Functional and Adaptive Construction for Rescue, an Analysis of the Approach Using Autonomous Robots

THÈSE N^{O} 7683 (2017)

PRÉSENTÉE LE 24 MAI 2017

À LA FACULTÉ DES SCIENCES ET TECHNIQUES DE L'INGÉNIEUR
LABORATOIRE DE SYSTÈMES ROBOTIQUES
PROGRAMME DOCTORAL EN ROBOTIQUE, CONTRÔLE ET SYSTÈMES INTELLIGENTS

ÉCOLE POLYTECHNIQUE FÉDÉRALE DE LAUSANNE

POUR L'OBTENTION DU GRADE DE DOCTEUR ÈS SCIENCES

PAR

Hadi ARDINY

acceptée sur proposition du jury:

Dr A. Karimi, président du jury
Prof. F. Mondada, directeur de thèse
Dr V. Trianni, rapporteur
Dr S. Witwicki, rapporteur
Prof. A. Ijspeert, rapporteur

ÉCOLE POLYTECHNIQUE
FÉDÉRALE DE LAUSANNE

Suisse
2017

God created man.
(And) taught him eloquence.
— Quran 55-2,3

To the loves of my life, my beloved wife, our sweet little daughters, and to my kind parents.

Acknowledgements

First and foremost, I would like to express my deepest gratitude to my advisor **Prof. Francesco Mondada** for his patience, motivation, continuous support, and valuable guidance with excellent advices. Francesco helped me in all the time of research and provided me with the freedom to explore different directions.

Besides my advisor, I would like to thank my committee members: **Prof. Auke Ijspeert**, **Dr. Vito Trianni**, **Dr. Stefan Witwicki** and **Dr. Alireza Karimi** for letting my defense be an enjoyable moment, and for their insightful comments and suggestions.

I would like to thank my colleagues from Mobots for friendly and comfortable working atmosphere. I wish to thank **Frank Bonnet** who helped me to not feel alone specially during the very last days. I would thank **Marcelo Elias de Oliveria** and **Mohammad Ehsanul Karim** for being a great officemate. Special thanks goes **Daniel Burnier** who was a passionate motivator and a kind-hearted person. I thank **Norbert Crot** for his kindness and his help with providing some mechanical parts. I would also thank to other Mobots people, present or former: **Morgane Chevalier**, **Gordana Gerber**, **Stéphane Magnenat**, **Alexey Gribovskiy**, **Fanny Riedo**, **Philippe Rétornaz**, **Florian Vaussard**, **Mariza Freire**, **Manon Briod**, **Maria María Beltran**, **Christophe Barraud**, and **Michael Bonani**. I had great times with these nice and helpful people.

My sincere thanks also goes to **Anne Remillet-Schaller**, **Lebet Corinne**, and **Claudia Marra** for their supports and helps.

Finally and most importantly, a special thanks to my family and friends. These four years were an exhausting adventure, so words cannot express how grateful I am to my beloved wife, **Zahra**. She was always full of energy and incented me to strive towards my goal. I would thank to Zahra for her kindness, patience, compassion, deep caring, and humour. Things would not have been also so nice without the support of my parents: **Davoud**, **Shahrbanoo** for all of the sacrifices that they have made on my behalf. Their prayer for me was what kept me so far. I thank to my little daughters, **Zeinab** and **Kowsar** for making me happy and proud as their father. I would also thank to my mother-in-law and father-in-law for their helps, especially during birth of my kid. At the end I would like express appreciation to my brothers and friends who supported and helped me.

Lausanne, March 2017

Hadi Ardiny

Abstract

Although recent advances in the field of robotics have greatly increased robotic capabilities for several applications, autonomous robots are still in their infancy regarding their support of on-site construction. Unlike production robots that are used, for instance, in automotive industries, autonomous robotic systems should be designed with special considerations, such as the complexity of the cluttered and dynamic working space, inaccuracy in positioning because of the nature of mobile systems, and so forth. Thus, construction has been known as a highly complex application field for robotic systems, especially in unknown environments.

In this thesis, we focus on the developments of an autonomous construction system by taking inspiration from architectural designs found in the animal kingdom. The construction system allows robots to build functional structures over a different range of environmental conditions or in unknown sites without the use of any predefined construction plan (i.e., blueprint). We call this approach *adaptive and functional autonomous construction (AFAC)*. Indeed, AFAC is a robust and intelligent system that can tackle unforeseen problems of unknown environments and faults made during the course of building.

We present two approaches for AFAC. One is a local approach that enables robots to locally sense features of their environment and then act immediately according to defined rules without using any global representation or knowledge of the world. Second is a global approach, which involves building structures by analyzing global representations and knowledge produced by the robots. This approach consists of three phases: the first phase is exploration, to map the unknown environment and to find important elements. In the second phase, an effective construction plan is autonomously computed using collected data and the defined functions. Finally, in the last phase, the robot builds structures based on the computed construction plan.

One interesting implementation of the introduced construction system is its potential use in a post-disaster environment, where an autonomous robotic system can perform construction tasks in the rescue operation. Robots can be employed to build protective structures for rescue functions, including stabilizing large structures or protecting victims in unknown environments, where the environment are perhaps covered by debris. Inasmuch as the robots cannot plan in advance to build these structures, the rescue application is a well-suited case for the study and implementation of AFAC in unknown and cluttered environments.

Abstract

Through implementation of AFAC in the rescue situations, we develop AFAC from scratch to the level at which the robots can autonomously apply AFAC's principles for use in the new classes of construction applications. We also study and compare the local and global approaches of AFAC. In addition, we develop new exploration methods and construction plan computation methodologies for the global approach to efficiently lead AFAC.

Keywords:
Autonomous construction, Adaptive and functional construction, Rescue, Exploration, Construction plan, Reactive construction, Plan-based construction.

Résumé

Malgré les récents progrès dans le domaine robotique, les robots autonomes sont encore à leurs débuts au niveau des applications de construction sur site. Contrairement aux robots de production, qui sont utilisés par exemple dans l'industrie automobile, les systèmes robotiques autonomes pour la construction doivent être conçus en tenant compte de la complexité de l'espace de travail encombré et dynamique, l'inexactitude dans le positionnement dû à la nature des systèmes mobiles, etc. Ainsi, la construction est connue comme un domaine d'application hautement complexe pour les systèmes robotiques, en particulier si l'environnement de travail n'est pas connu à l'avance.

Dans cette thèse, nous nous concentrons sur l'étude d'un système de construction autonome s'inspirant des animaux. Ce système de construction permet aux robots de construire des structures fonctionnelles sur une gamme variée de conditions environnementales et / ou sur des sites inconnus sans recours à un plan de construction prédéfini (par exemple, plan du bâtiment). En effet, il s'agit d'un système de construction intelligent et robuste qui vise à faire face à des problèmes imprévus d'environnements ou des défauts inconnus en cours de construction. Nous comparons deux approches pour la gestion du procédé de construction. La première est une approche locale, qui permet de s'appuyer sur les caractéristiques locales de l'environnement, puis agir de façon réactive selon des règles définies sans utiliser une représentation globale ou une connaissance du monde.

La deuxième est une approche globale qui construit des structures en s'appuyant sur des représentations de l'environnement et des connaissances du monde qui sont produites par les robots. Cette deuxième approche comporte trois phases : la première phase est l'exploration pour cartographier l'environnement inconnu et pour trouver des éléments importants. Dans la deuxième phase, un plan de construction efficace est calculé de façon autonome grâce aux données collectées et aux ressources à disposition. Enfin, le robot construit des structures basées sur le plan de construction calculé.

Une mise en œuvre intéressante et nouvelle pour ce système de construction consiste à sécuriser un environnement post-catastrophe afin de permettre les opérations de sauvetage. Les robots peuvent être utilisés pour construire des structures de protection pour les fonctions de sauvetage telles que des renforts structurels ou la protection des victimes par rapport à des dangers comme des radiations ou des éboulement. Dans ce type de situation, le robot

doit faire face à un environnement qui ne peut pas être prévu à l'avance. Cette application de sauvetage est donc un cas bien adapté à l'étude et à la mise en œuvre de la construction adaptative et fonctionnelle autonome (appelée AFAC).
La mise en œuvre de l'AFAC pour cette nouvelle application de sauvetage nous a permis de créer une architecture de contrôle à partir de zéro et atteindre un niveau dans lequel les robots peuvent appliquer de façon autonome les principes étudiés soit en simulation que sur des robots réels.
Cette étude nous a aussi permis d'identifier des besoins particuliers comme par exemple dans le domaine des algorithmes d'exploration ou les méthodes de calcul de plans de construction, les deux nécessaires pour conduire efficacement l'AFAC.

Mots clefs :
Construction autonome, Construction adaptative et fonctionnelle, Sauvetage, Exploration, Plan de construction, Construction réactive, Construction planifiée.

چکیده

علیرغم توسعه های شگرف در قابلیت های رباتیکی جهت بکارگیری در حوزه های گوناگون ، استفاده از ربات به منظور ساخت سازه ها به صورت خودکار، در ابتدای راه خود قرار دارد. در صنایع همچون خودرو سازی، ربات ها جز ضروری و تفکیک ناپذیر خط تولید مبدل گشته اند و به افزایش بهره وری و کاهش هزینه تولید کمک شایانی نموده اند. این در حالی است که محدودیت هایی نظیر محیط های نامرتب و غیر قابل پیش بینی ، استفاده از مصالح ساختمانی مشترک، پیچیدگی ها و تنوع های فرآیند های ساخت و غیره باعث شده است که چالش های جدی جهت استفاده از ربات در حوزه ساخت و ساز باشد.

در این تحقیق ما در صدد توسعه یک سیستم خودکار ساخت و ساز با الهام از طبیعت هستیم، این سیستم به ربات ها اجازه می دهد که بتوانند سازه های تابعی را در محیط های غیر قابل پیش بینی و تحت شرایط کاملا متفاوت بسازند، بدون اینکه نیازی به استفاده از نقشه از پیش تعریف شده داشته باشند. این سیستم یک سیستم ساخت و ساز تابعی و تطبیقی خودکار است که ما به اختصار آن را AFAC می نامیم. AFAC یک سیستم هوشمند و مقاوم است که می تواند بر مشکلات پیش بینی نشده در محیط ناشناخته و یا خطاهای هنگام ساخت و ساز فائق آید.

دو رویکرد متفاوت برای AFAC مطرح شده است: یک روش مبتنی بر رویکرد محلی، که در آن ربات ها به صورت محلی خصوصیات محیطی را درک می کنند و بر اساس یک سری قوانین از پیش تعیین شده، واکنش مورد نظر را پیاده سازی می کنند. در مقابل این رویکرد، رویکرد سرتاسری می باشد. در رویکرد سرتاسری نیاز است که یک مدل و نقشه از محیط وجود داشته باشد و بعد بر اساس آن، ربات ساخت یک سازه تابعی و تطبیقی را به پیش ببرد. AFAC طبق رویکرد سرتاسری، خود نیاز مند سه مرحله می باشد: در مرحله نخست، ربات ها باید به نقشه برداری از محیط و مشخص کردن مکان های عناصر مهم بپردازند، سپس ربات ها به محاسبه یک نقشه بهینه بر اساس اطلاعات جمع شده از مرحل قبل و توابع تعریف شده، می پردازند، و در آخر، ربات ها شروع به ساخت و ساز بر اساس نقشه محاسبه شده می کنند.

یکی از کاربردهای شگرف برای سیستم معرفی شده، استفاده از AFAC بعد از یک حادثه (نظیر زلزله، حوادث اتمی و ...) می باشد، جاییکه ربات ها می توانند با ساخت و ساز به ایمن کردن محیط بپردازند. ربات ها می توانند برای اهدافی همچون ممانعت از فرو ریختن دیوار و یا ساخت یک دیوار محافظتی بدور مواد خطرناک همچون رادیواکتیو،

بکار گرفته شوند.

از طرفی، از آنجاییکه ربات ها نمی توانند در یک چنین محیط پیچیده ای (محیط بعد از حادثه) ساخت و ساز خودکار را با یک نقشه از پیش تعیین شده، پیاده سازی نمایند، لذا این کاربری، یک بستر مناسب جهت پیاده سازی و توسعه AFAC می باشد. از طریق پیاده سازی AFAC، ما می توانیم به توسعه از مفاهیم ابتدایی به سطحی برسیم که ربات ها بتوانند برای یک ساخت و ساز خودکار برای کاربرد های نوین بکار گرفته شوند.

در کنار پیاده سازی AFAC، ما به شرح الگوریتم های نوین، جهت جستجو با ربات هایی که تحت محدودیت های شدید سنسوری می باشند، می پردازیم. همچنین نحوه محاسبه و بهینه سازی نقشه ساخت و ساز به صورت هوشمند را شرح خواهیم داد.

کلمات کلیدی:

ساخت و ساز خودکار، ساخت و ساز تطبیقی و تابعی، جستجو، نقشه ساخت، ساخت واکنشی، ساخت مبتنی بر طرح.

Contents

List of Figures

List of Tables

List of Algorithms

1 Introduction

1.1 Motivation

Termite mounds, bee hives, burrows, beaver dams, bird nests, and spider webs are structures that have garnered much attention in scientific communities. Understanding what principles animals use, how their behaviours emerge in construction, how these animals tackle faults during the course of construction, and identifying the factors that influence animals' structures can generate bio-inspired approaches for progress in human applications, for instance, autonomous construction.

Often, animal build structures based on desired functions and their essential requirements, such as prey capture, protection from predators, temperature and humidity control, mate attraction, and offspring protection [1, 2]. A more specific example is the ventilation systems in termite mounds; this consists of enormous channels and open spaces. This system brings in oxygen, as well as carries away carbon dioxide and heat that has accumulated from hundreds of thousands of individuals [1, 2]. Another example is the beaver's lodge that, among many other functions, helps beavers hide from predators. The beaver builds its entrance doors that are at least a foot underwater; the chamber itself is placed above the waterline to keep it dry. This shape of the structure prevents predators from invading, keeping the nest safe.

In addition, animal structures are adapted to fit with different environmental situations. For instance, the Baltimore oriole (Icterus galbula) suspends its nest from above or attaches it both above and below, securing it to numerous branches depending what available [2]. Beavers build their dams in accordance with the soil and topography of the environment, so two lodges are never the same [1]. In an experiment in the 1960s, the entrance tube and the antechamber above Baya weaver nest were removed, and a large hole was made on the other side. The birds demonstrated different behaviours: some birds repaired the nest to the same condition as it was before the damages, some birds blocked up the entrance, leaving the new hole with no

tube on the other side, and one closed both holes [1]. In some similar cases, animals repair damage in such a way that the repaired structure is unlike the original. These observations show that the animals can perform construction duties over a wide range of environmental conditions and build their structures in different ways.

Scientists are unclear about the principles of animal construction that produce such complex structures using only simple and stereotypical behaviours. For instance, how do termites build dealing with different conditions at the beginning and during construction? Are all termites equipped with the different blueprints needed for the different conditions and stages they may have to built their mounds [2]? Does something like a mental blueprint exist, or is it merely a stimulus–response chain? These questions have long been debated in the field of animal behaviour [2]. Nevertheless, it is obvious that animal architects can build huge structures under different environmental conditions, creating structures that can consistently perform the desired functions.

An interesting point is that the animal construction is *functional* and *adaptive*. If humans need to build a functional structure in traditional ways for different environmental situations, they must design a new blueprint for each situation in application such as search and rescue. Because of variable environmental conditions, it will not be possible to build structures according to a human blueprint, or if it is possible, the structure will not work optimally. Now, consider what happens if humans employ autonomous robots to build functional structures under different environmental conditions or in unknown sites; how can robots manage this type of construction? It is certainly infeasible to have robots acting based on a fixed blueprint.

In this research, by taking inspiration from nature, we focus on the study of autonomous construction systems that can potentially build functional structures over a different range of environmental conditions or in unknown sites. In fact, as we discuss in Section 1.2, we develop, implement, and study intelligent and dynamic construction systems that take as inputs objectives and environmental features to build robust and flexible structures in a rescue application, for instance.

1.2 Main approach

Today, space research agencies are attempting to build infrastructures without human intervention, and construction companies look to robots for the potential to improve construction quality, efficiency, and safety, not to mention flexibility in architectural design. Although recent advances in the robotic offer great benefits for using robotic capabilities for many applications, autonomous robots are still in their infancy regarding their support of on-site construction because construction is highly complex. Unlike production robots that are used,

for instance, in automotive industries, autonomous robotic systems should be designed with special considerations, such as the complexity of the cluttered and dynamic working space, inaccuracy in positioning because of the nature of mobile systems, and so forth.

In this thesis, we present a new autonomous construction system by taking inspiration from animals; doing this allows robots to be robust and intelligent and able to react to unforeseen problems in unknown environments and to faults made during the course of construction. Indeed, this autonomous construction system formulates thoughts and objectives behind structures for robots' use and offers both jobs, architect and labour, to robots. This new method is *adaptive and functional autonomous construction* and we call *AFAC*.

The AFAC method gives the ability for robots to autonomously build a functional structure based on required purposes (e.g., a ventilation system for thermal regulation) compared to using a predefined construction plan (e.g., blueprint). Adaptability is another important aspect of AFAC when structures need to be built in unknown sites or in sites with different environmental conditions. In other words, AFAC does not apply any predefined construction plan because forcing autonomous robots to build a structure based on the predefined plan would increase the probability of construction failure, especially in dynamic and unpredictable situations. Conversely, AFAC provides an intelligent and robust construction system that can be established based on the desired functions instead of the construction plan.

One interesting implementation of AFAC can be seen in the context of a rescue field, where robots are used to secure an environment by performing construction tasks. Disasters can harm a lot of people, ruin structures, and spread hazardous materials (e.g., radioactive sources). The post-disaster area has potential dangers, such as hazardous materials or collapsing walls, which restrict human agents from performing rescue tasks or securing the area. In addition, the robots cannot preemptively plan to build the rescue structures before an incident occurs. Therefore, AFAC can drive robots to build structures for rescue functions, such as stabilizing large structures or protecting victims in post-disaster environments.

1.3 Goals of the study

The first goal of our research is to **develop, implement, and study AFAC methods**. We first study several AFAC principles and then implement them for the rescue application; this project develops AFAC from scratch to a level at which the robots can build autonomously in real applications.

In the development of AFAC, we consider two different approaches: local and global approaches. Most bio-inspired construction methods in robotic literature used a local strategy. Robots locally sense the environmental features and then act immediately, according to a

rule-based method; they do this without using any global representation or knowledge of the world. In contrast to this local approach, we study a novel approach based on a flexible use of global representations. This global approach consists of three phases. The first phase is exploration. A robot explores the environment to map it and discover important elements (e.g., a victim). In the second phase, the robot uses the collected data and previously built world models to autonomously compute an effective construction plan. In the last phase, the robot builds based on the computed construction plan.

The second goal of this research is to optimize a critical aspect of the global approach: the exploration phase. We therefore **design and study effective exploration algorithms that fall under specific sensor limitations**. In radiation mapping, a limitation of sensors is their relatively long acquisition time to record data. The second limitation is its poor angular resolution (AR).[1] Poor AR prevents precisely and quickly localizing targets. Both limitations significantly degrade the performance of the exploration in terms of the completion time and accuracy.

The third goal of this research is to **provide and optimize construction plans**. This construction system needs a construction plan based on collected data and defined functions. We therefore design a method to compute the construction plan for the AFAC use. The construction plan presents where and how robots should build functional structures in an efficient way.

The final goal is to **assess how efficient AFAC is for the rescue application**. This study is a detailed demonstrations both in simulation and on a real robot. This analyze also tells us how efficient the approaches used to secure the environment and victims situations.

As AFAC to rescue is totally new, this thesis has mainly an exploratory role. Several methods are designed and studied in simulation and on a real system, highlighting the potentials and problems of this new construction system. Each of these methods could be the subject of a full PhD thesis, and we hope that this thesis will inspire other more detailed research projects.

1.4 Outline

We begin this work by presenting a comprehensive review on autonomous construction with mobile robots, which is found in Chapter 2. We address and classify the relevant studies in terms of robotic activities, materials, and robotic systems. We also identify ongoing challenges and discuss future robotic requirements for autonomous construction. Chapter 3 describes experimental setup and equipment that are used throughout this work. Note that we implement and study all required experiments on a robot (the marXbot) in simulation (the ARGoS)

[1] AR refers to the minimal angle between two targets to be distinguished in a single record.

and the real-word setup. In Chapter 4, we show two different exploration methods for the rescue application and provide an in-depth analysis of each performance. Chapter 5 shows how to produce and optimize construction plans based on collected data and built models. In Chapter 6, we demonstrate an autonomous construction system building separated artifacts. The goal is to assess the precision of the construction system. In Chapter 7, we implement AFAC based on the local and global approaches for the rescue application. Finally, in Chapter 8, we summarise our contributions and conclude the work.

2 A review on autonomous construction with mobile robots[1]

2.1 Introduction

In the absence of a general consensus on a clear definition for construction, we refer to it here as the work of building by fitting parts [4] or raw materials together. In other words, it is an activity that relates to the creation of physical artifacts. Construction is also distinct from mass manufacturing, in which a product is designed for production in large quantities; products made from construction are instead large and unique in form [5]. They have to be made on sites that are temporarily unstructured and cluttered and where labourers and robots might simultaneously work in shared spaces. We limit our review to the construction of structures whose approximate shape or functionality is predictable by a human user because doing so filters out research that does not really correspond to construction. Moreover, we do not study the maintenance or decommissioning of infrastructures in this work.

