

University of Genoa

DRIM

Ph.D. Program of National Interest in Robotics and Intelligent Machines

Administrative Headquarters: Università di Genova

Towards Fully Autonomous Inspection with Quadruped Robots in Unstructured and Safety-Critical Environments

by

Zoe Betta

Thesis submitted for the degree of *Doctor of Philosophy* (38° cycle)

December 2025

Antonio Sgorbissa — Supervisor and Head of the PhD program

Carmine Tommaso Recchiuto — *Supervisor*

Thesis Jury:

Vincent Frémont, *Ecole Centrale de Nantes* — External examiner

Nak Young Chong, *Japan Advanced Institute of Science and Technology* — External examiner

Emanuele Menegatti, *University of Padua* — Internal examiner

Dibris

Dipartimento di Informatica, Bioingegneria, Robotica e Ingegneria dei Sistemi

Declaration

I hereby declare that except where specific reference is made to the work of others, the contents of this dissertation are original and have not been submitted in whole or in part for consideration for any other degree or qualification in this, or any other university. This dissertation is my own work and contains nothing which is the outcome of work done in collaboration with others, except as specified in the text and Acknowledgements. This dissertation contains fewer than 65,000 words including appendices, bibliography, footnotes, tables and equations and has fewer than 150 figures.

Zoe Betta
August 2026

Acknowledgements

I am deeply grateful to my supervisor, Professor Sgorbissa, for their guidance, encouragement, and creativity throughout this research journey. I would also like to thank my co-supervisor, Professor Carmine Recchiuto, for their support and constructive feedback.

I would like to extend my sincere appreciation to the organizations and institutions with which I had the privilege of collaborating during this project.

I am particularly grateful to Dr. Robin Murphy for hosting me during my visiting period at Texas A&M University and for generously sharing her extensive experience in the real-world deployment of search-and-rescue robots, as well as the challenges associated with their use in emergency response scenarios.

I would like to thank all the staff of ANPAS (National Association of Public Assistance), the Civil Protection Department of the Umbria Region, and the Foligno Fire Brigade for their expertise, patience, and willingness to work with us in designing a feasible application. Their concerns, feedback, and insights were invaluable to this research.

I am also grateful to the Port Authority of Genoa for facilitating interactions with the various port terminals and for providing us with this opportunity. My sincere thanks also go to Terminal IMT, Terminal C. Steinweg - GMT S.r.l., and Terminal Rinfuse Genova for welcoming us and our robotic platforms, and for sharing their knowledge of the operational challenges within their environments.

Finally, I would like to thank Manolis Chiou, Tianshu Ruan, and Georgios Tzoumas for their expertise, collaboration, and the opportunity to contribute to the collection and analysis of insights from first responders around the world.

Abstract

Inspection operations in industrial and search-and-rescue (SAR) environments often expose workers to hazardous conditions, including unstable terrain, confined spaces, and partially inaccessible areas. In this context, autonomous robotic systems represent a promising solution for reducing human exposure to risk while increasing operational efficiency. Among mobile robotic platforms, quadruped robots offer significant advantages due to their mobility and ability to operate in complex and unstructured environments.

This thesis investigates the problem of autonomous inspection using quadruped robots in hazardous industrial and emergency scenarios. The work begins with a stakeholder-driven analysis involving first responders and industrial operators, with the objective of identifying operational requirements, expectations, and potential robotic applications for inspection tasks. Based on the collected requirements, the research focuses on the development of autonomous exploration and feature detection strategies for inspection-oriented missions.

To support autonomous inspection in multi-floor environments, the thesis proposes an exploration strategy aimed at maximizing environmental coverage under time constraints while accounting for the costs and risks associated with traversing different floors. In parallel, a probabilistic feature detection and localization framework is introduced to support the identification of relevant environmental features in inspection and SAR scenarios. The proposed method prioritizes robust and operationally meaningful detections over highly precise localization, reflecting the practical needs of first responders operating in time-critical conditions.

The thesis further investigates the integration of autonomous exploration and semantic feature detection, evaluating the influence of different exploration strategies on both environmental coverage and detection performance. The results demonstrate that exploration policies can significantly affect explored area coverage while maintaining comparable detection effectiveness.

To address more realistic operational conditions, the work proposes scenarios involving uneven terrain, dynamic obstacles, and task-level planning constraints. In particular, the thesis introduces a navigation framework that accounts for terrain slope and robot safety

during path planning, as well as an emotionally modulated local planner for navigation in dynamic environments. Finally, a PDDL-based planning framework is proposed to support inspection tasks in multi-floor emergency scenarios involving blocking obstacles and task coordination requirements.

The proposed approaches are validated through extensive simulation experiments and real-world testing with the Spot robot. Overall, the results demonstrate the feasibility of autonomous inspection using quadruped robots under realistic operational constraints, including limited time, partial observability, uneven terrain, dynamic obstacles, and high-level planning requirements. Collectively, this thesis contributes toward the development of autonomous quadruped robotic systems capable of supporting inspection operations in safety-critical real-world environments.

Table of contents

List of figures **x**

List of tables **xiii**

Nomenclature **xiv**

1 Introduction **1**

1.1 Problem statement 1
1.2 Research Objectives 3
1.3 Thesis structure 4
1.4 Contributions and publications 5

I Autonomous Exploration and Semantic Mapping for Inspection **9**

2 Related Works **10**

2.1 Robotic Inspection 10
2.1.1 Search and Rescue applications 11
2.1.2 Perception of stakeholders 13
2.2 Autonomous Exploration 14
2.3 Feature detection and localization 16
2.3.1 Measurement for object detection 17
2.4 Comparative Analysis Methodology 18
2.5 Conclusion 23

3 Qualitative Analysis of Stakeholder Requirements **24**

3.1 Study 1: Search and Rescue 25
3.1.1 Methodology 25

3.1.2 Results . . . 28
3.1.3 Discussion . . . 30
3.2 Study 2: Situational Awareness survey for FRs . . . 34
3.2.1 Methodology . . . 34
3.2.2 Results . . . 35
3.2.3 Discussion . . . 39
3.3 Study 3: Harbor inspection . . . 41
3.3.1 Methodology . . . 41
3.3.2 Results . . . 42
3.3.3 Discussion . . . 44
3.4 Conclusion . . . 47

4 Autonomous Exploration of Multi-Floor Buildings 49
4.1 Methodology . . . 50
4.2 Experiments . . . 54
4.3 Results . . . 56
4.4 Discussion . . . 57
4.5 Conclusion . . . 58

5 Probabilistic Semantic Mapping for Feature Localization 59
5.1 Methodology . . . 60
5.1.1 Detection Model . . . 61
5.1.2 Semantic Map . . . 62
5.1.3 Measurement . . . 65
5.2 Experiments . . . 67
5.2.1 Detection Model . . . 68
5.2.2 Semantic Mapping . . . 69
5.3 Results . . . 70
5.3.1 Detection Model . . . 70
5.3.2 Semantic Mapping . . . 71
5.4 Discussion . . . 74
5.5 Conclusion . . . 76

6 Integrating Exploration with Semantic Mapping 78
6.1 Methodology . . . 78
6.2 Experiments . . . 81

6.2.1 Simulated exploration . . . 83
6.3 Results . . . 84
6.3.1 Semantic Mapping . . . 84
6.3.2 Exploration and Search . . . 84
6.3.3 Simulated exploration . . . 86
6.4 Discussion . . . 88
6.4.1 Semantic Mapping . . . 88
6.4.2 Exploration and Search . . . 89
6.4.3 Simulated exploration . . . 91
6.5 Conclusion . . . 91

II Extensions Toward Robust Autonomous Systems 93

7 Related works 94
7.1 Robotic navigation . . . 94
7.1.1 Terrain aware navigation . . . 94
7.1.2 Emotion-Driven Navigation . . . 97
7.2 Symbolic planning in robotics . . . 98
7.2.1 Planning for quadrupeds in SAR scenarios . . . 101
7.3 Conclusion . . . 101

8 Slope-Aware Navigation Strategies 103
8.1 Methodology . . . 104
8.1.1 State sampling and validity checking . . . 106
8.1.2 Motion validity checking . . . 107
8.1.3 Finding a path . . . 110
8.1.4 System architecture . . . 110
8.2 Experiments . . . 111
8.2.1 Simulated experiments . . . 111
8.2.2 Real-world experiments . . . 112
8.3 Results . . . 112
8.3.1 Simulated Results . . . 112
8.3.2 Real-world results . . . 114
8.4 Discussion . . . 115
8.5 Conclusion . . . 116

9 Emotionally-modulated Artificial Potential Fields 118
9.1 Methodology . . . 119
9.1.1 Approach . . . 120
9.1.2 Implementation . . . 121
9.1.3 Anger . . . 122
9.1.4 Surprise . . . 123
9.2 Experiments . . . 125
9.2.1 Ablation experiments . . . 126
9.2.2 Comparison with other algorithms . . . 126
9.2.3 Dynamic obstacles . . . 127
9.2.4 Unstructured human interaction . . . 128
9.3 Results . . . 128
9.3.1 Ablation experiments . . . 128
9.3.2 Comparison with other algorithms . . . 129
9.3.3 Dynamic obstacles . . . 131
9.3.4 Unstructured human interaction . . . 133
9.4 Discussion . . . 134
9.5 Conclusion . . . 136

10 Search and Rescue Planning 137
10.1 Methodology . . . 138
10.1.1 Predicates . . . 138
10.1.2 Actions . . . 139
10.1.3 System architecture . . . 144
10.2 Experiments . . . 147
10.2.1 Simulated Actions . . . 148
10.2.2 Planning Evaluation . . . 153
10.2.3 Case study . . . 154
10.3 Results . . . 158
10.3.1 Planning evaluation . . . 158
10.4 Discussion . . . 161
10.5 Conclusion . . . 162

11 Conclusion 163
11.1 Summary of Contributions . . . 163
11.2 Limitations . . . 165

11.3 Future improvements 166

References **168**

List of figures

3.1 ANPAS and Civil Protection's Training Camp. 26
3.2 Data collection locations to evaluate the perceived motion abilities of the Spot robot from Boston Dynamics. 27
3.3 Results for the motion evaluation 29
3.4 Distributions of field experience and robot deployment level by age group. Robot deployment level refers to the frequency of using robots in the mission. Higher level means higher frequency. 35
3.5 Distribution of general attitudes towards robots on personal and societal levels. Most points are under the diagonal line (in the green zone). This indicates that most FRs have positive attitudes towards robots in both personal and societal contexts. 37
3.6 Score distributions of numerical questions. 38
3.7 Accuracy threshold values for achieving trust and usefulness: light blue bars refer to the accuracy threshold for achieving trust, and dark blue bars refer to the accuracy threshold for achieving usefulness. 40
3.8 Examples of locations at the Harbor of Genova where the robot can operate. 44

4.1 System Architecture. 50
4.2 Evaluation of x_f revenue gain. 53
4.3 Results of the exploration test: these maps have been created using the most performing configuration of exploration parameters. 55
4.4 Squared Meters explored changing the Floor Gain of the exploration. 56
4.5 Exploration rate comparison. 57

5.1 System Architecture. 61
5.2 Clearing area in front of the robot. 65

5.3 Spot (left) and ZED2 (right) cameras with a detected Person instance (Spot has two black&white frontal cameras.) 68
5.4 Spot System with ZED2 camera mounted. 69
5.5 Detected instances in two different locations. 71
5.6 Performance metrics at both locations. 73
5.7 IoU results for the different experiments 74

6.1 System architecture and interface. The Spot robot is equipped with a ZED2 camera connected to a Jetson Nano for GPU-based image processing, a Velodyne LiDAR, and a Spot Core for running all the required software modules. 80
6.2 Third location: environment and corresponding exploration results. 82
6.3 Simulator interfaces showing the bird's-eye environment and map views, together with the 3D first-person rendering. 84

8.1 Left: Climbing rocks at the Civil Protection camp in Foligno. Right: Carrying 6 bottles of water (9 kg) uphill in the DIBRIS Department's garden. 103
8.2 World frame $\langle w \rangle$, state frames $\langle s_1 \rangle$ and $\langle s_2 \rangle$, and related quantities. 106
8.3 System architecture implemented through ROS nodes (blocks) and topics (connectors). 109
8.4 Falling robot with $Phi = 10^\circ$ and $Theta = off$ in simulation. 113
8.5 Comparison of planned paths in simulation under different slope constraints. 113
8.6 Simulation results under different values of Φ and Θ. 114
8.7 Snapshots of Spot following a path uphill in different time instants. 114
8.8 Experimental results on the Spot robot under different Φ and Θ thresholds. . 115
8.9 Statistics about ϕ_r values over the motion of the robot. 115

9.1 Experiments at Zachary building with a robot and people exhibiting unpredictable behaviors. 119
9.2 Perceived obstacles and resulting repulsive force. 122
9.3 Effects on the repulsive field given the anger level of the robot. 124
9.4 Effects on the repulsive field given the anger level of the robot. 125
9.5 Experimental setting for comparing different local navigation algorithms and evaluating dynamic obstacles. 127
9.6 Ablation experiment with only Anger active. 128
9.7 Ablation experiment with only Surprise. 129

9.8 Ablation experiment with both Anger and Surprise active. 130
9.9 Comparison of performance metrics across experiments. 130
9.10 Distance measured each 30 seconds comparing the different algorithms. . . 131
9.11 Trajectory of the robot, with color representing the Anger level and linewidth representing the Surprise level. 132
9.12 Trajectory of the robot, with color representing the Anger level and linewidth representing the Surprise level. 134

10.1 System's architecture for the experiments with the Spot robot with a limited number of actions. 145
10.2 Simulated environments. 149
10.3 Real scenario on the second floor of the E building. 156
10.4 Frames from the implementation of a subset of actions on the Spot robot, simulating a rescue operation. 157
10.5 Planning time analysis under different environment configurations. 159
10.6 Planning performance under different complexity factors. 161

List of tables

2.1 Classification of recent SAR robotic works according to functional and operational characteristics. Rows are hierarchically ordered from left to right: for each criterion, entries without a checkmark precede those with a checkmark, while preserving the ordering established by the preceding criteria. 18

3.1 Questionnaire items related to the i-th area 28
3.2 Summary of unstructured interviews 31
3.3 Emerged Problems 45
3.4 Possible Solutions 46

5.1 Performance for different YOLOv5 models. 71
5.2 Results for the Semantic Mapping experiments. 75

6.1 Results for the Semantic Mapping experiments. 85
6.2 Area explored for the Exploration and Search set of experiments. 85
6.3 Floors visited, floors changed and actual time explored for the Exploration and Search set of experiments. 86
6.4 Results for the Exploration and Search experiments. 87
6.5 Results for the simulated Exploration and Search experiments. 88

9.1 Times to reach WPs in the Dynamic scenario 132

10.1 Predicates of the domain 139
10.2 PDDL actions 140
10.3 Parameters used for problem generation. 155
10.4 Success percentages for each parameter. 161

Nomenclature

Acronyms / Abbreviations

AI Artificial Intelligence

ANPAS National Association of Public Assistance

APF Artificial Potential Fields

CTL Computation Tree Logic

ECG Electocardiogram

EMAPF Emotionally Modulated Artificial Potential Fields

FN False Negative

FP False Positive

FR First Responder

GUI Graphical User Interface

HTN Hierarchical Task Network

IMU Inertial Measurement Unit

IoU Intersection Over Union

LLM Large Language Models

LPG Local Search for Planning Graphs

LTL Linear Temporal Logic

MCC Matthews Correlation Coefficient

ML Machine Learning

MODA Multi-Object Detection Accuracy

OMPL Open Motion Planning Library

PDDL Planning Domain Definition Language

R-CNN Region-Based Convolutional Neural Networks

Ro-Ro Roll-on/Roll-off

ROS Robot Operating System

RRT Rapidly-exploring Random Tree

SA Situational Awareness

SAR Search and Rescue

STRIPS Stanford Research Institute Problem Solver

TAMP Task and Motion Planning

TEB Timed Elastic Band

TN True Negative

TP True Positive

UAV Unmanned Aerial Vehicle

UGV Unmanned Ground Vehicles

VLA Visual Language Action Models

VLM Vision-Language Models

VR Virtual Reality

WP Waypoints

YOLO You Only Look Once

Chapter 1

Introduction

1.1 Problem statement

Inspection operations are essential in several application domains, including industrial maintenance, infrastructure monitoring, [117] and search-and-rescue (SAR) missions [143]. These activities often require operators to work in hazardous, inaccessible, or unstable environments, exposing personnel to significant safety risks. Industrial inspection scenarios may involve confined spaces, elevated structures, toxic substances, or damaged facilities. For example, [52] details a solution in which the Spot robot is deployed to inspect the small tunnels of the Paris metropolitan system. In this case, the robot has to traverse small tunnels to detect structural and thermal anomalies; the use of a mobile robot ensures that operators are not directly exposed to risks. On the other hand, SAR environments are frequently characterized by debris, uneven terrain, reduced visibility, and uncertain structural conditions [145]. As a result, improving worker safety during inspection tasks has become an increasingly important research topic in robotics and autonomous systems.

The growing demand for frequent and reliable inspection procedures has further increased interest in robotic solutions capable of operating remotely in dangerous environments. By increasing the physical distance between human operators and hazardous sites, robotic inspection systems can reduce direct human exposure to risk. Furthermore, autonomous robotic systems can enhance operational efficiency by performing tasks alongside trained personnel, enabling human workers to focus on higher-level decision-making and specialized interventions.

In inspection scenarios mobility is crucial, and depending on the environment, different robotic platforms have been deployed. Crawling robots, [6] have been deployed when the robot has to traverse very small passages. Wheeled robots [190], even if frequently deployed,

pose some challenges when the environment is not flat, and there is a need to move on top of obstacles. Unmanned Aerial Vehicles (UAV) [144], instead, provide very fast coverage of areas, but are limited by the current regulations in several environments, are not adequate for indoor environments, and require distance to humans. Among the various mobile robotic platforms, quadruped robots have emerged as a promising solution for inspection applications due to their advanced mobility capabilities. Unlike wheeled robots, quadruped platforms can traverse uneven terrain, climb obstacles, and operate in unstructured environments that are commonly encountered in industrial and SAR scenarios. Their locomotion capabilities enable deployment in areas that may be inaccessible to traditional robotic systems, making them particularly suitable for inspection tasks in complex real-world environments.

Despite recent advances in autonomous robotics, enabling quadruped robots to perform fully autonomous inspection missions remains a significant challenge. Inspection tasks require more than simple navigation through an environment. A robotic system must autonomously explore unknown areas while simultaneously detecting, localizing, and recording relevant environmental features or points of interest. This requires the integration of autonomous exploration, perception, localization, and environment understanding under uncertain operating conditions.

Existing research on autonomous exploration has primarily focused on generic exploration tasks, often limited to structured indoor environments or single-floor scenarios. Many approaches prioritize coverage of unknown areas without considering the specific requirements of inspection missions. In addition, current exploration strategies are frequently not optimized for operational constraints such as limited mission duration. As a result, existing systems may successfully explore an environment while failing to provide meaningful support for real-world inspection workflows. Similarly, in vision-based systems for feature detection, the primary focus is often on precise spatial localization rather than on maximizing the probability of correctly detecting the presence of relevant features or objects in the environment. More importantly, in literature, it is not mentioned the advantage of knowing that an area is correctly identified as empty is not mentioned, an important requirement for SAR scenarios, where First Responders (FRs) can safely ignore empty areas.

Therefore, there remains a need for autonomous robotic systems capable of performing inspection-oriented exploration in hazardous and unstructured environments. In particular, further research is required to enable quadruped robots to autonomously navigate unknown environments while detecting and localizing inspection-relevant features in a manner that supports both industrial inspection and SAR operations.

1.2 Research Objectives

This work focuses on the development of strategies to enhance inspection processes in both industrial settings and challenging, unpredictable environments such as post-disaster scenarios. Given the specialized nature of these applications and the need for close interaction with domain experts and field operators, the first stage of this research involved identifying the operational needs and requirements of each application through direct collaboration with practitioners.

Interviews with field operators revealed that autonomous exploration combined with environmental information acquisition is a fundamental requirement in both SAR and industrial inspection scenarios. In particular, a critical capability in these contexts is the detection and spatial localization of relevant environmental features during operation.

The first objective of this research is to design an efficient exploration strategy for multi-floor buildings and to analyze how different system parameters influence the coverage and efficiency of the exploration process. The second objective is to develop a vision-based approach capable of robustly localizing relevant features in 3D space during autonomous exploration.

To address these objectives, we propose a framework based on a layered probabilistic grid-based semantic map that is incrementally updated as new detections are acquired. The proposed mapping system enables the fusion of spatially consistent observations while increasing confidence through repeated detections over time.

During the development of the proposed framework, several challenges and potential improvements emerged. Traditionally, robotic navigation is divided into two main components: global navigation and local navigation. Global navigation assumes knowledge of the environment map and computes an optimal path toward a target location. Local navigation, by contrast, focuses on the robot's immediate surroundings and is responsible for path following and obstacle avoidance.

Conventional global navigation approaches often model the environment as a two-dimensional space and do not explicitly account for terrain traversability. To address this limitation, this thesis, in the third objective, investigates a navigation strategy capable of considering hazardous slopes and non-traversable areas during path planning.

Local navigation methods must ensure progress toward the assigned goal while maintaining safe distances from static and dynamic obstacles, such as moving people. In unpredictable environments, robots may become trapped, fail to reach their target, or exhibit unsafe behaviors leading to collisions. To mitigate these issues, this research explores a local navigation

strategy based on Artificial Potential Fields (APF) modulated through an emotion-inspired mechanism aimed at improving the robot's survivability and adaptability in uncertain conditions. Objective four is to evaluate whether emotion-inspired modulation can effectively influence the behavior of a non-social robot, enabling safer and more robust navigation in dynamic and unpredictable environments.

Finally, the analysis of tasks identified in collaboration with field operators showed that autonomous exploration and spatial feature detection represent only part of the overall operational requirements. In realistic scenarios, multiple tasks may need to be coordinated and executed simultaneously. Even during simple exploration, a quadruped robot may encounter a closed door or an obstacle blocking its path that can be removed by planning appropriate actions, for example, using an external robotic arm to operate the door handle and push the door open. For this reason, this thesis, objective five, also investigates whether a classical planning-domain formulation can adequately model and manage complex, dynamic, and unpredictable environments such as those encountered in SAR operations.

1.3 Thesis structure

This thesis is divided into two main parts. Part I presents the core contributions of the thesis, focusing on the proposed methodology for collecting the requirements of specialized workers and designing autonomous robotic inspection strategies tailored to their operational needs. Part II, instead, presents additional contributions aimed at extending and improving the proposed approaches by considering more challenging and unstructured environments.

Part I begins with Chapter 2, which reviews the state of the art in robotic inspection applications, with particular focus on autonomous exploration, feature detection, semantic mapping, and localization techniques relevant to industrial and SAR scenarios.

Chapter 3 presents the analysis of stakeholders' requirements, perceptions, and operational needs regarding the use of robotic systems for inspection tasks in partially structured environments. In particular, this chapter details the methodology adopted for collecting feedback from specialized workers and discusses the resulting operational requirements that guided the development of the proposed robotic solutions.

Based on the requirements identified through stakeholder analysis, Chapter 4 introduces an autonomous exploration strategy specifically designed for multi-floor buildings (objective 1). The proposed approach focuses on maximizing environment coverage within a limited mission time while accounting for the costs and risks associated with traversing different floors.

Chapter 5 proposes a strategy for the detection and localization of relevant environmental features (objective 2). The approach is based on probabilistic representations of the environment and integrates information from multiple detections to improve robustness. In particular, the strategy prioritizes reliable approximate localization over highly precise but potentially less robust detections, reflecting the practical requirements of inspection and SAR operations.

Finally, Chapter 6 presents the integration of autonomous exploration and feature detection within a unified inspection framework. The proposed system is evaluated in representative application scenarios, analysing the interaction between exploration strategies, environmental coverage, and detection performance.

Part II begins with Chapter 7, which reviews relevant literature in robotic navigation, traversability analysis, local path planning, and symbolic task planning for autonomous robotic systems operating in complex environments.

Chapter 8 introduces a navigation framework for environments containing slopes and uneven terrain (objective 3). The proposed approach explicitly accounts for robot safety constraints and traversability considerations in the path-planning process, with the objective of improving navigation robustness in challenging conditions.

Chapter 9 presents an emotionally modulated local path-planning strategy based on APF. The chapter analyses the influence of different emotional states on navigation behaviour and evaluates the proposed method in the presence of dynamic obstacles (objective 4).

Chapter 10 proposes a Planning Domain Definition Language (PDDL) domain designed according to the operational requirements identified from FR (objective 5). The chapter demonstrates the feasibility of symbolic task planning for robotic inspection missions in realistic multi-floor emergency scenarios through extensive simulation experiments.

Finally, Chapter 11 concludes the thesis by summarizing the main contributions, discussing the limitations of the proposed approaches, and outlining possible directions for future work.

1.4 Contributions and publications

Robotic inspection refers to the deployment of autonomous or semi-autonomous robotic systems to systematically examine environments, infrastructure, or objects in order to detect anomalies and collect data. These tasks require several core components of a robotic framework to operate together. Robots must be able to navigate safely in environments with a strong human presence while simultaneously collecting various types of data or performing specific tasks that improve access to relevant information.

Given the specific requirements of each application, the first step in developing an inspection framework was to gather insights from workers by collecting their needs and expectations. In particular, we aimed to obtain a comprehensive list of the tasks that humans—and, in the future, robots—must perform during inspection activities. I was responsible for designing a semi-structured interview protocol, conducting the interviews, and analyzing the resulting data. Using a similar procedure, we interviewed both FRs regarding inspection tasks in SAR environments, as well as harbor workers, to capture a more industrial perspective on inspections specific to harbor operations. The results from the interview with FR are published in [18], while the results from the interviews with FR have been presented as a Work in Progress to UR2025, [17]. I also had the opportunity to collaborate on the design and administration of a questionnaire aimed at collecting insights into the role of semantic information in emergency scenarios; the results are published in the following [165].

After conducting these interviews, I focused on developing the core capability components required for autonomous inspections. In particular, I was responsible for the design, experimental procedures, and analysis of the results for the following components:

- **Autonomous Exploration**

  After a building evacuation, it cannot be assumed that the map of the building is known; some areas may be inaccessible due to closed doors or fallen debris. Similarly, in industrial environments, the map is not always fully known a priori. Materials can be moved, causing areas to appear different—particularly in harbor environments, where containers are moved daily, continuously changing the surroundings. For this reason, an autonomous exploration algorithm is required. In particular, a system that can account for multiple floors and support different exploration strategies based on the specific requirements is needed. The results have been published in [22].

- **Probabilistic Semantic mapping**

  While moving through the environment, the robot must recognize and localize relevant features. We propose a probabilistic semantic mapping algorithm that uses computer vision to detect features relevant to SAR environments and localize them, while accounting for the increased confidence afforded by multiple detections. The results are presented in [23].

- **Integration of autonomous exploration and semantic mapping**

  We integrated the autonomous exploration algorithm with the probabilistic semantic mapping framework to study the feasibility of the approach. In particular, the

performance of the semantic mapping was evaluated by comparing it with different exploration behaviors. This work has been submitted to the Journal of Field Robotics, and is currently under review (major revision submitted) [21].

During my visiting period at Texas A&M, at Robin Murphy's Laboratory, I had the opportunity to work on the task of autonomous navigation in the presence of unpredictable environments. During an inspection task, especially in the context of SAR, the environment is not known, and people are frantically moving around, not considering the presence of the robot. For this reason, we focused on the development of a robust and safe local navigation algorithm in SAR contexts.

- **Robust navigation in unpredictable environments**

  When navigating in unknown environments, classical navigation algorithms may be insufficient, especially in the presence of dynamic obstacles. Inspired by the survival function of emotions, we developed a local navigation algorithm based on Emotionally Modulated Artificial Potential Fields (EMAPF). This algorithm enables the robot to move toward a goal without relying on a previously known map while reacting to dynamic obstacles. The results have been accepted to the Simulation of Adaptive Behavior 2026 conference [20].

Additionally, I had the opportunity to collaborate with several Master's thesis students who worked on developing other core capability components. I was responsible for collaborating with them on the design of the applications, supporting them during the experimental phases, and contributing to the preparation and presentation of the work for publication for the following components:

- **Slope-aware navigation**

  While navigating the environment, the robot may encounter steep slopes that could lead to malfunctions. Traditional navigation algorithms typically assume flat terrain and full traversability. By collecting information from the Light Detection and Ranging (LiDAR) sensor and representing it in an elevation map, we developed a planner that enables the robot to maintain safe slope levels while traveling toward the desired goal. This has been published in [210].

- **Planning system for SAR scenarios**

  As mentioned earlier, inspection tasks generally consist of multiple sub-tasks that must be coordinated. We propose a PDDL-based framework to fully simulate a SAR

scenario. The goal was to evaluate the performance and limitations of the planning framework in a SAR environment with multiple requirements. This is under review (major revision) for Robotics and Autonomous Systems [19].

Finally, I assisted my colleague with experiments for his work, which aimed to improve inspection tasks through immersive teleoperation rather than autonomous capabilities. From this collaboration, we have one work published [209] and one under review for Transportation Research Interdisciplinary Perspectives [211].

Part I

Autonomous Exploration and Semantic Mapping for Inspection

Chapter 2

Related Works

This chapter presents an analysis of the literature relevant to the first part of this thesis. It begins with a review of robotic inspection applications in both industrial and emergency-response scenarios. The chapter then examines existing approaches for assessing stakeholders' perceptions and acceptance of robotic solutions. Subsequently, the literature on autonomous exploration in hazardous environments is reviewed. Next, works addressing the detection and localization of environmental features are discussed, focusing on both the proposed methodologies and the metrics used for their evaluation. Finally, a structured review of studies that integrate autonomous exploration and feature detection within SAR contexts is presented.

2.1 Robotic Inspection

In Italy, 590,215 occupational injuries were reported in 2023, including 1,147 fatal cases [1]. To reduce the consequences of workplace accidents, research efforts have increasingly focused on robotic systems for autonomous infrastructure inspection [63, 207]. Various robotic platforms have been adopted for industrial inspection tasks, including wheeled robots, aerial drones, legged systems, and purpose-built devices. Certain robotic solutions are highly application-specific—for instance, those developed for pipe inspection [101, 148]—and are therefore not easily transferable to other contexts. Wheeled robots and aerial robots face significant limitations in their range of application, specifically requiring quasi-flat terrain and open aerial spaces that are free from people and obstacles. Quadruped robots, in contrast, are frequently highlighted for their versatility, terrain adaptability, and locomotion capabilities [182, 151]. As an example, the Spot robot was deployed in 2022 at Austria's largest power plant to improve operational safety [2], autonomously patrolling predefined

routes and reporting detected anomalies. Similarly, the Spot robot is used in the metro network of Paris to inspect hard-to-reach tunnels, evaluating the presence of cracks or structural damages [52].

While robotic inspection in civil environments has expanded in recent years, other domains have not been equally considered. This work, following the PNRR RAISE project in which I was involved, explores harbor settings—particularly landside ground operations—that remain relatively underexplored. Lattanzi et al. [117] provide a survey of inspection applications in domains such as buildings, bridges, and tunnels, yet harbors are not explicitly addressed. Other studies, such as those by Choi et al. [37] and Tani et al. [185], focus on underwater robotic systems for inspecting submerged port infrastructure. Santos et al. [173] propose a cooperative inspection approach involving drones and underwater robots, in which drones autonomously land on a surface vehicle after completing their tasks.

2.1.1 Search and Rescue applications

A subcategory of inspection concerns post-disaster scenarios, particularly in the context of SAR operations. Compared to industrial applications, these environments are significantly more challenging, as they involve higher uncertainty, dynamic conditions, and stricter operational requirements.

After a catastrophe such as an earthquake or fire, there is an urgent need to respond quickly and effectively due to the hazardous conditions faced by those affected. Tragically, many lives are often lost during rescue operations because of the limited time available to act, leaving individuals trapped or without sufficient time to escape the disaster. For example, Ritchie et al. [159] reports that in the last 10 years on average, 40,000 to 50,000 people die each year from natural disasters. On the other side, rescuers themselves are often affected by unpredictable events that occur during rescue operations. For instance, according to [150], 232 fire and rescue workers in the U.S. suffered fatal work injuries between 2011 and 2015.

Emergency scenarios are typically characterized by uneven, unfamiliar terrain and numerous uncontrollable obstacles that can impede escape or directly endanger lives. While professionals, such as firefighters, respond as quickly as possible, their primary priority is ensuring their own safety before initiating rescue operations. This, coupled with the difficulty of locating victims, underscores the importance of prioritizing the rescue of survivors. However, this is not always straightforward, as identifying and reaching survivors can be challenging, particularly if they are trapped under debris or severely injured. Therefore, despite the best

efforts of rescuers, additional technological tools–such as robots–are essential to enhance performance and reduce the number of fatalities and injuries.

In these tasks, robots can be either teleoperated [132] or autonomous [126]. In both cases, robots must prioritize human safety, ensuring that their actions do not introduce additional risks to people [43]. Teleoperated robots rely on reliable communication between the controller and the robot to efficiently transmit commands and information[196]. However, this can introduce latency, which may be mitigated by placing the operator closer to the robot; however, such proximity poses safety risks for the operator. Autonomous robots have the advantage of not relying on a communication link with operators but have more demanding requirements in terms of onboard capabilities, including artificial intelligence (AI) and machine learning (ML) algorithms for data processing and planning [180].

Either teleoperated or autonomous, different types of robots are suited to various environments and conditions. One category is underwater robots, which have been used to find drowned victims after the 2011 Tohoku Earthquake and Tsunami [91]. The demand for robots in such situations is high due to extreme conditions, including low temperatures and poor visibility.

Another type of robot used in SAR situations is aerial robots, also known as UAVs, 'aerial drones,' or simply 'drones' [129]. Among these, quadcopters are the most well-known for SAR operations, equipped with various sensors that enable them to efficiently cover large areas, albeit with limitations due to battery constraints. Thanks to their ability to fly, drones can rapidly survey vast regions, making them particularly useful for SAR operations in large outdoor environments. An example is the use of drones to detect and localize victims of avalanches proposed by Mimmo et al. [139].

The last category of robots used in SAR operations are Unmanned Ground Vehicles (UGVs), or simply 'ground robots'. While they are generally slower than drones, UGVs are better suited for indoor environments where precise navigation in confined spaces is essential. Due to their larger battery capacity, UGVs can operate for extended periods without needing recharges, which makes them ideal for long-duration missions. Moreover, ground robots can be equipped with a variety of sensors to support SAR operations, including LiDAR, high-resolution cameras, speakers, and thermal cameras [51]. Up until 2014, ground robots were the most frequently used robots in SAR scenarios [143]. The main challenge for ground robots in emergency scenarios is mobility. Several missions failed due to mobility issues; for example, the first configuration of the iRobot Packbot could not climb a steep metal staircase to access the environment after the Fukushima nuclear disaster [145]. With the development

of quadruped robots, notable improvements in mobility in hazardous terrains can be seen [205].

These robots are particularly promising for SAR missions because their legged structure enables them to climb stairs, maneuver through narrow or low-height spaces, and traverse rubble, mud, or water. Their versatility, adaptability to various terrains, and ability to be equipped with advanced sensory systems [42] make them the preferred choice in environments where ground conditions are highly uncertain due to disaster damage.

2.1.2 Perception of stakeholders

Due to the challenges inherent in inspection tasks—particularly in SAR scenarios—it is essential to involve end users in the design process of the proposed solutions. In the SAR context, several studies evaluate robotic systems from the perspective of end users, namely FRs. These works typically present an engineering solution that is subsequently tested and assessed through user-based evaluations.

In [121], a mobile robot was deployed and teleoperated inside tunnels, with FRs tasked to qualitatively evaluate the utility of 2D and 3D maps in a real-world scenario. A frequent drawback of these methodologies is the lack of involvement of responders in the design phase; they are brought into the process only after the system's implementation, rather than engaging in a collaborative design approach from the outset. In [104], the authors developed two interfaces for robot teleoperation: one based on a computer and the other utilizing a joystick. To evaluate these, 100 firefighters were enlisted to test both systems and subsequently complete a questionnaire assessing the ease of use, usefulness of both systems, and the requirements for remote controllers in emergency scenarios. However, this approach once again excludes end users from the design process, as it measures the users' perceptions of system usability only after full development. Similarly, the authors of [194] delve into the comprehensive design of a mobile robot and its control interface. To assess its effectiveness, they outline a method for evaluating (i) the workload, (ii) the degree of telepresence (i.e., the sensation of being physically present in the environment observed by the robot), and (iii) Situational Awareness (SA). For this assessment, 70 participants, including some firefighters, were tasked with conducting a simulated SAR operation. They were required to navigate through an unsafe building to locate casualties over a period of 30 minutes. To assess factors (i), (ii), and (iii), different questionnaires were administered to the participants after the simulation concluded. Notably, SA was evaluated both during the task and after its completion.

Additional crucial factors to consider are those affecting rescuers' trust in the robot, with trust being defined as the confidence in the robot's capability to successfully perform the necessary operations. In [102], the study examines the dynamics of end-users' or rescuers' trust in robotic systems as influenced by variations in system latency and autonomy levels. The research involved 80 participants who were tasked with completing objectives within a simulation game, under conditions of either remote control or semi-autonomy for the robot, and with varying degrees of communication latency. Following the task, participants completed questionnaires designed to assess the robot's trustworthiness, usability, and the perceived workload experienced during the operation. This approach aims to understand the factors that can impact rescuers' confidence in the robot's ability to successfully execute required operations.

From the viewpoint of individuals being rescued, it is essential to assess their acceptance of robot intervention and their opinions on the use of robots for such purposes. Both [187] and [50] delve into public perceptions regarding the deployment of drones, whether remotely operated or autonomous, focusing specifically on those who, while not controlling the drones, would benefit from their assistance. In [50], the researchers distributed a 25-item questionnaire to beachgoers following the demonstration of a drone rescue operation via a video presentation. Meanwhile, [187] engaged participants through an online survey, presenting two variations of the questionnaire: one referencing 'drones' and the other 'unmanned aerial vehicles.' This survey aimed to evaluate various dimensions of subjective drone perceptions influenced by prior knowledge, identification, and classification of drones, emphasizing the emotional response to drones used for general assistance or specific tasks like SAR. Also, it collected opinions on particular controversial and problematic use cases.

In all the examples mentioned above, evaluations were conducted through video recordings of operations or system simulations rather than having visual observation of the robot operating.

2.2 Autonomous Exploration

An essential capability in inspection scenarios—where a map may be unavailable or incomplete—is autonomous exploration. This allows the robot to generate an updated map that can subsequently be used for further operations.

Autonomous exploration with mobile robots has been studied extensively. One of the most commonly used approaches is frontier-based autonomous exploration [208]. This methodology relies on identifying the boundary areas between known and unknown regions

of the map. Frontiers are then evaluated based on the expected information gain from reaching a given frontier [193], the distance of the frontier from the robot's current position [62], or a combination of both metrics [55, 90].

Tranzatto et al. [188] propose a frontier-based exploration approach for robots, both UGVs and UAVs, tasked with exploring harsh underground environments. In their work, frontiers are evaluated based on the volumetric expected information gain along the path to reach each frontier. The method also accounts for the finite duration of the exploration mission, estimating the remaining mission time when evaluating each frontier. Due to the high computational overhead, the system is controlled by a base computer with WiFi communication. In a SAR scenario, however, it cannot always be assumed that a centralized controller is available or that a WiFi connection can be established. Among other works [60, 186, 69], Zhang et al. [213] propose a topological map that evaluates detected frontiers based on both the topological relationship between the frontier and the robot's position, as well as the Euclidean distance, to better estimate exploration cost. This approach does not consider the information gain in reaching the frontier. Wang et al. [200] propose an algorithm that evaluates frontiers based on the information gain of the entire path to reach them. While this is an effective approach, it also entails high computational overhead.

In the aforementioned works, exploration was limited to a single floor: the robot operated in an enclosed environment entirely on the same level. However, when developing applications for indoor buildings, it is important to account for the presence of staircases and multiple levels.

The work presented in [164] focuses on the exploration of an environment by two robots using a leader-follower paradigm that leverages their different capabilities. Only one of the robots is capable of moving on uphill ramps; consequently, when an elevation change is detected, this robot is tasked with exploring the different level. In the work of Kudriashov et al. [110], the focus is on constructing a map that accounts for the multiple floors of a building. A common focus in multi-floor autonomous building exploration is the strategy for traversing staircases and reaching the next floor. In particular, Garret et al. [72] propose a teleoperated robot capable of autonomously switching gaits when approaching stairs to complete the task of changing floors. Another study [162] presents a system that can detect stairs and evaluate their traversability. In this system, the robot autonomously explores one floor of the environment and climbs the stairs to a new floor only after the current floor has been fully explored. This analysis shows that, to our knowledge, there are no efficient multi-floor strategies in the literature. The task of exploring a multi-floor building is treated as a sequence of single-floor explorations, without considering the building as a whole.

2.3 Feature detection and localization

While operating in the environment, it is essential to acquire information about application-relevant features in real time, thereby enabling prompt responses in the event of emergencies.

In the past years, due to the development of neural networks, such as R-CNN (Region-Based Convolutional Neural Networks), Fast R-CNN, and variants, several different solutions have been developed for the recognition of objects with real-time capabilities, e.g. [66, 65, 34]. Among the most popular solutions in robotics due to its real-time performance is YOLO [157], You Only Look Once, a family of algorithms that allows for accurate object detection with real-time performance. YOLO algorithms can perform two kinds of tasks: object detection and object segmentation. Both of them aim at detecting objects in the scene and classifying said objects; the main difference is that object detection algorithms only return the bounding box containing the object while segmentation returns the mask of the pixels belonging to the detected object. Segmentation then allows for much more information to be retrieved, but as a result, is computationally heavier when compared with object detection tasks. The YOLO algorithms have been employed for several tasks. For example, the work of Kucukayan et al. [109] is targeted at developing a new model specifically designed for indoor people recognition using drones. This model improves people detection in SAR scenarios where the visibility may be suboptimal. However, the main drawback of this application is the lack of a method to localize the detected people, since the method only allows classification.

Some studies propose strategies for exploring an environment with the specific goal of locating human survivors in the shortest possible time. For example, [178] describes a strategy for exploring an unknown environment that considers several application-specific risk factors. In particular, when evaluating the next frontier, the strategy takes into account not only the typical factors considered in frontier-based exploration, but also the estimated likelihood of confirming a survivor's presence.

Other studies focus on the detection of people in an area using drones [181, 171, 216]. Sruthi et al. [181] describe a YOLO-based approach for detecting people in the area covered by a teleoperated drone, with an emphasis on precise localization using GPS coordinates. Similarly, Sambolek et al. [171] propose a YOLO-based model to detect a single person in the environment, localize them, and predict their trajectory if they are moving. Another line of research focuses on improving detection of people who are partially occluded by obstacles in disaster scenarios [216] and the detection of people using thermal images [29].

It is important to note that none of these works consider repeated detections of people—or other relevant features—to build a grid-like probabilistic map of the environment that updates,

through subsequent detections, the probability that a person is actually present rather than representing sensor noise.

A map that incorporates the representation of objects' meaning and purpose within an environment is typically referred to as a semantic map. Semantic maps merge the spatial information of obstacles in the environment with semantic information regarding the objects' nature [112, 30, 212]. Recognizing objects in the scene can be meaningful for enabling the robot with advanced navigation capabilities that go beyond the usual movements on almost flat surfaces.

For example, the author of [138] proposes a system capable of extracting information about terrain traversability and the presence of stairs from the point cloud data. The authors showed that their approach ensures more precise loop closures while also providing the possibility to remove dynamic obstacles from the map by exploiting the segmentation map generated from the model.

Object segmentation has been used in the work of Schonberger et al. [177] to improve localization. By employing a 3D representation of the environment and a semantic map, the robot can localize itself more accurately while navigating, adapting to changes in the environment while correctly computing loop closures.

In the work proposed by Canh et al. [31], a UAV aims at improving the metric map by incorporating information about encountered obstacles. Obstacles are classified using a YOLO-based model and object-specific attributes, such as the height of a chair, are merged into the navigation map.

In the aforementioned works, information derived from semantic maps has primarily been exploited to enhance the robot's locomotion capabilities, either by improving localization and mapping performance or by enabling additional capabilities, such as stair climbing.

2.3.1 Measurement for object detection

In the SAR scenario considered here, identifying empty areas is almost as important as locating relevant features, as it helps save FRs' time. Unfortunately, similar works aimed at detecting and tracking the location of relevant features focus on evaluating errors without accounting for the benefit of recognizing empty spaces in the environment. In computer vision applications, the primary metric for segmentation problems is the Intersection over Union (IoU) [158], which could ideally be used to measure the overlap between a grid-like semantic map of the environment and a corresponding ground-truth map reporting the exact locations of features to be detected. However, this metric relies on highly precise feature

Table 2.1 Classification of recent SAR robotic works according to functional and operational characteristics. Rows are hierarchically ordered from left to right: for each criterion, entries without a checkmark precede those with a checkmark, while preserving the ordering established by the preceding criteria.

Paper	SAR-specific	Visual feature detection	Feature Localization	Exploration	Multi-floor capable	Multi-floor aware	Victims-and-cleared space evaluation
[164], [162]	–	–	–	✓	✓	–	–
[98]	–	–	✓	✓	–	–	–
[206]	–	✓	–	✓	–	–	–
[179]	–	✓	✓	–	–	–	–
[14], [103]	–	✓	✓	✓	–	–	–
[147] [188], [178], [120], [176], [94], [47], [90], [4], [214]	✓	–	–	✓	–	–	–
[138], [12]	✓	–	–	✓	✓	–	–
Ch. 4	✓	–	–	✓	✓	✓	–
[70] [29], [83], [89], [39], [75], [11], [169]	✓	✓	–	–	–	–	–
[192]	✓	✓	–	✓	–	–	–
Ch. 5, [141], [166] , [181], [171], [216], [126], [59], [149], [195], [116], [168], [74]	✓	✓	✓	–	–	–	–
[199]	✓	✓	✓	✓	–	–	–
[198]	✓	✓	✓	✓	✓	–	–
Our Work	✓	✓	✓	✓	✓	✓	✓

localization and assumes that the size and shape of the detected area in the semantic map match those of the ground truth. In the proposed scenario, the focus is not on the exact dimensions of detected features or their precise localization in space—since FRs require only approximate information to later visually inspect the scene—making IoU an inadequate metric.

Berclaz et al. [16] present a fixed-camera system that detects the positions of people in the environment on a grid map. Evaluation is based on the Multi-Object Detection Accuracy (MODA) metric, which accounts for the number of errors relative to the total number of objects to be detected in a single video frame. Similarly, Chavdarova et al. [35] describe a system that collects images from multiple static cameras and detects the locations of people in the environment, also relying on MODA to measure detection performance. Martin et al. [134] propose a benchmark for a system in which a mobile robot collects images using a depth camera while moving, and evaluate different tracking algorithms using MODA.

All of these works use MODA to evaluate object detection performance by assessing detections in each frame rather than against the final ground truth. More importantly, MODA focuses on detected objects and significantly underestimates the importance of correctly identifying empty areas—i.e., regions where no people in need of help or other relevant features are present—which could otherwise be safely ignored by FRs.

2.4 Comparative Analysis Methodology

To further investigate the literature relevant to our proposed robotic inspection approach, we conducted a systematic survey to identify and analyze pertinent studies.

We searched the Scopus database and the IEEE Xplore Library using the following query: (“search and rescue” OR “disaster response”) AND (“unmanned ground vehicle” OR UGV OR “ground robot” OR “mobile robot”) AND (“autonomous exploration” OR “frontier-based exploration” OR “object detection” OR “visual perception” OR “computer vision” OR “semantic mapping”). This search returned 131 results from Scopus and 66 results from the IEEE Xplore Library.

Papers published before 2015 were excluded to ensure that only recent works were considered. In addition, review articles and papers not primarily focused on autonomous exploration or the visual detection of relevant features were excluded. We then identified seven relevant characteristics, which are reported in Table 2.1. Papers presenting fewer than two of these characteristics were also excluded to avoid generic studies focusing on basic problems, such as visual feature detection or exploration. This selection process resulted in a final set of 42 works, summarized in Table 2.1.

The first characteristic, shown in the second column, is *SAR-specific*, which identifies research works tailored to robots operating in SAR environments and addressing the specific challenges of such scenarios. Works not marked in this column may mention possible SAR applications—typically in the introduction—but do not focus on this application in depth. The third column is *Visual feature detection*, which marks works in which the robot detects relevant features using a camera. Articles are marked in the fourth column, *Feature localization*, when the detected features are also localized in space with respect to the robot. The fifth column focuses on *Exploration* and is marked when the authors propose a solution for autonomously or semi-autonomously exploring and mapping an unknown environment. The sixth column, *Multi-floor capable*, indicates whether the robot presented in the paper can move between different floors of a building. By contrast, the seventh column, *Multi-floor aware*, is marked only when the robot is not only capable of moving across different floors but also explicitly accounts for their presence and for the risks associated with traversing staircases during the exploration phase. Finally, the last column, *Victim-and-cleared-space evaluation*, identifies papers in which performance assessment accounts for both the localization of affected individuals and the identification of areas verified to be empty, as noted in Remark 7.

SAR-specific

Although all these works mention SAR as a possible application scenario, only some of them address the challenges of SAR in depth whereas others only mention SAR as among their potential applications. For instance, [103] proposes an improved frontier-based exploration

approach and highlights its potential use in SAR; [179] presents a visual object-detection system and identifies SAR as a possible application; and [162] incorporates robust staircase detection and traversal based on 3D LiDAR, enabling exploration across multiple floors. However, none of these systems is specifically designed for or evaluated in SAR scenarios. As a counterexample of work specifically focused on SAR, [178] present a strategy explicitly tailored for natural disaster environments, where the main objective is the rapid localization of survivors while accounting for environmental conditions that could affect the robot's performance. In their approach, the selection of the next frontier is determined according to seven criteria: frontier length, the ratio of unknown cells surrounding the frontier, the distance between the robot and the frontier, the estimated travel time to reach the frontier, the distance from the frontier to the robot control station, the estimated danger level along the path, and the probability of confirming the presence of survivors..

