *IEEE Transactions on Speech and Audio Processing* 2.2 (1994), pp. 291–298.

- [32] P. Geurts, D. Ernst e L. Wehenkel. «Extremely randomized trees». In: *Machine Learning* 63.1 (2006), pp. 3–42.
- [33] *GMapping ROS Repository*. [https://github.com/ros-perception/slam\\_gmapping](https://github.com/ros-perception/slam_gmapping). [Online; accessed 1-February-2016]. 2016.
- [34] N. Goerke e S. Braun. «Building Semantic Annotated Maps by Mobile Robots». In: *Proceedings of the Conference Towards Autonomous Robotic Systems*. 2009, pp. 149–156.
- [35] G. Grisetti, C. Stachniss e W. Burgard. «Improved Techniques for Grid Mapping With Rao-Blackwellized Particle Filters». In: *IEEE Transactions on Robotics* 23.1 (2007), pp. 34–46.
- [36] G. Grisetti, C. Stachniss e W. Burgard. «Improving Grid-based SLAM with Rao-Blackwellized Particle Filters by Adaptive Proposals and Selective Resampling». In: *Proceedings of the IEEE International Conference on Robotics and Automation*. 2005, pp. 2432–2437.
- [37] Park H. e Jun C. «A simple and fast algorithm for K-medoids clustering». In: *Expert Systems with Applications* 36.2 (2009), pp. 3336–3341.
- [38] J. Hao e J. B. Orlin. «A Faster Algorithm for Finding the Minimum Cut in a Graph». In: *Proceedings of the Third Annual Symposium on Discrete Algorithms*. 1992, pp. 165–174.
- [39] L. Heras, S. Ahmed, M. Liwicki, E. Valveny e G. Sanchez. «Statistical Segmentation and Structural Recognition for Floor Plan Interpretation». In: *International Journal on Document Analysis and Recognition* 17.3 (2014), pp. 221–237.
- [40] A. Howard e N. Roy. *The robotics data set repository (radish)*. <http://radish.sourceforge.net/>. 2016.
- [41] Bloch I. e Maitre H. «Fuzzy mathematical morphologies: A comparative study». In: *Pattern Recognition* 28.9 (1995), pp. 1341–1387.
- [42] S. Ikehata, H. Yan e Y. Furukawa. «Structured Indoor Modeling». In: *Proceedings of the IEEE International Conference on Computer Vision*. 2015, pp. 1323–1331.
- [43] J. Illingworth e J. Kittler. «A survey of the Hough transform». In: *Computer Vision, Graphics, and Image Processing* 44.1 (1988), pp. 87–116.
- [44] J. H. Jang e K. S. Hong. «Fast line segment grouping method for finding globally more favorable line segments». In: *Pattern Recognition* 35.10 (2002), pp. 2235–2247.

- [45] M. E. Jefferies, W. K. Yeap, L. Smith e D. Ferguson. «Building a map for robot navigation using a theory of cognitive maps». In: *Proceedings of the International Conference on Artificial Intelligence and Application*. 2001, pp. 348–353.
- [46] M. Juliá, A. Gil e O. Reinoso. «A comparison of path planning strategies for autonomous exploration and mapping of unknown environments». In: *Autonomous Robots* 33.4 (2012), pp. 427–444.
- [47] J. M. Keller, M. R. Gray e J. A. Givens. «A fuzzy K-nearest neighbor algorithm». In: *Transactions on Systems, Man, and Cybernetics* 15.4 (1985), pp. 580–585.
- [48] D. M. Kortenkamp. «Cognitive Maps for Mobile Robots: A Representation for Mapping and Navigation». Tesi di dott. Ann Arbor, MI, USA, 1993.
- [49] B. Kuipers, J. Modayil, P. Beeson, M. MacMahon e F. Savelli. «Local Metrical and Global Topological Maps in the Hybrid Spatial Semantic Hierarchy». In: *Proceedings of the IEEE International Conference on Robotics and Automation*. 2004, pp. 4845–4851.
- [50] R. Lakaemper. «Simultaneous multi-line-segment merging for robot mapping using Mean shift clustering». In: *Proceedings of the International Conference on Intelligent Robots and Systems*. 2009, pp. 1654–1660.
- [51] P. Lamon, A. Tapus, E. Glauser, N. Tomatis e R. Siegwart. «Environmental modeling with fingerprint sequences for topological global localization». In: *Proceedings of the International Conference on Intelligent Robots and Systems*. 2003, pp. 3781–3786.
- [52] S. L. Lauritzen e T. S. Richardson. «Chain graph models and their causal interpretations». In: *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 64.3 (2002), pp. 321–348.
- [53] K. L. Laviers. e G. L. Peterson. «Cognitive Robot Mapping with Polyline and an Absolute Space Representation». In: *Proceedings of the IEEE International Conference on Robotics and Automation*. 2004, pp. 3771–3776.
- [54] D. T. Lee e B. J. Schachter. «Two algorithms for constructing a Delaunay triangulation». In: *International Journal of Computer & Information Sciences* 9.3 (1980), pp. 219–242.

- [55] J. J. Leonard e H. F. Durrant-White. «Mobile robot localization by tracking geometric beacons». In: *IEEE Transactions on Robotics and Automation* 7.3 (1991), pp. 376–382.
- [56] Z. Liu e G. Von Wichert. «A generalizable knowledge framework for semantic indoor mapping based on Markov logic networks and data driven MCMC». In: *Future Generation Computer Systems* 36 (2014), pp. 42–56.
- [57] Z. Liu e G. Von Wichert. «Extracting semantic indoor maps from occupancy grids». In: *Robotics and Autonomous Systems* 62.5 (2014), pp. 663–674.
- [58] D. G. Lowe. «Object recognition from local scale-invariant features». In: *Proceedings of the IEEE International Conference on Computer Vision*. 1999, pp. 1150–1157.
- [59] M. Luperto e F. Amigoni. «Exploiting Structural Properties of Buildings Towards General Semantic Mapping Systems». In: *Proceedings of the International Conference on Intelligent Autonomous Systems*. 2014, pp. 375–387.
- [60] M. Luperto, L. D’Emilio e F. Amigoni. «A generative spectral model for semantic mapping of buildings». In: *Proceedings of the International Conference on Intelligent Robots and Systems*. 2015, pp. 4451–4458.
- [61] M. Luperto, A. Quattrini Li e F. Amigoni. «A System for Building Semantic Maps of Indoor Environments Exploiting the Concept of Building Typology». In: *Proceedings of the RoboCup*. 2013, pp. 504–515.
- [62] *Maps, simulation environments, room segmentation software*. <http://wiki.ros.org/iparoomsegmentation>, 2016. [Online; accessed 1-February-2016]. 2016.
- [63] K. Mikolajczyk e C. Schmid. «A performance evaluation of local descriptors». In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 27.10 (2005), pp. 1615–1630.
- [64] T. M. Mitchell. *Machine Learning*. McGraw-Hill, 1997.
- [65] Ó. M. Mozos. *Semantic Labeling of Places with Mobile Robots*. Springer Berlin Heidelberg, 2010.

- [66] Ó. M. Mozos, A. Rottmann, R. Triebel, P. Jensfelt e W. Burgard. «Semantic Labeling of Places using Information Extracted from Laser and Vision Sensor Data». In: *IEEE/RSJ IROS Workshop: From sensors to human spatial concepts*. 2006.
- [67] Ó. M. Mozos, A. Rottmann, R. Triebel, P. Jensfelt e W. Burgard. «Supervised Learning of Topological Maps using Semantic Information Extracted from Range Data». In: *Proceedings of the International Conference on Intelligent Robots and Systems*. 2006, pp. 2772–2777.
- [68] C. Mura, O. Mattausch, A. J. Villanueva, E. Gobbetti e R. Pajarola. «Automatic room detection and reconstruction in cluttered indoor environments with complex room layouts». In: *Computers & Graphics* 44 (2014), pp. 20–32.
- [69] C. Mura, O. Mattausch, A. J. Villanueva, E. Gobbetti e R. Pajarola. «Robust Reconstruction of Interior Building Structures with Multiple Rooms under Clutter and Occlusions». In: *Proceedings of the International Conference on Computer-Aided Design and Computer Graphics (CAD/Graphics)*. 2013, pp. 52–59.
- [70] K. Murphy. «Bayesian Map Learning in Dynamic Environments». In: *Proceedings of the Conference on Neural Information Processing Systems (NIPS)*. 2000, pp. 1015–1021.
- [71] P. F. M. Nacken. «A metric for line segments». In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 15.12 (1993), pp. 1312–1318.
- [72] *Nav2d\_navigator ROS Package Repository*. [https://github.com/skasperski/navigation\\_2d/tree/master/nav2d\\_navigator](https://github.com/skasperski/navigation_2d/tree/master/nav2d_navigator). [Online; accessed 1-February-2016]. 2016.
- [73] *NearestFrontierPlanner.cpp*. [https://github.com/skasperski/navigation\\_2d/blob/master/nav2d\\_exploration/src/NearestFrontierPlanner.cpp](https://github.com/skasperski/navigation_2d/blob/master/nav2d_exploration/src/NearestFrontierPlanner.cpp). [Online; accessed 1-February-2016]. 2016.
- [74] *Networkx Repository*. <https://github.com/networkx/networkx>. [Online; accessed 1-March-2016]. 2016.
- [75] E. Neufert e P. Neufert. *Architects' data*. Wiley-Blackwell, 2012.
- [76] A. Y. Ng, M. I. Jordan e Y. Weiss. «On Spectral Clustering: Analysis and an algorithm». In: *Advances in neural information processing systems*. MIT Press, 2001, pp. 849–856.

- [77] J. Oberlander, K. Uhl, J. M. Zollner e R. Dillmann. «A Region-based SLAM Algorithm Capturing Metric, Topological, and Semantic Properties». In: *Proceedings of the IEEE International Conference on Robotics and Automation*. 2008, pp. 1886–1891.
- [78] S. Ochmann, R. Vock, R. Wessel, M. Tamke e R. Klein. «Automatic Generation of Structural Building Descriptions from 3D Point Cloud Scans». In: *Proceedings of the International Conference on Computer Graphics Theory and Applications*. 2014, pp. 1–8.
- [79] S. Ochmann, R. Vock, R. Wessel, M. Tamke e R. Klein. «Automatic reconstruction of parametric building models from indoor point clouds». In: *Computers & Graphics* 54 (2016), pp. 94–103.
- [80] S. Ochmann, R. Vock, R. Wessel, M. Tamke e R. Klein. «Kinect and RGBD Images: Challenges and Applications». In: *Proceedings of the SIBGRAPI Conference on Graphics, Patterns and Images Tutorials*. 2012, pp. 36–49.
- [81] *OpenCV Repository*. <https://github.com/opencv/opencv>. [Online; accessed 1-March-2016]. 2016.
- [82] G. Oriolo, G. Ulivi e M. Vendittelli. «Fuzzy maps: A new tool for mobile robot perception and planning». In: *Journal of Robotic Systems* 14.3 (1997), pp. 179–197.
- [83] A. Pronobis e P. Jensfelt. «Large-scale Semantic Mapping and Reasoning with Heterogeneous Modalities». In: *Proceedings of the IEEE International Conference on Robotics and Automation*. 2012, pp. 3515–3522.
- [84] A. Pronobis, P. Jensfelt, K. Sjöö, H. Zender, G. J. M. Kruijff, O. M. Mozos e W. Burgard. «Semantic Modelling of Space». In: *Cognitive Systems*. Vol. 8. 2010, pp. 165–221.
- [85] *Pytesseract Repository*. <https://github.com/madmaze/pytesseract>. [Online; accessed 1-March-2016]. 2016.
- [86] M. Richardson e P. Domingos. «Markov logic networks». In: *Machine Learning* 62.1 (2006), pp. 107–136.
- [87] I. Rish. «An empirical study of the naive Bayes classifier». In: *Proceedings of the International Joint Conference on Artificial Intelligence*. 2001, pp. 41–46.
- [88] A. Saffiotti e P. Buschka. «A Virtual Sensor for Room Detection». In: *Proceedings of the International Conference on Intelligent Robots and Systems*. 2002, pp. 637–642.

- [89] A. Saffiotti e E. Fabrizi. «Augmenting Topology-Based Maps with Geometric Information». In: *Robotics and Autonomous Systems* 40.2-3 (2002), pp. 91–97.
- [90] H. Schweitzer, J. W. Bell e F. Wu. «Very Fast Template Matching». In: *Proceedings of the European Conference on Computer Vision*. 2002, pp. 358–372.
- [91] *Scikit-image Repository*. <https://github.com/scikit-image/scikit-image>. [Online; accessed 1-March-2016]. 2016.
- [92] *Scikit-learn Repository*. <https://github.com/scikit-learn/scikit-learn>. [Online; accessed 1-March-2016]. 2016.
- [93] S. G. Shahbandi, B. Astrand e R. Philppsen. «Sensor Based Adaptive Metric-Topological Cell Decomposition Method for Semantic Annotation of Structured Environments». In: *Proceedings of the International Conference on Control Automation Robotics & Vision*. 2014, pp. 1771–1777.
- [94] *Shapely Repository*. <https://github.com/Toblerity/Shapely>. [Online; accessed 1-March-2016]. 2016.
- [95] R. Simmons, D. Apfelbaum, W. Burgard, D. Fox, M. Moors, S. Thrun e H. Younes. «Coordination for Multi-Robot Exploration and Mapping». In: *Proceedings of the National Conference on Artificial Intelligence*. 2000, pp. 852–858.
- [96] K. Sjoo. «Semantic map segmentation using function-based energy maximization». In: *Proceedings of the IEEE International Conference on Robotics and Automation*. 2012, pp. 4066–4073.
- [97] C. Stachniss, O. M. Mozos e W. Burgard. «Speeding-up multi-robot exploration by considering semantic place information». In: *Proceedings of the IEEE International Conference on Robotics and Automation*. IEEE, mag. 2006, pp. 1692–1697.
- [98] *Stage ROS Repository*. <https://github.com/ros-gbp/stage-release>. [Online; accessed 1-February-2016]. 2016.
- [99] *Stage\_ros ROS Package Repository*. [https://github.com/ros-simulation/stage\\_ros](https://github.com/ros-simulation/stage_ros). [Online; accessed 1-February-2016]. 2016.
- [100] M. Stoer e F. Wagner. «A Simple Min-Cut Algorithm». In: *Journal of the Association for Computing Machinery* 44.4 (1997), pp. 585–591.

- [101] A. Tapus e R. Siegwart. «A Cognitive Modeling of Space using Fingerprints of Places for Mobile Robot Navigation». In: *Proceedings of the IEEE International Conference on Robotics and Automation*. 2006, pp. 1188–1193.
- [102] A. Tapus e R. Siegwart. «Incremental Robot Mapping with Fingerprints of Places». In: *Proceedings of the International Conference on Intelligent Robots and Systems*. 2005, pp. 2429–2434.
- [103] A. Tapus, N. Tomatis e R. Siegwart. «Topological Global Localization and Mapping with Fingerprints and Uncertainty». In: *Experimental Robotics IX: The 9th International Symposium on Experimental Robotics* (2006), pp. 99–111.
- [104] S. Theodoridis e K. Koutroumbas. *Pattern Recognition*. Academic Press, 2006.
- [105] S. Thrun. «Learning metric-topological maps for indoor mobile robot navigation». In: *Artificial Intelligence* 99.1 (1998), pp. 21–71.
- [106] S. Thrun e A. Bucken. «Integrating Grid-Based and Topological Maps for Mobile Robot Navigation». In: *Proceedings of the National Conference on Artificial Intelligence*. 1996, pp. 944–950.
- [107] S. Thrun, W. Burgard e D. Fox. *Probabilistic Robotics*. MIT Press, 2005.
- [108] S. Thrun, D. Fox, W. Burgard e F. Dellaert. «Robust Monte Carlo localization for mobile robots». In: *Artificial Intelligence* 128.1–2 (2001), pp. 99–141.
- [109] S. Thrun e J. J. Leonard. «Simultaneous Localization and Mapping». In: *Springer Handbook of Robotics*. 2008, pp. 871–889.
- [110] E. Turner e A. Zakhori. «Floor Plan Generation and Room Labeling of Indoor Environments from Laser Range Data». In: *Proceedings of the International Conference on Computer Graphics Theory and Applications*. 2014, pp. 1–12.
- [111] E. Turner, A. Zakhori e P. Chang. «Fast, Automated, Scalable Generation of Textured 3D Models of Indoor Environments». In: *IEEE Journal of Selected Topics in Signal Processing* 9.3 (2014), pp. 409–421.

- [112] Alexandrov V., Lees M., Krzhizhanovskaya V., Dongarra J., Sloot P.M.A., Andrade G., Ramos G., Madeira D., Sachetto R., Ferreira R. e Rocha L. «2013 International Conference on Computational Science G-DBSCAN: A GPU Accelerated Algorithm for Density-based Clustering». In: *Procedia Computer Science* 18 (2013), pp. 369–378.
- [113] R. Vaughan. «Massively multi-robot simulation in stage». In: *Swarm Intelligence* 2.2 (2008), pp. 189–208.
- [114] L. Vincent e P. Soille. «Watersheds in Digital Spaces: An Efficient Algorithm Based on Immersion Simulations». In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 13.6 (1991), pp. 583–598.
- [115] L. Wehenkel. «On Uncertainty Measures used for Decision Tree Induction». In: *Proceedings of the Conference on Information Processing and Management of Uncertainty in Knowledge-Based Systems*. 1996, pp. 413–418.
- [116] I. H. Witten e E. Frank. *Data Mining: Practical Machine Learning Tools and Techniques*. Morgan Kaufmann, 2005.
- [117] K. M. Wurm, C. Stachniss e W. Burgard. «Coordinated Multi-Robot Exploration using a Segmentation of the Environment». In: *Proceedings of the International Conference on Intelligent Robots and Systems*. 2008, pp. 1160–1165.
- [118] H. Zender, O. M. Mozos, P. Jensfelt, G. J. M. Kruijff e W. Burgard. «Conceptual spatial representations for indoor mobile robots». In: *Robotics and Autonomous Systems* 56.6 (2008), pp. 493–502.
- [119] S. Zhu, R. Zhang e Z. Tu. «Integrating bottom-up/top-down for object recognition by data driven Markov chain Monte Carlo». In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. Vol. 1. 2000, pp. 738–745.
- [120] Z. Zivkovic, B. Bakker e B. Kroese. «Hierarchical Map Building and Planning based on Graph Partitioning». In: *Proceedings of the IEEE International Conference on Robotics and Automation*. 2006, pp. 803–809.

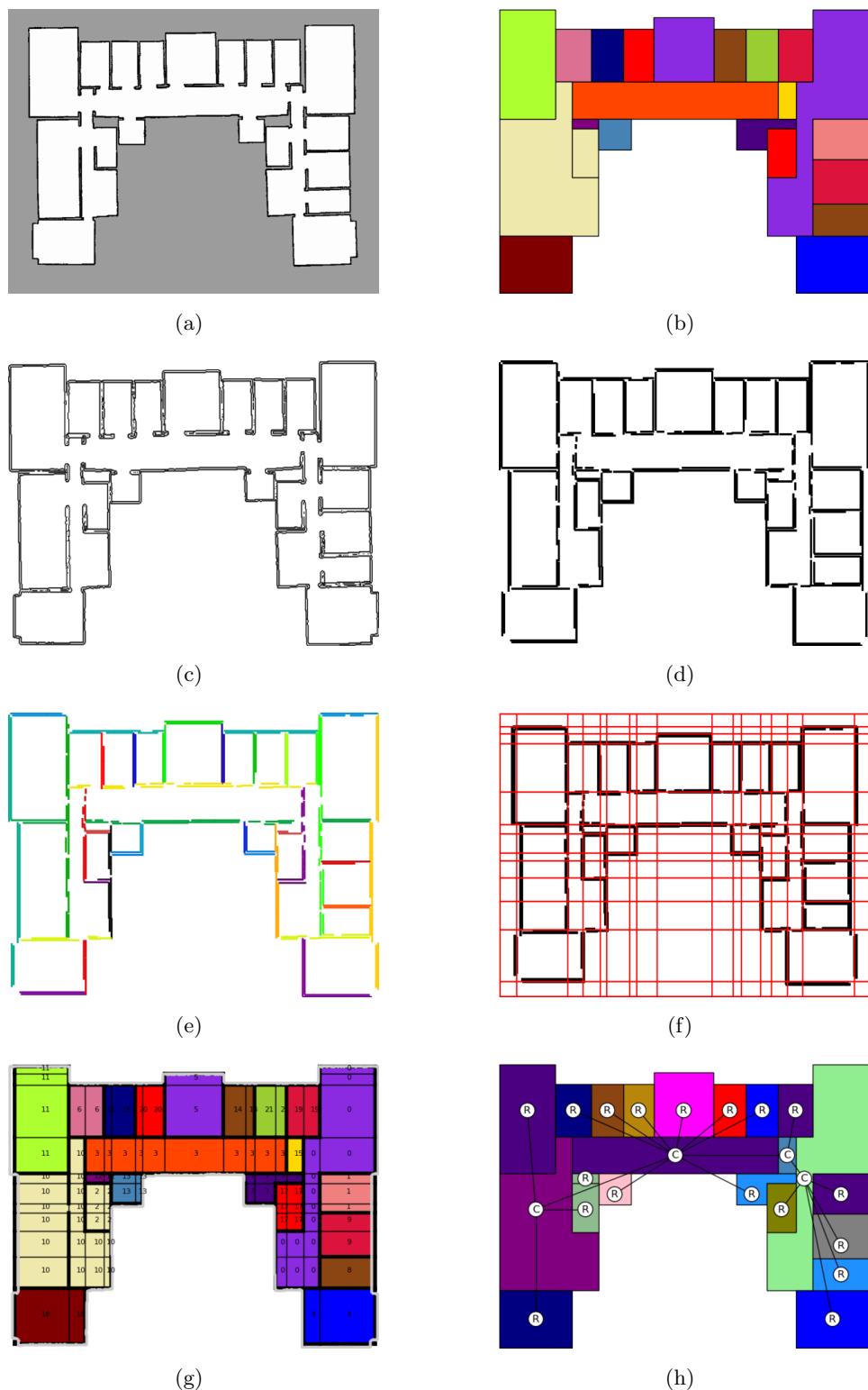


## Appendice A

# Risultati completi

In questa appendice sono mostrati in maniera esaustiva tutti i risultati ottenuti sperimentando l'approccio sviluppato in questo lavoro di tesi su 20 mappe a griglia complete (Figure A.1 - A.20), 10 mappe a griglia parziali (Figure A.21 - A.30), 6 planimetrie tratte da [27] (Figure A.31 - A.36) e 20 planimetrie senza mobilia tratte da [62] (Figure A.37 - A.56). Ciascuna figura si riferisce ad un diverso edificio, e la disposizione dei risultati è mostrata in Figura A.1: (a) mappa metrica di input, (b) layout delle stanze, (c) Canny edge detection, (d) Hough line transform, (e) clustering dei segmenti, (f) insieme di celle, (g) clustering delle celle eseguito con DBSCAN, (h) classificazione RC con centralità (quest'ultimo risultato è mostrato esclusivamente per le mappe a griglia complete).

Infine nelle Figure A.57 e A.58 sono mostrati i risultati tratti da [62] relativi ai metodi di segmentazione confrontati in [9] e sperimentati su 20 mappe senza mobilia.