Automation in construction is an interesting field that is focused on applying computer controlled processes and mechanization concepts. In other words, it deals with applying the latest automation technologies to construction subdivisions, whether in civil engineering (building, dams, bridges, etc.), architecture or in prefabrication of construction components [6]. Construction automation has been progressing to prevent worker injuries, reduce the construction process duration, and to be more cost-effective. Apart from the mentioned advantages, robots can also potentially perform construction tasks where a human presence is impossible, undesirable, or unsafe. For instance, construction in hazardous areas after natural or man-made disasters [7], construction under difficult physical conditions such as undersea or outer space locations [8], construction in areas that are not readily accessible to humans, and construction where an initial structure is required to prepare a human habitat [9]. In addition, advances in robotic systems and fabrication technologies have opened up new ways

[1] Most parts of this chapter were previously published in [3].

for architects to build sophisticated and elegant artifacts, as illustrated in 2.1.

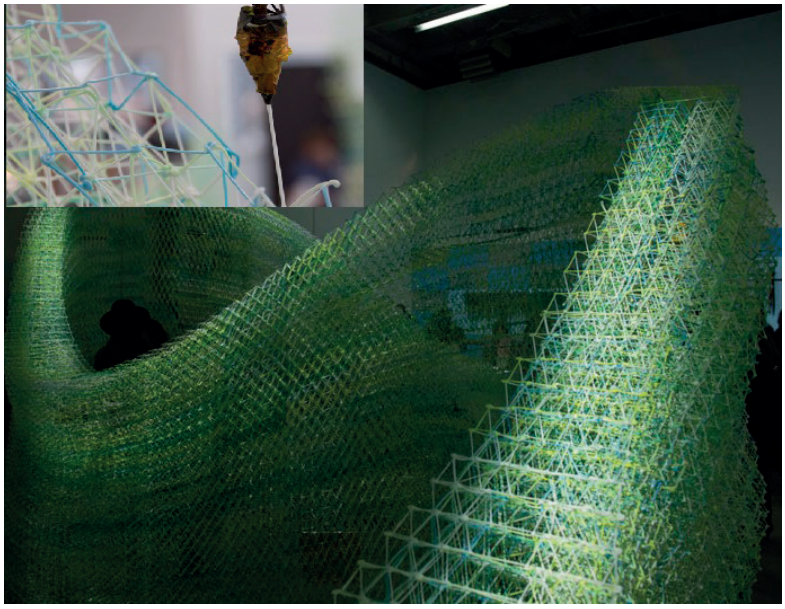


Figure 2.1 – A spatial and multi-colored mesh was printed by robotic machines, courtesy of [10].

However, autonomous mobile robots intended for construction should be designed while taking into consideration some key challenges; for instance, construction requires precise positioning, but mobile robots may have no common frame of reference regarding the construction. Construction sites are also highly complex working spaces where displacement and mechanical work requires a high level of dexterity. Moreover, one goal of autonomous construction is to prevent worker injuries; therefore, construction robots should be capable of safe interactions with workers.

Research in construction robotics and automation started in the 1980s, and since then, developments in robotic sciences have led to a wide range of robotic platforms. Because of this diversity, several construction robots were categorized as follows [5]: The first one consists of teleoperated systems, in which machines are under the remote control of humans; a human operator interprets the robot's situations and applies his or her intelligence to solve the problem, transmitting orders that are transformed into actions by the robot. The second category, programmable construction machines, enables the human operator to perform various tasks by choosing from a list of preprogrammed functions or by teaching the machine a new function. The third category consists of intelligent systems: unmanned construction robots accomplish their tasks either in a semi or fully autonomous mode. In the fully autonomous mode, robots are expected to complete tasks within a specific domain. In contrast, in the semi autonomous construction mode, a robot accomplishes its tasks with some levels of planning

made during interactions with a human supervisor.

In this chapter, we limit construction automation research to the use of autonomous (or semi-autonomous) mobile robots. The framework of the review consists of three main categories: construction activities, materials used in construction, and robotic systems. In Section 2.2.1, we study the construction activities. Section 2.2.2 is a look at the materials from various construction applications. Section 2.2.3 presents robots and robotics systems. Finally, in Section 2.3, we discuss challenges in construction with autonomous mobile robots and provide conclusions and future directions.

2.2 Research axes

2.2.1 Construction activities

Recent developments in robotic systems have led to a wide range of construction activities that are mostly based on civil infrastructure and house building; for example, a building's skeleton, erection and assembly, concrete compaction, and interior finishing in house construction [11]. There is no straightforward way to classify construction activities based on the subtasks or robotic types; however, the activities can be classified based on conventional construction processes, as follows [5]:

1. The handling process, aimed at placing solid substances together or building based on a specific construction map (e.g., bricklaying).

2. The assembling and joining process, for attaching rigid materials (e.g., welding).

3. The forming process, leading to artifacts (or environments) with specific shapes (e.g., cutting, machinery, liquid deposition, and digging).

Several robots were developed for automated handling and assembly during the last few decades. The handling process increases the building efficiency of final structures composed of many big and monolithic parts. In this category, we can find applications in which mobile robots are used to lay rigid material for construction purposes. Helm et al. [12] presented the in-situ construction using a ground mobile robot equipped with a six-DOF manipulator for a 3D structure made of bricks. In [13], flying robots built a brick-like tower by dropping down blocks one by one. Wismer et al. [14] used robots to place cube blocks (cubes with magnetic alignments and attachments) with different dimensions, creating a roofed structure. These applications could open new avenues for robotic use in civil purposes such as masonry. Masonry is time-consuming, repetitive, and labour intensive, and often results in back injury.

Therefore, it is excellent candidate for construction automation [15]. The elementary processes of masonry, such as bricklaying, were performed in a study done on the BRONCO robot [16].

Today, many companies employ robotic automation for on-site construction but only for very specific subtasks. Tiger Stone is designed for paving roads. Tiger Stone is placed in position with a remote control, and it starts to fill the site with blocks[17]. A semi-automated masonry (SAM) system is designed to work with the mason. The operator moves SAM's base, and it lifts and places each brick [18]. However, human–robot interaction is a challenging aspect because the environment is unstructured and littered with dynamic and heavy obstacles that are dangerous for humans. The proximity and vulnerability of the human in the interaction imposes strict restrictions on human and robotic activities in a shared environment [19]. Because of these and other challenges such as positioning, fully automated construction using mobile robots is not ready for commercial markets. Human workers are still, in most situations, more reliable, more efficient, and cheaper. For instance, an autonomous mobile robot will face many uncertainties and will have a hard time making the proper decision when it is laying a straight wall in a site with many obstructions, something a mason would have no issue with. Autonomous mobile robots still require additional development before they will be ready for fully automated commercial construction purposes.

The assembling and joining process is an important aspect of construction and a critical issue for mobile robot installation as well. Labourers are usually employed to manually align parts and connect them using bolts, welding, or other types of connections. These connection techniques are often not well-adapted to automatisation, pushing roboticists to redesign the connectors and joining mechanisms. In [20], aerial robots were used to construct a truss-like tower with magnetic nodes and bars. In [21], the robot moved autonomously and untethered through a truss structure to assemble and dissemble rods. KUKA MOIROS, a mobile, industrial robot system, can be equipped with advanced manipulators to handle welding processes [22].

Another activity is material shaping. This is one of the most interesting processes leading to digital fabrication. The most well-known method of digital fabrication by material shaping is additive manufacturing, also called 3D printing. An exemplary application of additive manufacturing in construction is contour crafting, which is a concrete-based layered fabrication technology developed for building a large structure in a single run [23]. Advances in robotic systems applied to the digital fabrication of large structures have opened new ways for architects to build elegant artifacts. Digital fabrication intends to fill the gaps between digital technologies and the physical construction process because design restrictions can be relaxed, allowing artifacts to be fabricated with high customization and sophistication [24]. In space applications, digital fabrication processes may be useful because space agencies could launch just the raw materials and reduce the transported volume. Volume, mass, and cost are significant factors in launching the spacecraft, so decreasing the size is very important,

particularly in space systems with large components such as antennas or panels. SpiderFab employs techniques of fused deposition modelling, using methods derived from an automated composite layup. SpiderFab will fabricate components on-orbit, enabling NASA to escape the volumetric limitations of launch [8].

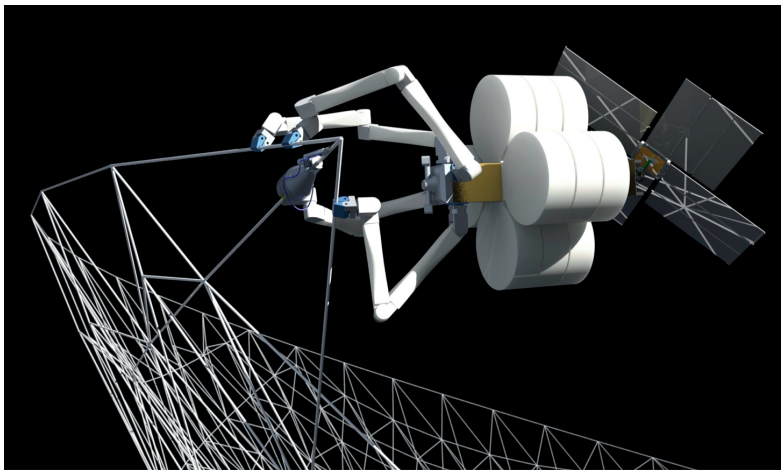


Figure 2.2 – SpiderFab fabricates a support structure onto a satellite, courtesy of [8].

Despite this rapid evolution in construction processes, most robotics systems used in digital fabrication are not mobile. Mobile robots inherently provide greater flexibility for digital fabrication because they can build artifacts that extend beyond fixed-based system constraints (e.g., the size of a 3D printer's frame constraint), but they require innovative solutions for positioning. Jokic et al. [25] used a compact and mobile head-positioning device to build 3D-shaped structures by using amorphous material deposition with mobile heads. This method allows an object to be printed independently of its size. Rétornaz [26] developed a two-level approach for the precise positioning mixing a long-range, low-precision localization with a short-range, high-precision positioning that is based on shared referential with the construction. With this approach, the author designed a special extruder mounted on a miniature mobile robot that could then deposit raw materials on rough surfaces or create free-standing structures.

In the near future, mobile robots may be used in construction, similar to how people use commercial 3D printers. The company MX3D, for instance, plans to fabricate a steel bridge based on additive manufacturing, as illustrated in Figure 2.3. In this project, robots will be used to create a bridge by welding molten metal to an existing structure while they move across what they have built. Two teams of robots will start building a bridge from opposite sides of a canal and will meet together in the middle [27]. In contrast, Napp and Nagpal [28] designed a mobile robot that is equipped with a foam tube used for depositing foam to create a ramp for inaccurate construction. The long-term goal of this application is to enable robots

to perform construction processes in emergency situations to create a passage for emergency responders to reach the target area (e.g., filling a ditch to cross it).

Figure 2.3 – Robots are going to autonomously create a steel bridge, courtesy of [27].

2.2.2 Materials

For autonomous robotic construction, the material's properties need to be taken into account, as the type of material can determine what kind of robot is needed to perform the construction process. The type of material should be based on the expected goal and will hence motivate the unique design of a robot and the related algorithms; additionally, factors such as shape and application of the structure, construction precision, construction speed, simplicity of the construction, amount of required material, and cost can heavily impact the robot's structure.

The nature of social animals provides impressive construction examples; worker ants dig into the earth to make their nests; termites build mounds with paste made out of water, sand, and clay and then deposit the mud while the mud is still wet. Some birds construct nests from small twigs and grasses without the help of binders. However, although human structures are usually more complex and need a combination of materials, simple materials are used in most of the research on robotic construction.

Figure 2.4 shows a possible taxonomy for the materials used, which confirms that the design and development of the robots has to be adapted to material's properties and target environment. The injection sprayer for creating foam needs a different design than that of an end-effector used for grasping rigid materials. Accordingly, amorphous materials can be applied by a robot with a simple sensory system and controller, but these would provide inaccurate structures. In contrast, structures made from rigid substances such as blocks or

rods are more precise. Moreover, rigid structures enable the robot to build faster structures.

Three types of materials for amorphous construction were investigated, regardless of robotic activities, in [29]: stiff prefabricated components and adhesives (toothpicks and glue), compliant prefabricated components (sandbags), and liquid depositions (casting foams). The largest expansion ratio of casting foams is an attractive point, but sufficient time is necessary to allow the foam to cure. Compliant bags comparatively need low mechanism complexity to be carried, but they have no way to expand and do not create permanent structures. Adhesive-covered objects, such as toothpicks and glue, have intermediate characteristic attributes, such as lower cure times compare to casting foams, and they have lager expansion ratios than sandbags. Soleymani et al. [30] addressed the use of deformable pockets (compliant bags) in constructing a protective linear wall. The properties of compliant bags have allowed for the use of a simple mechanism and simple controller to deposit them, but the wall is not perfectly linear. Napp and Nagpal [28] presented a model of construction to build an arbitrary shape with casting foams in unstructured environments. In [31], a mobile robot filled a ditch with two types of polyurethane foam: one- and two-component polyurethane foam. One-component foam needs one hour to cure and is expandable in a horizontal direction. In contrast, two-component foam cures within two minutes and is vertically expandable. These different properties pushed the researchers to implement two different construction algorithms. The result shows that two-component foam seems to be a more efficient material for construction purposes.

Autonomous construction is a complex process in which many failures can occur. These failures can propagate from one step to another. For instance, if a robot incorrectly grasps a block, it could destroy the built structure; thus, it is important to avoid or to correct these faults. Using self-aligning objects could be a way to decrease misalignment errors; for instance, bricks that are made from expanded foam and have physical features to achieve self-alignment and magnets for attachment [32]. In [14], foam bricks with several magnetic pins on the adjacent bricks' faces were used to build a roofed structure. Terada and Murata [33] presented a particular robotic assembler that autonomously manipulates, transports, and assembles the modules with automatic connectors. Today, companies are designing and manufacturing prefabricated components to increase construction speed and efficiency. New prefabricated components can be designed and made for robotic use in automated construction. For example, components with male–female connectors allow for automatic assembly in a more robust way [5].

Truss-like structures are composed of cube-shaped nodes and bar-shaped members. Members may be attached to create a simple cubic–lattice structure. In this way, one can build several layers on top of each other to create a tower. In [20], each face of a node had four circular slots, and there were protrusions at the two ends of each member for assembly. The magnets at

the centre of each face provided a snap fit connection. In [34], they reduced the number of magnets and the mass of the parts because the truss was constructed by aerial robots. In [34], the novel bidirectional geared rods and connectors were used to build a truss structure with female bidirectional and male bidirectional connectors.

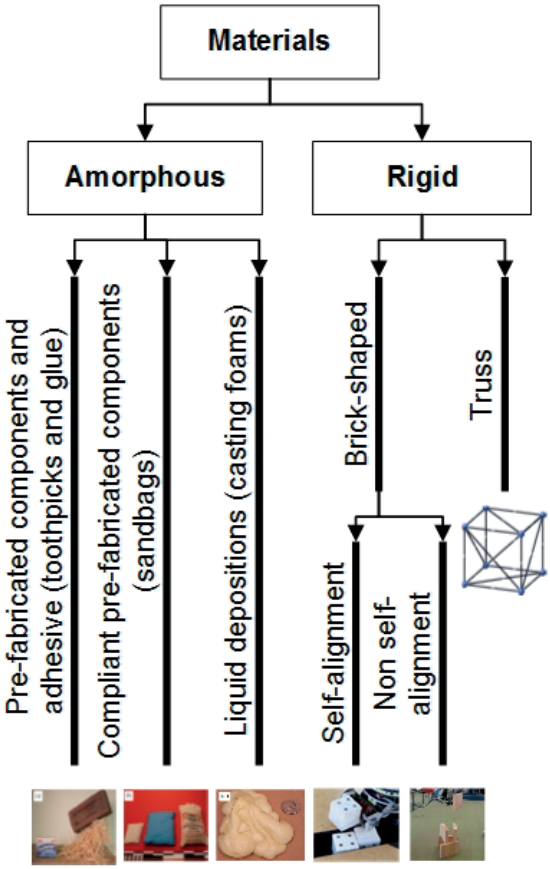


Figure 2.4 – Taxonomy of materials used in automated construction.

For parts that do not have self-alignment mechanisms, advanced robotic systems are needed to meet the requirements of construction automation. In [13], glued polystyrene bricks were carried by flying robots. A network of intercommunicating computer programs used a real-time camera system that helped the robots localize, find specific locations to pick up the blocks, and then drop the blocks off at another location. Helm et al. [12] presented dimRob equipped with an ABB manipulator. A 3D laser scanner scans the placed wooden bricks during fabrication and then sends this mapped measurement to the controller software to obtain the next commands. These examples show how the use of parts without self-alignment requires more accurate positioning solutions.

Research on the use of amorphous materials targets mainly digital fabrication, considering either continuous deposition or removal. Gershenfeld et al. [35] have addressed the imple-

mentation of this kind of material in digital fabrication. In addition to continuous deposition techniques, one can use digital materials that are composed of many discrete and self-aligning voxels that can be placed in specific locations within a lattice structure. Digital materials can open new doors for automated and coherent fabrications where functionality is integrated with the form [36].

2.2.3 Robotic systems

Generally speaking, robots have been progressing toward autonomous operations, which would allow them to tackle more complex issues such as uncertain and unpredictable situations. Construction sites are highly complex and dynamic working spaces, far from the highly predictable factory environment found in other industries such as car manufacturing. On the other hand, robots can be powerful and precise systems that reduce costs and operation time and increase efficiency. Moreover, robotic systems can be extremely flexible. In the field of construction, architects can, for instance, use these features to build fascinating and elegant artifacts, as illustrated in Figure 2.1. At present, although most autonomous construction mobile robots are at the experimental stage and far from commercialization, promising developments in the robotic field are addressing the challenges and technical limits that robots are facing in complex working spaces. In this section, we briefly study robots that have been used in the construction field and discuss the challenges with their subsystems.

2.2.3.1 Robotic platform

In the field of construction, robots are typically divided into ground robots and aerial robots. To our knowledge, there are no underwater robots used for construction. Aerial robots such as quad-rotors, a branch of unmanned autonomous vehicles (UAVs), have been developed by a considerable number of research groups. Construction systems could benefit from latest achievements of aerial robots; one potential with these robots is autonomously performing complex construction. Because accurate positioning is necessary in construction, an external localization system is employed to provide high-accuracy flight for construction tasks. Aerial robots fly to the construction point and place bricks directly in the required position without scaffolding. Structures can also be built according to highly complex designs because aerial robots move in the 3D space; therefore, they can place and manipulate material according to a precise digital blueprint. On the other hand, at the moment, most aerial robots have limited payload capabilities but several aerial robots can grasp and carry a heavy object in cooperation [36]. Another limitation concerns the aerodynamic considerations, the shape of the construction parts can affect the performance of control and stability; hence, construction parts must be designed so that they satisfy the aerodynamic constraints. At ETH Zurich, four quad-rotors were utilized to construct a brick-like tower. The positioning of the robots was

ensured by a real-time camera system that guided the robots according to a digital design, allowing the robots to pickup and drop objects [13]. Each robot used was a hummingbird quad-rotor that is approximately 55 cm in diameter, weighs approximately 500 g with the battery, and provides approximately 20 minutes of operation. The maximum payload is around 500 g. The VICON motion-tracking system was used to estimate the position and orientation of the picked-up objects and aerial vehicle locations. This system provides position feedback at 150 Hz, with marker position accuracy on the order of a millimetre. The low-level controller can execute three manoeuvres, hovering at any specified position and travelling the trajectory between any two desired points. A higher level was needed to perform the assembly task with multiple quad-rotors in coordination [20].

In contrast to aerial robots, ground robots are more stable and controllable. In addition, they can carry heavier objects that are more complex in terms of shape; although, they can hardly access each point of the construction space without a scaffold or additional tools such as a manipulator. Magnenat et al. [37] used a miniature robot to grasp ferromagnetic self-aligning blocks. They employed odometry, camera, and laser distance data for mapping; they also employed the front camera and proximity sensors to provide the required information for picking up and dropping off blocks. An extension of this work was used to build a roofed structure. In this task, they used a VICON system to estimate the position of the robots [14]. Stroupe et al. [38] presented construction by using two robotic platforms: SRR and SRR2K, both in an outdoor environment. Each rover is equipped with a forward-facing stereo camera and a four-DOF arm. A three-axis force-torque sensor on the gripper helps the rover perform manipulation used for transporting and placing rods. They used a model that is precise for manipulator positioning but may be inaccurate for world coordinates. Helm et al. [12] presented dimRob, which has a mobile base and is equipped with a manipulator [2]. It has a 2D line scanner on the mobile base, as well as a 3D scanner to detect objects. Two vacuum grippers are embedded so that it can grasp the object either from the top or the side. Unlike the other mobile robots discussed here, this robot was designed for in-situ construction. To avoid back injury and general injuries in construction, Jung et al. [39] employed humanoid robots for floor tiling. They hope that the use of this system will become feasible within the next few years at small locations where the operation is too time-consuming for a human worker.

In summary, robots are rarely used for building in unstructured environments, where many dynamic obstacles are encountered. The cluttered and unstructured nature of these construction environments limits robot mobility, manipulation, and cooperation. Therefore, automated construction needs further development to be exploited to its fullest potential.

[2] ABB IRB 4600.