Visual Feature Detection and Localization

Some studies focus on the detection of people in an area using drones [181, 171, 216]. Sruthi et al. [181] describe a YOLO-based approach for detecting people in the area covered by a teleoperated drone, with an emphasis on precise localization using GPS coordinates. Similarly, Sambolek et al. [171] propose a YOLO-based model to detect a single person in the environment, localize them, and predict their trajectory if they are moving. Another line of research focuses on improving detection of people who are partially occluded by obstacles in disaster scenarios [216] and the detection of people using thermal images [29].

Some approaches employ wheeled robots [199, 192, 149, 195, 168] or tracked platforms [198, 74], while others use quadruped robots [126] or heterogeneous teams composed of ground and aerial robots [59, 116, 89] to detect and localize relevant features with respect to a fixed reference frame in the environment.

Several works propose novel datasets or detection pipelines specifically designed to address the challenges of SAR environments [29, 83, 39, 75, 11, 169]. For instance, [75] presents a system for human detection in smoke-filled environments, while [39] trains a model capable of identifying partially occluded individuals.

Other research works exploit semantic maps for localization purposes [138, 103]. A semantic map encodes not only the spatial arrangement of elements within an environment, but also their semantic meaning and functional role. These representations combine geometric information about obstacles with semantic labels describing the nature of detected objects [112, 30, 212]. However, only a limited number of works address the integration of localized relevant features into semantic maps. For example, Tranzatto et al. [188] propose a solution

in which the positions of detected elements are directly annotated on the generated map. Among the others, [98] proposes a semantic mapping strategy that does not rely on visual data but instead uses a linear solenoid as an actuator to tap walls, objects, and other surfaces. Microphones record the resulting impact sounds to classify the materials based on their acoustic responses.

It is important to note that none of these works addresses the advantage of correctly evaluating empty space together with the detection of victims, as reported in the eighth column, which is a key characteristic in the context of SAR.

Exploration

Some works focusing on autonomous exploration are tailored to the SAR scenario [176, 4]. For example, [178] describes a strategy for exploring an unknown environment that considers several application-specific risk factors. In particular, when evaluating the next frontier, the strategy takes into account not only the typical factors considered in frontier-based exploration, but also the estimated likelihood of confirming a survivor's presence. Similarly, Tranzatto et al. [188] propose a frontier-based exploration approach for robots, both UGVs and UAVs, tasked with exploring harsh underground environments. In their work, frontiers are evaluated based on the volumetric expected information gain along the path to reach each frontier. The method also accounts for the finite duration of the exploration mission, estimating the remaining mission time when evaluating each frontier. Due to the high computational overhead, the system is controlled by a base computer with WiFi communication. Similarly, the exploration of an unknown environment with multiple robots collaborating [120, 214] is computed with a base computer and a stable WiFi connection. The work in [147] combines frontier-based exploration with deep reinforcement learning for autonomous exploration of unknown cluttered environments. Authors show that, compared with other frontier-based methods, it explores a larger area earlier, potentially enabling the detection of more victims during the initial, time-critical phase.

Other works address the problem of exploration and feature detection, with or without feature localization [199, 192, 14], but focus on a single floor, and the robot is not able to move between floors.

Multi-floor capable

In the aforementioned works, exploration was limited to a single floor: the robot operated in an enclosed environment entirely on the same level. However, when developing applications

for indoor buildings, it is important to account for the presence of staircases and multiple levels. For example, the work of Kudriashov et al. [110] focuses on constructing a map that accounts for the multiple floors of a building; however, it is not included in the table because the system is teleoperated.

A common focus in multi-floor autonomous building exploration is the strategy for traversing staircases and reaching the next floor. In [164], a cooperative exploration strategy involving two heterogeneous robots is proposed: since only one robot is able to traverse inclined ramps, it is assigned to investigate elevated areas whenever a change in floor level is encountered. The work of Roh et al. [162] addresses multi-level exploration through a system capable of detecting staircases and assessing whether they can be traversed, but the traversing is completed only when the current floor is fully discovered. The system described in [138] extracts properties related to terrain traversability and identifies staircases directly from point cloud data.

This survey shows that, to our knowledge, there are no efficient multi-floor strategies in the literature. The task of exploring a multi-floor building is treated as a sequence of single-floor explorations, without considering the building as a whole. Other works, in outdoor confined spaces, account for different levels, and staircases, by only evaluating their traversability [12].

The work most closely related to our developed system is that of Wang et al. [198], in which the authors propose a complete system that enables the robot to autonomously explore the environment while locating relevant features. In addition, the robot is able to move between floors, with a specific focus on Wi-Fi connectivity. However, the robot is teleoperated to a lower floor, after which exploration is completed only on that floor. The exploration process therefore does not consider when it would be convenient to move to a different floor in order to fully explore the building. Moreover, detections are evaluated only in terms of positive detections, without considering the quality of empty-space identification.

Multi-floor aware and Victims-and-cleared-space evaluation

The present work is the only one to address the two problems mentioned in columns seven and eight, i.e., taking into account the presence of multiple floors as an input to optimize autonomous exploration, and assessing performance by accounting for both the localization of affected individuals and the identification of areas verified to be empty.

Regarding multi-floor-aware exploration, Chapter 4 introduced the concept of *Stair Frontiers* as a new type of frontier that not only separates known and unknown space, but also provides access to a different floor. This concept accounts for the expected information

gain obtained by reaching the new floor, the risks associated with climbing the stairs, and the time required to complete the stair-climbing operation. Furthermore, given a limited exploration time, we evaluated the trade-off between moving toward unexplored floors and thoroughly exploring the floor on which the robot is currently operating. It should be noted that Chapter 4 completely ignored the problem of people detection, which is integrated in the present work for the first time through experiments in an actual building that evaluate how exploration strategies affect both coverage and the support provided to FRs in detecting victims and assessing cleared space.

2.5 Conclusion

This chapter reviewed the state of the art in robotic inspection, autonomous exploration, and environment perception, with a focus on harbor and search-and-rescue scenarios. The literature highlights the growing adoption of robotic systems in these domains, while also revealing several limitations that hinder their deployment in realistic operational settings.

The review showed that the requirements of first responders and other stakeholders are often insufficiently considered during system design, and that robot performance and trust are frequently evaluated through video-based studies rather than direct interaction. In autonomous exploration, frontier-based methods remain the predominant approach, but most solutions are designed for single-floor environments and lack explicit awareness of multi-level structures. Similarly, existing feature detection and localization systems do not fully exploit repeated observations to increase confidence in detections and often overlook the importance of correctly identifying empty space.

These findings reveal a gap between current approaches and the requirements of realistic inspection scenarios. This thesis addresses this gap by developing an inspection framework that is aware of the spatial structure of multi-level environments, supports reliable feature detection through repeated observations, and explicitly accounts for the identification of empty space.

Chapter 3

Qualitative Analysis of Stakeholder Requirements

Even if work to implement robotic inspections started in the early 1980s [117], inspection tasks are generally completed by specialized workers. As a consequence, there are two challenges in developing robotic systems for inspections: robots and their developers must be aware of the different tasks completed by workers, and the workers must accept robots. For these reasons, we first decided to gather insights from specialized workers, completing inspection missions as their duties. The aim was to answer this research question:

> *What are their needs and expectations regarding the use of a quadruped robot in inspection tasks?*

In this thesis, inspection tasks are divided into two categories: industrial inspection and emergency inspection. Emergency inspections are carried out after a disaster and typically require specialized robotic operations in environments with limited prior information. In contrast, industrial inspections take place in better-known environments and are performed more frequently.

Given the significant differences between these two application domains, insights from workers were collected and analyzed separately. Additionally, for industrial inspection, the focus was placed on the harbor environment, which is highly dynamic and presents several challenges.

To gather these insights, we conducted three separate studies. The first was carried out in person with FRs in Foligno in April 2023, the second consisted of an online questionnaire administered to FRs between March and July 2025, and the third was conducted in person at several terminals in the Port of Genoa, between March and December 2024.

This section describes the methodology and results of the semi-structured interviews conducted with FRs, along with a questionnaire administered to gather insights into the required SA. It also presents the semi-structured interviews carried out with harbor workers. The findings are then reported separately for each group.

3.1 Study 1: Search and Rescue

3.1.1 Methodology

As a first step, we aimed to assess rescuers' perceptions of the robot's capabilities and how these could be utilized in a SAR mission. To effectively evaluate the robot's motion capabilities, we chose to demonstrate it operating in an unstructured environment, highlighting both its limitations and potential.

For this reason, the interviews were conducted at the Civil Protection and ANPAS (National Association of Public Assistance) training camp in Foligno, shown in Figure 3.1. This facility, opened in July 2022, is primarily used for training canine units. At the Foligno training camp, we carried out two main activities:

- two testing sessions to evaluate performance across different terrains and motion strategies;
- a semi-structured interview to gather insights into potential applications of a quadruped robot in SAR missions.

To avoid any preconceptions about robots, we deemed it important to first demonstrate to FRs, many of whom had never seen a robot before, the actual motion capabilities of a mobile robot. For this reason, the first two testing sessions had the dual objective of both showing what the Spot robot from Boston Dynamics could do and gathering insights on the perceived performance across different terrains.

Motion evaluation

To measure the perception of the mobility of the robot from the perspective of FRs, a questionnaire was designed, as reported in Table 3.1. To prevent misunderstandings, the questionnaire was translated into Italian. A total of N=7 rescuers from ANPAS, Civil Protection, and Fire Fighters agreed to participate in this study. The testing was completed at the training camp of Foligno, which presents four zones that simulate different collapse



Figure 3.1 ANPAS and Civil Protection's Training Camp.

types, shown in Figure 3.1: (1) partially collapsed building, (2) reinforced concrete ceiling collapse, (3) pancake collapse, and (4) complete collapse. In these 4 zones, we identified seven areas that presented different challenges for the robot's mobility.

1. **House**: In the partially collapsed building, a room filled with debris to test the robot's ability to move on top of small, unstable obstacles.

2. **Grass**: According to the Spot user guide, the robot may encounter challenges when moving through tall grass outdoors, especially if obstacle avoidance is activated. To investigate this, we tested the robot both moving alongside and over grassy areas.

3. **Big Blocks**: To evaluate the robot's ability to traverse a set of varied obstacles arranged together, we teleoperated it over a group of large, stable blocks.

4. **Small rocks on flat ground**: To analyze the robot's stability in the presence of ground that may shift under its weight, we teleoperated the robot to traverse a patch of small rocks located on mostly flat ground.

5. **Small rocks to climb**: Similarly, to study the capability of the robot to traverse small mounds of unstable debris, we teleoperated the robot to ascend and subsequently descend a small pile of bricks.

6. **Tunnel**: To test the robot's obstacle avoidance functionalities, the robot was teleoperated through a tunnel.

7. **Dark room**: The robot bases its obstacle avoidance capabilities on the mounted cameras. To evaluate performance in conditions of reduced lighting, we teleoperated the robot inside a room with small rocks on the ground. Once the robot entered, we

closed the door to obstruct most of the external light. FRs were able to observe the robot move through a small opening in the walls.



(a) House



(b) Grass



(c) Big Blocks



(d) Small rocks on flat ground



(e) Small rocks to climb



(f) Tunnel



(g) Dark room

Figure 3.2 Data collection locations to evaluate the perceived motion abilities of the Spot robot from Boston Dynamics.

Additionally, two different locomotion modalities were identified: the walk gait, where at least two legs are on the ground during motion, and the crawl gait, where at least three legs are always in contact with the ground during motion.

After establishing the seven areas for the teleoperation and the two gait modalities, we formulated two hypotheses regarding the perception of the robot's motion capabilities:

- H1: There are differences in rescuers' perceptions of the robot's capabilities when it is teleoperated in different environments.
- H2: There are differences in rescuers' perceptions between the crawl and walk locomotion modes when the robot is teleoperated in different areas.

Based on the Spot User guide, we anticipated the grass area to receive the more negative evaluations, while the crawl locomotion would receive the more positive evaluations.

The first testing session aimed at evaluating H1; we teleoperated the robot in the seven areas, with the crawl gait, since we expected it to achieve optimal performance. We asked rescuers to assess the robot's mobility using the questionnaire for each area.

The second testing session, instead, aimed at testing H2; we selected four areas out of the seven previously studied, and we teleoperated the robot twice, using both gaits, without disclosing the teleoperation mode to participants to avoid biases. We then asked participants to assess the robot's performance using the questionnaire. The four selected areas are 'Grass', 'Small rocks on flat ground', 'Small rocks to climb' and 'Tunnel'.

Table 3.1 Questionnaire items related to the i-th area

Not stable				Very stable
1	2	3	4	5
Not quick				Very quick
1	2	3	4	5
It doesn't risk falling				It frequently risks falling
1	2	3	4	5
It doesn't stop for obstacles				It often stops for obstacles
1	2	3	4	5
It does not avoid or climb obstacles				It avoids well or climbs well obstacles
1	2	3	4	5

Semi-structured interview

Following the two testing sessions, we gathered all participants in the same room to collect their feedback on the experiments and, more importantly, to gain insights into best practices in rescue operations and their views on how the robot could be used to support such activities. The interviews were transcribed verbatim and subsequently analyzed to extract relevant insights.

We started by presenting the goals of our project: the robot is designed to autonomously explore a building in order to locate victims and assess the safety of the environment. We then asked about the current procedures used by rescuers to carry out these tasks and whether they considered that the robot could improve these processes. When the answer was positive, we further explored how they would envision adapting these procedures to integrate a robotic system.

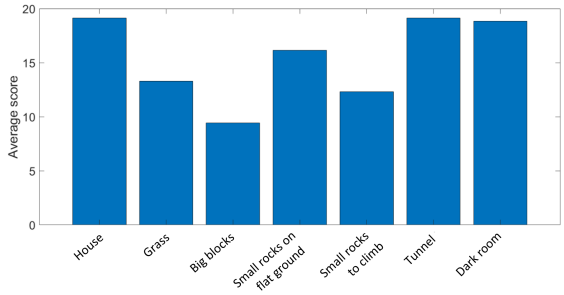
3.1.2 Results

First, we present the results of the motion evaluation; then, we describe the results of the unstructured interviews. It is important to note that, while teleoperating the robot, we observed several faults. In the case of a severe fault, such as a fall, the robot ceases all operations, turning off all motors until an operator manually restarts it. For this reason, since the presence of an operator cannot be guaranteed near the robot during autonomous operations, it is important to understand when faults may occur and to work toward safer motion strategies that prevent them.

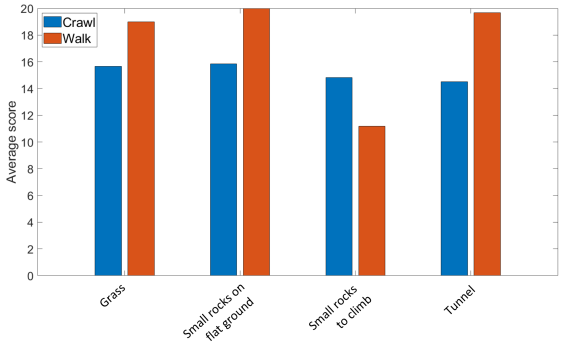
As expected, the most common faults happened while navigating through grass, as explained by the Spot User manual, grass can be mistaken for an obstacle. The robot, attempting to move away from collisions, moves quickly, losing stability, and leading to falls. Another significant fault happens when the robot attempts to climb over obstacles. When descending on the other side, the ground can become unstable, or there could be an unexpected elevation, different from the starting one, leading to the robot losing its balance and falling.

Motion evaluation

To evaluate the rescuers' perceptions during the first session, we combined the scores from the five items associated with each area, yielding a single score per participant for each area. Since not all seven participants completed the questionnaire for every area, the 'Small Rocks on Flat Ground,' 'Small Rocks to Climb,' and 'Dark Room' conditions each contain only six responses. Figure 3.3a illustrates the resulting values for the various areas, averaged across participants.



(a) Scores in different areas, averaged over participants - first testing session.



(b) Scores for different gaits and in different areas - second testing session.

Figure 3.3 Results for the motion evaluation

To evaluate Hypothesis 1, namely whether significant differences exist across areas, we employed a non-parametric approach given the small sample size. The Kruskal–Wallis H test, with $\alpha = 0.05$, revealed a statistically significant effect of the area on the dependent variable, $\chi^2(6) = 31.55$, $p < .001$. The mean rank values were 35.64 for 'House', 15.79 for 'Grass', 6.07 for 'Big blocks', 25.58 for 'Small rocks on flat ground', 13.67 for 'Small rocks to climb', 33.93 for 'Tunnel', and 34.25 for 'Dark room'.

Post-hoc comparisons were conducted using Dunn's test with a Bonferroni-adjusted significance level of 0.0024. The results showed significant differences between the following pairs: 'Big blocks' vs. 'House', 'Big blocks' vs. 'Tunnel', and 'Big blocks' vs. 'Dark room'.

During the second testing session, the rescuers evaluated the robot's mobility using either the walk or crawl gaits in four of the seven areas: 'Grass,' 'Small rocks on flat ground,' 'Small rocks to climb,' and 'Tunnel.' The questionnaires were completed by 6 rescuers because one of them had to leave earlier. We summed the points from the 5 items associated with each area to obtain one score per participant, per area, and per gait type.

To test hypothesis 2, a Mann-Whitney U test was performed with alpha=0.05. The results indicated that, for 'Grass', crawl had a significantly lower score than walk, $U = 3.5$, $p = .018$. For 'Small rocks on flat ground', crawl had a significantly lower score than walk, $U = 4$, $p = .029$. For 'Small rocks to climb', there was no significant difference between the score of crawl and walk, $U = 26.5$, $p = .194$. For 'Tunnel', crawl had a significantly lower score than walk, $U = 4$, $p = .026$.

Semi-structured interview

Table 3.2 reports the results of the interviews conducted with rescuers, summarizing the different topics discussed based on verbatim transcripts. The first column lists the tasks to be performed, while the second describes how these tasks are currently executed according to the procedures and protocols of the Italian Civil Protection. The third column presents the rescuers' suggestions on how an autonomous quadruped robot could support and potentially improve the implementation of these procedures.

3.1.3 Discussion

Perceived Performance

The results support Hypothesis 1. They indicate that the robot's perceived performance changes depending on the terrain, performing more effectively in some conditions and less effectively in others, with "Big blocks" emerging as the most difficult environment. During the design phase, it may therefore be useful to consider strategies that limit exposure to particularly challenging terrains in order to improve perceived reliability and trust. Issues encountered while the robot moves across grass or climbs small rocks appear to negatively influence rescuers' evaluations, as illustrated in Figure 3.3a. However, for "Grass" and "Small blocks to climb," it remains difficult to identify clear differences compared to other terrains, likely due to the relatively small number of participants who completed the questionnaire.

Table 3.2 Summary of unstructured interviews

Task to be accomplished	How the task is accomplished now	Suggestions on how to do it with Spot
Checking safety of the scene	Door evaluation, if collapsed or not	
	Look for parts of plaster or ceiling on the floor	Obstacle detection
	Gas sensors to evaluate the presence of explosives	Gas sensors mounted onboard
	Gas sensors to evaluate oxygen percentage	Gas sensors mounted onboard
	Gas sensor to verify and exclude the presence of harmful gasses such as for example CO_2	Gas sensors mounted on board
	Temperature of the room	Thermometer mounted onboard
	Visual evaluation of cracks and crevices	Computer vision and ML to decide what are important elements to save (i.e. cracked wall, crevices, etc.)
	Load tests for stairs	
Checking consciousness	Calling them to assess response to call	Speaker and microphone onboard, Natural Language Processing
	Touching them to assess touch response	Using the robot's arm, if present, or leg
	Pressing on the shoulder nerve to assess pain response	
Checking health Parameters	Body temperature	Thermal cameras
	ECG sensors to retrieve information on the circulation and the heart	Stable contact of the body for 10 seconds with a sensor
Detecting people	Explore the area and stop for each person before moving forward	Explore without stopping and sending information to rescuers that will stop for each person
	Look around and listen with care in the case that people might be scared and hidden	Infrared sensors mounted onboard to recognize human shape
Finding a path to reach people	Manually evaluate the safest path with the information from the evaluation of the safety of the scene	Save the robot trajectory so that it can be followed by rescuers when they reach the person
Retrieving information from the scene	Physically enter the building	Control room outside the building with a simplified interface to acquire and visualize data

The findings also support Hypothesis 2, though in the opposite direction to what was initially expected: the "crawl" locomotion mode is rated less favorably than the "walk" mode. While the Spot User Guide suggests that walking is suitable for general mobility and crawling provides greater stability on uneven terrain, the experimental results do not fully align with this guidance. The walking gait received higher average scores than crawling on "Grass," "Small rocks on flat ground," and "Tunnel," whereas crawling only showed better performance on "Small rocks to climb". This difference, however, is not statistically strong due to the limited sample size. Overall, rescuers rated walking as preferable in three out of the four environments. One possible explanation is that the crawl mode leads to overly cautious behavior, causing exaggerated corrective actions and occasional loss of balance. These observations highlight the importance of conducting practical evaluations when deploying such systems in real emergency contexts.

This study has several important limitations that should be considered.

First, the number of rescuers involved was relatively limited. It is worth noting that rescuers are a highly valuable and time-constrained group, making it difficult to involve them in activities such as observing and evaluating Spot's performance.

Second, a more thorough evaluation would benefit from allowing rescuers to observe the robot over a longer duration. As discussed previously, the robot experienced failures in certain environments, which may have negatively influenced participants' judgments. Providing more time for observation, with repeated task execution, could potentially lead to different perceptions.

Finally, combining subjective evaluations with objective performance metrics—such as task completion time—would provide a more complete understanding of how the robot's behavior affects rescuers' impressions. This dimension was not explored in the present study.

Unstructured Interviews

An analysis of Table 3.2 reveals that, for certain tasks, the FRs did not propose any robotic solution—for example to assess the response to pain. It is important to emphasize that this interview focused solely on the application of a quadruped robot, specifically the Spot robot from Boston Dynamics, as observed during the previous testing phase. Therefore, it is not possible to conclude that no robot is suitable for this task, nor that the Spot robot, with potential modifications, could not be adapted to perform it. The only conclusion that can be drawn is that the Spot robot, in its current form as observed during the demonstration, was considered inadequate for certain tasks. When first entering a building after a disaster, the initial step is to evaluate the safety of the scene. FRs proposed integrating gas sensors

and thermal cameras into the robot to assess air quality, as well as to determine whether specialized equipment is needed. Additionally, the development of specialized computer vision techniques has been proposed to evaluate for the presence of obstacles and to identify said obstacles to provide a better understanding of the structural stability of the building.

Once the structural stability of the building has been verified, the other main task of FRs is to detect people in the environment. FRs mentioned that for this operation, it is not sufficient to rely only on sight, and consequently on cameras in the case of robots, since people may be hidden from view. There is a need to integrate information from other sensors, such as infrared sensors, as well as microphones, to detect human sounds.

When people are found in the environment by the robot, it has to communicate to FRs both their position and a safe path to reach them. Additionally, it is considered useful for the robot to retrieve some health parameters. In particular, the robot should assess the level of consciousness by using a speaker and microphones to detect responses to calls, and by tapping the person to evaluate their response to touch. Other relevant health parameters to assess include the person's temperature, using thermal cameras, and circulation. This last task can be completed with an Electrocardiogram (ECG) sensor, which needs to be placed on the body for 10 seconds with stable contact, achievable with a robotic arm mounted on top of the robot.

With SAR operations, our initial assumption was that the robot would function autonomously to collect information before rescuers reached the site, after which it would be directly controlled by them during the intervention. However, this view was not supported by the rescuers we consulted. They argued that teleoperating a robot in emergency situations is neither practical nor efficient, and instead suggested that robotic systems should continue operating autonomously alongside their activities.

Although perspectives may differ among individuals, the concerns raised deserve attention. The rescuers pointed out that training all personnel to effectively teleoperate a robot would require substantial time and resources. Maintaining proficiency would also demand frequent use, as occasional training would not be sufficient for reliable operation in critical situations. In addition, when rescuers locate a victim—especially one who is injured—they must dedicate time to assessing the person's condition and the severity of injuries. During this time, it would be more effective for the robot to keep exploring the environment and collecting information autonomously, rather than remaining idle until an operator becomes available.

Based on these considerations, the proposed approach is for the robot to begin its mission autonomously before FRs arrive and to continue operating independently alongside them. In

doing so, it can continuously gather relevant data, including updates on evolving conditions, and provide this information to support rescuers in their tasks.

3.2 Study 2: Situational Awareness survey for FRs

3.2.1 Methodology

While collaborating with researchers from Queen Mary University of London, the University of Birmingham, and the University of Bristol in the field of SAR robotics, we decided to investigate how FRs acquire SA and whether they would be open to using robots that provide semantic information to support and improve the decision-making process. Situational Awareness has been defined by Endlsey [57] as the understanding of the environment and is divided into three levels - **i)** the perception of the elements in the environment within a volume of time and space, **ii)** the comprehension of their meaning, and **iii)** the projection of their status in the near future. To this end, we conducted a survey and collected insights and demands through online and paper questionnaires. To reach field experts in different countries, the questionnaires were translated from English into Italian, Greek, and Chinese by members of our multi-national research group, by using the back-translation technique. This helps to minimize any misunderstandings caused by second-language users, particularly given the abstract concepts in the questionnaire. The questionnaire consists of three sections: Background and basic demographic information, General attitudes towards robots, and Situational awareness and use of semantic information.

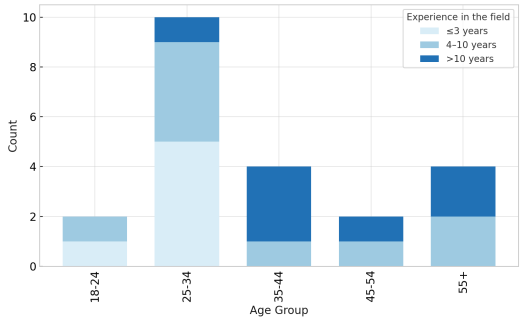
In the questionnaire, we investigate the FRs' backgrounds from two key dimensions: personal information and experience with robots. This section comprises 17 questions. The first four questions aim to investigate basic demographic information such as age, education level, and geographical location. The following five questions focus on experience in SAR environments, specifically the respondents' roles in missions and the experience they have gained during FR operations. Finally, the last eight questions investigate the use of robots in SAR missions, both in terms of the respondents' previous experience and their expectations for future applications.

Next, we propose a validated questionnaire, General Attitudes Towards Robots (GAToRS) [107], as they may affect how FRs perceive domain-specific robotics. The Italian version of the GAToRS has been translated following the validated translation provided by Carradore et al. [33]. The questionnaire is composed of 25 items investigating the general attitude towards robots, and subsequently the general acceptance of robotic applications.

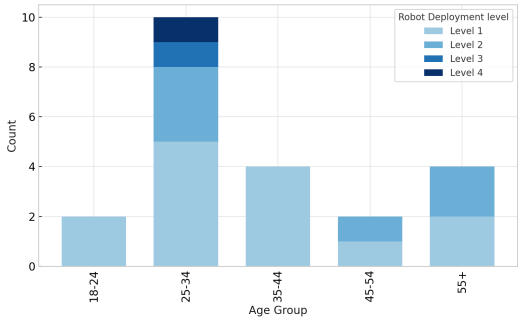
In the questionnaire section **Situational Awareness and Use of Semantic Information**, we first define SA and semantic information to avoid any misunderstanding. Subsequently, through 15 questions, both multiple-choice and open-ended, we survey FRs regarding their usage, experience, expectations, and overall opinions on SA and semantics. In particular, we aim to understand which information is available at the beginning of a mission and how FRs enhance their understanding using semantic information. Finally, we investigate how AI can support missions and what level of accuracy is required for FRs to trust the intervention of an AI agent.

By leveraging our professional networks and contacts, we successfully invited 22 FRs from eight countries to participate in the survey. All FRs are safety and emergency professionals with relevant experience, covering multiple countries and emergency roles, e.g. firefighters, emergency medical personnel, technical operators, on-site commanders, operational scientists, vehicle drivers, and others.

3.2.2 Results



(a) Field experience distribution by age.



(b) Robot deployment level distribution by age.

Figure 3.4 Distributions of field experience and robot deployment level by age group. Robot deployment level refers to the frequency of using robots in the mission. Higher level means higher frequency.

The recruited FRs come from China, Japan, Germany, Iraq, Italy, Greece, the UK, and the USA. While there is an unavoidable imbalance in regional representation, it is difficult to recruit FRs globally or achieve equal numbers. However, all participants share similar standards of training and operational procedures in the field, so this imbalance is expected to have limited impact on the findings.

Demographically, females are underrepresented, with only three female FRs participating. This likely reflects the physically demanding nature of many FR roles, which are still

predominantly male in most countries. In terms of age, FRs under 35 make up about half of the sample; they typically serve on the front line, while older FRs tend to hold senior roles such as commanders or scientific advisors providing guidance.

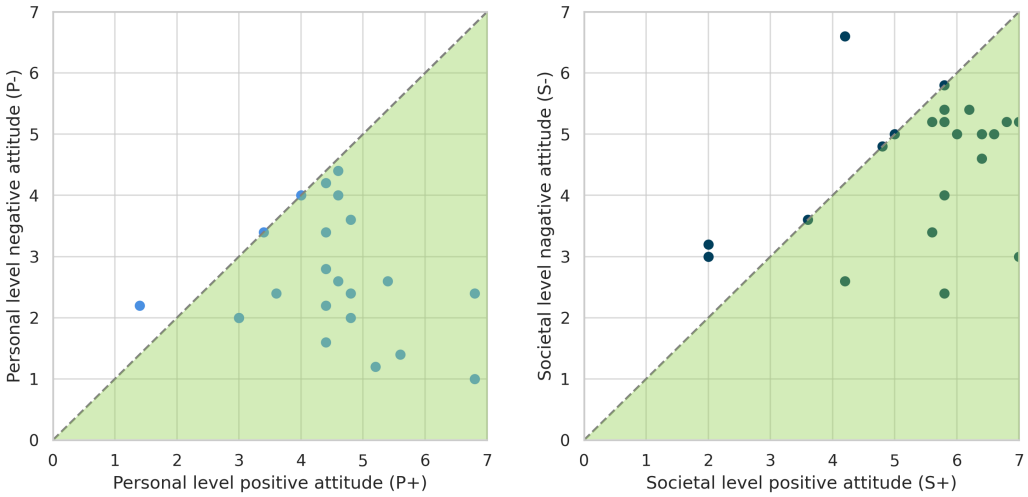
Educationally, FRs generally do not require very high formal qualifications, with over 70% holding below a master's degree or equivalent. Nevertheless, all roles require specialized training, skills, experience, and domain knowledge to handle emergencies effectively.

Regarding experience levels, the sample is relatively balanced: junior (<3 years), intermediate (4–10 years), and senior (>10 years) FRs account for 27%, 42%, and 31%, respectively. The study also collected information on their robotics experience and current perspectives on robot deployment in the field.

From the distributions in 3.4, the opinions of both older and younger FRs on current robot deployments in the field converge. Most FRs feel that robot deployments are still not very advanced, and few of the deployed robots can operate fully autonomously.

The scale of the **General attitude towards robot** section differentiates **i)** comfort and enjoyment around robots, **ii)** unease and anxiety around robots, **iii)** rational hopes about robots in general (at the societal level), and **iv)** rational worries about robots in general (at the societal level). Hence, the questionnaire is divided into four dimensions: personal-level positive attitudes (P+), which capture positive emotions such as feeling relaxed, trust, and friendliness toward robots; personal-level negative attitudes (P-), which reflect negative emotions such as anxiety, fear, or discomfort; societal-level positive attitudes (S+), which relate to expectations that robots can benefit society by assisting humans and operating in hazardous environments; and societal-level negative attitudes (S-), which concern broader social risks such as unemployment, regulatory challenges, and changes to social structures.

The distributions of FR's attitudes are presented in 3.5. The y-axis refers to the negative view, and the x-axis refers to the positive view. In 3.5a, most points are close to or below the diagonal line (in the green zone), where positive views are more than negative views. In other words, most of the FRs demonstrated more positive personal views than negative personal views. Only one FR who flies the drone in the field shows more negative personal attitudes than positive attitudes. Apart from them, most FRs indicate a certain degree of trust, relaxation, and welcome towards robots. At the same time, they have some concerns and reservations. However, these are fewer than their positive views. In 3.5b, the participants show greater concern about robots at a societal level. Comparing with the 3.5a, we notice that most points cluster in the upper right of the green area. This suggests that most participants are confident about the positive influence of robots on society. However, they are more concerned about the negative impacts of robots on society than about the negative impact on



(a) Personal Level Attitude (b) Societal Level Attitude

Figure 3.5 Distribution of general attitudes towards robots on personal and societal levels. Most points are under the diagonal line (in the green zone). This indicates that most FRs have positive attitudes towards robots in both personal and societal contexts.

individuals. Moreover, there are three FRs above the diagonal line, i.e. three participants (13%) indicated stronger concerns about the impact of robots on society in general. In summary, most FRs are willing to trust and use robots without having very strong concerns about societal impact.

In the section **Views about SA and use of semantic information**, the numerical questions with a marking scale and open-ended questions are provided to obtain more specific ideas on different occasions, e.g. different task backgrounds. Specifically, the numerical questions are listed below:

- How much do you have an estimation of the Situational Awareness before the mission? (Q43)
- How often do you experience loss of concentration, fatigue, and cognitive overload? (Q47)
- To what extent do you think having semantic information is useful in building Situational Awareness? (Q49)
- To what extent is semantic understanding and reasoning used during missions? (Q50)
- Can semantic information and semantic scene understanding help predict or mitigate an emergency that was not predicted or foreseen before the mission? (Q53)

- Imagine you have some AI agent assisting in your mission (e.g. in a robot, in a computer, others). How much do you agree with the statement "It is helpful if the AI agent can process an understanding of the situation"? (Q54)

These questions are answered on a Likert scale [96] from one to five, where one refers to a low level (e.g. totally disagree, low frequency etc.), and five refers to a high level (totally agree, high frequency etc.). The results of the numerical questions are shown in 3.6.

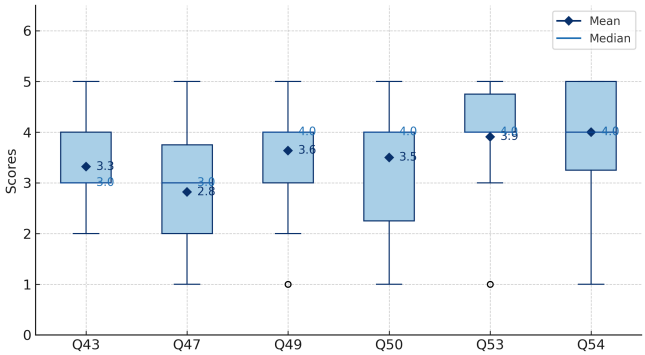


Figure 3.6 Score distributions of numerical questions.

The open-ended questions covered a wide range of topics. They have two types of formats: fill-in questions and multiple choice (additional choices not listed are supported by adding after the last choice option). Below, we summarise the answers provided by participants. Where digits are given in brackets, these refer to the number of FRs who aligned with certain choices and options in their answers. Moreover, the answers to fill-in questions are presented with no preference, i.e. we do not rank the importance of the information from participants. The information known before a mission (Q42) includes the type of event (e.g., location, hazardous materials), its extent, the people or animals involved and their distribution, required interventions and task goals, weather forecasts, information from emergency call centers based on onsite reports, and the availability of nearby resources such as medical centers and airlifting services. Methods used to obtain SA (Q44) include cameras/images (16), sensor readings (14), communication with people on site (such as civilians and team members) (16), in-person exploration (16), and, less frequently, comparison with previous data sources such as Google Maps (1). Challenges in obtaining SA (Q45) involve unstable or slow communication with sensors, delays in team communication, malfunctioning perception devices, rapidly changing environments, difficulties in data integration and incomplete data, complex terrain and debris, low visibility or darkness, conflicts in air traffic control for drone deployment (e.g., interference with helicopters), lack of sound or 360-degree vision, and false

or inaccurate information. The top three important semantics (Q46) include the type of event and affected entities, spatial-temporal positioning of entities and resources (e.g., estimating paths and schedules), and interactions between entities such as obstacles blocking ambulances or risks from hazardous materials, as well as task-relevant location information and resource availability (e.g., medical centers, airlifting services). Reasons for fatigue and cognitive overload (Q48) include long 24-hour shift rotations, chaotic and high-pressure environments, fragmented and uncertain information, communication and coordination difficulties, high responsibility and fear of error, harsh physical conditions (e.g., heat, protective gear, injuries), technological and infrastructure limitations, and multitasking across conflicting demands. Finally, current methods for delivering semantics (Q51) primarily use vision (20) and sound (15), with limited use of haptics (4) and vibration (4), while ideal methods (Q52) increase reliance on vision (21), sound (19), and also greater use of haptics (7) and vibration (11).

Finally, two questions about semantics accuracy were surveyed: the accuracy threshold for having trust in semantics, and the accuracy threshold for usefulness of semantics (See 3.7). Only 20 participants gave a specific accuracy. The other two participants provided individual insights about this comparison. One participant thinks any accuracy is fine, as long as FRs are informed about the reliability of the accuracy. Another participant feels that the accuracy requirements vary depending on different occasions. But this participant feels that such sensors should show better performance than human perception and inference. The mean accuracy threshold for achieving trust in semantics is 74.6% (i.e. participants will trust the semantics if it has over 75% accuracy), and the mean accuracy threshold for usefulness is 67.8% (i.e. participants will use the semantics if it has over 68% accuracy). In 3.7, it can be seen that the trust threshold is generally higher than the usefulness threshold for most participants. It indicates that having semantics can be useful even if participants do not feel they can completely trust them. In other words, FRs are eager and expecting to see increasing amounts of semantic information being used in a useful way in the field, but it will take longer before these are fully trusted

3.2.3 Discussion

Due to the number of FRs in this study, we briefly analyze their feedback by discussing their mean, median, and distribution. We analyze their feedback by involving the related topic together with numerical and open-ended questions.

FRs use a variety of methods to obtain SA (Q44). Communication with other people from the site, and in-person exploration, are the most popular ways to obtain information,

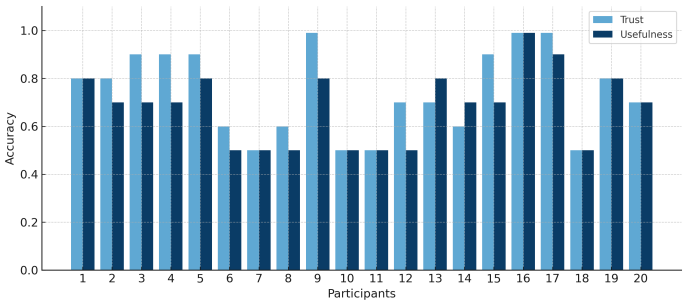


Figure 3.7 Accuracy threshold values for achieving trust and usefulness: light blue bars refer to the accuracy threshold for achieving trust, and dark blue bars refer to the accuracy threshold for achieving usefulness.

suggesting that human perception still dominates in obtaining SA. FRs have not yet deeply depended on robot sensors such as cameras. However, such sensors are still welcomed in many deployments and gained 15 votes out of 21. Based on the feedback of (Q46) that aimed to understand the most important semantics, we find many of them are low-level semantics that can be obtained by robots e.g., location, distance. Some of them are high-level semantics e.g., risk. And many of them are context, e.g. medical center, emergency infrastructure.

Combining the Q49, Q50, Q51, Q52 and Q53, about FRs' views on semantic information. Most FRs feel semantics has a positive effect on obtaining SA (mean 3.6 on a score from one to five, median 4, see Figure 3.6). Most FRs have the consensus that semantics help to predict the situation (mean 3.9 on a score from one to five, median 4, see 3.6). In terms of the delivery methods of semantics, current methods are mostly a combination of vision and sound (from fixed or robot cameras at the base camp). Haptics and vibrations are used in a few special tasks from the joystick, e.g. the deployment of drones. Hence, these answers denote that multimodal is the best option to build SA. FRs who operate the drones are expecting to see an increase in haptics and vibration in other robotic tasks, along with the increasing application of semantics in the future.

Q54 indicates FRs have the consensus that AI understanding is valuable in the mission (mean 4 on a score from 1-5, median 4). They probably expect the AI-generated SA to help them reduce the cognitive workload and avoid potential risks. Based on the feedback from Q55 and Q56, we compare the background of FRs who have a higher accuracy threshold for usefulness than for trust. Moreover, some FRs believe the threshold of trust and usefulness in this context is the same concept. I.e., if they are willing to use a certain type of semantic information, it means that they already trust it. We assume the difference might come from variation in the different types of tasks that each participant performs in their work. Some

tasks and decisions might have much stricter requirements on data accuracy than others. Hence, these FRs might emphasize the accuracy of the semantics more highly.

We gained some additional insights from FR comments in the open-ended questions. For instance, one FR highlights the importance of distance to the object (e.g. distance to the gas tank), which could help reduce potential risks. One senior FR with experience in robot deployments, especially drone deployment, feels that the acceptance accuracy for trustworthy and useful semantics varies according to different missions and scenarios. By analysing his background, we found that he was the leader of the squad and participated in many different types of response missions.

In summary, most FRs across a variety of backgrounds are willing to apply semantics to help them build SA. On one hand, senior FRs have more experience in using semantics to obtain SA, and show higher expectations for using semantics-based SA in the future. On the other hand, they do not feel so dependent on assisted SA compared with other groups. They have sufficient expertise and experience to leverage information from multiple sources and mitigate or overcome the impact of system errors or incomplete information.

3.3 Study 3: Harbor inspection

3.3.1 Methodology

We had the opportunity to collaborate with the Port Authority of Genoa within the context of the PNRR RAISE Project. Through this collaboration, we aimed to introduce robots into the harbor environment and integrate them into routine operational activities. Genova's harbor, the largest port in Italy, has a terrain extension of 7,000,000 m^2. Given the large scale and outdoor environment, a drone might initially seem like a suitable solution. Drones are capable of moving rapidly between locations and can offer perspectives that are not easily accessible to workers on the ground. However, they are constrained by limited battery life and often require special permits, which may not always be obtainable in certain ports—for instance, in the harbor of Port of Genoa due to its proximity to the airport and the associated flight restrictions. Furthermore, drones can pose safety risks when operating near personnel or within confined spaces where the likelihood of collisions is higher. In contrast, quadruped robots provide longer operational autonomy and are designed to function safely in close proximity to workers without the need for special permits.

The harbor is composed of multiple terminals, each dedicated to handling different types of goods. Among these, we identified three terminals where a quadruped robot could potentially enhance operational efficiency.

- Terminal IMT: specializes in handling containerized goods, providing also storage for hazardous materials.
- Terminal C. Steinweg - GMT SRL: focuses on aluminum products, for example, slabs and coils.
- Terminal Rinfuse Genova; stores bulks goods in open-air containers.

To collect the requirements in each of the terminals, we completed a semi-structured interview. A total of N=10 people agreed to participate in this study, 5 people from Terminal IMT, 3 people from Terminal C. Steinweg,and 2 people from Terminal Rinfuse Genova. At each terminal, we selected interview participants covering a variety of roles and professional backgrounds. Specifically, we involved a coordinator in charge of workforce management, the safety manager, and several operational workers. In addition, the Head of the Health and Safety Inspection Department of the Genoa Port Authority, who is responsible for overseeing safety within the port area, was present at every interview.

Each interview was comprised of three phases:

- *Preliminary*: aimed at understanding the workflow and pinpointing potential critical issues within each terminal.
- *Demonstration*: during which we showcased the capabilities of the Boston Dynamics Spot robot.
- *Design*: in which we collaborated with participants to explore potential applications, based on their impressions and feedback on the robot.

During this process, we decided to discuss our previous research only after the interview was completed to avoid bias. The aim was to collect the creative input of workers, to ensure that their ideas would not be influenced by discussing previous implementations. All interviews were audio-recorded and transcribed verbatim for analysis.

3.3.2 Results

From the first phase of the interviews, we aimed at identifying the recurring themes. In Table 3.3, the findings are reported. In the first column of the table, we include an identifier

to facilitate further discussion. In the second column, we have a brief description of each discussed problem. In the third column, we classify the critical areas based on the specific needs and the locations where the robot might operate. The fourth column reports the priority assigned to the different problems. Based on the amount of time devoted to each issue during the interviews and the language used by participants to express the urgency of addressing it, each problem was classified as high, medium, or low priority. Column five indicates the expected frequency of inspections. This classification distinguishes between inspections that are carried out regularly, such as on a daily basis, and those that are performed at specific moments following an event, for example, when a ship arrives at the harbor.

Looking at the third column, it is possible to notice that three different areas have been identified: storage yards (60%), ships' hold (27%), and cranes (13%). The storage yard is located in several terminals and is used to store both containers, as illustrated in Fig. 3.8a, and bulk goods, as shown in Fig. 3.8b. It serves as a temporary storage area for all materials and goods received at the port that must be held before being transported to other destinations, either by ship or by land.

In some cases, inspections must be carried out onboard vessels, typically in the ship's hold, shown in Fig. 3.8c. This is required because, upon arrival at the harbor for unloading, workers may lack information about the internal safety conditions (e.g., due to adverse weather in the preceding days). A similar situation occurs when personnel need to work in confined spaces during ship repair operations, where the safety conditions are also unknown in advance.

Finally, during the interviews, while gathering information on the safety challenges encountered by workers, the topic of aerial operations was raised. This category encompasses all tasks that require an operator to work at height using cranes, an example of cranes requiring maintenance is in Fig. 3.8d.

We then moved on to the next phase, in which we examined each problem and reviewed the interviews to identify the possible solutions mentioned for each case. For every proposed solution, we determined the most suitable robot, as reported in Table 3.4.

In the first column, we provide the identifier associated with the corresponding problem. It is important to note that all solutions were proposed in collaboration with workers. Although each problem is paired with a solution, not all of them fall within the scope of this study.

The second column describes the proposed solution, while the third column reports the robot preferred by the workers. While the study primarily focuses on applications involving a quadruped robot, workers also suggested some use cases for drones.



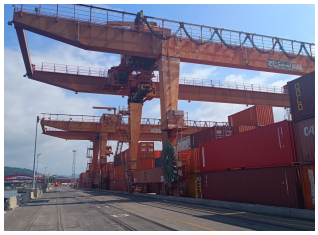
(a) Spot robot inspecting a container at the Harbor of Genova, Terminal Messina.



(b) Terminal Rinfuse Genova storing bulked goods (coal).



(c) Spot robot walking beside a ship's hold at the Terminal Messina.



(d) Cranes to move containers at the Terminal Messina.

Figure 3.8 Examples of locations at the Harbor of Genova where the robot can operate.

Finally, the fourth column indicates the level of autonomy expected by the workers. Based on their descriptions of the tasks, we classified the applications into three categories: teleoperation, shared autonomy, and full autonomy.

3.3.3 Discussion

A different perspective emerged from the interview with harbor workers, where it was possible to find a solution for every emerging problem. We begin by examining which tasks in the storage yard can be effectively carried out by a quadruped robot.

In hazardous goods areas, activities such as detecting chemical leaks (2), monitoring radiation levels (3), and identifying fire hazards (4) are well-suited to this type of platform. Thanks to its ability to traverse uneven terrain, step over obstacles, and operate in proximity to people, a quadruped robot can access locations that are difficult to reach with other systems. Its capacity to move close to ground-level elements and operate within confined spaces enables detailed inspection near potential sources of risk, improving the accuracy of detection. To perform these operations, the robot must be equipped with appropriate sensing technologies, including gas detectors, radiation sensors, and thermal imaging devices.

Regarding the monitoring of self-heating phenomena in bulk storage areas (6), the interviewed operators indicated a preference for quadruped robots over drones. This is

Table 3.3 Emerged Problems

	Problems	Location	Priority	Check repetition
1	Custom inspection	Storage Yard	Low	When needed
2	Chemical leaks in the hazardous good park	Storage Yard	High	Repeated
3	Radioactive materials in hazardous good park	Storage Yard	High	Repeated
4	Fires at the hazardous goods park	Storage Yard	High	Repeated
5	Incorrect piling of containers	Storage Yard	Medium	Repeated
6	Self-combustion of bulk storage (e.g. coal)	Storage Yard	High	Repeated
7	Unauthorized people in restricted access areas	Storage Yard	Low	Repeated
8	Wrong application of train seals	Storage Yard	Low	Repeated
9	Poor air quality in ships' holds	Ships' hold	High	When needed
10	Unstable cargo due to weather adversities or incorrect loading	Ships' hold	High	When needed
11	Poor air quality in naval repairs	Ships' hold	High	When needed
12	Radioactive materials in naval repairs	Ships' hold	High	When needed
13	X-ray inspection of cranes	Cranes	Low	Repeated
14	Inspection of cranes with penetrating liquids	Cranes	Low	Repeated
15	The Twist-mechanisms of containers get blocked	Storage Yard	Medium	When needed

mainly due to the robot's ability to physically move across the surface of the material, allowing more accurate temperature measurements when equipped with thermal cameras. Furthermore, when fitted with high-resolution cameras and an inertial measurement unit (IMU), a quadruped robot can better evaluate container stacking conditions (5), both in terms of alignment and stability relative to gravity. In contrast, aerial systems may be more affected by environmental factors such as wind.

For surveillance tasks like controlling access to restricted zones (7) and verifying the correct placement of train seals (8), both drones and quadruped robots can be effective if equipped with suitable cameras. Drones offer faster coverage and broader data collection, whereas quadruped robots provide higher precision in identifying individuals and analyzing specific details such as seals.

For operations inside ship holds, the robot must carry additional sensing equipment, such as radiation detectors, gas sensors, and 3D LiDAR systems.

In the case of bulk cargo inspections (9) or Ro-Ro (Roll-On Roll-Off) inspections upon vessel arrival (10), LiDAR technology enables the creation of detailed three-dimensional models of the environment. These models support the evaluation of cargo stability, for example, by identifying damaged securing elements or displaced loads. A quadruped robot is particularly advantageous in this context, as it can navigate beneath structures and traverse irregular surfaces within the hold.