*Figura A.1: Esempio 1: (a) mappa metrica di input, (b) layout delle stanze, (c) Canny edge detection, (d) Hough line transform, (e) clustering dei segmenti, (f) insieme di celle, (g) clustering delle celle eseguito con DBSCAN, (h) classificazione RC con centralità.*

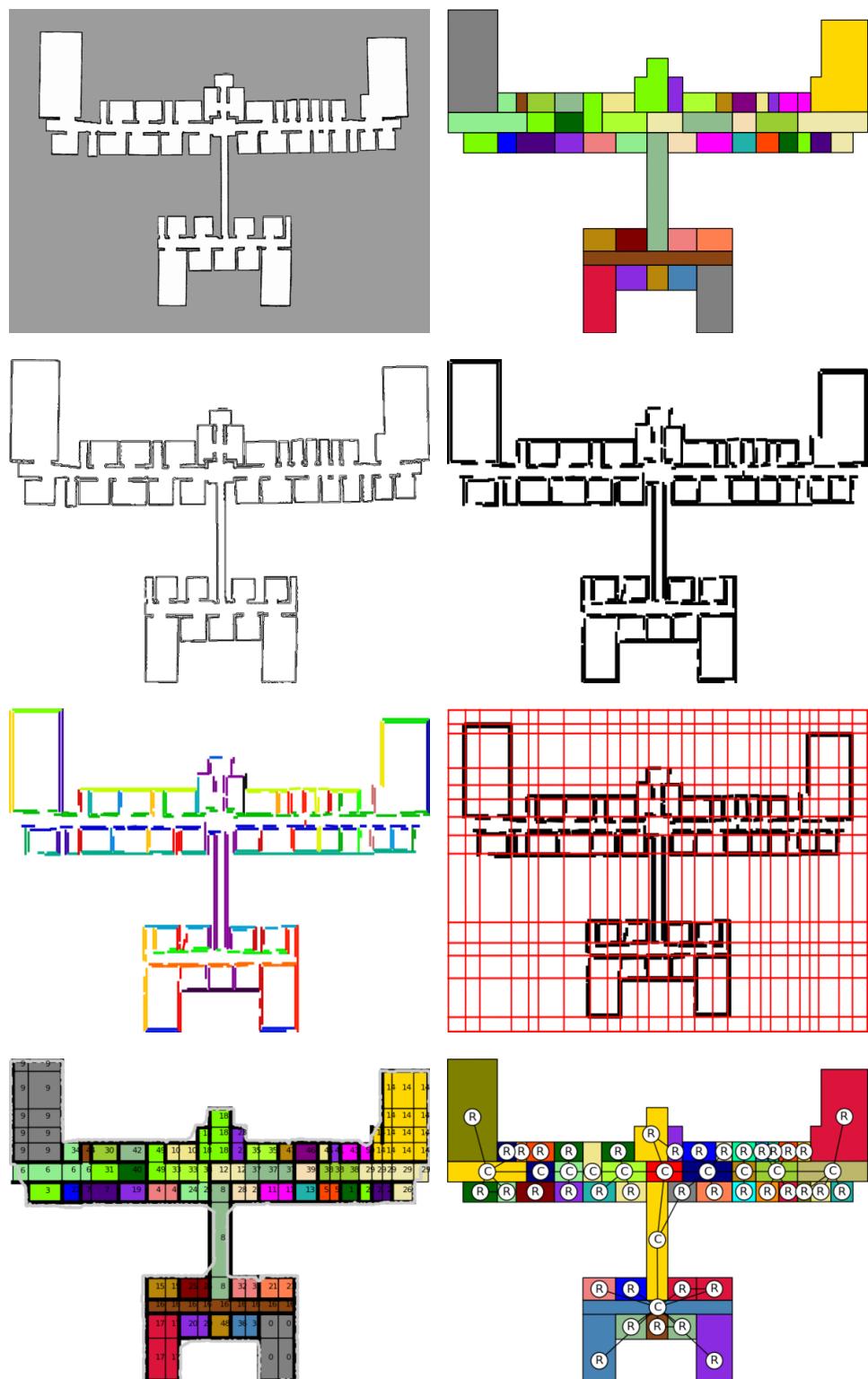


Figura A.2: Esempio 2.

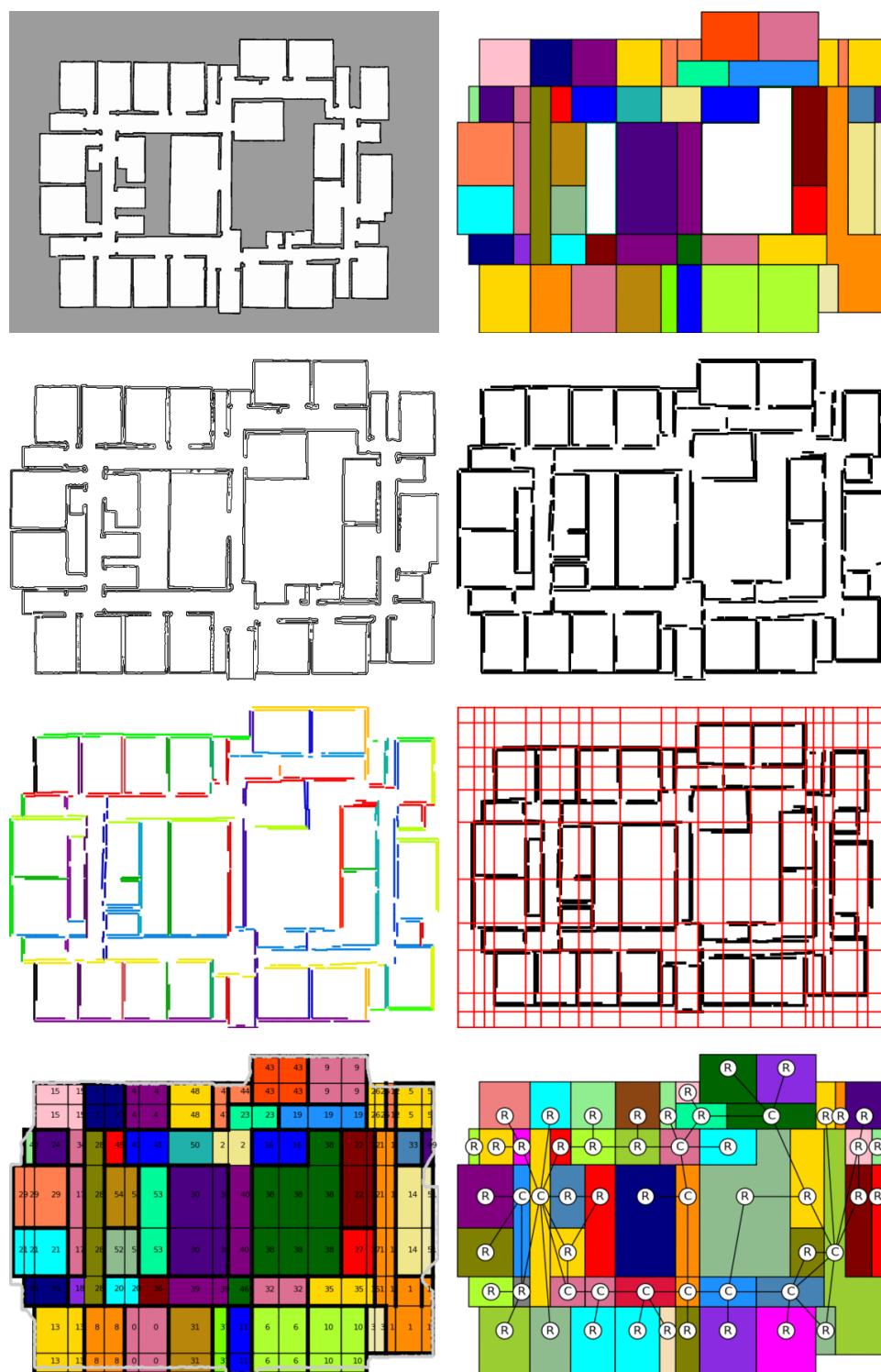


Figura A.3: Esempio 3.

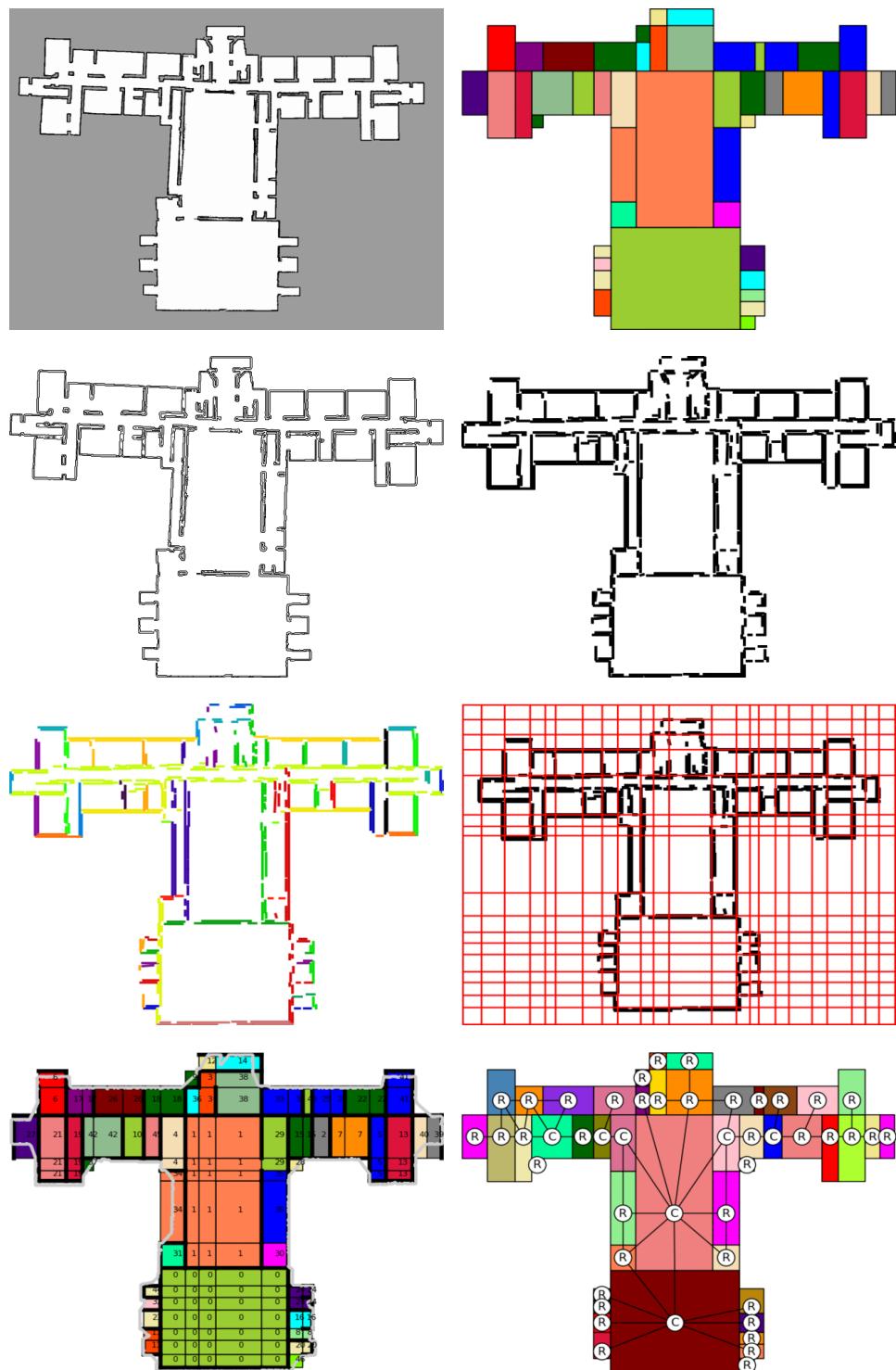


Figura A.4: Esempio 4.

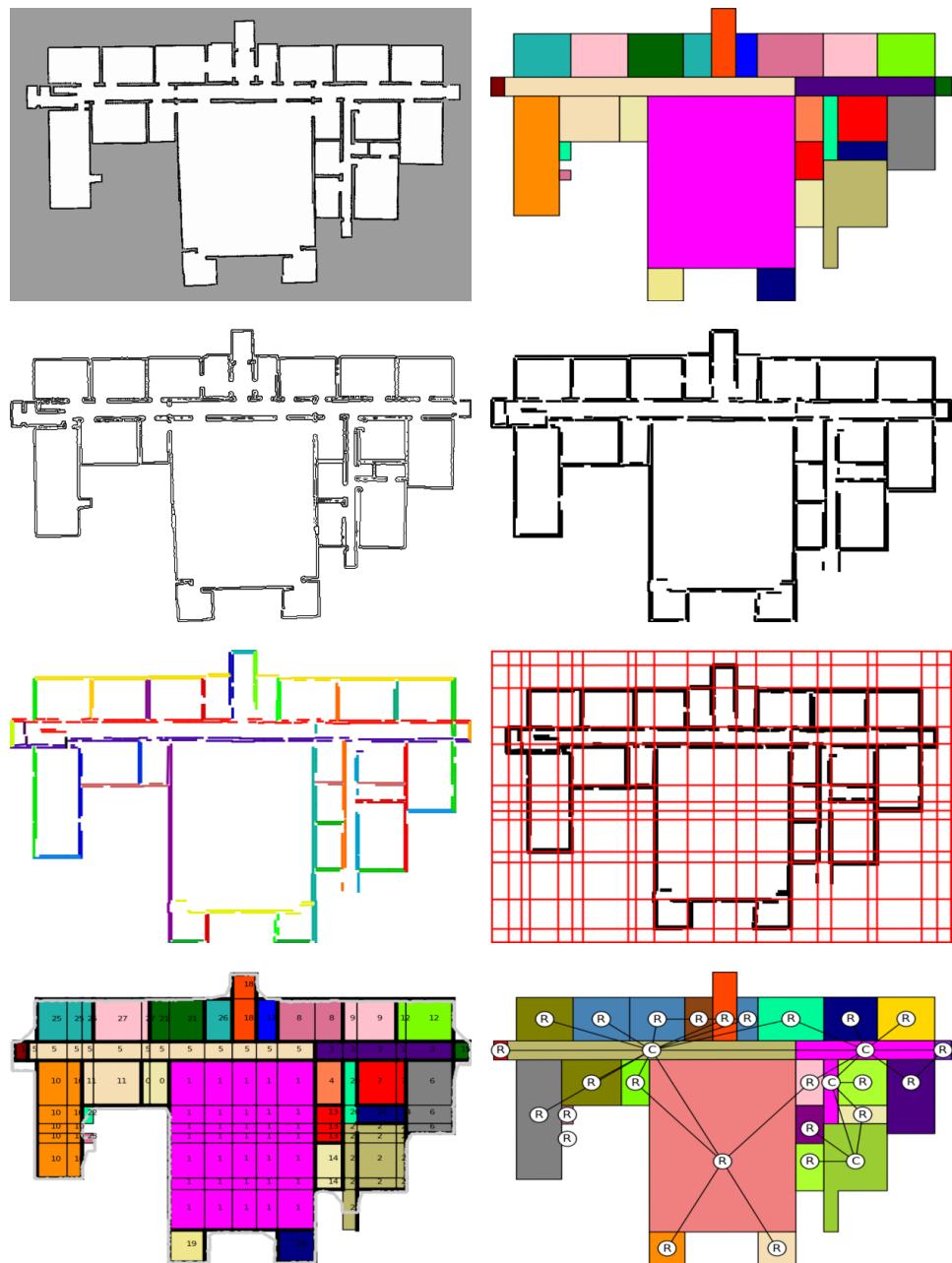


Figura A.5: Esempio 5.



Figura A.6: Esempio 6.

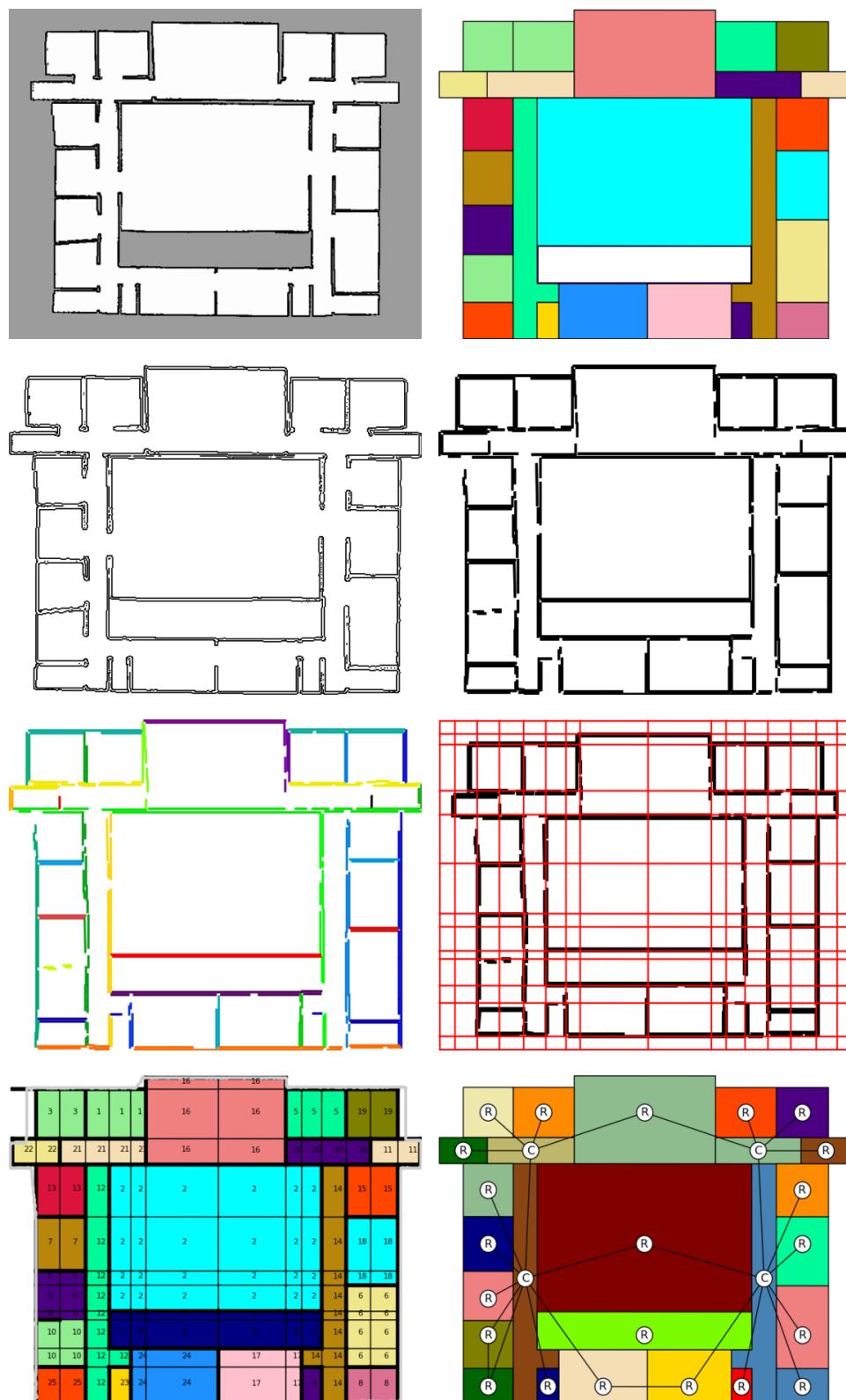


Figura A.7: Esempio 7.

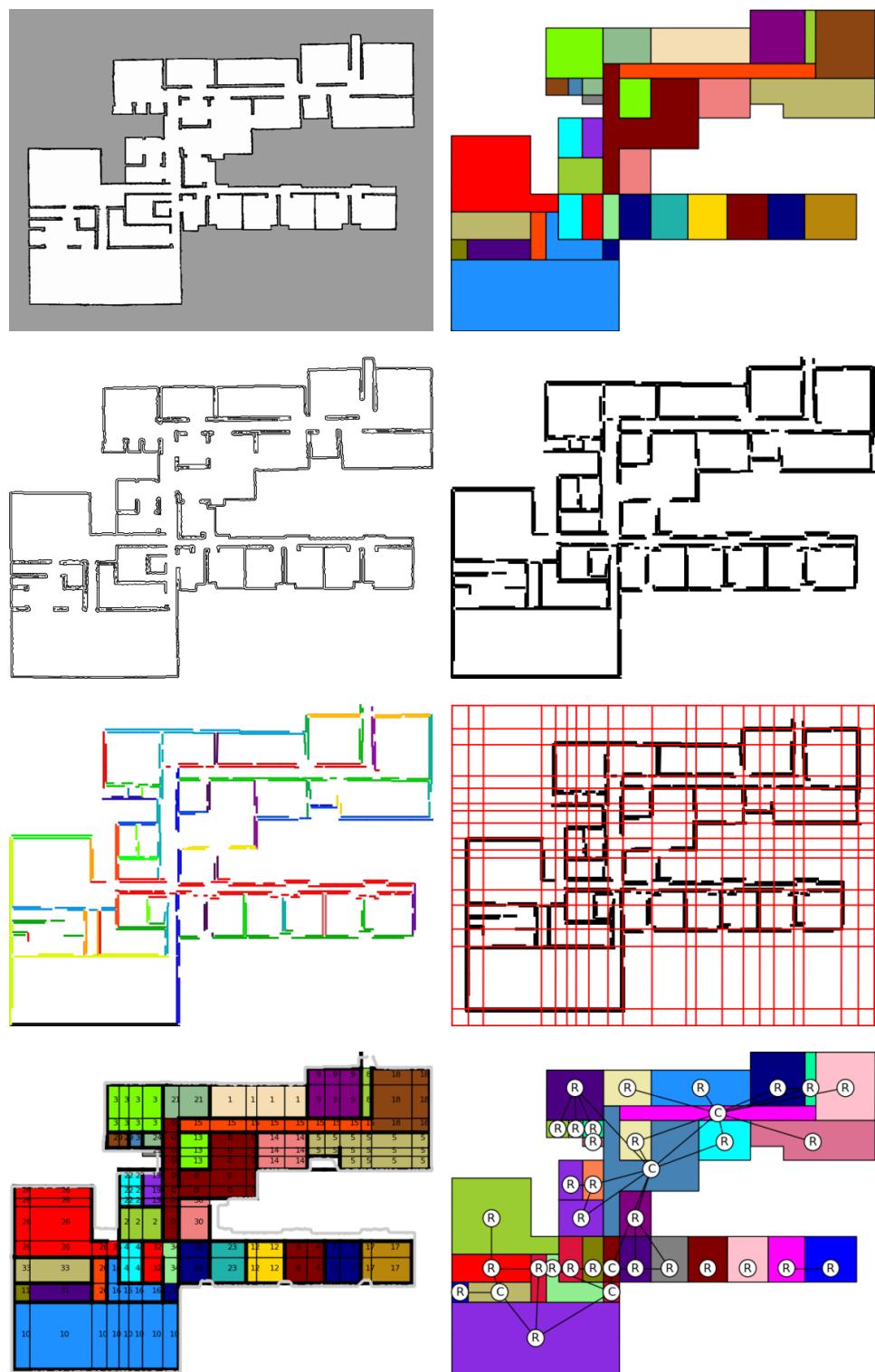


Figura A.8: Esempio 8.

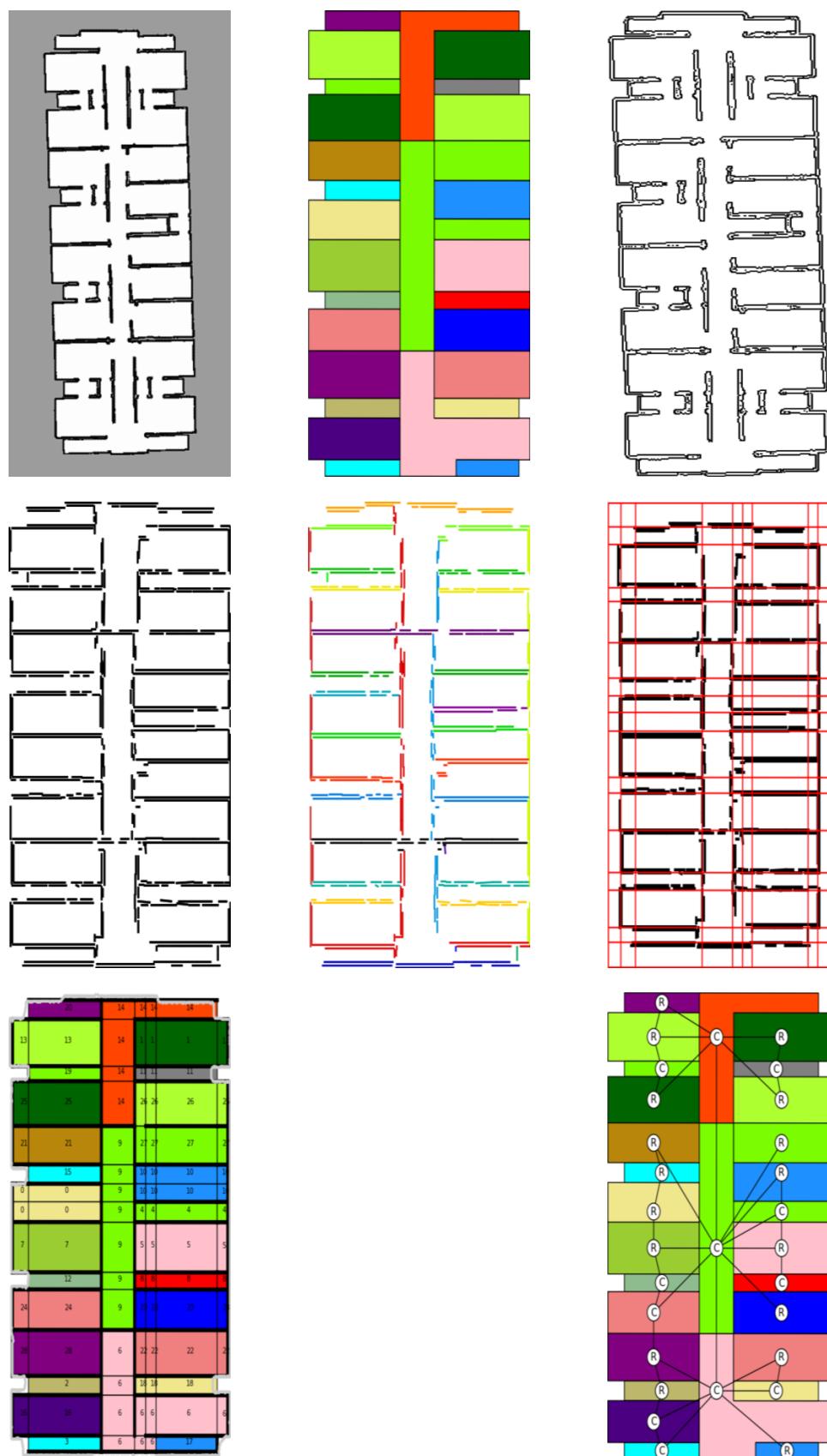


Figura A.9: Esempio 9.