2.2.3.2 Positioning systems

Construction processes always need precise positioning systems, especially where a structure has to be built based on a blueprint. Currently, the accuracy of positioning technologies ranges from metre to sub-millimetre precision levels. Depending on situations and hardware limitations, high accuracy might not always be achievable. Research shows that the required accuracy for traditional construction can be easily achieved by machines that have a fixed mechanical link with the construction and, therefore, rely on absolute positioning (e.g., contour crafting [23]). In contrast, mobile robots, by nature, do not have a fixed-referential point, and their positioning systems are not as accurate as fixed-base systems. Therefore, they need to employ external tracking systems to compensate for this shortage. The global navigation satellite system (GNSS) could be used for outdoor construction, but its precision is not sufficient for some construction activities, such as bricklaying. In addition, this system does not work for indoor space, and robots might use their own localization systems. Proprioceptive systems such as odometry, as well as inertial measurement unit systems (IMU), have accumulated and drift errors, so they are not reliable. Exteroceptive systems such as laser range scanners and cameras, could be helpful.

In [40], a mobile robot was equipped with a manipulator which had a laser range scanner. The robot sweeps its arm to create a 3D map of its surroundings. Then, the robot finds its location by comparing this map with an initial scan of the environment. By updating a map based on the CAD model of the structure, the robot is able to make adaptations during construction. Elapsed time is one challenge of this method because the robot needs significantly more time to build a small brick wall. A similar robot, dimRob, has successfully built a brick wall. The robot moved and localized itself based on the CAD map and two metal disks as markers. With each step, the robot was fixed and supported by side-hinged telescopic outriggers. In fact, dimRob was anchored to the ground, which prevented the robot from moving many times during construction. It must be also repositioned manually in each step [12]. Rétornaz [26] used a two-step methods, depositing part of the material in the first step, then measuring the positioning of this first deposition to recalibrate the whole system and performing the final deposition with high precision. Ardiny et al. [41] presented an autonomous construction system for building separated artifacts with simple blocks. The approach was based on the combination of a self-positioning system, simultaneous localization and mapping (SLAM), and short-range relative localization system to build coherent artifacts.

External cameras, such as motion capture systems, provide the precise position of the objects. As mentioned earlier, some studies used this system to localize robots [13, 14, 20]. Additionally, inaccurate external systems, such as a global positioning system (GPS) can be used for some construction activities. In [42], an autonomous excavator was equipped with a GPS receiver and IMU was targeted to shape the entire construction site using mobile excavation. To

achieve this task, in addition to the positioning system, there needs to be a path-planning algorithm that is an extended A* path-planning algorithm. However, a precise self-positioning system is still generally a challenge for autonomous mobile construction systems. If robots were to have better self-positioning systems, they could build sophisticated artifacts, as well as 3D printers that do not have the printer-size constraint.

2.2.3.3 Bio-inspired approach

Animals not only have artistic aspects, but they also consider functional features, such as ventilation, temperature regulation, multiple escape routes, and structural strength. For instance, a study on termite mounds showed that termite nest construction processes were influenced by thermoregulation and gas exchange, which was a reason behind the different mound architectures [43]. Nests may be built by individuals or by social animals working together. The construction activities of social insects show how a complex structure can emerge from the actions of many independent workers that are using simple rules and local information, even if there is no experimental data to prove that something like mental blueprints are used by a single insect [44, 45]. Some researchers believe that animals use a mental blueprints; however, other researchers believe that animals build a structure based on local interactions [2].

Werfel et al. [32] presented a ground mobile robot (TERMES) to perform automated construction inspired by the building activities of termites. The robots use passive solid building blocks as landmarks for local interactions. The goal of the research was to use insect principles to build a user-defined structure for human purposes. An off-line compiler generates traffic rules depending on a user-defined blueprint, and then, robots have to follow these rules during construction. Soleymani et al. [30] used two biological mechanisms, stigmergy and templates, to guide a robot. The robot had to deposit sandbags to build a protective wall without relying on a central planner, an external computer, or a motion capture system. The interactive system is another approach in which agents use not only environmental feedback but also two-way dynamic feedback from the environment. This means that the agents change the environment while simultaneously the environment impacts the ongoing actions, generating a two-way feedback loop used to construct structures based on functional blueprints [46].

Indeed, bio-inspired construction principles and human architecture have fundamentally different approaches. Humans build structures based on a blueprints, and the construction processes are centrally driven by the plan. To follow this approach, robots must have a global representation of the environment and precise positioning systems to be able to build a structure based on pre-specified blueprints in a known environment. In contrast, in bio-inspired construction, agents perform tasks in a decentralized, self-organized manner. Bio-inspired approaches are elegant because simple mobile robots are able to run autonomous

construction by following compiled rules and then performing reactive algorithms based on the local approach. Each individual can act independently; interaction between them and the interactions of each agent within the environment ensures an autonomous construction without a conventional blueprint. Compared to human strategies, the bio-inspired approach can be more robust to failure because of its decentralized methods, which can be very flexible and even include self-repair mechanisms.

(a) Termite mound (b) Termite-inspired robots are building a tower

Figure 2.5 – The left photo is a cathedral termite mound in the northern territory of Australia; photo by J Brewtermite. In the right image, robots try to construct a tower based on a bio-inspired method, courtesy of [32].

We plan to develop a new method based on the bio-inspired approach that can be used for human applications (e.g., rescue), and we call it AFAC. In this new method, autonomous robots build structures that need to be adapted to environmental features for desired functions and do so without using any blueprints. Forcing autonomous robots to construct a structure based on a predefined construction plan will increase the probability of construction failures because of uncertainties in construction environments and inaccuracies of measurement systems (e.g., positioning systems). In contrast, using an autonomous construction system will increase the capabilities of robots to perform autonomous construction. Relaxing the constraints coming from a human blueprint (or human plan) by using AFAC allows robots to be intelligent about the construction process.

We organize and develop the AFAC method based on two different approaches: the local

and global. The most bio-inspired construction methods in robotic literature presented here utilized the local approach (e.g., [46]). Robots locally sense the environmental features and then act immediately according to a rule-based method, all without using any global representation or knowledge of the world. In contrast to a local approach, AFAC based on the global approach can include a reasoning system to analyze the important environmental information from global representations, then driving the robots to build efficiently by making a series of global decisions. AFAC based on the global approach is beyond state-of-the-art.

2.2.3.4 Multi-robot systems (MRSs)

MRS is a relatively new field focused on the control of and collaboration between robots, which can either be homogeneous or heterogeneous robots. In fact, a remarkable characteristic of MRS is the ability for robots to work with one another to reach a common goal. Robots can have similar or different tasks depending on their roles and environmental conditions. Several researchers have studied MRSs which take inspiration from social animals, such as bees, ants, fish, or birds [47]. MRSs have some advantages compared to using a single robot, including parallelism, robustness, and a low-cost of operation [48]. The studies in MRS have showed activities that can be used in construction, such as object clustering and material assembling, collective transport of material, and collective decision making, to allocate the robots to different subtasks of the construction process [49].

Some research presented the construction of specific structures whose shapes were fully prespecified and requested by a user who provided only a high-level description. Werfel [50] demonstrated in a simulation a method by which robots are able to build two-dimensional structures of desired shapes by using blocks. A robot acts as a stationary beacon and leader. Many robots stand on the corners. Other robots then build linear or curved walls between the corners. The leader also provides information about the building process of this structure for other robots. In another study, Werfel et al. [51] presented 3D collective construction, in which large numbers of autonomous robots built large-scale structures. Robots were independently controlled and coordinated their actions implicitly through manipulation of the shared environment.

Some researchers explicitly drew inspiration from biological concepts such as stigmergy. Parker and Zhang [52] presented a swarm construction algorithm to control robotic bulldozers in the creation of a clear region in a field of gravel (nest). Robots used a technique known as blind bulldozing, which was inspired from an ant-nest-building strategy. These robots use the minimal amount of sensory and mechanical resources required by the algorithm. They clear away debris to build their circular nests. Werfel and Nagpal [53] presented algorithms by which robots build user-specified structures without human intervention. Robots apply

the stigmergy concept and are independently deployed to collect square blocks. In another work [54], they presented algorithms for the adaptive construction of structures. The shape of the final structure can be defined by its environmental elements. For instance, a team of robots may be charged to build a protective barrier of a given thickness around a hazardous chemical spill. In contrast, some construction algorithms use an external guide. Melhuish et al. [55] reported wall building by groups of robots; this wall building was inspired by nest construction behaviours in ants. Two templates were used by the robots to build their wall.

We can briefly review research of MRS for construction from the control architecture point of view. The most common architectures are centralized, decentralized, and hybrid. In the centralized architecture, a single control agent leads the entire team. The centralized system has a global plan to guide the robots in performing tasks; it is suitable for situations in which the leader (or main agent) can observe robots, and easily broadcast commands to the whole team [56]. In [13, 20], building a particular structure based on a centralized system was the final goal, so a team of UAVs assembled structures using simple objects. In contrast, the decentralized architectures does not depend on a leader agent and all agents are equal with respect to control, so it is robust to failure [57]. However, the problem of this control system is that it may not perform construction as coherent as a centralized system. As already described, in bio-inspired construction that can be one type of a decentralized system, agents perform tasks in a decentralized, self-organized manner [30, 32]. Decentralized control systems were also used in other autonomous construction research [51, 52]. The last approach is the hybrid control architecture that combines local control with higher-level control approaches to benefit from advantages of both centralized and decentralized control systems. This system improves robustness and influences the entire team's actions through global goals [58]. In this thesis, AFAC based on the local approach is used a decentralized control architecture. AFAC based on the global approach is similar to the hybrid control system but the global plan is created or updated by an autonomous system instead of using predefined goals or plans (e.g., blueprint).

2.3 Challenges and conclusion

2.3.1 Challenges

1. Autonomous construction requires robots to make decisions in reaction to rich sensory input. These decisions are difficult because of the unstructured nature of construction environments coupled with the unpredictability of physical interactions with some materials. Much of the work done on autonomous construction sidesteps this challenge, either by giving up on construction precision or by imposing unrealistically pristine configurations on the environment. For robots to be eventually used in fully auto-

mated construction sites, there is a need to adapt more sophisticated decision-making techniques that treat autonomous construction with the richness it deserves. In particular, there is an absence of construction planning methods that model uncertainty in robots' actions and of reasoning methods that clarify complex construction situations.

2. Existing construction processes need precise positioning, which can be achieved by machines that have a fixed mechanical link with the construction site; therefore, they rely on absolute positioning because of the common reference frame. Mobile robots, by their very nature, do not have a fixed-referential point, and their positioning systems are not as accurate as fixed-base robots. Therefore, they need to employ external tracking systems (e.g., camera, GPS) or supplementary methods. The precision of the current self-positioning systems that are not linked to the built structure is not sufficient enough to support construction processes in unknown environments; therefore, mobile robots have to employ new technologies to progress in this domain.

3. As discussed, each robotic platform has its own restrictions regarding the functionality and versatility of autonomous construction. The physical characteristics of a robot may not allow it to handle a complete construction process. Depending on the shape, type, and size of a structure or environment, there needs to be specific robotic behaviours that cannot be handled by an autonomous mobile robot. Therefore, there needs to be an improvement in the versatility of construction robots, the utilization of a group of heterogeneous mobile robots to handle several disparate situations, or the reliance on human–robot cooperation.

4. To the best of our knowledge, collaboration between autonomous mobile robots and human workers in construction has never been studied; although, some studies address the use of semi-autonomous robots for on-site construction. Collaboration between labourers and autonomous mobile robots (even if only in close proximity) has enormous potential but introduces new challenges, especially in terms of safety.

5. In assembly processes, robots are usually expected to align parts and connect them using bolts or welding or by assembling prefabricated components. The problem is that the specifications for tolerances are not always achievable in practice because of the reshaping of a structure's components during construction. One example is an anchor rod used as a fastener connects the foundation and the structural steel columns. Labourers put rods inside a concrete foundation before the concrete dries. Hence, the rod pattern might not match the hole pattern of the corresponding column; this

problem may result in assembly failures. In a real situation, human workers will fix these problems rather than wait for replacement components to be fabricated and delivered because most construction projects are under tight schedules [5, 59].

In autonomous construction, the goal is to increase productivity, and waiting for new components decreases the speed of the process. If robots are to one day replace human construction workers, new methods should be developed to tackle tolerance problems during construction.

6. Today, companies are designing and manufacturing prefabricated components to increase construction speed and efficiency. New prefabricated components can be designed and produced for robotic use in automated construction. For example, components with male–female connectors allow for increased robustness in automatic assembly [5]. Additionally, adapting the design of gripping mechanisms to the components' shapes can yield a more efficient and precise automated construction.

7. Autonomous construction consists of sequential and repetitive tasks that can be executed by a group of robots, but the field of MRS is still not mature enough to be used in real construction applications. One reason for this is that the variety of construction tasks require heterogeneous robots to work together to build a structure. Dealing with heterogeneity, and determining how to design and optimally integrate a robot team working in a shared space with shared materials, is an ongoing research challenge.

8. When a construction process consists of a sequence of tasks to be performed by robots, task failures may occur from one step to another, requiring robots to address the failures caused by previous steps. Therefore, the reliability of robotic systems in interaction with faults or failures is another challenge.

9. Automated construction inherited some challenges from autonomous robots. For instance, uncertainty in sensing, reasoning, and acting are critical factors that impact robot performance.

2.3.2 Conclusion

Construction automation is progressing to improve the quality and flexibility of construction; it has the potential to be applied where human presence is impossible, unsafe, or prohibitively

expensive. Among the several possible approaches, autonomous mobile robotics seems to have the greatest potential, but it also introduces many challenges. Construction presents conditions that are difficult for robotic systems: the environment is particularly cluttered, unstructured, and may require collaboration with human workers.

In this chapter, we have presented the existing research on automated construction with mobile robots from different perspectives. First, we clarified which kind of construction is discussed, as construction can refer to wide range of elementary processes. We carefully defined autonomous construction based on what has been done in this field, to help focus on the promising areas of research and categorize the robotic activities that deal with construction operations. We described the different material types used by robots. Materials influence robots' design and the construction algorithms because of the their properties. Additionally, we looked at some bio-inspired research that aims to mimic the construction behaviours of animal architects. We also looked at robots and related auxiliary systems from a hardware point of view. In particular, we reported studies conducted on ground robots and aerial robots and described auxiliary systems such as external cameras used to help robots tackle uncertainty and positioning. The ultimate goal in this field seems to be construction performed by a group of robots and taking advantage a distributed heterogeneous approach, but the complexity of the task and system has pushed researchers to target only simple multi-robot construction scenarios, or to treat robots independently to decrease complexity. Therefore, autonomous robots are still far from being employed in commercial construction.

The industry still hopes to eventually reach a technological level that will allow ones to drop robots off at a site and come back several months later to see a enormous and fantastic building. Although this is far from current robotic capabilities, it is clear that research is progressing across this highly interdisciplinary field, in an effort to provide solutions to the demand for robots in construction.

In this thesis, we advance state-of-the-art robotics by developing and implementing AFAC, in which autonomous robots are employed to build a structure that can be adapted to different environmental features for targeted functions. We take inspiration from animals construction; the idea is to relax the constraints associated with a predefined plan (i.e., blueprint), allowing the robots to be intelligent about construction. The intelligent construction system also allows the robots to be more robust and flexible when encountering environmental conditions and faults made during the course of construction; these problems can be solved with renewing construction commands, which is valuable for autonomous construction. Moreover, in this work, we introduce the idea of autonomous construction for rescue applications. In these situations, the robot can stabilize large structures and/or protect victims by performing construction tasks. To our knowledge, there is no previous work about the use of autonomous construction for rescue applications.

3 Experimental setup and equipment

As briefly described in Chapter 1, we aim to employ robots for autonomous construction in a rescue application. The robots are designed to build protective structures in an environment after a nuclear disaster. Here, we introduce the setup, robotic platform, simulator, and middle software that will be used throughout this work. The major features and important characteristics of each one (e.g., the setup) is also described.

In Section 3.1, we focus on the equipment presented in literature or commercial markets that is used for the detection of victims or radioactive hotspots. In Section 3.2, the robotic platform and its middle software are described. In Section 3.3, we introduce the simulator. Section 3.4 shows how the detection systems are implemented in the robotic platform. Section 3.5 shows the ROS platform and ROS libraries that are used. Finally, the characteristics of the test-bed are presented in Section 3.6.

3.1 Detection systems

Normally, after a nuclear disaster, rescue robots must search the environment to find victims and the radioactive hotspots scattered in the post-disaster environment. In this section, we focus on the specific tools and technologies that are required for detection in a rescue operation.

3.1.1 Radioactive hotspot detection systems

The detection and localization of radioactive sources is a main issue for the nuclear industry and for security applications (e.g., post-accidental interventions) [60]. As we will discuss in Section 4.2.2, using a directional detector, such as a gamma camera, is one interesting method for radioactive hotspot detection and localization. In this research, the robot is supposed to be

equipped with a gamma camera for the global approach, so we will briefly study the principles and technologies in gamma cameras.

Gamma cameras are currently stablished based on two technologies: Compton scattering and coded masks [60]. The Compton camera usually measures the position and energy of the incident photon where the Compton scattering occurs; the scattered photon is then absorbed. As shown in Figure 3.1, a photon emitted from a source hits the middle plane with the initial energy E_0 and scatters with energy E_1. The scattered photon is assumed to be fully absorbed within the bottom plane. The axis and angle (Equation 3.2) of each incident photon can

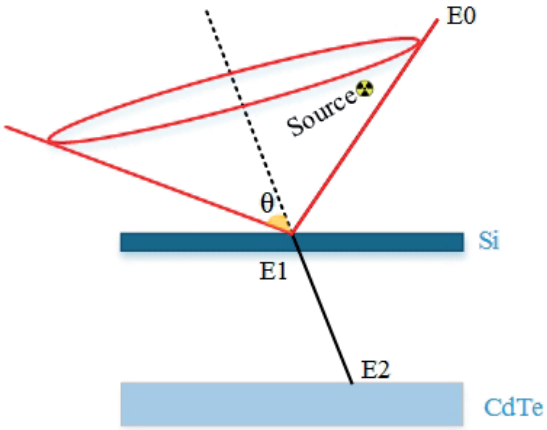


Figure 3.1 – A schematic of a Si/CdTe Compton camera; in which a wave is scattered through the middle semiconductor detector (Si) and then absorbed by the bottom semiconductor detector (CdTe). The red cone shows possible gamma ray source locations; its intersections with the next cones will determine the final position of the source.

an imaginary cone, so the intersections of the rebuilt cones show the source location. The problem regarding the real setup is that locations are broadened because of real detector size, finite energy resolution and Doppler broadening; so by means of Monte-Carlo simulations, Compton imaging has been optimized to better show sources' locations [61, 62].

$$E_0 = E_1 + E_2 \tag{3.1}$$

$$cos(\theta) = 1 - \frac{m_e c^2}{E_2} + \frac{m_e c^2}{E_1 + E_2} \tag{3.2}$$

Where

θ is the scattering angle, E_1 is the energy of photon when it hits the middle layer and E_2 is the energy of the absorbed photon in the bottom layer.

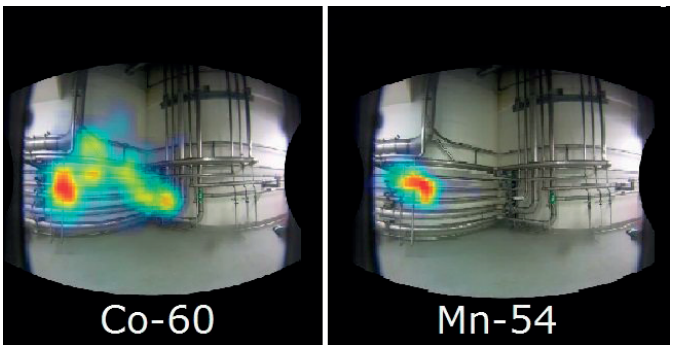


Figure 3.2 – An image generated by a Compton camera (Polaris-H) shows Co-60 and Mn-54 in the various pipes, courtesy of H3D Inc.

Since 1958, considerable progress has been made to improve gamma cameras based on coded masks [63]. The GAMPIX is an example of this type of camera. The GAMPIX consists of three components: the detection system, that was hybridized with a 1 mm thick CdTe substrate called Timepix; a coded mask that was used instead of a pinhole collimator to increase the sensitivity of the camera; and finally, the USB module that enables this gamma camera to plug into an acquisition computer. The final system is a compact package and has weighs less than a Compton gamma cameras (Table 3.1).

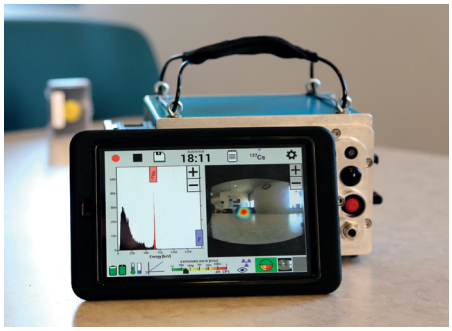
Figure 3.3 – An image that was made by a iPIX camera shows a hotspot inside a nuclear waste package. The measurement time is 1 sec and the dose rate near the hotspot is about 300 $\frac{\mu Sv}{h}$, courtesy of [64].

3.1.2 Victim detection systems

Currently, rescue robots are relatively new in the field of robotics, and they are used as assistants for human rescue teams in hazardous and restricted environments. One service that can be supported by robots in victim body detection. Detection of a victim is a complicated task, especially in an unstructured environment that has an additional challenges such as fire.