Table 3.4 Possible Solutions

	Design solutions	Preferred robot	Autonomy Level
1	Moving inside the container to verify the contents	Drone	Teleoperated
2	Inspection to verify the presence of chemicals near the containers	Quadruped robot	Autonomous
3	Inspection to verify the presence of radioactions in the hazardous goods park	Quadruped robot	Autonomous
4	Inspection to verify the presence of fires in the hazardous goods park	Quadruped robot	Autonomous
5	Visual inspection to verify correct piling of containers	Quadruped robot	Autonomous
6	Inspection to monitor the temperature of the bulk goods	Quadruped robot	Autonomous
7	Inspection to verify the presence of people in restricted areas	Quadruped robot/-drone	Autonomous
8	Visual inspection of train seals	Quadruped robot/-drone	Autonomous
9	Inspection to verify the percentage of CO_2 and O_2 in the ships' holds	Quadruped robot	Mixed Autonomy
10	3D map of the cargo to verify its positioning	Quadruped robot	Mixed Autonomy
11	Inspection to verify the percentage of CO_2 and O_2 during naval repairs	Quadruped robot	Mixed Autonomy
12	Inspection to verify the presence of radioactions during naval repair	Quadruped robot	Mixed Autonomy
13	X-Ray sensors on drones	Drone	Autonomous
14	Drones equipped with penetrating liquids and high resolution cameras	Drone	Autonomous
15	Visual inspection of the Twist mechanism	Drone	Teleoperated

For naval maintenance scenarios (11 and 12), the sensing requirements are more limited, typically involving gas and radiation monitoring. Here, the robot is primarily used to access confined or hazardous areas before personnel begin their work.

In both ship unloading and maintenance applications, the robot's size and maneuverability are critical. It must be compact and agile enough to operate effectively within restricted and cluttered spaces.

Additional noteworthy applications were identified, although they do not involve quadruped robots. For customs inspections (1), the objective is to use a highly agile robotic system capable of navigating inside partially empty containers to inspect their contents. In this context, a drone represents a suitable solution, as it can move within the container without applying pressure to potentially delicate items.

Furthermore, the interviewed workers highlighted the need to enhance crane inspection processes through robotic technologies. At present, these inspections are performed by personnel suspended at height, who apply penetrant liquids to detect cracks or conduct X-ray

analyses on specific areas. As a possible improvement, the use of drones was suggested (13 and 14). The issue related to failures of the twist-lock mechanism is not addressed here, as its complexity would require additional information to develop a viable solution (15).

By looking at the fourth column of Table 3.4, it is possible to see that workers were not against the use of the robot, in a mixed autonomy paradigm. In particular, tasks that require repeated execution in time, (2) through (8), are a better fit to implement complete autonomous behavior. Additionally, due to the highly dynamic nature of harbor environments, relying on fixed navigation routes is impractical. Instead, the system should be equipped with core autonomous robotics capabilities, such as environment exploration, mapping, path planning, and the detection of relevant features. On the contrary, tasks (1) and (9) through (12), must be performed as needed, either immediately after the ship arrives for unloading or prior to maintenance activities; the use of teleoperation could be considered. In the case of Ro-Ro vessels, for instance, the robot might be remotely guided to the far end of the hold to gather data. However, the presence of obstacles and congestion in such environments can interfere with operation. For this reason, adopting a hybrid autonomy strategy would be more effective in maintaining continuous functionality. The robot could operate under direct control when communication is reliable and transition to autonomous modes whenever the connection degrades.

3.4 Conclusion

We propose three different studies to investigate both the perception of stakeholders towards robotics, as well as what is needed in each scenario. In particular, we propose two semi-structured interviews to design possible robotic applications with both FRs and harbor workers, and an online questionnaire to gather insights on what could be useful to increase SA during a rescue mission.

During the interviews with both rescuers and harbor workers, the expectation about the autonomy level of the robot emerged. FRs mainly imagined a completely autonomous robot, while, on the other hand, harbor workers were more interested in a mixed autonomy paradigm, where the robot is able to complete repetitive tasks in autonomy but can be controlled, if needed, for more specific tasks that are to be completed sporadically.

During the interviews, most FRs and workers showed strong interest in the use of robotic technologies to enhance workplace safety, although some reservations were also expressed. In particular, a few workers raised concerns about possible job displacement. Moreover, there were doubts about whether robots could match human precision in certain tasks, potentially

affecting safety outcomes. Involving the end users throughout the development process is essential to fully understand and address these concerns during the design stage. This approach can help ensure that robots are seen not as substitutes for human labor, but as tools that support workers and contribute to safer working conditions.

Analysing the results from the online questionnaire, we gathered insights on the type of information that FRs need to increase SA. Additionally, we discovered that in general, FRs are willing to apply semantics to help them build SA; additionally, they think that having additional semantics is useful even when they cannot fully trust the system.

Chapter 4

Autonomous Exploration of Multi-Floor Buildings

Following the interviews both with FRs and with harbor workers, detailed in Chapter 3, the need for autonomy emerged for both robotic applications. At different levels, both disaster management and industrial environments pose challenges for operational planning; disaster sites are often unpredictable due to structural damage and furniture collapse. On the other hand, the organization of industrial sites is more predictable, but at the same time, large obstacles are moved around daily, for example, containers in the port environment.

Given the application's specificity and stricter requirements, the following section focuses primarily on the SAR application. Please note that everything can be applied to the harbor environment by relaxing some constraints and fine-tuning the approach.

Suppose the goal is to develop a robotic solution to support the immediate post-disaster phase, such as after an earthquake. The robot must perform an initial inspection of a partially collapsed building, collecting as much information as possible before FRs arrive, to provide them with an overview of the building's interior, as mentioned in Section 3. The information to be acquired is of a different nature, as the robot should gather information about the state of the building, including both its structural condition and the presence of people. To further increase complexity, most buildings have multiple floors, highlighting the need for robots capable of climbing stairs and for dedicated strategies to manage multi-floor exploration. Given the aforementioned scenario, this chapter proposes a system for autonomously exploring a previously unknown environment in a time-efficient way. The proposed framework was implemented on a quadruped robot.

4.1 Methodology

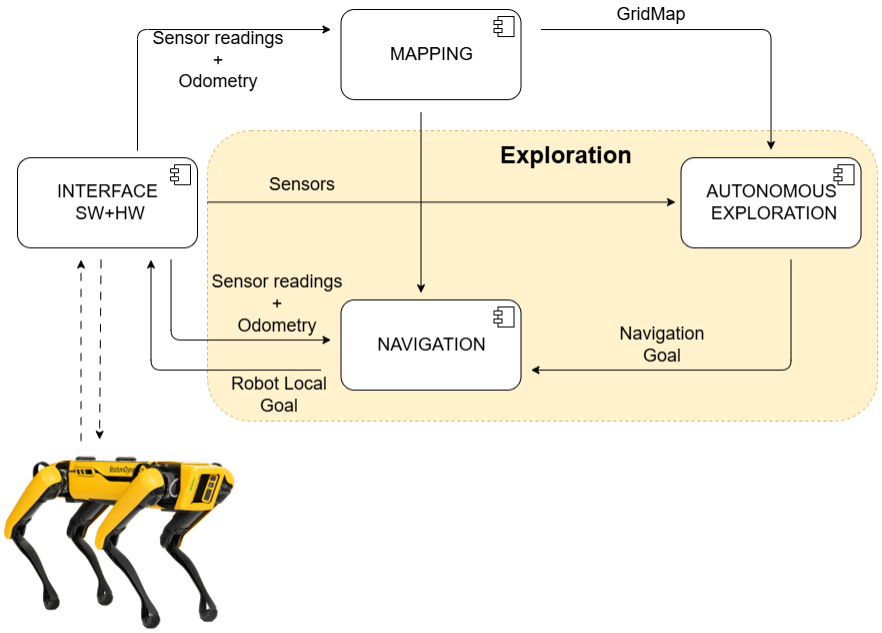


Figure 4.1 System Architecture.

The system architecture for autonomous exploration of multi-floor buildings is shown in Figure 4.1. The Exploration Module, highlighted in yellow, represents the core contribution of this chapter. To explore and map unknown areas of the environment, the Exploration Module implements a frontier-based strategy [208]. The algorithm, in line with several similar approaches, uses the Rapidly-exploring Random Tree (RRT) algorithm [191] for frontier detection and employs a metric based on the expected information gain to select the next frontier to explore and the estimated risk in reaching it. However, in a multi-floor SAR domain, several aspects differ from standard exploration and must be addressed.

Firstly, since the application involves multi-floor buildings, it is necessary to identify frontiers that lead to different floors, hereafter referred to as Stair Frontiers. Unlike standard frontiers, which separate explored and unexplored areas, Stair Frontiers rely on the system's ability to represent stairs as links between different floors and to take their semantic meaning into account in the exploration strategy. A Stair Frontier leads to a floor that has not yet been explored and is therefore potentially highly informative in terms of expected information gain if the robot moves toward it.

Secondly, exploration can be conducted in different styles, with the choice of style depending on the application domain. In SAR applications, for instance, where the robot

explores the environment before FRs enter the area, the ability to acquire information within a limited time frame (e.g., 10 minutes) can be more important than exhaustively exploring every small recess.

These principles are reflected in the exploration strategy we implemented, which is designed to achieve the following objectives — of which the last three particularly reflect the peculiarities of SAR within a limited time frame.

O1. Prioritize frontiers with high expected information gain.

O2. Prefer frontiers located at the boundary between explored and unexplored space that are sufficiently close to the robot, minimizing travel time and reducing the risk of selecting unreachable targets.

O3. Avoid selecting spurious frontiers in the immediate vicinity of the robot, which can typically be explored through in-place sensing or naturally during navigation.

O4. Consider the amount of unexplored space remaining on each floor, avoiding unnecessary time spent on floors that are nearly fully explored while preventing overly frequent floor changes.

O5. Favor straight paths over tortuous ones, promoting the complete exploration of hallways before entering rooms or deviating to explore lateral branches.

O6. Introduce persistence in frontier selection across consecutive iterations by preferring frontiers aligned with the robot's current direction of motion, reducing frequent target switching before frontiers are reached.

To determine which frontier to visit, each frontier is evaluated against all the criteria described above, as expressed mathematically in Equation 4.1 (for standard frontiers) and Equation 4.4 (for Stair Frontiers). In both equations, x_f represents the 2D position of the frontier's center, and x_r represents the position and orientation of the robot at the time of the frontier evaluation. As defined in previous works, such as [60], the revenue associated with reaching a frontier is composed of gains, which are added to the revenue, and costs, which are subtracted.

$$\begin{aligned} R = I_w \cdot h(x_f, x_r) \cdot I(x_f) - D_w \cdot D(x_r, x_f) - V_w \cdot V(x_r, x_f) \\ + F_w \cdot F(x_r) - C_w \cdot C(x_r, x_f) + O_w \cdot O(x_r, x_f). \end{aligned} \tag{4.1}$$

The first term is the information gain $I(x_f)$, which quantifies the amount of unknown space near the frontier (corresponding to objective O1). It is computed as the number of square meters of unknown space within a circular area surrounding the frontier. The information gain is multiplied by the information weight I_w and the hysteresis gain $h(x_f, x_r)$, both ranging from 0 to 1. Since this application prioritizes the rapid exploration of new areas rather than the exhaustive coverage of a single area, the information gain is reduced when the robot is already in the vicinity of the frontier under evaluation, thereby preventing the selection of spurious frontiers. Specifically, the hysteresis gain is set to zero when the robot is close to the frontier. Along the same line, the third term is the vicinity cost $V(x_r, x_f)$, which reduces the value of a frontier in inverse proportion to its distance when the robot is within a 5-meter radius of the frontier. The vicinity cost is multiplied by the vicinity weight V_w, which also ranges from 0 to 1. Together, the hysteresis gain and vicinity cost address objective O3.

The second term in Equation 4.1, the distance cost $D(x_r, x_f)$, accounts for the distance between the robot and the frontier, with greater distances resulting in higher costs (objective O2). This term reflects the preference, in this application, for visiting frontiers that are not too far away, ensuring that the travel time remains reasonable while avoiding increased risk.

The fourth term is the floor gain, $F(x_r)$, which weights the estimated unexplored space on the floor where the frontier is located (objective O4). Equation 4.2 shows how the floor gain is computed: it is the maximum area of the floor $A_{\max}(x_r)$, obtained from approximate knowledge of the building (indicated by the dashed red line in Figure 4.2), minus the currently explored area $A(x_r)$ (shown as the shaded area in Figure 4.2):

$$F(x_r) = A_{max} - A(x_r). \tag{4.2}$$

The floor gain is multiplied by the floor weight F_w, which is less than one and accounts for uncertainty in the remaining space to be explored. When comparing frontiers on the same floor, the floor gain is identical, rendering the effect of this term negligible.

The last two terms are optimization terms; their weight in the evaluation is smaller compared to the other terms, and they are primarily useful when two frontiers have similar values, helping to select the frontier that is more promising. Specifically, optimization requires the curve cost $C(x_r, x_f)$ and the orientation gain $O(x_r, x_f)$, each weighted by the respective factors C_w and O_w, ranging from 0 to 1. The curve cost, defined in Equation 4.3, is calculated as the ratio between the actual path length to reach the frontier $L(x_r, x_f)$ and the

Euclidean distance $D(x_r, x_f)$ (illustrated in Fig. 4.2), favoring straighter paths (objective O5):

$$C(x_r, x_f) = \frac{L(x_r, x_f)}{D(x_r, x_f)}. \tag{4.3}$$

Whenever the path cannot be computed using the A* algorithm [85], the frontier is considered unreachable, and the path length $L(x_r, x_f)$, and consequently the curve cost $C(x_r, x_f)$, are saturated to a high value. The orientation gain $O(x_r, x_f)$ is computed by evaluating the angle θ_r between the robot's heading and the straight line from x_r to x_f. If θ_r lies within the interval $[-15°, 15°]$, then $O(x_r, x_f) = 1$; otherwise, $O(x_r, x_f) = 0$. This encourages the robot to continue moving in the same direction (objective O6).

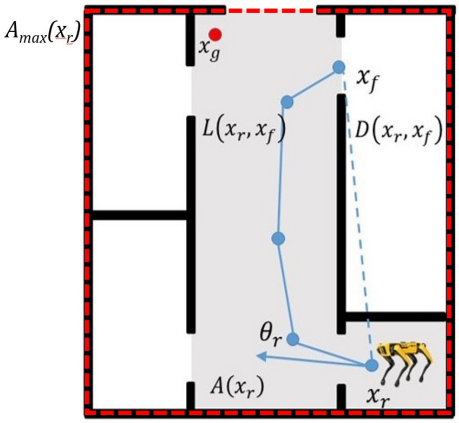


Figure 4.2 Evaluation of x_f revenue gain.

Stair Frontiers must be handled differently. Equation 4.4, describing the revenue for a Stair Frontier, is very similar to Equation 4.1 used for frontiers on the same floor as the robot; the primary difference is the absence of the information gain $I(x_f)$ when evaluating a Stair Frontier. Furthermore, the floor gain $F(newfloor)$ is computed not for the floor the robot is currently on, as before, but for the floor the robot is expected to reach by climbing the stairs:

$$\begin{aligned} R = & -D_w \cdot D(x_r, x_f) + V_w \cdot V_c(x_r, x_f) \\ & + F_w \cdot F(newfloor) + O_w \cdot O(x_r, x_f) - C_w \cdot C(x_r, x_f). \end{aligned} \tag{4.4}$$

When evaluating Stair Frontiers in Equation 4.4, the information gain is completely replaced by the floor gain computed for the new floor (thus, the equation addresses all objectives O1–O6). If the new floor is completely unexplored, the expected information gain from exploring the Stair Frontier corresponds to $A_{\max}$; otherwise, it corresponds to the

portion of the new floor that remains unexplored, as defined by Equation 4.2 and computed for the new floor.

It can be observed that the floor gain affects the decision only when comparing frontiers on different floors. It also influences the robot's exploration style: a higher floor gain corresponds to a "greedier" behavior, where the robot changes floors more frequently, relying more on the potential information to be gained by moving to a new floor. Conversely, lower floor gains promote a more cautious strategy, in which the robot attempts to fully explore the current floor before moving to another.

4.2 Experiments

The experimental procedure aimed at verifying the feasibility of this approach and to analyze the different behaviors and exploration styles that emerge by changing parameters. Tests are performed as follows in a four-story building:

- the exploration algorithm is executed for 10 minutes, by recording the area explored every minute;
- the exploration is repeated 9 times, by changing the values of F_w, C_w, and O_w in Eq. 4.1 and Eq. 4.4.

Even though the building has multiple floors, all experiments were performed on the same floor, simulating a fixed time of 1 minute to climb the stairs. This approach was adopted to ensure that differences in the results were not due to variations in accessible areas or the time required to climb stairs, which could have made the experiment more complex and less predictable. For the same reason, stairs were detected using Aruco markers, ensuring precise and reliable localization of the staircases. The building has the maps shown in Figure 4.3. The differences between the various floors have been simulated by opening and closing doors, thereby making certain areas accessible while blocking others on different floors.

Remark 1. *The Spot robot is capable of reliably climbing stairs using the embedded functionalities provided by the manufacturer; therefore, testing this capability is beyond the scope of this article. In contrast, since we focus on exploration strategies, possible delays or unexpected events during stair climbing could affect the explored area and the number of detected people within the allotted 10-minute time frame, making different runs incomparable.*

Concerning parameters in Eq. 4.1 and Eq. 4.4, one experiment is taken as a baseline and uses only the scores typically used in frontier-based algorithms, i.e. the information gain $I(x_f, x_r)$ and the distance to the frontier $D(x_f, x_r)$. The other 8 experiments were done a) by varying the floor gain F_w (chosen in the set $\{0.01, 0.05, 0.1, 0.5\}$, and b) by setting either zero or a positive value the two weights, O_w and C_w, that score the quality of the frontier depending on the straightness of the path to reach it.

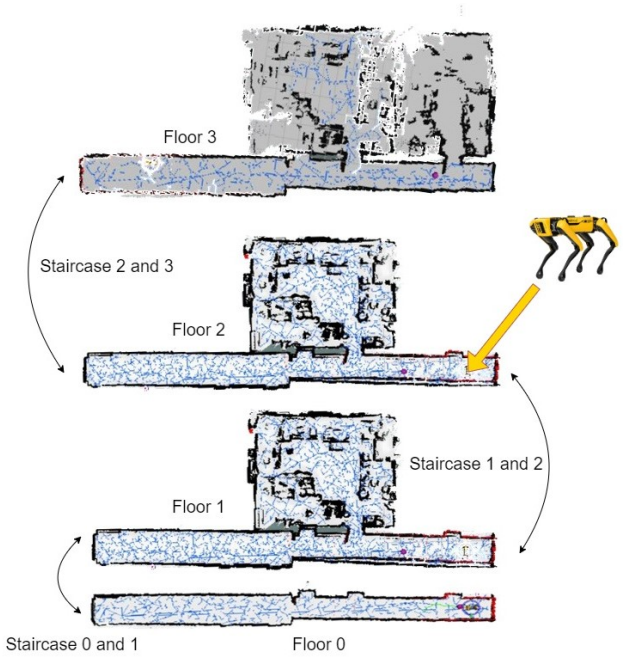


Figure 4.3 Results of the exploration test: these maps have been created using the most performing configuration of exploration parameters.

During all experiments, the robot was initially placed on the second floor at the rightmost side of the corridor, at the location pointed by the arrow in Figure 4.3. The robot was positioned near a staircase to enable it to recognize the Floor frontier from the start of the operation. Thanks to this, we avoid the case where the robot never finds a staircase during exploration: this must be avoided since the main goal is to check how tuning parameters change the robot's behaviour, considering the possibility of switching floors. For example, if the robot were to start on the rightmost side of the corridor on Floor 0, it would have to completely explore the floor before reaching the staircase, regardless of the choice of parameters. To maximize the effect of the parameters on the robot's exploration behavior, it was therefore decided to start all experiments from the same location, where the robot could

detect the staircases. Further experiments, with the robot starting from different positions, not all in proximity to staircases, are analyzed in Chapter 6.

As concerns the weights that account for the information gain I_w and the distance from the frontiers D_w, they have been kept constant in different experiments since they are not objects of the analysis.

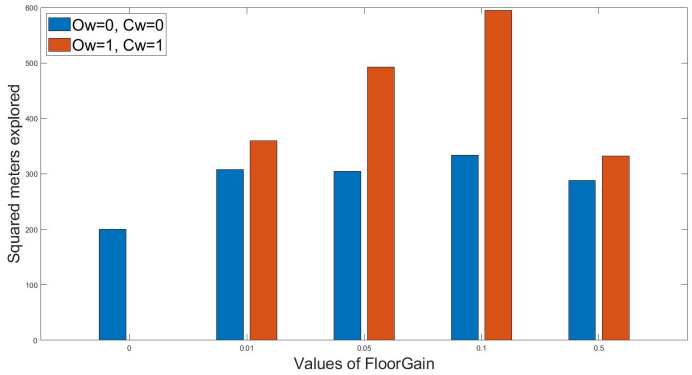


Figure 4.4 Squared Meters explored changing the Floor Gain of the exploration.

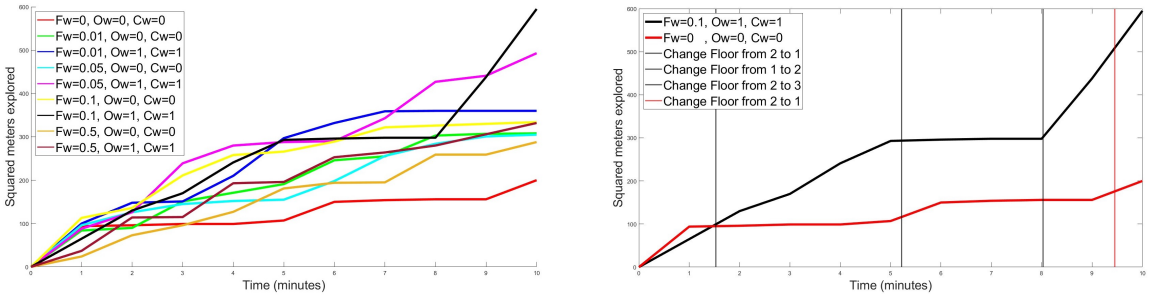
4.3 Results

Figure 4.4 represents the explored area after 10 minutes for each test. Blue columns correspond to setting parameters without aiming to reduce the tortuosity of the path (O_w and C_w set to 0). Orange columns instead correspond to setting both of them to 1. Please note that the choice of the optimization weights has been set by performing preliminary experiments both in simulation and with Spot, such as to ensure that the "reduction of tortuosity" plays a role only when comparing two frontiers with a similar expected information gain.

Compared with the baseline trial with $F_w = 0$ the performances increase by setting a positive value for F_w (blue columns). All parameter configurations achieve a wider explored area than the baseline trial, with a maximum when $F_w = 0.1$. With $F_w = 0.1$, $O_w = 1$, $C_w = 1$, we achieve the maximum explored area in 10 minutes, equal to $595m^2$. When F_w is too low, the behaviour will be similar to the baseline, where the robot waits for almost completing the exploration on each floor before moving to the next one.

Figure 4.5a shows how the explored area increase with time, whereas Figure 4.5b focuses on the more and the less performing trials by reporting also the time instants when the robot switches to a different floor (vertical lines). By inspecting Figure 4.5a, it can be noted that

the run in which the robot explored the biggest area ($F_w = 0.1$, $O_w = 1$, $C_w = 1$) was not the most performing throughout the entire period but only in the last minutes. This means that, given a shorter period, a different configuration of parameters might achieve better results. Figure 4.5b focuses on the best configuration ($F_w = 0.1$, $O_w = 1$, $C_w = 1$) that, after the second minute, outperforms the baseline one.



(a) Exploration Rate of all Trials.

(b) Exploration rate of Best and Worst trial.

Figure 4.5 Exploration rate comparison.

4.4 Discussion

By observing Figure 4.4, we can notice an increase in the area explored as F_w increases, but if F_w is too high, the robot changes the floor too often, without using the available time for exploration, but rather for switching between floors. Specifically, when $F_w = 0.5$, the robot immediately changes the floor, explores a small area, and changes the floor again, moving back and forth without actually achieving its main task, spending a lot of time climbing staircases.

By comparing the two sets of experiments in Figure 4.4 (i.e., red and blue columns), taking into account the quality of the path to a frontier always yields better results: when the robot has only 10 minutes to spend, it is better to quickly explore a floor moving straight along the main hallway, rather than wasting time entering rooms or exploring the small spaces in each room. For other applications or other time horizons, it might be convenient to set the parameters in different ways. Additionally, the differences with changing F_w values are less noticeable. From this, we can conclude that not taking into account the optimization parameters makes the exploration less efficient, resulting in the robot discovering fewer Stair Frontiers.

Analysing the behaviors of the robot in time, Figure 4.5a, we can see that the best performing trial at 10 minutes was not the best throughout the entire task. With only 5

minutes of exploration, for example, the best results would be achieved by setting $F_w = 0.01$: a lower value ensures higher coverage in the first minutes because the robot does not waste time climbing stairs, during which we assume that the robot is not exploring (this is not necessarily true in a real case, where people needing help might be on the stairs). It can also be noted that, after the third minute, the baseline configuration is always the worst, whereas it beats some other configurations in the first minutes. Once again, this happens in those cases where the configurations lead the robot to switch floors immediately, without exploring the initial floor.

By looking at the single trials, in Figure 4.5b, in the baseline configuration (red plot), the explored area grows slowly with time, since the robot thoroughly explores the second floor before moving away from it. This happens almost at the end: the robot reaches the new floor at 9.45, which allows it to explore only a small part of it. Generally speaking, the robot explores the majority of the first floor in the first minute and then it keeps discovering very small areas (see the plateau in the plot). In the most performing configuration (black plot), the robot decides very soon to switch to the first floor, and this produces a quick increment in the plot. When the first floor is almost completely covered, the robot switches back to the second floor. Since this floor has already been partially explored, the plot shows a very small increment versus time until the robot discovers the staircase leading to the third floor, on the other side of the hallway.

From the previous analysis, it can be remarked that the plot of the explored area versus time shows a plateau whenever a floor has been mostly explored. In this work, this information was not used to decide when to move to a different floor. However, it can be extremely useful in making a decision and will be studied in future implementations.

4.5 Conclusion

In this chapter, we presented a frontier-based approach for multi-floor exploration under time constraints. The proposed system was evaluated using the Boston Dynamics Spot robot to assess the impact of different parameters on the area explored within a fixed time budget of 10 minutes.

The experimental results highlighted the influence of the proposed floor-switching policy and path-selection strategy, which prioritizes less tortuous routes. These findings suggest that the proposed approach can improve exploration efficiency in time-critical scenarios, where rapid situational awareness is required before human rescuers arrive and begin operations.

Chapter 5

Probabilistic Semantic Mapping for Feature Localization

Based on requirements from FRs and harbor workers, Chapter 3, several applications involving feature detection using visual sensors were detailed during the exploration phase, such as identifying people or assessing the proper stacking of containers. We chose to focus on the SAR scenario due to its stricter operational requirements. In particular, during the exploration phase aimed at locating people and other relevant features, high localization accuracy is not strictly required – the robot's task is limited to exploring the environment and detecting people and other environmental features, and not intervening on the person through mobile manipulation or similar tasks. Rather, location information must be reported to FRs for subsequent intervention. Therefore, it is sufficient to provide positional accuracy that enables FRs to reach a location close enough to visually identify the affected individuals. Secondly, knowing with a high degree of confidence that an area is empty allows FRs to prioritize locations where people have been detected. For this reason, it is vital not only to detect all individuals present in the environment, but also to correctly identify empty spaces. By avoiding unnecessary visits to such areas, rescuers can ensure faster and safer interventions while minimizing the time spent inside the building.

These concepts are summarized in the following remarks, which guided the development and evaluation of our solution.

Remark 2. *In the considered SAR scenario, the locations of people and environmental features are primarily intended for use by FRs in a later phase. Consequently, it is more important to determine with high confidence that a person or relevant feature is present within visual range in a given area than to precisely localize it in space.*

Remark 3. *Knowing that no people are present in a given area is almost as important as knowing that people are present. Indeed, areas explored by the robot in which no people are detected can be safely ignored by FRs in a later phase, thereby improving both their personal safety and operational efficiency.*

Focusing on vision-based strategies in the context of SAR, we can summarize that the most critical information to gather includes: the number of people present and, if possible, their condition; the presence and location of wall cracks; an assessment of stairway conditions; and the positioning of doors, with the dual purpose of identifying areas inaccessible to the robot and, for future research, check for door collapes.

The contribution of this chapter is twofold: first, the development of a machine-learning (ML) model to detect people and relevant environmental features in SAR scenarios; and second, the integration of an ML-based vision system into a quadruped robot to generate semantic maps of the scene, identifying people and other relevant environmental features specific to SAR scenarios.

5.1 Methodology

Our goal is to recognize people and significant features in the environment during the locomotion of a quadruped robot and store them in a semantic map using only onboard sensors and computational units. We aim to develop a solution that can operate without the need for an internet connection, considering potential network disruptions or unavailability. Moreover, such a solution should be as hardware-independent as possible and applicable to various scenarios.

To minimize computational overload and enhance modularity, we partitioned the computations across two distinct onboard computational units. This approach facilitates the interchangeability of the two modules while ensuring reliable communication and data transfer between them, thereby creating a more robust application. As shown in Fig. 5.1, the system consists of an external stereo camera, a Detection Model for processing received images, a Recognition Interface that, given all information regarding the locomotion and navigation of the robot, computes the absolute position in space of the detected features, and a Mapping unit to save the location of detected features in space.

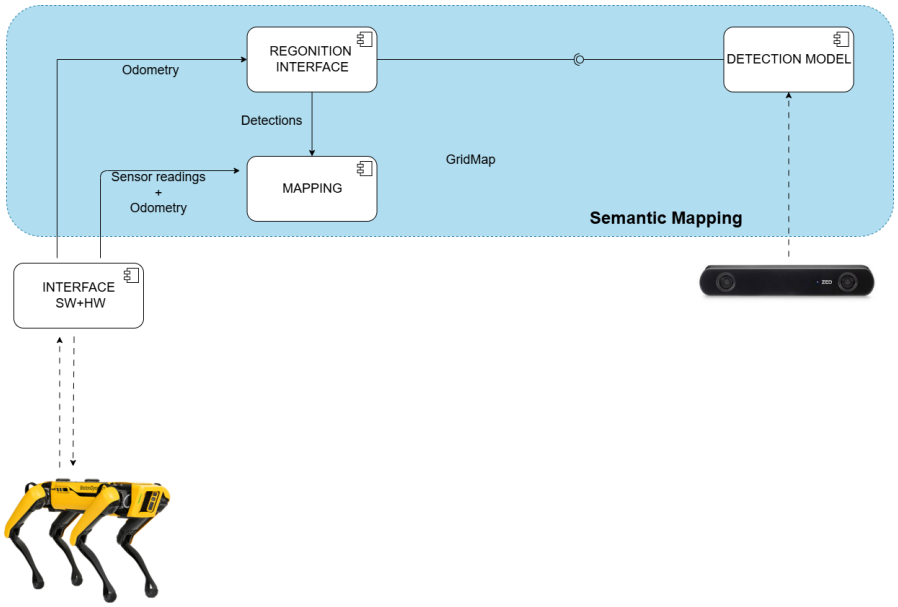


Figure 5.1 System Architecture.

5.1.1 Detection Model

For the Detection Model, our aim was to implement an algorithm capable of recognizing significant objects in the scene. Based on interviews with members of ANPAS, we decided to focus on detecting people, stairs, cracks, and doors. Recognizing the location of doors is particularly valuable for planning safe paths to areas the robot cannot explore. Furthermore, considering potential future applications, knowing the location of a door could facilitate implementing a system where the robot autonomously opens the door, an often underestimated problem that reveals to be critical for autonomous exploration of any indoor environment [81].

To implement a model capable of recognizing these classes, we needed to create a proper dataset by loading and labelling the images on Roboflow[1]. For detecting people, we utilized a subset of randomly chosen images from the COCO dataset [125]. Regarding crack detection, we compiled images from two different sources [128, 131] to include both wall cracks and concrete cracks. For stairs, we utilized images from a publicly available dataset [48], taken in various buildings, along with additional images of stairs collected in Genoa using smartphones. For door images, we relied on datasets publicly available at [146] and [7]. The final dataset comprises 6665 images with the instances for each class divided as follows: 10972 Person instances, 4552 Crack instances, 1430 Door instances, 1193 Stair instances.

[1]The dataset is publicly available at https://universe.roboflow.com/stair-eyhvv/rex-cue-2

Before deploying the dataset to train the model, we resized all images to 416×416, while maintaining their aspect ratio by padding the images with black borders. To expand the dataset and consequently improve the robustness and reliability of the system, we performed some augmentation steps. In particular, we applied the following transformations: horizontal flip, rotation between -15 °and 15°, brightness modification between -20% and 20%, blur effect up to 2.5px, noise addition up to 0.3% of pixels, cutouts of 3 boxes with 10% size each. The YOLO model was trained for 100 epochs with a batch size of 64, without freezing any layers.

During run-time operations, the model is used to detect objects in the scene using a ZED2 camera, considering only one of the two stereo frames. Subsequently, the object is localized with respect to the camera frame, see Section 5.1.2. From the detection model, the following information is retrieved: the class of the object, the confidence of the detection, and the position, with respect to the camera frame of the detected object. These data are then leveraged to create a semantic map of the environment. This vision system has several known limitations; in particular, detection can be made difficult by the viewpoint, possible occlusions, or the presence of smoke. These limitations are intrinsic to the choice of sensor. We are currently working on improving active perception using a robotic arm to avoid obstacle occlusions within the camera's field of view, as well as integrating different sensor types, such as microphones and thermal cameras, to compensate for these limitations and propose a more robust system.

Given the specificity of the application for which this system has been developed for, namely a SAR scenario with a legged robot, we opted to capture and process images from the camera at a fixed frequency of 2Hz, which seemed reasonable considering the maximum speed of the robot used for testing, 1.6m/s. This was done to mitigate a high computational overhead and minimize the hardware system requirements. To avoid a communication overload between the Computer Vision Unit and the Navigation Unit, we chose to avoid sending the camera stream, and instead transmit only the detections comprised of object class, confidence, and position with respect to the camera frame.

5.1.2 Semantic Map

The results of the detection process, including confidence of each detected class and object position, is transmitted to the Mapping unit. This unit also receives data from the physical robot regarding its spatial location through mapping and localization, performed using the rtab framework by IntroLab, [115]. Our objective is to correctly position the detected people,

cracks, doors, and stairs in the map by considering the current position and orientation of the robot in the environment.

When incorporating the detected features into the geometrical map, it is important to consider the confidence that the detection corresponds to ground truth rather than being perceptual noise. To this end, we took inspiration from probabilistic occupancy grids [56]. This process ultimately results in a multi-layered grid, where each layer holds thematic information about the spatial distribution and the confidence regarding the presence of: (1) obstacles, known and unknown space; (2) Person instances; (3) Crack instances; (4) Door instances; (5) Stair instances. It is important to observe that layers (2)-(5), similar to layer (1), do not depict detected features as objects. Instead, they present detected features in a raster format rather than a vector format, associating each cell with a confidence level of containing that specific feature. Layers (2)-(5) have a resolution of 0.5m, where each cell can range in value from 0 to 100, representing the confidence of detection in that location.

Once the detection is received, the position of the object in the map frame $\mathscr{M}$ is calculated using Eq. 5.1, where ${}^{\mathscr{M}}o$ is the object position with respect to $\mathscr{M}$; ${}^{\mathscr{M}}T_{\mathscr{B}}$ is the transformation matrix between the body frame of the robot $\mathscr{B}$ and $\mathscr{M}$; ${}^{\mathscr{B}}T_{\mathscr{C}}$ is the transformation matrix between the frame of the camera $\mathscr{C}$ and $\mathscr{B}$; ${}^{\mathscr{C}}o$ is the object position with respect to $\mathscr{C}$.

$$ {}^{\mathscr{M}}o = {}^{\mathscr{M}}T_{\mathscr{B}} \quad {}^{\mathscr{B}}T_{\mathscr{C}} \quad {}^{\mathscr{C}}o \tag{5.1} $$

The object is then added to the corresponding layer $\mathscr{M}^L$, depending on the class L of the detection. The process of updating the cell values for a given layer is explained in Algorithm 1. It is worth noting that, following each detection, 9 cells, each of area $0.25m^2$ in the layer centered in ${}^{\mathscr{M}}o$ are updated corresponding to a total area of $2.25m^2$. This has been done because, in our application, it is more critical to provide accurate information regarding the number of recognized features and the probability of recognition rather than pinpointing their exact location, Remark 2. By implementing this algorithm, each detected feature increases the confidence value of the surrounding space, allowing two nearby areas recording detections in the same layer to be merged into a larger detection.

Following the same logic as a probabilistic occupancy grid map, if the robot does not detect anything in a certain area of the map, that area should be progressively cleared of previous detections, if any. Considering the SAR application, it was deemed more important to recognize features within the environment (i.e., true positive, TP), rather than avoid false positives (FP), Remark 2

Since, however, the number of FP must be limited, a mechanism was implemented to periodically decrease the value of cells within the approximate field of view of the camera by

Algorithm 1: Map Update Algorithm

Input : N detections $\mathscr{O} = \{{}^{\mathscr{M}}o_1, \ldots, {}^{\mathscr{M}}o_N\}$,
a map layer $\mathscr{M}^i = \{map_i(1,1), \ldots, map_i(x,y), \ldots, map_i(\text{maxX}, \text{maxY})\}$

Output : updated map layer $\mathscr{M}^i$

for *each* ${}^{\mathscr{M}}o_i \in \mathscr{O}$ **do**
- $Cells \leftarrow \emptyset$;
- $(x,y) \leftarrow \text{getMapCell}({}^{\mathscr{M}}o_i)$;
- $Cells \leftarrow \{(x,y)\}$;
- $Cells \leftarrow Cells \cup 8\_\text{neighbours}(x,y)$;
- **for** $(j,k) \in Cells$ **do**
  - **if** $(j,k) = (x,y)$ **then**
    - $v \leftarrow \text{confidence}({}^{\mathscr{M}}o_i)$;
  - **else**
    - $v \leftarrow \lambda \cdot \text{confidence}({}^{\mathscr{M}}o_i)$;
  - **end**
  - $Prob \leftarrow 0.5 \cdot v + 0.5$;
  - $LogOdd \leftarrow \log\left(\frac{Prob}{1-Prob}\right) + \log\left(\frac{map_{j,k}}{1-map_{j,k}}\right)$;
  - $map_{j,k} \leftarrow 100 \cdot \frac{e^{LogOdd}}{1+e^{LogOdd}}$;
- **end**

end

a fixed amount. We approximated the camera's field of view to a triangular area with a vertex angle of 120°and a maximum depth of 4m, Figure 5.2. This simple mechanism allows for the removal of any incorrect detections, i.e. detections that are not repeatedly done by the robot.

Remark 4. *Since it is more critical to miss a detection (FN) than to have a FP, especially when searching for people, we decided to implement the clearing mechanism to operate at a slower rate (0.5Hz) compared to the mechanism that detects a cell as occupied (2Hz).*

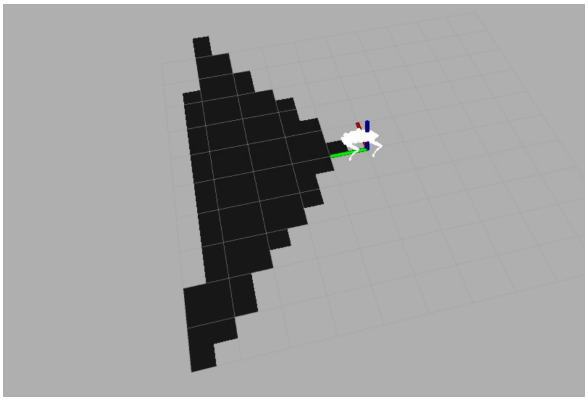


Figure 5.2 Clearing area in front of the robot.

5.1.3 Measurement

Assessing the quality of the semantic mapping is not trivial and can be considered as a contribution of this work. Suppose we now have a ground truth map for each layer of the semantic map, composed of cells with the same resolution, but containing the actual information provided by researchers about which cells should be labeled with each feature i. To evaluate the system, it is reasonable to compare the ground truth maps of people, doors, and stairs (semantic mapping experiment) and of people only (exploration and search experiment) with the corresponding obtained semantic maps. This comparison requires a clear definition of a successful detection, as it is not possible in this domain to directly apply metrics from multi-object classification or segmentation. Recall that the i^{th} layer of the semantic map reports, for each cell $map_i(x,y)$, the confidence that the cell contains an instance of element i. When the grid resolution is high, and each cell corresponds to a small portion of terrain, a single person, door, or staircase may be represented as a probability distribution spanning multiple cells, reflecting uncertainty in the actual position of features –where each cell in a semantic map layer can be considered occupied if its value exceeds a given confidence threshold.

As described in Section 2.3.1, the main metrics used for similar problems are IoU and MODA. In principle, IoU could be used to compute the degree of overlap between a layer i of the semantic map and the corresponding layer in the ground truth map. However, this approach is not suitable, as accurately computing the intersection of segmented areas requires highly precise localization and exact shapes of the areas corresponding to a feature—something that is difficult to achieve with our method—and may not even be necessary (Remark 2). Similarly, MODA proves inappropriate because it evaluates detections on a per-frame basis against ground truth rather than assessing their contribution to the final semantic representation, and it largely ignores the problem of correctly identifying empty areas. However, due to the nature of the application described, correctly identifying cells occupied by people (i.e., TP) is not the sole objective: accurately mapping empty areas in the semantic map (i.e., true negatives, TN) is equally important (Remark 3). This is particularly relevant when searching for people, since empty areas are generally larger than occupied ones and thus correspond to a greater number of cells (i.e., the dataset is unbalanced); exploring these empty areas would waste FRs' time.

In this section, we propose an alternative method for measuring the performance of a system designed to localize people (and other features) in a SAR environment. For illustrative purposes, we initially assume a very low resolution, in which each cell of the semantic map corresponds to an area of 2.25 m^2. Under this assumption, each feature is mapped to a single large cell. After thresholding the probability values in the semantic map layer, each cell is classified as either containing the feature or not. We also assume that the actual environment is divided into cells of the same resolution, referred to as the ground truth map. If a cell is detected by the system as containing a person (or another feature) and the corresponding cell in the ground truth map is also occupied, it is considered a TP. A FP occurs when a cell is detected as occupied but is empty in the ground truth map. Conversely, a cell that is occupied in the ground truth map but not detected by the system corresponds to a false negative (FN). In this simplified case with low-resolution maps, calculating the number of TNs is straightforward: TN corresponds to the cells that are empty in both maps.

Assuming that a person (or another feature) fully occupies a 2.25 m^2 cell or not is unrealistic, as it does not account for uncertainty in sensing and localization. As anticipated, we consider square cells with a smaller side (e.g., 0.5 m), and a detected feature is represented in the semantic map as a probability distribution over a 3×3 neighborhood of cells. Specifically, the alternative method we propose for computing TPs, FPs, and FNs is based on the following principle: in a SAR application, if a person in the ground truth map is within the visible range of their detected location in the semantic map (i.e., even if the detection does not

perfectly overlap), a FR reaching that location will be able to see the person and initiate rescue procedures (Remark 2). Accordingly, TPs, FPs, and FNs are defined as follows:

- TP: A person (or another feature) is considered *found* in the environment if there is at least one cell marked as occupied in the semantic map within a visibility radius of V_r meters from their actual position in the ground truth map. Each person (or other feature) in the ground truth map can produce at most one TP.
- FN: A person (or another feature) is considered *not found* if there are no cells marked as occupied in the semantic map within V_r meters of their actual position in the ground truth map. Again, each person (or other feature) in the ground truth map can produce at most one FN.
- FP: Neighboring cells in the semantic map marked as possibly containing a feature are first clustered using k-means [10] and the Elbow method [154]. An FP corresponds to clusters that are not within a distance V_r of any person (or other feature) in the ground truth map. Each cluster can produce at most one FP.

Starting from the number of TP, FP, and FN, we calculated Precision=TP/TP+FP, Recall=TP/TP+FN, and F1-score=2TP/(2TP+FP+FN). We do not calculate TN at this stage, as its computation is not straightforward from the preceding discussion. TN cannot be clustered, as done for FPs, and thus requires additional considerations. The estimation of TN is presented in Chapter 6.

5.2 Experiments

The Spot robot is equipped with 5 depth cameras that provide a 360-degree field of view. These cameras are monochrome and low-resolution, and therefore insufficient for accurate detections as they provide low-quality images and imprecise detections. To improve Spot's detection capabilities, we installed an external ZED2[2] camera from StereoLabs, a color depth camera commonly used for AI and Virtual Reality (VR) applications. In Fig. 5.3, the difference between images acquired from the ZED2 camera and the cameras already mounted on the robot can be observed.

The ZED2 camera requires an NVIDIA GPU and a USB 3.0 connection. The Spot Core, an external computation unit provided by Boston Dynamics, offers a USB 3.0 port but does not include an NVIDIA GPU. Instead, it features an i5 Intel® 8th Gen (Whiskey Lake-U)

[2]https://www.stereolabs.com/products/zed-2

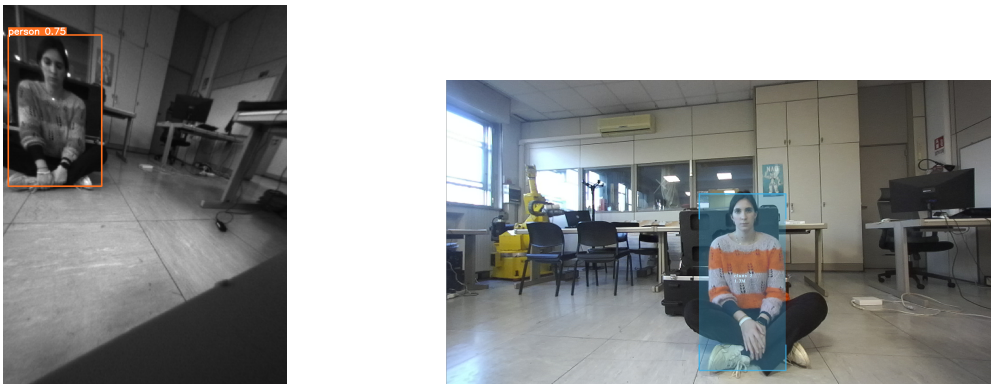


Figure 5.3 Spot (left) and ZED2 (right) cameras with a detected Person instance (Spot has two black&white frontal cameras.)

Core™ CCG Lifecycle. Consequently, a separate computational unit is necessary. For the computer vision component, we selected the Jetson Nano Developer kit, which is equipped with a 128-core Maxwell NVIDIA GPU and a Quad-core ARM A57 @ 1.43 GHz CPU. Summarizing, the Jetson Nano Developer kit hosts the Computer Vision Unit, while the Spot Core incorporates the Navigation Unit, integrating the navigation algorithm with information from the Computer Vision Unit.

To have a precise localization and a mapping algorithm, we installed a VLP-16 LIDAR sensor on top of the robot. This sensor generates a point cloud that can be exploited to create a precise map of the environment.

To securely mount the ZED2 camera on top of the Spot robot, we developed a 3D printed lightweight support, as shown in Fig. 5.4. This support ensures the stability of the ZED2 position and prevents it from falling during motion. Additionally, the platform accommodates the Computer Vision unit, and a power bank, which supplies power to both the Computer Vision unit and the camera.

5.2.1 Detection Model

The ZED2 provides an SDK that enables easy integration with ML models YOLOv5 [95] and YOLOv7 [197]. Unfortunately, the NVIDIA Jetson Nano is only compatible with YOLOv5, and therefore We opted to train this latter model. Specifically, our focus was on the YOLOv5 nano, small, and medium models (YOLOv5n, YOLOv5s, YOLOv5m). We trained all three models and collected their performance metrics.

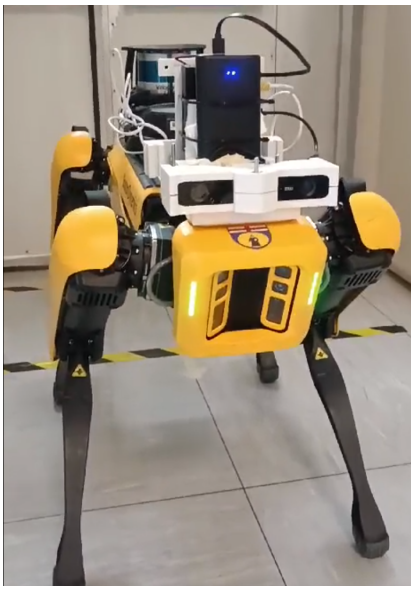


Figure 5.4 Spot System with ZED2 camera mounted.

5.2.2 Semantic Mapping

The experiments aimed to test the functionalities of the semantic mapping algorithms, rather than assessing the motion abilities of the robot. The experiment involved teleoperating the robot along a predefined path marked on the floor, using stickers with arrows, while the robot acquired and processed images. This experiment was conducted at two different locations, with 5 participants at each location. Each participant completed the task 5 times, with the experimenter changing the position of people and the states of doors (open/closed) before each run. During the experiments it happened for two trials that the data were not recorded correctly and the corresponding data were discarded. Consequently, we completed a total of 48 trials.

Remark 5. *We decided to manually drive the Spot robot along a predefined path to ensure that the final robot's trajectory is as repeatable as possible, something we could not ensure when the robot was autonomously exploring the environment. For the same reason, we instructed operators to control the robot along the path without purposefully turning its camera toward people or relevant environmental features.*

Due to the absence of cracks in the chosen locations, we did not test the detection of Crack instances. In the experiments described in the following section, our focus is on evaluating the detection of Door, Person, Stair instances[3].