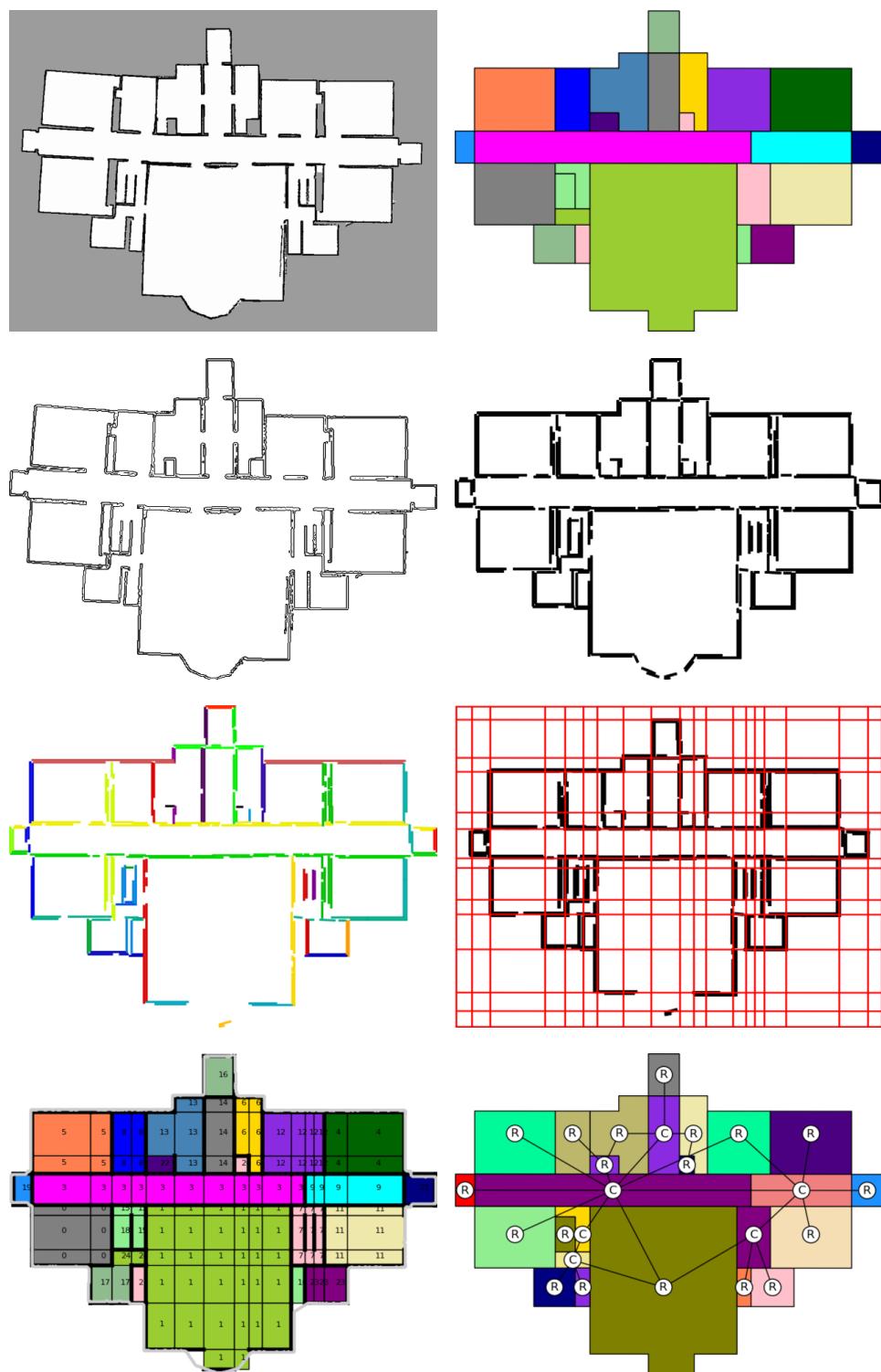


Figura A.10: Esempio 10.

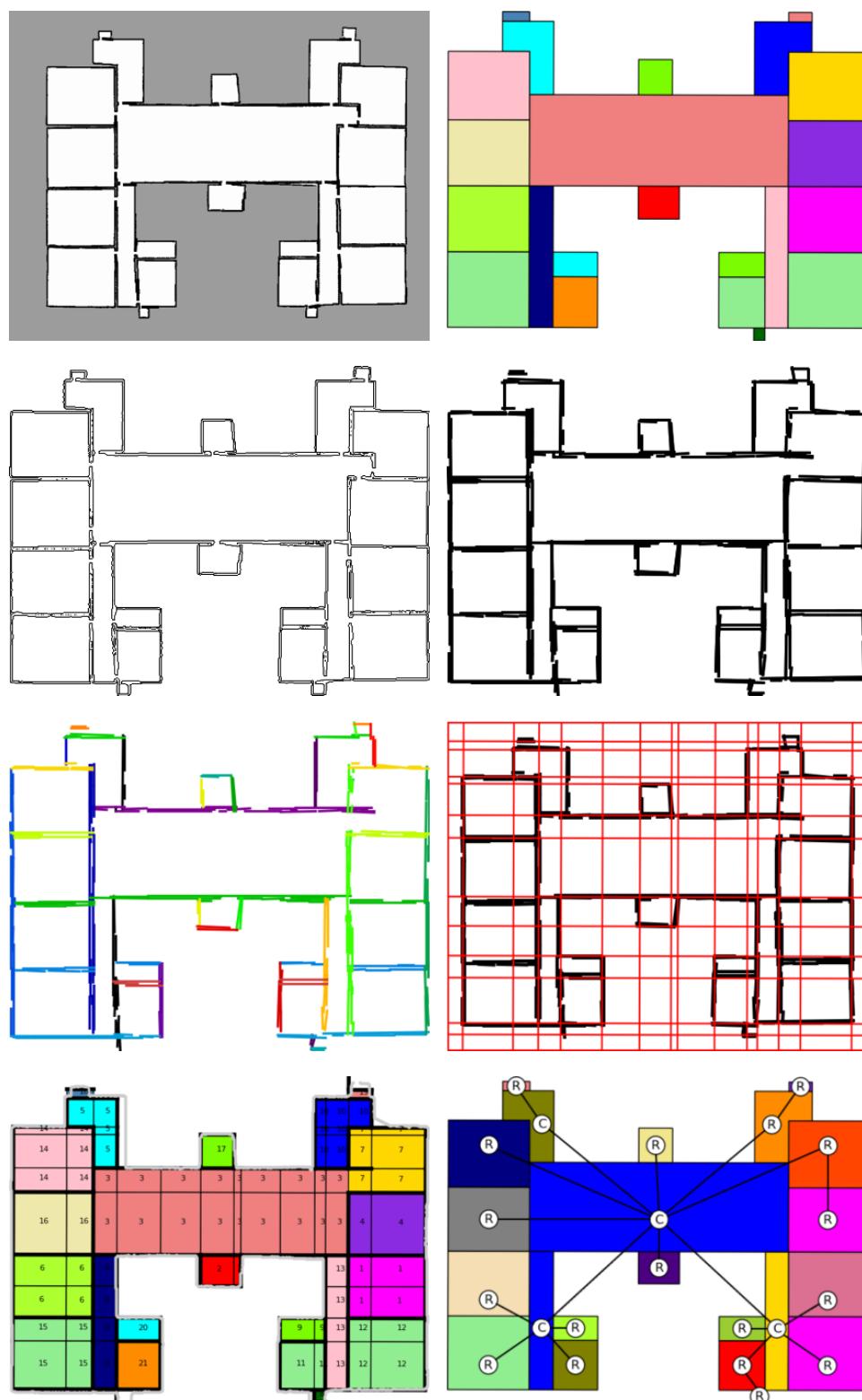


Figura A.11: Esempio 11.



Figura A.12: Esempio 12.

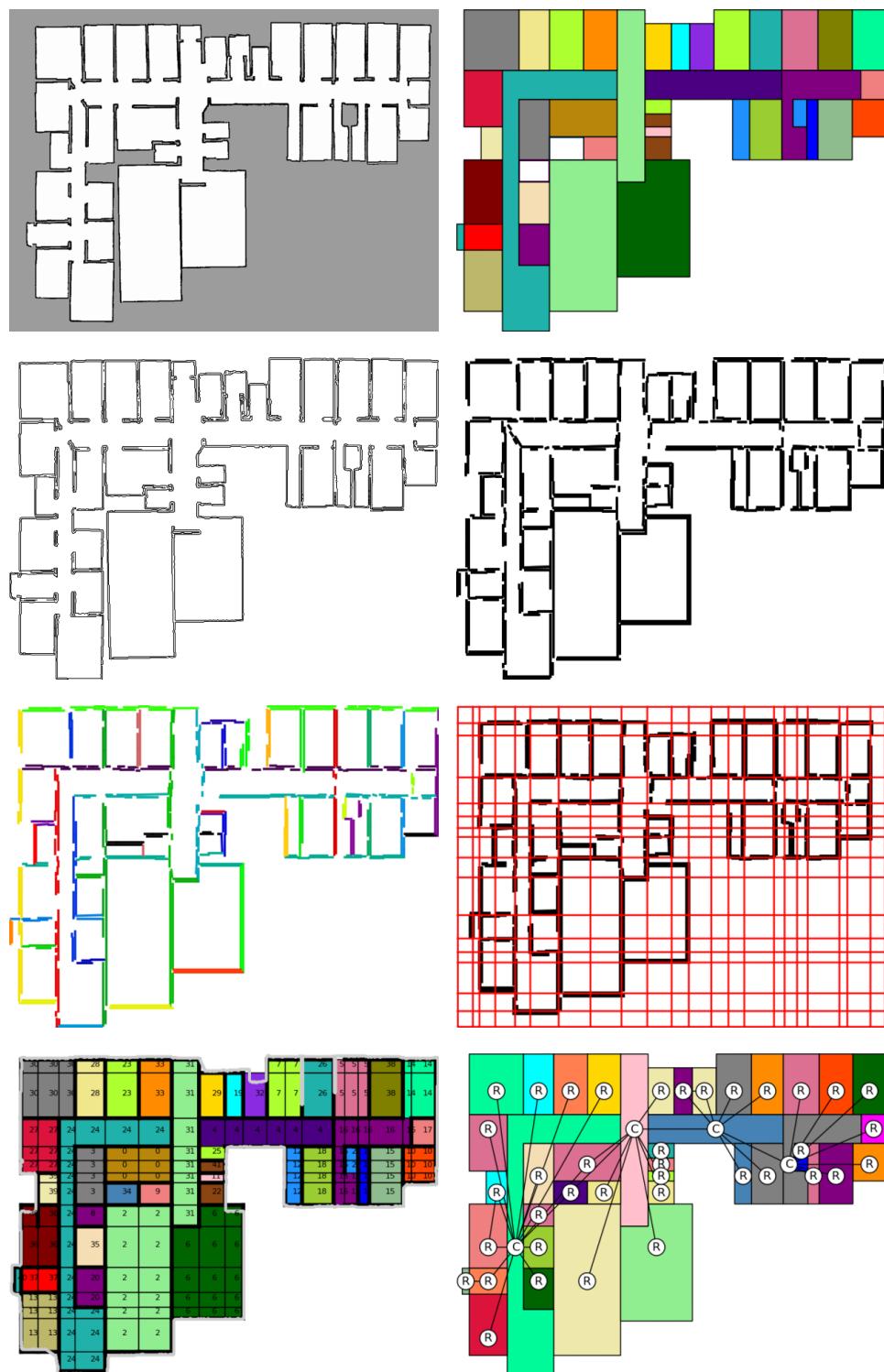


Figura A.13: Esempio 13.

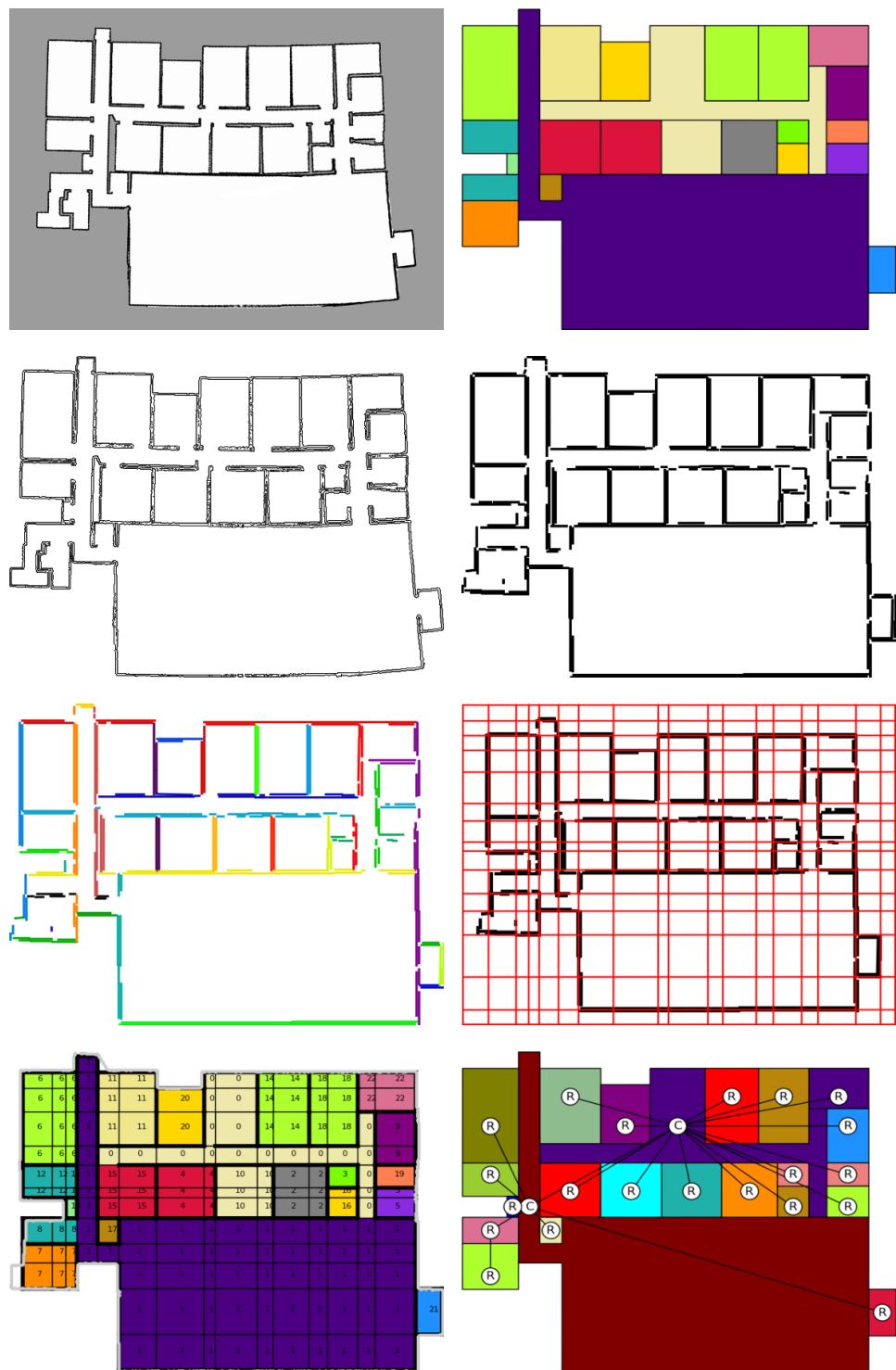


Figura A.14: Esempio 14.

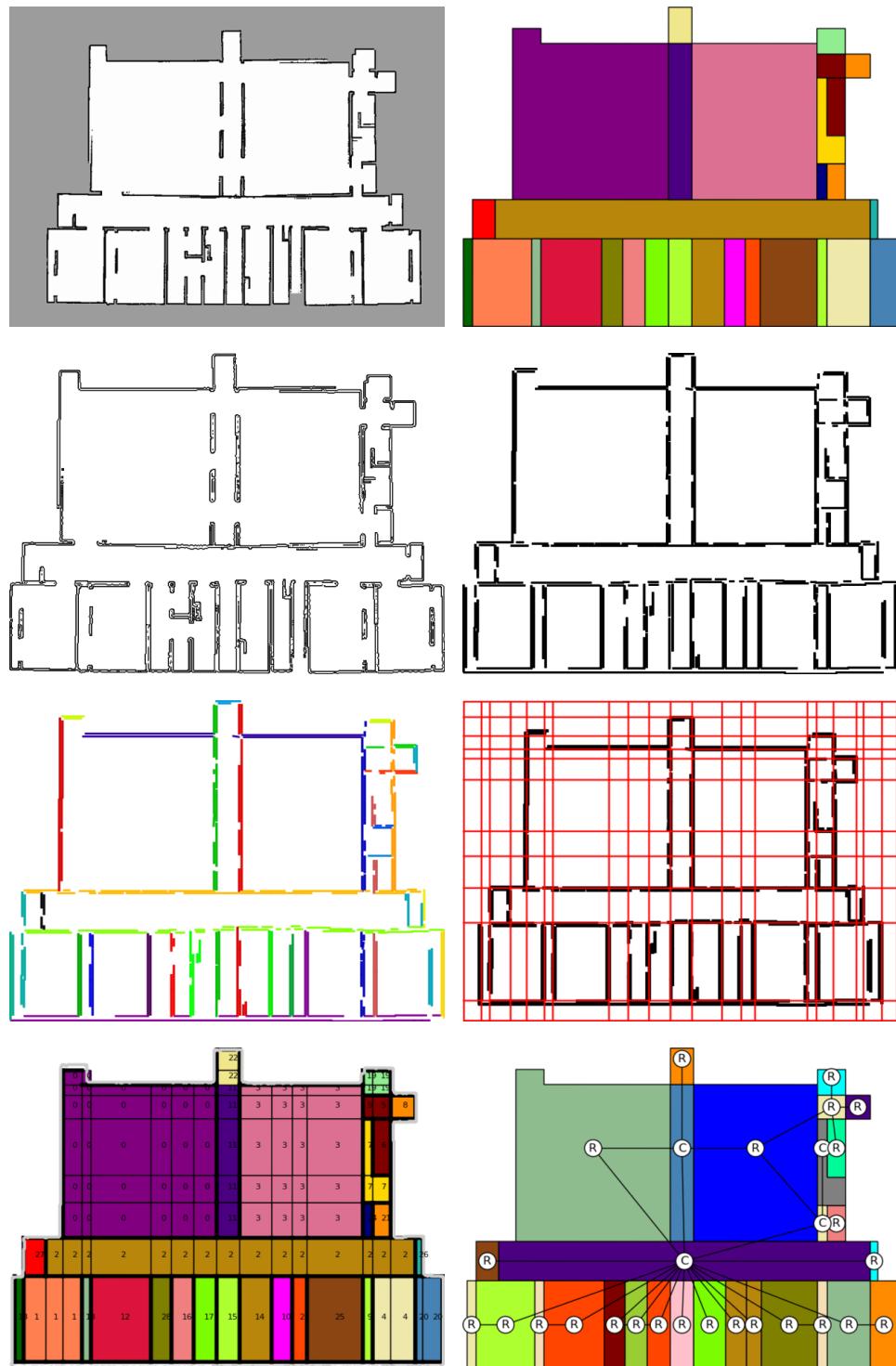


Figura A.15: Esempio 15.

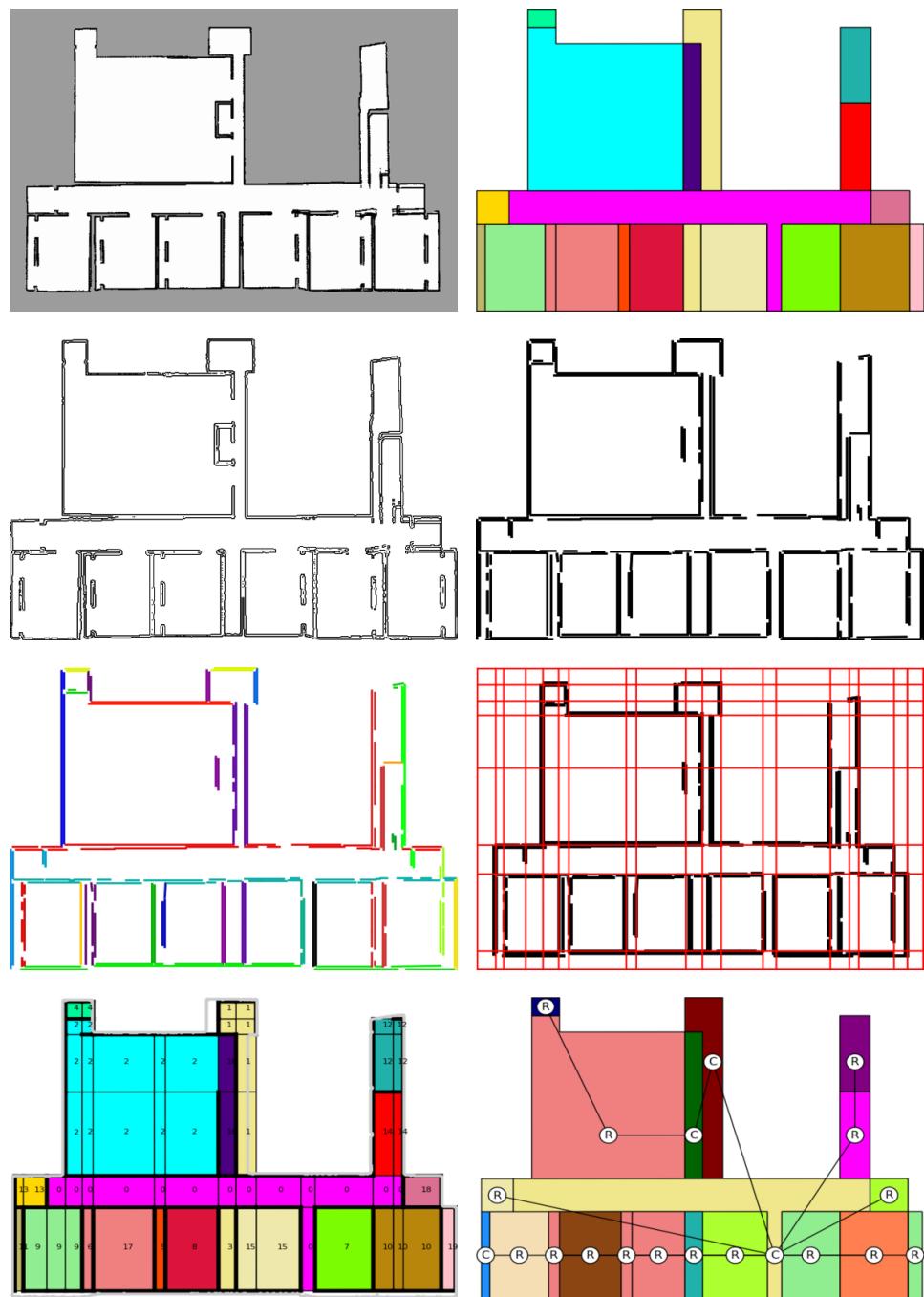


Figura A.16: Esempio 16.