(a) iPIX, courtesy of CANBERRA Company

(b) AISense, courtesy of AISense



(c) Polaris-H, courtesy of H3D Inc

Figure 3.4 – The selected commercial directional detectors.

Table 3.1 – Specifications of the selected commercial directional detectors

Detectors	Type	FOV (deg)	AR (deg)	Acquisition time (sec)	Dimensions (cm)	Weight (kg)
Polaris-H	Compton (CZT)	360	20-30	30<	21×19×13	4.04
HSL-Lite	coded mask and dynamic imaging mask	60	<10	900<	26.1×20.4×33.7	6.5
Toshiba	-	60	-	-	38×11×24.1	9.8
iPIX	GAMPIX coded mask	45-50	2.5-6	1<	10×10×19	2
AISense	hotspot locator (without camera)	360	-	0.1	13×14.5×14.5	2.2

Robots employ different kinds of sensors and methods to detect live human bodies. Their sensors work based on live-body characteristics, such as temperature, voice, motion, body shape, skin color, and breathing [65].

Most related research has addressed vision systems for detecting human presence. The vision systems can include different kinds of vision sensors, such as color camera [66], stereo vision [67], and infrared cameras [68]. Other types of sensors, such as heat sensors [69], microphones [70], and CO2 sensors [71], were also used to detect heat wavelength, sound, and the carbon dioxide emission, respectively, of a live body. In [72], a microwave life-detection system was used to locate victims who were buried under rubble; the system analyzes reflected microwave signals for extracting breathing and heartbeat signals.

Inasmuch as each sensor may detect different physical phenomena and may have different weaknesses, depending on the environmental conditions, reliable results cannot be found with only one kind of sensor. Therefore, some research employed a combination of sensors to improve the reliability and robustness of the final results [65, 73].

3.2 The Robotic system

In this section, we introduce a robot that is employed throughout our experiments; it is a miniature and modular robot called marXbot. The marXbot was developed by the Laboratoire de Systèmes Robotiques (LSRO) at the École Polytechnique Fédérale de Lausanne (EPFL) [74, 75].

3.2.1 Robotic modules

As illustrated in Figure 3.5, the marXbot used for our experiments consists of five independent modules. From bottom to top, they are as follows: a base module for mobility of the robot (Section 3.2.1.1), a gripper for manipulating the objects (Section 3.2.1.2), the range and bearing system for discovering the relative position of other surrounding robots (Section 3.2.1.3), the main board and front camera (Section 3.2.1.4), and a distance scanner module to do SLAM (Section 3.2.1.5).

3.2.1.1 Base module

The base module that drives the marXbot is made of of two treels, which is a combination of a track and a wheel. Two geared DC motors are linked to the treels to produce the robot's motions. Depending on user's choice, the PID controller acts on the current, velocity, or position to drive the motors. The treels help the robot manoeuvre through rough terrains.

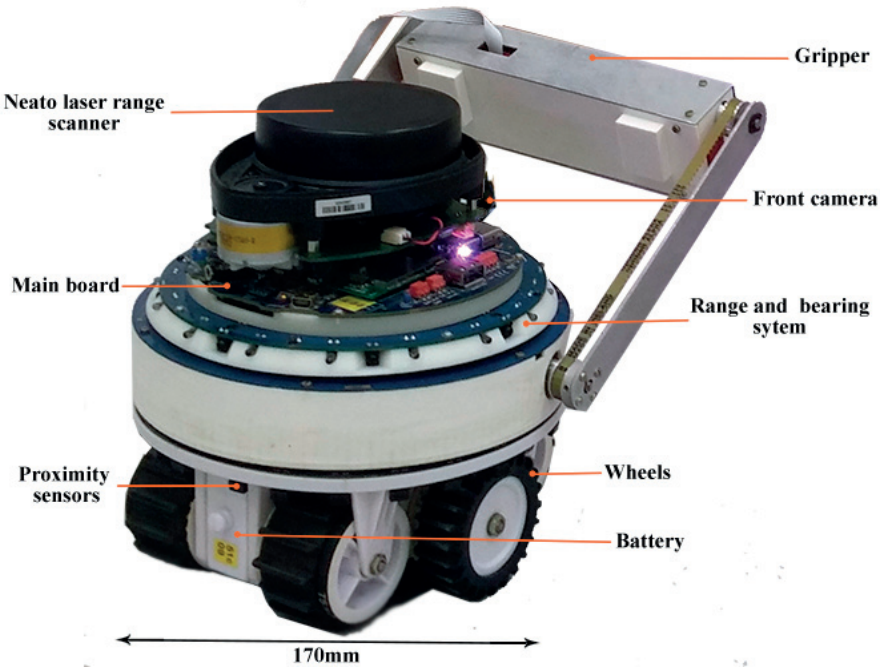


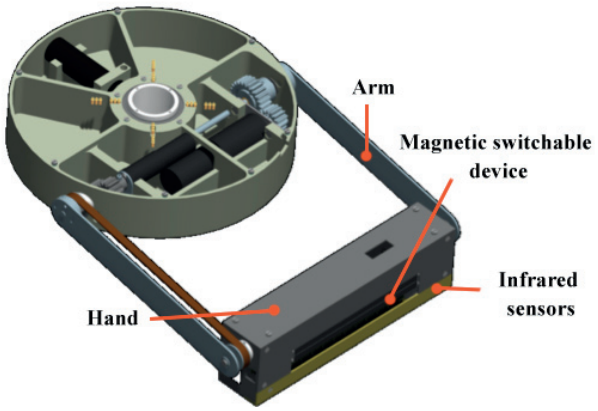
Figure 3.5 – The marXbot, with five its modules, that was used for our experiments.

The base module has a gyroscope and an accelerometer that can measure the orientation and acceleration of the robot. Twenty-four short-range infrared proximity sensors are embedded around the robot to detect and avoid obstacles. The 38 Wh lithium polymer battery can supply energy up to 7 hours of continuous operation [76].

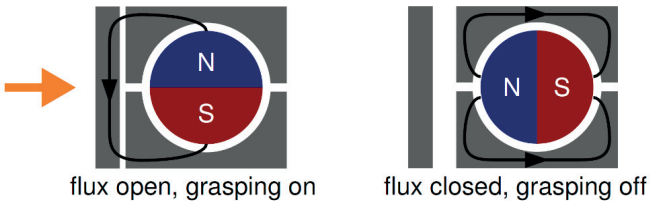
3.2.1.2 Gripper

This module (Figure 3.6a) is designed to grasp, displace, and carry small objects. It has a magnetic switchable device that can grasp or release a ferromagnetic object. In this device, a permanent magnet rotates inside two pieces of metal to conduct a magnetic flux. As illustrated in Figure 3.6b, the gripper grasps external ferromagnetic objects when the flux is open; otherwise, it cannot attract external objects.

The arms have three degrees of freedom: the rotation of the gripper around the main body, the lifting movement of the arm, and the tilt of the hand. The position, speed and torque of each degree of freedom can be controlled. Twenty-two infrared proximity sensors are placed on the head (10 in front, 10 under the head, and two sensors on each side of the gripper). These sensors allow the marXbot to detect objects for grasping at a specific location of the gripper, for instance.



(a) The CAD version of the gripper, courtesy of LSRO



(b) The magnetic switchable device, courtesy of [76]

Figure 3.6 – The magnetic gripper module.

3.2.1.3 Range and bearing system

The range and bearing system is based on infrared sensors that provides communication between the marXbots. With this module, marXbots can exchange 12 bytes of data every 100 ms and get the relative positions of the neighbouring robots at a distance of up to 5 m.

3.2.1.4 Main board

Although the modules have their own microcontrollers and work independently,[1] the main board is necessary for managing the communication between the modules and an output (e.g., external PC). In addition, sensors such as the camera, that needs high bandwidth and require large amounts of computation can be supported by the central computer of the main board.

The central computer is based on a 533 MHz Freescale i.MX31 application processor, 128 MB of RAM, and 32 MB of NOR flash memory. It is assembled on the main board that provides the necessary connections to other peripherals and the main processor (Figure 3.7). The processor runs a lightweight Linux distribution. A 3 megapixels camera is installed on the board, and it

[1] Section 3.2.2 shows a global overview of the electronic architecture.

is linked directly to i.MX31 [76].

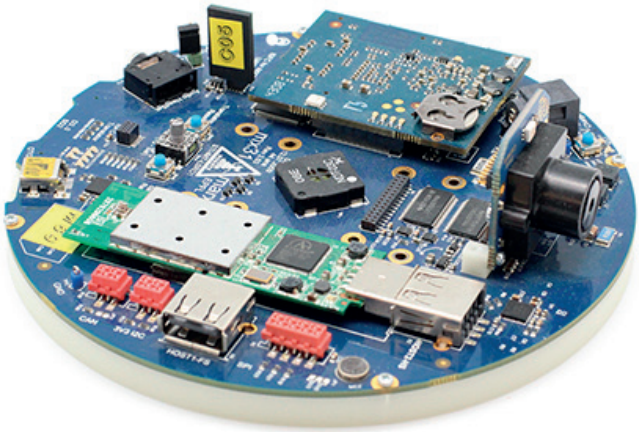


Figure 3.7 – The top view of the main board with the i.MX31 processor.

3.2.1.5 Laser range scanner

A laser range scanner is a sensing system that measures the distances to surrounding objects with laser beams and produces 2D- or 3D-point cloud data for mapping, localization and environment modelling. For the marXbot, the Neato laser range scanner, which originally belonged to a robotic vacuum cleaner, [2] was employed. The main characteristics of this sensor are that it is compact and very cheap ($30) compared to other laser range scanners. It incorporates a laser, CMOS imager, and electronics to compute distances based on the triangulation of a 650 nm laser beam. The laser scanner performs a 360-degree laser scan with an angular resolution of 1 degree. It can detect objects placed within a 6-meters, and its accuracy is 3 cm at 6 m [77].

3.2.2 Electronic architecture

The marXbot consists of several modules that work independently of each other. Each module includes a core of one or more microcontroller(s) and peripherals (sensors and actuators) that help it perform special tasks. The microcontroller controls the module and communication with other modules, as well as the main board. For instance, the DC motors of the treels are controlled by the generation of a pulse width modulation (PWM), the analogue signals of infrared sensors have to be converted into digital data.

Using a single, central computer instead of many microcontrollers creates a large and complex

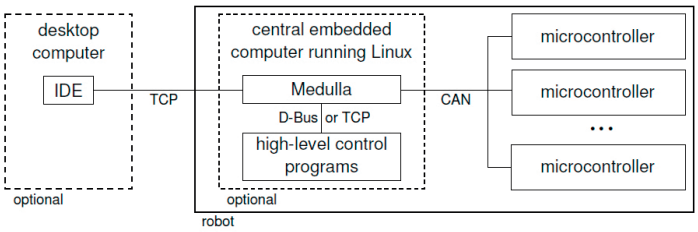
[2] It is developed by the Neato company.

wiring system because the central computer must be connected to the different peripherals. In addition, the central computer may not provide enough interfaces, such as timers, analog-to-digital converters, or input–output pins [76]. Using independent microcontrollers for modularized parts reduces wiring because the microcontrollers are placed on the modules that are the closest to the peripherals. Using a multitude of microcontrollers also distributes the computation efforts, memory needs, and other hardware aspects such as timers and pins.

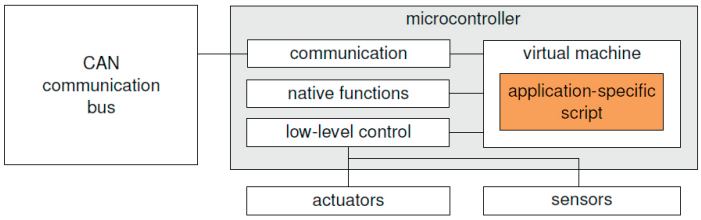
To connect microcontrollers to each other and to the main board, a controller-area network (CAN) bus is embedded in the electronic architecture. This provides a shared communication bus to facilitate the exchange of data between modules and with the main board. The microcontrollers are also networked by an inter-integrated circuit (I2C) bus that is used by the main board's i.MX31 to perform the power management tasks, for instance, waking up the modules.

3.2.3 Aseba

Aseba is an event-based architecture developed by LSRO to address the programming and running of the distributed microcontrollers network. The core of Aseba is a virtual machine (VM) placed inside each microcontroller; the core runs user codes and provides communication through events. Because the modules work independently from each other, Aseba is designed to support decentralized communications between microcontrollers. Thus, events that are defined by Aseba allow microcontrollers to transmit data on the CAN bus (Figure 3.8) [76]. The user code can be written in a user-friendly scripting language and using an integrated



(a) A typical Aseba network in marXbot



(b) A microcontroller in the Aseba network

Figure 3.8 – The scheme of Aseba on the marXbot, courtesy of [76].

development environment (IDE); then, the code will immediately be loaded onto VM with

just a click.

3.3 ARGoS simulator

ARGoS is used to model and simulate large-scale swarms of mobile robots. It was initially developed under the EU-funded Swarmanoid project[3] that aimed to study coordination between different kinds of robots. One of these three robots was the marXbot.

With ARGoS, users can customize the environment, add a new feature to the robot to enhance its functionality, and do so much more. Functionality can be added in the form of new sensors, actuators, and robot components. The users can also determine environments' shapes or the properties of each object, such as shape, color, or mass, or they can modify visualizations, including physics engines [78]. Figure 3.9 shows the global architecture of ARGoS that was founded on a modular approach. All modules are plug-ins that can be overridden for maximum flexibility [78].

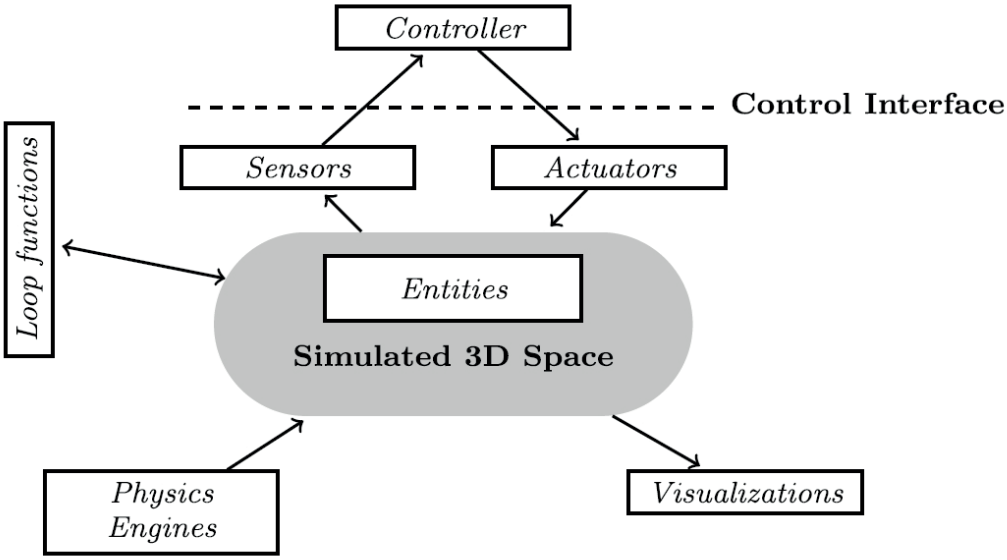


Figure 3.9 – The architecture of ARGoS, courtesy of [78].

We have chosen ARGoS for performing our experiments, because ARGoS was developed based on the marXbot to run robotic experiments; in addition, it is fast and customizable for specific experiments.

[3] www.swarmanoid.org.

3.4 Implementation of detection systems on the real and simulated robot

We have presented the detection system, marXbot, and the ARGoS simulator. Now, we model and implement the detection system on the robot. Sections 3.4.1 and 3.4.2 describe how to implement the radioactive hotspot and victim detection systems with the specifications presented in Table 3.2.

Table 3.2 – Specifications of detectors are used in our experiments

Detectors	Target	Range (m)	FOV (deg)	AR (deg)	Acquisition time (sec)
Gamma camera [4]	Source	5	360	2.5	30
Front camera	Victim	1	45	-	0
Laser range scanner	Walls	5	360	1	0

3.4.1 Radioactive hotspot detection system

In compliance with our restrictions required for providing a safe test-bed in the laboratory, we could not use real radioactive materials. Therefore, we replaced them with materials that have similar behaviours. The customized gamma camera is also modelled alongside the new radiation matter.

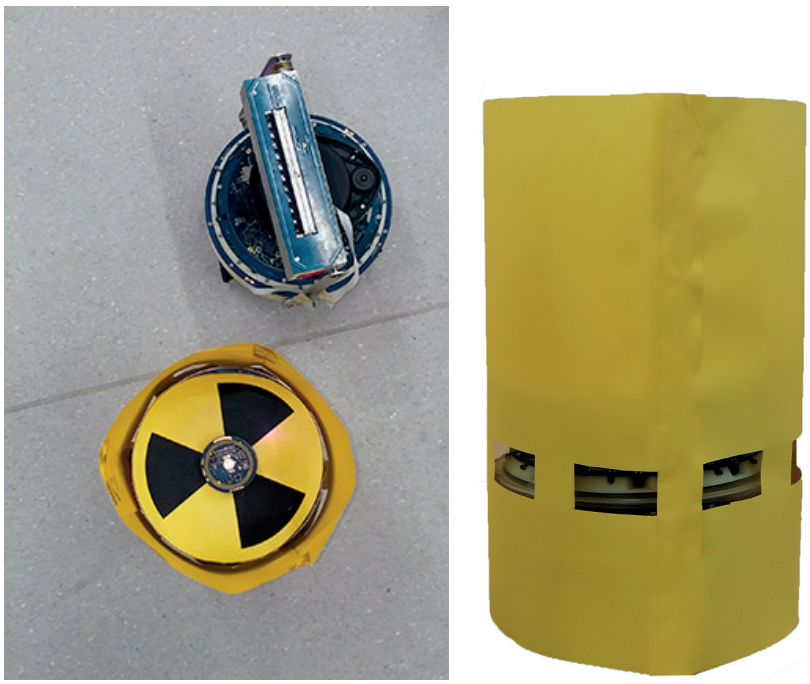
In the real-world setup, sources are other marXbots, in which case the range and bearing systems send infrared rays instead of radioactive waves. All sources are supposed to emit radioactive waves with the same intensity; the intensity is inversely proportional to the square of the distance. An object that is close to a source absorbs $0.01\frac{Gy}{s}$ radiation. The range and bearing system of the main marXbot acts as a detection system. It can estimate its distances to sources and then compute the corresponding intensity. In the simulation, sources are lights with the same intensity, which are likewise inversely proportional to the square of the distance. A series of light sensors is defined around the robot to model a directional detector.

As discussed, despite the fact that much progress has been made with gamma cameras, there are still some limitations in current gamma cameras for autonomous exploration. The first limitation is the relatively long acquisition time, which is the main challenge for an effective exploration in an unknown environment.[5] The second limitation is AR[6]. Most gamma cameras have bad ARs, for instance 20 degrees or more. Therefore, if a gamma camera localizes a

[4] This customized gamma camera is not compatible with the gamma cameras that were introduced in Section 3.1.1. One solution might be to equip the robot with an array of gamma cameras to achieve this customized gamma camera, for instance. The effects of the gamma camera parameters will be studied in Section 4.6.2.

[5] In Chapter 4, we will discuss exploration methods in more detail.

[6] AR (angular resolution) refers to the minimal angle between two radioactive sources to be separately distinguished in a single radiation image.



(a) Top view (b) Side view

Figure 3.10 – The source is another marXbot in which the range and bearing system sends infrared rays instead of radioactive waves.

hotspot far from the robot's place, there may be other possible sources around it. More specifically, the red-dotted line in Figure 3.11 illustrates an imaginary AR district. In this district, the gamma camera will not be able to distinguish separate hotspots, so there might be more than one radioactive source here.

Regarding the AR challenge, we guessed some source candidates in the contaminated area and within the AR district so that the distance between them should be higher than d_s[7]. Once the gamma camera detects a contaminated region, the source candidates are marked on the region by the laser scanner. The laser scanner helps the robot estimate hotspots' locations in the robot coordination system. Afterwards, the robot applies particular methods to find real radioactive hotspots among the source candidates; for instance, it can move near them to get a better estimate of the sources through close-up investigations.

In addition, we need to define some mechanisms that are defined to approve or discard a source candidate. First, each source candidate takes a particular variable that indicates the probability of it being a real source; we called this `support`. Then, three rules are defined to update `supports`, as follows:

[7] d_s is a user specified value that is required as a minimum distance between two separated sources. It determines the required accuracy for the radioactive hotspot localization.