[3]A video of experiments is available at https://youtu.be/lCpxnwUiR6I

The first location was on the second floor of E Building of the DIBRIS Department at the University of Genova, located at Via Opera Pia 13. This site featured 20 doors and two staircases—one ascending and one descending—that the robot could encounter and detect. Regarding human presence, two individuals were positioned in five distinct configurations, corresponding to the five runs in which each participant was involved. In each configuration, the individuals alternated among three poses: sitting, standing, and lying down.

The second location was the ground floor of Villa Bonino, headquarters of DIBRIS and located at Viale Causa 13, Genova. This site featured 9 doors and 3 staircases, with one leading downstairs and two upstairs. Conducting experiments in an entirely vacant building at this location was not possible, leading to scenarios where more than two individuals were present in the environment. Specifically, in 12 runs, 3 individuals were present, and in 4 runs, there were 4 individuals. Given that the positions of these additional people were not fixed in advance, we utilized the recorded footage from the robot's cameras during its movement to identify and document their locations during data analysis.

5.3 Results

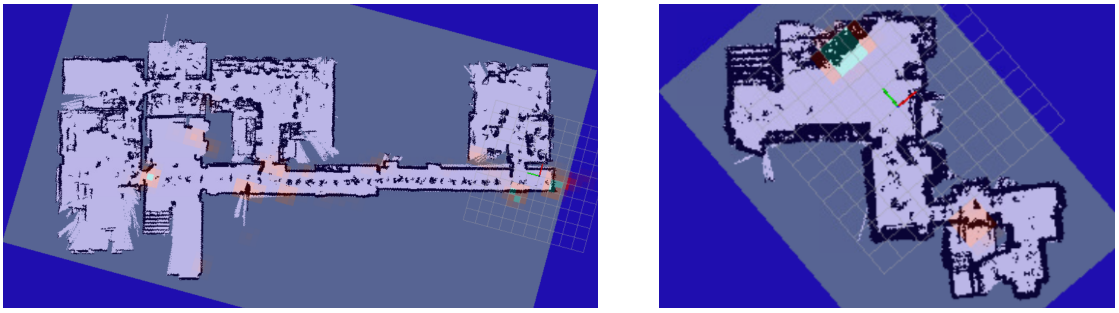
5.3.1 Detection Model

The results of the different detection models trained are summarized in Table 5.1. The 'P' column stands for Precision=TP/(TP+FP), which measures the accuracy of the detections, i.e., how many of the instances predicted as positive are actually positive. 'R' stands for Recall=TP/(TP+FN), evaluating the proportion of actual positive cases that the model correctly predicts as positive. The other two metrics are useful for integrating the information from Precision and Recall into a single value. The 'MaP50', which stands for Mean Average Precision at IoU 50, calculates the area under the Precision-Recall curve for each class when the IoU is greater than 50, and then averages the results. Likewise, the 'MaP50-95' is the Mean Average Precision at IoU thresholds from 50 to 95. It computes the Precision-Recall curve for different threshold levels (ranging from 50 to 95) and averages the area under the curve.

We tested all of the aforementioned models on the available hardware and observed that the difference in performance between the models was not significant. However, the inference time was notably slower for the small and medium models. To ensure a faster pipeline for processing images at the 2Hz rate, we decided to evaluate our algorithm using

	MaP50	MaP50-95	P	R
YOLOv5n	0.517	0.251	0.609	0.499
YOLOv5s	0.544	0.2	0.649	0.508
YOLOv5m	0.584	0.32	0.681	0.559

Table 5.1 Performance for different YOLOv5 models.



(a) Obstacles (black) and Door instances (yellow) at the first location.

(b) Obstacles (black) and Person instances (yellow) at the second location.

Figure 5.5 Detected instances in two different locations.

the model based on YOLOv5n. The performances of this model for each class are presented in Table 5.1.

From these results, it is possible to notice that the class performing worse is the Crack class in all the aforementioned metrics, followed by the Door class. These results might be likely be improved by modifying the dataset to make it more robust and diverse with a greater variety of images. However, in the present work, the performance are deemed sufficient to pass to the next phase, where detected features should be incorporated into the map's layers.

5.3.2 Semantic Mapping

Figure 5.5a shows the layer for obstacles, in black, and the one for Door instances, in yellow, at the first location. Figure 5.5b displays the layers for obstacles, in black, and the one for Person instances, in yellow, at the second location.

Specifically, to compute metrics, we recorded the obtained layers of the semantic map, and the experimenter recorded a ground-truth map for each class –Door, Person, Stair – based on the actual position of the instances of the corresponding class in the environment. We refer to the actual instances of people and environmental features in the ground truth map as 'sources', where each source is associated with a corresponding class L.

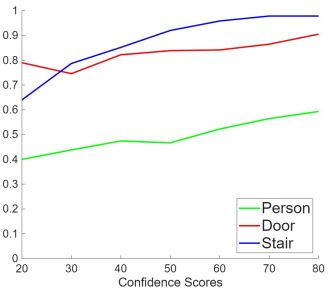
As a first step, we decided to process the map layers $\mathscr{M}^L$ by filtering them based on such confidence level. To accomplish this, each cell of a map layer $map_{i,j}$ was considered as 'occupied' if the confidence level was greater than or equal to a given threshold. We repeated this analysis for the following set of confidence thresholds $th \in \{20,30,40,50,60,70,80\}$.

For the experiments performed in the first location, we obtain TP=35, FP=40, FN=13 when setting a threshold of $th = 50$. With this threshold, it is evident that we have a significant number of FN=13 instances of Person not detected. As previously mentioned, in this application, FNs are significantly more critical than FPs, especially for the Person class. Spending time investigating a FP is not a major concern, given that the number of false alarms is relatively low compared to the total exploration time of the robot, Remark 4. On the other hand, missing a person needing help (FN) can have dramatic effects. In the second location, we obtain TP=65, FP=29, FN=3 when setting a threshold of $th = 50$. It is noticeable that the number of FN decreases, alongside an increase in the number of TPs. Please notice that, as mentioned earlier, the more relevant score is Recall, which measures the model's ability not to miss actual instances to be detected, i.e., it decreases as the number of FNs increases. Limiting the number of FP, as measured by Precision, is less relevant given that the time spent by rescuers to manually check TPs and FPs is low compared with the total exploration time, Remark 4. The F1-Score combines information from both Precision and Recall metrics and is useful for seeking a balance between them.

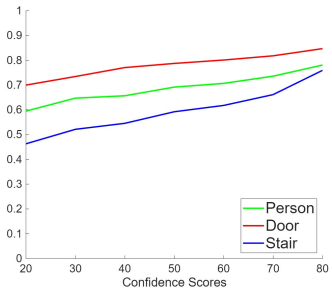
The plots of Precision, Recall, and F1 score versus the designated threshold th for the first location are shown in Fig. 5.6. From these plots, it is possible to notice that the Precision, Figure 5.6a, increases with th for all classes, whereas Recall, Figure 5.6c decreases for all classes. By considering the F1-score Figure 5.6e, it is observed that it decreases with th for the Door class, but increases for the People and Stair classes.

The plots of the four metrics versus the threshold th for the second location are shown in Figure 5.6. The Precision, Figure 5.6b, increases as the confidence threshold, th. On the other hand, the Recall, Figure 5.6d, remains high for the Person and Door classes but decreases for the Stair class. The F1-Score, Figure 5.6f, remains high for the three classes and tends to increase with an increase in the confidence threshold.

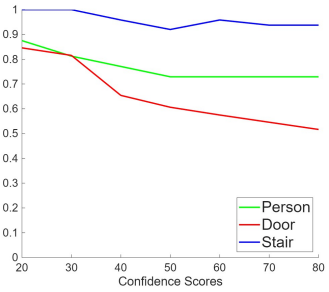
For each class, we also compared the layers of the ground-truth map, manually created to store information about the actual position of sources during the experiments, with the layers of the map obtained during the experiments. To achieve this, we first binarized each layer of the map using a confidence threshold $th \in \{20,30,40,50,60,70,80\}$, in the same way as detailed above. Then, for each run, we calculated the IoU for the binarized layer and



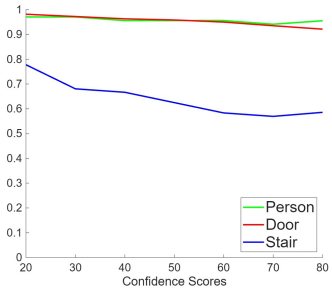
(a) Precision: first location



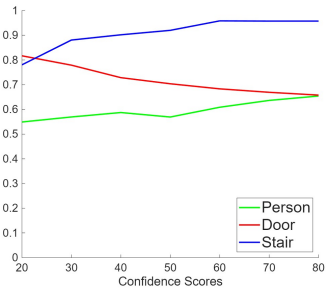
(b) Precision: second location



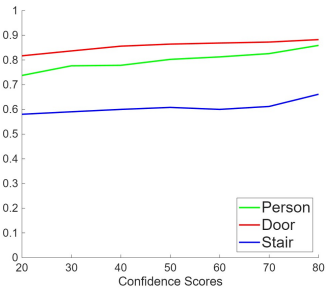
(c) Recall: first location



(d) Recall: second location



(e) F1-score: first location



(f) F1-score: second location

Figure 5.6 Performance metrics at both locations.



(a) IoU: first location

(b) IoU: second location

Figure 5.7 IoU results for the different experiments

the corresponding layer of the ground-truth map. Finally, we averaged the results over all runs. In Figure 5.7, the IoU for different thresholds th are shown for the two locations.

Given that the F1 score, which balances precision and recall, improves as the confidence level increases for both locations, we propose a more detailed analysis of the results obtained with a confidence threshold set to th=80.

Table 5.2 presents the results of the semantic mapping algorithms at the first and second locations. Specifically, Table 5.2a shows the confusion matrix for the People class at the first location, while Table 5.2b shows the corresponding matrix for the second location. Similarly, Tables 5.2c and 5.2d report the confusion matrices for the Door class at the first and second locations, respectively. Finally, Tables 5.2e and 5.2f present the confusion matrices for the Stair class at locations 1 and 2, respectively.

5.4 Discussion

The experiments performed confirm the feasibility of the approach and pave the way for future developments in which Spot will autonomously explore the environment, possibly alerting rescuers to intervene at a later phase to verify whether detected people and other features correspond to the actual situation in the real world.

Experiments yield different results in terms of TP, FP, and FN depending on the threshold used to filter semantic maps and the environment where they were performed, i.e., either the first or the second location. This can depend on the fact that the two environments are quite different, with wider corridors, doors, and stairs at the E Building of DIBRIS, and narrower ones in Villa Bonino. Also, results depend on the specific configurations that have been chosen to distribute people in the environment. Please remember that the operators controlling the robot are not allowed to deviate from the path or turn the robot to better

Loc 1 People	Predicted Positives	Predicted Negatives
Actual Positives	35	13
Actual Negatives	24	–

(a) Confusion matrix for the People class in the first location.

Loc 2 People	Predicted Positives	Predicted Negatives
Actual Positives	64	3
Actual Negatives	18	–

(b) Confusion matrix for the People class in the second location.

Loc 1 Doors	Predicted Positives	Predicted Negatives
Actual Positives	248	232
Actual Negatives	26	–

(c) Confusion matrix for the Door class in the first location.

Loc 2 Doors	Predicted Positives	Predicted Negatives
Actual Positives	199	17
Actual Negatives	36	–

(d) Confusion matrix for the Door class in the second location.

Loc 1 Stairs	Predicted Positives	Predicted Negatives
Actual Positives	45	3
Actual Negatives	1	–

(e) Confusion matrix for the Stair class in the first location.

Loc 2 Stairs	Predicted Positives	Predicted Negatives
Actual Positives	41	29
Actual Negatives	13	–

(f) Confusion matrix for the Stair class in the second location.

Table 5.2 Results for the Semantic Mapping experiments.

visualize interesting features, and therefore the robot - in most cases - sees them in the periphery of its visual field, Remark 5.

For instance, when setting the threshold $th = 50$ and considering Person instances in the first and the second location, the number of TP increases from 35 to 65, respectively; the number of FP decreases from 40 to 29; and the number of FN decreases from 13 to 3. This difference can be attributed to the fact that, in the former location, two persons were asked to position themselves in two very challenging configurations, lying on the floor, once in a room, and once on the staircase going up. In these configurations, the instances of people were scarcely recognized. This issue could stem from deficiencies in the model, which might be mitigated by incorporating additional images of people lying on the floor. Moreover, results could potentially be enhanced by permitting the robot to briefly adjust its orientation towards them during motion to capture better images.

Generally speaking, Precision and Recall are better for the Person and Door classes in the second location than in the first one. This might be due to more challenging configuration for people. In contrast, the Stair class performed better in the experiments at the first location but was the worst-performing class in the second location. This discrepancy can be explained by the smaller size of the stairs in Villa Bonino and the pre-determined path, that required the robot to recognize them only with peripheral vision.

These considerations also hold when comparing IoU scores, resulting in higher scores for the second location concerning the Door and Stair classes and higher scores for the first location concerning the Person class.

By looking specifically at the results for $th = 80$, in both locations, the number of FPs—instances incorrectly detected as people—is higher than expected. However, it must be compared to the number of TN, which will be calculated in Chapter 6. It is important to note that the dataset is unbalanced, with the majority of areas containing no detectable features, which confirms the importance of estimating TN and evaluating the ability of the system to correctly identify empty areas.

5.5 Conclusion

In this chapter, we presented a system capable of integrating semantic information extracted by a YOLO-based detector into the layers of a semantic map. The system was specifically designed for SAR scenarios, addressing challenges that are characteristic of these environments.

The proposed approach was evaluated using the Boston Dynamics Spot robot in multiple locations. The results showed that the system successfully detected the majority of Person, Door, and Stair instances while maintaining a manageable number of false positives. Although false detections were occasionally generated, their impact on the overall mission was limited, as they could be quickly verified by rescuers compared to the time required for autonomous exploration. These results demonstrate the potential of the proposed semantic mapping framework to support situational awareness in SAR operations.

Chapter 6

Integrating Exploration with Semantic Mapping

Following the results from Chapters 4 and 5, we propose the integration of the two subsystems to evaluate the feasibility of the complete system, both capable of performing an autonomous exploration task within a limited time while also collecting information about the presence and location of relevant features.

We first describe the novelty in the integration of the two subsystems and how we propose this integration. Subsequently, to better highlight the differences in what was proposed before, both in Chapters 4 and 5, and what is present in the literature, we complete a brief survey of software solutions relevant to the SAR operational environment explored in this article, namely, multi-floor exploration and the localization of affected people and other relevant features.

6.1 Methodology

While focusing on the inspection task in a SAR scenario it is important to notice that there are some peculiarities. First, during the exploration phase aimed at locating people, high localization accuracy is not strictly required – the robot's task is limited to exploring the environment and detecting people, and not intervening on the person through mobile manipulation or similar tasks. Rather, location information must be reported to FRs for subsequent intervention. Therefore, it is sufficient to provide positional accuracy that enables FRs to reach a location close enough to visually identify the affected individuals.

Second, knowing with a high degree of confidence that an area is empty allows FRs to prioritize locations where people have been detected. For this reason, it is vital not only

to detect all individuals present in the environment, but also to correctly identify empty spaces. By avoiding unnecessary visits to such areas, rescuers can ensure faster and safer interventions while minimizing the time spent inside the building.

These concepts are summarized in the following remarks, which guided the development and evaluation of our solution.

Remark 6. *In the considered SAR scenario, the locations of people are primarily intended to inform FRs in a later phase. Consequently, it is more important to determine with high confidence whether a person is present within visual range in a given area than to precisely localize them in space.*

Remark 7. *Knowing that no people are present in a given area is almost as important as knowing that people are present. Indeed, areas explored by the robot in which no people are detected can be safely ignored by FRs in a later phase, thereby improving both their personal safety and operational efficiency.*

We propose an important novelty in the metric used to assess the quality of robotic SAR. In particular, we introduce a method for estimating the number of TNs as follows, which was omitted in Chapter 5.

Estimating the number of TNs requires particular care, since empty cells cannot be clustered (as is done for FPs in Chapter 4) to coherently define areas that are empty in both the semantic map and the ground truth map. Therefore, we adopt the following rationale. An ideal correct detection occupies an area C_s (e.g., $9 \times 0.5 \times 0.5$, corresponding to 9 cells, each with a side length of 0.5m). The area associated with a single FN is assumed to be equal to that of a TP, i.e., the area occupied by a person (and, by extension, by a generic feature) in the ground truth map. Similarly, the area occupied by an FP—even if the corresponding cluster in the semantic map contains more than 9 cells—can be approximated as equal to that of a single feature, C_s, and is therefore considered equivalent to FN and TP.

Following this rationale, the estimated number of classifications to be performed in an environment of area A_s is given by A_s/C_s, where each classification can result in a TP, FP, TN, or FN. The number of TNs can then be estimated as $TN = A_s/C_s - TP - FP - FN$.

Additionally, since we expect the dataset to be strongly unbalanced toward areas without any feature to be detected, we compute and report Sensitivity $= \frac{TP}{TP+FN}$, which measures the proportion of true positives among all actual positives in the environment, Specificity $= \frac{TN}{TN+FP}$, which measures the proportion of true negatives among all actual negatives (i.e., empty space), and the Balanced Accuracy, defined as the average of Sensitivity and Specificity. Additionally, we compute and report Matthews Correlation Coefficient MCC, MCC $=$

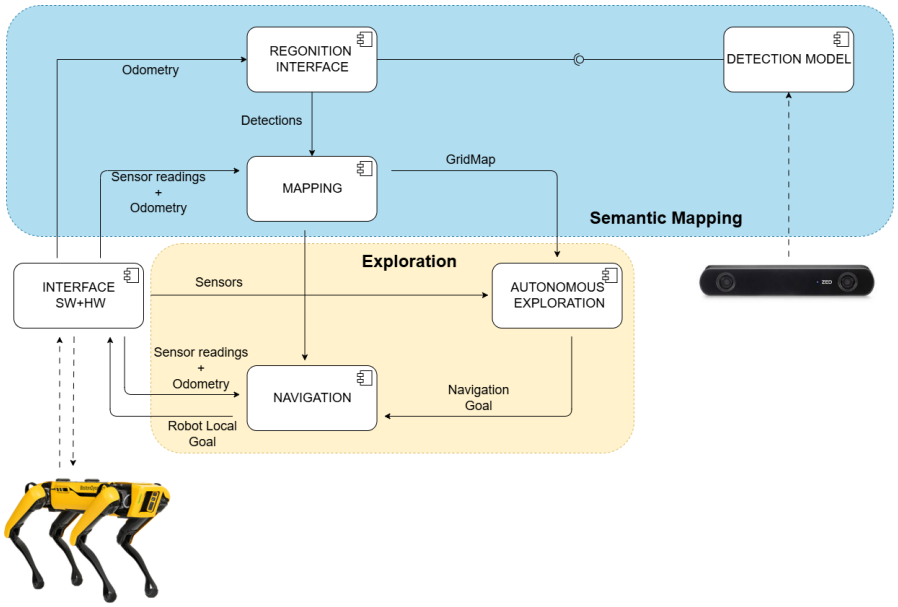


Figure 6.1 System architecture and interface. The Spot robot is equipped with a ZED2 camera connected to a Jetson Nano for GPU-based image processing, a Velodyne LiDAR, and a Spot Core for running all the required software modules.

$\frac{TP \cdot TN - FP \cdot FN}{\sqrt{(TP+FP)(TP+FN)(TN+FP)(TN+FN)}}$, which measures the correlation between predicted and true labels in binary classification problems, particularly in the presence of class imbalance. Unlike the F1-score, the MCC incorporates all four components of the confusion matrix (TP, TN, FP, and FN) and is therefore appropriate when the costs associated with positive and negative errors are comparable. The MCC ranges from +1, indicating perfect prediction, to -1, indicating total disagreement, with 0 corresponding to random predictions.

We then propose the integration of a system for multi-floor autonomous exploration, supporting different exploration styles, with a semantic mapping system specifically designed for SAR scenarios, and their implementation within a Robot Operating System (ROS)- based architecture governing the behavior of a quadruped robot. Additionally, we complete an experimental evaluation with the Spot robot to assess the performance of the proposed system under different exploration styles.

Figure 6.1 illustrates the system architecture for a mobile robot capable of autonomously exploring the environment while recognizing relevant features and people, and recording their locations in a semantic map. For clarity, the system is divided into two components: the Semantic Mapping component (highlighted in light blue, and described in Chapter 5) and the Exploration component (highlighted in yellow and described in Chapter 4).

6.2 Experiments

To evaluate the novel metrics, estimating the empty space correctly identified, and evaluating the detection system, knowing it is an unbalanced dataset, we reuse the dataset collected in Chapter 5, focusing exclusively on the detection of People. Additionally, the system is evaluated in terms of its ability to accurately identify both the number and location of people, while also assessing how this performance is influenced by the exploration style. In particular, two research questions are examined:

- (R1) Does the semantic map generated by the system accurately represent the number and location of relevant features, as well as the extent of empty space?
- (R2) How does the exploration strategy affect the total area explored, the detection of relevant features, and the extent of empty space in the environment?

The choice of evaluating just people is motivated by the comparison of different multi-floor exploration strategies aimed at maximizing the use of the allotted exploration time before FRs intervene. Second, it motivated the introduction of the estimation of TNs.

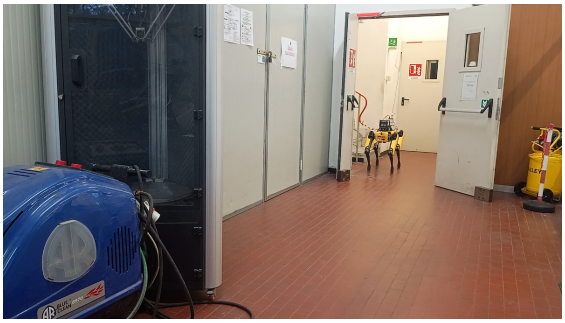
Semantic Mapping

To evaluate the dataset collected in Chapter 5, we calculated the confusion matrices at $th = 80$ for the People class, with the estimation of TNs. Table 6.1 presents the results of the semantic mapping algorithms at the first and second locations.

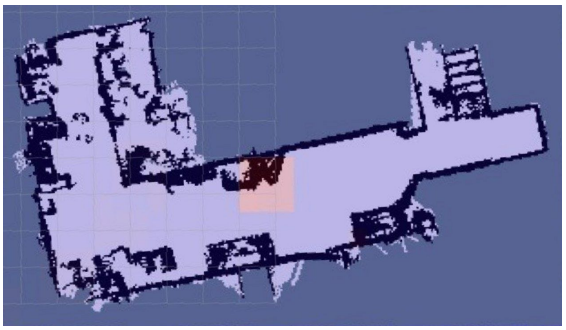
Exploration and Search

In Chapter 9, we already evaluated the impact of the weights in Equations 4.1 and 4.4. However, a key difference with the present work is that only the coverage achieved during geometric exploration was considered, while the system was not capable of detecting features and building a semantic map.

By contrast, this chapter proposes the integration of exploration and semantic mapping, and evaluates the resulting impact of different multi-floor exploration strategies on the system's capability to identify victims. To this end, we considered only O4, O5, and O6 in order to analyze how floor switching and path optimization affect the robot's ability to explore larger areas. Specifically, we focused on the contribution of the floor weight F_w and the curvature and orientation weights O_w and C_w, by enabling ($Opt = ON$) or disabling ($Opt = OFF$) path optimization. $Opt = ON$ indicates that the optimization terms $O(x_r, x_f)$



(a) A corridor at the third location used in the exploration and search experiment.



(b) Results of the exploration and search experiment in the third location, map layer for the class People. Obstacles are shown in black and detected features in yellow.

Figure 6.2 Third location: environment and corresponding exploration results.

and $C(x_r, x_f)$ in Equations 4.1 and 4.4 are taken into account, whereas $Opt = OFF$ sets both terms to 0.

Four different parameter configurations were tested to influence the exploration style. The first configuration $(F_w = 0, Opt = ON)$ favors thorough exploration, in which the robot attempts to fully explore the current floor before moving to the next. The second and third configurations are balanced $(F_w = 0.3)$; the difference between them lies in the optimization parameters, which are enabled $(Opt = ON)$ and disabled $(Opt = OFF)$, respectively. Finally, the fourth configuration $(F_w = 1, Opt = ON)$ represents a "greedy" strategy, in which the robot prioritizes exploring multiple floors. For each parameter configuration, the experiment was repeated 5 times, changing the floor the robot was starting on, the starting position within the floor, and the location of people in the environment (one person per floor, either standing or sitting).

For this set of experiments, the robot was tasked with exploring a four-floor building for 10 minutes per trial. Figure 6.2 shows part of the environment corresponding to the third location. The considered scenario does not reproduce a fully collapsed disaster site. Damage may range from slight to total collapse [80], and INSARAG procedures include room-to-room clearance of non-collapsed structures [93]. Accordingly, this work focuses on environments where robot mobility remains feasible, while local obstacles, debris, and uneven terrain are handled by Spot's low-level locomotion and control.

Even though the building has multiple floors, all experiments were performed on the same floor, simulating a fixed time of 1 minute to climb the stairs. This approach was adopted to ensure that differences in the results were not due to variations in accessible areas or the

time required to climb stairs, which could have made the experiment more complex and less predictable, Remark 1.

For this set of experiments, it was not possible to guarantee that the building was empty, as it is normally used by researchers and university students during their daily activities. Therefore, in addition to the person placed in a controlled location by the experimenter, four additional "unplanned" individuals were present across three different trials. For the analysis of these trials, the locations of all people in the environment were determined a posteriori using video recorded by the robot's onboard cameras during the experiments.

Metrics for geometric exploration, that is, the average explored area per floor A_f, the total explored area A_{tot}, the number of visited floors V_f, the number of floor changes C_f, and the total exploration time t_{exp}, are computed to assess how coverage changes under different exploration strategies. SAR classification metrics, as well as standard IoU and IoU with inverted classes, are computed to assess how the detection of people changes.

6.2.1 Simulated exploration

A simulator was developed in Python to confirm the effects of different parameter configurations on the exploration strategy observed in the real-world experiments[1]. The simulator provides both a bird's-eye view of the environment and the generated map, as well as a 3D first-person rendering of the environment. It can be used to test the exploration strategies defined by Equations 4.1 and 4.4, as well as the people-detection functionality, as shown in Figure 6.3. However, simulating detection accuracy was not considered meaningful in the absence of actual video streams. Therefore, the simulations presented here focus exclusively on exploration metrics (A_f, A_{tot}, V_f, C_f, and t_{exp}). Note that the architecture evaluated in simulation is not ROS-based, as the purpose is to assess the impact of the exploration strategies defined by Equations 4.1 and 4.4 independently of the specific robotic implementation of the SLAM and navigation algorithms.

Simulated experiments were conducted using the same parameter configurations as the real-robot experiments to compare exploration metrics across the six conditions defined by all combinations of $F_w \in \{0, 0.3, 1\}$ and $Opt \in \{ON, OFF\}$. A total of 170 randomly generated four-storey buildings, with the same overall floor area but different room layouts, door locations, obstacles, and initial robot positions, were evaluated under each condition, resulting in 1,020 simulation runs. Each run had an allotted duration of 10 minutes, with 1 minute reserved for transitions between floors.

[1] https://github.com/ZoeBetta/exploration_semantic_simulation.git

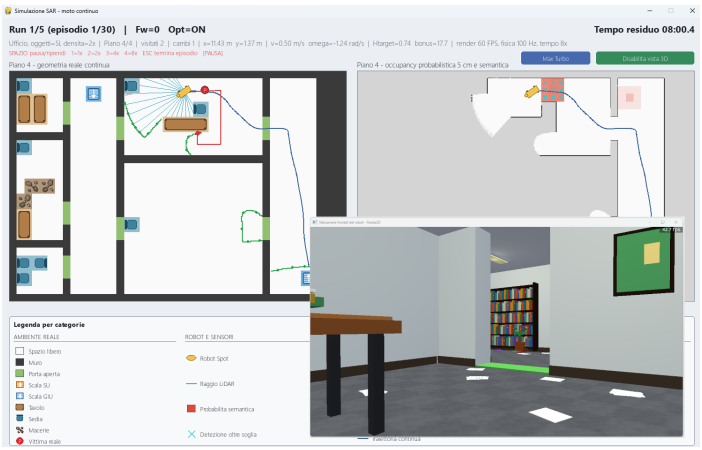


Figure 6.3 Simulator interfaces showing the bird's-eye environment and map views, together with the 3D first-person rendering.

6.3 Results

6.3.1 Semantic Mapping

Specifically, Table 6.1a shows the confusion matrix for the People class at the first location, while Table 6.1b shows the corresponding matrix for the second location.

As expected, the dataset is strongly unbalanced toward areas without any features to be detected. For this reason, it is important to use metrics that are appropriate for unbalanced datasets.

For each class, Table 6.1c reports SAR-Sensitivity, SAR-Specificity, SAR-Balanced Accuracy, and SAR-MCC, where the SAR prefix indicates that TPs, FPs, TNs, and FNs have been computed according to the method described in Chapter 5 and in Section 6.1. The same table also reports the IoU and the IoU computed with inverted classes for comparison, both calculated cell by cell between the ground-truth and semantic maps.

The results indicate that recognition performance is largely consistent across the two locations, with the highest performance obtained in the second location.

6.3.2 Exploration and Search

Concerning exploration, in Tables 6.2, 6.3, and 6.4e, the first, second, and third rows correspond to configurations with $F_w = 0$, $F_w = 0.3$, and $F_w = 1$, respectively. Recall that a higher F_w indicates that the robot is more inclined to explore Stair Frontiers to reach unexplored floors. In all of these configurations, `Opt=ON`, meaning that the Curve weight C_w and the Orientation weight O_w are set to nonzero values. The fourth row corresponds to

Loc 1 People	Predicted Positives	Predicted Negatives
Actual Positives	35	13
Actual Negatives	24	4097

(a) Confusion matrix for the People class in the first location.

Loc 2 People	Predicted Positives	Predicted Negatives
Actual Positives	64	3
Actual Negatives	18	759

(b) Confusion matrix for the People class in the second location.

Location	SAR-Sensitivity	SAR-Specificity	SAR-Bal. Acc.	SAR-MCC	IoU	IoU inv. class.
People Location 1	0.729	0.994	0.862	0.653	0.345	0.997
People Location 2	0.955	0.977	0.966	0.851	0.499	0.992
People Total	0.861	0.991	0.926	0.772	0.422	0.938

(c) Metrics calculated in the first and second locations for the People class.

Table 6.1 Results for the Semantic Mapping experiments.

a balanced exploration style, with $F_w = 0.3$ but with optimization turned off ($C_w = 0$ and $O_w = 0$).

Table 6.2 reports the average explored areas for each set of experiments. In particular, the column A_f shows the mean and standard deviation, across 5 runs, of the area covered per floor, expressed as a percentage of the total explorable area E_f. As expected, increasing F_w makes the exploration strategy more "greedy": A_f decreases because the robot explores, on average, a smaller portion of each floor. The A_{tot} column shows the average area explored across the entire building, also expressed as a percentage, assuming four floors, each with explorable area E_f.

An analysis was performed to assess whether the observed differences among the four conditions were statistically significant in terms of the area explored in the 10-minutes alloted, A_{tot}. Given the small sample size, the statistical analysis was interpreted with caution and used mainly as exploratory support. The four conditions were compared using a non-parametric Kruskal–Wallis test. The test did not reveal a statistically significant difference among conditions ($H(3) = 4.90$, $p = 0.179$), with an effect size of $\varepsilon^2 = 0.119$.

F_w	Opt	A_f	A_{tot}
0	ON	$73.41\% \pm 23.07\%$	$44.05\% \pm 6.31\%$
0.3	ON	$70.39\% \pm 25.56\%$	$52.79\% \pm 8.25\%$
1	ON	$59.14\% \pm 25.49\%$	$47.31\% \pm 7.74\%$
0.3	OFF	$61.31\% \pm 28.85\%$	$39.85\% \pm 9.01\%$

Table 6.2 Area explored for the Exploration and Search set of experiments.

Table 6.3 reports the exploration results for each parameter configuration, including the number of floors visited by the robot (floors visited), the number of times the robot changed floors—requiring additional time to traverse the stairs (floors changed)—and the total exploration time, t_{expl}. It can be observed that both the number of floors visited, V_f, and the number of floor changes, C_f increase as the floor weight F_w increases. Similarly, the average exploration time t_{expl}, defined as the time spent exploring without including the time required to change floors, decreases as the robot changes floors more frequently, since the maximum allotted time of 10 minutes is spent more on moving up and down the stairs than on exploration.

F_w	Opt	V_f	C_f	t_{expl}
0	ON	12	16	462.8s $\pm$ 44.71s
0.3	ON	16	19	425.60s $\pm$ 29.89s
1	ON	16	21	408.2s $\pm$ 25.57s
0.3	OFF	13	14	492s $\pm$ 58.79s

Table 6.3 Floors visited, floors changed and actual time explored for the Exploration and Search set of experiments.

Table 6.4 reports the confusion matrices for each parameter configuration and the corresponding SAR-classification metrics to evaluate the performance of semantic mapping during autonomous exploration. The same table also reports the IoU and the IoU computed with inverted classes for comparison. It is important to note that FNs—people present in the ground truth map but not detected in the semantic map—are counted only for areas actually explored by the robot; individuals located in unexplored areas, such as other floors, were not included. This approach is justified because the extent of exploration is already captured by A_f and A_{tot}, while the classification metrics are intended to show that, even during autonomous navigation, the robot can visually inspect the environment and make accurate decisions about the presence of people.

The results indicate that recognition performance is largely consistent across the different configurations. The highest performance are obtained for configurations with $F_w = 0$, which prioritize fully covering a floor before moving to the next.

6.3.3 Simulated exploration

The corresponding metrics are reported in Table 6.5.

A one-way ANOVA was conducted to compare the total explored area, A_{tot}, across the six conditions defined by all combinations of $F_w \in \{0, 0.3, 1\}$ and $Opt \in \{ON, OFF\}$. The

	Predicted Positives	Predicted Negatives
Actual Positives	10	4
Actual Negatives	17	277

(a) Confusion matrix for the People class in the third location with $F_w = 0.3$ and $Opt = ON$.

	Predicted Positives	Predicted Negatives
Actual Positives	8	2
Actual Negatives	17	206

(b) Confusion matrix for the People class in the third location with $F_w = 0.3$ and $Opt = OFF$.

	Predicted Positives	Predicted Negatives
Actual Positives	9	2
Actual Negatives	13	233

(c) Confusion matrix for the People class in the third location with $F_w = 0$ and $Opt = ON$.

	Predicted Positives	Predicted Negatives
Actual Positives	10	4
Actual Negatives	19	241

(d) Confusion matrix for the People class in the third location with $F_w = 1$ and $Opt = ON$.

F_w	Opt	SAR-Sensitivity	SAR-Specificity	SAR-Bal. Acc.	SAR-MCC	IoU	IoU inv. class.
0	ON	0.818	0.947	0.883	0.554	0.178	0.994
0.3	ON	0.714	0.942	0.828	0.484	0.177	0.993
1	ON	0.714	0.927	0.821	0.459	0.123	0.994
0.3	OFF	0.8	0.924	0.862	0.474	0.145	0.992

(e) Metrics calculated for the People class in the third location for different values of F_w and Opt.

Table 6.4 Results for the Exploration and Search experiments.

analysis revealed a statistically significant difference among the conditions, $F(5,1014) = 37.45$, $p < .001$, $\eta^2 = .156$, indicating a large overall effect of the parameter configuration on A_{tot}. Descriptively, the configurations with $F_w = 0.3$ yielded higher values of A_{tot} than those with the other F_w values, both when optimization was enabled and when it was disabled. Bonferroni-adjusted Tukey–Kramer post-hoc comparisons, which are not reported in full for the sake of brevity, showed that almost all comparisons between configurations with $F_w = 0.3$ and configurations with other F_w values were statistically significant, with $p < .001$. The corresponding Hedges' g values ranged from 0.55 to 1.29, indicating moderate to large effects. The largest difference was observed between the condition with $F_w = 0.3$ and $Opt = ON$ ($M = 63.18$, $SD = 7.40$) and the condition with $F_w = 0$ and $Opt = OFF$ ($M = 52.82$, $SD = 8.55$), Hedges' $g = 1.29$. The only exception was the comparison between $(F_w = 0.3, Opt = OFF)$ and $(F_w = 1, Opt = ON)$, which was not statistically significant.

F_w	Opt	A_f	A_{tot}	V_f	C_f	t_{expl}
0	ON	87.24% ± 11.46%	55.05% ± 10.06%	436	471	448.29 s ± 33.65 s
0.3	ON	66.49% ± 9.67%	63.18% ± 7.40%	652	757	345.00 s ± 45.35 s
1	ON	60.91% ± 10.34%	58.78% ± 8.52%	661	853	313.13 s ± 55.69 s
0	OFF	86.60% ± 11.82%	52.82% ± 8.55%	423	436	459.78 s ± 41.89 s
0.3	OFF	63.03% ± 9.23%	60.33% ± 7.72%	655	769	337.76 s ± 44.22 s
1	OFF	56.33% ± 9.75%	54.46% ± 8.31%	662	915	290.87 s ± 52.16 s

Table 6.5 Results for the simulated Exploration and Search experiments.

When the data were pooled across the three F_w values, Welch's independent-samples t-test showed that A_{tot} was significantly higher with optimization enabled ($M = 59.00$, $SD = 9.32$) than with optimization disabled ($M = 55.87$, $SD = 8.80$), $t(1014.58) = 5.52$, $p < .001$, Hedges' $g = 0.35$. Thus, optimization produced a statistically significant, although relatively small, overall increase in the total explored area.

The analysis stratified by F_w, pooling the observations across the $Opt = ON$ and $Opt = OFF$ conditions, also revealed a statistically significant difference among the three F_w levels, $F(2,1017) = 72.37$, $p < .001$, $\eta^2 = .125$, corresponding to a moderate effect. Bonferroni-adjusted Tukey–Kramer post-hoc comparisons showed that A_{tot} was significantly higher for $F_w = 0.3$ ($M = 61.76$, $SD = 7.68$) than for $F_w = 0$ ($M = 53.94$, $SD = 9.39$), $p < .001$, with a large effect, Hedges' $g = 0.91$. Moreover, A_{tot} was significantly higher for $F_w = 0.3$ than for $F_w = 1$ ($M = 56.62$, $SD = 8.68$), $p < .001$, with a moderate effect, Hedges' $g = 0.63$. Finally, A_{tot} was significantly higher for $F_w = 1$ than for $F_w = 0$, $p < .001$, although the effect was small, Hedges' $g = 0.30$.

Finally note that t_{exp} is higher in configurations with $F_w = 0$ as they require a smaller number of floor changes, but this does not correspond to a higher explored area A_{tot}.

6.4 Discussion

This section analyzes and discusses the results in relation to the research questions presented earlier.

6.4.1 Semantic Mapping

In both locations, the number of FPs—instances incorrectly detected as people—is higher than expected. However, when compared with the number of TNs, this number is much less significant. It is important to note that the dataset is unbalanced, with the majority of

areas containing no detectable features. Correctly identifying empty areas is crucial in SAR applications, as it allows FRs to spend less time exploring areas already verified as empty by the robot (Remark 7). The metrics reported in Table 6.1c account for the strong class imbalance in the dataset and therefore provide a more accurate assessment of the system's performance. In particular, SAR-Specificity measures the proportion of correctly identified empty spaces (TNs) relative to instances where empty spaces are incorrectly classified as occupied (FPs). As such, it reflects the efficiency gained by FRs: higher SAR-Specificity corresponds to less time spent verifying false detections and more time saved by avoiding the re-exploration of areas already cleared by the robot.

Table 6.1c shows also that confusion matrices, and consequently the newly introduced SAR classification metrics, consistently capture two key aspects: the number of correctly detected people is higher than the number of undetected people, and the amount of correctly identified empty space is significantly higher than the amount of empty space incorrectly classified as containing people. By contrast, the IoU yields extremely low performance values, which do not accurately represent the robot's capability to search for victims, whereas the IoU computed with inverted classes is coherent with the expected results.

These results answer research question R1: by introducing a suitable metric tailored to the SAR context, semantic mapping is shown to represent the number and location of relevant features, as well as the extent of empty space, with a SAR-Balanced Accuracy higher than 0.9.

6.4.2 Exploration and Search

The exploration style is reflected in the area explored per floor, A_f, as expected. In Table 6.2, the highest A_f corresponds to the thorough configuration ($F_w = 0, Opt = ON$), where the robot attempts to fully explore one floor before moving to the next. The minimum average area per floor is observed in the "greedy" configuration ($F_w = 1, Opt = ON$). Conversely, the total area explored across the building reaches a maximum in the balanced trial ($F_w = 0.3, Opt = ON$). This outcome can be explained by two considerations. In the thorough trial ($F_w = 0, Opt = ON$), time is spent exploring small residual areas towards the end of each floor, a phenomenon known as the submodularity problem [26]. In the greedy trial ($F_w = 1, Opt = ON$), the robot discovers a large area immediately upon visiting a new floor, but a significant portion of the allotted time is spent climbing stairs. Interestingly, there is no linear relationship between actual exploration time and area explored. In Table 6.3, the exploration time is highest in the thorough configuration and lowest in the greedy case. In

the balanced configuration ($F_w = 0.3, Opt = ON$), the robot changes floors once the current floor is mostly explored, resulting in the most efficient strategy for maximizing area coverage within the 10-minute time limit.

Examining the metrics in Table 6.4e, the best-performing configuration is the thorough exploration strategy ($F_w = 0, Opt = ON$). This is likely because the robot spends more time on each floor and passes in front of each person more frequently, thereby increasing all performance metrics. The effect is particularly noticeable for SAR-Sensitivity, whereas SAR-Specificity shows only minor variation. This outcome is expected; however, in a SAR application, it may be reasonable to trade some accuracy for faster exploration, minimizing the time required to inspect all areas.

Each configuration exhibits some instances of FNs, which can be attributed to the difference in sensor fields of view. The occupancy grid map, which stores information about obstacles and free space, is generated and updated using data from the Velodyne LiDAR, which has a 360° field of view. In contrast, the ZED2 camera used for detecting people has a limited perceptual field. While the LiDAR's range was reduced to approximate the camera's field of view, it was maintained at 360° to ensure safe navigation and avoid potential collisions during SLAM. As a result, the robot may map areas where a person is located but never observe the person within the camera's view. One potential solution would be to filter LiDAR readings to precisely match the camera's field of view, preventing the mapping of areas detectable by the LiDAR but not visible to the camera, thereby improving coverage in the semantic map layers.

Similarly to the semantic mapping experiments, the number of FPs remains relatively high. In this set of experiments, the building was mostly empty, with only one person per floor, which makes the comparison with TPs less favorable. However, SAR-Specificity remains high, indicating that the robot correctly identifies empty areas most of the time. This ensures that FRs can save time by avoiding unnecessary exploration of areas already verified as empty by the robot, as highlighted in Remark 7. Additionally, the presence of FPs is not a major concern when developing a screening tool that can save lives: it is preferable to be occasionally overcautious and allow FRs to inspect an area that may contain an injured person, even if it later turns out not to, rather than failing to identify an area where a person is actually present.

As in the previous experiment, which focused only on feature detection, Table 6.4 shows that confusion matrices, and consequently the newly introduced SAR classification metrics, consistently capture the performance of the system, whereas the IoU returns extremely low values.

These results answer research question R2: the exploration strategy primarily affects the area explored, although statistical significance cannot be inferred due to the limited number of trials, rather than the system's ability to detect people and estimate the extent of empty space.

6.4.3 Simulated exploration

The simulation results were intended to confirm and further support the findings obtained in the real-robot experiments by assessing the environment-exploration strategy defined by Equations 4.1 and 4.4. However, they were not intended to fully reproduce the ROS-based robotic architecture implemented on Spot and therefore differ from the real-robot experiments in several implementation details. Likewise, the simulated environments are less challenging than real-world settings containing furniture and other objects, which likely affected the performance of the exploration strategies and parameter configurations. This applies, among other factors, to the influence of the optimization weights O_w and C_w, which are particularly important in the real-robot experiments to prevent the robot from continuously deviating from straight paths.

Despite these differences, the simulations produced results consistent with those obtained in the real-robot experiments. This supports the intuition that an appropriate trade-off must be found between thoroughly exploring each floor and repeatedly switching between floors within the allotted 10-minute period.

6.5 Conclusion

This work analyzes the feasibility of a system capable of autonomously exploring a multi-floor building while searching for people (and possibly other features that can be important in SAR scenarios).

Autonomous exploration results show that a faster and less exhaustive approach, in which the robot switches to the next floor before completing the current one, is advantageous. The results reveal clear differences in coverage across exploration strategies: while the thorough strategy achieves the highest per-floor coverage, the balanced strategy yields the largest overall explored area.

Regarding detection performance, the system was evaluated in three distinct environments: two under teleoperation and one during autonomous exploration. The newly introduced metric proves to better reflect the system's capability to detect people and empty areas than

traditional metrics. Although some variations in performance arise across locations, likely due to environmental differences and the placement of individuals intended to challenge the system, no significant degradation is observed during autonomous operation.

Part II

Extensions Toward Robust Autonomous Systems

Chapter 7

Related works

This chapter presents a review of the literature relevant to the second part of this thesis. It first examines approaches to robotic navigation in hazardous environments, focusing on terrain-aware navigation. Subsequently, it reviews navigation strategies designed to cope with uncertainty and unpredictability, with a particular focus on the use of emotion-inspired mechanisms. Finally, the chapter analyzes the application of symbolic planning in robotics and its role in enabling autonomous decision-making.

7.1 Robotic navigation

In any inspection task, the first fundamental requirement for a robot is the ability to move and navigate safely within its environment. Robotic navigation encompasses all aspects related to how a robot moves through space. This may include foot placement [127], path planning for safer traversal [202], and the ability to react to dynamic obstacles [153]. In the following sections, we review current trends in robotics related to navigation in harsh environments, with a focus on quadruped robots, where direct control of foot placement may not be available.

7.1.1 Terrain aware navigation

Numerous strategies have been developed for efficient path planning in 2D and 3D environments under various constraints [100, 111, 152, 183, 68, 122]. Additionally, to facilitate rapid prototyping of robotic systems, frameworks that aggregate different planning solutions have been introduced. One notable example, is the Open Motion Planning Library (OMPL)

[183]. OMPL is a comprehensive suite of tools for sampling-based motion planning [99], fully integrated with the ROS and extensively utilized in the robotic field.

The task of adapting to different types of terrains has been studied in the literature using several sensors and strategies. Some strategies are aimed at understanding the traversability and properties of the terrains. The authors of [136] propose a solution to plan the motion of a quadruped robot based on a terrain map that represents, for each cell, the cost of placing the foot of the robot in that position. This map is constructed locally using a depth camera and uses information about height deviation, slope, and curvature to obtain the final value for each cell.

In the work detailed in [184] the authors propose a strategy to segment images from a depth camera into background, road, and obstacles. Additionally, the authors propose a new way to estimate the ground plane, assuming that roads are mostly flat. By using this information, the environment is divided into traversable and non-traversable areas. This doesn't provide information on the type of terrain the robot either has to face or is facing.

The work proposed by Lin et al. [124] details a solution that allows the robot to move differently depending on the type of slope it is facing. The slope estimation is completed using the IMU and the position of feet. The robot can determine only the slope it is currently facing and not other significant characteristics of the ground. Another significant property of the ground plane, as stated in [27], is the ability of the ground to support the weight of the robot, which is the rigidity of the ground plane. Also, in this case, the robot is determining only the properties of the ground it is observing and not a classification useful for adapting to different terrains.

Different sensors can be used for the estimation of the type of terrain. In the works seen before, the authors propose strategies using IMUs, Lidars, and cameras to estimate the properties of the terrain. The work detailed in [86], proposes a solution to estimate the traversability of terrains based on the information received from a fisheye camera. The information from the model is used to generate a costmap representing the traversability of the terrain taking into account specific scenarios that the robot encounters, such as rails and stairs. Using Lidar and camera information, Cao et al. [32] propose a solution to classify the terrain as land, grass, curbstones, stairs, and background. Using camera and laser sensor fusion allows for a recognition of the terrain in front of the robot, not the terrain the robot is standing on.

One of the constraints that has to be acknowledged when navigating hazardous terrain is the slope the robot has to face. To navigate while considering slope constraints, the planner typically requires an elevation map, which is then used to calculate the terrain gradient.

Various types of multi-dimensional maps are utilized for path planning. Typically, 2.5D maps capture the terrain's height at each cell, offering less precision but greater computational efficiency [61]. In contrast, 3D maps represent the entire space around the robot but require more computational resources. The work by Miki et al. [137] introduces a method for efficiently computing 3D maps using GPUs, as well as a strategy for computing traversability near the robot and navigating rough terrains.

In the specific context of path planning on rough terrains, Chu et al. [38] propose an integrated system for mapping and navigation. This system employs a triangular-based grid map to depict the ground, incorporating both slopes and obstacles, if any. At each planning stage, the robot selects its next move from a set of finite possible actions to progress towards the goal while seeking a relatively flat path. Depending on the action selected, the system then devises a gait that safely executes the required movement. The primary limitations of this approach are that the navigation algorithm does not produce a complete and optimized path to the goal, nor does it fully verify the feasibility of the path.

Other approaches have been presented in [113] and [45], which propose algorithms for autonomous exploration of subterranean environments with legged robots. The path-planning algorithm relies on a local elevation map to minimize computational costs, while the global plan is created by disregarding terrain slope constraints and focusing solely on obstacle avoidance. As the robot navigates, it verifies against the local elevation map whether the slope constraints for traversability are satisfied, assuming areas not covered by the elevation map to be safe. However, if the robot encounters an area deemed non-traversable according to the local map, it recalculates the global path. These planning approaches produce a global path but do not ensure the safety and traversability of such a path, potentially leading the robot to expend time on routes that may not safely reach the goal.

In the work by Santos et al. [172], the planner computes the traversability of each cell based on whether the ground's normal in that cell indicates a safe inclination. The path to the goal is then calculated considering cell traversability. While this method ensures the creation of a safe path, its concept of traversability is somewhat limited as it only accounts for the maximum slope of the terrain and overlooks the gradient's projection in the robot's motion direction, an aspect our work addresses. Consequently, this approach fails to recognize that a cell might be traversable in one direction but not in another, thereby restricting pathfinding opportunities.