Figura A.17: Esempio 17.

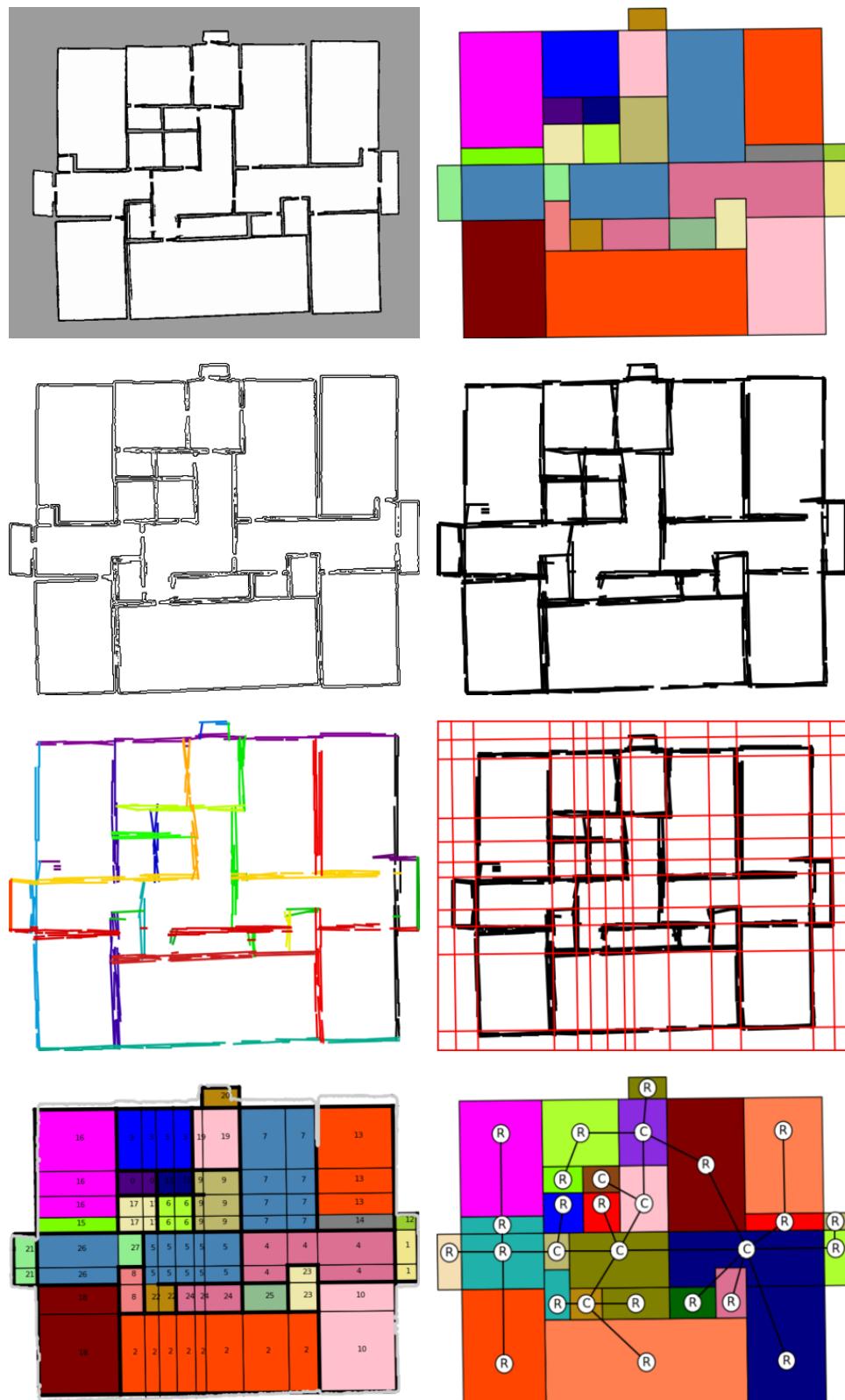


Figura A.18: Esempio 18.

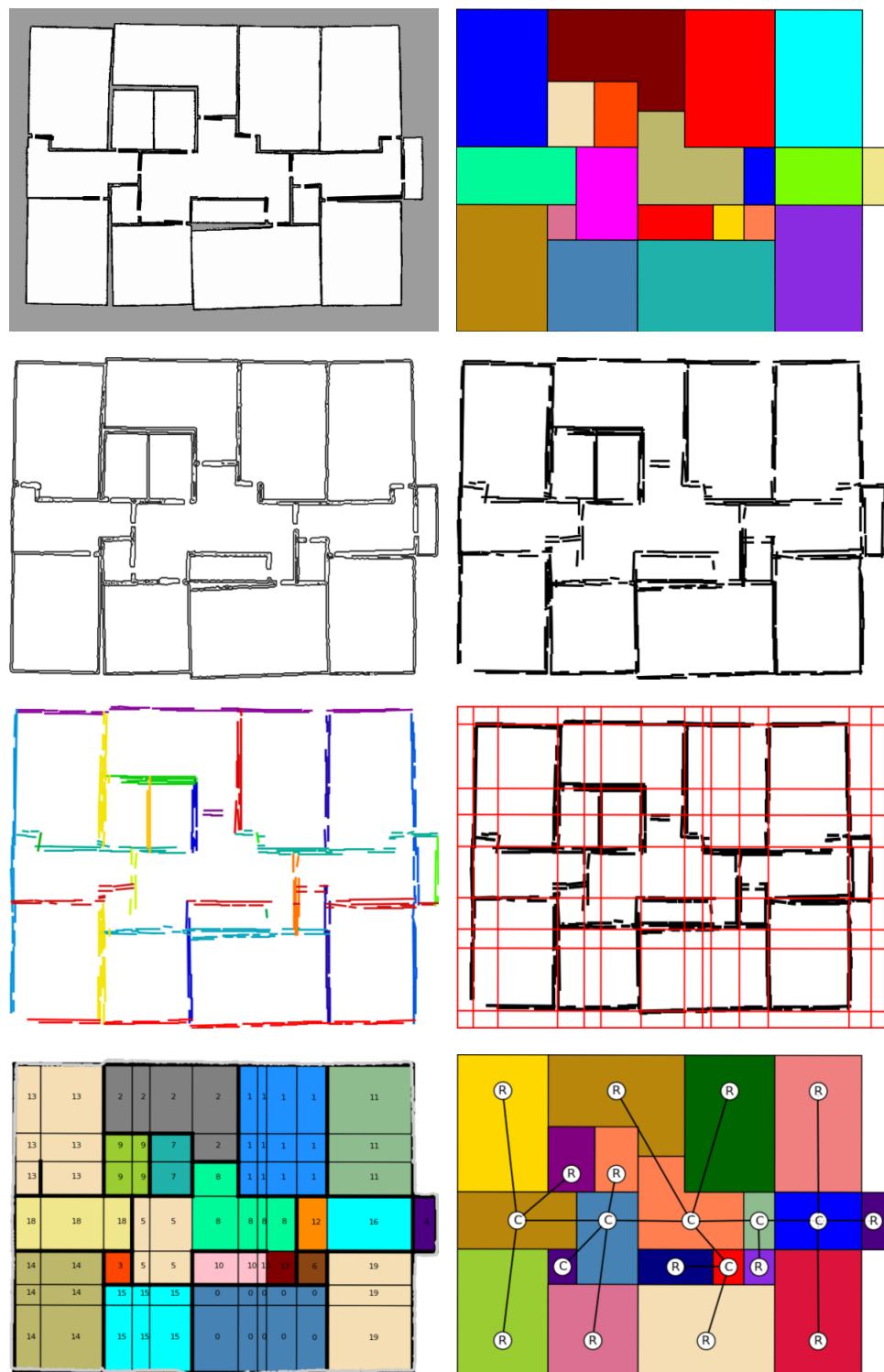


Figura A.19: Esempio 19.

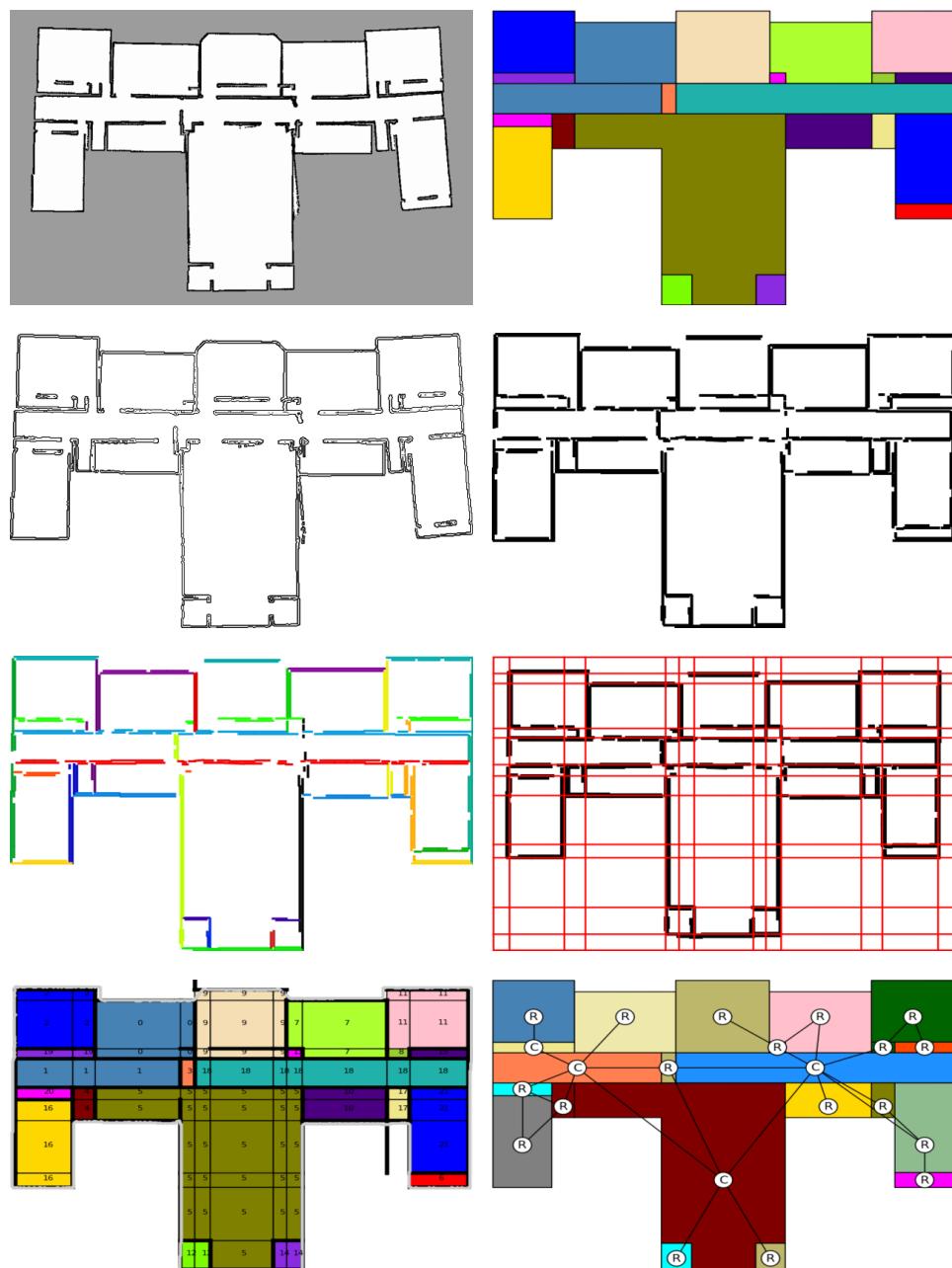


Figura A.20: Esempio 20.



Figura A.21: Esempio 21.



Figura A.22: Esempio 22.

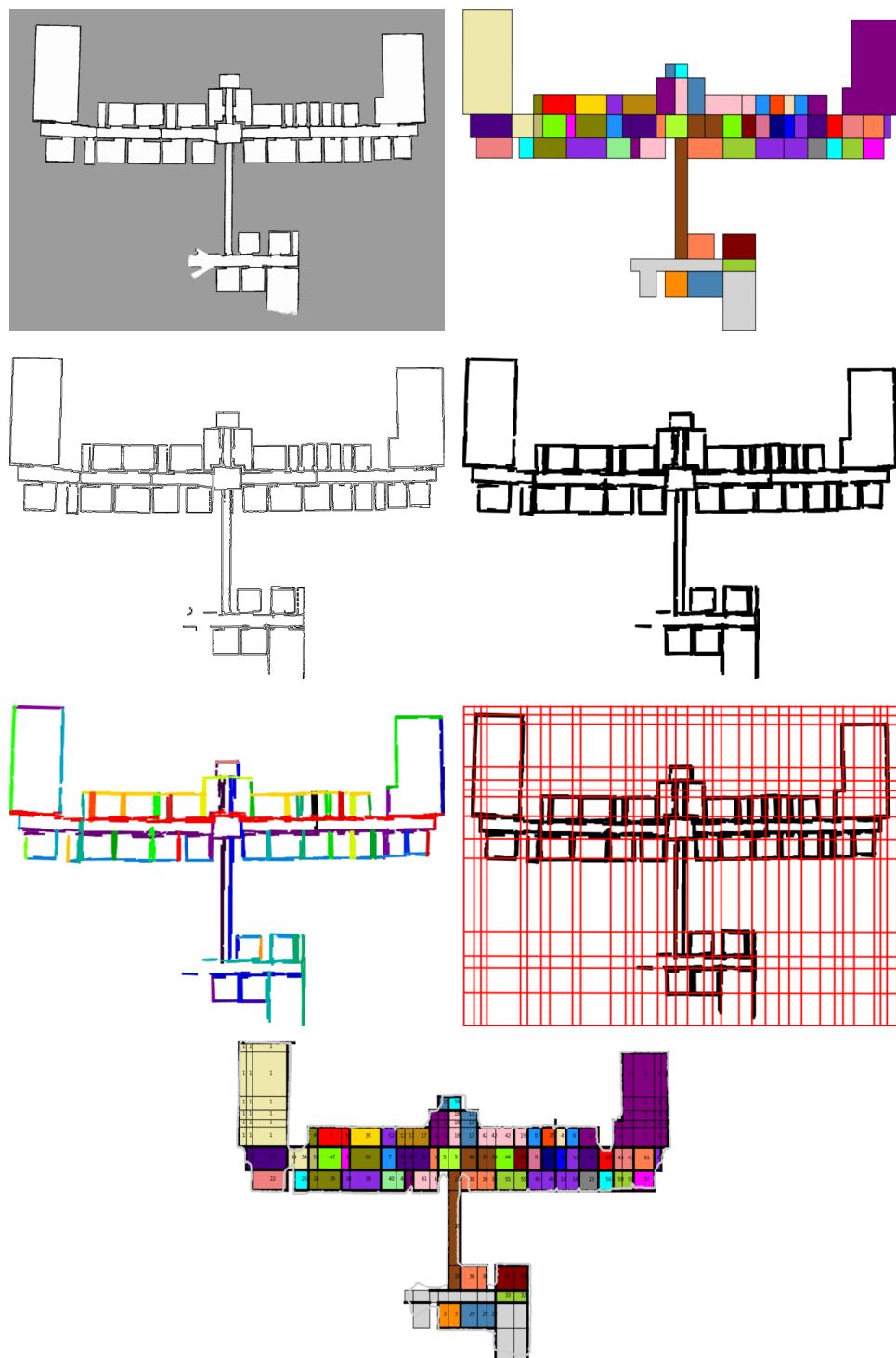


Figura A.23: Esempio 23.



Figura A.24: Esempio 24.

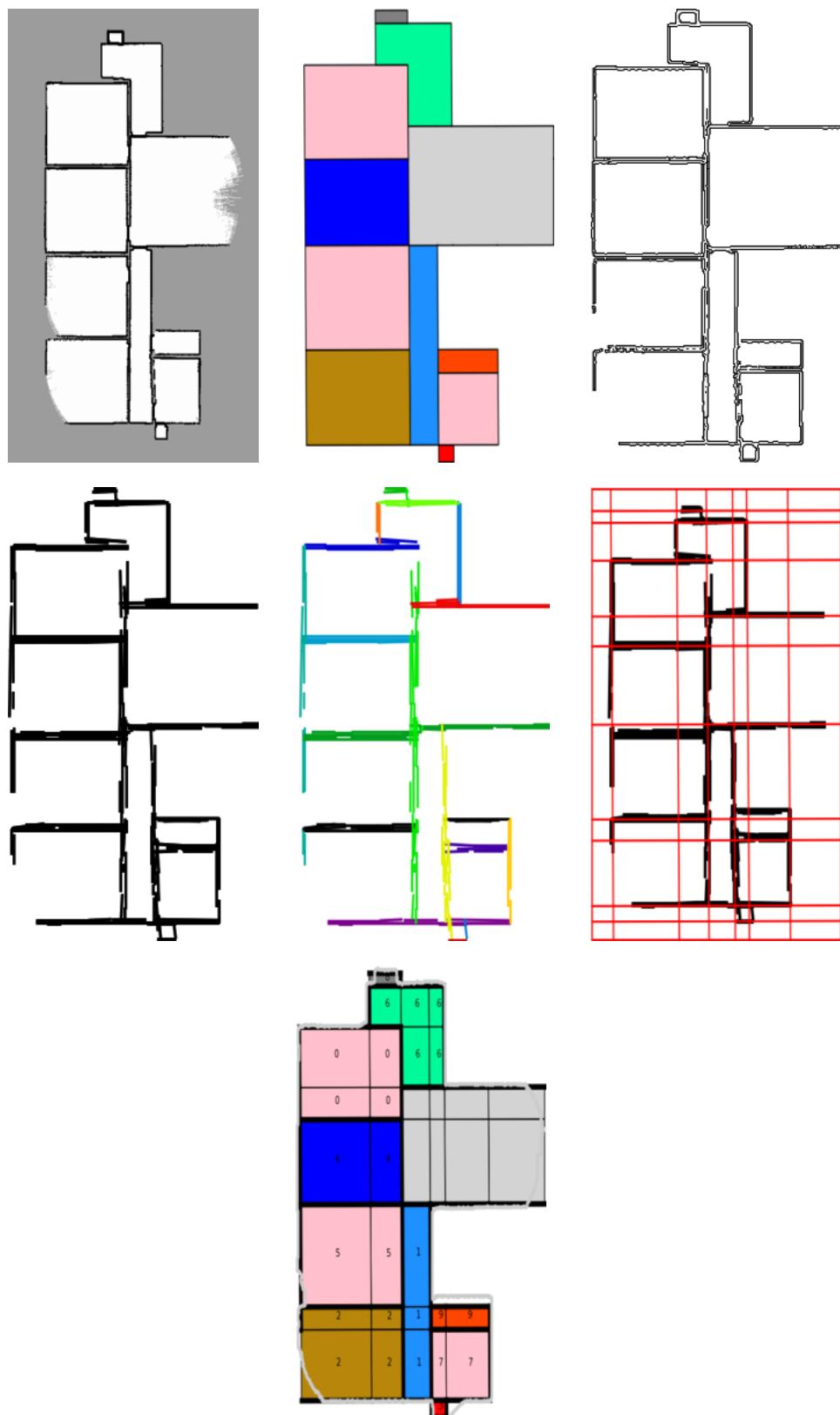


Figura A.25: Esempio 25.

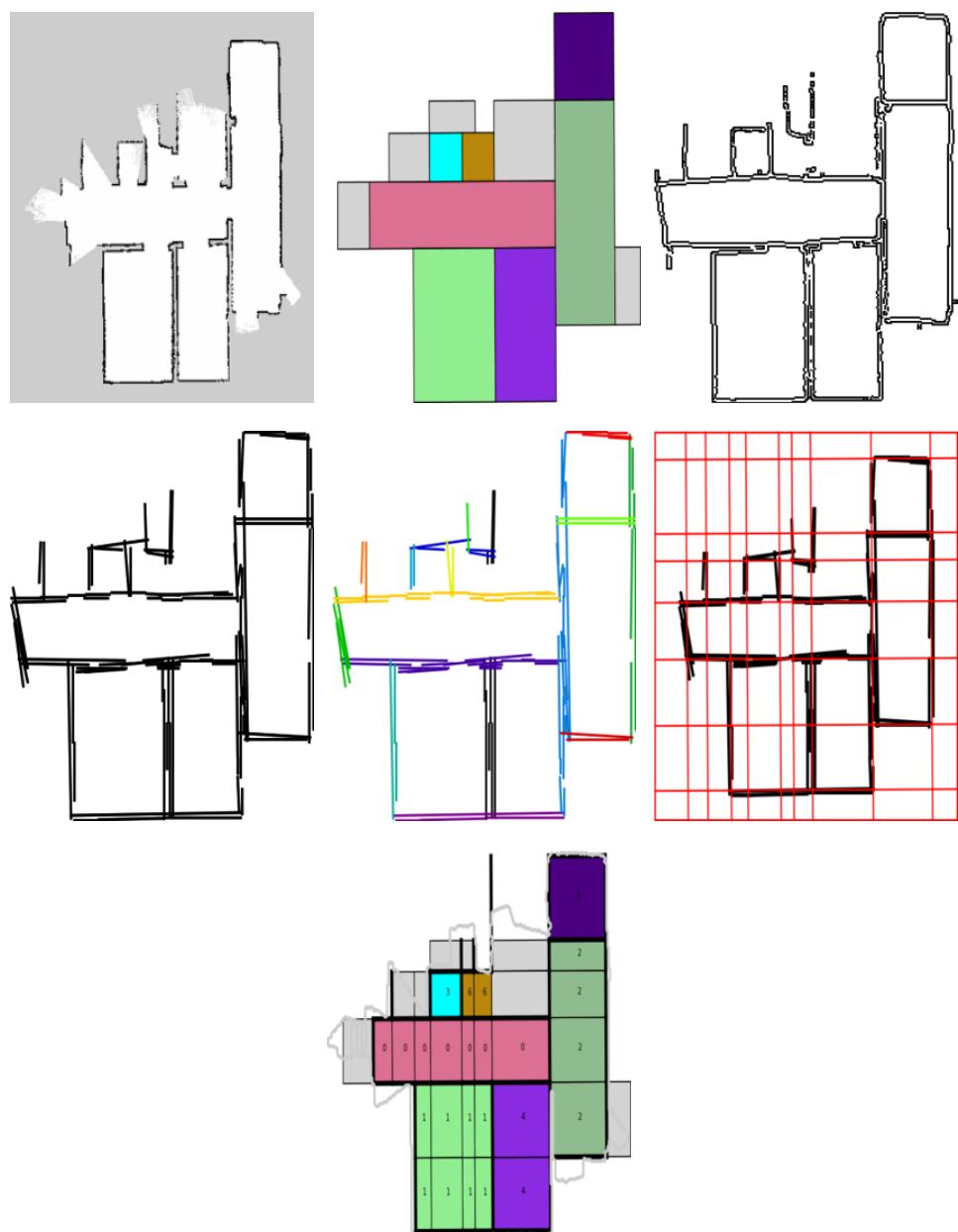


Figura A.26: Esempio 26.