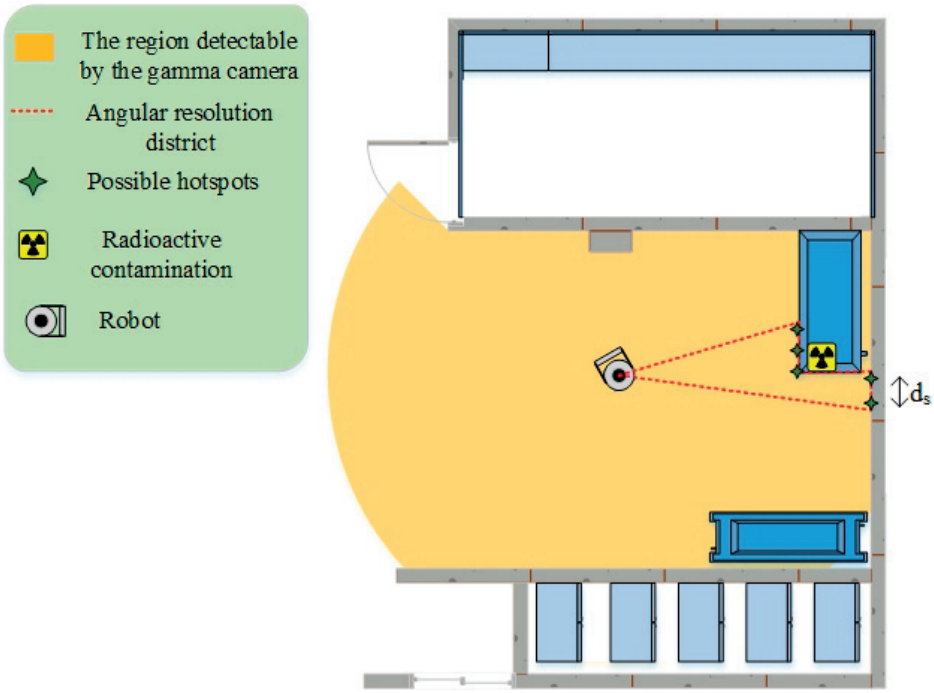


Figure 3.11 – An example showing the possible radioactive hotspots in the contaminated area within a AR district.

1. The `support` of each source candidates rises if it is visible by the activated gamma camera. As the robot can not separately distinguish sources that are placed on the same AR district (red-dotted zone in Figure 3.11), `supports` of all candidates on the contaminated AR district increase.

2. Each candidate source that can be seen by the activated gamma camera, its `support` decreases. The gamma camera will be activated on different locations, so `supports` of fake source candidates will quickly reach the zero value.

3. `Support` of each source always decreases over time. The rate of this decrease is often very low (Table 3.3).

Table 3.3 – The rates for updating `supports`

Rule	1st	2nd	3rd
Value ($\frac{support}{sec}$)	1.0	-0.8	-0.05

Note that the rate of increase is always greater than the rate of decrease (considering both the first and second rules). Regarding these rules and rates, `support` of a real source always increases over time; so the source candidates with higher `support` are likely to be a real radioactive source.

After finding new source candidates and updating `supports`, we need a method to approve a real hotspot. Algorithm 1 shows how a candidate will be approved if it is a real source. This algorithm chooses visible candidates whose `supports` are greater than a defined threshold. Next, the algorithm investigates other possible candidates that may exist on the same AR district (red-dotted zone in Figure 3.11). If the selected candidate is alone on the AR district, it will be approved. Whenever a candidate's `support` reaches less than zero, it will be automatically discarded.

Algorithm 1 Approving the real radioactive hotspot

for ($i = 1$ *to source_number*) **do**
  if (*support of source*[i] $>$ *defined threshold*) and (*source*[i] *within the gamma camera range*?) **then**
    if (*is not there any other source on the AR distric*?) **then**
      source[i].*state* $\leftarrow$ *approved*

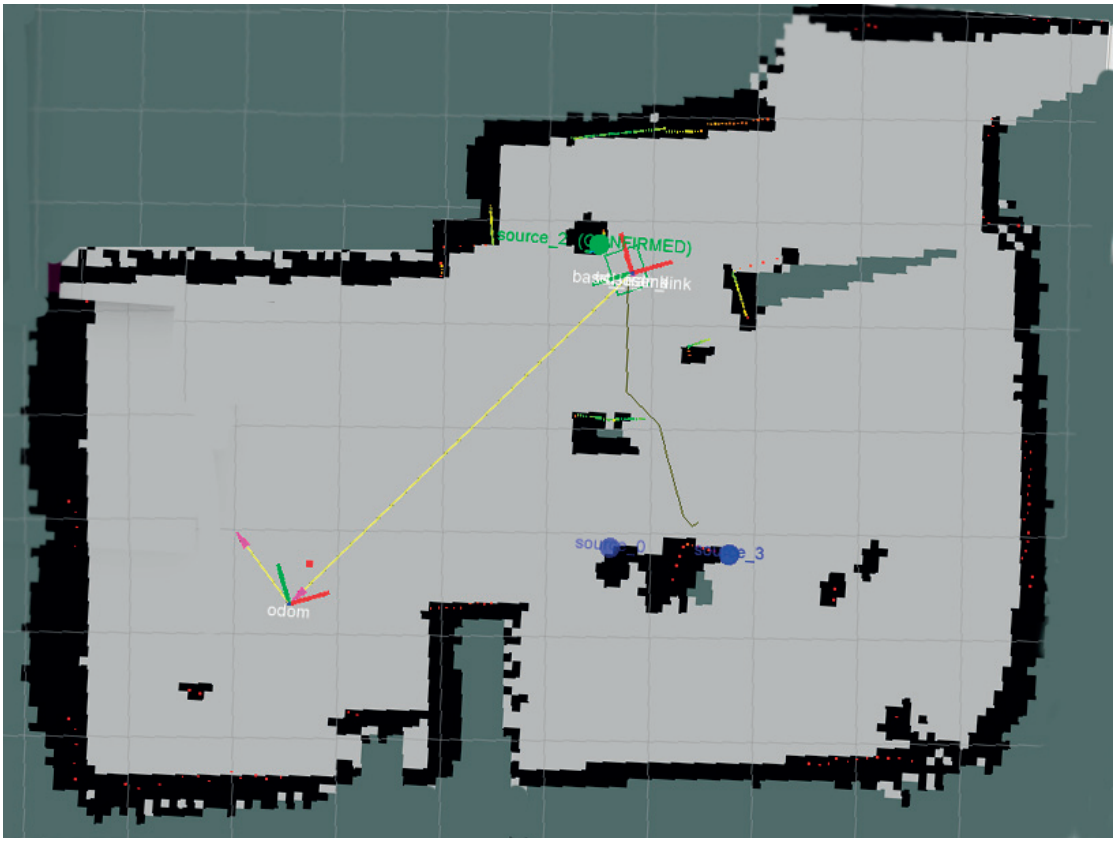


Figure 3.12 – An image of the exploration experiment was made by the `rviz` (a tool for visualization in ROS). The robot approved a candidate source and then planned to move toward two other sources for close-up investigations.

3.4.2 Victims detection system

In this research, we used the front camera of the robot as a vision system to detect pseudo-victims in the real-world setup. As shown in Figure 3.13, a pseudo-victim is a human shape filled with and distinguished by a red color. The red color is extracted by image processing methods; afterwards, the victim's location is estimated in the robot coordination system using laser range scanner data.

(a) The processed image

(b) The detected victim

Figure 3.13 – Real-time processing of the video made by the front camera during victim detection. The green circle marks the detected victim.

Like the source candidates mentioned above, each victim candidate takes a `support` variable that indicates its probability of being a real victim. Three rules are defined to update `supports`, as follows:

1. `Support` of each victim raises if the robot sees the victim with its front camera.
2. `Support` of each victim decreases if it can be detectable with the front camera. The robot sees victim candidates on different locations, so `supports` of fake candidates will quickly reach the zero value.
3. `Support` of each victim decreases over time. The rate of this decrease is often very low.

Note that the rate of increase is always greater than the rate of decrease (considering both the first and second mechanisms). As a result, `supports` of real victims increase over time. The rates in Table 3.3 were used for updating the `supports` of victim candidates.

In the simulator, the absence of the simulated front camera forced us to model its behaviour. Hence, the victims are detected as long as the victims are located within the coverage area of the imaginary sensor. The positions of the victims were previously stored in an external text file, and the robot would just need to read and compare the positions of the victims with the current position of the robot.

3.5 ROS packages

ROS is a meta-operating system[8] used for different kinds of robotic capabilities. It provides the required services from an operating system, such as hardware abstraction, low-level device control, implementation of commonly used functionality, message passing between processes, and package management [79]. ROS also contains many open-source packages for coding in several functionalities into the robots.

Figure 3.14 represents the architecture of our autonomous construction system, which directs robots toward protective structures building in the rescue application. The units that are inside the different boxes of Figure 3.14 either are completely developed in ROS or benefit from other ROS libraries. In this section, we describe the units of `basic box`. Other units will be described in the following chapters.

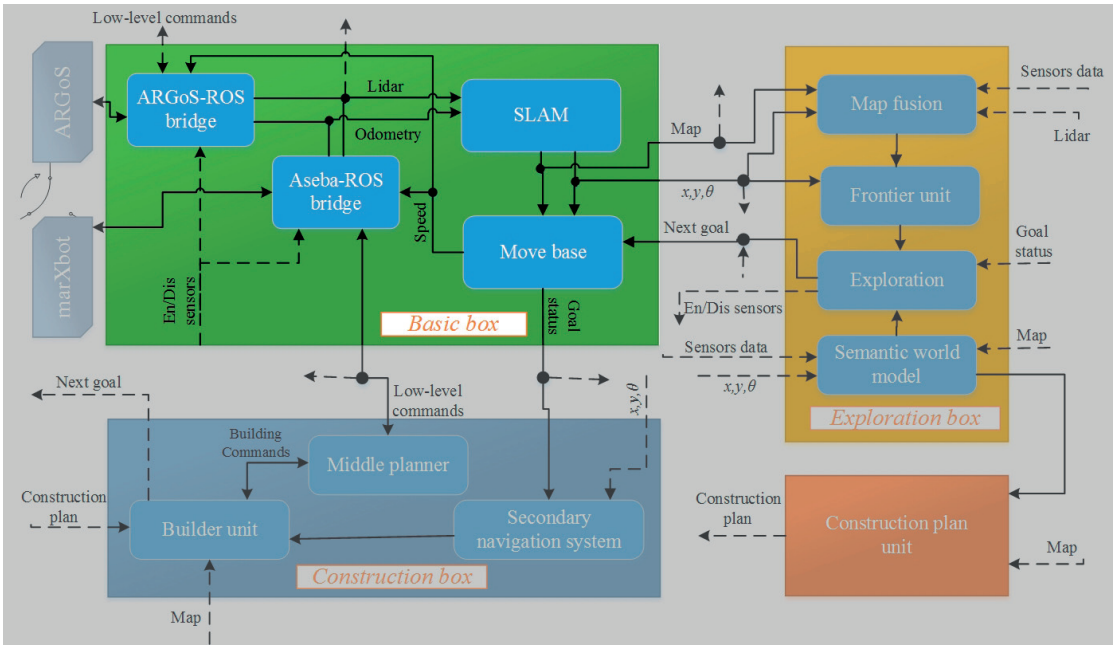


Figure 3.14 – The architecture of the autonomous construction system that directs robots to build protective structures in the rescue application. The units inside the `basic box` are developed based on the ROS platform.

3.5.1 Hector_mapping package for performing SLAM

This package is embedded in the `SLAM` unit and provides laser-based 2D SLAM without using odometry, because odometry is usually unreliable and inaccurate; therefore, the system works purely based on the scan-matching method. Scan matching is a process of comparing

[8] A platfom between an operating system and middleware.

consecutive scans to each other or to an existing map to determine a robot's position. The result can be very accurate if laser scanners have low distance measurement noise and high scan rates [80].

3.5.2 Move_base package for navigation

The `move base` unit applied the `move_base` package that was developed in ROS to help a robot navigate from its current location to a target point. This package consists of several parts; the main parts are: a path planner to produce required trajectories for robot navigation and a costmap linked to the planner.

Costmaps are 2D maps that are inflated based on the occupancy grid map and a userspecified inflation radius. More specifically, the costmap takes into account sensor data and a static map when it stores and updates information about obstacles. It propagates a cost border outwards from each occupied cell until it reaches an inflation radius is reached. In the next step, the path planner produces kinematic trajectories based on the costmaps while considering the robot's shape [81].

3.5.3 Aseba_ROS_bridge and ARGoS_ROS_bridge

These packages are middle software designed in ROS to connect units and low-level hardware. They are designed to convert raw data from sensors to defined ROS formats or to enable actuators according to high-level ROS commands. For instance, scan and odometery topics are publishers for the laser scanner and odometry data that are converted to ROS formats. Much like the scan and odometery, the speed command[9] is translated into the left and right speeds of the treels using these middle software.

In addition, a low-level obstacle avoidance with the ability to detect nearby objects and avoid a collision is embedded inside these packages, helping the robot react quickly environmental changes. This prevents the robots from being damaged if other units (e.g., move base) do not work properly.

This package also interprets video stream. It benefits from `OpenCv` libraries[10] and processes real-time video from the camera. It detects objects that are distinguished by particular features such as color or QR code. In addition, the other units can enable or disable the camera when they require camera data.

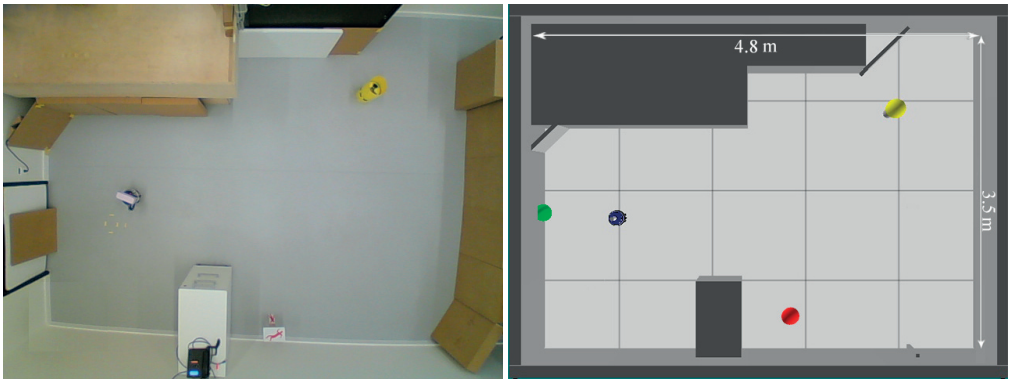
[9] Messages of `cmd_vel` topic.
[10] www.opencv.org.

3.6 Environments

The environment is part of an office room, and it is flat area surrounded by walls. Figure 3.16 shows the neat environment and Figure 3.17 shows the cluttered environment.

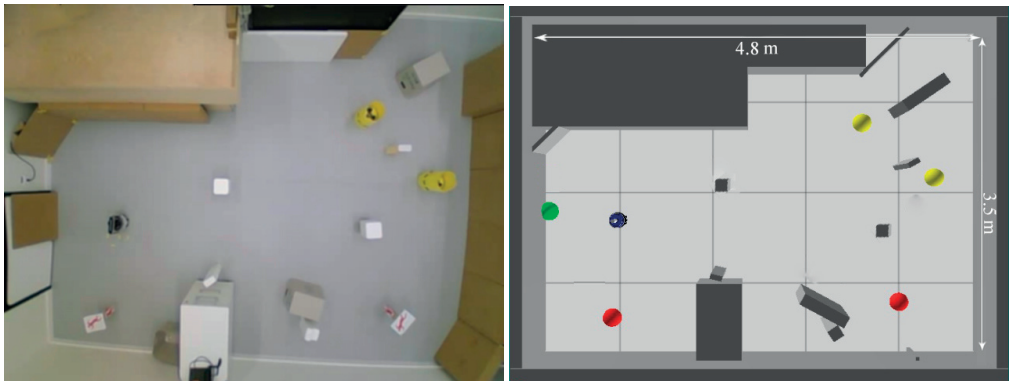
Each block that is supposed to be used by a robot is 6 cm in length, 6 cm in width, and 18 cm in height; each one weighs approximately 20 g. A ferromagnetic plate at the bottom of the block allows the marXbot to grasp the component (magnetic switchable device). The size and weight of the block were chosen to satisfy the requirements of the robot's gripper and the laser scanner (Figure 3.15).

Figure 3.15 – A polystyrene block to be used as a construction element for the experiments.



(a) The real-world setup (b) The simulation

Figure 3.16 – A top view of the neat environment used in the real-world setup and its simulation. A victim (red object) and a source (yellow object) are randomly placed in the environment.



(a) The real-world setup (b) The simulation

Figure 3.17 – A top view of the cluttered environment used in the real-world setup and its simulation. Two victims (red objects) and two sources (yellow objects) are randomly placed in the cluttered environment.

4 Autonomous exploration for rescue after a nuclear disaster

4.1 Introduction

The meltdown of the Chernobyl nuclear power plant began on April 26, 1986 and was the worst civilian nuclear accident in history. Thirty-one of 237 people who had acute radiation sickness died within the first 3 months [82]. Several thousand fatal cancers (e.g., thyroid cancer) were reported among the people who were in the contaminated areas in the years following the accident. This accident showed that we cannot believe nuclear infrastructures to be reliable and safe systems that cannot fail, even if they were built using high-security standards. Furthermore, some countries such as the United States, UK, and France produced and warehoused tons of nuclear weapons. The existence of these weapons is always a threat to humankind. If these weapons were to be used again or tested improperly, it would create disasters worse than the atomic bombings of Hiroshima and Nagasaki. In case a nuclear incident occurs, the world must be ready to save human lives and decrease potentially serious radioactive effects.

Rescue robots can be used to mitigate nuclear disaster's effects. Many robotics companies and research groups have presented different robotic platforms that can be used as assistants for human rescue teams in hazardous and restricted environments. An important task for rescue robots is to map and find elements (e.g., victims, hazardous materials) inside a structure, in caves, or out in open areas. These robots are usually expected to quickly accomplish exploration without increasing the risk [83].

A nuclear accident can be a devastating disaster, injuring countless people, ruining structures, and contaminating large areas. The post-disaster area is a threat to the human life and long-term health, so it restricts rescuers from being in the zone too long or even at all. Autonomous robots can effectively handle the exploration tasks, providing vital information about the situation. For instance, robots could collect information about radioactive hotspots and

victims' situations and localize this information on a global map. Based on this information, rescuers can decide what the best way is to save victims (e.g., a victim that has been exposed to high doses of radiation has to be removed sooner than others).

The scenario we used was an environment after a nuclear disaster. The autonomous robots used for this situation were designed for exploration, having gamma cameras for radioactive hotspot detection, a IR camera for victim detection, and a laser range scanner to map the environment. The main contribution of this research is to study methods for an efficient exploration using an autonomous robot equipped with a gamma camera. Even though much progress has been made in gamma camera technologies, there are still some limitations that stifle its potential in exploration applications. The first limitation is the relatively long acquisition time of a gamma camera when preparing a radiation image; this is the main challenge for real-time exploration. The second limitation is poor angular resolution (AR). The poor AR makes it impossible to precisely and quickly localize radioactive hotspots that are placed far from a robot, by use of a single image.

In summary, we developed and implemented new exploration algorithms for the detection and localization of radioactive hotspots in a reasonable amount of time to go with the mapping of the unknown environment. Heuristic and multi-criteria decision making (MCDM) are two different methods used for autonomously performing exploration because the robots take into account the limitations of using a gamma camera. Exploration is a key element of AFAC based on the global approach. This studies possible improvements to decrease the exploration time and increase efficiency. These improvements have been elaborated at the very end of the thesis, based on the observations of the experiments. Therefore the main experiments of this thesis do not take advantage of this approach and are based on the heuristic method.

The rest of the chapter is organized as follows: in Section 4.2, we review the existing works on robotic exploration. In Section 4.3, we state the problem and describe the assumptions that we have made. Our exploration strategies are presented in Section 4.4. The control architecture is described in Section 4.5. Section 4.6 presents and discusses experimental results. Section 4.7 concludes the chapter and proposes future works.

4.2 Related work

In this section, we present the literature related to exploration strategies for rescue applications and radioactive hotspot detection. In Section 4.2.1, we address exploration strategies that could be used in a rescue situation then, we investigate more specifically the methods that were used to detect and localize radioactive hotspots in Section 4.2.2.

4.2.1 Exploration strategies for rescue application

Exploration strategies can be categorized into three groups: 1) the simplest exploration strategy that drives robots on fixed paths, 2) random movement to explore the environment, and 3) exploration methods that choose the next target location based on the highest utility value.

In the first category, robots move on predefined trajectories. For instance, after a mine disaster, rescue teams attempt to receive information by sending the robot along a fixed path or a canal [84]. Aerial robots equipped with cameras will perform assessments and surveillance of the environment by moving along predefined trajectories. For instance, a UAV (or a group of UAVs) can assist rescuers in sweeping through regions for victims by analyzing human appearances or movements [85, 86]. McGee and Hedrick [87] studied an exploration method for finding mobile intruders or evaders. A search robot was equipped with a special sensor to detect the evader within the sensor region. Without any assumption about the motion of the evader, the robot could start to cover the region by following a predefined trajectory.

An example of the second category (random movement) is the foraging approach. In this approach, a large number of individuals must search the environment and find targets (e.g., food); they then must transport the targets to one or more depot sites [88]. For instance, in [89], multiple robots randomly explored an unknown environment in search of a specific food location and then carried the food to the hive location [90]. Search and rescue, landmine clearance, and harvesting are diverse real-world applications for foraging robots. In another, different random exploration, Arkin and Diaz [91] presented an anchored random exploration strategy that did not use any environmental information. In this strategy, the robots move randomly, but they must stay inside their communication networks.