Given the significance of energy considerations in planetary exploration, [170] introduces a rover planning method that incorporates vehicle dynamics and energy management. Specifically, it proposes a cost function that accounts for path length, terrain roughness, roll and

pitch angles, and estimated power consumption and generation via solar panels. Other studies focus on the mechanical and control facets of dynamic quadrupedal locomotion across steep slopes. For instance, [106] introduces passive-adaptive planar feet and optimized grouser pads to decrease sinkage and enhance traction on both flat and inclined granular surfaces.

7.1.2 Emotion-Driven Navigation

Another aspect that has been deeply studied in the context of navigation is the presence of dynamic obstacles [215, 155, 54]. A solution to the inherent limitation of traditional planning systems is to use a mixed architecture with a slower deliberative layer that plans the global path and a reactive navigation layer whose objective is to react to unpredictability in the environment [215]. In biology, primates and humans use emotions to react to and interact with the environment [46].

While a notable body of work exists on the emotional regulation of robotic behavior, relatively few studies focus on emotional regulation in navigation. Emotions have been used in past works to modulate the behavioral responses of robots. Psychologists and neuroscientists study how emotions regulate the response of animals to the environment. Similarly, in robotics, emotions can modulate how robotic agents respond to the environment before deliberation. In complex environments, there is not always time to carefully deliberate; emotions can help agents react to the environment as a basis for behavior [9]. Since emotions are used by animals to increase their chances of survival, they can similarly be applied to robots to enhance their reactions to the environment and improve their chances of survival. In the review presented by Savery et al. [175], the authors analyze the use of emotions in robotic solutions. Out of 285 papers analyzed, only 55 propose using emotions to regulate robot behavior, while 230 use emotions solely to communicate the internal state to users.

Emotions have been used primarily to coordinate the behavior of robots with other agents, and also with people. One of the more notable works is that of Breazeal et al. [28], which uses internal drives and emotions to regulate the behavior of the Kismet robot. The robot's objective is to interact with people, and emotions are used to regulate its actions. Similarly, emotions have been used to regulate task allocation in multi-robot systems [13, 67]. In these works, emotions serve as mechanisms to allocate resources cooperatively. Emotions are employed both to increase the survival of individual agents and to improve the coordination of agents in achieving a shared goal. In these works, emotions have been used mainly to coordinate between agents and users, but they also have a strong survival function; emotions can directly regulate how a robot moves in the environment.

Some past works propose the use of emotions to improve the navigation capabilities of a robot. The work of Tsakanova et al. [189] aims at developing a system that modulates both the immune system and the emotional system. The article proposes the development of an artificial amygdala that generates a level of fear based on current stimuli and previous experiences. Fear changes the level of antibodies in the immune system, and in the case of a navigation problem, this alters parameters in the go-to behavior. The work proposed in [114] presents a system that regulates the behavior of a swarm of robots by using emotions both for self-regulation and to interact with other robots.

Similarly, the work presented in [82] aims at developing an algorithm for multi-robot control based on the emotional regulation of the agents. The system behaves according to a winner-takes-all policy on the active emotions. All the emotions depend both on external stimuli and on the robot's previous state; the emotions also persist when the stimulus is not present. This paper proposes a strategy in which emotions modify both the motor and perceptual schemas of the robot by adjusting parameters such as the field of view and the optimal speed.

The work of Lee-Johnson et al. [119] proposes a framework that regulates the motion of the robot based on its internal emotional state. The testing was done only in simulation, without comparisons to other state-of-the-art algorithms. Additionally, emotions modulate only the motor schema. The motion of the robot is influenced by the following emotions: Fear, Anger, Surprise, Happiness, and Sadness. Emotions depend both on the robot's internally modulated state and on external environmental stimuli. It is interesting to note that in this paper, the authors distinguish between reactive emotions, which influence only the internal state of the robot, and deliberative emotions, which are embedded in the map and used for path planning.

The aforementioned studies test the algorithm only in simulation, making it difficult to draw conclusions about the robot's performance in the presence of unpredictability. Additionally, in very few studies do emotions influence both motor parameters and the perception of obstacles [82].

7.2 Symbolic planning in robotics

To coordinate the various tasks of an inspection application, one possible approach is the use of symbolic planning, which enables the generation of long action sequences while also allowing the system to respond to environmental uncertainty. There are several types of

planning frameworks, each suited to different problem domains and objectives. Some of the most widely used frameworks [87, 15, 36] are based on PDDL.

PDDL and its subsequent versions [79, 64, 77] are standardized and widely used planning languages in AI. PDDL separates the planning problem into two components: the domain and the problem. The domain file describes relevant objects, predicates that define properties associated with those objects, and actions characterized by preconditions that must be satisfied before execution and effects that describe the outcomes of the actions. The problem file specifies the initial state (the objects and their properties as defined by predicates) and the goal state (the predicates that must be true upon task completion). As an evolution of the earliest planning language STRIPS (Stanford Research Institute Problem Solver [73]), PDDL supports more complex planning features, such as concatenated goals, conjunctive and disjunctive actions, and advanced temporal and numeric reasoning.

PDDL supports various solvers, each suited to specific domains and problem requirements. Fast Forward (FF) [88] uses forward search with heuristics (e.g., relaxed planning graph) for fast, efficient planning, but it may produce suboptimal plans as it prioritizes speed over quality. Local Search for Planning Graphs (LPG) [78] applies local search to iteratively refine plans, handling numerical and temporal constraints, making it useful for resource and timing-critical tasks. COLIN [41]) addresses continuous and linear numeric changes but can face computational challenges in large-scale scenarios. METRIC-FF [87] extends FF with metric planning capabilities, supporting numerical and temporal constraints for resource optimization. OPTIC [15] generates optimal temporal plans, managing time and resources effectively, though it performs better in highly time-constrained contexts. POPF [36] employs partial order planning for flexible action sequencing, accommodating temporal and numeric constraints while maintaining efficiency through heuristic searches.

More complex formalisms exist, such as Linear Temporal Logic (LTL) used for specifying and reasoning about temporal properties of systems [108]. Extending classical propositional logic, LTL introduces temporal operators such as ‘next’, ‘globally’, ‘finally’, and ‘until’, enabling the expression of how a system’s state evolves over time. LTL has been used in partially known environments to control a multi-robot team [71]. However, defining goals in LTL involves reasoning about system behavior over time rather than directly specifying end states, which can be less intuitive for some applications. Computation Tree Logic (CTL) [44], like LTL, is a temporal logic used to define system properties. CTL differs from LTL by allowing reasoning about multiple possible future paths from any given state, making it particularly useful for systems represented by state-transition graphs. CTL was used to

improve a multi-robot delivery system by veryfing the behavior of the entire system at each time instance [97].

Hierarchical Task Network (HTN) Planning [76] decomposes complex tasks into smaller, manageable subtasks. HTN is effective in domains where hierarchical decomposition is natural but less flexible and harder to generalize across different domains. As a procedural approach, HTN focuses on how tasks are planned and relies on predefined methods, limiting its adaptability in dynamic environments requiring frequent replanning. HTN has been used in social robots to extract human intent, in particular, HTN has been used to represent the person's mental state [92].

Task and Motion Planning (TAMP) [72] integrates task planning and motion planning to address problems involving high-level goals and low-level actions. TAMP is particularly well-suited for manipulator robots that require specific trajectory execution [25]. However, its complexity and computational demands arise from the detailed knowledge required for both tasks and motion planning. Furthermore, the tight coupling between tasks and motions can reduce modularity, making TAMP less suitable for situations that prioritize high-level goals over specific motion details.

Some of the most recent approaches to robotic planning introduce the use of Large Language Models (LLMs) to integrate natural language instructions into planning systems [118]. In this work, LLMs are leveraged to decompose a task and generate an appropriate sequence of actions to complete it. Similarly, Visual Language Action models (VLAs) [174] follow an approach inspired by LLMs and Vision-Language Models (VLMs) [123]. These approaches are based on the idea of avoiding the use of explicit planning languages altogether, treating task and motion planning as a generative process trained on large amounts of data. However, while LLMs are promising for understanding natural language, and VLAs show potential for motion planning, they still lack the advanced capabilities of symbolic planners to handle long, complex plans in vast domains and problem spaces, as required by SAR scenarios.

While HTN focuses on task decomposition, PDDL with the POPF solver provides a more general framework that does not rely on predefined hierarchies and allows for replanning. Unlike TAMP, PDDL does not require direct integration with motion planning, which is unnecessary for the scope of this project. For the same reasons, PDDL is a better choice than CTL and LTL, as it is specifically designed for planning rather than for reasoning about temporal properties or model checking. With respect to VLAs, PDDL is capable of addressing the complex planning problems required by SAR scenarios. Finally, PDDL is the

standard in robotic planning, offering widespread support, integration with ROS and ROS2 [135], and benefiting from a large and active community.

7.2.1 Planning for quadrupeds in SAR scenarios

Due to the flexibility in describing the tasks a robot can perform, planning frameworks have been used to control complex systems in demanding environments. In this section, we aim at reviewing works about planning for quadruped robot in the SAR scenario.

To the best of the authors' knowledge, only a limited number of studies have attempted to implement planning systems for robots in SAR scenarios. For example, Kirsch et al. [105], claim to target improvements in SAR operations; however, their system is evaluated only on a blocks world problem using the Spot robot from Boston Dynamics. The domain is therefore not specific to SAR, as it implements a blocks manipulation task. Another work proposes the SAR scenario without the use of quadruped robots; Roberts et al. [160] propose a Goal Lifecycle Network to plan robot actions. Although the system is evaluated in a SAR scenario, the actions relevant to this domain are not described in detail. The SAR setting is primarily used to demonstrate the system rather than being the main focus of the study.

Other relevant studies propose planning systems to coordinate the operations of heterogeneous robots, including quadrupeds, in different scenarios. For example, Zhou et al. [217] propose an HTN system to coordinate the operations of both quadruped and wheeled robots in a hospital environment. In particular, their system monitors and responds to situations in which a robot, due to external disturbances, cannot complete its assigned task. Similarly, [49] implements a Behavior Tree architecture to manage fallbacks in navigation tasks. Finally, planning systems for mobile robots, in particular quadrupeds, are used in [161] to coordinate multiple robots collaborating to complete a task; however, no specific reference to SAR scenarios is provided.

7.3 Conclusion

This chapter reviewed the state of the art in terrain-aware navigation, adaptive robot behavior, and symbolic planning for autonomous robotic systems. While significant progress has been made in each of these areas, several limitations remain that hinder the deployment of robots in complex and unpredictable search-and-rescue environments.

The review of terrain-aware navigation approaches showed that existing methods typically focus on specific aspects of the problem, such as traversability estimation, terrain

characterization, or slope-constrained path planning. However, these approaches generally lack an integrated framework that combines terrain classification with direction-dependent traversability analysis, limiting the robot's ability to adapt its navigation strategy while ensuring safe and feasible motion across diverse terrain conditions.

Similarly, approaches that exploit emotion-inspired mechanisms to cope with dynamic obstacles and environmental uncertainty have demonstrated promising results. Nevertheless, most studies have been evaluated only in simulation, and the majority influence either the perceptual or the motor components of the robotic architecture, rather than both. As a result, their effectiveness in realistic operational scenarios remains largely unexplored.

Finally, the analysis of symbolic planning approaches revealed a lack of planning domains specifically designed for SAR operations. Furthermore, the capabilities and limitations of existing domains have not been systematically evaluated in the context of SAR tasks.

The research presented in the following chapters addresses these gaps by investigating navigation and planning strategies tailored to the requirements of autonomous SAR operations.

Chapter 8

Slope-Aware Navigation Strategies

Seeing the robot operate with a certain degree of autonomy, as presented in 6, where the robot is tasked with autonomously exploring an unknown environment, other areas of improvement emerged. In particular, it is essential to design methods capable of determining suitable paths while accounting for motion-related constraints, such as maximum climbable slope, terrain slipperiness, obstacle height limits, and similar factors [204].

The situation is further complicated by the nature of commercially available robots. Unlike research platforms, many systems intended for practical deployment rely on closed architectures, limiting direct access to low-level locomotion control.

Given these limitations, one of the main controllable aspects at the software level is the trajectory itself. For this reason, the objective is to design a path planning approach that explicitly considers the difficulties associated with traversing inclined surfaces, both ascending and descending.



Figure 8.1 Left: Climbing rocks at the Civil Protection camp in Foligno. Right: Carrying 6 bottles of water (9 kg) uphill in the DIBRIS Department's garden.

This direction is motivated by two main factors. The first is safety: it is well known that quadruped robots may experience reduced stability when moving over steep or uneven terrain (Figure 8.1, left), a problem that becomes more pronounced under load (Figure 8.1, right). The second factor, though less frequently emphasized, is energy efficiency [170]. Humans intuitively account for both effort and time when navigating slopes, typically balancing two elements: total path length and the effective gradient along the direction of travel.

By concentrating on high-level planning strategies, it becomes possible to develop slope-aware navigation methods that remain applicable across different robot platforms, regardless of their physical properties or whether their locomotion systems are open or closed.

The objective is to propose a path-planning algorithm capable of identifying a sequence of waypoints that satisfies a set of constraints. The robot can subsequently use a state-of-the-art path-following strategy to avoid obstacles and safely navigate to the goal. This chapter first describes the theoretical implementation of the system, followed by the experimental procedure. The results are then discussed in terms of path length and the robot's body positioning.

8.1 Methodology

To generate a path from the start $(x_{start}, y_{start}, \psi_{start})$, to the goal $(x_{goal}, y_{goal}, \psi_{goal})$, the core idea is to choose a set of state frames sampled randomly from the elevation map. These sampled state frames function as nodes in a roadmap, which are interconnected by edges. Both nodes and edges are chosen to adhere to the following constraints:

- at each node, the Pitch angle, ϕ_r must remain below predefined thresholds;
- at each node, the Roll angle, θ_r must remain below predefined thresholds;
- along each edge, the maximum slopes must not exceed predefined thresholds;
- along each edge, the angular rotations must not exceed predefined thresholds.

Once the roadmap has been constructed, an optimal trajectory is computed using a state-of-the-art planning algorithm [183], with the objective of minimizing the overall path length.

The three subtask that comprise this approach are detailed in the following subsections: Section 8.1.1 details the state sampling and validity checking to identify the nodes of the roadmap, Section 8.1.2 details the validation of edges of the roadmap, while Section 8.1.3 details the task of planning a path given the roadmap.

Below is a list of the required terms, Figure 8.2.

- $\langle w \rangle$: World reference frame.
- $\langle b \rangle$: Robot's base frame.
- $\langle s_p \rangle$, $p = 1, \ldots, P$: Randomly sampled state frames.
- ${}^{w}n_p$: Normal vector of the map surface in the vicinity of frame $\langle s_p \rangle$, expressed with respect to $\langle w \rangle$.
- x_p, y_p, z_p: Position describing the origin of a state frame $\langle s_p \rangle$ with respect to $\langle w \rangle$.
- ψ_p, ϕ_p, θ_p: Heading angle, longitudinal inclination angle, and lateral inclination angle, describing the orientation of the state frame $\langle s_p \rangle$ with respect to $\langle w \rangle$.
- x_r, y_r, z_r: Position describing the origin of the robot's base frame $\langle b \rangle$ with respect to $\langle w \rangle$.
- ψ_r, ϕ_r, θ_r: Heading angle, longitudinal inclination angle, and lateral inclination angle, describing the orientation of the base frame $\langle b \rangle$ with respect to $\langle w \rangle$.
- Φ and Θ: Imposed thresholds on the longitudinal and lateral inclination angles, respectively.
- L_{pq}, α_{pq}, $\Delta\psi_{pq}$: the Euclidean distance between two state frames $\langle s_p \rangle$ and $\langle s_q \rangle$, the slope of the straight line connecting them, the difference between the corresponding heading angles;
- L_{min}, L_{max}, $Alpha$, $Delta$: thresholds for L, α_{pq} and $\Delta\psi_{pq}$ between two waypoints.

Remark 8. *The main difference from other approaches is that the planning is optimized only on the node distance. The other constraints are checked and ensured in a previous phase to validate nodes and connections. Specifically, rather than assigning an infinite cost to path segments that fail to meet constraints during planning as in previous works [170], we eliminate infeasible nodes early in the sampling phase. In other words, the state frames $\langle s_p \rangle$ are not determined by randomly generating a triplet (x_p, y_p, ψ_p). Instead, we generate only a couple (x_p, y_p) and then search for two frames $\langle s_p \rangle$ and $\langle s_{p+1} \rangle$ with opposite angles ψ_p and $\psi_p + \pi$, such that they comply with the slope constraints for planning (which effectively reduces the sampling space from 3D to 2D). Similarly, edges will only be considered during planning if they satisfy constraints related to node distance, slope, and the maximum rotation allowed between adjacent nodes.*

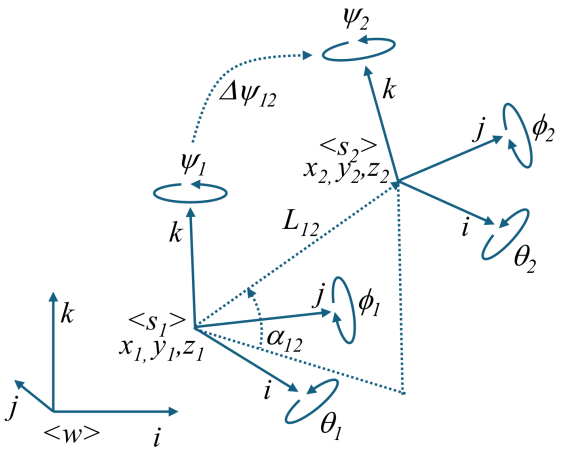


Figure 8.2 World frame $\langle w \rangle$, state frames $\langle s_1 \rangle$ and $\langle s_2 \rangle$, and related quantities.

8.1.1 State sampling and validity checking

The generation of a valid state frame $\langle s_p \rangle$ begins by randomly sampling the planar coordinates x_p and y_p within the state space, while the corresponding height z_p is obtained from the elevation map. To account for orientation, an intermediate frame $\langle s'_p \rangle$ is first defined. This frame shares the same position as $\langle s_p \rangle$, is assigned a randomly chosen heading angle ψ_p, and assumes zero roll and pitch, i.e., $\phi_p = \theta_p = 0$. The associated rotation matrix between the world frame $\langle w \rangle$ and $\langle s'_p \rangle$ is then derived using Equation 8.1, where $[\hat{k}]$ is the skew-symmetric matrix of the vector $\hat{k}$.

$$ {}^{w}_{s'_p}R = e^{[\hat{k}]\psi_p}; \hat{k} = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}, \tag{8.1} $$

In the next step, the surface normal of the terrain around the sampled frame $\langle s'_p \rangle$ is extracted from the elevation map and expressed in the world frame $\langle w \rangle$, denoted as ${}^{w}n_p$. This vector is then transformed into the local frame $\langle s'_p \rangle$, yielding ${}^{s'_p}n$, as defined in Eq. 8.2.

$$ {}^{s'_p}n = {}^{w}_{s'_p}R^{-1\,w}n_p. \tag{8.2} $$

Let us now return to the state frame $\langle s_p \rangle$, where the assumption $\phi_p = \theta_p = 0$ is no longer imposed. The pitch and roll angles, indicated as ϕ_p and θ_p, are obtained according to Equation 8.3. For clarity, the normal vector expressed in $\langle s'_p \rangle$ is written in component form as ${}^{s'_p}n = \begin{bmatrix} n_1 & n_2 & n_3 \end{bmatrix}^{\mathsf{T}}$.

$$\phi_p = tan^{-1}(n_1/n_3), \quad \theta_p = -tan^{-1}(n_2/n_3). \tag{8.3}$$

Ultimately, the rotation matrix between $\langle s_p \rangle$ and $\langle w \rangle$ can be determined using Equation 8.4, where $[\hat{i}]$ and $[\hat{j}]$ the skew-symmetric matrices of $\hat{i}$ and $\hat{j}$.

$${}^{w}_{s_p}R = e^{[\hat{i}]\theta_p} e^{[\hat{j}]\phi_p} e^{[\hat{k}]\psi_p}; \hat{i} = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}, \hat{j} = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}, \tag{8.4}$$

The path constraints include limits on the admissible absolute values of the robot's pitch and roll angles. To satisfy the prescribed bounds Φ and Θ, the frame $\langle s'_p \rangle$ may be iteratively rotated about its k-axis, with the corresponding frame $\langle s_p \rangle$ updated each time, until the constraints are fulfilled. If no suitable value of ψ_p can be identified, a new position is sampled within the map.

After a feasible state $\langle s_p \rangle$ has been obtained, an additional frame $\langle s_{p+1} \rangle$ can be defined at the same location with a heading offset of $\psi_{p+1} = \psi_p + \pi$. This configuration is likewise considered valid and included in the roadmap. During the planning phase, preference is given to the state whose orientation requires the smallest rotational adjustment from the robot. This enables the robot to exploit backwards motions when transitioning between successive waypoints. A summary of this procedure is provided in Algorithm 2.

8.1.2 Motion validity checking

After validating the nodes, the next step in constructing the roadmap is to establish connections between them through edges. As with nodes, these edges must also be checked for feasibility, as detailed in Algorithm 3. Figure 8.2 illustrates a transition between two sampled states $\langle s_p \rangle$ and $\langle s_q \rangle$, whose positions in the world frame $\langle w \rangle$ are given by (x_p, y_p, z_p) and (x_q, y_q, z_q). In the figure, L_{pq} denotes the length of the motion, ψ_p and ψ_q represent the corresponding heading angles, and α_{pq} indicates the slope of the motion, as defined in Equation 8.5.

$$\alpha_{pq} = \tan^{-1}\left((z_p - z_q) / \sqrt{(x_p - x_q)^2 + (y_p - y_q)^2} \right) \tag{8.5}$$

The constraints for an edge, connecting $\langle s_p \rangle$ and $\langle s_q \rangle$, to be considered valid, are as follows.

Algorithm 2: *stateValidityCheck*

input : Position x_p, y_p
parameter : Elevation map m, thresholds Phi, $Theta$
output : $valid$, $\langle s_p \rangle$, $\langle s_{p+1} \rangle$

$z_p \leftarrow getHeight(x_p, y_p, m)$;
$\psi_p \leftarrow random(0, \pi)$;
$\langle s'_p \rangle \leftarrow frame(x_p, y_p, z_p, \psi_p, 0, 0)$;
$valid \leftarrow false$;
while $\neg valid \wedge \neg allValuesChecked(\psi_p)$ **do**
  ${}^{w}n_p \leftarrow getNormal(\langle s'_p \rangle, m)$;
  $\phi_p \leftarrow computePhi({}^{w}n_p)$;
  $\theta_p \leftarrow computeTheta({}^{w}n_p)$;
  if $validAngles(\phi_p, \theta_p, Phi, Theta)$ **then**
    $valid \leftarrow true$;
    $\langle s_p \rangle \leftarrow frame(x_p, y_p, z_p, \psi_p, \phi_p, \theta_p)$;
    $\langle s_{p+1} \rangle \leftarrow frame(x_p, y_p, z_p, \psi_p + \pi, \phi_p, \theta_p)$;
  else
    $\psi_p \leftarrow rotate(\psi_p)$;
  end
end

- The Euclidean distance L_{pq} between two states connected by an admissible transition must lie within the bounds defined by L_{min} and L_{max}.
- The inclination α_{pq} associated with a feasible motion is required to remain below a prescribed limit $Alpha$.
- The angular variation $\Delta\psi_{pq}$ between the heading angles ψ_p and ψ_q of the connected states must not exceed a given threshold $Delta$.

Remark 9. *In general, connectivity is determined by whether the path between two nodes is free of obstacles. In this work, however, obstacle avoidance is delegated to the onboard system, assuming that local planning and path-following modules can effectively handle such situations during execution.*

Remark 10. *The yaw variation $\Delta\psi_{pq}$ is taken into account only when the slope α_{pq} exceeds a predefined threshold. Under this condition, since the robot is considered on steep terrain, a limit is imposed on the allowable rotation during transitions between states. Consequently, given that each state frame $\langle s_p \rangle$ is associated with a corresponding frame $\langle s_{p+1} \rangle$ such that $\psi_{p+1} = \psi_p + \pi$, the planner favors sequences of waypoints that reduce the need for rotational maneuvers, leveraging the robot's ability to move both forward and backward.*

Algorithm 3: *roadmapConstructor*

parameter : Thresholds L_{min}, L_{max}, $Alpha$, $Delta$
output : Set of nodes N, set of edges E

$N \leftarrow \emptyset$, $E \leftarrow \emptyset$;
while $needMore(N)$ **do**
  $valid = false$;
  while $\neg valid$ **do**
    $x_p, y_p \leftarrow stateSample()$;
    $valid$, $\langle s_p \rangle$, $\langle s_{p+1} \rangle \leftarrow stateValidityCheck(x_p, y_p)$;
  end
  $N \leftarrow N \cup \{\langle s_p \rangle, \langle s_{p+1} \rangle\}$;
end
for $\langle s_p \rangle \in N$ **do**
  $N_c \leftarrow nearestNeighbors(N, \langle s_p \rangle)$;
  for $\langle s_q \rangle \in N_c$ **do**
    $L_{pq} \leftarrow computeLength(\langle s_p \rangle, \langle s_q \rangle)$;
    $\alpha_{pq} \leftarrow computeSlope(\langle s_p \rangle, \langle s_q \rangle)$;
    $\Delta\psi_{pq} \leftarrow computeAngleDiff(\langle s_p \rangle, \langle s_q \rangle)$;
    if $validateMotion(L_{pq}, \alpha_{pq}, \Delta\psi_{pq}, L_{min}, L_{max}, Alpha, Delta)$ **then**
      $E \leftarrow E \cup \{(\langle s_p \rangle, \langle s_q \rangle)\}$;
    end
  end
end

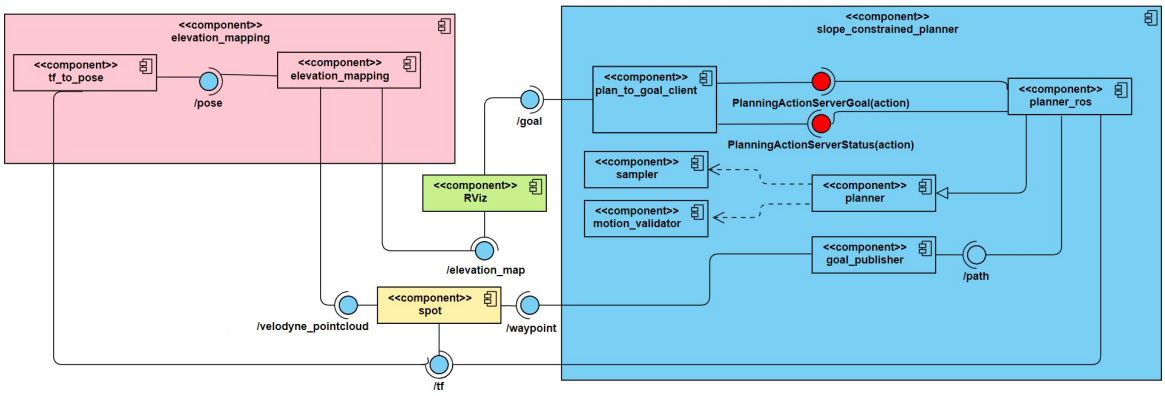


Figure 8.3 System architecture implemented through ROS nodes (blocks) and topics (connectors).

8.1.3 Finding a path

For the path planning stage, we make use of the OMPL [183]. OMPL includes a broad set of planners that, to different degrees, correspond to the three-step process described earlier: sampling and validating states, verifying the feasibility of transitions, and computing an optimal path over a graph of valid configurations and connections.

In this work, we adopt the LazyPRM* algorithm [24] because of its increased efficiency in graph search. However, we adapt it by omitting the validation stage within the planner itself, under the assumption that all nodes and edges provided in the input roadmap have already been verified as feasible.

8.1.4 System architecture

The ROS architecture shown in Figure 8.3 integrates the main ideas introduced in the previous sections.

The red module, `elevation_mapping`, groups the components responsible for constructing and maintaining the environmental elevation representation. It builds a height map from an incoming PointCloud stream. The robot used in this work (Spot) provides onboard computing resources for running custom software, although it lacks a dedicated GPU. For this reason, a lightweight 2.5D representation was selected [61]. This design choice introduces certain limitations, in particular the fact that the extent of the elevation map is constrained by CPU performance, which consequently limits the maximum navigable distance to the target.

The blue module, `slope_constrained_planner`, corresponds to the motion planning subsystem and includes several internal elements not described in detail in this paper. The overall design is inspired by OMPL [183] and, with respect to planner structure, also by ArtPlanner [203]. Two main submodules are exposed: `sampler` and `motion_validator`, which have already been introduced in Sections 8.1.1 and 8.1.2, respectively.

Remark 11. *The components responsible for local path planning and trajectory following, which guide the robot through successive waypoints from the initial position to the goal, differ between the real Spot platform and its simulated counterpart (in Figure 8.3, these functions are implicitly contained within the* `spot` *module). On the physical robot, the sequence of waypoints is provided directly to the onboard navigation system developed by Boston Dynamics, which leverages five body-mounted depth cameras to perceive the environment and perform local navigation.*

In simulation, waypoint tracking is instead handled by a simplified PID-based controller that we implemented, which generates velocity commands for the robot. The simulator [156] subsequently translates these commands into the corresponding locomotion patterns.

8.2 Experiments

To assess the proposed approach, experiments were carried out both in simulation and on the Spot robot from Boston Dynamics. In both scenarios, the focus was on analyzing the generated trajectories and the influence of slope-related constraints on the final paths. For each run, the following data were recorded: (i) the planned output expressed as a sequence of waypoints, (ii) the actual values of ϕ_r and θ_r measured during robot motion from the initial to the goal configuration, and (iii) any instances where plan execution was unsuccessful.

8.2.1 Simulated experiments

The simulations were executed on an ASUS laptop featuring an Intel Core i7-6700HQ processor, an NVIDIA GeForce GTX 960M graphics card, and 16 GB of RAM. The simulation framework was based on the 'wolf' package [156], which provides the robot model, control pipeline, and a virtual environment.

To evaluate performance under different terrain inclinations, the environment was manually partitioned into regions with varying slopes and classified into 'high-slope' and 'low-slope' areas according to the magnitude of the gradient. In each trial, the robot was initialized at a random location within one of these regions, while the goal position was set 4 meters away. Each run was further labeled as either 'ascending' or 'descending' depending on whether the goal was higher or lower in elevation than the starting position. This resulted in four experimental conditions: 'high slope moving up,' 'high slope moving down,' 'low slope moving up,' and 'low slope moving down'. For each of the four experimental conditions, we selected 3 different sets of starting configuration and goal configuration. To evaluate the behavior of the system based on Φ and Θ thresholds, for each start-goal configuration, we repeated the experiments 8 times: 4 experiments without setting a threshold Θ for θ_r and changing the threshold Φ for ϕ_r in the set $\{off, 30^\circ, 20^\circ, 10^\circ\}$; 4 experiments by setting a fixed threshold of 20° on θ_r and letting the threshold on ϕ_r varying within the same set as above. In total, we performed $4 \times 3 \times 8 = 96$ experimental runs.

8.2.2 Real-world experiments

The proposed method was validated on the Spot platform developed by Boston Dynamics. To improve the quality of the elevation mapping, the robot was equipped with a LiDAR sensor, and a Spot Core unit was integrated to provide additional onboard computation. This module includes an Intel® Core™ i5 8th Gen (Whiskey Lake-U) processor and 16 GB of RAM.

The experimental trials were carried out in the parking area of the DIBRIS department at the University of Genoa, where two regions characterized by noticeably different average slopes were selected. In the steeper region, each experiment was repeated twice, considering two different map sizes—one smaller and one larger.

We designated three sets of starting and goal positions, each 4meters apart, to reduce computational complexity. After reaching the target, the roles of start and goal were swapped, enabling the collection of performance data for both ascending and descending motions. Each configuration was tested using five different values of the Φ threshold: *off*, 20°, 10°, 8°, 5°, while the Θ constraint was either deactivated ($\Theta = \mathit{off}$) or fixed at $\Theta = 20°$.

Overall, this resulted in $3 \times 2 \times 5 = 60$ trials (62 in total, accounting for two repeated runs). The robot was unable to complete the task in two cases: one due to the planner failing to produce a feasible path, and another caused by communication problems with the LiDAR sensor, which led to a halt in operation. In contrast to the simulation results, no falls were observed during the real-world experiments.

8.3 Results

8.3.1 Simulated Results

Three fails were recorded. One occurred when the planner was unable to identify a feasible path within the allotted computation time. The remaining two were caused by the lateral tipping of the robot during motion. In both of these cases, the roll constraint on θ_r was not enforced ($\Phi = 10°$, $\Theta = \mathit{off}$), which allowed the robot to reach unsafe roll configurations, as illustrated in Figure 8.4.

Figure 8.5 presents example trajectories obtained in two experimental runs, showing the sequence of state frames associated with the waypoints. In Figure 8.5a, the robot does not follow a direct line toward the goal; instead, the trajectory exhibits a zig-zag pattern. This behavior suggests that the robot moves approximately along directions tangent to the isoclines in order to keep the value of ϕ_r within acceptable limits.

Figure 8.5b highlights a different situation, where the robot deviates considerably from the straight-line path when such a route is not feasible. This detour can again be interpreted as a strategy to reduce the component of the terrain slope along the direction of motion.

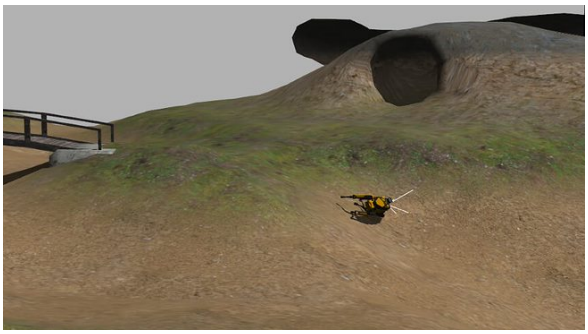
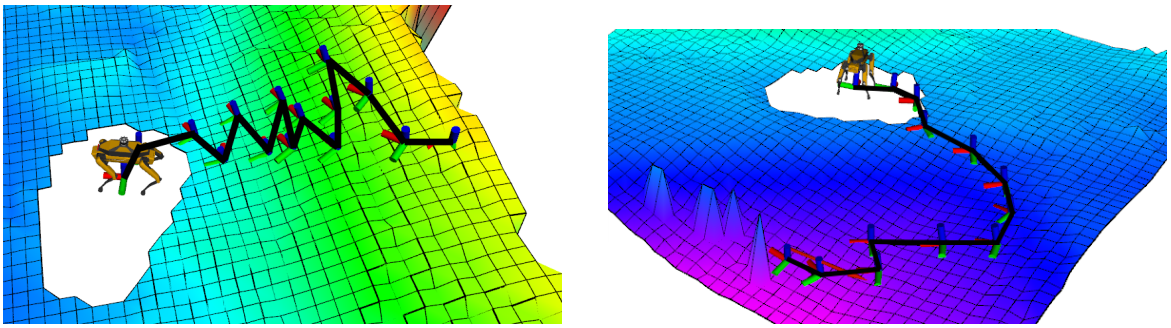


Figure 8.4 Falling robot with $Phi = 10^\circ$ and $Theta = off$ in simulation.

Figure 8.6a shows a box plot illustrating the ratio between the total path length—computed as the sum of Euclidean distances between successive waypoints—and the straight-line Euclidean distance from the start to the goal. The results are grouped according to the values of Φ and Θ. It can be observed that configurations with a smaller value of $\Phi = 10^\circ$ tend to produce larger ratios, reflecting longer trajectories toward the goal. In contrast, little variation is noticeable between cases where $\Theta = 20^\circ$ and those where no constraint is imposed ($\Theta = off$). This behavior is likely due to the terrain characteristics, as the slopes encountered by the robot are generally below 20° across most of the environment.

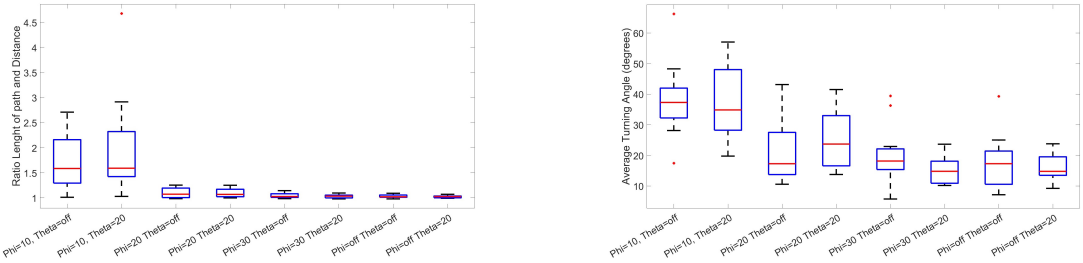


(a) $\Phi = 10^\circ$, $\Theta = off$ (b) $\Phi = 10^\circ$, $\Theta = 20^\circ$

Figure 8.5 Comparison of planned paths in simulation under different slope constraints.

The average turning angle is computed by examining triplets of consecutive waypoints: for each group of three points, the angle defined by the connecting segments is evaluated, and the final value is obtained by averaging these angles over the entire path.

The pattern observed for the average turning angle in Figure 8.6b is consistent with the behavior of the path length ratio shown in Figure 8.6a, suggesting that larger values of Φ



(a) Path length ratio vs. Euclidean distance.

(b) Average turning angle of the path.

Figure 8.6 Simulation results under different values of Φ and Θ.

and Θ generally produce smoother, less winding trajectories. However, this effect appears less evident in Figure 8.6b than in Figure 8.6a. In particular, when the robot reaches the goal by bypassing steep regions—favoring a detour strategy (Figure 8.5b) instead of a zig-zag motion (Figure 8.5a)—the resulting average turning angle may remain relatively low even when smaller thresholds Φ and Θ are used.

8.3.2 Real-world results

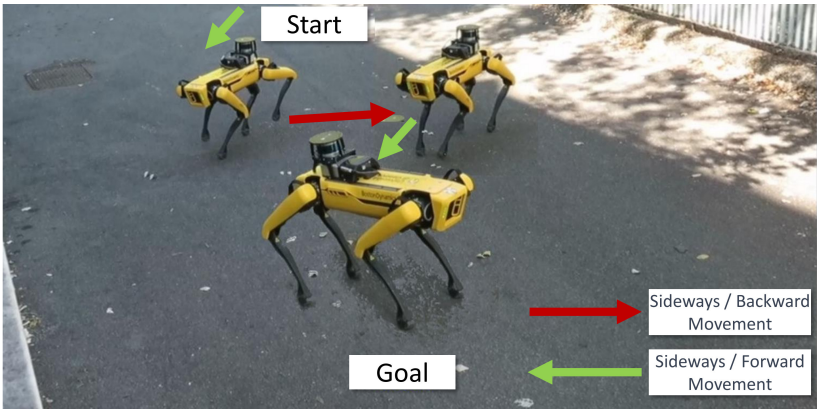
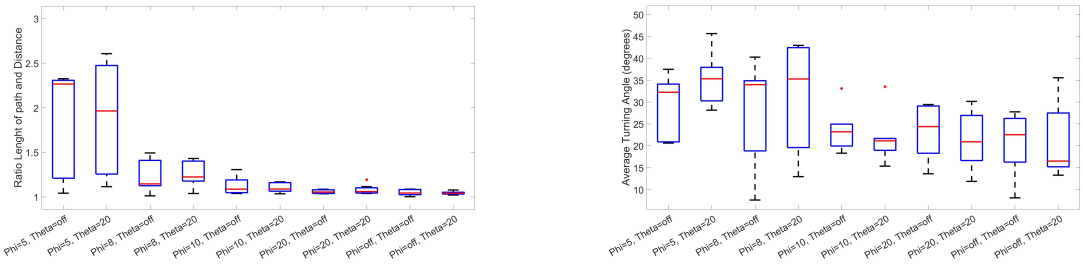


Figure 8.7 Snapshots of Spot following a path uphill in different time instants.

Figure 8.7 shows a sequence of images capturing the robot as it executes the planned trajectory during a real-world trial on the steeper terrain, with $\Phi = 5°$ and $\Theta = 20°$.

Similarly to what was done to evaluate for the simulation results Figure 8.8a illustrates the ratio of the path length to the Euclidean distance from the start to the goal. Consistent with expectations, lower Φ thresholds result in higher path-to-distance ratios. As can be seen in 8.7, the robot adopts a zigzag trajectory, moving back and forth, to avoid advancing directly toward the steepest gradient.

Figure 8.8b reports the average turning angle. The largest values are associated with the most restrictive thresholds ($\Phi = 5°$ and $\Theta = 20°$). A notable variability is also visible for



(a) Path length ratio vs. Euclidean distance.

(b) Average turning angle of the path.

Figure 8.8 Experimental results on the Spot robot under different Φ and Θ thresholds.

$\Phi = 8°$, which can be explained by cases where the robot is able to follow a nearly straight trajectory toward the goal. For larger values of Φ, the differences tend to diminish, as they are limited by the actual terrain inclinations encountered during the experiments.

Figure 8.9 shows a box plot summarizing the distribution of the ϕ_r angle over all trials and along the entire trajectories. A clear distinction emerges for the case with $\Phi = 5°$, where the measured ϕ_r occasionally exceeds the imposed limit, whereas in experiments with larger Φ values such violations occur much less frequently.

An analogous plot for θ_r is not included, since the terrain inclination encountered during the experiments consistently stays below the threshold $\Theta = 20°$, and therefore no exceedances were observed.

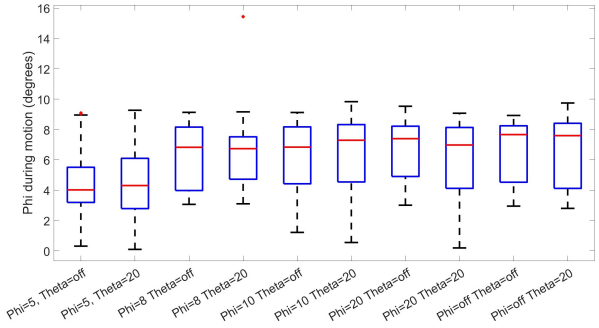


Figure 8.9 Statistics about ϕ_r values over the motion of the robot.

8.4 Discussion

The experimental evaluation in a simulation enviroment, with the possibility to test more aggressive slope, demonstrated the feasibility of the approach. More importantly, it showed that in certain situations, failure to consider the slope the robot has to face, may results in critical failures. Similarly, the observed differences were observed even in the real-world

experiments. Some emerging strategies emerge from the robot navigating uneven terrains such as moving in the gradient direction, zigzagging, or circumnavigating dangerous areas. These trajectory were not imposed to the robot in the coding phase, but were later observed during the trials. It can be theorized that these emerging behaviors can be useful in both managing terrain characteristics such as the slope but also optimizing energy use.

The proposed approach presents several limitations. First, the system builds an elevation map from onboard sensing and relies on it for planning, but it does not currently support replanning as new sensory information becomes available during execution. As a result, the planning horizon is effectively bounded by the size of the initial map, and any discrepancies between the precomputed map and the actual environment are not updated once the robot is in motion.

During navigation, the framework depends entirely on local perception and the robot's locomotion capabilities. As observed in both simulation and real-world trials, this can lead to ϕ_r and θ_r occasionally exceeding the prescribed limits. This issue is further influenced by the assumption that the terrain slope between consecutive waypoints remains approximately constant, which may not always be valid. Reducing the spacing between waypoints could help mitigate this effect, although it would likely increase computational requirements.

A second limitation is the lack of testing on highly challenging terrains. In most outdoor conditions, the Spot platform was able to ascend slopes directly along the gradient without difficulty—even when carrying an additional load of approximately 9 kg (six bottles of water, see Figure 8.1)—and did not require alternative navigation strategies to reach the goal. The environments used for real-world experiments, such as the parking area, feature relatively mild inclinations suitable for vehicle traffic, and therefore do not pose a significant risk of instability for the robot. It should be noted, however, that Spot represents only one example of a quadruped system, and other robots may encounter greater difficulty under similar conditions. In other more challenging terrains, such as the rocky mound observed in Foligno, Figure 8.1, it was possible to observe failures even with the Spot robot from Boston Dynamics. Nevertheless, no clear evidence was found to suggest that a particular path choice would consistently yield better performance in that setting.

8.5 Conclusion

In this chapter, we presented a global path-planning algorithm that accounts for the slope constraints of quadruped robots operating in outdoor environments. The proposed approach enables the generation of safer trajectories by trading off path length for terrain traversability.

Both simulation and real-world experiments with the Boston Dynamics Spot robot demonstrated that considering slope constraints can significantly alter the selected route and prevent failures that may occur when terrain inclination is ignored.

The planner was validated through 96 simulated and 60 real-world runs. The results showed that stricter pitch and roll constraints (Φ and Θ) increased the detour ratio and path tortuosity compared with the unconstrained baseline ($p < 0.001$), while maintaining the robot's actual pitch and roll angles (ϕ_r and θ_r) within the specified limits and significantly lower than those observed without constraints ($p < 0.001$). Furthermore, the inclusion of roll constraints reduced the number of observed failures, indicating the potential of the proposed approach to improve the robustness and safety of outdoor navigation on uneven terrain.

Chapter 9

Emotionally-modulated Artificial Potential Fields

Navigation in unstructured environments, like the application described in Chapter 6, comprises both challenges concerning the terrain the robot has to traverse, analyzed in Chapter 10, and challenges with obstacles that may be present in the environment. The problem of robot navigation has been extensively studied, with planning-based approaches forming the core of most solutions. However, in dynamic environments where unexpected obstacles or sensor noise alter the scene, the typical response is to trigger replanning. This process can limit the system's ability to react promptly to rapidly changing situations. Mobile robots that rely exclusively on deliberative planning for motion execution may struggle to respond effectively to unforeseen obstacles, such as pedestrians.

Figure 9.1 shows a snapshot from experiments conducted at the Zachary building of Texas A&M University, where the robot operated in an environment shared with people. Different human behaviors can be observed: some individuals maintained a safe distance while monitoring the robot, one approached closely in a playful manner, and others walked past without interacting with it. Given the variability of such responses, it is not feasible to anticipate all possible behaviors or explicitly incorporate them into a fully safe motion plan from the beginning of the navigation task.

In both a SAR scenario or the inspection of a work environment, the robot must operate in an environment where people might be present, in particular, either FRs or workers busy with completing other tasks. As mentioned in 3.1.3, FRs, in particular, envision an autonomous robot capable of completing a series of tasks independently, both before FRs arrive and in parallel with FRs while they are busy carrying out more demanding tasks. For these reasons, a robot capable of operating in these scenarios has to react to unpredictable environments, in

the presence of people moving around, while maintaining a high level of safety and reliability of operations.

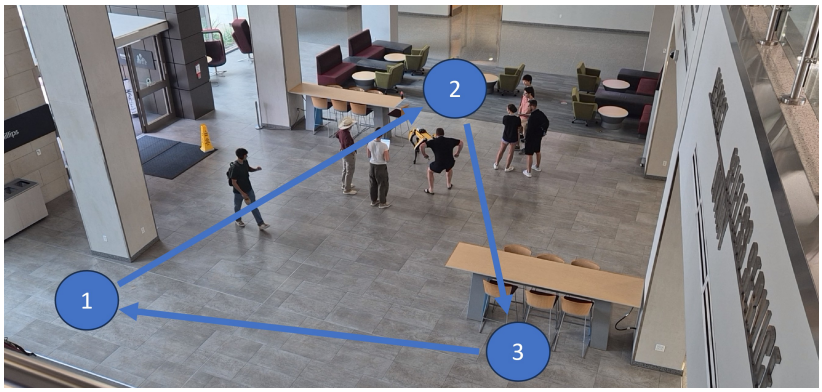


Figure 9.1 Experiments at Zachary building with a robot and people exhibiting unpredictable behaviors.

A common way to address the limitations of purely deliberative planning systems is to adopt a hybrid architecture, combining a high-level planner responsible for generating a global path with a lower-level reactive module that handles unexpected changes in the environment. Such mixed approaches have been widely investigated over the past decades [140, 3].

In biological systems, including primates and humans, emotional mechanisms play a key role in guiding responses to environmental stimuli and in shaping interactions with the surrounding world [46]. In a similar way, emotion-inspired models can be leveraged to improve the reactivity of robotic navigation in uncertain or dynamic settings. Such mechanisms may also serve as regulators of environmental responses, modulating the behavior of the reactive control layer. In the following chapter we propose a navigation system based on an emotion-inspired mechanism to regulate the reactive layer of the navigation framework. In this formulation, emotional states are used to adjust how the robot responds to environmental stimuli in a reactive way.

9.1 Methodology

In this section, we outline the biological foundations underlying an emotion-driven navigation system and describe how these principles are translated into our computational framework. First, we analyze how emotions influence reactions in biology. Secondly, we apply the same

principles to an APF-based framework, focusing on how the framework reflects biological systems, and how this can improve the navigation task in unpredictable environments.

9.1.1 Approach

Ekman defines six emotions, called the Basic Emotions: Anger, Surprise, Happiness, Sadness, Fear, and Disgust [53]. These emotions are universal, meaning they are present and recognizable in people from different cultures, as well as in primates [84, 5, 201]. These emotions arise as reactions to a stimulus, are easily distinguishable, and have physiological effects. It is also important to note that the Basic Emotions are triggered by universal situations, occurring when something significant happens, both for the achievement of goals and for survival. Anger is triggered when something prevents the achievement of a personal goal. Surprise is triggered when reality differs from predictions. Happiness indicates the achievement of a motivational goal. Sadness corresponds to the loss of something, material or not. Fear signals the recognition of something dangerous. Finally, disgust is triggered when something offends the taste or moral sense.

The Basic Emotions have a functional purpose: they serve as responses to specific triggers. In the following two subsections, the functional purpose and effects of Anger and Surprise will be explained in detail.

Anger

Anger is related to the impossibility of reaching a goal; whenever the path to the goal is blocked by something or someone, anger is activated. Anger is the response to an interference with the achievement of goals.