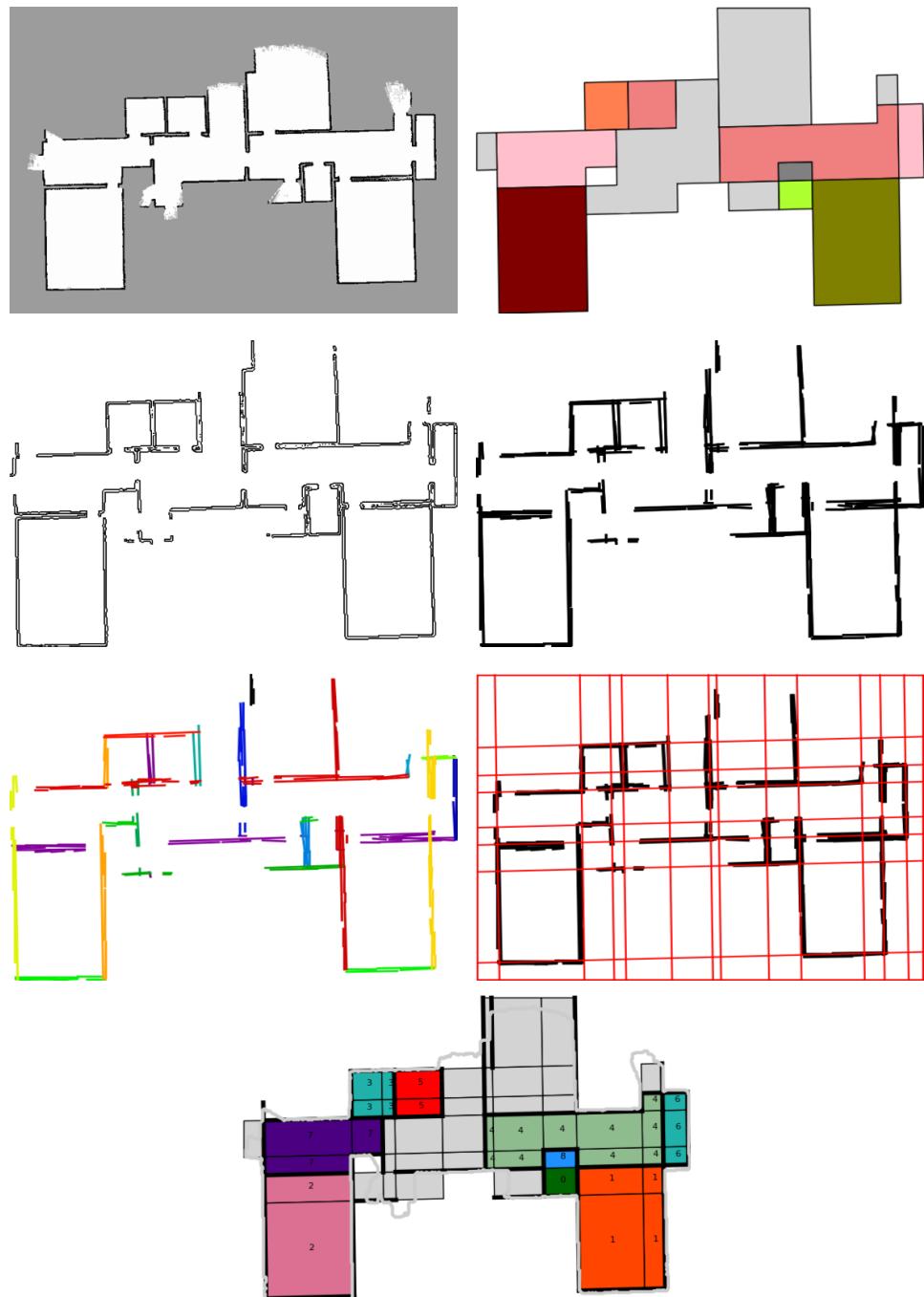


Figura A.27: Esempio 27.



Figura A.28: Esempio 28.



Figura A.29: Esempio 29.

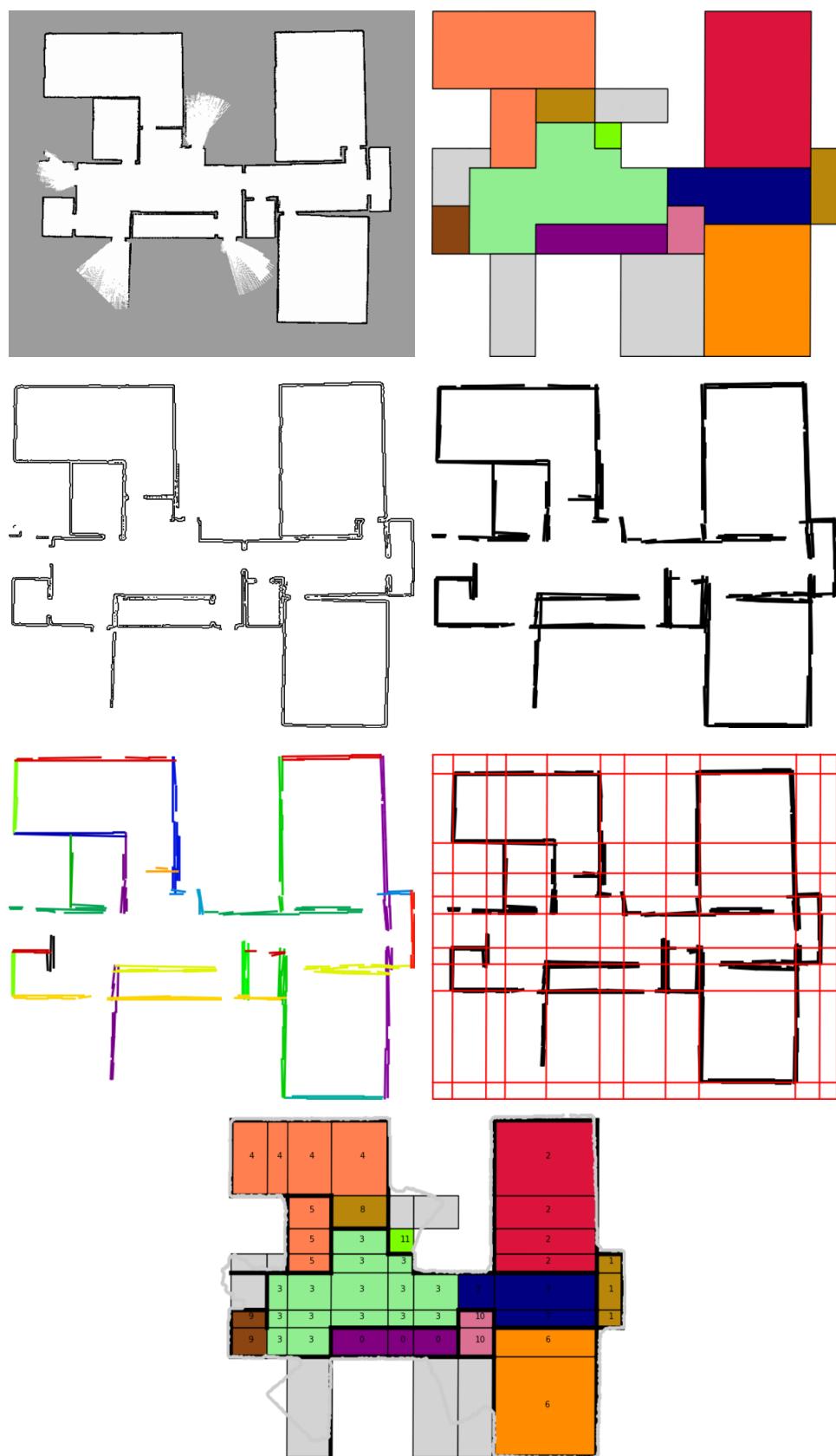


Figura A.30: Esempio 30.



Figura A.31: Esempio 31.

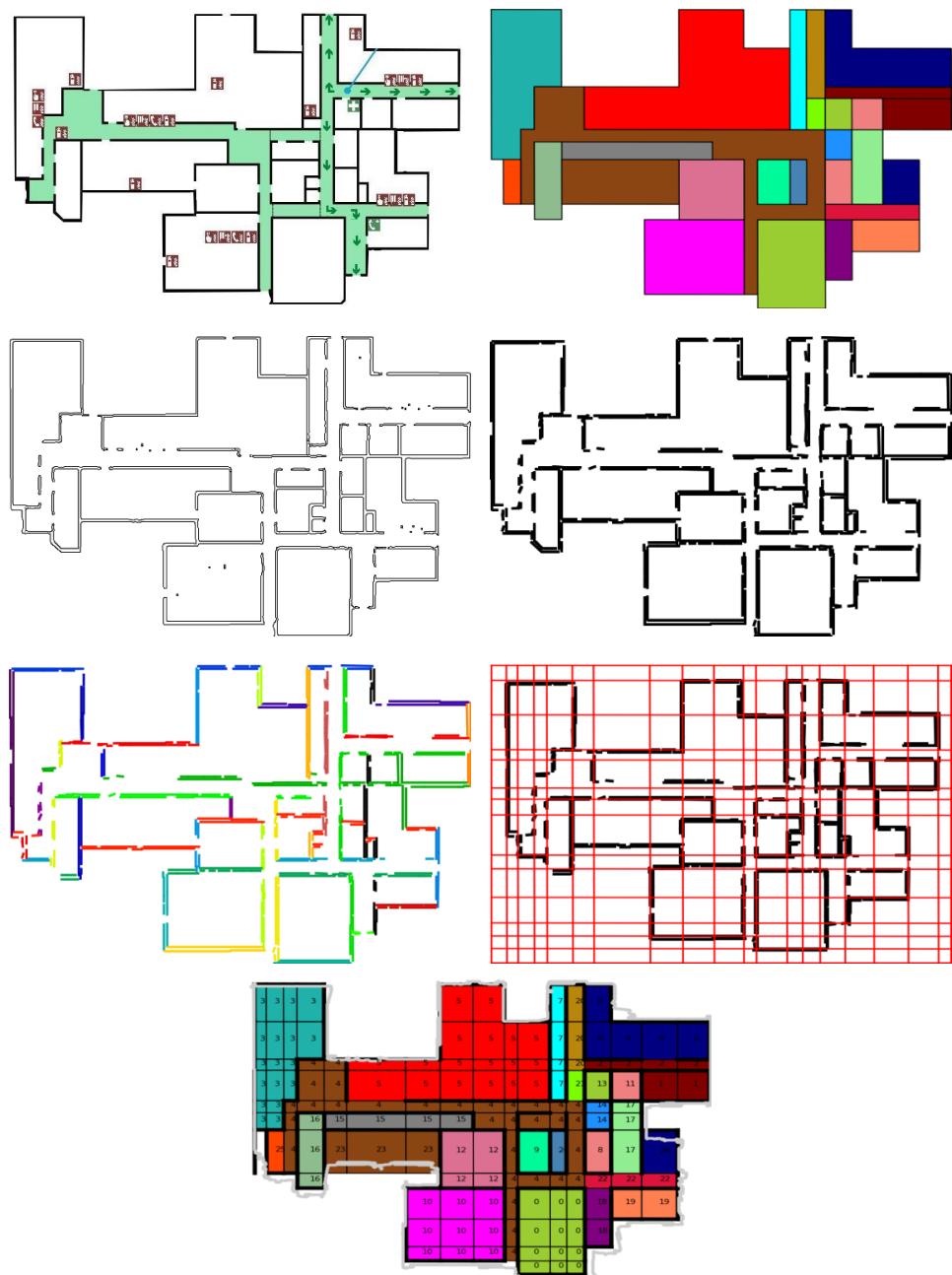


Figura A.32: Esempio 32.

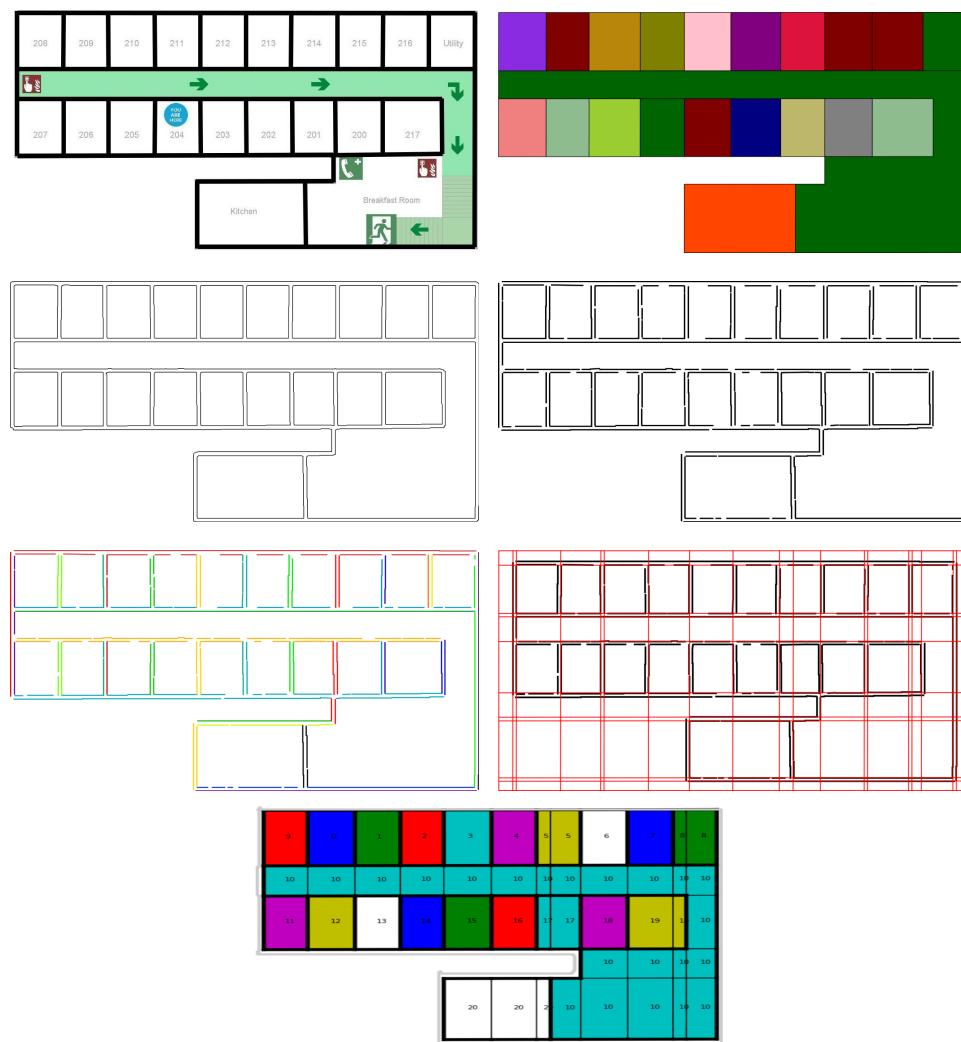


Figura A.33: Esempio 33.

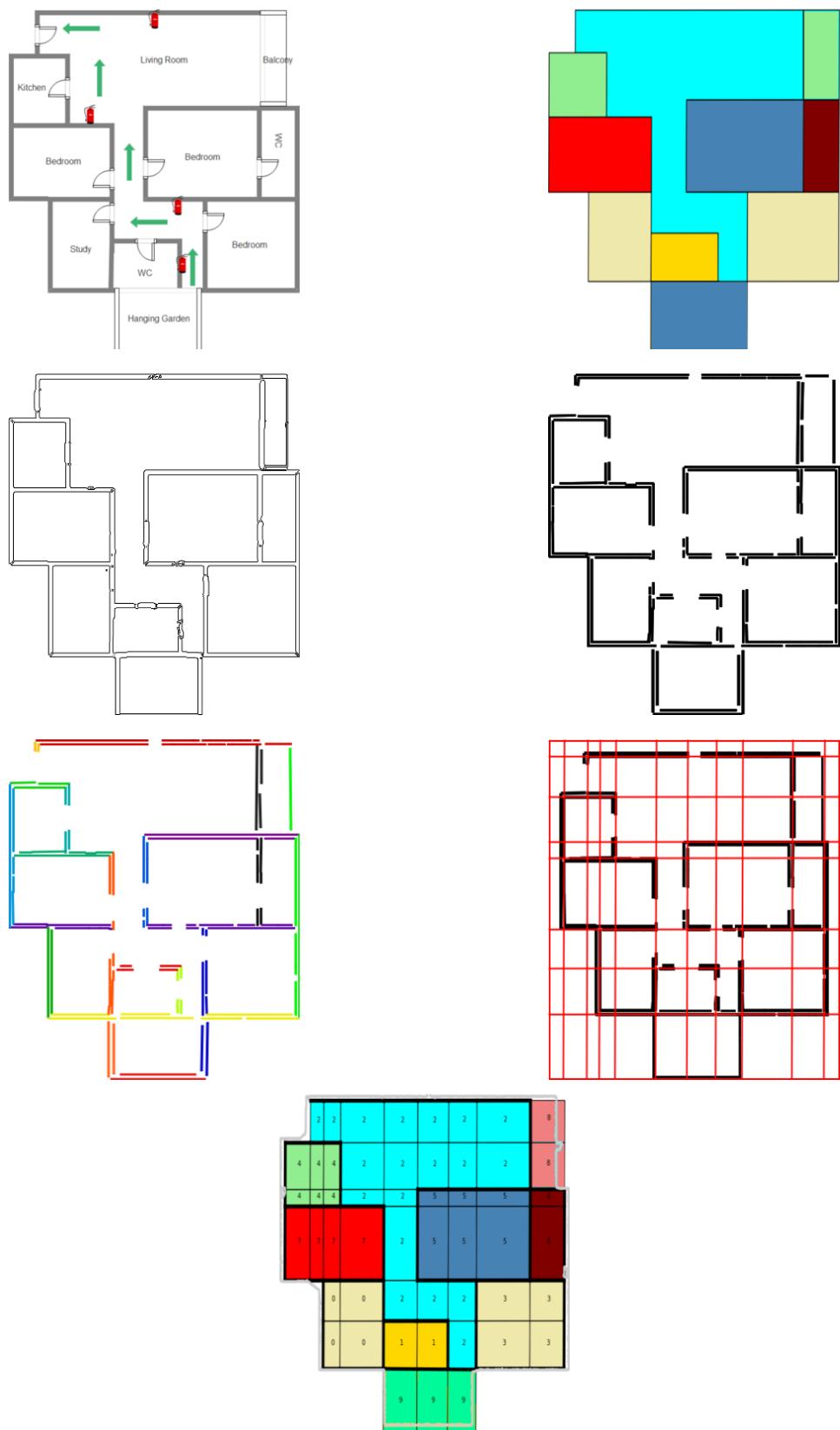


Figura A.34: Esempio 34.

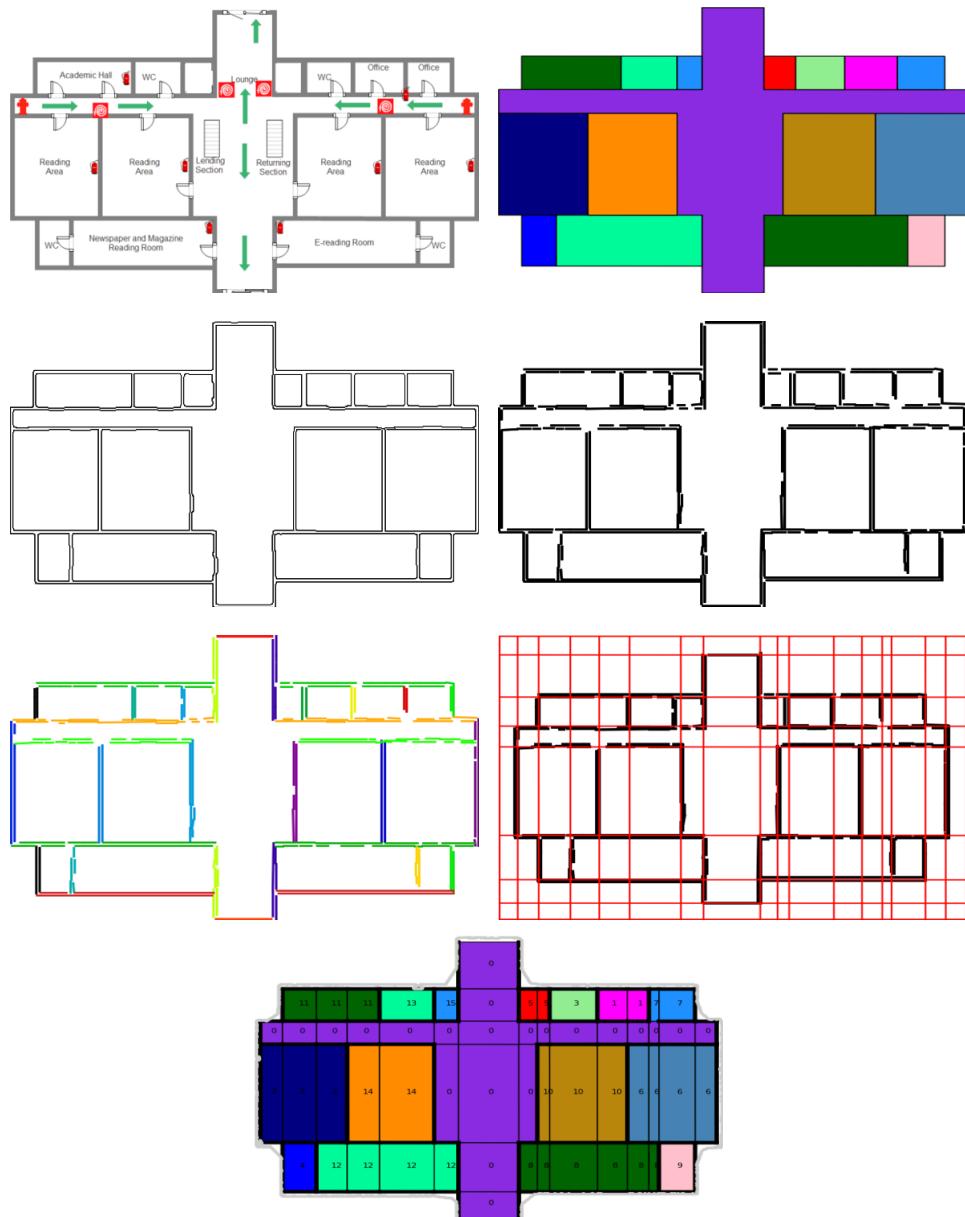


Figura A.35: Esempio 35.

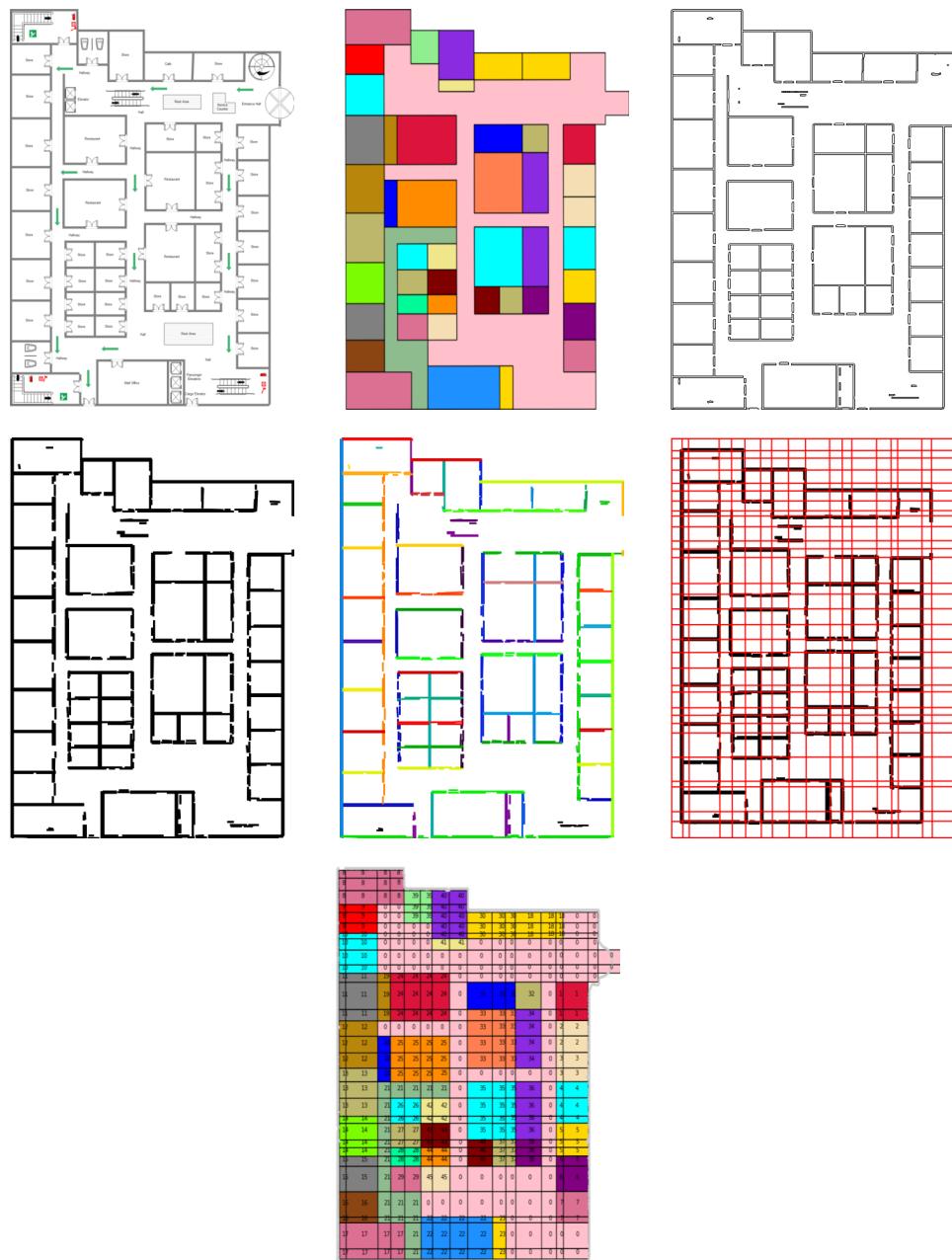


Figura A.36: Esempio 36.