The purpose of autonomous exploration is to efficiently complete an operation within a reasonable amount of time, so some exploration systems evaluate point candidates and then direct robots to the best point. Yamauchi [92] introduced the frontier approach to explore an unknown environment. Frontier points are made on the boundary that divides explored from unexplored space. Then, robots choose one of points among them based on a defined rule, for instance, selecting the nearest point to the current position of the robot. By following such an approach, exploration strategies could obtain more accurate information and minimize travel costs by optimizing exploration. These kinds of exploration strategies defined utility functions to choose the next point. For instance, Burgard et al. [93] presented a utility function that simultaneously takes into account the cost of reaching an unexplored location and the benefit acquired from there. Wirth and Pellenz [94] presented an approach that takes into account the distance to the next frontier and the difficulty of the path for the robot. The difficulty can be either crossing narrow paths or being in open spaces where the sensors cannot detect a landmarks. Therefore, the robot chooses the safest path with the lowest cost to reach the

target point. In the exploration approach used in [95], the decision is made based on the next best view (NBV). This means that the exploration system drives a robot to discover most information when it reaches the target location.

Some exploration strategies based on utility functions take into account several criteria. In these strategies, the utility of a point is a combination of different criteria (e.g., travelling cost, total number of sensing operations, total number of stops to reach location, etc.), so the selected point with the highest utility value is a better choice for satisfying all criteria.Tovar et al. [96] proposed an efficient exploration algorithm in which the utility function combines several criteria in a multiplicative form to evaluate candidate frontier points. The research in [97] proposed a more general way to choose the best candidate location. In this multi-objective method, the values of the involved factors are kept separate from each other without combining them in a utility function; hence, a set of nondominated points (Pareto frontier) could be selected. Basilico and Amigoni [98] implemented a method based on MCDM for robots employed in search and rescue applications. In this research, a robot started to find targets (e.g., victims) in an unknown environment. One feature of this MCDM is that it considers dependency between criteria because of the interactions between some criteria (e.g., travel cost and battery life). In [99], a multi-robot system was designed for real-time exploration. The concept behind this work was to simultaneously explore and create a network within a chain of robots (for communication between robots). Some robots looked for victims and map the environment while other robots acted as relay terminals between explorer robots and the base station. Each explorer robot selected the next target location among frontier points using a utility function.

The related works in relevant literature have shown that we must design our exploration strategy based on the principles of the third category because we need to carry out rescue activities in an efficient way. Therefore, our exploration methods were designed to drive the robot to a point with the highest utility.

4.2.2 Radioactive hotspot detection and localization

In this section, we categorize the research of radioactive hotspot detection and localization according to robots' detectors, as follows: non-directional (e.g., NaI scintillation detectors [100]) and directional (e.g., gamma camera).

The non-directional detectors are often used to map environments in terms of radiation intensity, or a network of embedded non-directional detectors are used to find stationary or mobile sources. In [101, 102, 103], the network of fixed radiation sensors were used to identify radioactive stationary or mobile sources amidst background radiation. Baidoo-Williams [104] has defined the smallest number of fixed sensors to theoretically localize sources based on

their gamma-ray count.

For applications such as rescue operations, mobile detection systems are certainly needed to search in post-disaster environments. In [105], an aerial robot was equipped with a non-directional radiation sensor; it was tasked with effectively mapping the whole environment and provide a radiation map, and then localizing strong radioactive sources. Its algorithm to localize an arbitrary number of radioactive sources and estimate their intensity was based on the Hough transformation. Newaz et al. [106] presented a method to quickly and accurately estimate multiple radioactive sources using a radiation map.

Directional detectors (e.g., the CZT Compton camera) are used in a lot of related research. Most of them obtained 2D directional image that use stationary detector systems [107, 108, 109]. In [110], two fixed-coded cameras were used to estimate the spatial coordinates of a radioactive source by employing triangulation methods. In this research, the sources were placed far from the detector in respect to the detector size.

In opposition to stationary detector systems, in [111], a detection system (an array composed of 18 CZT detectors) and a positioning system were placed on top of a cart to localize two radioactive sources. The detection system was manually moved, and its position was recorded to reconstruct a radiation image in continuous motion. It could localize the point sources within 10 cm of the true source position after 4.5 minutes. Deb et al. [112] compared the detection systems (directional and non-directional detectors). Two different algorithms were tailored to estimate the location and rate of an unknown static radioactive source by fusing together information from multiple detectors; their principle is similar to the related works presented above.

The exploration strategies developed and presented in Section 4.4 are novel solutions for autonomous exploration using a gamma camera and they are beyond the state-of-the-art. The aim of these exploration strategies is for a mobile detection system, which is composed of directional detectors with long acquisition time and poor AR limitations, to be efficient and autonomous.

4.3 Problem statement

As mentioned, we face two limitations in the use of a gamma camera. The first limitation is the relatively long acquisition time, which is a key issue for a mobile and autonomous exploration. Although Jaworski and He [111] presented a mobile exploration based on a series of data from a detection system (an array composed of 18 CZT detectors), its quality depends on the precision of the localization system and motion pattern; these factors that are not predictable or accurate enough in an autonomous system. The AISense (Table 3.1) with 0.1

sec of acquisition time seems to be a good choice for our application, but this detector cannot detect radioactive sources separately if there are more than one of them in the environment. In the case of more than one source, the device will assume that it is one hotspot only and will determine its location as being the hotspot somewhere between the two real sources (i.e., centre of mass of sources), depending on the individual activities of the hotspots and distances to each. As soon as the robot moves closer to one of them, this hotspot will take precedence over the other. This behaviour causes an inefficient exploration and leads to the inability to find all radioactive sources.

Furthermore, the need for an efficient algorithm avoid us to think about methods such as the use of non-directional detectors for the radioactive hotspot detection because these methods often drive robots to sweep the whole environment for mapping and then find radioactive hotspots, as discussed in the related works presented above.

The problem with using a non-real time detector (e.g., gamma camera) is that a robot needs to stop many times to obtain the required information. The question is how and where must a robot stop to obtain information? To answer this question, we need an algorithm to drive the robot so that the robot stops only at optimal points for a few minutes (i.e., the acquisition time) to acquire as much information about the radioactive data as possible while concurrently making progress in its exploration. The second limitation of gamma cameras is that they usually have poor AR. This prevents it to precisely localize radioactive hotspots placed far from the robot by a single image. Therefore the robot needs to move toward the possible sources for better estimations of them.

In conclusion, we require exploration algorithms that can accomplish the exploration tasks as quickly as possible and that are confined to the characteristics of the gamma camera. In this chapter, we present two different exploration methods to address the problems of autonomous exploration when using a gamma camera for a rescue situation.

4.4 Exploration strategies

Rescue robots need different sensors, such as sonar, a laser range scanner, and cameras to map and provide a world model of required elements. In our scenario, a robot composed of a gamma camera, IR-camera, and laser range scanner was designed to map and build a world model of elements (e.g., radioactive sources). Heuristic exploration and MCDM are two exploration methods developed for exploring the environment and recognizing elements after a nuclear disaster. As previously discussed, in addition to the time constraint for the rescue operation, we deal with two main limitations connected to the gamma camera's characteristics: the long acquisition time and poor AR. The designed algorithms address the challenges for

performing an effective exploration.

4.4.1 The heuristic exploration method

Heuristic exploration is one strategy designed to address the issue of autonomous exploration with limited sensors. Because of the acquisition time limitation, the robot must stop for a certain amount of time to obtain a single radiation image. Determining where the robot should stop is the first issue of this algorithm, and it impacts exploration performance; our rule is to stop the robot wherever a defined amount of the gamma camera coverage area has not already been covered. The second limitation is poor AR that detracts from precisely detecting and localizing the radioactive sources. Therefore the robot must move nearer to radioactive source candidates to improve their location.

Algorithm 2 describes a heuristic exploration. In this algorithm, the robot continues its job within a defined loop until there is neither a frontier point nor candidate (e.g., possible source). For the sake of the acquisition time of the gamma camera, the robot must stop to acquire sensory data. We define a rule that stops the robot and enables the gamma camera. The robot always checks how much area the gamma camera has already covered. If it is less than a defined threshold, for instance k%, the robot stops to record a new radiation image. To provide better understanding, Figure 4.1 illustrates a robot equipped with a gamma camera whose field of view (FOV) is 180 degrees; less than 50% of the detector coverage area is already covered, so it stops. After the new radiation image is attained, the new possible radioactive hotspots are considered according to the method presented in Section 3.4.1 and then added to the candidate list. Afterwards, the robot moves toward the nearest hotspot to approve or discard it based on the method presented in Algorithm 1.

The victim detection system is intended to be a real-time system; a robot searches for possible victims as it moves. Detected victims are also added to the candidate list to be checked in a close-up investigation.

Through a particular function, we remove some redundant hotspot and victim candidates before the next step. First, we need to remove candidates whose `support`[1] is less than zero. Second, some candidates are falsely identified from sources or victims because of poor AR or inaccuracies of the positioning system, so they must be removed. This function, `REMOVE-REDUNDANT(sources)` and `REMOVE-REDUNDANT(victims)`, is used to remove redundant candidates among possible radioactive hotspots and victims.

[1] See Section 3.4.1 for the description about `support`.

Algorithm 2 The pseudo-code for heuristic exploration

```
next_frontier_point ← GET-NEXT-FRONTIER-POINT()
while (next_frontier_point.size > 0) or (candidate.size > 0) do
    if (((covered area)/(the gamma camera coverage area)) < k%) then
        STOP(acquisition_time)
        ENABLE-THE-GAMMA-CAMERA()
        candidate ← candidate + new_possible_sources          ▷ See Section 3.4.1
    if (found a victim?) then
        candidate ← candidate + new_possible_victims          ▷ See Section 3.4.2
    if (candidate.size > 0) then
        next_candidate ← FIND-NEAREST-CANDIDATE()
        MOVE(next_candidate.position)
        if (next_candidate.type == source) then
            STOP(acquisition_time)
            ENABLE-THE-GAMMA-CAMERA()
            APPROVE-REAL-SOURCES()                            ▷ See Algorithm 1
            candidate ← candidate + new_possible_sources
            REMOVE-REDUNDANT(sources)
        else if (next_candidate.type == victim) then
            APPROVE-REAL-VICTIM()
            REMOVE-REDUNDANT(victims)
    else
        next_frontier_point ← GET-NEXT-FRONTIER-POINT()
        MOVE(next_frontier_point)
```

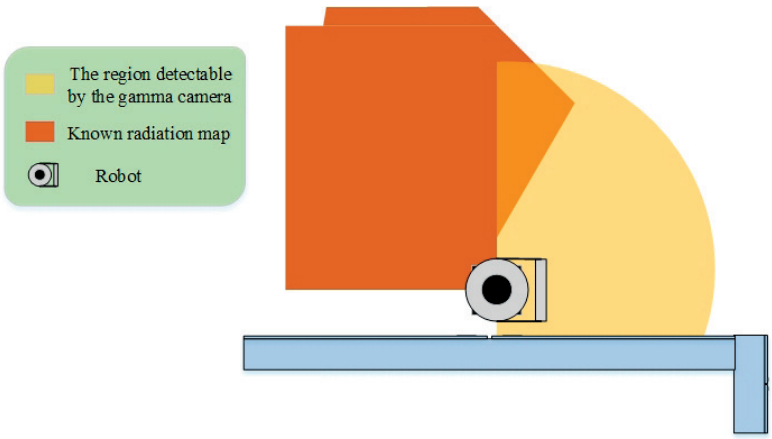


Figure 4.1 – A robot equipped with a gamma camera with a 180 degree FOV is ready to record a new radiation image.

4.4.2 MCDM method

One method of exploration presented in Section 4.2.1 (related work), was to efficiently drive robots by means of an effective decision in each step. The core of this type of exploration method was to formulate a utility function based on the criteria defined by a designer. The utility function then evaluates all available candidates to present the next target (i.e., point or task) with the highest utility value. The selected target was the most eligible candidate to satisfy the criteria.

The main purpose of autonomous exploration in a rescue operations is usually to efficiently perform the rescue within a reasonable amount of time, but the sensors' limitations (e.g., the long acquisition time of the gamma camera) confine robotic activities. In this section, we develop the MCDM method that takes into account the sensors' limitations in performing an effective autonomous exploration. We define a global utility function that consists of criteria and some of these criteria are associated with sensors' limitations. An action based on a sequence of selected points that have the highest utility values results in an effective exploration. In Sections 4.4.2.1 and 4.4.2.2, we present the criteria and utility functions, respectively. Section 4.4.2.3 describes how the MCDM method is implemented and how it works.

4.4.2.1 The criteria

The idea behind the exploration method is to incrementally explore the environment by choosing the best point in each step according to the criteria. The criteria are defined in such a way as to include important factors that impact the exploration progress considerably. We define the criteria of our case for candidate points below.

4.4.2.1.1 Detectable areas

Most exploration strategies were developed to discover environments and build maps in the shortest amount of time. Therefore, if we already know how much area will be discovered when the robot reaches the target point, we can optimize the exploration by choosing the points with the highest gain. The detectable areas (Sd) show the amount of information gained from maps when the robot is on the target point.

In our scenario, exploration is required to collect different data and to build related maps; the robot operates on the geometric, radiation, and victim maps. The geometric map is an occupancy grid map that represents the environment by way of an array of cells; the value of each cell represent the probability of it being occupied by an obstacle. The radiation and victim maps are likewise composed of an array of cells, but binary values are assigned to

these cells. The value of a cell will set to one if it has fallen within the coverage area of a detector. The purpose of radiation and victim maps is to show how much of the environment is covered for finding radioactive hotspots and victims. The position of victims and radioactive sources are stored in the `semantic world model`[2] based on the global position of the robot. Because the geometric map is often probabilistic, a particular metric is required to measure Sd. We compute the entropy of each cell of a detectable area according to information theory (Shannon information [113]). Equation 4.1 shows how to compute Sd:

$$H(c_i) = \begin{cases} 0, & \text{if } Pr(c_i) = 1 \\ 0, & \text{if } Pr(c_i) = 0 \\ (Pr(c_i) \times log_2(\frac{1}{Pr(c_i)}) + (1 - Pr(c_i)) \times log_2(\frac{1}{1 - Pr(c_i)}), & \text{otherwise} \end{cases} \tag{4.1a}$$

$$Sd = \sum_{c_i \in C} H(c_i) \tag{4.1b}$$

Where

c_i is the number of cell i; $Pr(c_i)$ is a probability value that represent occupancy status of that cell and it comes from a probabilistic map; H is the entropy value for each cell; and C is the coverage area of the sensor.

We have the three maps and we need to estimate the potential extension of each map for candidate points. Sd_{victim}, $Sd_{radiation}$, and $Sd_{geometric}$ will be information gained about new discovered regions for victim, radiation, and geometric maps, respectively, when the robot reaches the target point. The geometric map is only a probabilistic map among the three maps and each cell of this map has a value between zero (i.e., free cell) and one (i.e., occupied cell), thus, Equation 4.1a is used to compute $Sd_{geometric}$. The radiation map and victim map are binary maps and each cell of them is either zero or one. Therefore, we just count the cells with zero values within the coverage area of the related sensor to compute Sd_{victim}, $Sd_{radiation}$.

4.4.2.1.2 Distance

A common factor for optimizing exploration is the Euclidean distance (d) between the current position of the robot and a target point. It would be best to compute the shortest path between the robot and a target point based on the geometric map while this path takes into account obstacles and walls. However, finding shortest paths based on the geometric map for many

[2] See Section 4.5.4.

points could drastically raise the computation time and effort, so, the Euclidean distance is considered.

4.4.2.1.3 The total of the *supports*

As discussed in Sections 3.4.1 and 3.4.2, each element has a `support` variable that indicates the probability of it being a real element. This sounds reasonable enough if we consider the `support` to be one criterion. This criterion can direct robots toward candidates that are more likely to be real elements. U_{tos} indicates the sub-utility associated with this criterion.

4.4.2.1.4 Number of possible real elements

If a robot can approve the element (e.g., sources) quickly, false candidates made from the approved elements (because of poor AR or inaccuracies in the positioning system) will be removed. Obviously, having fewer candidates causes the exploration method to progress quicker. U_{pr} is the sub-utility for the number of real elements that can be approved.

4.4.2.2 Definition of the utility function

The exploration method directs the robot to map and localize elements within a limited amount of time by selecting a point with the highest utility value. Utility values are computed with a global utility function that consists of different criteria such that each criterion is associated with one important factor in the exploration scenario.

To combine the criteria for measuring utility values, we normalize the sub-utilities linked to the criteria to the common scale I = [0, 1]. The target is selected only in the present situation of the environment for all available points at any step. Hence, we can normalize a sub-utility of a point with respect to the maximum and minimum values of the present set of points (P) as follows:

$$\hat{U(p)} = \frac{U(p) - \min\limits_{\forall q \in P}(U(q))}{\max\limits_{\forall q \in P}(U(q)) - \min\limits_{\forall q \in P}(U(q))} \quad p \in P \tag{4.2}$$

Where

U and $\hat{U}$ are the utility value and a normalized utility value respectively.

Literature presents several methods for combining these criteria and providing final utility value. The most common way is to multiply criteria with corresponding weights and then use a summation of these terms to provide the utility value. Equation 4.3 shows our global utility

function:

$$GU(p) = w_1 \times (1 - \hat{U_d}(p)) + w_2 \times \hat{U_{Sd_{victim}}}(p) + w_3 \times \hat{U_{Sd_{radiation}}}(p) + w_4 \times \hat{U_{Sd_{geometric}}}(p) + w_5 \times \hat{U_{tos_{victim}}}(p) + w_6 \times \hat{U_{tos_{source}}}(p) + w_7 \times \hat{U_{pr_{victim}}}(p) + w_8 \times \hat{U_{pr_{source}}}(p) \tag{4.3}$$

$$(w_1 + w_2 + w_3 + w_4 + w_5 + w_6 + w_7 + w_8 = 1)$$

Where

$w_{i \in [1,2,3,4,5,6,7,8]}$ is a corresponding weight.

It is obvious that the process in the exploration will be influenced if the associated weights change. Each weight is related to one criterion, so changing the weights affects the exploration procedure, and also allows the robot to explore different situations.

4.4.2.3 MCDM method implementation

The pseudo-code in Algorithm 3 shows what this MCDM method performs. It drives a robot to the point with highest utility value until there are no more points. The `candidate_points` keeps all candidate points. The points consist of frontier points and checkpoints that are randomly made around a possible element for a close-up investigation. The checkpoints are created based on a distance that depends on AR and d_s[3]. In addition, if the FOV is not 360 degrees, each checkpoint is split into a few new points with the same location but with different yaw angles to provide multiple views for the next target.

In the next step, the utility function measures the worth of each candidate point based on multiple criteria. The target point is the point with the highest utility. If the chosen point is a point used for approving a radioactive hotspot or for an extension of radiation areas, the robot stops there and enables the gamma camera. Subsequently, the new possible radioactive hotspot will be added to the candidate list for further investigation. Algorithm 1 is used to check a radioactive hotspot candidate and may approve the candidate. If the chosen point is a point used for approving a victim, the robot starts further recognition processes, such as checking the `support` in our scenario or other sensor data (e.g., microphone) in a real rescue application.

[3] d_s is a user specified value that is required as a minimum distance between two separated sources. It determines the required accuracy for the radioactive hotspot localization. See Section 3.4.1.

Algorithm 3 The pseudo-code of the MCDM method

```
frontier_points ← GET-FRONTIER-POINT()
candidate_points ← UPDATE-POINT(frontier_points)
while (candidate_points.size > 0) do
    highest_candidate ← GLOBAL-UTILITY(candidate_point)        ▷ Calculate utility
values for all candidate points and return the highest one.
    MOVE(highest_candidate.position)
    if (Is there any source candidate?) or (Is it a radiation map extention?) then
        STOP(acquisition_time)
        ENABLE-THE-GAMMA-CAMERA()
        candidate ← candidate + new_possible_sources
        APPROVE-REAL-SOURCES()                                 ▷ See Algorithm 1
        REMOVE-REDUNDANT(sources)
    if (highest_candidate.type == victim) then
        APPROVE-REAL-VICTIM()
        REMOVE-REDUNDANT(victim)
    if (found a victim?) then
        candidate ← candidate + new_possible_victims           ▷ See Section 3.4.2
    new_points ← PRODUCE-NEW-POINT(candidate)
    frontier_points ← GET-FRONTIER-POINT()
    candidate_points ← UPDATE-POINT(new_points, frontier_points)
```

As mentioned, we must remove some redundant hotspot and victim candidates before the next step. First, we need to remove candidates whose `support` is less than zero. Second, some candidates are falsely identified from sources or victims because of poor AR or inaccuracies in the positioning system, they must be removed. Therefore, `REMOVE-REDUNDANT(sources)` and `REMOVE-REDUNDANT(victims)` are used to remove redundant candidates among possible hotspots and victims. Next, the `candidate_points` list must be refreshed because the attributes of some candidates have been changed or some points have been placed in inaccessible areas. The `UPDATE-POINT` command will update the point list and replace the old points with new ones.

4.5 The exploration system

In this section, we design a search architecture that can implement two different exploration algorithms that were described above. The main goal of the thesis is to design an autonomous construction system (AFAC) that directs robots to build protective structures in a rescue application. The robots build these structures for rescue functions, such as stabilizing large structures or protecting victims. One approach of the autonomous construction system, the global approach, requires a global representation, or knowledge of the world, so the exploration algorithms presented here must provide required information about the world

for autonomous construction. Figure 4.2 represents the architecture of the autonomous construction system based on the global approach, in which the `exploration box` applies the exploration algorithms to provide the information required for construction use. It directs the robot in efficient exploration and detects and denotes required elements (victims and radioactive hotspots) in a world model. The `exploration box` consists of four packages; they are as follows: the map fusion combines the geometric map made by SLAM, the radiation map, and the victim map; then the `frontier unit` produces frontier points based on the combined maps; the exploration unit applies either the heuristic exploration or MCDM method; and the `semantic world model` stores and tracks the positions and latest states of the detected victims or radioactive hotspots. Below, we talk about these units and their roles in more detail.