Anger functions to remove the obstacle and reach the goal, initiating various mechanisms to actively eliminate the blocking factors. As a result, people become ready to do whatever it takes to remove the blockage; they experience increased blood pressure, sweating, a higher respiratory rate, increased swallowing, and tingling in the hands.

Surprise

Surprise is the reaction to something unexpected and unpredictable. When people experience the breaking of predictive schemes, it triggers the emotional response of surprise. An example of a situation that triggers a surprise reaction is sudden, unexpected movements.

Its functional purpose is to collect more information about the surroundings; people focus their attention more, trying to capture additional details. To this end, people open their eyes

wider to let in more light and inhale more oxygen. The goal is to gather information and prepare for a faster reaction if needed.

9.1.2 Implementation

Historically, robots based on biological systems have been described in terms of behaviors [8]. Behaviors are defined as sequences of a perceptual schema and a motor schema. Perceptual schemas describe how information is extracted and processed, while motor schemas describe how this information is used to generate the robot's motion. We propose an architecture where a simple navigation behavior, based on APF, is influenced by emotions. In particular, Surprise influences the perceptual schema, while Anger influences the motor schema. This section first describes the perceptual and motor schemas implemented on the robot; subsequently, it details the effect of emotions on the schemas. The code of the proposed algorithm is available on GitHub[1].

The perceptual schema of the robot collects data from the sensors and processes it into signals usable for navigation. The main requirement for this module is fast data processing; therefore, a compact representation of perceptual information is used. Obstacles around the robot are detected using a ray-tracing approach. At discrete angular intervals, sensing rays are cast, and for each ray, only the position of the closest obstacle is recorded. This results in a sparse yet representative model of the surrounding environment. The perceptual schema is regulated by two parameters: the *angular intervals* and the *maximum distance*, D_{max}, at which obstacles are perceived. The positions of the obstacles are then provided to the motor schema, which elaborates them to generate the repulsive force field.

For the potential fields, different magnitude profiles have been implemented in past works [142], depending on the requirements and the desired behavior for each potential field. The magnitude of the repulsive field is a hyperbolic function to ensure that the robot moves away faster from closer obstacles, while the effect is reduced if the obstacle is far away. This function is described in Eq. 9.1. In this equation, q_{repel} is the coefficient regulating how steep the curve is, and *Dist* is the distance of the obstacle to the robot. Additionally, the force is saturated at a maximum value to avoid very high forces and unpredictable robot behaviors. Beyond the maximum distance, which is the same as in the perceptual schema, the force is zero since no obstacles are detected at those distances. It is important to notice that for the repulsive force, two parameters regulate the shape of the function: *maximum force*, F_{max}, and *force coefficient*, q_{repel}. In Fig. 9.2, an example of obstacles perceived by the robot and the

[1]https://github.com/ZoeBetta/EMAPF.git

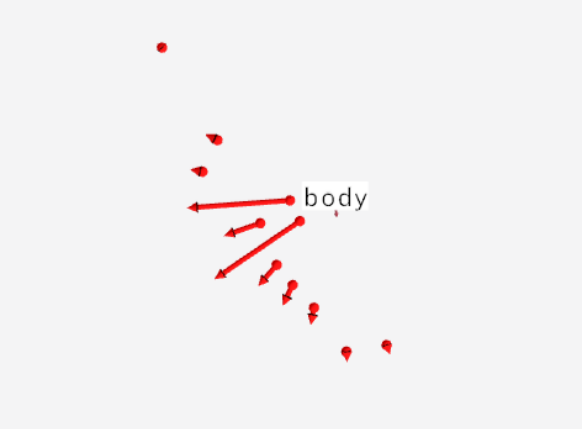


Figure 9.2 Perceived obstacles and resulting repulsive force.

resulting force field is shown. The robot is at the center, where the word "body" is written. We can see that, on the left side of the robot, two obstacles are particularly close; the arrows represent the force field, which is perceived as repulsive (negative sign) by the robot. We can also see that obstacles that are far away result in a lower force.

$$RepulsiveForce = \frac{q_{repel}}{Dist} \tag{9.1}$$

$$AttractiveForce = \begin{cases} q_{attract} \cdot Dist & \text{if} Dist \leq Thresh_D \\ ConstForce & \text{if} Dist > Thresh_D \end{cases} \tag{9.2}$$

The attractive field is used to guide the robot towards the WP. The magnitude profile for the attractive force is a constant field, as described in Eq. 9.2. Only in the close vicinity of the goal does the magnitude profile decrease linearly to avoid overshooting and to achieve smoother behavior.

Emotions are used to regulate the parameters of the perceptual and motor schemas, such as: *angular intervals*, D_{max}, F_{max}, and q_{repel}. As Anger increases, both F_{max} and q_r increase, while as Surprise increases, A_{int} decreases and D_{max} increases

9.1.3 Anger

The internal state Anger increases when the robot is not making progress towards the goal; it depends on the first-order derivative of the distance to the goal. Anger is related to the impossibility of reaching the goal—the absence of progress in achieving the objective. The objective of the robot is to reach a position in space, reducing the distance to that position.

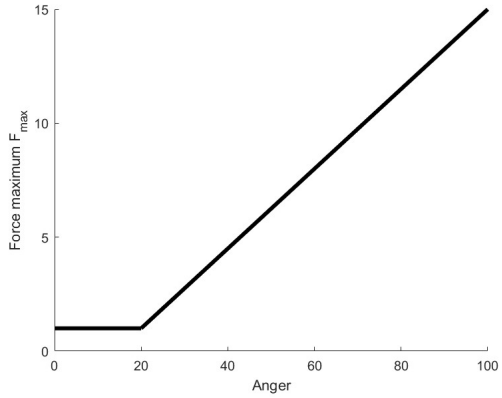
For this reason, Anger increases when the distance to the final position remains unchanged or increases. In Eq. 9.3, the equations regulating the Anger mechanism are presented. The value of the emotion at time t, $A(t)$, is calculated as the sum of the previous emotional value, $A(t-1)$, and a value that varies depending on the robot's progress. If the first-order derivative of the distance to the goal, $\dot{Dist}$, is zero, the robot is not making progress, and Anger increases by a constant value. If the distance increases (positive derivative), Anger increases proportionally to the derivative. If the distance decreases (negative derivative), Anger decreases. It is important to note that, as time increases, Anger grows faster even with the same derivative. The variable Min represents the number of minutes elapsed since the WP was set as the position in space to reach. If a long time passes before the robot reaches the WP, the Anger level increases faster, while its decrease is not dependent on the time passed, making the robot more likely to maintain a higher level of Anger for longer.

$$A(t) = \begin{cases} A(t-1) + \gamma \cdot (Min+1) \cdot Const & \text{if } \dot{Dist} = 0, \\ A(t-1) + \gamma \cdot \dot{Dist} & \text{if } \dot{Dist} < 0, \\ A(t-1) + \gamma \cdot (Min+1) \cdot \dot{Dist} & \text{if } \dot{Dist} > 0, \end{cases} \tag{9.3}$$

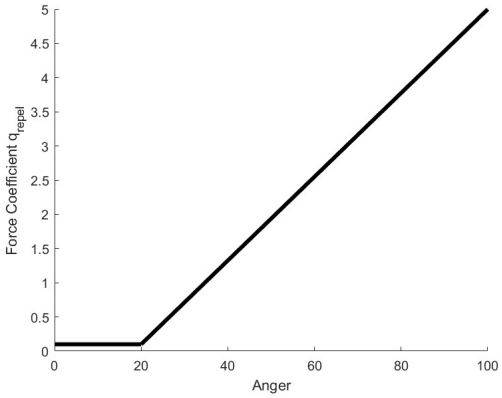
Anger affects the repulsive field generated by obstacles; it changes how strong the repulsion to the obstacle is. The higher the anger level, the higher the repulsion. In people, Anger is associated with removing obstacles that block progress toward a goal. In this context, obstacles prevent the robot from reaching its goal. By moving further away from them, the robot may find an alternative path. For this reason, Anger strengthens the repulsive field, causing the robot to move further away from obstacles as Anger increases. The Anger level is used to regulate the motor schema parameters of the robot, in particular F_{max} and q_{repel}. As shown in Fig. 9.3, while the Anger level on the x-axis is below a given threshold, the parameters remain constant; after the threshold is exceeded, they increase linearly until a maximum value is reached. If the Anger level increases beyond the maximum, the WP is canceled, and the robot attempts to proceed to the subsequent WP opportunistically.

9.1.4 Surprise

The internal state Surprise increases if the predicted location of obstacles differs from the actual obstacles in the environment. Surprise is related to breaking the predictive schema; if reality differs from previous predictions, Surprise is activated. Each time the obstacles are received, the correct locations are compared to the obstacle locations at the previous time step. The percentage of obstacles that do not match between the prediction and the



(a) Maximum repulsive force as a function of anger.



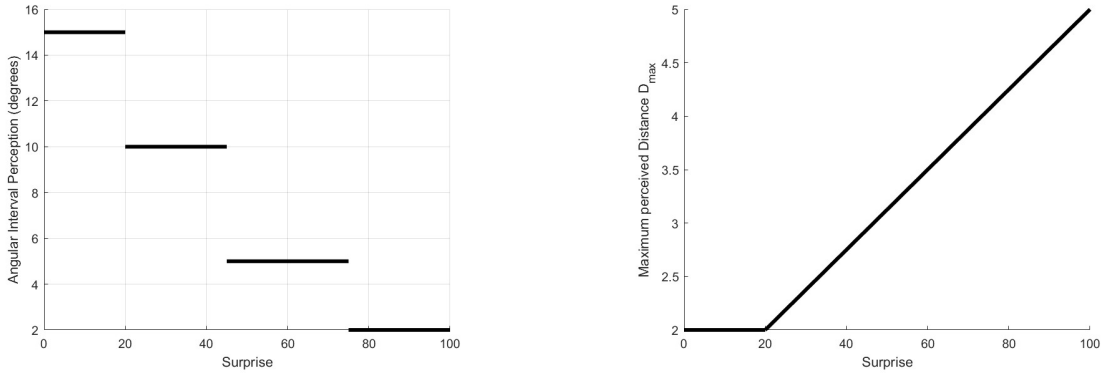
(b) Repulsive force coefficient as a function of anger.

Figure 9.3 Effects on the repulsive field given the anger level of the robot.

actual reading, $mismatch_{\%}$, is calculated and used to increase the Surprise level. As shown in Eq. 9.4, if the mismatch is greater than a given threshold, the Surprise value increases proportionally to the percentage of mismatched obstacles. Conversely, if the percentage is lower, the Surprise level decreases by a constant value. Similarly to Anger, the Surprise level at time t, $S(t)$, depends on the Surprise level at the previous time, $S(t-1)$.

$$S(t) = \begin{cases} S(t-1) + \gamma \cdot mismatch_{\%} & \text{If } mismatch_{\%} > T, \\ S(t-1) - \gamma & \text{If } mismatch_{\%} \leq T, \end{cases} \tag{9.4}$$

Surprise changes what the robot perceives; it decreases the discretization of obstacles and increases the distance at which obstacles are detected. Surprise functions to enhance perception. Similarly, when people are surprised, they gather more information about their environment. As shown in Fig. 9.4a, the *angular interval* decreases as a step function. In Fig. 9.4b, the maximum perceived distance of obstacles, D_{max}, is shown as a function of the Surprise level. Until the Surprise level reaches a threshold, the maximum distance remains constant; beyond this threshold, it grows linearly until the maximum Surprise value. The Surprise value is limited to a maximum, but this value does not affect the behavior of the robot. A possible future implementation could involve slowing the robot down if Surprise reaches its maximum value.



(a) Angular distance for the obstacle discretization as a function of the surprise level.

(b) Maximum obstacle perception distance as a function of the surprise level.

Figure 9.4 Effects on the repulsive field given the anger level of the robot.

9.2 Experiments

The experimental procedure is described in the following section. First, we aim to evaluate the effect of single emotions on the ability of the system to navigate, to better isolate the effect of emotions. Secondly, the experiments comparing EMAPF to other state-of-the-art local navigation algorithms are described and then analyzed to show the difference in performance of the systems. Subsequently, structured experiments with dynamic obstacles are described and discussed to evaluate the effect of emotions on the resulting navigation of the robot. Finally, experiments with people in an unstructured environment are proposed.

The experimental protocol aims to evaluate the effect of emotions and internal state on the navigation of a robot in the environment. If a difference is observed, the experiments aim to explore whether this difference leads to an increase in the robot's performance in the navigation task. Additionally, it is necessary to study what happens when obstacles are unpredictable, such as humans moving around the robot.

To better identify the individual effect of emotions, we tested the system in a simple simulation environment. To evaluate the performance of the system, both in comparison to other local navigation algorithms and in the presence of dynamic obstacles, a total of 20 structured runs were conducted. The tests were performed with the Spot robot from Boston Dynamics, and all computations were run on the onboard SpotCore computer. In addition, data from 47 minutes and 52 seconds in the presence of unstructured dynamic obstacles were recorded. In the following sections, the experiments used to compare different algorithms and to evaluate the behavior of the robot in the presence of structured and unstructured dynamic obstacles are presented.

9.2.1 Ablation experiments

To study the effect of single emotions in a navigation task, a Gazebo simulation environment was developed. Navigation was implemented on a simple omnidirectional robot [58]. Additionally, a sphere moving back and forth was introduced. The navigation was tested under three conditions: with only Anger active, with only Surprise active, and with both emotions active.

9.2.2 Comparison with other algorithms

The EMAPF framework was evaluated in comparison with the APF system and the Timed Elastic Band (TEB) system [167]. APF and TEB are widely used algorithms for local navigation. For the comparison with other algorithms, the robot was tasked with reaching two waypoints (WPs) and a final goal. The time to reach the final goal and the success rate at each WP are discussed; additionally, the distance is measured every 30 seconds to show the initiative of the robot. In Fig. 9.5, the overhead view of the environment is shown for the comparison of the different algorithms. The star marks the starting location, and each numbered circle represents a WP. The WPs were progressively more difficult to reach. WP1 was in a straight line in front of the starting position, with no obstacles in front of the robot; to reach WP2, the robot could not follow a straight-line path due the presence of obstacles, and additionally, it was inside a C-shaped structure, with the opening on the side the robot was coming from; WP3 was inside a C-shaped structure, with the opening on the opposite side from the robot's direction of arrival, and another C-shaped structure was in between, where the robot could enter and get stuck. This location was chosen to stress the system in the presence of these C-shaped structures, which are especially challenging for APF-based systems due to the presence of local minima. We run each algorithm 5 times, for a total of 15 runs. Each run began with the robot tasked with reaching WP1, followed by WP2 and the final WP, and was terminated for one of three reasons: (i) the robot reached the final goal, (ii) 10 minutes had passed since the start of the experiment, or (iii) the robot decided not to pursue the final WP because it was deemed unreachable. The EMAPF system outperformed the other two algorithms both in terms of time to complete the task and success rate in reaching the final WP, while showing greater initiative.

EMAPF is the purely reactive system, in this specific implementation, it uses Anger and Surprise to modulate the response to the environment, as described previously. The weights for the emotions and the parameters were determined through a series of experiments conducted both indoors and outdoors. For the APF algorithm, the same architecture was

replicated with all emotions set to zero to evaluate the difference in performance without the effect of emotions ($\gamma_1, \gamma_2, \gamma$ set to zero in Eq. 9.3 and in Eq. 9.4). Finally, the TEB algorithm is a state-of-the-art local navigation algorithm, compatible with the MoveBase Navigation Stack available on ROS. It is particularly relevant for the Spot robot, a holonomic quadruped, because of its compatibility with holonomic robots. For this algorithm only, a local costmap was implemented. TEB was chosen for its widespread adoption.

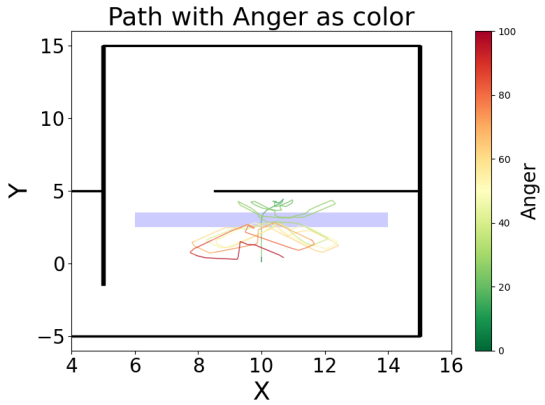


Figure 9.5 Experimental setting for comparing different local navigation algorithms and evaluating dynamic obstacles.

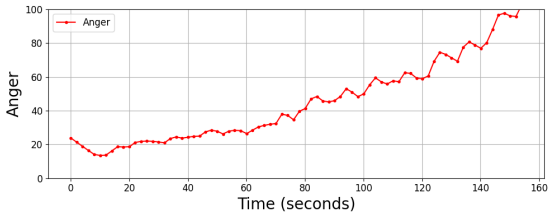
9.2.3 Dynamic obstacles

The system was tested in 5 trials with dynamic obstacles by placing a moving obstacle in front of the robot for one minute, in five different positions, one per trial. The time to reach the goal has been analyzed, as well as the qualitative discussion of the effect of emotions on the navigation.

The task given to the robot was the same as in the experiments described in Sec. 9.2.2: the robot started at the star symbol and had to reach three subsequent WPs. The difference was that in each trial, a dynamic obstacle moved along a half-circle path in the direction of the robot's motion. The person moved in front of the robot at five different points, one per each trial, along the path: once between the start and the first WP (first row of Table 9.1), twice between the first and the second WP (second and third row), and twice between the second and the final WP (fourth and fifth row). In Fig. 9.5, the location of the WPs, and with green squares, the location of the dynamic obstacles. The stopping conditions for the experiments were the same as those described for the comparison experiments.



(a) Path of the robot, color representing the Anger.



(b) Anger level in time.

Figure 9.6 Ablation experiment with only Anger active.

9.2.4 Unstructured human interaction

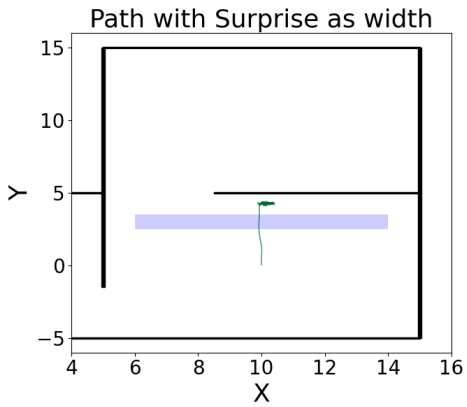
The system was tested on July 8th, 2025, at the Zachary building of Texas A&M, in the presence of people who were asked to interact with the robot as they wished. Qualitative measures of how the emotional system changes depending on the environment are analyzed. The robot was tasked with moving between three locations in cycles, as shown in Fig. 9.1. People were free to interact with the robot and move around it. The total operation time was 47 minutes and 52 seconds. The interactions were recorded using videos from a bird's-eye view perspective.

9.3 Results

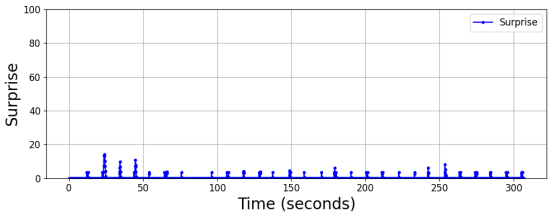
9.3.1 Ablation experiments

The results are presented in Figures 9.6, 9.7, and 9.8. In these figures, the Gazebo walls are represented in black, and the area in which the sphere moves is shown in light blue. The path of the robot is shown, with the values of Surprise and Anger encoded as the width and color of the line, respectively. Higher widths indicate higher Surprise levels, while colors shifting toward red represent higher Anger values. Higher widths indicate higher Surprise levels, while colors shifting toward red represent higher Anger values.

Only the combination of both emotions leads to successful navigation. In Figure 9.6, on the left is the path of the robot when only Anger is activated, while on the right is the Anger level in time. When only Surprise is activated Figure 9.7, on the left is the path of



(a) Path of the robot, with width representing Surprise.



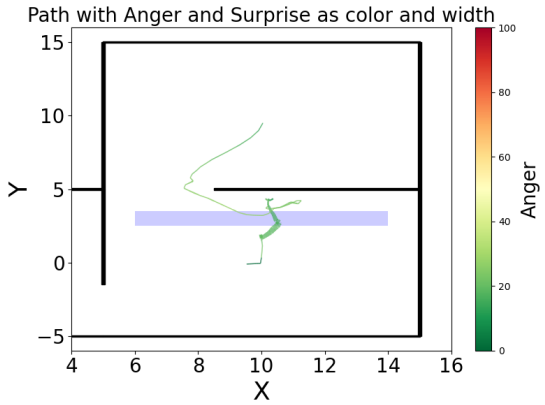
(b) Surprise level in time.

Figure 9.7 Ablation experiment with only Surprise.

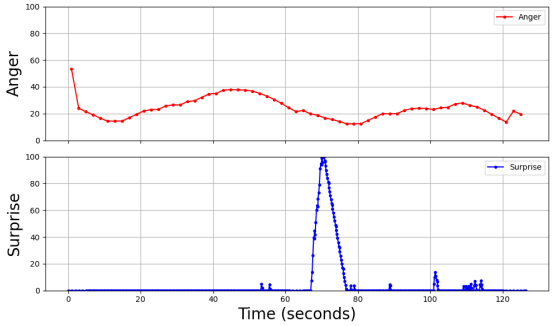
the robot with the Surprise represented as width, and on the right the Surprise level in time. In this case, we stopped the simulation after 5 minutes, since the robot was not making any meaningful progress. In contrast, when both Anger and Surprise are active, the robot is able to reach the goal, Figure 9.8a. We can also see that the robot increases its level of both Anger and Surprise, represented in Figure 9.8b. The increase of Anger is not linear, and after the beginning, where it increases, to reach a peak, it decreases when the robot makes progress towards the goal. Similarly, for the Surprise level, it increases in the proximity of the moving sphere and decreases when further away from it.

9.3.2 Comparison with other algorithms

Compared to the other two systems, EMAPF achieved the goal more consistently. In Fig. 9.9a, the results are shown: in blue, the number of times the robot reached WP1; in yellow, the number of times it reached the second WP; and in green, the number of times it reached the final WP. The horizontal blue lines represent the maximum number of reachable WPs. It can be observed that all algorithms reached WP1. Both the EMAPF and the APF algorithms always reached WP2; however, TEB reached the second WP only twice, and the third WP was never reached. EMAPF reached the final WP four times, compared to two instances for the APF system. Additionally, the TEB algorithm reported two critical failures in which the system abruptly stopped working; the time was saturated to 600 seconds for those. The total time of operation for the three navigation algorithms is shown in Figure 9.9b. EMAPF exhibits a relatively low median execution time (approximately 260 s), with a moderate spread of values ranging roughly between 190 s and 460 s. APF shows a higher median time



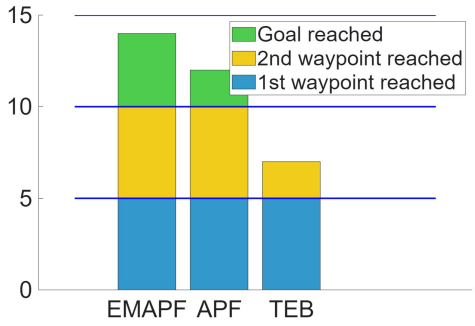
(a) Path of the robot, color representing Anger, and width representing Surprise.



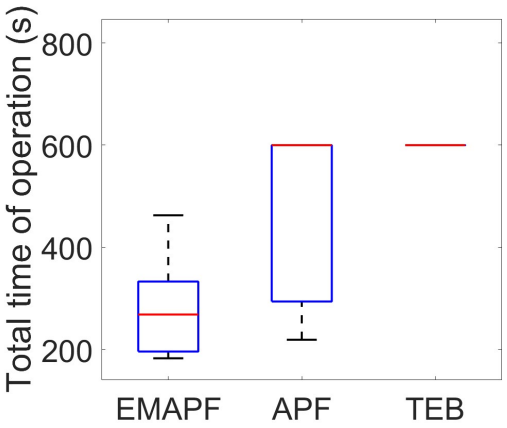
(b) Anger and Surprise level in time.

Figure 9.8 Ablation experiment with both Anger and Surprise active.

(around 600 s) and a larger variability, with values spanning from about 220 s to 600 s. In contrast, TEB presents a single observed value at approximately 600 s, indicating that the algorithm was never able to reach WP3. By comparing the times to complete the mission, using Mann-Whitney U tests with $\alpha = 0.1$ showed significant differences: EMAPF vs. APF, $p = 0.0907$; EMAPF vs. TEB, $p = 0.0075$.



(a) Number of times each algorithm reached the WPs across 5 runs per algorithm.



(b) Time to reach the final WP.

Figure 9.9 Comparison of performance metrics across experiments.

Qualitatively, the system controlled with EMAPF moves around more and shows greater initiative, attempting to find better paths to reach the goal. To quantify this difference, the position of the robot was recorded every 30 seconds, and the Euclidean distance between two subsequent positions was calculated. In Fig. 9.10, this measure is shown for all three algorithms. The average distance for EMAPF is higher, and the difference is statistically

significant. The EMAPF system moves more and covers more distance than APF. Welch's two-sample t-test with $\alpha = 0.05$ indicated significant differences: EMAPF vs. APF, $p = 0.0030$; EMAPF vs. TEB, $p = 0.0002$.

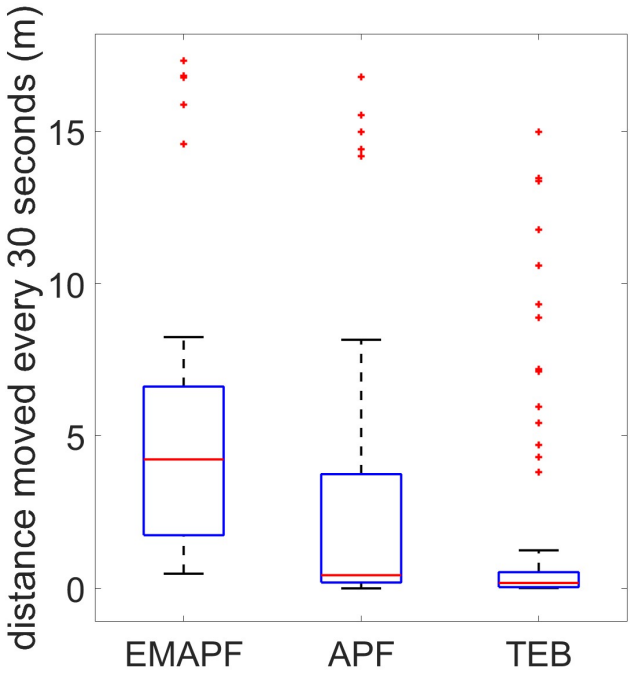
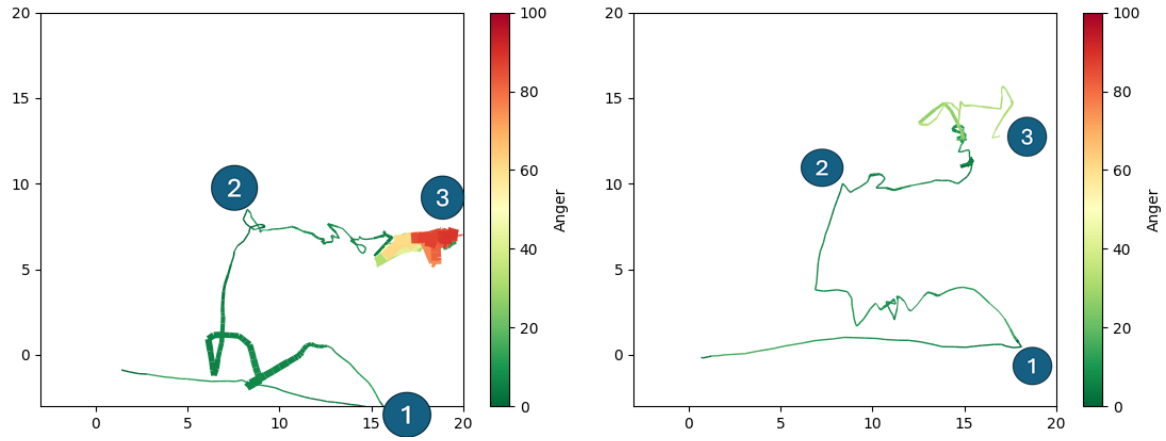


Figure 9.10 Distance measured each 30 seconds comparing the different algorithms.

9.3.3 Dynamic obstacles

In Fig. 9.11, the path of the robot is shown as described in Section 9.2.1. In Fig. 9.11a, it can be seen that the robot's Surprise level increases at two points: once near the beginning, when close to the flowers, due to the texture of the plants and the wind moving the leaves, and again when the robot is blocked near the goal. Near the final WP, Anger also increases, as the robot is unable to make progress due to a person obstructing it. By comparison, in Fig. 9.11b, where no dynamic obstacles were present, Anger also increases toward the last WP, but less than when dynamic obstacles physically block the robot. This is because the last WP is very difficult to reach. Surprise also increases at times in this case, but to a lesser extent than with dynamic obstacles, primarily due to intrinsic sensor noise. Overall, both Anger and Surprise increase more when dynamic obstacles are present.



(a) Trajectory of the robot with dynamic obstacles.

(b) Trajectory of the robot without dynamic obstacles.

Figure 9.11 Trajectory of the robot, with color representing the Anger level and linewidth representing the Surprise level.

Table 9.1 Times to reach WPs in the Dynamic scenario

Location of dynamic obstacle	Time from Start to WP 1	Time from WP 1 to WP 2	Time from WP 2 to Stop	WP 3 reached
start-WP1	90.72 s	45.13 s	196.93 s	True
WP1-WP2	23.76 s	138.5 s	135.11 s	False
WP1-WP2	29.3 s	118.65 s	360.99 s	False
WP2-WP3	26.02 s	63.86 s	373.22 s	False
WP2-WP3	30.62 s	96.62 s	138.29 s	True

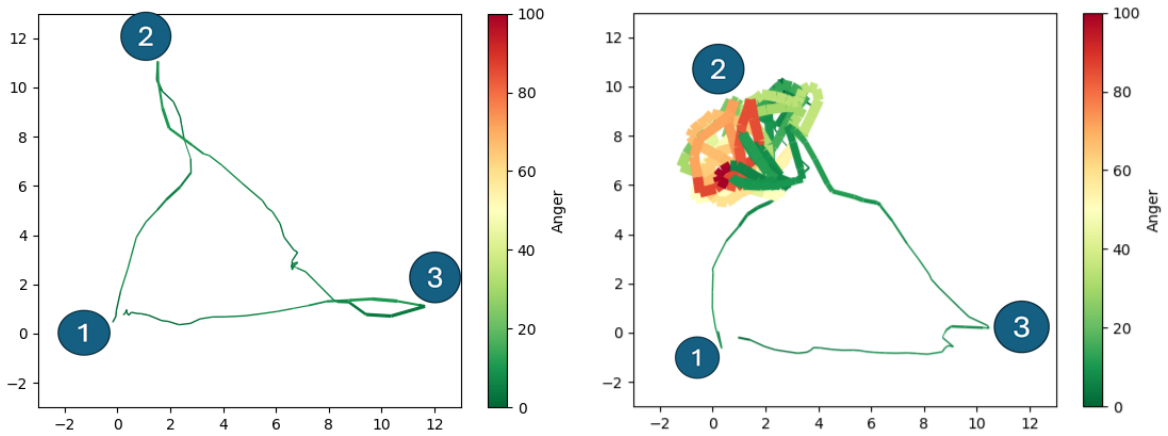
Table 9.1 reports the success rate for the dynamic trials experiments, which is analyzed to understand how dynamic obstacles influence the robot's operations. The first column reports the path segment where the dynamic obstacle appeared: between the start and WP1 in the first trial, between WP1 and WP2 in the second and third trials, and between WP2 and WP3 in the fourth and fifth trials. The second and third columns show the time it takes for the robot to reach each WP, measured from when the WP is received to when it is reached. The fourth column instead reports the time from when the last WP is assigned to the robot until the end of the experiment—either because the WP is reached or deemed unreachable by the system. The last column of the table reports the success in reaching the final WP. When comparing the number of successful trials in the presence of dynamic obstacles (2 times) with the number of successful completions reported in Fig. 9.9a for the same task with only static obstacles (4 times), it is evident that the robot reaches the goal less frequently when dynamic obstacles are present. However, the success rate is the same as in the scenario with only static obstacles using the APF without emotional modulation (2 times).

Looking at the times recorded in the dynamic scenario trials, reported in Table 9.1, it is possible to notice that, even when the robot decides not to pursue the last goal, the decision is dependent on Anger dynamics, and not on a mere evaluation of the time expired since the robot started.

9.3.4 Unstructured human interaction

Comparing a cycle of WPs without people (Fig. 9.12a) and with people (Fig. 9.12b), it can be observed that Anger and Surprise, represented as described previously for Fig. 9.11, increase more when people are present. When no people are passing by, the goals are simple, and Anger remains low, except for small fluctuations. Similarly, since there are few dynamic obstacles, the robot's Surprise does not increase significantly; the fluctuations are mainly due to sensor noise.

In contrast, in the presence of people, especially while reaching the second WP, people actively interact with the robot, attempting to block it. An example of this situation can be seen in Fig. 9.1. Anger increases, with fluctuations corresponding to the small progress the robot makes before being blocked again. In this case, WP changes are triggered by the level of Anger. Similarly, Surprise increases from the moment people begin interacting with the robot and remains high.



(a) Trajectory of the robot without people around.

(b) Trajectory of the robot with people interacting with it.

Figure 9.12 Trajectory of the robot, with color representing the Anger level and linewidth representing the Surprise level.

9.4 Discussion

By looking at the ablation experiments, in particular where only Anger is active (Figure 9.6), the robot moves to avoid the dynamic obstacle, but because the perceptual schema is unchanged, it reacts to the sphere only when it is close, making timely reactions more difficult. The robot merely escapes from the sphere when it is very close, moving towards the starting position instead of towards the goal. The goal is removed when the Anger level reaches its maximum A_{max}.

On the contrary, by observing Figure 9.7, when only Surprise is active, the robot reaches a local minimum near the wall and becomes stuck, unable to explore alternative strategies. In this scenario, Surprise is briefly activated whenever the sphere passes behind the robot; in Figure 9.7b, it is possible to see small periodic increments, but because the sphere quickly moves away, the robot perceives this as noise, and the Surprise level does not increase significantly. Since Anger is deactivated, the simulation is stopped after 5 minutes.

In the simulation with both Anger and Surprise active (Figure 9.8), the robot moves away from the wall, avoiding the local minima due to the effect of Anger. It is then closer to the moving sphere when it detects it, and Surprise is fully activated, allowing the robot to successfully move away from the sphere while still perceiving it and reach the goal.

In the comparison experiments, the time required to complete the task or terminate the trial was analyzed. EMAPF showed a lower average completion time, and this difference was statistically significant. For the two critical failures of the TEB algorithm, the time

was saturated at 600 seconds. This difference can be explained by the fact that EMAPF, when Anger increases beyond its maximum level, opportunistically cancels the WP, whereas APF continues attempting to reach it; when APF is unable to reach the final WP, the trial is terminated after 10 minutes.

By comparing EMAPF and the APF system, it can be observed that the robot equipped with emotions, when not making progress, begins to explore the environment more actively, while the emotionless robot tends to get stuck in loops or local minima.

Comparing the Anger and Surprise levels in experiments with and without dynamic obstacles, it is possible to observe that in the presence of dynamic obstacles, emotions reach higher levels compared to the case of static obstacles only.

Additionally, the reader might be confused by the fact that, in the first trial—when the person moves between the start and WP1—the robot successfully reaches WP1, and subsequently WP2 and WP3. In contrast, in the second and third trials, when the person moves between WP1 and WP2, the robot can reach WP2 but struggles to reach WP3, even though no additional dynamic obstacle is present in the final path segments. This can be attributed to random factors, as goal reachability can never be fully guaranteed due to the absence of a global map and the reactive nature of the approach.

Looking specifically at the fourth column of Table 9.1 representing the times, it can be observed that in the second trial, the time before stopping while the robot was heading to WP3 (135.11 s) is shorter than the time it took to successfully reach WP3 in the first trial (196.93 s). This difference can be attributed to the fact that, in the first trial, the robot was actively making progress toward the goal, while in the second trial, it was not—causing the Anger level to rise more quickly. Remark that no fixed timeout stops the robot arbitrarily; instead, it may continue as long as it is still making meaningful progress toward the WP.

Finally, Emotions in people interact with each other; similarly, the results of this work suggest that Anger and Surprise may be related. In Figure 9.12b, it is possible to see that near the second WP, both Anger and Surprise reach high levels. This interaction can be explained by the fact that the perceptual and motor schema, influenced by emotions, are strongly correlated. A higher Anger level causes the robot to move faster, which reduces the precision of obstacle perception and, in turn, causes the Surprise level to rise. This can also be observed in Fig. 9.11a, where toward the third WP, both Anger and Surprise are high. It is important to note that there are also cases where only one emotion is active. In Fig. 9.11a, between the first and second WP, only Surprise is activated, while Anger remains very low at all times. Similarly, in Fig. 9.11b, toward the third WP, only Anger is activated, and Surprise

remains limited. Future testing aims at better evaluating the interactions between emotions and how those interactions influence the system's performance.

9.5 Conclusion

This chapter presented EMAPF, a framework for reactive local navigation, and demonstrated its application through an emotion-based navigation system incorporating the emotions of Anger and Surprise. These emotions were modeled to regulate the robot's motor and perceptual schemas, respectively. Unlike most previous studies on emotion-based robotics, which primarily focus on motor behavior and are evaluated only in simulation, the proposed approach integrates emotional influences across multiple navigation components and has been validated on a real robotic platform.

As a reactive local navigation system, EMAPF is cognitively plausible and inherently robust to dynamic obstacles. Experimental results indicate that the proposed approach achieves improved navigation performance compared with state-of-the-art local navigation methods. Furthermore, the analysis of the emotional mechanisms provides insight into how emotion-inspired modulation can influence navigation behavior.

The framework was extensively evaluated through real-world experiments, including comparisons with established local navigation algorithms and tests in environments containing both structured and unstructured dynamic obstacles. The results demonstrate the feasibility of the proposed approach and highlight its potential to improve navigation performance in dynamic environments.

Chapter 10

Search and Rescue Planning

In an unpredictable scenario, even while accounting for slopes in the environment, Chapter 8, or unpredictable obstacles, Chapter 9, navigation cannot be assumed as simple as moving to a different position; there might be a need to open doors with externally mounted robotic arms. To add to the complexity of navigation, there are also other tasks that, in general, a robot has to complete while inspecting a partially known environment; for this reason, there is a need to coordinate more complex sequences of operations. Indeed, among the key requirements for fully autonomous task execution are effective planning capabilities, which require a sufficiently expressive planning language, such as the PDDL. These elements play a crucial role in enabling robots to execute tasks autonomously while accounting for constraints related to time, resources, and environmental uncertainty [79, 36]. Similarly to what was done for Chapter 4, also in this chapter the focus is on a SAR application, due to the stricter requirements, and the higher level of unpredictability in the environment.

Although the importance of planning is widely acknowledged, only a few attempts have been made to integrate the basic sensorimotor capabilities of a quadruped robot with a planner able to address complex situations in vast environments during SAR operations. Such scenarios typically require deciding when it is time to explore or to assess whether staircases or corridors allow passage or are obstructed by debris; when the robot should activate its door-opening skill or determine whether doors are blocked; and whether the robot should initiate a procedure to identify people requiring assistance, with varying degrees of consciousness.

In this Chapter we presents a novel framework for deploying a quadruped robot in SAR scenarios, using the Boston Dynamics Spot robot as a case study. The key contributions of this work are as follows: the integration of PDDL-based planning to enable Spot to perform adaptive and efficient mission execution by selecting specialized robotic actions for assessing

the state of the environment and victims, evaluating their condition, and reporting using advanced sensory payloads; thorough simulation testing to evaluate the planning system's behavior as a function of the environment description and to determine its limits, particularly in terms of planning time; and an experimental evaluation of the proposed system conducted on the actual Spot robot in a real-world scenario that emulates a disaster situation.

10.1 Methodology

We designed a PDDL domain, comprised of predicates and actions. The actions have been designed to enable Spot to navigate a post-disaster environment in search of people. Once a person is located, the robot prioritizes assessing their condition and reporting their status. More specifically, our aim is to describe a domain that matches the expectations of FRs, described in Chapter 3 in a SAR scenario with a quadruped robot. In particular, the robot's inspection tasks include assessing the safety of the environment, detecting the presence of people (even when partially or fully hidden) using multiple sensors, and evaluating their condition. In particular, the robot is expected to gather key information such as the level of consciousness and vital health parameters (e.g., breathing, pulse, and body temperature). Additionally, once individuals are identified, the robot should determine whether safe paths exist to reach them and, if feasible, guide them toward an exit.

10.1.1 Predicates

In PDDL, predicates describe the properties of objects and their relationships within a planning domain. They establish conditions that can be true or false, reflecting the state of the world at a given time.

In line with the tasks that the robot must complete, the predicates defined for this problem domain are shown in Table 10.1.

To mention a few, predicates such as `stairs_connected`, `passage_checked`, `door_closed`, and `door_blocked` are used to describe the connectivity of the environment, i.e., to indicate whether a passage or a stair is traversable or climbable, or instead obstructed by debris, and whether a door is closed or blocked. The predicates `person_detected`, `person_evaluated`, `person_reported`, and `person_state` describe the condition of individuals found in the area, in accordance with the protocols adopted by rescuers to assess their state of consciousness. The predicates `gas_checked`, `crack_checked`, and `environment_checked` describe

the analyses performed in the environment through dedicated sensors, as typically used by rescuers during their operations.

Definition	Description
`(robot_at ?r - robot ?l - location)`	Specifies the robot's current location
`(is_free ?r - robot)`	The robot is not currently engaged in other actions
`(emergency ?r - robot)`	The robot is handling an emergency
`(not_emergency ?r - robot)`	The robot is not handling any active emergency
`(connected ?l1 ?l2 - location)`	Two locations are physically connected
`(passage_checked ?l1 ?l2 - location)`	The connection between two locations has been verified traversable
`(is_exit ?l - location)`	A room allows communication with FRs
`(stairs_connected ?l1 - location ?s - stairs)`	A room and a staircase are physically connected
`(stairs_checked ?s-stairs ?l1 ?l2-location)`	The stairs are currently traversable
`(room_open ?l - location)`	A room that is accessible and open
`(door_closed ?l - location)`	A door to a room that is closed but not blocked
`(door_blocked ?l - location)`	A door that is closed and cannot be opened by the robot
`(door_notchecked ?l - location)`	The condition of the door has not been determined yet
`(searched ?r- robot ?l - location)`	A location has been searched by the robot to look for people
`(person_detected ?p - person ?l - location)`	Tracks locations where people have been found
`(person_evaluated ?p - person)`	A person has been evaluated and their position is known
`(person_reported ?p - person)`	A person's status has been reported, marking the final stage of the assessment
`(person_state ?p - person ?s - state)`	Tracks the evaluation status of a person, recording their state after evaluation
`(dialog_finished ?p - person)`	Indicates that the interaction or dialogue with a person has concluded
`(person_dialog ?p - person ?c - consciousness)`	The details of a person's evaluation following dialogue, including their state of consciousness
`(battery_unchecked ?r - robot)`	The robot's battery level has not been checked
`(battery_checked ?r - robot)`	The robot's battery level has been checked
`(gas_checked ?l - location)`	The robot has checked the location for harmful gases
`(crack_checked ?l - location)`	The robot has checked for cracks at the location
`(environment_checked ?l - location)`	The robot has completed all the environmental checks required in the room

Table 10.1 Predicates of the domain

10.1.2 Actions

In PDDL, actions are used to change the state of the world toward the achievement of goal states. They are defined by their parameters, preconditions (i.e., predicates that must be true for the action to be executed), and effects (i.e., predicates that become true as a result of executing the action). The actions available in this domain are shown in Table 10.2: the first column reports the name of each action along with its parameters, while the second column provides a brief description.

Action	Parameters	Description
`move`	`(?r-robot ?l1 ?l2-location)`	Makes the robot navigate between two locations of the environment
`search`	`(?r-robot ?l-location)`	Makes the robot inspect a location, to detect people
`evaluate`	`(?l-location ?p-person ?r-robot)`	Orients the robot to face the person to better assess the person's position in space
`dialog`	`(?l-location ?p-person ?r-robot)`	Initiates a conversation between the robot and the person to assess consciousness to the call
`report`	`(?l-location ?p-person ?r-robot)`	Collects all information about each person and updates them if needed
`check`	`(?r-robot ?l1-location)`	Monitors the robot's battery level
`checkgas`	`(?r-robot ?l-location)`	Uses gas sensors to measure oxygen percentage and check for the presence of harmful gases
`checkcrack`	`(?r-robot ?l-location)`	Visually detects the presence of cracks
`checktemp`	`(?r-robot ?l-location)`	Verifies whether the temperature is safe
`reportemergency`	`(?r-robot ?l-location)`	Reports the emergency state to FRs
`checkstairs`	`(?r-robot ?s- stairs ?l1 ?l2-location)`	Controls if the staircase is traversable
`checkpassage`	`(?r-robot ?l1 ?l2-location)`	Visually checks if the connection between two locations is traversable
`climbstairs`	`(?r-robot ?l1 ?l2-location ?s-stairs)`	Allows the robot to switch floors
`checkroomopen`	`(?r-robot ?l-location)`	Checks if the door to a location is open
`opendoor`	`(?r-robot ?l-location)`	Opens the door
`searchoutside`	`(?r-robot ?l-location)`	Uses thermal cameras to detect the presence of people from outside a room

Table 10.2 PDDL actions

Remark 12. *After planning, when actions are executed, the expected effects may not materialize. For instance, this happens when a procedure used to check a condition (e.g., whether a corridor allows passage) returns an unexpected state (e.g., the corridor is blocked). In that case, the current plan will no longer be considered valid, and the planner will be triggered again.*

Move to

The `move` action enables the robot `r` to navigate between different locations, `l1` and `l2`, in the environment. The preconditions are as follows: `l1` and `l2` must be connected, `r` must be at `l1`, and the battery must be verified as sufficient to support the movement. The effects are that `r` is no longer at `l1` but at `l2`, and the battery check status is invalidated.

Search

The `search` action enables the robot `r` to visually scan a specified location `l` for people. The preconditions are as follows: `r` must be available (i.e., not performing any other actions) and must be located in `l`. During the action, `r` becomes unavailable but returns to an available

state once the search is completed. The effects are that `l` is marked as explored and that no people are present in it. If this expected outcome does not materialize and a person is instead detected during execution, the predicate `person_detected` is set to true and added to the problem, triggering replanning. Additionally, the goal is updated to include the need to evaluate the person's state. Additionally, the battery status is marked as unchecked, with the relevant predicates updated accordingly.

Evaluate

The `evaluate` action allows the robot `r`, located at `l`, to focus on a person `p` by adjusting its heading to face them. Once focused, the robot assesses the person's position, determining whether they are standing, sitting, or lying down. The preconditions are as follows: `r` must be available, `p` must have been detected at `l`, and `r` must be at `l`. During the action, `r` becomes unavailable until the task is completed. Upon completion, `p` is marked as evaluated.

Dialog

The `dialog` action initiates a conversation between the person `p` and robot `r` to assess the person's level of consciousness at the location `l`. The preconditions are as follows: `r` must be available, `p` must have already been evaluated, and `r` must be in `l`. During the action, the robot becomes unavailable until the dialog is completed. Upon completion, the dialogue is marked as finished. This action expects the person to be either unconscious or confused. If the person is conscious, the action fails, and the `emergency` is set to true, triggering a replan, to lead the person outside the building or wherever the FR are.

Report

The `report` action involves compiling all collected information by the robot `r` about the person `p` at the location `l` and writing it to local storage, ensuring that the information is continuously updated. The preconditions are as follows: `r` must be available to perform the action, the dialog with `p` must have been completed, and `r` must be located in `l`. During the action, `r` remains occupied until the task is finished. Upon completion, `p` is marked as reported.

Check

The `check` action monitors the robot's `r` battery level to determine whether exploration should be halted if the battery is insufficient to return to the exit from the location `l`. The

conditions required to execute this action are as follows: `r` must be available, be located at `l`, and have the battery status marked as unchecked. The effects of this action include `r` remaining occupied until the task is completed. Upon completion, the battery is marked as checked. If the battery level is deemed insufficient to complete the task, the action fails, the `emergency` is set to true, and a replan is triggered, assuming that the battery charger is outside the building or wherever the FR are.

Check gas

The `check_gas` action enables the robot `r` to monitor the gas sensors to measure the oxygen percentage and detect the presence of harmful gases, e.g., CO_2 in the location `l`. The preconditions are as follows: `r` must be in `l` it is checking, the door must be open, and there must not be any emergency. The effect is that the gas is marked as checked in the corresponding location. It is expected that the oxygen percentage is sufficient and that no harmful gases are detected for the action to succeed. If this is not the case, the action fails, and `emergency` is set to true, triggering a replan.

Check crack

Similar to the gas-checking action, the `check_crack` action implements a visual strategy for the robot `r` to detect cracks in the location `l`. This action moves the robot to look around while running a computer vision model to detect and localize cracks. The preconditions are as follows: `r` must be in `l` and must be available; there must not be an active emergency, the room must be open, and the gas must have been checked. The effect is that the cracks are marked as checked. This action expects that no cracks are detected; if cracks are detected, the action fails, `emergency` is set to true, and a replan is triggered.

Check temperature

In this action, the robot `r` verifies that the location `l` temperature is not unsafe using thermal cameras. The preconditions are as follows: `r` must be in `l`, must be free, and must have already checked for cracks. The effect is that the environment is marked as checked in `l`, where `r` is. This action fails if the robot detects an unsafe temperature, setting `emergency` as true, triggering a replan.

Report emergency

Whenever the robot `r` is in the emergency status, the robot must report the emergency to the FRs. The preconditions are as follows: `r` must be at the exit location, or wherever FRs are, and be available. The effect is that `r` is marked as not in an emergency status, resuming the normal order of operations.