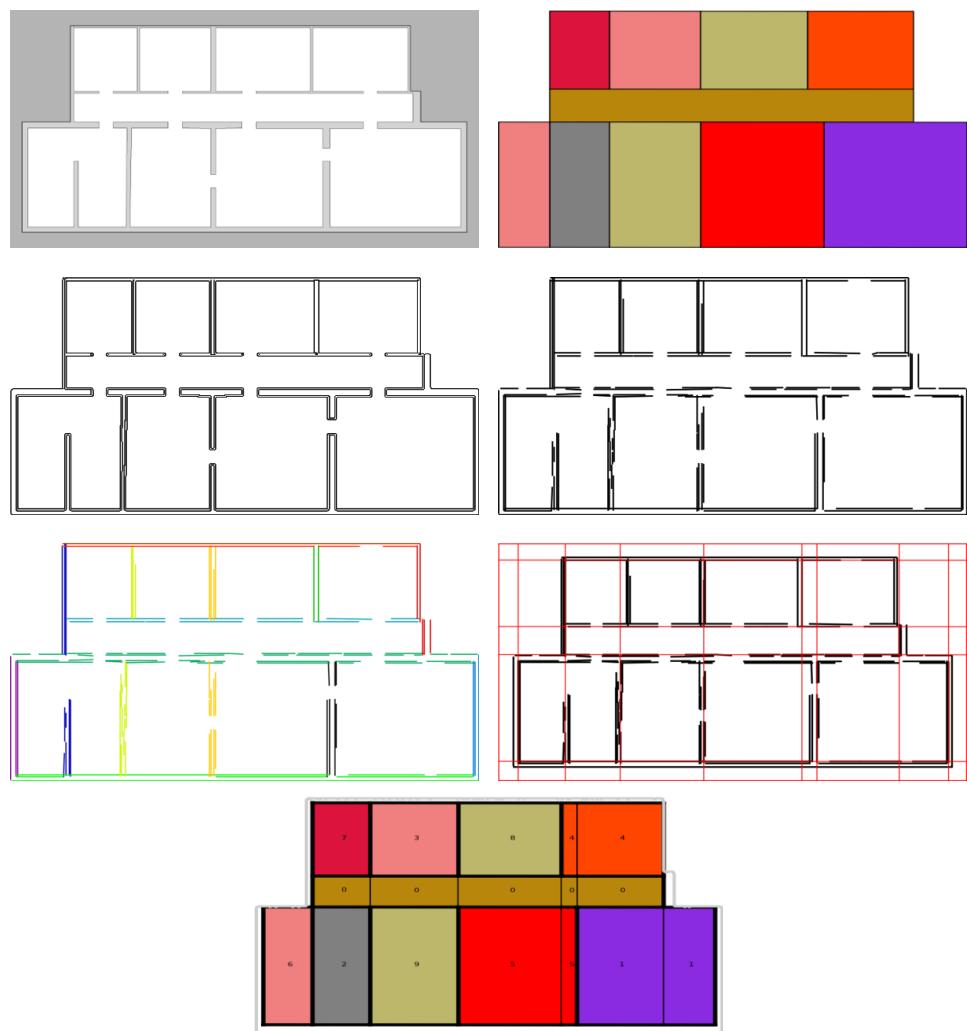


Figura A.37: Esempio 37.

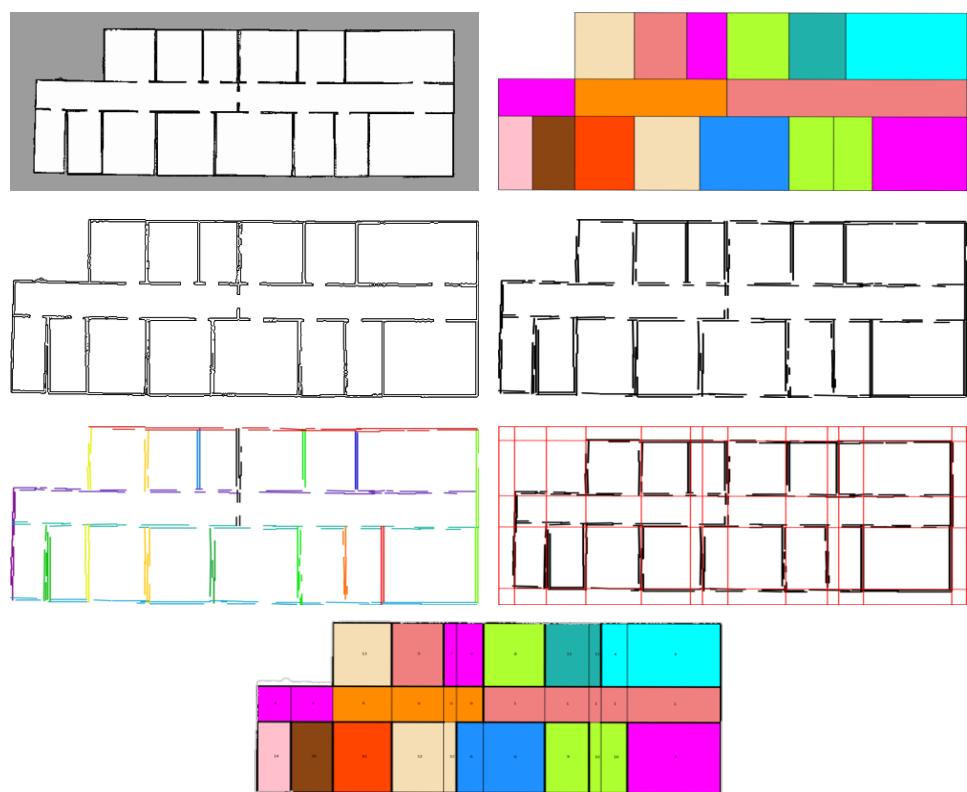


Figura A.38: Esempio 38.

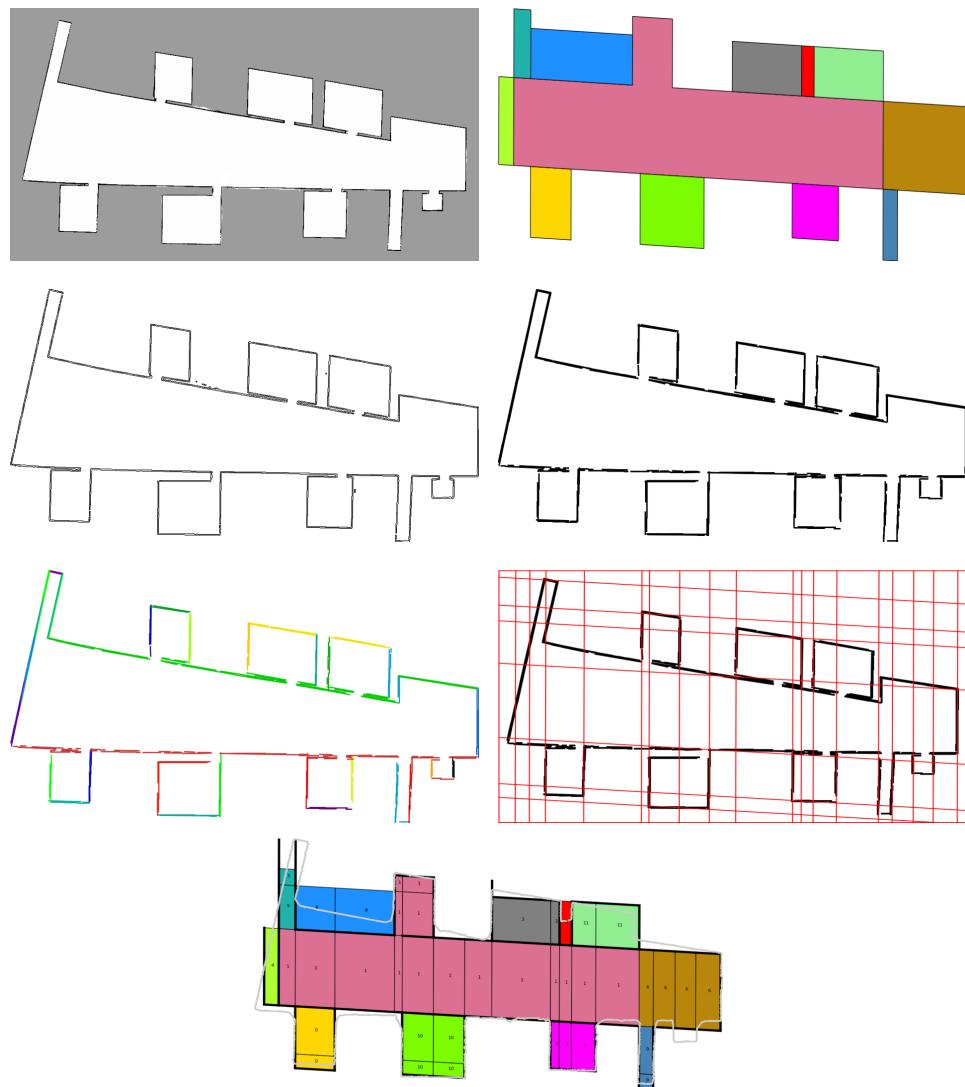


Figura A.39: Esempio 39.

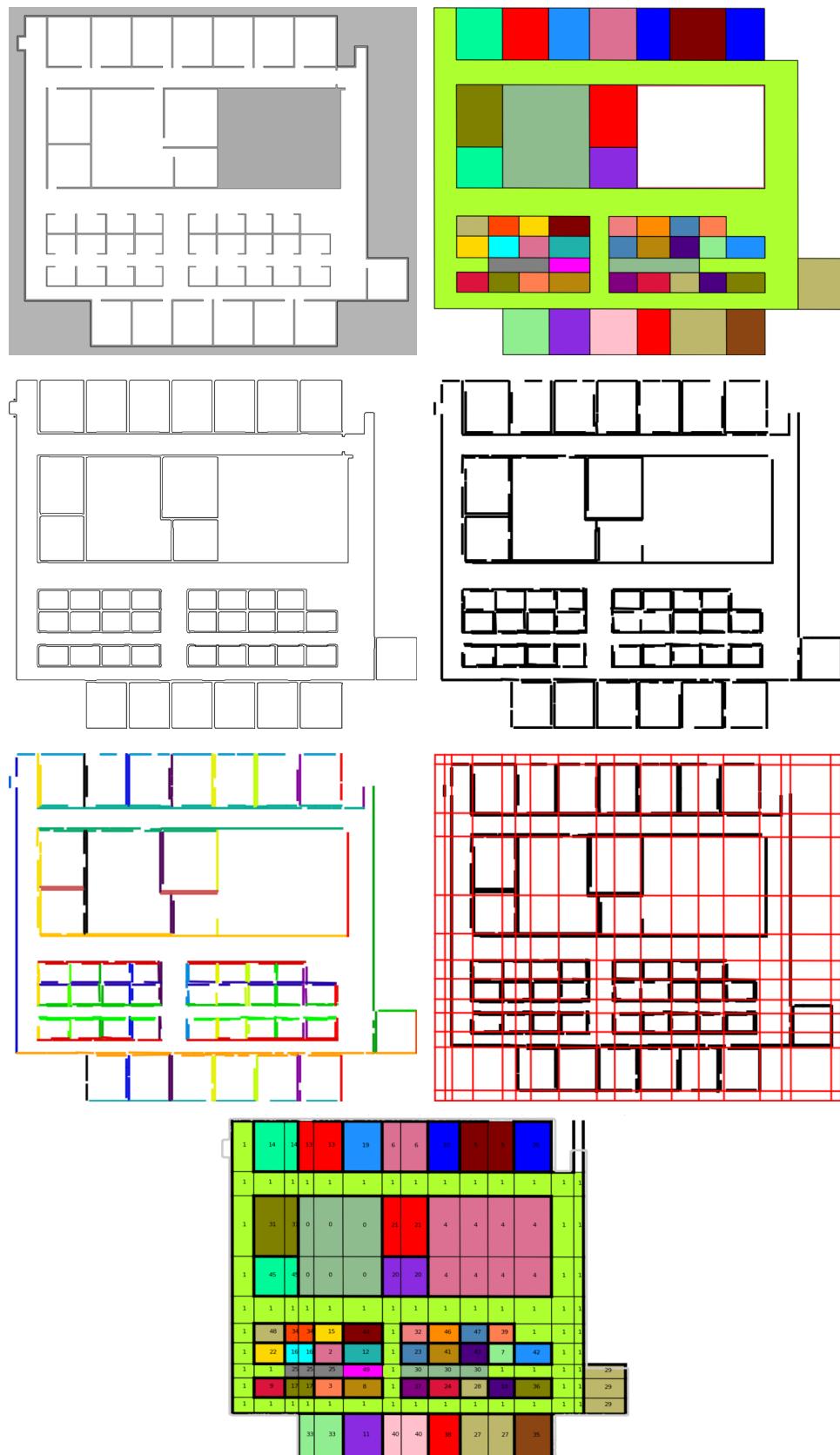
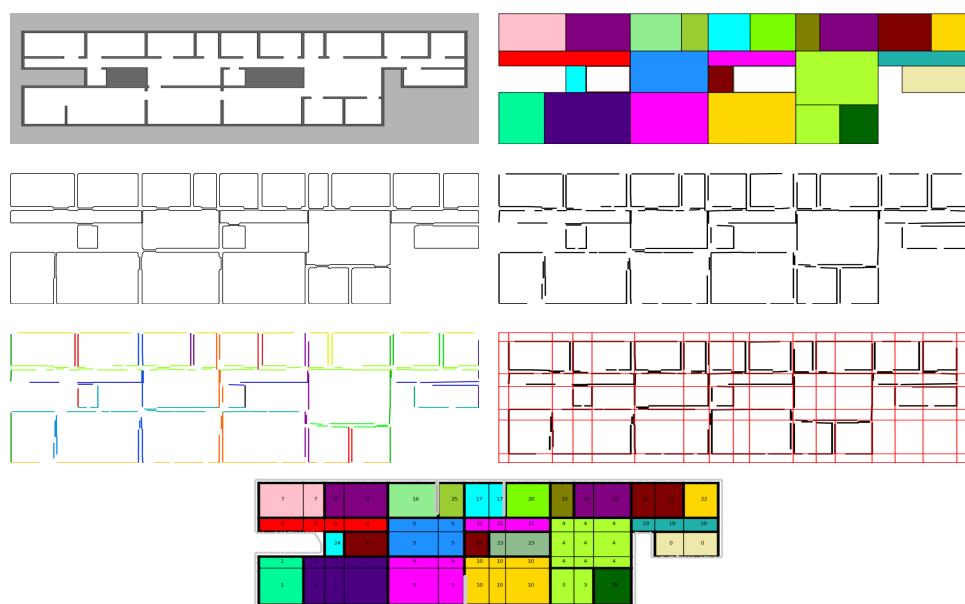


Figura A.40: Esempio 40.

*Figura A.41: Esempio 41.*

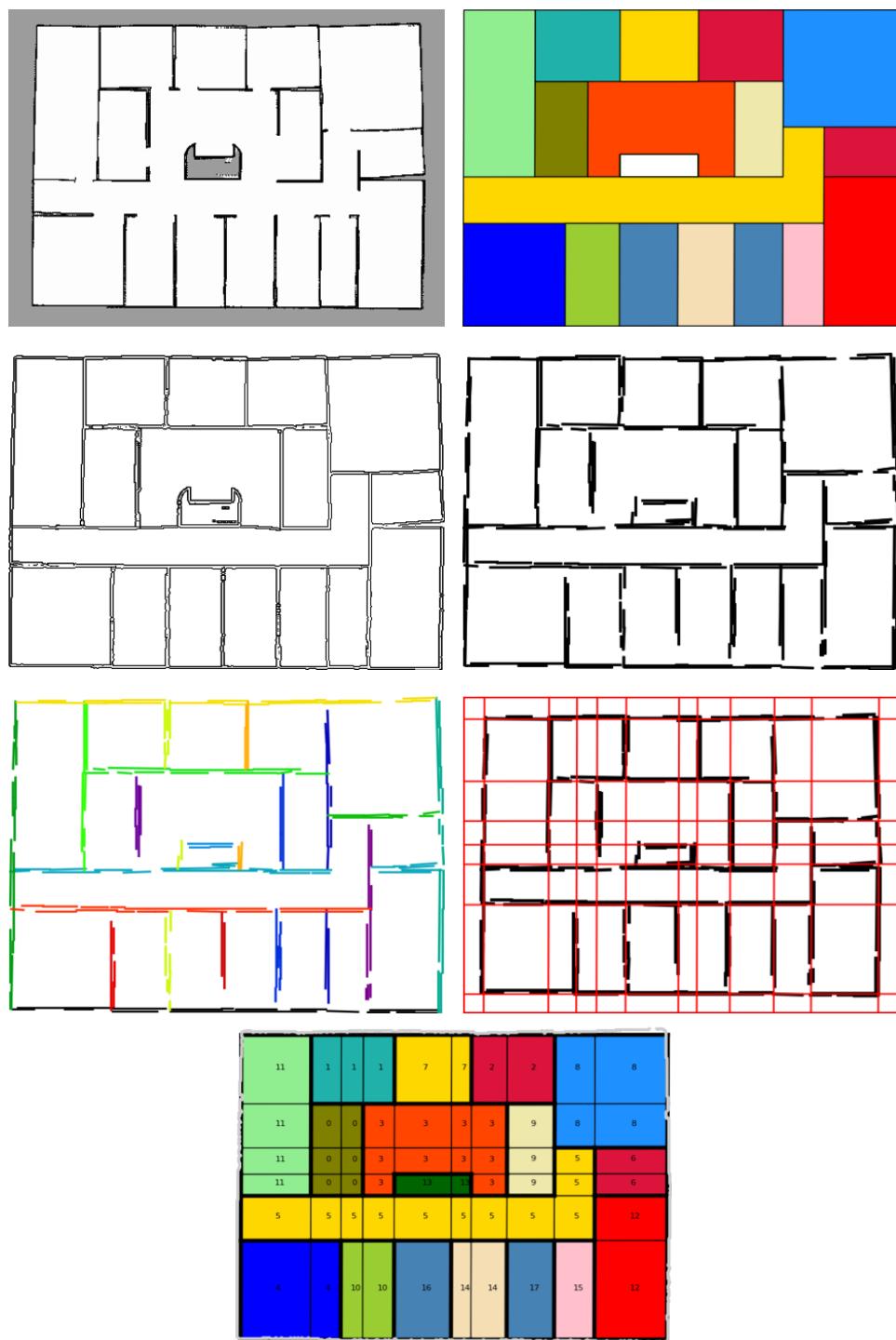


Figura A.42: Esempio 42.



Figura A.43: Esempio 43.

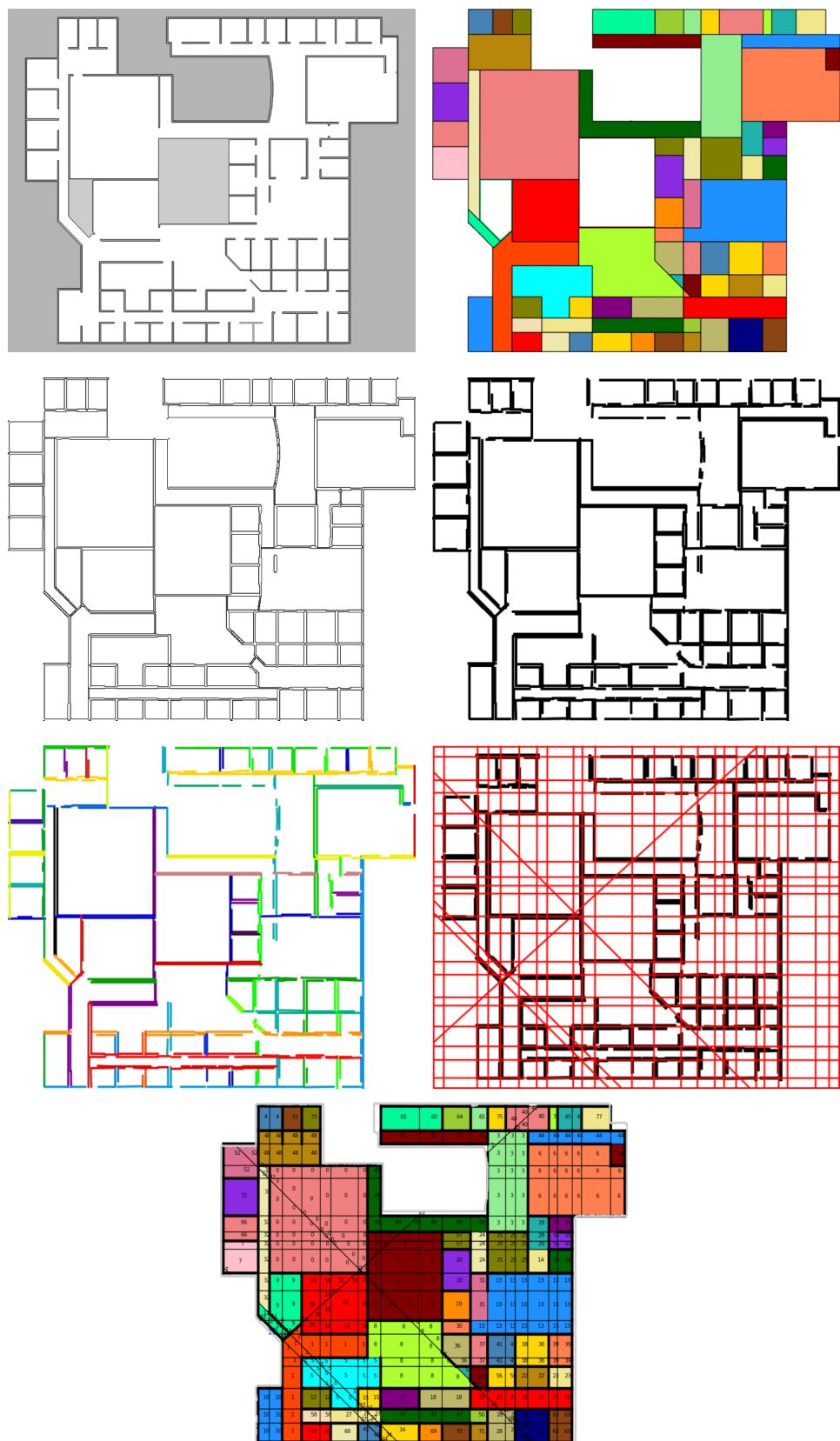


Figura A.44: Esempio 44.