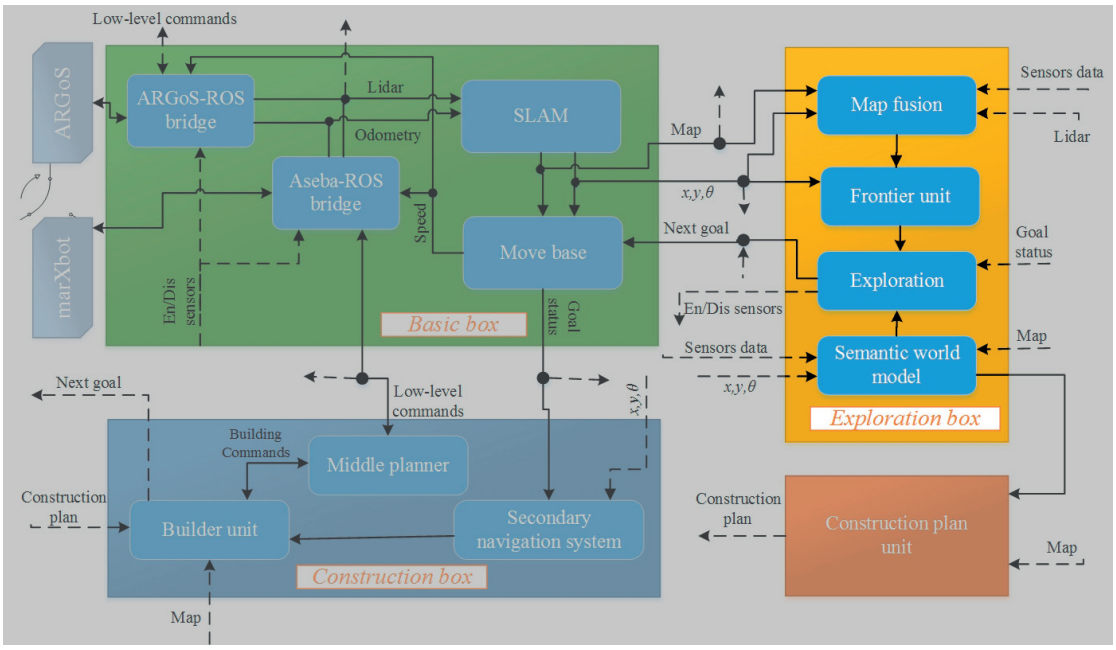


Figure 4.2 – The architecture of the autonomous construction system for the rescue application. The `exploration box` is represented as one part of the construction system for exploring the environment. It consists of four packages: `map fusion`, `frontier unit`, `exploration`, and `semantic world model`.

4.5.1 Maps and fusion

In a rescue situation, exploration is often required to collect different data and to build related maps. As mentioned in Section 4.4.2.1.1, the robot provides the geometric, radiation, and victim maps. The geometric map is an occupancy grid map that represents the environment by way of an array of cells; the radiation and victim maps are likewise composed of an array of cells, but binary values are assigned to these cells. The purpose of radiation and victim

maps is to show how much of the environment is covered for finding radioactive hotspots and victims. The position of victims and radioactive sources are stored in the `semantic world model`[4] based on the global position of the robot. The unit `map fusion` is designed based on the `grid_map` package of the ROS libraries [114]. This package is a multilayer map that stores maps such as victims, radiation, elevation maps, and more. SLAM provides an occupancy grid map that is directly stored as the main layer. The second layer is the victim map, which has the same resolution as the main map. The `basic box` sends feedback and the position of the detected victim for updating the map with the determined frequency. The third layer, the radiation map, is produced in the same way as the victim map. It also shows how much of the environment is being covered in the search for radioactive hotspots.

4.5.2 Frontier unit

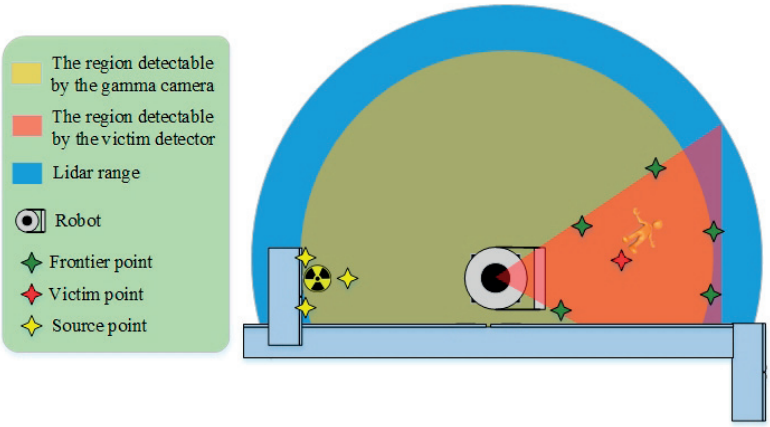
The frontier algorithm produces the next target point on the boundary between free cells and unexplored areas. It aims to extend the map into new territory by choosing one of these points [92]. The explorations presented in the literature have exploited a single map (e.g., occupancy grid map) to create frontier points, whereas our exploration methods need to consider one or two other maps (victim and radiation maps). Now, the question is: how maps have to be considered for the production of frontier points?

To answer this question, we fuse the maps together to build a minimum map. The minimum map is the common between known open spaces of the three maps. Each cell of the minimum map has to be recognized in all three maps, and must to be placed on the free cell of the geometric map. In addition, we add occupied cells that were distinguished by the geometric map to the minimum maps; this avoids producing frontier points of inaccessible regions (Figure 4.3).

The minimum map is built in `map fusion` as the fourth layer in alignment with the other maps. The `frontier unit` receives the minimum map as an input and then produces points based on it.

For the heuristic method presented in Section 4.4.1, the minimum map is established based on only victim and geometric maps. In this method, the radiation map will extend when the defined rule for the gamma camera is satisfied. The rule is that if less than k% of the detector coverage area is already covered, the gamma camera will be activated. Thus, we may have some gaps (or holes) in the radiation map where the robot may never encounter the required condition to start the process of enabling the gamma camera. Some frontier points will constantly appear on the borders of these holes. These frontier points will cause the robot to get stuck. Thus, we modified the minimum map to a combination of victim and geometric

[4] See Section 4.5.4.



(a) Candidate points

(b) The minimum map that is a combination of the different maps; free, occupied and undiscovered cells are shown by the white, black, and grey colors respectively.

Figure 4.3 – Candidate points are as follows: frontier points on the boundary of the minimum map, points around possible sources, and point around possible victims.

maps to use the heuristic method.

4.5.3 Exploration unit

The core of this unit is to apply the exploration methods which were presented in Section 4.4.

4.5.4 Semantic world model

It is necessary to have a world model to store and track all elements (victims and radioactive hotspots). After finding an element, we need to add it, or update its attributes, such as position and state to a world model. This model is a global world representation enriched with semantic information over time. It tracks information about elements and updates their attributes. For instance, the state of an element is modified when the robot approves or discards the element based on the more investigations; and the precision of an element's location will be improved when a robot moves closer to the element.

The `hector_worldmodel stack` is designed in ROS to collect and integrate semantic attributes of elements in the world model. This package subscribes to messages about victim and radioactive hotspots and then fuses all gathered information together into a single world model [115, 116]. Our semantic world model was established based on this package.

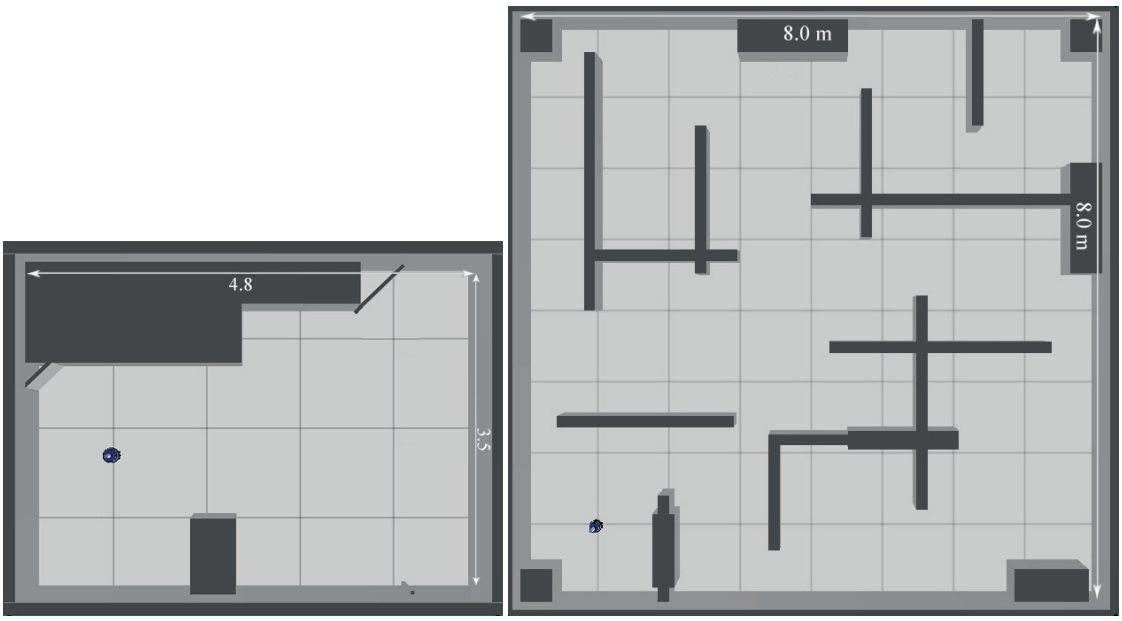
4.6 Experiments and evaluations

The two methods proposed for autonomous exploration are implemented and extensively tested in simulation (ARGoS). We also perform a series of real experiments with the marXbot to validate the simulation results. In Chapter 3, the marXbot, ARGoS, and other components of the test-bed were described. In the simulation, the sources are lights. In the real-world setup, sources are other marXbots, and their range and bearing systems are used as radioactive hotspots.

Because the gamma camera limitations are a main challenge of our autonomous exploration, we study and compare the performance of the MCDM and heuristic methods in a post-disaster environment without victims. Hence, the robot performs the exploration methods to find only radioactive sources and to maps this environment. Using an environment without any victim provides a better understanding of effects of the gamma camera's limitations on the exploration performances.

Three types of environments are considered in the experiments. The first environment, introduced in Figure 4.4a, is named environment *A*; if it is filled with debris and obstacles, we

named it environment *C*. We also consider another different environment where the length is 8 m and the width is 8 m and where there are many internal walls; we named it environment *B*.



(a) Environment A (b) Environment B

Figure 4.4 – A top view of the environments (A, B) used in the simulation.

We also change the gamma camera parameters to discover how they impact exploration performance. The values in Table 4.1 are assumed for FOV and AR. The best gamma camera has a FOV of 360 degrees and an AR of 2.5 degrees. We first change the FOV from 360 to 45 degrees to see the effects on exploration performance. Accordingly, we change AR from 2.5 to 30 degrees to study the related effects on performance. Each state (each column in Table 4.1) was repeated ten times for the three types of environment.

Table 4.1 – The values are assumed for the parameters of the customized gamma camera

States	1	2	3	4	5	6	7
FOV (deg)	360	360	360	360	180	90	45
AR (deg)	30	20	10	2.5	2.5	2.5	2.5

The MCDM method operates based on a utility function that consists of criteria and corresponding weights. We can achieve different exploration behaviours by changing the weights. However, the main goal of this research is to introduce and study the new exploration methods, so we do not investigate effects of different weights on MCDM performances nor a way for finding the best weights (e.g., in terms of the completion time) in this research. For choosing

the weights of MCDM, we define manually different weights and then run exploration experiments; later we come back to choose the best weights based on the performed experiments. Table 4.2 reports the weights of Equation 4.3 (utility function) that have been selected among the proposed weights. The set of weights consists of two stages: first when we have candidate sources on the environment; second there is no candidate source or the robot has not detected candidate source yet. Moreover, the weights associated with the victims are equal to zero because we have already assumed that there are no victims in order to focus on the gamma camera challenges.

Table 4.2 – The weights were applied in the utility function

Condition	w_1	w_2	w_3	w_4	w_5	w_6	w_7	w_8
With a candidate hotspot	0.3	0	0.1	0.05	0	0.3	0	0.25
Without any candidate hotspot	0.3	0	0.6	0.1	0	0	0	0

The termination conditions for exploration are as follows: if the exploration time exceeds from a user-defined time, or the robot entirely scans an environment to complete all maps (i.e., radiation, victim, and geometric maps).

4.6.1 Comparison of the two exploration methods

In this section, we study and compare the heuristic and MCDM methods in an environment with radioactive contamination hazards. Ten trials were performed for each exploration method in simulation, and two radioactive hotspots were randomly placed in the environment for each trial.

The geometric and radiation map coverage percentages are represented in Figures 4.5 and 4.6 for environment B, respectively.[5] The middle bar shows the best gamma camera (FOV of 360 and AR of 2.5 degrees), and the other bars show degraded gamma cameras. The geometric map coverage percentage is almost constant for MCDM and coverage is about 10% higher than the heuristic method. For building the radiation map, MCDM worked significantly better than the heuristic method. Additionally, the radiation chart shows the heuristic method discovered about 20% less of the regions than the MCDM method. In terms of the termination conditions, the graphs of the geometric map coverage percentages show that the robot could not entirely explore the environment in the heuristic method within the given time.

Moreover, the environment included two radioactive sources that were randomly placed. Table 4.3 show that the likelihood of hotspot detection in the heuristic method was 93% and

[5] In order to avoid repetition, we show only the results of environment B as they are representative of the other observed results. This environment is more complex and better highlights the interesting aspects.

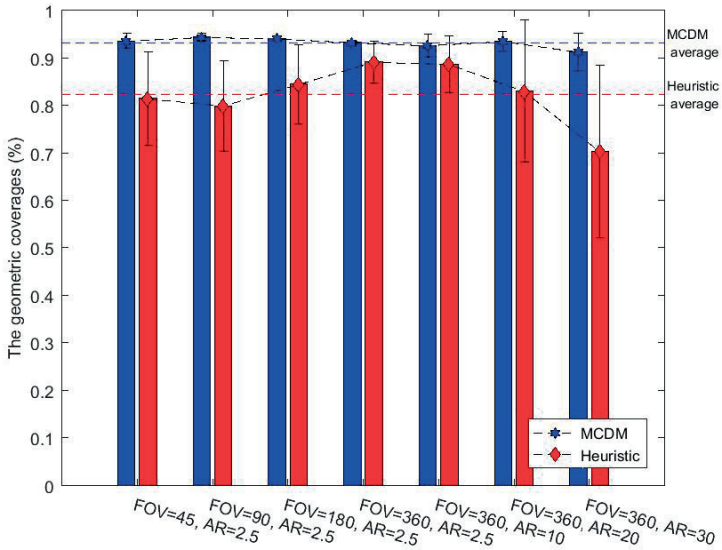


Figure 4.5 – The geometric map coverage percentages for environment B.

50% for the first and second radioactive hotspots, respectively, but the likelihoods are near 100% in MCDM for both hotspots.

Table 4.3 – The likelihood of hotspot detection

Method	First source	Second source
Heuristic	93%	50%
MCDM	100%	97%

The comparison between the two methods also shows that the heuristic method is more heavily influenced by the gamma camera's parameters. The geometric coverage percentages in the MCDM method are almost constant over different values of the gamma camera's parameters, but the percentages decrease in the heuristic method when the parameters move away from the ideal values. Changing the FOV could impact the heuristic method more than the MCDM method in terms of the radiation coverage percentages. The AR variation also affected the radiation coverage percentages in the heuristic method while it had little impact on the coverage percentages in the MCDM method.

In conclusion, we compared the performance of two exploration methods to demonstrate which approach would be the appropriate method. According to the results, the MCDM method is a more efficient algorithm. The geometric and radiation map coverage percentages grew 30% faster in the MCDM method than in the heuristic method. The MCDM method could localize almost all radioactive hotspots but the heuristic method missed some radioactive sources. Moreover, it seems that the MCDM method is more robust with different types of gamma cameras while the heuristic method is more heavily influenced by the gamma camera's

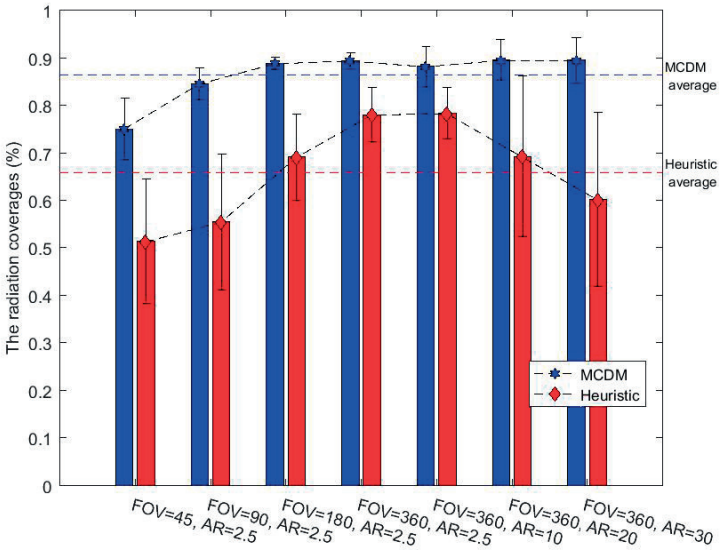


Figure 4.6 – The radiation map coverage percentages for environment B

parameters. These improvements in exploration will benefit the autonomous construction system because the algorithm based on MCDM can facilitate a quicker exploration, and cause significantly fewer radioactive hotspots to be missed.

4.6.2 Further studies on the MCDM method

In the previous section, we showed that the MCDM method is a more efficient exploration algorithm compared to the heuristic method. In this section, we studied the better exploration method, the MCDM method, in more detail. We also studied the gamma camera's parameters and showed how they impact exploration and detection. The values in Table 4.1 were given for FOV and AR.

In addition to studying the effects of the gamma camera's parameters on exploration, we investigate which kind of gamma camera is more appropriate for radioactive hotspot detection and localization: the gamma camera with a FOV of 360 degrees and an AR of 30 degrees (high FOV and poor AR) or the gamma camera with a FOV of 45 degrees and an AR 2.5 degrees (low FOV and good AR). Regarding the commercial gamma cameras reported in Table 3.1, either with the high FOV and poor AR (e.g., polaris-H) or low FOV and good AR (e.g., iPIX) are usually available. Therefore, answering this will indicate which gamma camera is the best choice for mobile robots.

Figures 4.7 and 4.10 chart the progress of the geometric coverages based on set of FOV and AR over time; Figures 4.9 and 4.10 show the radiation coverages based on set of FOV and AR over

time. They show that the complexity of the environment influences the growth of the map. The surface area of environment B is four times bigger than A (or C), but the graphs of B took ten times as long to reach a steady state. It seems that the internal walls effectively decreased the rate of map extension because they blocked the laser scanner's range.

The second important factor for the growth of the map is the parameters of the gamma camera. Because of the use of the maximum FOV scan (360 degrees), the fastest growth occurs in all three environments, especially with the radiation map. The graphs of environment B present the effects of these parameters; Figure 4.9 shows that if we degrade the gamma camera from a FOV of 360 degrees to 45 degrees, the coverage speed will decrease approximately 50%. In contrast, if we degrade the gamma camera from an AR of 2.5 degrees to 30 degrees, the coverage speed will decrease about 10%, as illustrated in 4.10.

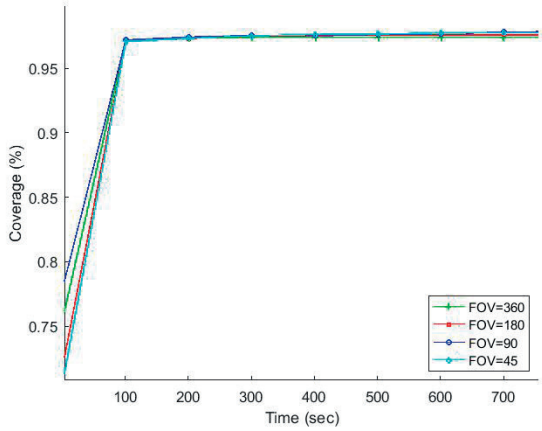
Figures 4.11 and 4.12 show the number of times the robot stops for recording radiation images. As we discussed in Section 4.4.2, the radiation image could be applied to extend the maps or check the possible radioactive hotspots. The graphs show that the robot stopped more when it had a gamma camera with low FOV or poor AR. The robot attempted to stop more for low FOV than for poor AR.

The graphs in Figure 4.13 show the time needed to localize radioactive hotspots for a set of FOV and AR. The graphs show that the localization time decreased with improvements to the gamma camera. Likewise, in these graphs, we can see that the gamma camera with a FOV of 45 degrees performed worse than the gamma camera with an AR of 30 degrees.