Check stairs

When approaching a staircase `s` connecting two locations `l1` and `l2`, for the first time, the robot `r` must check whether the stairs are traversable. The robot must position itself facing the stairs to visually evaluate them for the presence of debris and cracks. The preconditions are as follows: `r` must be at `l1` connected to the staircase and must be available. The effect is that the stairs are marked as checked. It is expected that the stairs are traversable; if they are not, the predicates describing the connectivity between the locations and staircases are removed from the problem file, and a replan is triggered.

Check passage

Similarly to what is done for the stairs, when the robot `r` approaches a connection between two locations `l1` and `l2` for the first time, it must visually check for the presence of obstacles, such as collapsed ceilings or furniture that might make the passage unsafe. The preconditions are as follows: `r` must be in `l1` and must be free. The effect is that the passage is marked as checked. It is expected that the passage is always traversable. If that is not the case, the connection between the two locations is removed from the problem, and a replan is triggered.

Climb stairs

The `climb_stairs` action allows the robot `r` to switch floors of the building, climbing the staircase `s`, moving from location `l1` to location `l2`. The preconditions are as follows: `s` has to be checked, `r` is in `l1`, and is free. The effects are that `r` will be free again and will be at `l2`.

Check room open

When the robot `r` reaches a location `l` for the first time, it must check whether the room is open, meaning it can safely enter, or whether the door is closed. This action completes the task by visually inspecting the door. The preconditions are as follows: `r` must be available,

be at `l`, and there must not be any active emergency. The effect is that the door is marked as checked. It is expected that the door is open. If the action fails, the door is marked as closed in the problem file, and the assertion of a predicate different from the expected effects triggers replanning.

Open door

Whenever the robot `r` is in front of a closed door at location `l`, it tries to open it using the arm mounted on its back. The preconditions are as follows: `r` must be available and at `l`. The effects are that the door is opened, and `r` can enter `l` to complete the necessary operations. If the action fails, the door is marked as blocked in the problem file, meaning that the room is inaccessible, and a replan is triggered. When the replan is triggered, the environmental check in the room is also removed from the goal.

Search outside

By analyzing thermal camera images, this action allows the robot `r` to detect the presence of people even from outside a location `l`. For this reason, if the door is blocked, this action checks for the presence of people outside. The preconditions are as follows: `r` must be available at `l` of the blocked door, and there must not be any emergency. This action expects that no person will be found. If the robot detects a person, the `emergency` is asserted to ensure that the robot communicates to the FRs the presence of a person who cannot be evaluated, and this triggers a replan.

10.1.3 System architecture

Figure 10.1 shows the complete architecture for the framework proposed. The software architecture, including both the complete planning system and the action implementation on the Spot robot, is available on Github[1]. The nodes highlighted in purple were created specifically for this project, whereas the blue nodes represent modules provided by ROS2 [135, 163]. The yellow node represents the controller node, which processes information from various actions and updates the PDDL problem based on current data. This section presents and explains the ROS2 implementation of all actions in detail.

The ZED2 camera is connected to an NVIDIA processor outside of the Spot CORE, due to GPU requirements, as previously detailed in 4. A node named Camera Information Client

[1]https://github.com/ZoeBetta/PlannerSAR.git

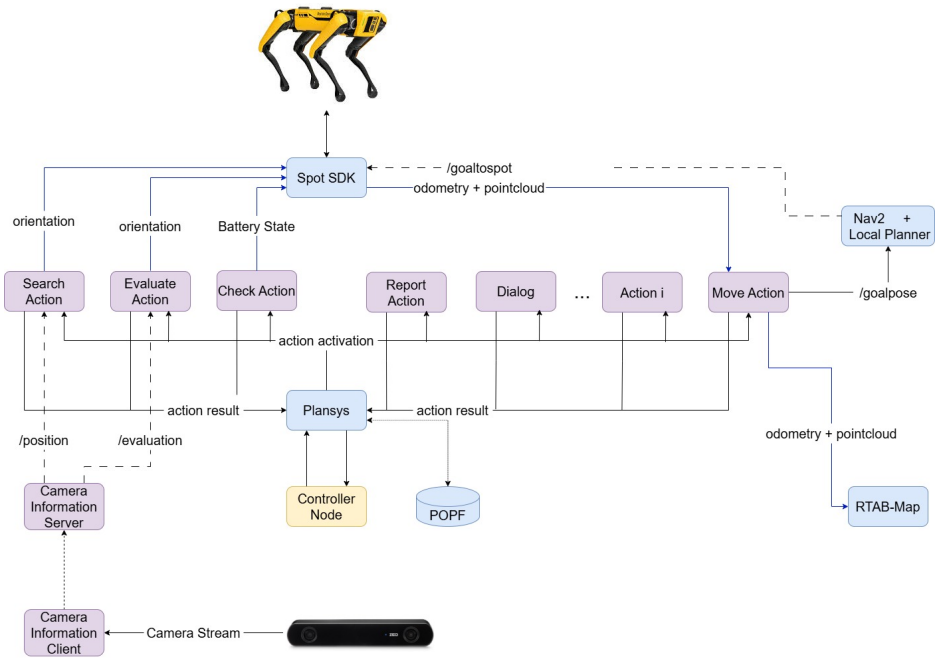


Figure 10.1 System's architecture for the experiments with the Spot robot with a limited number of actions.

was implemented on this external processor, and a TCP socket connection was established with the nodes integrated within the Spot CORE. Except for this node, all other nodes run inside a Docker container that operates Ubuntu 20.04 with ROS2 installed, enabling seamless portability to other Spot COREs if necessary.

Solid blue arrows represent the acquisition of information through the Spot SDK. In particular, the battery status, the odometry, and the point cloud from the connected LiDAR are retrieved. Solid black arrows connecting the Planner to actions represent the activation of actions when necessary or upon the completion of previous actions in the plan. Conversely, the arrows from each action to the Planner indicate feedback provided during execution and the final outcome upon completion. Dashed arrows represent publish/subscribe communication, with the associated topics specified for each arrow.

The Controller Node, the Camera Information Client Node, and the other ROS2 nodes that do not directly implement a PDDL action (i.e., Nav2, RTAB-Map, and Spot SDK nodes) are described below in greater detail.

Controller Node

Shown in yellow in Figure 10.1 is the Controller Node. Its objective is to manage the communication between the ROS2 actions and the Planner based on PlanSys [133]. PlanSys

is a PDDL planning framework integrated into ROS, using the default POPF [36] plan solver. PlanSys consists of multiple nodes: whenever the Planner node needs to generate a plan, it calls the Domain Expert and Problem Expert nodes to provide the necessary domain and problem information, respectively. Given the PDDL domain and problem, PlanSys, using the POPF planning algorithm, is used to generate a coherent sequence of actions based on the current state of the environment. However, the initially created plan may become obsolete at any moment, typically due to sensor readings that may cause the assertion of predicates that do not match the expected effects (Remark 12), thus requiring replanning to develop a plan better suited to the current situation. The "controller" node is responsible for plan management: once a plan is generated, it monitors action execution via action feedback (provided by PlanSys), updates the problem when actions fail, and manages replanning requests.

For instance, whenever a person is detected, the Controller adds an object of type "person" for each individual found and asserts a predicate that links each person to their respective location, retrieved from the information stored in the problem, in order to maintain this information. Next, replanning updates the goal to include predicates for evaluating, reporting, and engaging in dialogue with each detected person. Finally, the Controller Node resumes execution.

Camera Information Client

On board the ZED2 camera processor, the SteroLabs library is used for body detection and tracking. For each detected individual, a message containing the following information is sent to the Camera Information Server: ID, confidence level, action state, keypoints, keypoint confidence scores, and the positions of each keypoint. The received data is split into two parts. The first part, which includes the position, confidence, and ID of the detected person, is published to the `/position` topic. The second part, which contains details such as confidence, action state, keypoints, keypoint confidence scores, and positions of all keypoints, is published to the `/evaluation` topic. Both topics are shown in Figure 10.1 with dashed arrows. This information is then leveraged by the `search` and `evaluate` actions to collect data about people in the environment.

Additional ROS2 Nodes

The ROS2 framework has different modules already available for quick development. The following modules have been used to implement actions on the Spot robot for a case study. In the following section, the different modules used are briefly described:

- Nav2 [130], the navigation stack embedded in ROS2, enabling mobile robots to navigate complex environments through various functionalities, including moving from one point to another, adding intermediate poses, and performing tasks such as object following, localization, mapping, path planning, path following, and obstacle avoidance. Nav2 typically relies on a Global Planner, which determines an optimal path from the start to the goal position, and a Local Planner, which manages real-time obstacle avoidance and path execution, ensuring the robot follows the global plan while reacting to dynamic obstacles. We used its capabilities for the `move` action.
- RTAB-Map [115], a graph-based SLAM approach designed for RGB-D, stereo, and LiDAR sensors, providing loop closure detection through a bag-of-words technique to assess whether a new image corresponds to a previously visited location or a new one. Upon confirming a loop closure hypothesis, a new constraint is added to the graph, and a graph optimizer minimizes mapping errors. It was used to determine the position of the robot in the environment, and to ensure safe navigation during the `move` action.
- The Spot SDK, a library to retrieve the sensor information and send motion commands to the robot. In particular, the `check` action, was dependent on the information retrieved about the battery state from the Spot SDK.

10.2 Experiments

Three different sets of experiments will be detailed in the following sections. The first set of experiments analyzes case studies (two of which are discussed in detail) to evaluate the replanning capabilities of the proposed ROS integration by simulating challenging scenarios and recording how the robot reacts, updating the plan accordingly.

The second set of experiments aims to formally study the limits of the PDDL planning in the given domain. In particular, how planning times and success rates are influenced by variations in the number of assertions in the initial problems (i.e., the breadth and complexity of the environment), as well as by other parameters.

Finally, the last case study aims to verify, on a simplified domain, the implementation of the system on the Spot robot. It considers simple SAR scenarios in which the robot plans the exploration of a partially known environment while locating people.

Every experiment was run in a Docker container running Ubuntu 22.04 and ROS2 Humble. For the first and second sets of experiments, the Docker container ran on an AMD Ryzen 9 6900HX with Radeon Graphics processor and 32GB of RAM. In the third set of experiments, the Docker container was deployed on the SpotCore, which has an Intel® Core™ i5 8th Gen (Whiskey Lake-U) processor and 16GB of RAM.

10.2.1 Simulated Actions

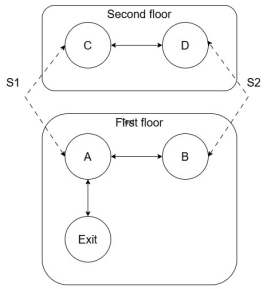
To analyze the system's response to unpredictability, we thoroughly tested it by randomly activating action failures. In the following section, two runs exhibiting particularly interesting behavior are presented as examples.

First simulated experiment

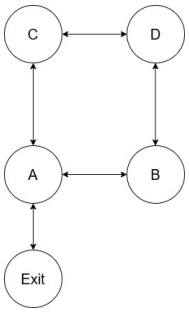
In the first simulated experiment, an environment consisting of four rooms distributed across two different floors is presented. On the first floor are rooms `a` and `b`, while on the second floor are rooms `c` and `d`. The two floors are connected by two staircases: `s1` connects rooms `a` and `c`, and `s2` connects rooms `b` and `d`. Additionally, the `exit` location is connected to room `a` and contains the battery needed to recharge the robot. The environment is represented in Figure 10.2a, where the arrows represent passages between rooms on the same floor, while the dashed arrows represent staircases. The initial state of the environment is reported in Listing 10.1. The goal at the beginning of the operations is reported in Listing 10.2. The robot has to search for people in rooms `a, b,c,d`; additionally, it has to complete all environmental checks in the same rooms.

Listing 10.1 Initial State Simulation 1

```
(robot_at spot a)
(connected exit a)
(connected a exit)
(connected a b)
(connected b a)
(connected c d)
(connected d c)
(stairs_connected a s1)
```



(a) Environment for the first simulated experiment.



(b) Environment for the second simulated experiment.

Figure 10.2 Simulated environments.

```
(stairs_connected c s1)
(stairs_connected b s2)
(stairs_connected d s2)
(not_emergency spot)
(is_free spot)
(battery_unchecked spot)
(door_notchecked a)
(door_notchecked b)
(door_notchecked c)
(door_notchecked d)
```

Listing 10.2 Goal Simulation 1

```
( :goal( and
( searched a )
( searched b )
( searched c )
( searched d )
( environment_checked a )
( environment_checked b )
( environment_checked c )
( environment_checked d )))
```

The planner is able to generate the plan in 429.15 milliseconds; the plan is reported in Listing 10.3.

Listing 10.3 Plan Simulation 1.1

```
(check spot a)
(checkstairs spot s1 a c)
```

```
(checkpassage spot a b)
(checkroomopen spot a)
(checkgas spot a)
(checkcrack spot a)
(checktemp spot a)
(search spot a)
(check spot a)
(climbstairs spot a c s1)
...
(checktemp spot b)
```

The plan is not reported in its entirety for the sake of brevity. Note that, initially, the robot checks its battery status `check spot a)`, the availability of stairs `s1` (`checkstairs spot s1 a c`), the passage between rooms `a` and `b` (`checkpassage spot a b`), the presence of gas, cracks, and the temperature in room `a`, and finally the presence of people. After the environmental checks, and the search for people, the battery is checked once again, since in principle some time has elapsed, requiring monitoring. However, during the execution of the action `(checkstairs spot s1 a c)`, the action fails, indicating that staircase `s1`, which connects `a` and `c`, is not traversable. The problem is updated to remove the predicates `(stairs_connected a s1)` and `(stairs_connected c s1)`, and a replan is triggered. The system is able to find an alternative plan in 286 milliseconds. The new plan is reported in Listing 10.4.

Listing 10.4 Plan Simulation 1.2

```
...
(search spot a)
(check spot a)
(move spot a b)
(checkstairs spot s2 b d)
(check spot b)
(checkroomopen spot b)
(checkgas spot b)
(checkcrack spot b)
(checktemp spot b)
(search spot b)
(climbstairs spot b d s2)
(checkpassage spot d c)
(check spot d)
```

```
...
(search spot d)
```

The system is able to adapt, the plan is invariated in the evaluation of room a, it then explores the first floor by moving from a to b (`move spot a b`). then checking stairs s2 (`checkstairs spot s2 b d`), and finaly moving to the second floor using the alternate staircase (`climbstairs spot b d s2`). After climbing the stairs, while checking the battery levels in room d (`check spot d`), a low battery level is detected, triggering the emergency state and a subsequent replan. The replan is completed in 254 milliseconds, and the generated plan is reported in Listing 10.5.

Listing 10.5 Plan Simulation 1.3

```
...
(reportemergency spot exit)
...
(search spot d)
```

The robot first moves to the exit location, first climbing down the s2 staircases, and then from b to a, to recharge its batteries. After that, it resumes the sequence of operations, searching for people and evaluating environmental conditions in rooms d and c.

Second simulated experiment

The second run, in contrast, has four rooms a, b, c, and d, all situated on the same floor. Room a is connected to rooms b and c, while room d is connected to rooms b and c. The exit location, where FRs and the charger are, is connected to location a. The environment is shown in Figure 10.2b, and the initial state is described in Listing 10.6. The goal is the same as in the first simulated experiment, reported in Listing 10.2. The first plan is generated in 339 milliseconds and is reported in Listing 10.7.

Listing 10.6 Initial State Simulation 2

```
(robot_at spot a)
(connected exit a)
(connected a exit)
(connected a b)
(connected b a)
(connected a c)
(connected c a)
```

```
(connected c d)
(connected d c)
(connected b d)
(connected d b)
(not_emergency spot)
(is_free spot)
(battery_unchecked spot)
(door_notchecked a)
(door_notchecked b)
(door_notchecked c)
(door_notchecked d)
```

Listing 10.7 Plan Simulation 2.1

```
(check spot a)
(checkpassage spot a b)
(checkpassage spot a c)
(checkroomopen spot a)
(search spot a)
(check spot a)
(checkgas spot a)
(checkcrack spot a)
(checktemp spot a)
(move spot a b)
...
(checktemp spot c)
```

According to the plan, the robot initially checks its battery status and the links between `a` and the adjecent rooms, then searches for people and potential issues such as gas leaks, cracks, or abnormal temperatures, before moving to room `b`. While checking whether the passage connecting rooms `a` and `b` is traversable using the action `(checkpassage spot a b)`, the action fails, causing the robot to remove the predicate `(connected a b)` from the problem and triggering a replan. The replan took 273 milliseconds, and the new plan is reported in Listing 10.8.

Listing 10.8 Plan Simulation 2.2

```
(checkpassage spot a c)
(checkroomopen spot a)
(search spot a)
```

```
(check spot a)
(checkgas spot a)
(checkcrack spot a)
(checktemp spot a)
(move spot a c)
...
(move spot c d)
...
(move spot d b)
(checkroomopen spot b)
...
(checktemp spot b)
```

The robot changes the sequence in which it visits the rooms. Instead of moving to room `b` first, it searches rooms `c` and `d` initially. Upon reaching room `b`, the action that checks whether the door is open, `(checkroomopen spot b)`, fails, indicating that the door is closed. The problem is updated by adding the predicate `(door closed b)`, which triggers a replan completed in 238 milliseconds. The new plan is reported in Listing 10.9.

Listing 10.9 Plan Simulation 2.3

```
(opendoor spot b)
(search spot b)
(checkgas spot b)
(checkcrack spot b)
(checktemp spot b)
```

In this new plan, before completing the operations of searching for people and evaluating environmental conditions, the robot must first open the door using the action `(opendoor spot b)`. Since no further actions fail, the plan is completed.

10.2.2 Planning Evaluation

The second set of experiments aims to evaluate the behaviour of the planner by analyzing how different parameters used to generate problems randomly influence the resulting solutions. In particular, the goal is to identify potential limitations in the environments that the planning system can handle. For each generated problem, the *planning time* is recorded for successfully found plans, along with the planner's ability to find a solution.

To perform a thorough analysis, we first identified the environmental parameters required to generate a problem file. These parameters are listed in Table 10.3: the first column reports the parameter name; the second column provides a description of each parameter; the third column enumerates the values considered during the experiments; and, finally, the fourth column reports the reference value used for all parameters that are not under investigation in the current set of experiments. Parameters p_3 and p_4 influence the initial state of the robot rather than the environment itself. For this reason, they were randomly assigned in each trial, and their impact on planning time was not analyzed. Since the goal of the analysis was to study planning times, the parameter intervals were selected to avoid generating too many configurations in which the goal would be unreachable. In the following analysis, when referring to planning times, we consider only those for which the planner was able to successfully generate a plan. The planner was unable to generate a plan for two possible reasons: either the plan was not solvable, or the resolution required too many resources, such as RAM, resulting in a timeout.

For the remaining parameters, we investigated their impact on planning performance by varying one parameter at a time, selecting values from the third column of Table 10.3 while keeping all other parameters fixed at the mid value reported in the fourth column, in order to avoid a combinatorial explosion of test cases.

To analyze the effect of each parameter $p_i \in p_1, p_2, p_5, p_6, p_7, p_8, p_9, p_{10}$, we generated $N = 550$ problems for each possible value of p_i, for a total of 17600 runs. For each parameter, the specific instances of blocked links and stairs, closed or blocked doors, and the locations of people and rooms already visited were randomly selected.

This procedure could produce configurations for which no valid plan exists. For example, if all connections to a room are blocked, the room becomes unreachable, and the robot cannot satisfy the goal of searching the room and evaluating the environment.

After generating the problems, we used the planner POPF [40] with the default parameters; this is the same planner and the same parameters used by PlanSys to generate the plan. For each problem and its corresponding generated plan, we recorded the number of predicates in the problem file, the number of goal predicates to satisfy, the length of the generated plan, and the planning times.

10.2.3 Case study

The experiments involving the Spot robot were conducted on the second floor of the E Building at Via dell'Opera Pia 13, University of Genoa. These experiments took place

Param	Description	Values	Mid value
p_1	Number of floors	1, 2, 3, 4	2
p_2	Rooms per floor	4, 9, 16, 25	9
p_3	Initial position	any room	–
p_4	Emergency at start	yes, no	–
p_5	Blocked passages (%)	0, 5, 10, 15	10
p_6	Blocked stairs	0, 1, 2	1
p_7	Closed doors (%)	0, 10, 20, 30	10
p_8	Blocked doors (%)	0, 5, 10, 15	5
p_9	Number of people	0, 5, 10, 15, 20	10
p_{10}	Rooms visited (%)	0, 10, 20, 30	20

Table 10.3 Parameters used for problem generation.

in multiple locations, including a corridor and two separate rooms, to replicate a realistic environment for the robot's operation.

Only a small subset of actions was implemented in ROS2 and subsequently on the Spot robot. Specifically, as shown in Fig. 10.1, we implemented the `search` and `evaluate` actions to locate people in need of assistance and assess their health status; the `check` action to monitor the battery state; the `report` action to communicate results to FRs; and the `move` action to navigate within the environment. Other actions included in the planning domain were not implemented in this case study, either because the required sensors (e.g., a gas sensor) or actuation capabilities (e.g., an arm to open doors) were not available at the time the experiments were conducted, or because the experimental environment did not allow them to be properly used (e.g., simulating gas leaks in our department is not feasible).

The experiments were conducted using the Spot robot, equipped with the Spot CORE, a LiDAR, and a ZED2 camera. To achieve the desired behavior, various adjustments were made during the code development process, including fine-tuning thresholds for people detection and pose estimation. Multiple tests were performed to ensure the system's reliability by defining locations within the environment, which were then used to initialize the PDDL problem (Figure 10.3).

The PDDL initial state definition for this case study is presented in Listing 10.10.

Listing 10.10 Problem Real actions

```
(robot_at spot exit)
(connected exit frontlab2)
(connected frontlab2 exit)
(connected centerlab2 frontlab2)
(connected frontlab2 centerlab2)
```

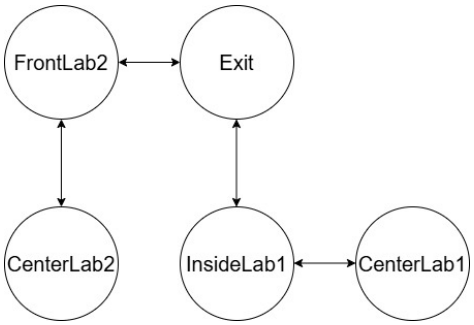


Figure 10.3 Real scenario on the second floor of the E building.

```
(connected exit insidelab1)
(connected insidelab1 exit)
(connected centerlab1 insidelab1)
(connected insidelab1 centerlab1)
(connected exit battery_location)
(connected battery_location exit)
(is_free spot)
(battery_unchecked spot)
```

For this example of application, in line with the problem's objectives, the robot will only search at the center of the rooms to locate people.

According to the domain definition, at each location, the robot will check its battery status and decide whether to continue with the plan or return to recharge. The `battery_location` is treated as equivalent to the `exit` location, corresponding to the building entrance, where the robot will return after finding victims to report its findings to the rescuers.

The goal is reported in Listing 10.11.

Listing 10.11 Goal 1 Case Study

```
(:goal (and (searched spot CenterLab1)
            (searched spot CenterLab2)))
```

In Figure 10.4, some snapshots from the operations of the Spot robot completing a possible SAR task in a simplified environment are shown. In Figure 10.4a, the Spot robot is moving from the starting location where the FRs are coordinating operations, `exit`, to the location `FrontLab2`, used as a waypoint. After moving to the location `CenterLab2`, the robot rotates in place to locate people in the environment with the mounted ZED2 camera, as shown in Figure 10.4b. Once the person is detected, the problem is updated to add the instance of the person in the correct location, and the goal is changed, adding the need to evaluate the person; the updated goal is shown in Listing 10.12.



(a) Moving to the first waypoint.



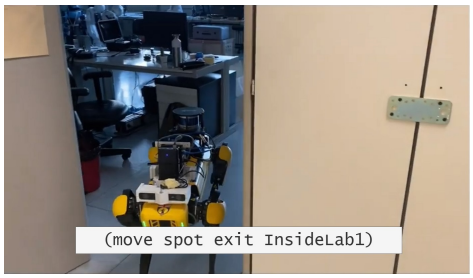
(b) Rotating to locate people.



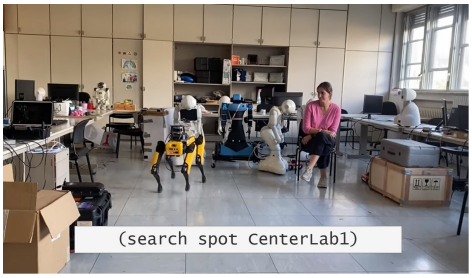
(c) Evaluating the position of the person.



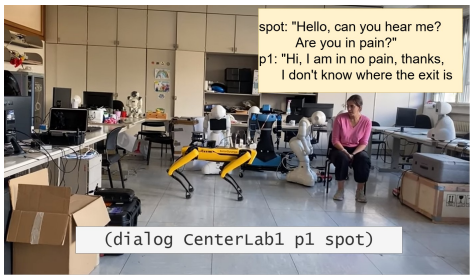
(d) Assessing response to the call.



(e) Moving to the second location.



(f) Rotating to locate people.



(g) Assessing response to the call.



(h) Leading the person to the exit.

Figure 10.4 Frames from the implementation of a subset of actions on the Spot robot, simulating a rescue operation.

Listing 10.12 Goal 2 Case Study

```
(:goal (and (searched spot CenterLab1)
            (searched spot CenterLab2)
            (person_evaluated p0)
```

```
        (person_reported p0)
        (dialog_finished p0)))
```

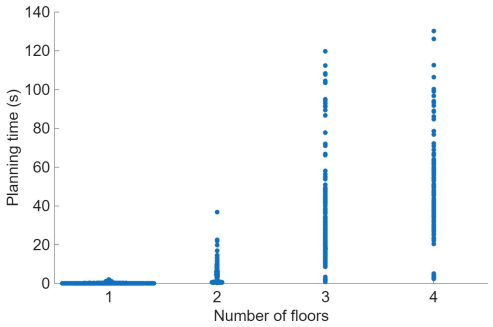
Updating the problem and the goal, the robot has to replan its next actions. The next action becomes `(evaluate CenterLab2 p0 spot)`, where the robot positions itself to face the person and detects the posture (standing, sitting, or lying down), as shown in Figure 10.4c. After the posture of the person is evaluated, the robot completes a dialog to assess the response to the call, as shown in Figure 10.4d; in this case, the person does not answer and is therefore marked as unconscious by the robot. Since the person is unconscious, the robot can proceed with the plan, moving towards `CenterLab1`. In Figure 10.4e, the robot is shown completing the action `(move spot exit InsideLab1)`. Once the robot reaches the center of the lab, it rotates in place again to look for people, as shown in Figure 10.4f. Also in this case, a person is detected, a replan is triggered, and the goals for evaluating the condition of the person are added. The robot rotates to face the person and evaluates the posture, after which it completes the `(dialog CenterLab1 p1 spot)` action to assess the response to the call. In this case (Figure 10.4g), the person answers the robot; therefore, a replan is triggered to lead the person towards the exit location. In Figure 10.4h, it is possible to see the robot completing the `move` action while the person is following it.

10.3 Results

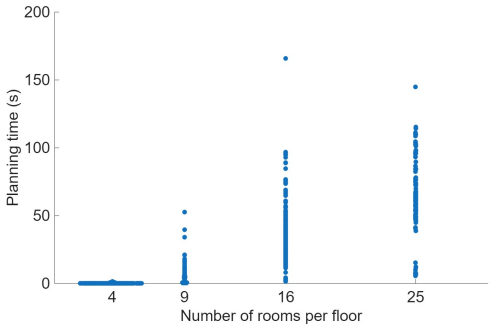
10.3.1 Planning evaluation

In Table 10.4, the parameters are reported in the first column, with the possible values they can take; in the second column, the percentages of success for each parameter value; in the third column, the percentages of plans that were unsolvable; and finally, in the fourth column, the percentage of times that planning was halted due to insufficient resources.

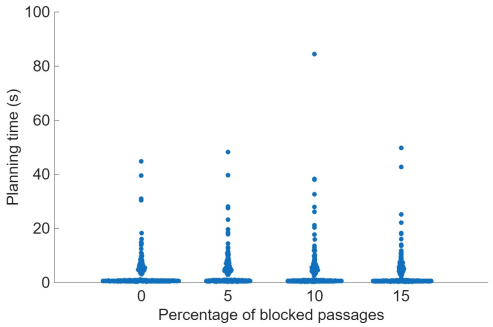
While changing p_1, the number of floors in the environment, we recorded the planning times shown in Figure 10.5a. It is clear that as the number of floors increases, the planning time also increases, reaching peaks of 120 seconds. This suggests a correlation between the number of floors and the computational effort required by the planner (by keeping the number of rooms per floor constant to 9). This observation is further confirmed by the number of plans that failed due to resources errors: when only one floor is present, no plan failed; with two floors, only one plan failed out of 550; with three floors, 36 plans failed out of 550; and with four floors, 161 plans failed out of 550.



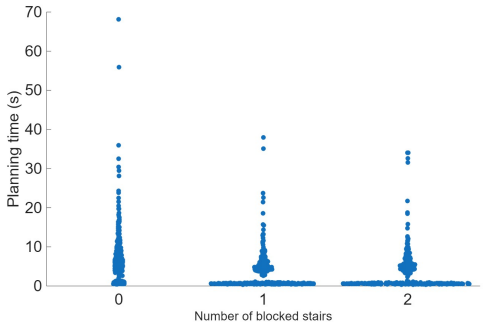
(a) Planning times changing the number of floors p_1.



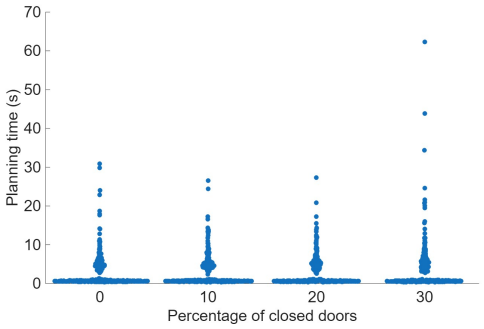
(b) Planning times changing the number of rooms on each floor p_2.



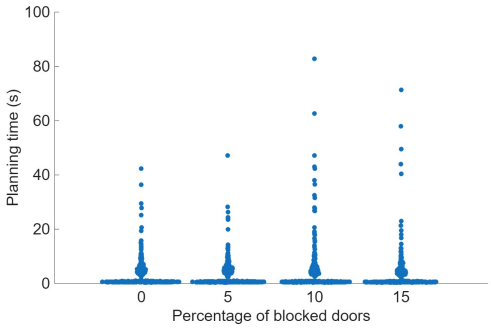
(c) Planning times changing the percentage of blocked passages p_5.



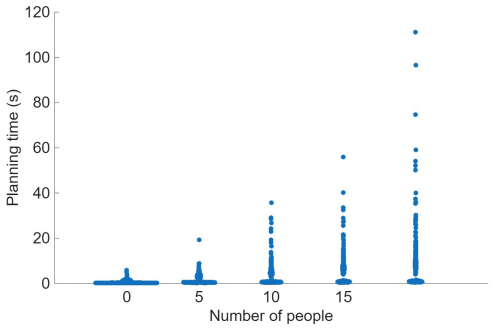
(d) Planning times changing the number of blocked stairs p_6.



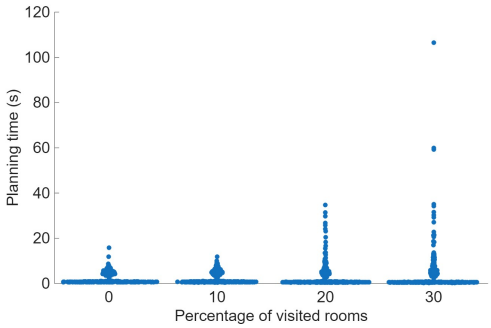
(e) Planning times changing the percentage of closed doors p_7.



(f) Planning times changing the percentage of blocked doors p_8.



(g) Planning times changing the number of people p_9.



(h) Planning times changing the percentage of visited rooms p_{10}.

Figure 10.5 Planning time analysis under different environment configurations.

A similar behavior can be observed in Figure 10.5b. In this case, increasing the number of rooms on each floor also increases the planning times. Similarly, the number of plans that failed due to timeouts rises as the number of rooms per floor increases. For 4 and 9 rooms, no timeouts were recorded. However, with 16 rooms per floor, the planner failed 41 times out of 550, and with 25 rooms per floor, it returned a resource error on 258 occasions out of 550.

In contrast, varying the percentage of blocked passages (e.g., corridors blocked by debris that the robot cannot traverse) does not impact planning times. In Figure 10.5c, the planning times are shown as a function of the percentage of blocked passages. It is clear that the majority of the points are below 20 seconds for each configuration. A similar behavior can be observed in Figure 10.5d, where the x-axis represents the number of blocked staircases. For this parameter as well, the planning times are not influenced by the number of blocked stairs.

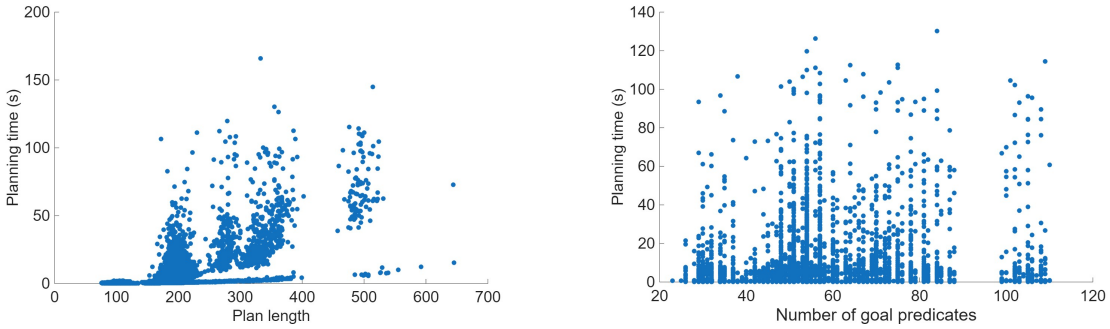
When examining parameters p_7 and p_8, the results are presented in Figure 10.5e and Figure 10.5f, respectively. It is evident that, in these cases as well, the planning times remain under 20 seconds for all configurations, indicating that the number of closed and blocked doors does not affect the planner's performance. We can conclude that changing the connectivity of the system does not significantly affect planning times.

In Figure 10.5g, the planning times are shown for different numbers of people in the environment. The planning time increases as the number of people increases. This can be explained by the fact that adding a person to the environment generally adds three actions to be completed: evaluation, dialog, and reporting. It is worth noting that, although the number of people influences the planning time, it does not affect the number of timeouts due to resource limitations, which remain below five out of 550 for all possible configurations.

Finally, Figure 10.5h shows the planning times as a function of the percentage of already visited rooms, p_{10}. The planning time slightly increases as this percentage rises. This may be because, if a room has already been visited, it does not need to be explored again. Since the planner is trying to optimize the plan, it searches for alternative paths that prioritize unvisited rooms, making the planning process more complex, even though the number of goals decreases.

We also analyze the performance of the planner as a function of plan length. In Figure 10.6a, the x-axis represents the length of the generated plan, while the y-axis represents the time required to generate it. In general, as expected, the planning time increases with plan length; longer plans tend to require longer planning times.

Similarly, we wanted to examine whether the number of goal predicates to be satisfied affects the planning time. Figure 10.6b presents the planning times as a function of the number of goal predicates.



(a) Planning times as a function of the plan action length.

(b) Planning times depending on the number of goal predicates to be satisfied.

Figure 10.6 Planning performance under different complexity factors.

Param	Success (%)	Unsolvable (%)	Insufficient computing resources (%)
$p_1 \in \{1,2,3,4\}$	100.0, 70.4, 61.1, 29.6	0, 29.4, 32.4, 41.1	0, 0.2, 6.5, 29.3
$p_2 \in \{4,9,16,25\}$	73.6, 74.2, 65.1, 19.5	26.4, 25.8, 27.4, 33.6	0, 0, 7.5, 46.9
$p_5 \in \{0,5,10,15\}$	78.2, 74.9, 70.6, 67.6	21.8, 25.1, 28.7, 31.8	0, 0, 0.7, 0.6
$p_6 \in \{0,1,2\}$	68.2, 70.4, 69.8	31.1, 29.3, 29.5	0.7, 0.4, 0.7
$p_7 \in \{0,10,20,30\}$	72.5, 72.4, 73.6, 75.8	27.3, 26.9, 26.4, 23.5	0.2, 0.7, 0, 0.7
$p_8 \in \{0,5,10,15\}$	72.5, 72.4, 73.3, 73.4	27.5, 27.1, 26.2, 26.2	0, 0.5, 0.5, 0.4
$p_9 \in \{0,5,10,15,20\}$	70.5, 74.7, 69.3, 73.8, 71.6364	29.5, 25.3, 30.4, 25.5, 28.1818	0, 0, 0.3, 0.7, 0.1818
$p_{10} \in \{0,10,20,30\}$	71.6, 73.6, 69.5, 73.3	28.4, 26.0, 30.5, 25.4	0, 0.4, 0, 1.3

Table 10.4 Success percentages for each parameter.

10.4 Discussion

In the second set of experiments, we sought to study the effect of each parameter describing the environment on planning times. From the results obtained, the number of floors and the number of rooms per floor strongly influence both the planning times and the resources needed to generate a plan, with a consequent decrease in the success rate. This could pose a problem for larger public buildings, such as hospitals, which typically have many floors and a high number of rooms per floor. A possible solution would be to describe the space in a simpler way, aggregating different rooms into areas of the building. Another parameter that strongly influences planning times is the number of people in the environment. This can be explained by the fact that adding a person generally introduces three actions to the plan. However, this increase in planning time is not necessarily associated with a lower success rate. A parameter that slightly influences planning time is the percentage of already visited rooms. The planning times increase, but not as significantly as for the other parameters mentioned previously. The remaining parameters—the percentage of blocked passages, the number of blocked stairs, the percentage of closed doors, and the percentage of blocked doors—do not affect planning times.

As a confirmation, we examined whether there is a relationship between planning times and the length of the generated plan. It can be observed that as the length of the plan increases, the planner's time also increases. In particular, for plans longer than 500 actions, the lowest planning time is on the order of ten seconds, whereas for plans of about 100 actions, planning times are almost all on the order of one second.

The simulations and actual tests with the Spot robot demonstrate the robot's ability to complete a set of tasks, even in the presence of unpredictability, where actions might fail.

10.5 Conclusion

This chapter presented a system architecture and planning framework tailored to SAR scenarios, in which a robot is required to execute tasks derived from the needs and requirements identified with first responders in Chapter 3.

The proposed planning domain was extensively evaluated using simulated problem instances with an increasing number of asserted predicates to investigate the scalability limits of the system. In particular, the relationship between the complexity of the environment description and planning time was analyzed. As expected, a clear correlation was observed between planning time and environmental complexity: larger environments with more floors, rooms, and people lead to increased computation time. In contrast, the presence of blocked passages, doors, or stairs does not significantly affect planning performance.

The system was further validated in real-world experiments using the Boston Dynamics Spot robot in our laboratory setting, where the action set was reduced for implementation and logistical constraints.

Both simulation and real-world results demonstrate the feasibility of the proposed approach, as well as its robustness in handling dynamic and unpredictable environments typical of SAR operations.

Chapter 11

Conclusion

11.1 Summary of Contributions

In conclusion, this thesis addressed the problem of autonomous inspection in hazardous industrial and SAR environments through the use of quadruped robotic platforms. The work was motivated by the need to improve worker safety by reducing direct human exposure to dangerous areas while increasing the level of autonomy in inspection operations.

Due to the specificity of the application and the need to collaborate with trained personnel, we first analyze the requirements of stakeholders. In particular, we propose three studies to collect stakeholders' opinions on robotic solutions and to gather information about the requirements of their operations. Data from FRs and harbor workers have been analyzed to propose a list of possible robotic applications for the task of robotic inspection. Following the interviews, we decided to focus more on the autonomy side of the robotic application. In particular, on autonomous exploration and detection of relevant features.

To this end, we study and propose a strategy to autonomously explore a multi-floor building, taking into account a fixed time to collect as much information as possible, and the rewards, with the associated risks in climbing the stairs. We tested the proposed approach with the quadruped robot Spot from Boston Dynamics to ensure the feasibility of the approach.

Subsequently, we propose a strategy to visually detect and localize in the environment relevant features. This strategy is based on the concept of probabilistic grid map representations, where each cell holds the probability of the cell being occupied by a feature. The detection system takes into account the peculiarities of inspection tasks, especially in the context of SAR. In particular, it assumes that it is more convenient to localize with a higher level of certainty and a less precise localization, assuming that if FRs are able to reach the proximity of the detection, they will be able to see it.

Finally, we propose the integration of autonomous navigation and feature detection, with particular focus on SAR applications. The experimental results demonstrate that different exploration strategies—one aiming to fully explore each floor before moving to the next, and another prioritizing floor transitions as early as possible—do not significantly affect detection performance, although they can substantially influence the extent of environment exploration and coverage.

The first part of the thesis assumes that the navigation is on mostly flat terrain, other than staircases, and that the obstacles are fixed. In the second part, we relax these assumptions and propose novel strategies to account for uneven terrains, as well as dynamic obstacles, and also to account for more complex scenarios.

In particular, we propose a solution for navigating in space that takes into account the slope the robot must face, both laterally and frontally. We thoroughly tested this system in simulation to demonstrate the effectiveness of the method and the resulting improvement in robot safety. We also validated it on a real Spot robot in a simplified scenario to verify the feasibility of the approach and its impact on navigation.

Similarly, we propose a strategy for navigation in environments with dynamic obstacles. We introduce a local planner based on APF, modulated by emotional states. We tested this system in simulation to evaluate the effect of individual emotions, as well as on the Spot robot. The robot experiments were first conducted in the absence of dynamic obstacles to assess the system's ability to reach the goal. We then tested it with structured dynamic obstacles to analyze the system's reaction and behavior. Finally, we evaluated the system in the presence of unstructured dynamic obstacles to ensure safety during navigation in more realistic conditions.

Finally, it is not always possible to assume that the robot's path is free of blocking obstacles, such as closed doors. In general, a quadruped robot can be equipped with a robotic arm and used to open doors; however, this requires a high level of task coordination. For this reason, we propose a PDDL domain tailored in detail to the requirements of FRs, capable of planning inspection tasks in multi-floor buildings. We then evaluate the feasibility of this approach and investigate whether a PDDL framework can adequately represent and plan for realistic emergency scenarios. The feasibility of this approach is also evaluated using the Spot robot with a limited set of actions.

Overall, the contributions of this thesis establish a framework for autonomous inspection using quadruped robots in hazardous and unstructured environments. The work demonstrates the feasibility of the proposed approaches under realistic operational constraints, including limited time, partial observability, uneven terrain, dynamic obstacles, and task-level reasoning.

Across all stages of the thesis, from stakeholder-driven requirement analysis to full system validation, the results consistently show that meaningful inspection autonomy can be achieved without relying on continuous human supervision, even in complex multi-floor and SAR scenarios. In particular, the proposed approaches demonstrate that exploration and feature detection can be jointly optimized in a way that remains feasible on real robotic platforms such as Spot, while preserving robustness in challenging environments. Collectively, these results provide strong evidence that quadruped robots can move beyond assisted teleoperation towards fully autonomous inspection agents capable of operating in real-world safety-critical deployments.

11.2 Limitations

This work demonstrates promising capabilities for operating in unpredictable and unstructured environments. However, several limitations remain. The primary limitation is that the proposed components have not yet been fully integrated into a single unified system. In particular, the autonomous exploration framework assumes that all areas of interest are reachable by the robot, which simplifies the handling of occlusions, blocked passages, and inaccessible regions that may occur in real operational scenarios. As a consequence, the task planning and navigation components are not yet fully integrated within the overall inspection pipeline, limiting the ability of the system to reason jointly over high-level mission constraints and low-level motion feasibility. Similarly, although the Spot robot is capable of climbing stairs, the development and integration of a fully autonomous stair-climbing strategy was not considered a priority within the scope of this thesis, and vertical navigation remains partially dependent on simplified assumptions.

Additionally, the validation of the proposed systems on the real robot remains limited. Experimental evaluation with physical robotic platforms is inherently time-consuming and often requires restricting access to operational areas, which can be impractical in active working environments. Furthermore, the simulation environment only partially captures the unpredictability of real-world conditions, including dynamic changes in the environment, occlusions, and sensor noise affecting physical robotic systems, which may influence both perception and navigation performance. Due to the inherent challenges of physical robot testing and the limitations of simulating a highly unpredictable environment, the comparison to state-of-the-art systems remains limited, in favor of a more thorough evaluation of the proposed system.

The proposed system has been evaluated exclusively using the Spot robot from Boston Dynamics, due to its mobility capabilities and adaptability to harsh terrains. However, most of the work described in this thesis could also be extended to different types of mobile robots. In particular, the work described in Chapter 5 could be implemented on both UAVs and wheeled robots, with only minor modifications. More substantial adaptations would be required for the work described in Chapters 8 and 9, where navigation has been implemented assuming holonomic motion capabilities, which are not generally available on wheeled robots. Similarly, with regard to the work presented in Chapter 10, its implementation on different robotic platforms should be carefully studied and evaluated to ensure that the different capabilities and constraints of each robot are adequately represented within the SAR domain.

For similar reasons, the selection of system parameters was not optimised through formal fine-tuning procedures. Instead, parameters were empirically adjusted on the real robotic platform to ensure stable behaviour and practical feasibility during deployment. While this approach enabled successful demonstrations, it does not guarantee global optimality or transferability across significantly different environments.

Finally, the system was not evaluated in a real emergency scenario in direct collaboration with stakeholders. Most experiments were conducted within university facilities, in relatively structured environments and in the absence of debris, severe degradation, or the operational constraints typically encountered in real disaster response situations. As a result, the generalisation of the proposed methods to fully realistic emergency conditions remains to be further validated.

11.3 Future improvements

Future improvements should focus on developing a more stable and optimised version of the proposed protocols, with particular emphasis on robustness, validation, and real-world deployment. In particular, future work should address a more systematic and formal approach to parameter selection and tuning, moving beyond empirical adjustment towards principled optimisation methods. This includes the use of structured hyperparameter optimisation strategies and systematic sensitivity analysis to better understand the impact of individual parameters on system performance and stability. Such improvements would contribute to increased repeatability, improved generalisation across different environments, and more reliable deployment in real-world operational scenarios.

A key direction is further evaluation of system accuracy to ensure that the generated information is reliable and trusted by end users. Results from the online questionnaire indicate that FRs expect approximately 68% accuracy to consider the robot useful, and around 75% accuracy to trust it during operational use. Experimental results from the integration of autonomous exploration and semantic mapping show detection performance exceeding 80%, suggesting that the system may meet or surpass the expected usability threshold. However, these assumptions require further validation in collaboration with stakeholders to confirm whether the generated outputs are indeed perceived as reliable and actionable in practice. For this reason, additional user studies and field trials are necessary to better assess system usability and trust in operational contexts.

Future work should also extend experimental validation to more challenging and unstructured environments, including dedicated field sites such as the Civil Protection camp in Foligno, in order to evaluate system behaviour under realistic and degraded conditions. In particular, testing should include scenarios with uneven terrain, cluttered and partially obstructed spaces, and varying levels of visibility and accessibility, as typically encountered in real emergency response operations. This would enable a more comprehensive assessment of system robustness, including navigation reliability, perception performance, and overall mission success under non-ideal conditions. Such extended validation would also provide valuable feedback for refining both the autonomy stack and the perception modules, ensuring better transferability from controlled environments to real-world deployments. Additionally, it is necessary to extend the validation and testing to include comparisons with state-of-the-art algorithms, to better understand the effect of the proposed system and its possible limitations.

Finally, the problem of how information is communicated to workers has not yet been investigated in this thesis. At present, it is assumed that the information generated by the system is presented through an external graphical user interface (GUI) and made readily accessible to operators. However, in high-stress and time-critical scenarios, such as emergency response operations, the design of information delivery becomes a crucial factor. In particular, it is necessary to investigate effective strategies for presenting information in a way that is both rapidly interpretable and operationally meaningful, without increasing the cognitive load on workers. Future work should therefore explore human–robot interaction and information visualisation approaches that prioritise clarity, SA, and decision support, potentially including multimodal interfaces and context-aware information filtering to better support FRs in demanding environments.

References

[1] (2024a). Banca dati statistica.

[2] (2024b). Spot Makes Austria's Largest Power Plant Safer.

[3] Adiuku, N., Avdelidis, N. P., Tang, G., and Plastropoulos, A. (2024). Improved hybrid model for obstacle detection and avoidance in robot operating system framework (rapidly exploring random tree and dynamic windows approach). *Sensors*, 24(7).

[4] Agishev, R., Petricek, T., and Zimmermann, K. (2022). Trajectory Optimization Using Learned Robot-Terrain Interaction Model in Exploration of Large Subterranean Environments. *IEEE Robot. Autom. Lett.*, 7(2):3365–3371.

[5] André, C. and Lelord, F. (2001). *The Power of Emotions: Love, Anger, Joy*. Odile Jacob.

[6] Arai, M., Tanaka, Y., Hirose, S., Kuwahara, H., and Tsukui, S. (2008). Development of "souryu-iv" and "souryu-v:" serially connected crawler vehicles for in-rubble searching operations. *Journal of Field Robotics*, 25(1-2):31–65.

[7] Arduengo, M., Torras, C., and Sentis, L. (2021). Robust and adaptive door operation with a mobile robot. *Intelligent Service Robotics*.

[8] Arkin, R. C. (1998). *An Behavior-based Robotics*. MIT Press, Cambridge, MA, USA, 1st edition.

[9] Arkin, R. C. and Ulam, P. (2009). An ethical adaptor: Behavioral modification derived from moral emotions. In *2009 IEEE International Symposium on Computational Intelligence in Robotics and Automation - (CIRA)*, pages 381–387.

[10] Arthur, D. and Vassilvitskii, S. (2007). K-Means++: The Advantages of Careful Seeding. In *SODA '07*, volume 8, pages 1027–1035.