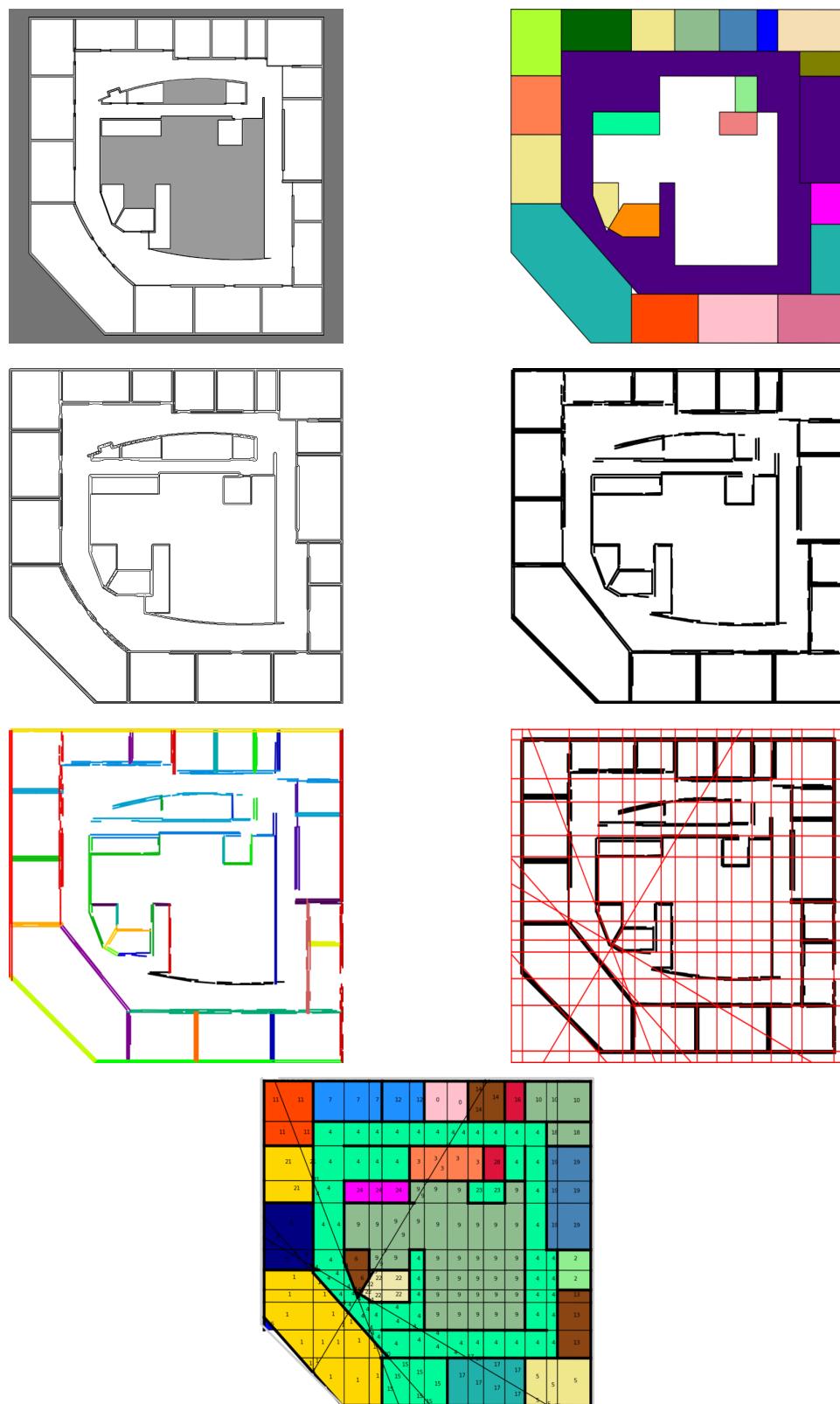


Figura A.45: Esempio 45.

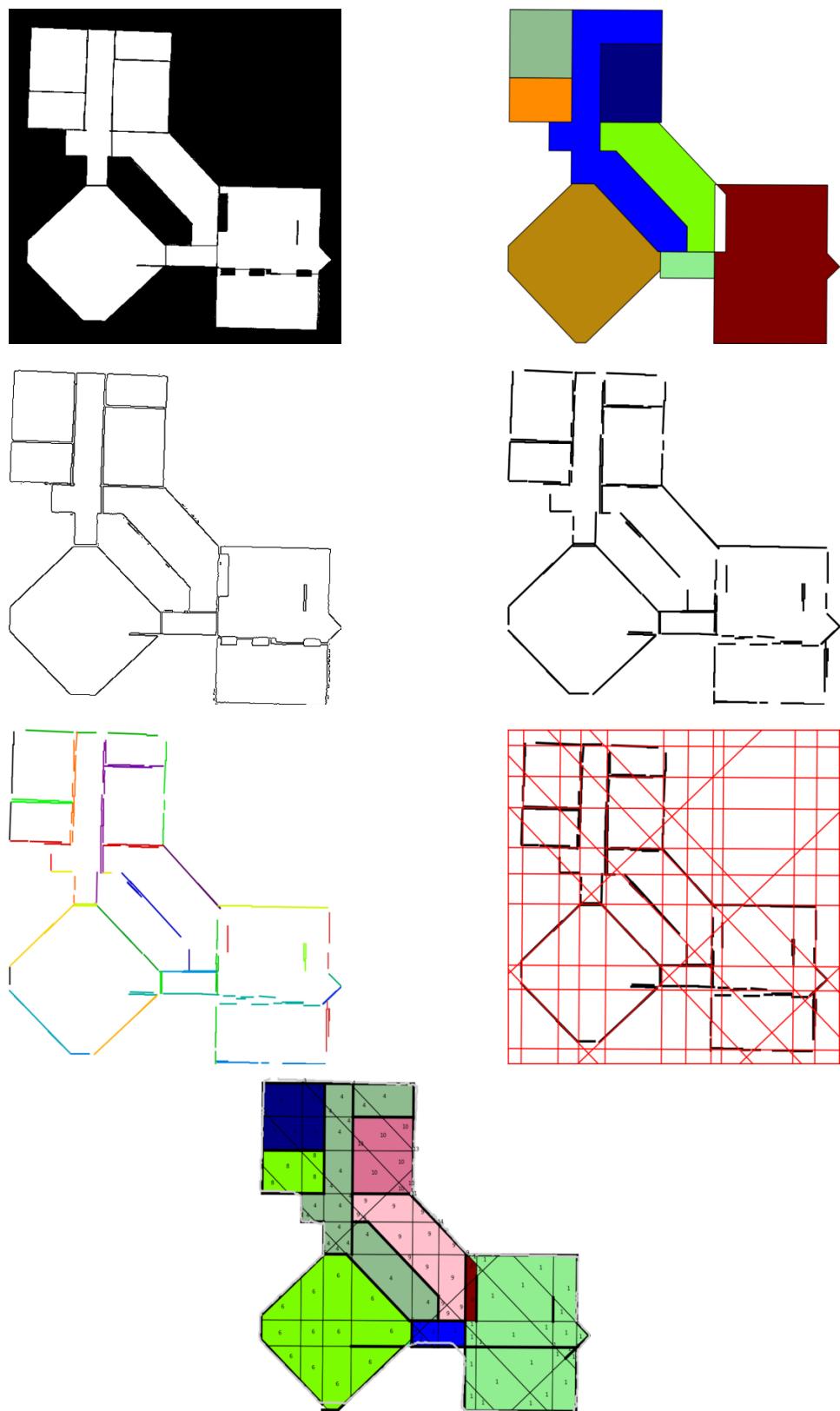


Figura A.46: Esempio 46.

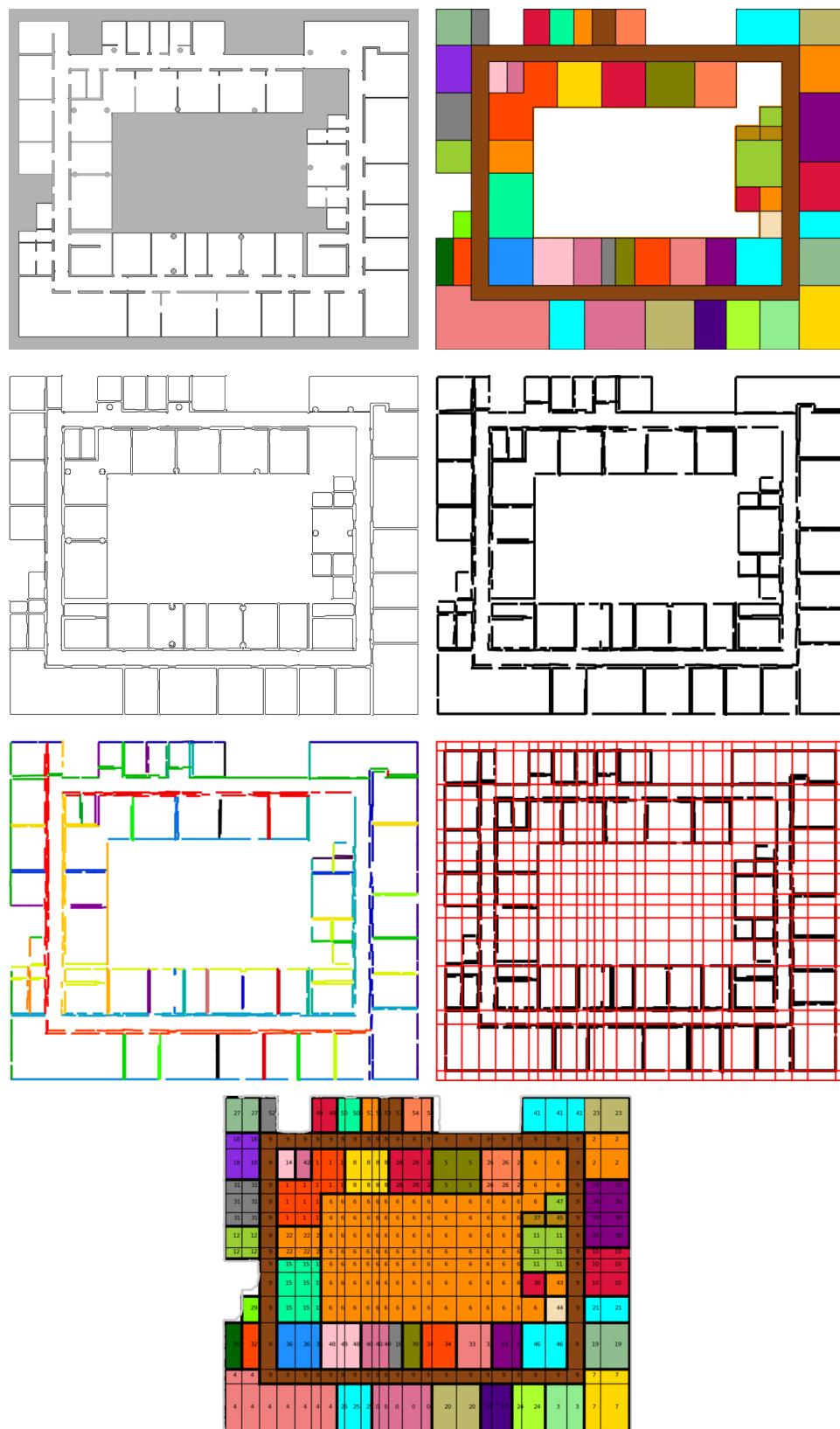


Figura A.47: Esempio 47.

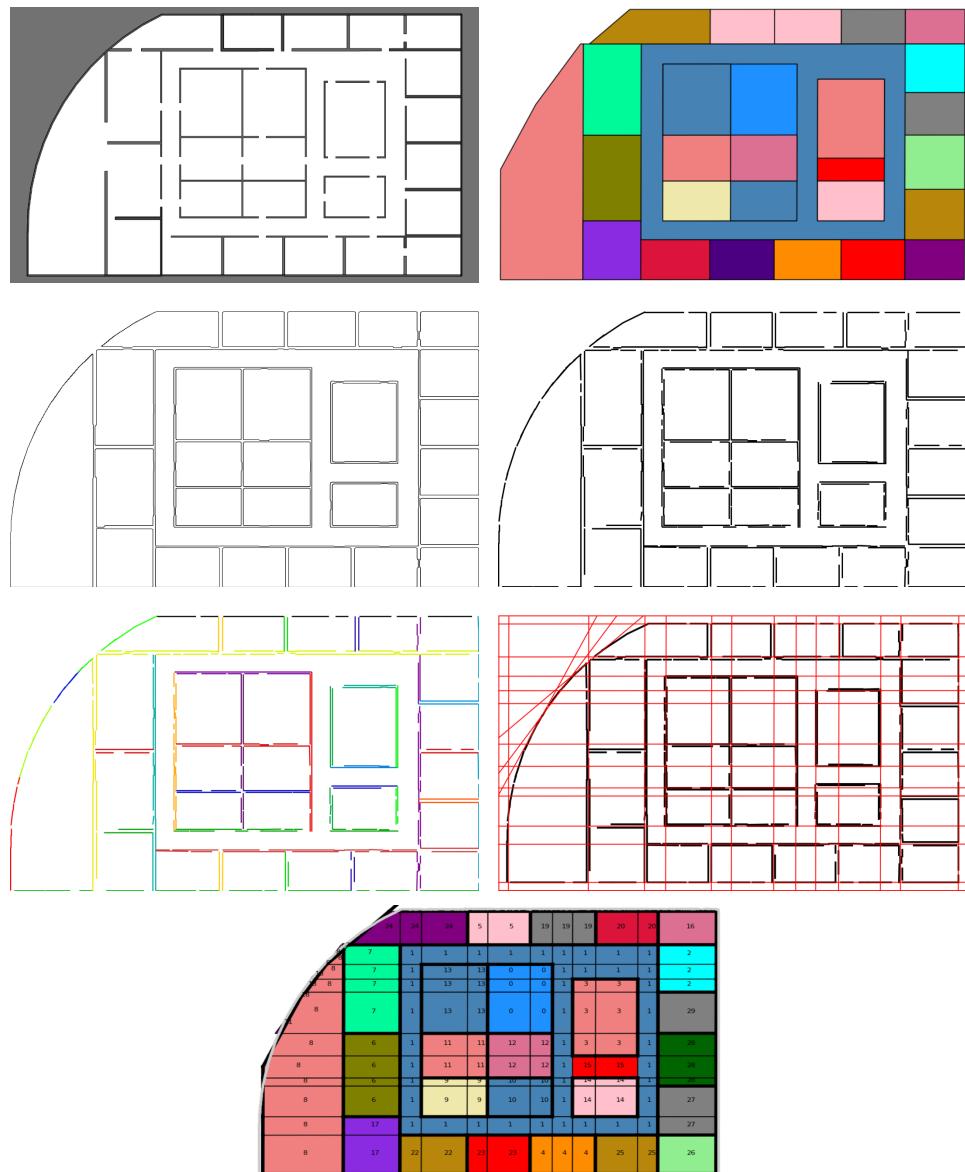


Figura A.48: Esempio 48.

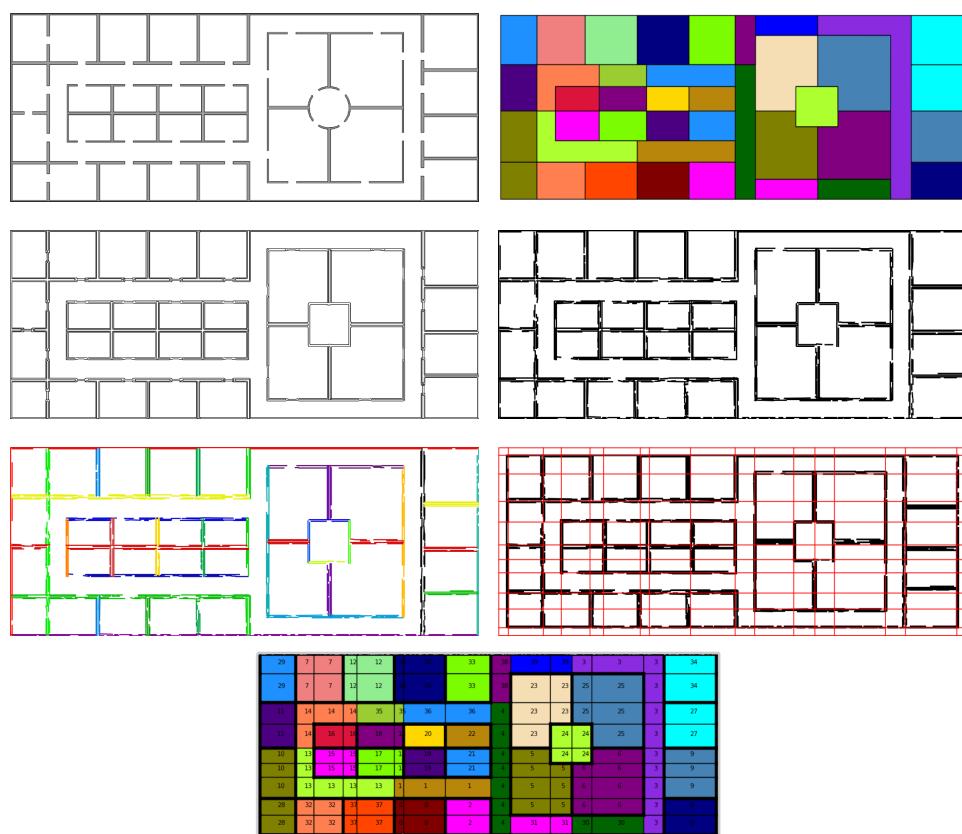


Figura A.49: Esempio 49.

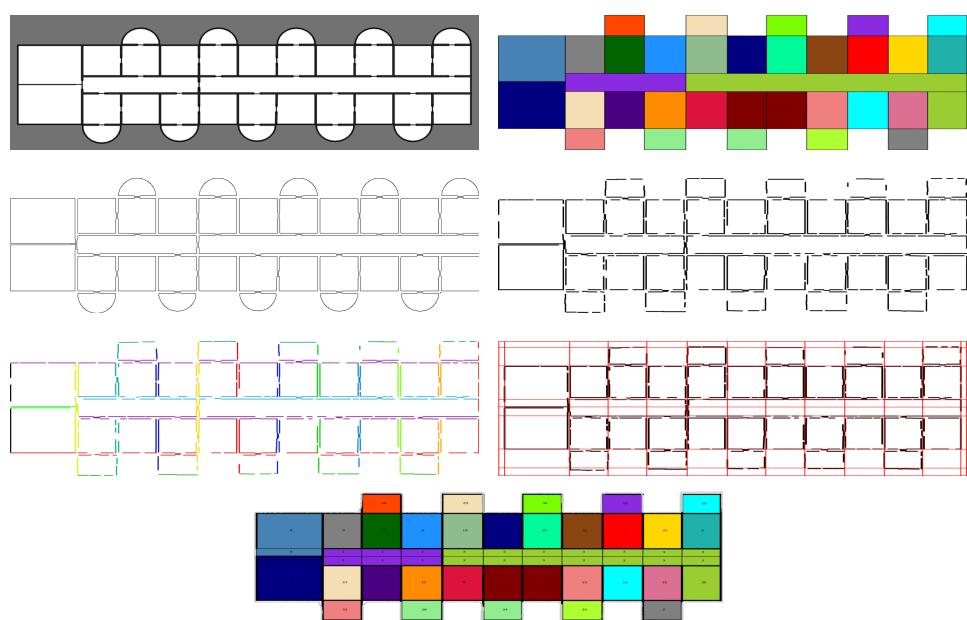


Figura A.50: Esempio 50.

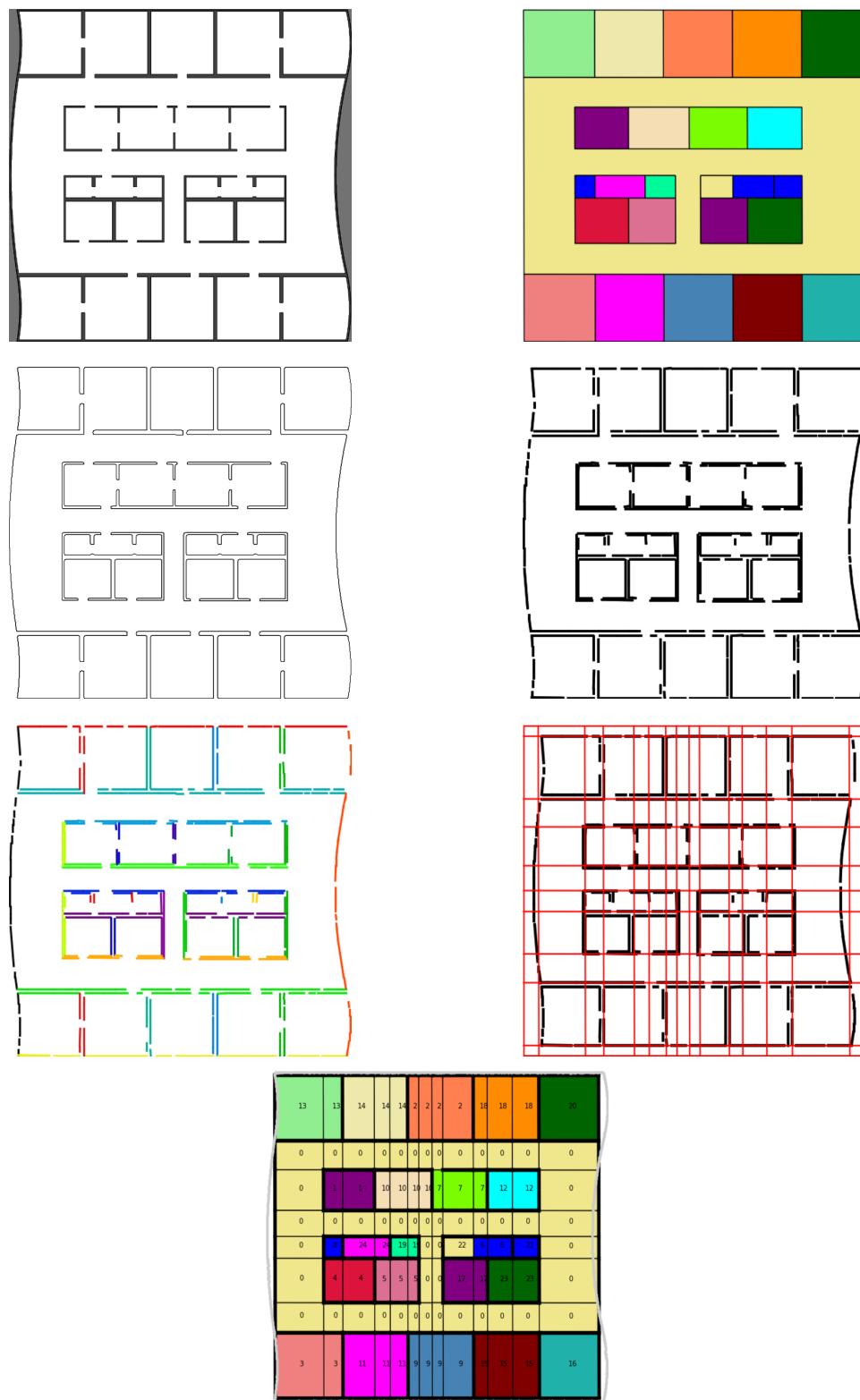


Figura A.51: Esempio 51.

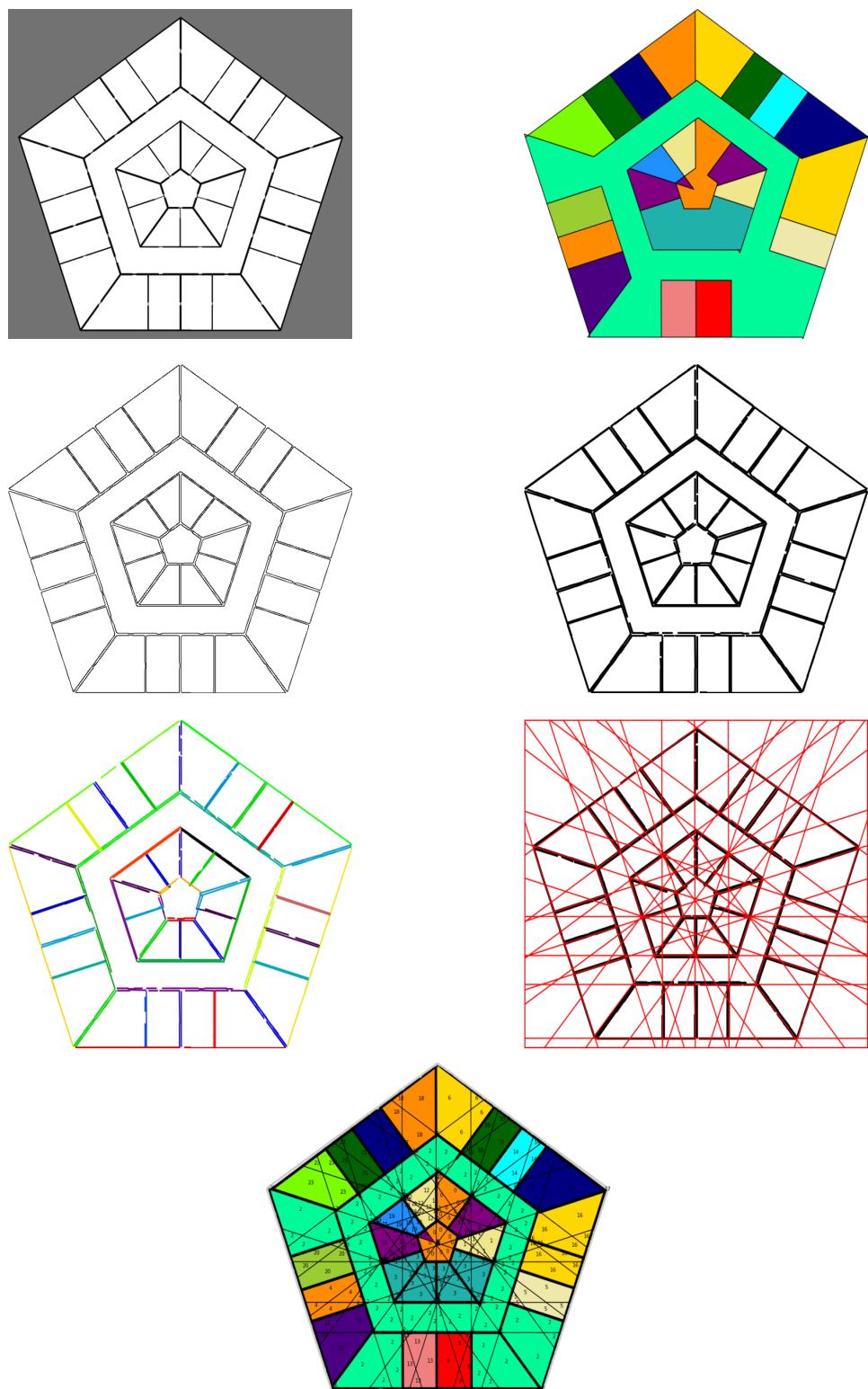


Figura A.52: Esempio 52.

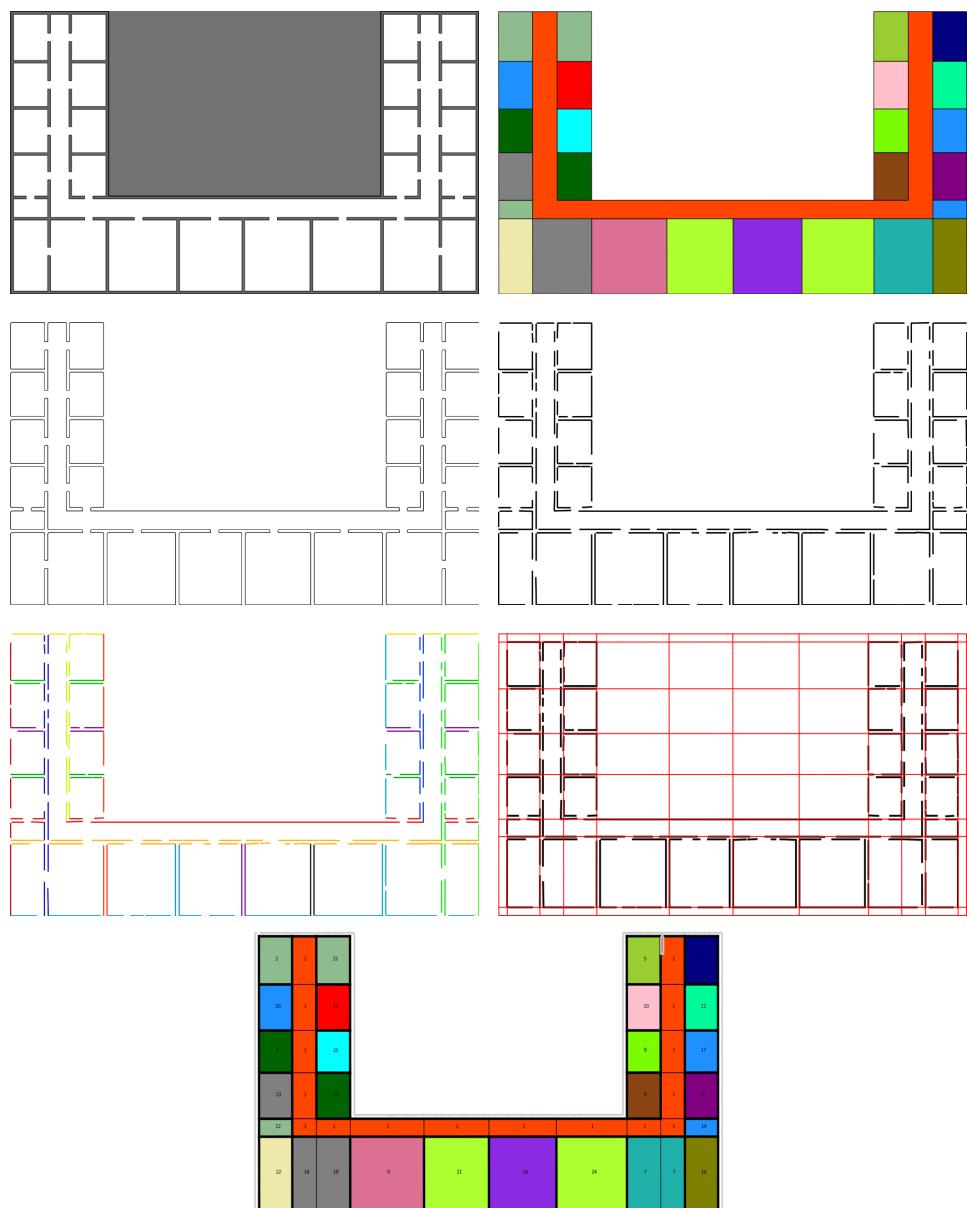


Figura A.53: Esempio 53.

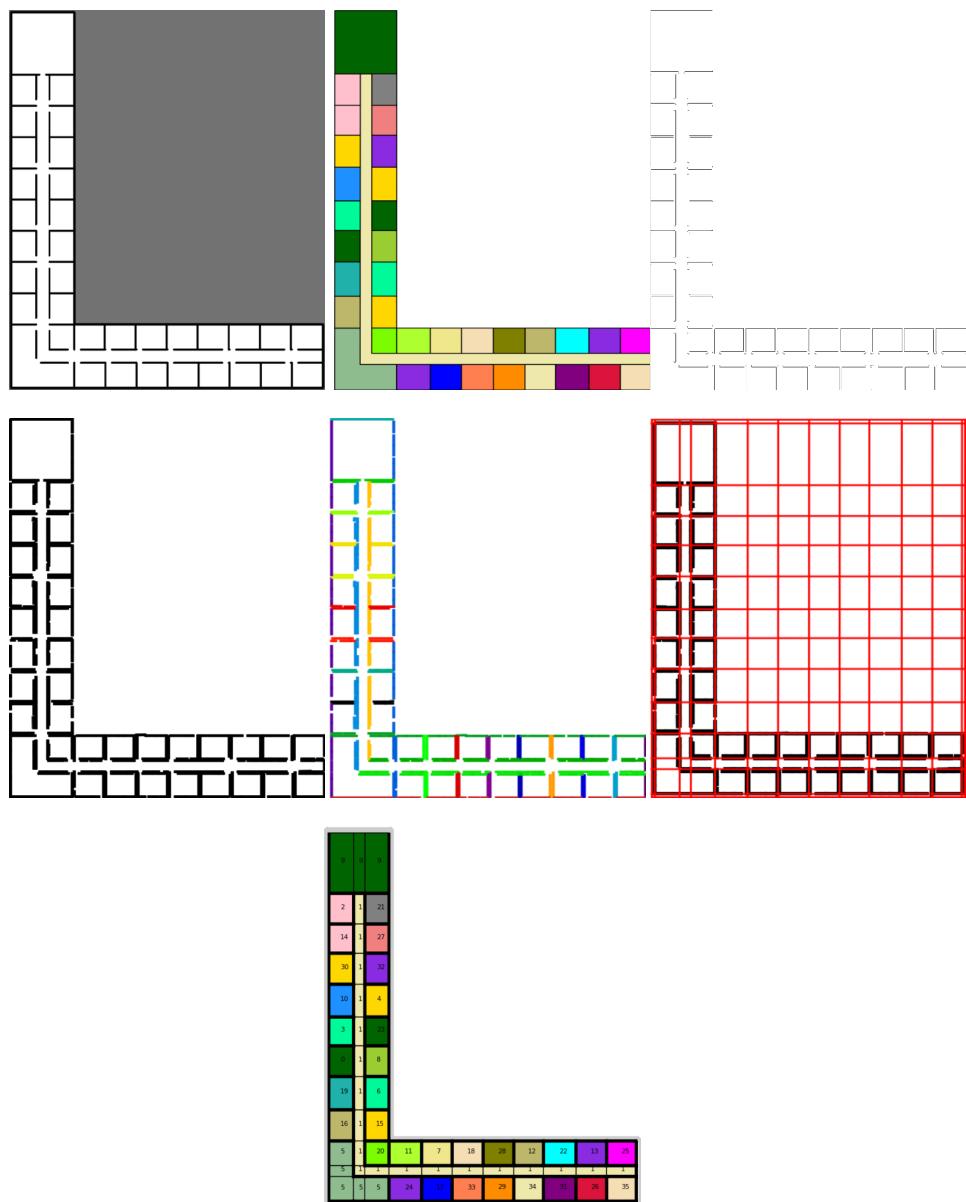


Figura A.54: Esempio 54.

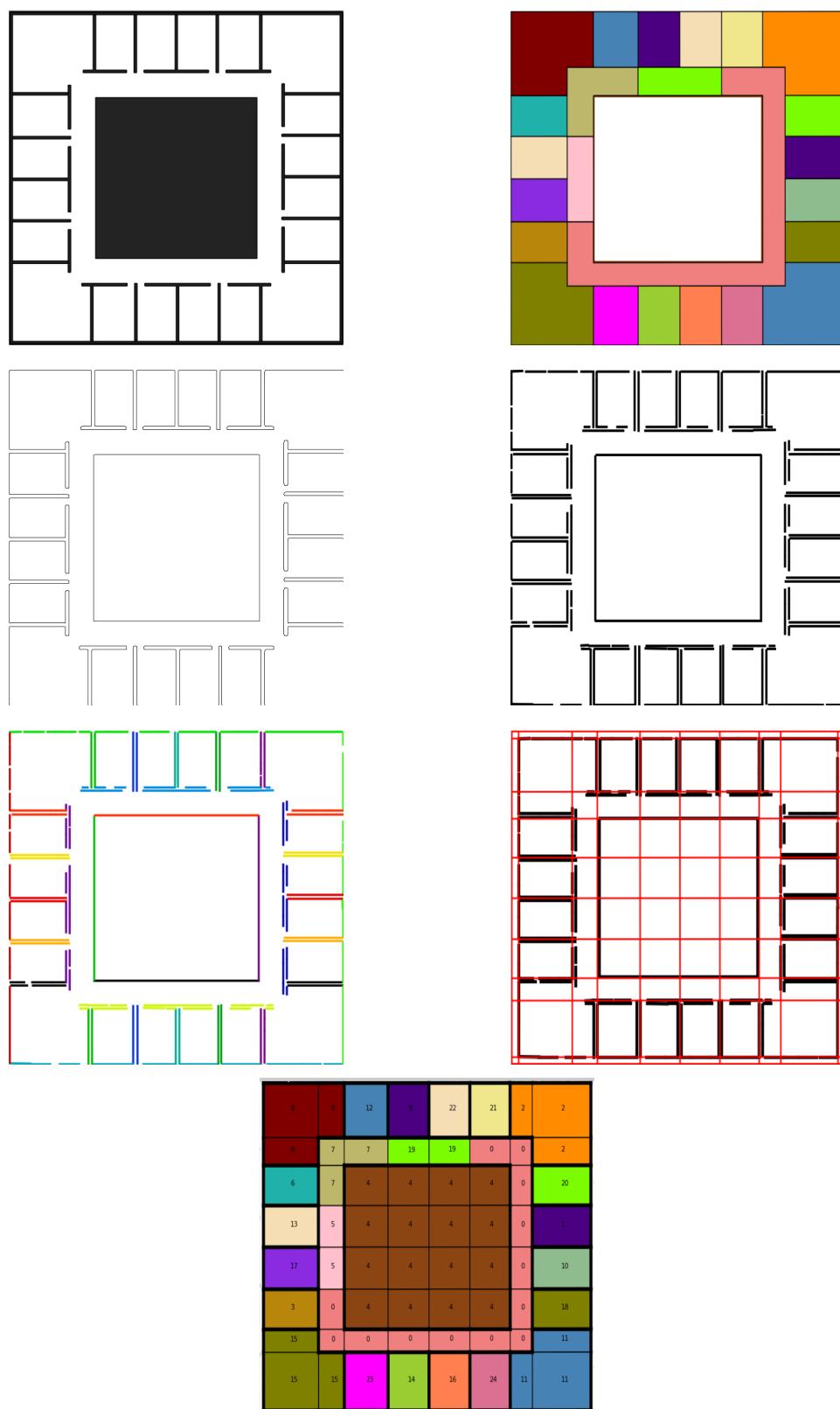


Figura A.55: Esempio 55.

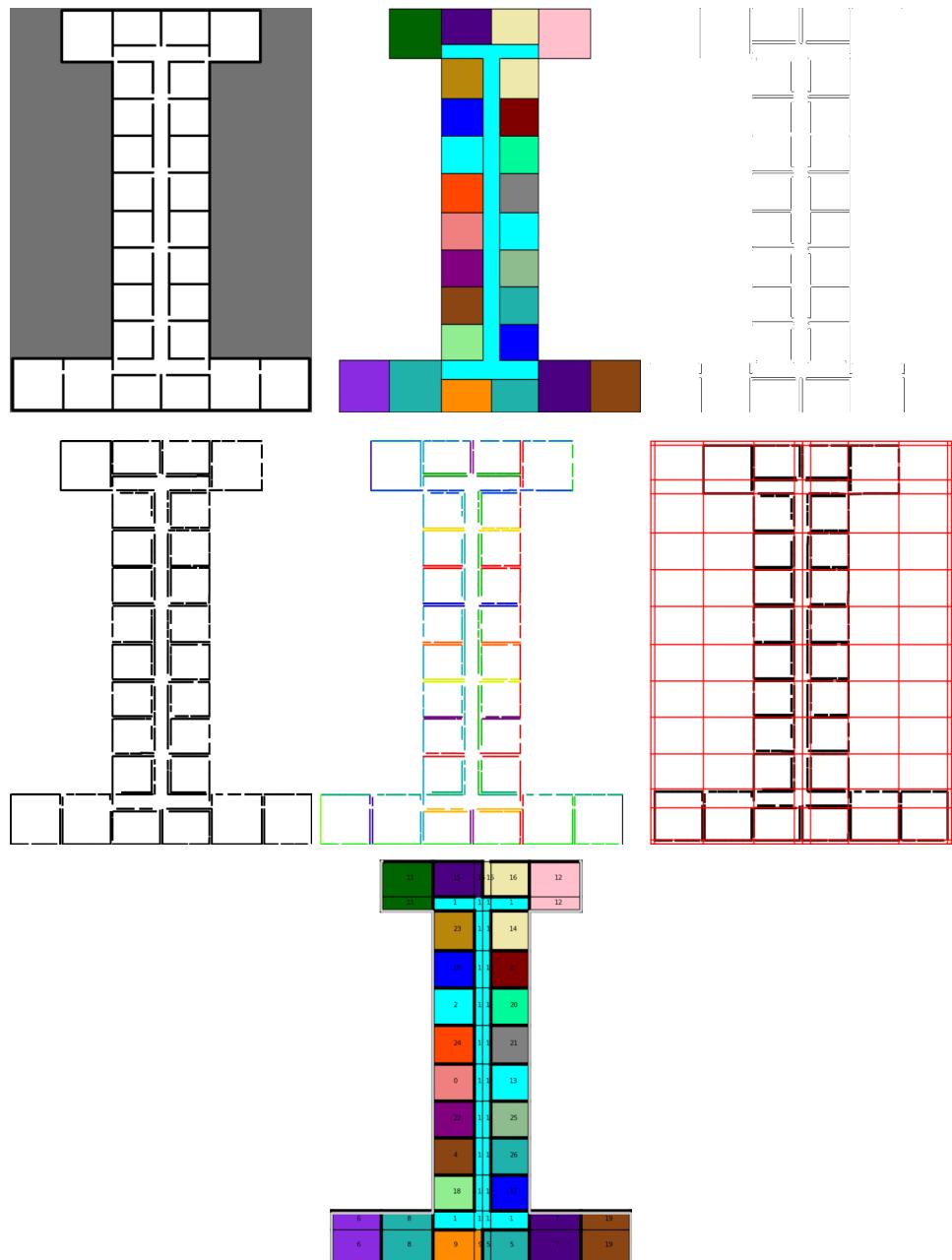
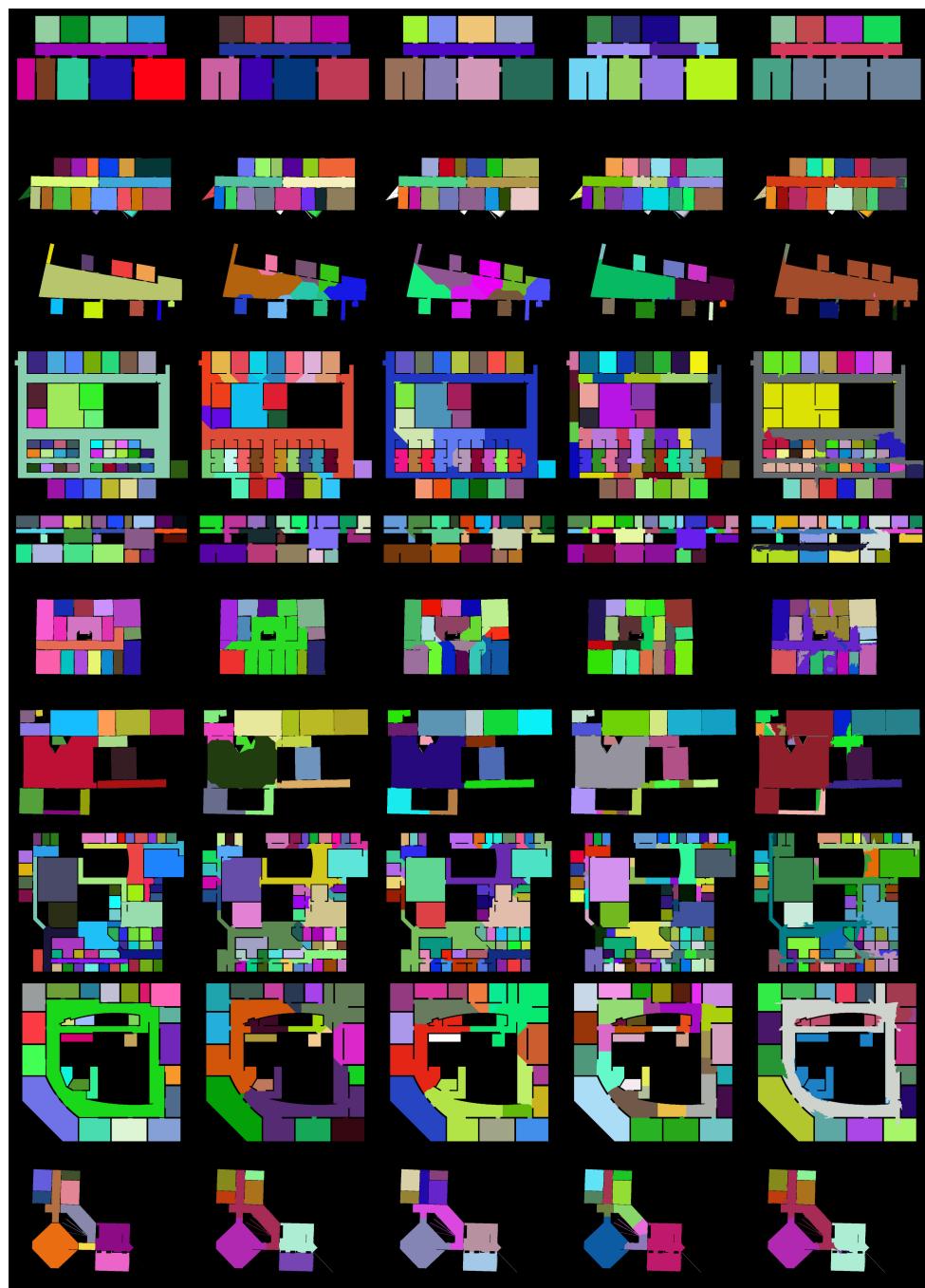
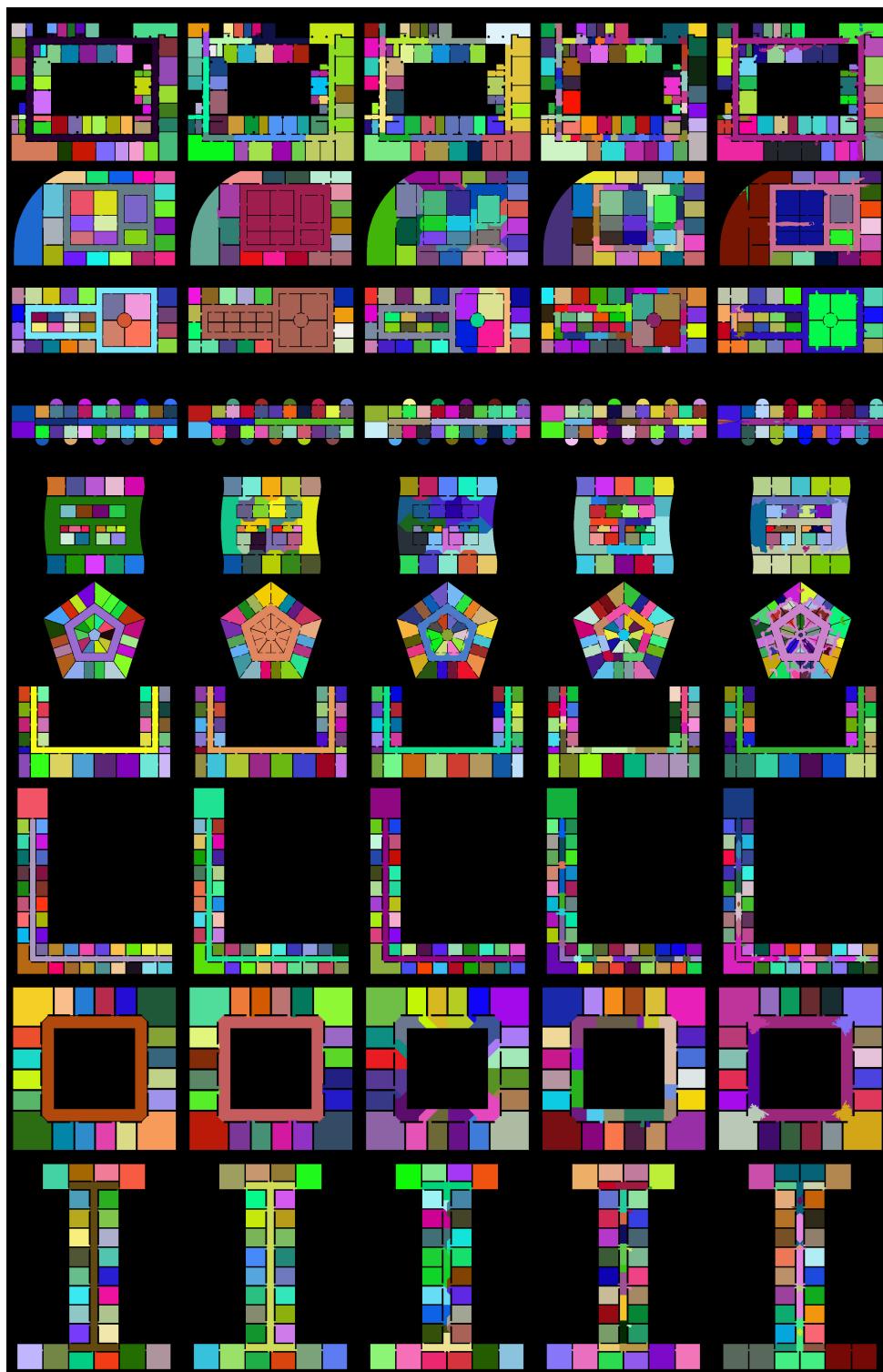


Figura A.56: Esempio 56.



*Figura A.57: Risultati di segmentazione tratti da [62] ottenuti applicando i metodi di segmentazione confrontati in [9]. Ad ogni stanza è associato un colore diverso. Le colonne consistono in (da sinistra a destra): mappa metrica di partenza, segmentazione morfologica, segmentazione basata su distance transform, segmentazione basata su Voronoi graph, segmentazione basata su feature.*



*Figura A.58: Risultati di segmentazione tratti da [62] ottenuti applicando i metodi di segmentazione confrontati in [9]. Ad ogni stanza è associato un colore diverso. Le colonne consistono in (da sinistra a destra): mappa metrica di partenza, segmentazione morfologica, segmentazione basata su distance transform, segmentazione basata su Voronoi graph, segmentazione basata su feature.*