[11] Avinaash, M., Andrew, D., Visnu, S., Abrar, M., and Seenu, N. (2021). Design and simulation of disaster mitigation robot using machine vision. *J. Phys.: Conf. Ser.*, 2115(1):012009.

[12] Azpurua, H., Campos, M. F. M., and Macharet, D. G. (2021). Three-dimensional Terrain Aware Autonomous Exploration for Subterranean and Confined Spaces. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2443–2449, Xi'an, China. IEEE.

[13] Banik, S. C., Watanabe, K., Habib, M. K., and Izumi, K. (2008). An emotion-based task sharing approach for a cooperative multiagent robotic system. In *2008 IEEE International Conference on Mechatronics and Automation*, pages 77–82.

[14] Basit, A., Pudasaini, N., and Shafique, M. (2025). ROVER: Autonomous Open-Vocabulary Object Searching in Unexplored Environments Using VLM-Driven Scene Understanding. In *2025 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8, Rome, Italy. IEEE.

[15] Benton, J., Coles, A., and Coles, A. (2012). Temporal planning with preferences and time-dependent continuous costs. In *ICAPS 2012 - Proceedings of the 22nd International Conference on Automated Planning and Scheduling*, page 2 – 10.

[16] Berclaz, J., Shahrokni, A., Fleuret, F., Ferryman, J., and Fua, P. (2009). Evaluation of probabilistic occupancy map people detection for surveillance systems. In *Proceedings of the Eleventh IEEE International Workshop on Performance Evaluation of Tracking and Surveillance*, page 55–62.

[17] Betta, Z., Corongiu, D., Recchiuto, C. T., and Sgorbissa, A. (2025). Enhancing Worker Safety in Harbors Using Quadruped Robots. Accepted to the Work in Progress for the 22nd International Conference on Ubiquitous Robots UR2025.

[18] Betta, Z., Gaudino, A., Benini, A., Recchiuto, C. T., and Sgorbissa, A. (2024a). Perceptions and Opinions of Rescuers about a Quadruped Robot in an Earthquake Scenario. In *2024 33rd IEEE International Conference on Robot and Human Interactive Communication (ROMAN)*, pages 1092–1099.

[19] Betta, Z., Gracia, I. C., Sgorbissa, A., and Recchiuto, C. T. (2026a). A quadruped robot planning actions in disaster scenarios. Under review (major revision) at Robotics and Autonomous Systems.

[20] Betta, Z., Hykin, J., Sgorbissa, A., and Murphy, R. R. (2026b). Emapf: Emotionally modulated local navigation with artificial potential fields. In *2026 18th Int. Conf. on the Simulation of Adaptive Behavior, SAB 2026*.

[21] Betta, Z., Oneto, L., Recchiuto, C. T., and Sgorbissa, A. (2026c). Exploration and semantic mapping for search and rescue with a quadruped robot in multi-floor buildings. Under review (major revision submitted) at Journal of Field Robotics, Wiley.

[22] Betta, Z., Paneri, S., Gaudino, A., Benini, A., Recchiuto, C. T., and Sgorbissa, A. (2023). Multi-floor danger and responsiveness assessment with autonomous legged robots in catastrophic scenarios. In *2023 32nd IEEE International Conference on Robot and Human Interactive Communication (RO-MAN)*, pages 1063–1070.

[23] Betta, Z., Recchiuto, C. T., and Sgorbissa, A. (2024b). People, cracks, stairs, and doors: vision-based semantic mapping with a quadruped robot supporting first responders in search & rescue. In *2024 33rd IEEE International Conference on Robot and Human Interactive Communication (ROMAN)*, pages 1878–1885.

[24] Bohlin, R. and Kavraki, L. (2000). Path planning using lazy prm. In *Proc ICRA 2000*, volume 1, pages 521–528.

[25] Bouhsain, S. A., Alami, R., and Siméon, T. (2024). Extending task and motion planning with feasibility prediction: Towards multi-robot manipulation planning of realistic objects. In *2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 10318–10325.

[26] Bouman, A., Ott, J., Kim, S.-K., Chen, K., Kochenderfer, M. J., Lopez, B., Aghamohammadi, A.-a., and Burdick, J. (2022). Adaptive Coverage Path Planning for Efficient Exploration of Unknown Environments. In *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 11916–11923.

[27] Bradley, D. M., Chang, J. K., Silver, D., Powers, M., Herman, H., Rander, P., and Stentz, A. (2015). Scene understanding for a high-mobility walking robot. In *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1144–1151.

[28] Breazeal (Ferrell), C. (1998). A motivational system for regulating human-robot interaction. In *Proceedings of the Fifteenth National/Tenth Conference on Artificial Intelligence/Innovative Applications of Artificial Intelligence*, AAAI '98/IAAI '98, pages 54–62, USA. American Association for Artificial Intelligence.

[29] Byukusenge, P. and Zhang, Y. (2022). Life Detection Based on UAVs - Thermal Images in Search and Rescue Operation. In *2022 IEEE 22nd International Conference on Communication Technology (ICCT)*, pages 1728–1731.

[30] Cadena, C., Carlone, L., Carrillo, H., Latif, Y., Scaramuzza, D., Neira, J., Reid, I., and Leonard, J. J. (2016). Past, Present, and Future of Simultaneous Localization and Mapping: Toward the Robust-Perception Age. *IEEE Trans. on Robotics*, 32(6):1309–1332.

[31] Canh, T. N., Elibol, A., Chong, N. Y., and HoangVan, X. (2023). Object-Oriented Semantic Mapping for Reliable UAVs Navigation. In *2023 12th International Conference on Control, Automation and Information Sciences (ICCAIS)*, pages 139–144.

[32] Cao, J., Chen, T., Jiang, H., Lu, G., Zhang, G., Rong, X., and Li, Y. (2023). Autonomous Locomotion Control Strategy for Quadruped Robot Based on Visual Lidar Fusion Terrain Recognition. In *2023 2nd International Conference on Automation, Robotics and Computer Engineering (ICARCE)*, pages 1–6.

[33] Carradore, M., Artioli, G., and Sarli, A. (2024). The general attitudes towards robots scale (gators): A preliminary validation of the italian version. *International Journal of Social Robotics*, 16(9):2001–2018.

[34] Cha, Y.-J., Choi, W., Suh, G., Mahmoudkhani, S., and Büyüköztürk, O. (2018). Autonomous Structural Visual Inspection Using Region-Based Deep Learning for Detecting Multiple Damage Types. *Comput.-Aided Civ. Infrastruct.*, 33(9):731 – 747.

[35] Chavdarova, T. and Fleuret, F. (2017). Deep multi-camera people detection. In *2017 16th IEEE International Conference on Machine Learning and Applications (ICMLA)*, pages 848–853.

[36] Chen, Q. and Pan, Y.-J. (2024). An Optimal Task Planning and Agent-aware Allocation Algorithm in Collaborative Tasks Combining with PDDL and POPF. arXiv: 2407.08534.

[37] Choi, J., Lee, Y., Kim, T., Jung, J., and Choi, H.-T. (2017). Development of a ROV for visual inspection of harbor structures. In *Proc. UT'17)*, pages 1–4.

[38] Chu, X., Zhang, Q., Zhou, Y., Wen, W., Li, X., and Liu, W. (2020). Locomotion planning for quadruped robot walking on lunar rough terrain. In *Proceedings of the International Astronautical Congress, IAC*, volume 2020-October. Type: Conference paper.

[39] Chung, M.-A., Zhang, Z.-X., Zhang, J.-H., Lin, C.-W., Hsu, C.-C., Hsieh, M.-C., Huang, S.-J., Chen, K.-X., and Chai, S.-Y. (2024). A Six-wheeled Rescue Robot Dog based on YOLOv7. In *2024 International Conference on Consumer Electronics - Taiwan (ICCE-Taiwan)*, pages 405–406, Taichung, Taiwan. IEEE.

[40] Coles, A., Coles, A., Fox, M., and Long, D. (2010). Forward-chaining partial-order planning. *Proceedings of the International Conference on Automated Planning and Scheduling*, 20(1):42–49.

[41] Coles, A. J., Coles, A., Fox, M., and Long, D. (2014). COLIN: Planning with Continuous Linear Numeric Change. *CoRR*, abs/1401.5857. arXiv: 1401.5857.

[42] Cruz Ulloa, C., del Cerro, J., and Barrientos, A. (2023). Mixed-reality for quadruped-robotic guidance in SAR tasks. *Journal of Computational Design and Engineering*, 10(4):1479–1489.

[43] Dadheech, P. (2022). Real-Time Survivor Detection System in SaR Missions Using Robots. *Drones*, 6:1–20.

[44] Dal Lago, U., Pistore, M., and Traverso, P. (2002). Planning with a Language for Extended Goals. In *Proc. AAAI'02*, pages 447–454.

[45] Dang, T., Tranzatto, M., Khattak, S., Mascarich, F., Alexis, K., and Hutter, M. (2020). Graph-based subterranean exploration path planning using aerial and legged robots. *Journal of Field Robotics*, 37(8):1363–1388.

[46] Darwin, C. R. (1872). *The Expression of the Emotions in Man and Animals*. John Murray, London, 1st edition.

[47] Das, S. K., Sur, A., and Hossain, M. F. (2024). Collaborative Path Planning and Target Search in Multi-Robot Systems with PID-Controlled Uniform Area Partitioning. In *2024 27th International Conference on Computer and Information Technology (ICCIT)*, pages 3325–3330, Cox's Bazar, Bangladesh. IEEE.

[48] DATACLUSTER LABS (2022). Stairs image dataset.

[49] De Luca, A., Muratore, L., and Tsagarakis, N. G. (2023). Autonomous navigation with online replanning and recovery behaviors for wheeled-legged robots using behavior trees. *IEEE Robotics and Automation Letters*, 8(10):6803–6810.

[50] Del-Real, C. and Díaz-Fernández, A. M. (2021). Lifeguards in the sky: Examining the public acceptance of beach-rescue drones. *Technology in Society*, 64:1–11.

[51] Delmerico, J., Mintchev, S., Giusti, A., Gromov, B., Melo, K., Horvat, T., Cadena, C., Hutter, M., Ijspeert, A., Floreano, D., Gambardella, L. M., Siegwart, R., and Scaramuzza, D. (2019). The current state and future outlook of rescue robotics. *Journal of Field Robotics*, 36(7):1171 – 1191.

[52] Dynamics, B. (2023). RATP Case Study, Spot Boston Dynamics.

[53] Ekman, P. (1993). Facial expression and emotion. *Am Psychol*, 48(4):384–392.

[54] Elaoua, A., Nadour, M., Cherroun, L., Mahdi, M. T., and Hafaifa, A. (2024). Real-time obstacle detection and path adjustment for autonomous robots using opencv. In *2024 International Conference on Telecommunications and Intelligent Systems (ICTIS)*, pages 1–5.

[55] Eldemiry, A., Zou, Y., Li, Y., Wen, C.-Y., and Chen, W. (2022). Autonomous Exploration of Unknown Indoor Environments for High-Quality Mapping Using Feature-Based RGB-D SLAM. *Sensors*, 22(14).

[56] Elfes, A. (1989). Using occupancy grids for mobile robot perception and navigation. *Computer*, 22(6):46–57.

[57] Endsley, M. R. (1995). Toward a theory of situation awareness in dynamic systems. *Human Factors*, 37(1):32–64.

[58] ERC-BPGC (2020). Omnibase. https://github.com/ERC-BPGC/omnibase.git.

[59] Esteves Henriques, B., Baglioni, M., and Jamshidnejad, A. (2024). Camera-based mapping in search-and-rescue via flying and ground robot teams. *Machine Vision and Applications*, 35(5):117.

[60] Fang, B., Ding, J., and Wang, Z. (2019). Autonomous robotic exploration based on frontier point optimization and multistep path planning. *IEEE Access*, 7:46104–46113.

[61] Fankhauser, P., Bloesch, M., and Hutter, M. (2018). Probabilistic terrain mapping for mobile robots with uncertain localization. *IEEE Robot Autom Lett*, 3(4):3019–3026.

[62] Filho, L. E. S. A. and Júnior, C. L. N. (2022). Multi-robot Autonomous Exploration and Map Merging in Unknown Environments. In *2022 IEEE International Systems Conference (SysCon)*, pages 1–8.

[63] Fischer, G. K. J., Bergau, M., Gómez-Rosal, D. A., Wachaja, A., Graeter, J., Odenweller, M., Piechottka, U., Höflinger, F., Gosala, N., Wetzel, N., Büscher, D., Valada, A., and Burgard, W. (2024). Evaluation of a Smart Mobile Robotic System for Industrial Plant Inspection and Supervision. *IEEE Sensors Journal*, 24(12):19684–19697.

[64] Fox, M. and Long, D. (2003). PDDL2.1: An extension to PDDL for expressing temporal planning domains. *Journal of Artificial Intelligence Research*, 20:61 – 124. Type: Article.

[65] Fu, J., Zheng, H., and Mei, T. (2017). Look closer to see better: Recurrent attention convolutional neural network for fine-grained image recognition. In *IEEE CVPR 2017*, volume 2017-January, pages 4476 – 4484.

[66] Fuentes, A., Yoon, S., Kim, S. C., and Park, D. S. (2017). A robust deep-learning-based detector for real-time tomato plant diseases and pests recognition. *Sensors (Switzerland)*, 17(9).

[67] Gage, A. and Murphy, R. R. (2004). Affective recruitment of distributed heterogeneous agents. In *Proceedings of the 19th National Conference on Artifical Intelligence*, AAAI'04, pages 14–19. AAAI Press.

[68] Gammell, J. D., Srinivasa, S. S., and Barfoot, T. D. (2015). Batch informed trees (BIT*): Sampling-based optimal planning via the heuristically guided search of implicit random geometric graphs. In *Proc ICRA 2015*, page 3067 – 3074.

[69] Gao, Y., Ji, B., Tao, H., and Yuan, R. (2024). A Novel Frontier-Based Multi-Robot Cooperative Exploration Method. In *2024 9th Asia-Pacific Conference on Intelligent Robot Systems (ACIRS)*, pages 186–191.

[70] Garcia, M., Milshteyn, A., Lin, A., Herman, G., Rad, K., Liu, C., and Boussalis, H. (2017). Algorithmic approaches and embedded architectures for robotic search and rescue. In *Proc. Int. Conf. Computational Science and Computational Intelligence (CSCI 2017)*, pages 420–425.

[71] García, S., Menghi, C., and Pelliccione, P. (2019). Mapmaker: Performing multi-robot ltl planning under uncertainty. In *2019 IEEE/ACM 2nd International Workshop on Robotics Software Engineering (RoSE)*, pages 1–4.

[72] Garrett, C. R., Chitnis, R., Holladay, R., Kim, B., Silver, T., Kaelbling, L. P., and Lozano-Pérez, T. (2021). Integrated Task and Motion Planning. *Annual Review of Control, Robotics, and Autonomous Systems*, 4(Volume 4, 2021):265–293. Type: Journal Article.

[73] Geffner, H. (2000). Functional Strips: A More Flexible Language for Planning and Problem Solving. In Minker, J., editor, *Logic-Based Artificial Intelligence*, pages 187–209. Springer US, Boston, MA.

[74] Gelfert, S. (2022). 3d thermal detection of humans in smoky indoor environments. In *2022 7th International Conference on Mechanical Engineering and Robotics Research (ICMERR)*, pages 201–208.

[75] Gelfert, S. (2023). Real Time Victim Detection in Smoky Environments with Mobile Robot and Multi-sensor Unit Using Deep Learning. *Lecture Notes in Networks and Systems*, 642 LNNS:351–364.

[76] Georgievski, I. and Aiello, M. (2015). Htn planning: Overview, comparison, and beyond. *Artificial Intelligence*, 222:124–156.

[77] Gerevini, A. and Long, D. (2005). Plan constraints and preferences in pddl3 the language of the fifth international planning competition. *ICAPS 2006*.

[78] Gerevini, A. and Serina, I. (2002). LPG: A Planner Based on Local Search for Planning Graphs with Action Costs. In *Proceedings of the 6th International Conference on Artificial Intelligence Planning Systems, AIPS 2002*, pages 13 – 22. Type: Conference paper.

[79] Ghallab, M., Knoblock, C., Wilkins, D., Barrett, A., Christianson, D., Friedman, M., Kwok, C., Golden, K., Penberthy, S., Smith, D., Sun, Y., and Weld, D. (1998). PDDL - The Planning Domain Definition Language.

[80] Grünthal, G., editor (1998). *European Macroseismic Scale 1998*, volume 15 of *Cahiers du Centre Européen de Géodynamique et de Séismologie*. Centre Européen de Géodynamique et de Séismologie, Luxembourg.

[81] Gu, S., Holly, E., Lillicrap, T., and Levine, S. (2017). Deep reinforcement learning for robotic manipulation with asynchronous off-policy updates. In *IEEE ICRA 2017*, pages 3389 – 3396.

[82] Guzzi, J., Giusti, A., Gambardella, L. M., and Di Caro, G. A. (2018). A model of artificial emotions for behavior-modulation and implicit coordination in multi-robot systems. In *Proceedings of the Genetic and Evolutionary Computation Conference*, GECCO '18, page 21–28, New York, NY, USA. Association for Computing Machinery.

[83] Habibi, A. and Hajaiej, Z. (2025). Aerial-Ground Collaboration for Fast and Accurate Hazard Detection in SAR Missions. In *2025 IEEE International Conference on Advanced Systems and Emergent Technologies (IC_ASET)*, pages 1–6, Mammamet-Yasmine, Tunisia. IEEE.

[84] Harris, P. L. (1991). *Children and Emotion: The Development of Psychological Understanding*. Wiley.

[85] Hart, P., Nilsson, N., and Raphael, B. (1968). A Formal Basis for the Heuristic Determination of Minimum Cost Paths. *IEEE Transactions on Systems Science and Cybernetics*, 4(2):100–107.

[86] Hirose, N., Sadeghian, A., Vázquez, M., Goebel, P., and Savarese, S. (2018). GONet: A Semi-Supervised Deep Learning Approach For Traversability Estimation. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3044–3051.

[87] Hoffmann, J. (2003). The metric-ff planning system: Translating "ignoring delete lists" to numeric state variables. *Journal of Artificial Intelligence Research*, 20:291 – 341.

[88] Hoffmann, J. and Nebel, B. (2001). The ff planning system: Fast plan generation through heuristic search. *Journal of Artificial Intelligence Research*, 14:263 – 312.

[89] Huang, C.-H., Chen, Y.-C., Hsu, C.-Y., Yang, J.-Y., and Chang, C.-H. (2024). FPGA-based UAV and UGV for search and rescue applications: A case study. *Computers and Electrical Engineering*, 119:109491.

[90] Huang, J., Zhou, B., Fan, Z., Zhu, Y., Jie, Y., Li, L., and Cheng, H. (2023). FAEL: Fast Autonomous Exploration for Large-scale Environments With a Mobile Robot. *IEEE Robot. Autom. Lett.*, 8(3):1667–1674.

[91] Huang, Y.-W., Sasaki, Y., Harakawa, Y., Fukushima, E. F., and Hirose, S. (2011). Operation of underwater rescue robot anchor diver iii during the 2011 tohoku earthquake and tsunami. In *OCEANS'11 MTS/IEEE KONA*, pages 1–6.

[92] Idrees, I., Yun, T., Sharma, N., Deng, Y., Gopalan, N., Konidaris, G., and Tellex, S. (2023). Improved inference of human intent by combining plan recognition and language feedback. In *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 7976–7983.

[93] International Search and Rescue Advisory Group (INSARAG) (2020). *INSARAG Guidelines 2020, Volume III: Operational Field Guide, Annex B20: Assessment, Search and Rescue Level*. United Nations Office for the Coordination of Humanitarian Affairs (OCHA).

[94] Jia, Y., Tang, W., Sun, H., Yang, J., Liu, B., and Wang, C. (2024). Portable Planner for Enhancing Ground Robots Exploration Performance in Unstructured Environments. *IEEE Robot. Autom. Lett.*, 9(11):9295–9302.

[95] Jocher, G. (2020). YOLOv5 by Ultralytics. Version Number: 7.0.

[96] Joshi, A., Kale, S., Chandel, S., and Pal, D. K. (2015). Likert scale: Explored and explained. *British journal of applied science & technology*, 7(4):396.

[97] Kahoul, M., Kahloul, L., and Tigane, S. (2025). Formal specification, and verification of a cooperative multi-robots system for delivery service. In *2025 International Symposium on iNnovative Informatics of Biskra (ISNIB)*, pages 1–6.

[98] Kannan, S. S., Jo, W., Parasuraman, R., and Min, B.-C. (2020). Material mapping in unknown environments using tapping sound. In *Proc. IEEE/RSJ Int. Conf. Intelligent Robots and Systems (IROS 2020)*, pages 4855–4861.

[99] Karaman, S. and Frazzoli, E. (2011). Sampling-based algorithms for optimal motion planning. *Int J Rob Res*, 30(7):846 – 894.

[100] Kavraki, L. E., Švestka, P., Latombe, J.-C., and Overmars, M. H. (1996). Probabilistic roadmaps for path planning in high-dimensional configuration spaces. *IEEE Trans Robot Autom*, 12(4):566 – 580.

[101] Khan, M. B., Chuthong, T., Danh Do, C., Thor, M., Billeschou, P., Larsen, J. C., and Manoonpong, P. (2020). iCrawl: An Inchworm-Inspired Crawling Robot. *IEEE Access*, 8:200655–200668.

[102] Khasawneh, A., Rogers, H., Bertrand, J., Madathil, K. C., and Gramopadhye, A. (2019). Human adaptation to latency in teleoperated multi-robot human-agent search and rescue teams. *Automation in Construction*, 99:265–277.

[103] Kim, H., Kim, H., Lee, S., and Lee, H. (2022). Autonomous Exploration in a Cluttered Environment for a Mobile Robot With 2D-Map Segmentation and Object Detection. *IEEE Robot. Autom. Lett.*, 7(3):6343–6350.

[104] Kim, Y.-D., Kang, J.-H., Sun, D.-H., Moon, J.-I., Ryuh, Y.-S., and An, J. (2010). Design and implementation of user friendly remote controllers for rescue robots in fire sites. In *Proc. of 2010 SICE Annual Conference, SICE 2010*, pages 875–880.

[105] Kirsch, M., Pawar, S., Ferrein, A., and Schiffer, S. (2024). Golog++ hits the (right) spot: Interfacing golog with a quadruped rescue robot for high-level missions. In *Proceedings of the 16th International Conference on Agents and Artificial Intelligence - Volume 3: ICAART*, pages 1052–1059. INSTICC, SciTePress.

[106] Kolvenbach, H., Arm, P., Hampp, E., Dietsche, A., Bickel, V. T., Sun, B., Meyer, C., and Hutter, M. (2021). Traversing steep and granular martian analog slopes with a dynamic quadrupedal robot. *ArXiv*, abs/2106.01974.

[107] Koverola, M., Kunnari, A., Sundvall, J., and Laakasuo, M. (2022). General attitudes towards robots scale (gators): A new instrument for social surveys. *International Journal of Social Robotics*, 14(7):1559–1581.

[108] Kress-Gazit, H., Fainekos, G. E., and Pappas, G. J. (2009). Temporal-logic-based reactive mission and motion planning. *IEEE Transactions on Robotics*, 25(6):1370 – 1381.

[109] Kucukayan, G. and Karacan, H. (2024). YOLO-IHD: Improved Real-Time Human Detection System for Indoor Drones. *Sensors*, 24(3).

[110] Kudriashov, A., Buratowski, T., Garus, J., and Giergiel, M. (2021). 3D Environment Exploration with SLAM for Autonomous Mobile Robot Control. *WSEAS Trans. on Systems and Control*, 16:450–456.

[111] Kuffner Jr., J. J. and La Valle, S. M. (2000). RRT-connect: an efficient approach to single-query path planning. In *Proc. ICRA 2000*, volume 2, page 995 – 1001.

[112] Kuipers, B. and Byun, Y.-T. (1991). A robot exploration and mapping strategy based on a semantic hierarchy of spatial representations. *Robot. Auton. Syst.*, 8(1-2):47 – 63.

[113] Kulkarni, M., Dharmadhikari, M., Tranzatto, M., Zimmermann, S., Reijgwart, V., De Petris, P., Nguyen, H., Khedekar, N., Papachristos, C., Ott, L., Siegwart, R., Hutter, M., and Alexis, K. (2022). Autonomous Teamed Exploration of Subterranean Environments using Legged and Aerial Robots. In *2022 International Conference on Robotics and Automation (ICRA)*, pages 3306–3313.

[114] Kuremoto, T., Obayashi, M., Kobayashi, K., and Feng, L.-B. (2010). Autonomic Behaviors of Swarm Robots Driven by Emotion and Curiosity. In Li, K., Jia, L., Sun, X., Fei, M., and Irwin, G. W., editors, *Life System Modeling and Intelligent Computing*, pages 541–547, Berlin, Heidelberg. Springer Berlin Heidelberg.

[115] Labbé, M. and Michaud, F. (2018). RTAB-Map as an open-source lidar and visual simultaneous localization and mapping library for large-scale and long-term online operation. *Journal of Field Robotics*, 36:416–446.

[116] Larkin, D., Novitzky, M., Kim, J., and Korpela, C. M. (2021). ATAK Integration through ROS for Autonomous Air-ground Team. In *2021 International Conference on Unmanned Aircraft Systems (ICUAS)*, pages 1116–1122, Athens, Greece. IEEE.

[117] Lattanzi, D. and Miller, G. (2017). Review of Robotic Infrastructure Inspection Systems. *Journal of Infrastructure Systems*, 23(3):04017004.

[118] Lee, H.-Y., Zhou, P., Duan, A., Ma, W., Yang, C., and Navarro-Alarcon, D. (2026). Non-prehensile tool-object manipulation by integrating llm-based planning and manoeuvrability-driven controls. *Robotics and Computer-Integrated Manufacturing*, 100:103231.

[119] Lee-Johnson, C. P. and Carnegie, D. A. (2010). Mobile Robot Navigation Modulated by Artificial Emotions. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 40(2):469–480. Type: Journal Article.

[120] Lei, Y., Hou, J., Ma, P., and Ma, M. (2025). Voronoi-GRU-Based Multi-Robot Collaborative Exploration in Unknown Environments. *Applied Sciences*, 15(6):3313.

[121] Leingartner, M., Maurer, J., Ferrein, A., and Steinbauer, G. (2016). Evaluation of Sensors and Mapping Approaches for Disasters in Tunnels. *Journal of Field Robotics*, 33(8):1037 – 1057. Type: Article.

[122] Li, Y., Littlefield, Z., and Bekris, K. E. (2016). Asymptotically optimal sampling-based kinodynamic planning. *Int J Rob Res*, 35(5):528 – 564.

[123] Li, Y., Yu, Y., Zhang, Q., Liang, C., He, P., Chen, W., and Zhao, T. (2023). Losparse: Structured compression of large language models based on low-rank and sparse approximation. In *Proceedings of Machine Learning Research*, volume 202, page 19730 – 19742.

[124] Lin, P., Yin, B., Zhang, L., and Zhao, Y. (2024). Adaptive control algorithm for quadruped robots in unknown high-slope terrain. *Journal of Engineering Research (Kuwait)*.

[125] Lin, T.-Y., Maire, M., Belongie, S., Bourdev, L., Girshick, R., Hays, J., Perona, P., Ramanan, D., Zitnick, C. L., and Dollár, P. (2015). Microsoft COCO: Common Objects in Context. *arXiv:1405.0312*.

[126] Lindqvist, B., Karlsson, S., Koval, A., Tevetzidis, I., Haluška, J., Kanellakis, C., akbar Agha-mohammadi, A., and Nikolakopoulos, G. (2022). Multimodality robotic systems: Integrated combined legged-aerial mobility for subterranean search-and-rescue. *Robotics and Autonomous Systems*, 154:104134.

[127] Liu, K., Gu, J., He, X., and Jia, J. (2025). Safety-critical motion optimization for quadruped robots on offshore platforms: A hierarchical nonlinear model predictive control framework based on foothold optimization and control barrier function. *Control Engineering Practice*, 165:106559.

[128] Lumbanraja, F., Pramswary, R., and Aristoteles, A. (2022). Classification of Cracked Concrete Images Using Convolutional Neural Algorithm. In *AIP Conference Proceedings*, volume 2563.

[129] Lyu, M., Zhao, Y., Huang, C., and Huang, H. (2023). Unmanned aerial vehicles for search and rescue: A survey. *Remote Sensing*, 15(13).

[130] Macenski, S., Foote, T., Gerkey, B., Lalancette, C., and Woodall, W. (2022). Robot Operating System 2: Design, architecture, and uses in the wild. *Science Robotics*, 7(66).

[131] Maguire, M., Dorafshan, S., and Thomas, R. J. (2018). SDNET2018: A concrete crack image dataset for machine learning applications.

[132] Marques, C., Cristóvão, J., Alvito, P., Lima, P., Frazão, J., Ribeiro, I., and Ventura, R. (2007). A search and rescue robot with tele-operated tether docking system. *Industrial Robot: An International Journal*, 34(4):332–338.

[133] Martín, F., Ginés, J., Rodríguez, F. J., and Matellán, V. (2021). PlanSys2: A Planning System Framework for ROS2. In *IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2021, Prague, Czech Republic, September 27 - October 1, 2021*. IEEE.

[134] Martín-Martín, R., Patel, M., Rezatofighi, H., Shenoi, A., Gwak, J., Frankel, E., Sadeghian, A., and Savarese, S. (2023). JRDB: A Dataset and Benchmark of Egocentric Robot Visual Perception of Humans in Built Environments. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(6):6748–6765.

[135] Maruyama, Y., Kato, S., and Azumi, T. (2016). Exploring the performance of ros2. In *Proceedings of the 13th International Conference on Embedded Software, EMSOFT 2016*.

[136] Mastalli, C., Havoutis, I., Focchi, M., Caldwell, D. G., and Semini, C. (2020). Motion Planning for Quadrupedal Locomotion: Coupled Planning, Terrain Mapping, and Whole-Body Control. *IEEE Transactions on Robotics*, 36(6):1635–1648.

[137] Miki, T., Wellhausen, L., Grandia, R., Jenelten, F., Homberger, T., and Hutter, M. (2022). Elevation mapping for locomotion and navigation using GPU. In *Proc IROS 2022*, pages 2273–2280.

[138] Miles, M. J., Biggie, H., and Heckman, C. (2023). Terrain-aware semantic mapping for cooperative subterranean exploration. *Frontiers in Robotics and AI*, 10. Type: Article.

[139] Mimmo, N., Bernard, P., and Marconi, L. (2021). Avalanche victim search via robust observers. *IEEE Transactions on Control Systems Technology*, 29(4):1450–1461.

[140] Mondal, S., Pal, M., Sen, S., Utkarsh, P., Bhattacharjee, N., and Singha, T. (2025). Autonomous navigation in dynamic environments using modified pure pursuit, vector field histogram, and lidar-slam. In *Proc. ICOEI 2025*, page 1660 – 1664.

[141] Motepe, S. and Stopforth, R. (2015). Mechatronic integration for search and rescue applications: UAV vision system for mining and manufacturing environments. In Brusa, E., editor, *Mechatronics: Principles, Technologies and Applications*, pages 119–138. Nova Science Publishers, New York, NY, USA.

[142] Murphy, R. R. (2000). *Introduction to AI Robotics*. MIT Press, Cambridge, MA, USA, 1st edition.

[143] Murphy, R. R. (2014). *Disaster robotics*. MIT press.

[144] Murphy, R. R., Steimle, E., Griffin, C., Cullins, C., Hall, M., and Pratt, K. (2008). Cooperative use of unmanned sea surface and micro aerial vehicles at hurricane wilma. *Journal of Field Robotics*, 25(3):164–180.

[145] Murphy, R. R., Tadokoro, S., and Kleiner, A. (2016). *Disaster Robotics*, pages 1577–1604. Springer International Publishing, Cham.

[146] Naji, M. (2022). doors Dataset. Publication Title: Roboflow Universe Published: https://universe.roboflow.com/mohammed-naji/doors-6g8eb.

[147] Niroui, F., Zhang, K., Kashino, Z., and Nejat, G. (2019). Deep reinforcement learning robot for search and rescue applications: Exploration in unknown cluttered environments. *IEEE Robot. Autom. Lett.*, 4(2):610–617.

[148] Nivethika, S. D., Nivethetha, T., Priyadharshini, P., Nithyasri, V. T., SenthilPandian, M., and Sivaprasad, R. (2022). Design and Development of Pipe inspection Snake Locomotion Robot. In *Proc. ICPECTS'22*, pages 1–5.

[149] O'Brien, R. T., Walz, E., Hammonds, K. R., and Rumbaugh, M. (2024). Experimental Investigation of a Semi-Autonomous Search and Rescue System. In *2024 UKACC 14th International Conference on Control (CONTROL)*, pages 66–71, Winchester, United Kingdom. IEEE.

[150] of Labor Statistics, U. B. (2017). Fatal work injuries for fire and rescue workers.

[151] Parkinson, B., Wolf, A., Galambos, P., and Széll, K. (2023). Assessment of the Utilization of Quadruped Robots in Pharmaceutical Research and Development Laboratories. In *Proc. INES'23*, pages 221–228.

[152] Plaku, E., Bekris, K. E., Chen, B. Y., Ladd, A. M., and Kavraki, L. E. (2005). Sampling-based roadmap of trees for parallel motion planning. *IEEE Trans Robot*, 21(4):597 – 608.

[153] Prajapati, D., Mehra, A., Kumar, P., Rana, A., Surya Prakash, S. K., and Shukla, A. (2026). Mppf: Mobile probe potential field for real-time multi-agent navigation in industrial environments. *IEEE Access*, 14:11084–11095.

[154] Purnima Bholowalia, A. K. (2014). EBK-Means: A Clustering Technique based on Elbow Method and K-Means in WSN. *Int. J. Comput.*, 105(9):17–24.

[155] Qie, B. (2025). An improved a*-based hybrid algorithm for integrated path planning and dynamic obstacle avoidance. In *2025 IEEE 7th International Conference on Power, Intelligent Computing and Systems (ICPICS)*, pages 1831–1835.

[156] Raiola, G., Mingo Hoffman, E., Focchi, M., Tsagarakis, N., and Semini, C. (2020). A simple yet effective whole-body locomotion framework for quadruped robots. *Front. Robot. AI*, 7:159–176.

[157] Redmon, J., Divvala, S., Girshick, R., and Farhadi, A. (2016). You only look once: Unified, real-time object detection. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 779–788.

[158] Rezatofighi, H., Tsoi, N., Gwak, J., Sadeghian, A., Reid, I., and Savarese, S. (2019). Generalized Intersection Over Union: A Metric and a Loss for Bounding Box Regression. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 658–666.

[159] Ritchie, H., Rosado, P., and Roser, M. (2022). Natural disasters. *Our World in Data*. https://ourworldindata.org/natural-disasters.

[160] Roberts, M., Hiatt, L. M., Shetty, V., Brumback, B., Enochs, B., and Jampathom, P. (2021). Goal lifecycle networks for robotics. *The International FLAIRS Conference Proceedings*, 34.

[161] Rochefoucauld, V. D. L., Ratsamee, P., Lacroix, S., Ingrand, F., and Uranishi, Y. (2025). A modular execution architecture for robust multi-robot planning and acting in trans-media environments. *IEEE Access*, 13:186208–186230.

[162] Roh, J., Kim, J., Park, C., and Shim, D. H. (2025). Autonomous Multi-Floor and Narrow Indoor Exploration Using Multi-Criteria Decision-Making Approach. In *2025 22nd International Conference on Ubiquitous Robots (UR)*, pages 260–267.

[163] ROS2 (2024). Robot operating system 2.

[164] Rosas-Cervantes, V., Hoang, Q.-D., Lee, S.-G., and Choi, J.-H. (2021). Multi-Robot 2.5D Localization and Mapping Using a Monte Carlo Algorithm on a Multi-Level Surface. *Sensors*, 21:1–16.

[165] Ruan, T., Betta, Z., Tzoumas, G., Stolkin, R., and Chiou, M. (2025). First Responders' Perceptions of Semantic Information for Situational Awareness in Robot-Assisted Emergency Response. In *2025 IEEE International Symposium on Safety, Security and Rescue Robotics (SSRR)*.

[166] Rudol, P. and Doherty, P. (2019). Evaluation of human body detection using deep neural networks with highly compressed videos for UAV search and rescue missions. In Nayak, A. C. and Sharma, A., editors, *PRICAI 2019: Trends in Artificial Intelligence*, volume 11672 of *Lecture Notes in Computer Science*, pages 402–417. Springer International Publishing, Cham.

[167] Rösmann, C., Hoffmann, F., and Bertram, T. (2017). Kinodynamic trajectory optimization and control for car-like robots. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 5681–5686.

[168] S., D., R, A. A., Subhash, S., B.P., S., R., R., and S., N. (2025). Intelligent rescue rover for human detection using transfer learning and embedded systems*. In *2025 Advanced Computing and Communication Technologies for High Performance Applications (ACCTHPA)*, pages 1–6.

[169] Sacchetto, S., Price, T., and Honary, V. (2025). Evaluating real-time facial recognition on a ugv platform: Effects of angle, distance, and training image orientation. In *2025 IEEE 16th Annual Ubiquitous Computing, Electronics & Mobile Communication Conference (UEMCON)*, pages 0029–0033.

[170] Sakayori, G. and Ishigami, G. (2021). Energy-aware trajectory planning for planetary rovers. *Adv. Robot.*, 35(21-22):1302 – 1316.

[171] Sambolek, S. and Ivasic-Kos, M. (2025). Person Detection and Geolocation Estimation in Drone Images. *SN Computer Science*, 6(4). Type: Article.

[172] Santos, L., Santos, F., Mendes, J., Costa, P., Lima, J., Reis, R., and Shinde, P. (2020). Path Planning Aware of Robot's Center of Mass for Steep Slope Vineyards. *Robotica*, 38(4):684–698.

[173] Santos, M. C., Bartlett, B., Schneider, V. E., Brádaigh, F. O., Blanck, B., Santos, P. C., Trslic, P., Riordan, J., and Dooly, G. (2024). Cooperative Unmanned Aerial and Surface Vehicles for Extended Coverage in Maritime Environments. *IEEE Access*, 12:9206 – 9219.

[174] Sautenkov, O., Yaqoot, Y., Lykov, A., Mustafa, M. A., Tadevosyan, G., Akhmetkazy, A., Cabrera, M. A., Martynov, M., Karaf, S., and Tsetserukou, D. (2025). Uav-vla: Vision-language-action system for large scale aerial mission generation. In *2025 20th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, pages 1588–1592.

[175] Savery, R. and Weinberg, G. (2020). A Survey of Robotics and Emotion: Classifications and Models of Emotional Interaction. In *2020 29th IEEE International Conference on Robot and Human Interactive Communication (RO-MAN)*, pages 986–993.

[176] Schwaiger, S., Muster, L., Scherl, A., Trivisonne, P., Wöber, W., and Thalhammer, S. (2024). Towards full autonomy in mobile robot navigation and manipulation. *Elektrotech. Inftech.*, 141(6):400–410.

[177] Schönberger, J. L., Pollefeys, M., Geiger, A., and Sattler, T. (2018). Semantic Visual Localization. *arXiv:1712.05773*.

[178] Semenas, R. and Bausys, R. (2021). Adaptive Strategy for Environment Exploration in Search and Rescue Missions by Autonomous Robot. In Sharma, H., Gupta, M. K., Tomar, G. S., and Lipo, W., editors, *Proc. Communication and Intelligent Systems*, pages 335–353.

[179] Sharma, A. M., Syed, I. A., Sharma, B., Jamal, A., and Deodhare, D. (2018). Visual object detection for an autonomous indoor robotic system. In Chaudhuri, B. B., Kankanhalli, M. S., and Raman, B., editors, *Proc. 2nd Int. Conf. Computer Vision and Image Processing*, volume 703 of *Adv. Intell. Syst. Comput.*, pages 217–228. Springer, Singapore.

[180] Sifat, A. H., Bharmal, B., Zeng, H., Huang, J.-B., Jung, C., and Williams, R. K. (2023). Towards computational awareness in autonomous robots: an empirical study of computational kernels. *Complex & Intelligent Systems*, 9(6):6269–6295.

[181] Sruthi, M. S., Poovathingal, M. J., Nandana, V. N., Lakshmi, S., Samshad, M., and Sudeesh, V. S. (2021). YOLOv5 based Open-Source UAV for Human Detection during Search And Rescue (SAR). In *2021 International Conference on Advances in Computing and Communications (ICACC)*, pages 1–6.

[182] Stefaniak, P. and Anufriiev, S. (2020). Method of Defining Diagnostic Features to Monitor the Condition of the Belt Conveyor Gearbox with the Use of the Legged Inspection Robot. In *Intelligent Information and Database Systems*, pages 158–167, Singapore. Springer Singapore.

[183] Sucan, I. A., Moll, M., and Kavraki, L. E. (2012). The open motion planning library. *IEEE Robot Autom Mag*, 19(4):72–82.

[184] Sun, J., Meng, Y., Tan, J., Sun, C., Zhang, J., Ding, N., Qian, H., and Zhang, A. (2017). A vision-based perception framework for outdoor navigation tasks applicable to legged robots. In *2017 Chinese Automation Congress (CAC)*, pages 2894–2899.

[185] Tani, S., Ruscio, F., and Costanzi, R. (2024). Preliminary Online Validation of a Visual-Acoustic-based Framework for Autonomous Underwater Structure Inspections. In *OCEANS 2024 - Halifax*, pages 1–8.

[186] Tauler-Rossello, A., Garcia-Fidalgo, E., and Ortiz, A. (2024). A Frontier-Based Exploration Approach Making Use of Jointly Updated Local and Global Trees. In *2024 IEEE 29th International Conference on Emerging Technologies and Factory Automation (ETFA)*, pages 1–4.

[187] Tepylo, N., Debelle, L., and Laliberte, J. (2023). Public perception of remotely piloted aircraft systems in Canada. *Technology in Society*, 73:1–12.

[188] Tranzatto, M., Mascarich, F., Bernreiter, L., Godinho, C., Camurri, M., Khattak, S., Dang, T., Reijgwart, V., Löje, J., Wisth, D., Zimmermann, S., Nguyen, H., Fehr, M., Solanka, L., Buchanan, R., Bjelonic, M., Khedekar, N., Valceschini, M., Jenelten, F., Dharmadhikari, M., Hornberger, T., De Petris, P., Wellhausen, L., Kulkami, M., Miki, T., Hirsch, S., Montenegro, M., Papachristos, C., Tresoldi, F., Carius, J., Valsecchi, G., Lee, J., Meyer, K., Wu, X., Nieto, J., Smith, A., Hutter, M., Siegwart, R., Mueller, M., Fallon, M., and Alexis, K. (2022). CERBERUS: Autonomous Legged and Aerial Robotic Exploration in the Tunnel and Urban Circuits of the DARPA Subterranean Challenge. *Field Robotics*, 2:274–324.

[189] Tsankova, D. D. (2009). Emotional intervention on an action selection mechanism based on artificial immune networks for navigation of autonomous agents. *Adaptive Behavior*, 17(2):135–152. Type: Journal Article.

[190] Tseng, Y.-H., Kang, S.-C., Chang, J.-R., and Lee, C.-H. (2011). Strategies for autonomous robots to inspect pavement distresses. *Automation in Construction*, 20(8):1156–1172.

[191] Umari, H. and Mukhopadhyay, S. (2017). Autonomous robotic exploration based on multiple rapidly-exploring randomized trees. In *Proc. 2017 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, IROS 2017*, pages 1396–1402.

[192] V Y, H., S, M., Aruna, T. N., Gayana, R., J, A., Choudhry, M. D., and M, S. (2024). Path Planning Solution for Intelligent Robots Using Ex* RRT Algorithm in Disaster Relief. In *2024 5th International Conference on Recent Trends in Computer Science and Technology (ICRTCST)*, pages 413–418, Jamshedpur, India. IEEE.

[193] Vielfaure, D., Arseneault, S., Lajoie, P.-Y., and Beltrame, G. (2022). DORA: Distributed Online Risk-Aware Explorer. In *2022 International Conference on Robotics and Automation (ICRA)*, pages 6919–6926.

[194] Virk, G. S., Gatsoulis, Y., Parack, M., and Kherada, A. (2008). Mobile Robotic Issues for Urban Search and Rescue. In *Proc. 2006 17th World Congress Int. Federation of Automatic Control, IFAC 2006*, volume 41, pages 3098–3103.

[195] Walz, E., Hammonds, K. R., Rumbaugh, M., and O'Brien, R. T. (2023). Semi-Autonomous Search and Rescue System. In *2023 International Conference on Unmanned Aircraft Systems (ICUAS)*, pages 1207–1212, Warsaw, Poland. IEEE.

[196] Wang, C., Luo, R., Whitney, J. P., and Padir, T. (2025a). Practical Approaches Towards Transparent and Stable Bilateral Teleoperation Under Time-Varying Network Delay. *International Journal of Social Robotics*, 17(3):409–426.

[197] Wang, C.-Y., Bochkovskiy, A., and Liao, H.-Y. M. (2022). YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors. *arXiv:2207.02696*.

[198] Wang, G., Wang, W., Ding, P., Liu, Y., Wang, H., Fan, Z., Bai, H., Hongbiao, Z., and Du, Z. (2023). Development of a search and rescue robot system for the underground building environment. *Journal of Field Robotics*, 40(3):655–683.

[199] Wang, H., Wang, G., Zhu, H., Du, Z., and Wang, W. (2025b). A semantic-aware measurement-driven autonomous target search framework for UGV in complex unknown environments. *Measurement Science and Technology*, 36.

[200] Wang, Y., Liu, H., Shao, X., Yao, W., and Chen, Y. (2026). GA-FMP: An efficient greedy-accelerated fast marching planner for autonomous UGV exploration. *Information Sciences*, 729:122873.

[201] Watt Smith, T. (2015). *The Book of Human Emotions: An Encyclopedia of Feeling from Anger to Wanderlust*. Profile Books Ltd (in partnership with Wellcome Collection), London.

[202] Wellhausen, L., Dosovitskiy, A., Ranftl, R., Walas, K., Cadena, C., and Hutter, M. (2019). Where Should I Walk? Predicting Terrain Properties From Images Via Self-Supervised Learning. *IEEE Robotics and Automation Letters*, 4(2):1509–1516.

[203] Wellhausen, L. and Hutter, M. (2023). ArtPlanner: Robust legged robot navigation in the field. *J Field Robot*, 3:413–434.

[204] Wermelinger, M., Fankhauser, P., Diethelm, R., Krusi, P., Siegwart, R., and Hutter, M. (2016). Navigation planning for legged robots in challenging terrain. *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*.

[205] Wisth, D., Camurri, M., and Fallon, M. (2023). Vilens: Visual, inertial, lidar, and leg odometry for all-terrain legged robots. *IEEE Transactions on Robotics*, 39(1):309–326.

[206] Wu, S., Su, S., Huang, Y., Jiang, H., and Chu, Y. (2025). Unmanned Vehicle Unknown Environment Exploration Algorithm Based on Motion Primitives. *Unmanned Systems*.

[207] Xi, X., Zhao, J., Wang, H., Yan, T., and Liu, J. (2024). Research on Recognition Algorithm of Power Facility Inspection Robot Based on Computer Vision. In *Proc. ICCSIE '24*, pages 446–453, New York, NY, USA.

[208] Yamauchi, B. (1997). A frontier-based approach for autonomous exploration. In *Proc. 1997 IEEE Int. Symp. on Computational Intelligence in Robotics and Automation, CIRA'97.*, pages 146–151.

[209] Yousefi, A., Betta, Z., Mottola, G., Recchiuto, C. T., and Sgorbissa, A. (2024). Immersive control of a quadruped robot with Virtual Reality Eye-wear. In *2024 33rd IEEE International Conference on Robot and Human Interactive Communication (ROMAN)*, pages 49–55.

[210] Yousefi, A., Betta, Z., Recchiuto, C. T., and Sgorbissa, A. (2026a). Slope-constrained global path planning for quadruped robots. *Intel Serv Robotics*, 19(2):34.

[211] Yousefi, A., Depalo, S., Betta, Z., Recchiuto, C. T., and Sgorbissa, A. (2026b). Applications of unmanned aerial vehicles in the maritime transportation industry: a scoping review. Under review at Transportation Research Interdisciplinary Perspectives.

[212] Yu, C., Liu, Z., Liu, X.-J., Xie, F., Yang, Y., Wei, Q., and Fei, Q. (2018). DS-SLAM: A Semantic Visual SLAM towards Dynamic Environments. In *IEEE IROS 2018*, pages 1168 – 1174.

[213] Zhang, M., Li, Y., Chi, W., Zhang, W., Lin, X., and Zhang, Z. (2025). A Topological Map based Autonomous Exploration Method for Mobile Robots. In *2025 IEEE International Conference on Real-time Computing and Robotics (RCAR)*, pages 848–853.

[214] Zhang, S., Zhang, X., Li, T., Yuan, J., and Fang, Y. (2022). Fast Active Aerial Exploration for Traversable Path Finding of Ground Robots in Unknown Environments. *IEEE Trans. Instrum. Meas.*, 71:1–13.

[215] Zhao, K. and Ning, L. (2022). Hybrid navigation method for multiple robots facing dynamic obstacles. *Tsinghua Science and Technology*, 27(6):894–901.

[216] Zhao, Z.-A., Wang, S., Chen, M.-X., Mao, Y.-J., Chan, A. C.-H., Lai, D. K.-H., Wong, D. W.-C., and Cheung, J. C.-W. (2025). Enhancing Human Detection in Occlusion-Heavy Disaster Scenarios: A Visibility-Enhanced DINO (VE-DINO) Model with Reassembled Occlusion Dataset. *Smart Cities*, 8(1).

[217] Zhou, Z., Lee, D. J., Yoshinaga, Y., Balakirsky, S., Guo, D., and Zhao, Y. (2022). Reactive task allocation and planning for quadrupedal and wheeled robot teaming. In *2022 IEEE 18th International Conference on Automation Science and Engineering (CASE)*, pages 2110–2117.