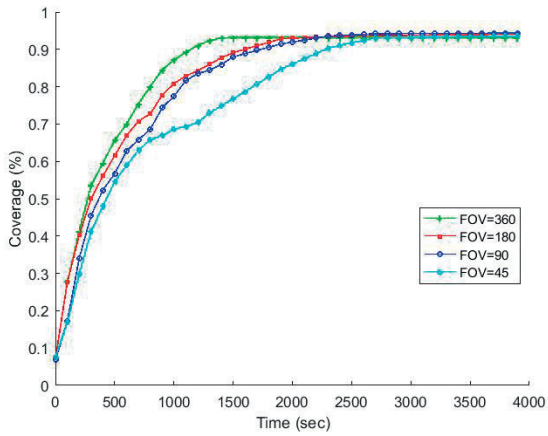
The studies done in the simulation were validated by experiments performed on the real robot (marXbot). The specification of the gamma camera was reported in Table 3.2,[6] so we could only run experiments for a robot with a FOV of 360 degrees and an AR of 2.5 degrees. Environments A and C were considered, and ten trials were run for each of them. Environment B is designed in the simulation to provide the results. Figures 4.14 and 4.15 address the difference between the geometric and radiation map coverage in the simulation and with the real robot, respectively. The practical coverages from the real experiments are slightly less than the simulation coverages. There might be some small unknown regions near the walls that could have trapped the robot. In creating our exploration algorithms, we considered using a border around the walls to prevent the robot from becoming stuck. This border was thicker in the practical experiments because of inaccuracies of the positioning and mapping systems. Therefore, the real robot ignored more unknown small regions, so the practical coverage is slightly less than the simulation coverage.

Figure 4.16 reveals the amount of extended map area covered per each stop. It shows that by

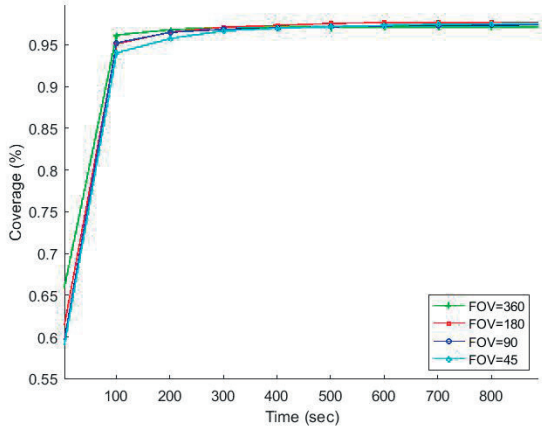
[6]The gamma camera is modelled with the range and bearing system.



(a) Environment A

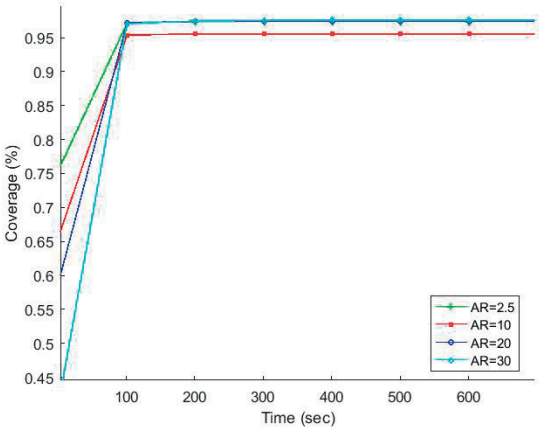


(b) Environment B

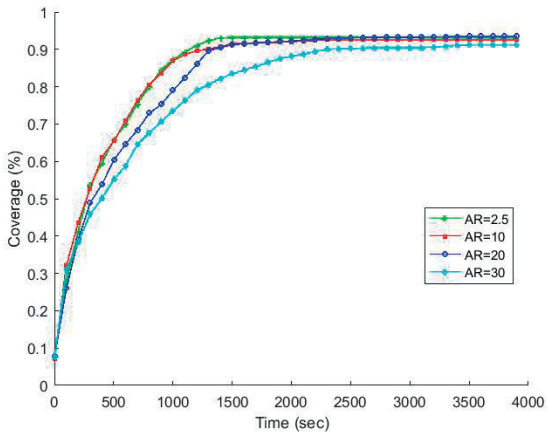


(c) Environment C

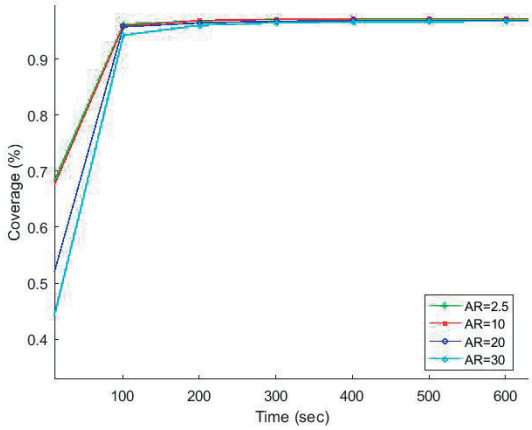
Figure 4.7 – The geometric map coverage over time based on a set of FOV, AR is equal to 2.5 degrees.



(a) Environment A

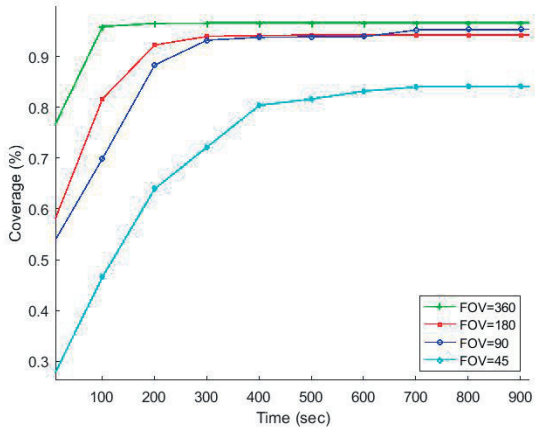


(b) Environment B

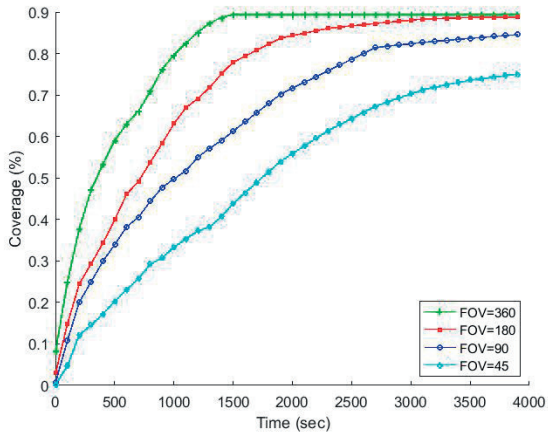


(c) Environment C

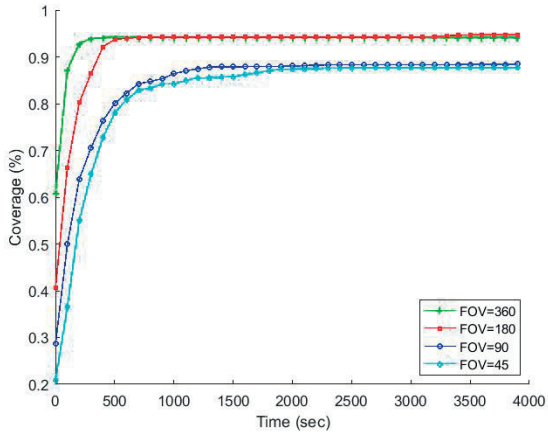
Figure 4.8 – The geometric map coverage over time based on a set of AR, FOV is equal to 360 degrees.



(a) Environment A

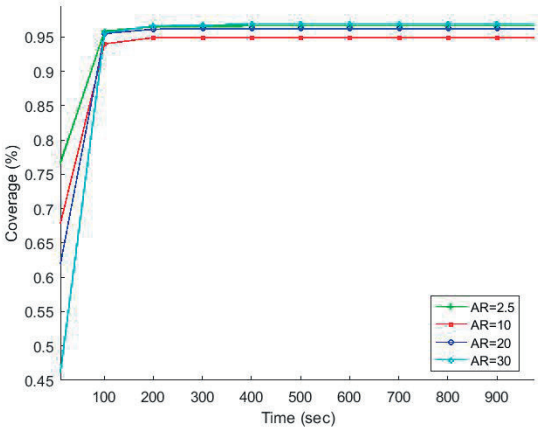


(b) Environment B

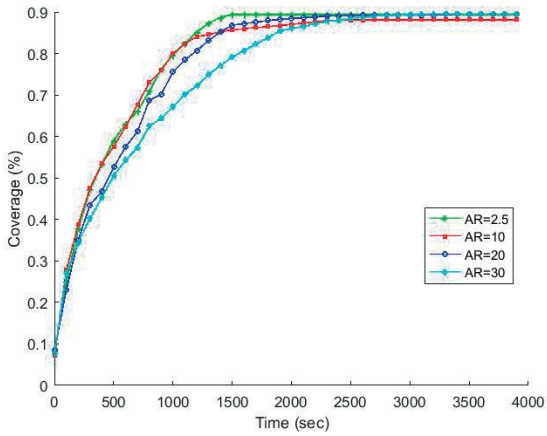


(c) Environment C

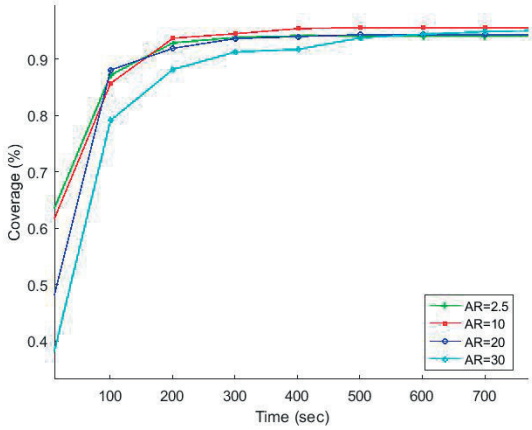
Figure 4.9 – The radiation map coverage over time based on a set of FOV, AR is equal to 2.5 degrees.



(a) Environment A

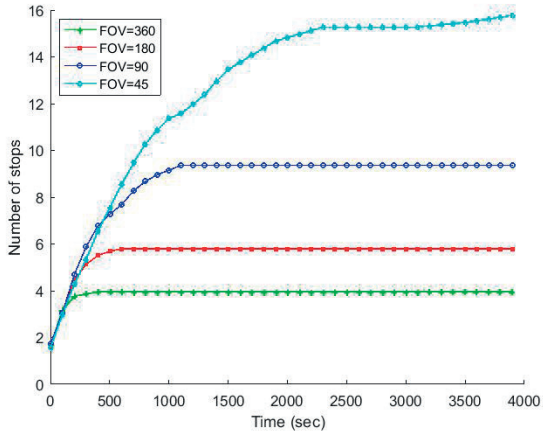


(b) Environment B

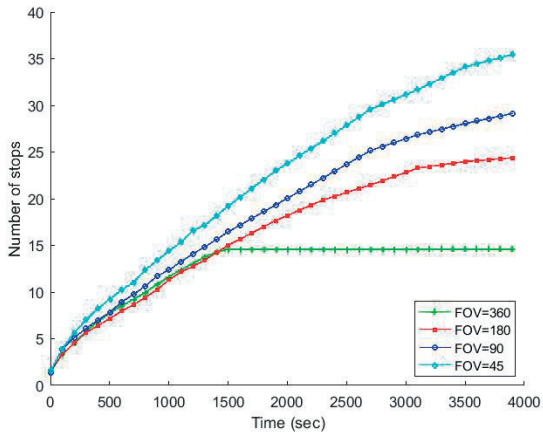


(c) Environment C

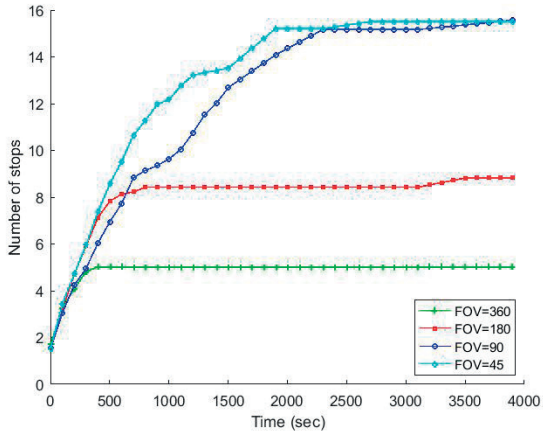
Figure 4.10 – The radiation map coverage over time based on a set of AR, FOV is equal to 360 degrees.



(a) Environment A

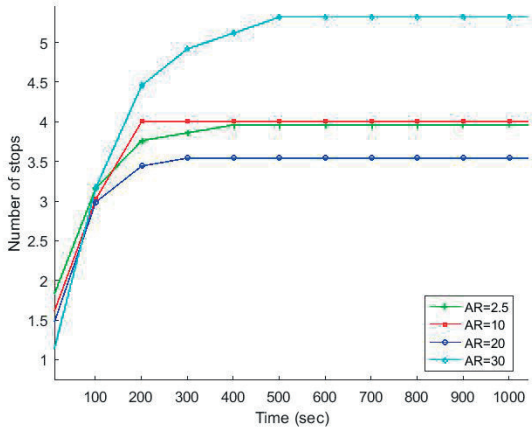


(b) Environment B

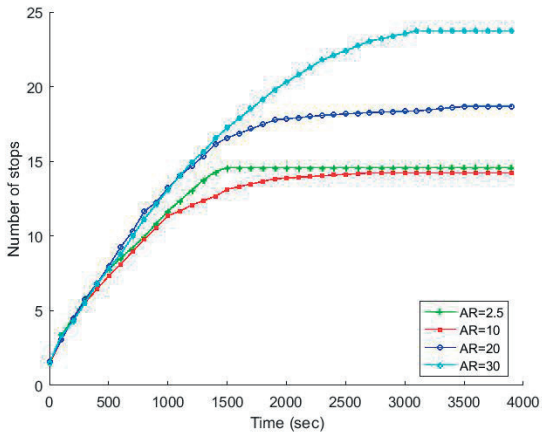


(c) Environment C

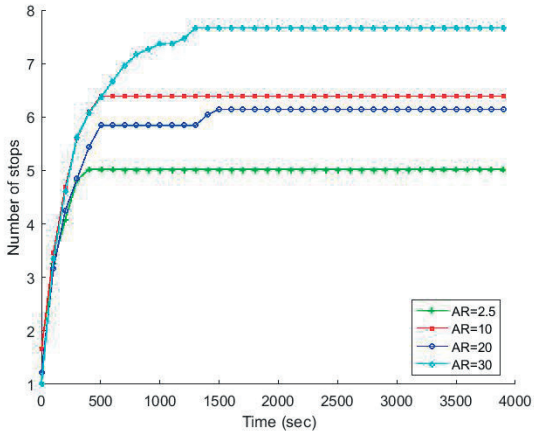
Figure 4.11 – The number of stops required for the gamma camera to give a radiation image in MCDM based on a set of FOV, AR is equal to 2.5 degrees.



(a) Environment A

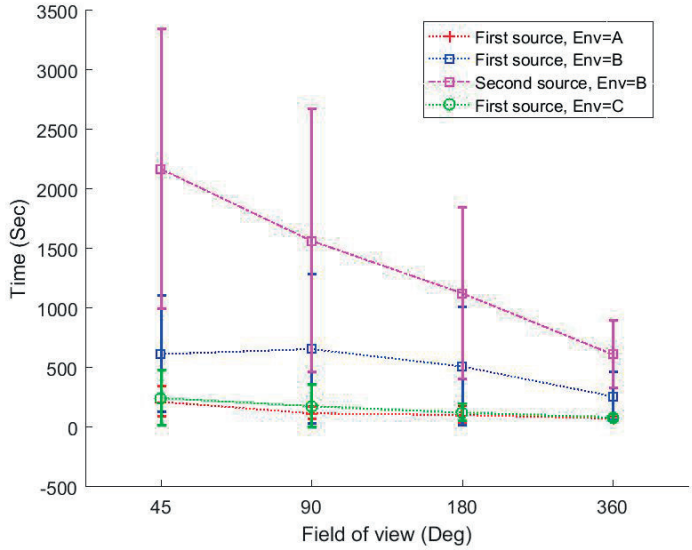


(b) Environment B

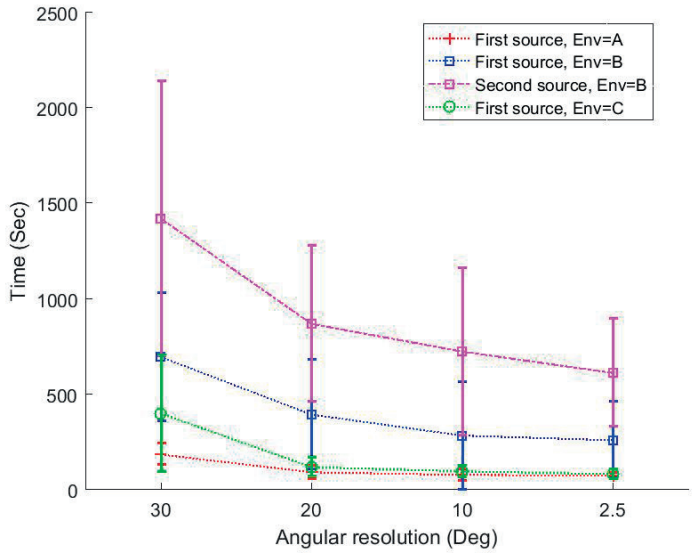


(c) Environment C

Figure 4.12 – The number of stops required for the gamma camera to give a radiation image in MCDM based on a set of AR, FOV is equal to 360 degrees.



(a) Based on a set of FOV



(b) Based on a set of AR

Figure 4.13 – The time required for radioactive hotspot localization

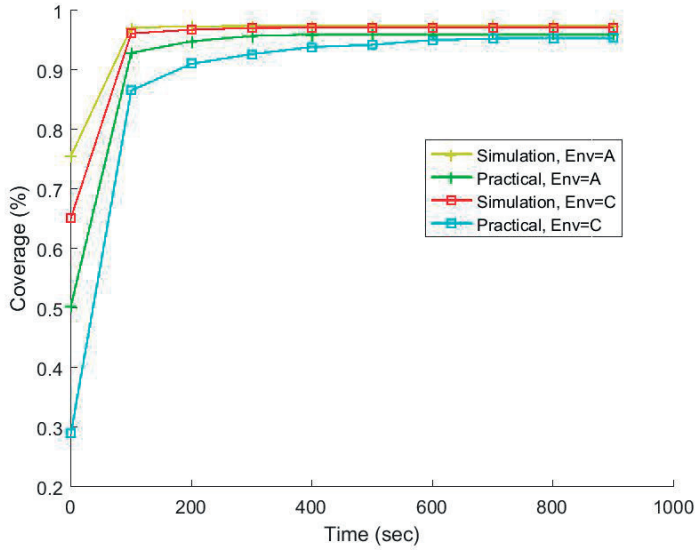


Figure 4.14 – Comparison of the coverage of the geometric map from the simulation, and from using the real robot.

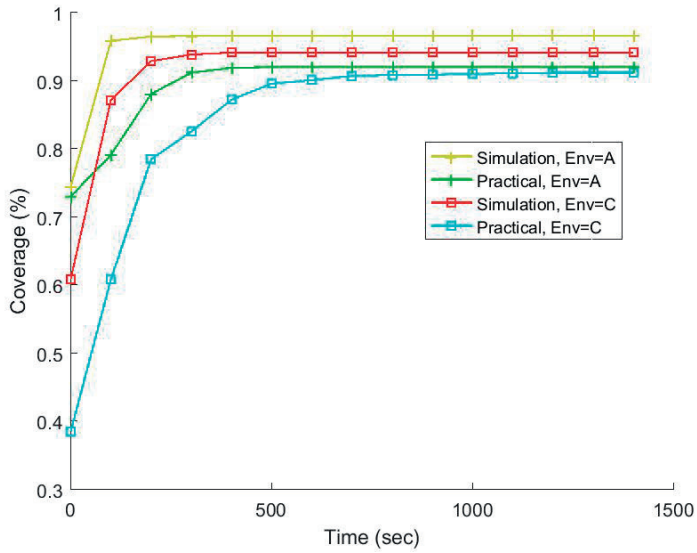


Figure 4.15 – Comparison of the coverage of the radiation map from the simulation, and from using the real robot.

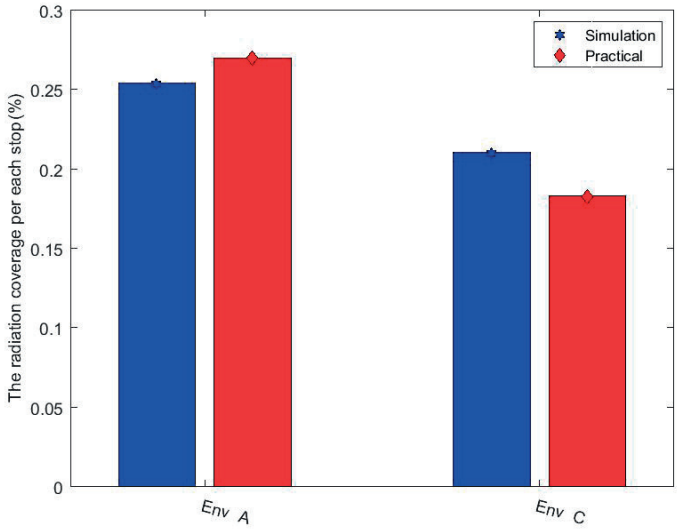


Figure 4.16 – The average mapped area covered per each stop.

increasing the complexity (from environment A to C), the coverage per stop decreases because the debris and objects on the environment obstructed the gamma camera's view.

Now, we revisit the question of which kind of gamma camera is more appropriate for radioactive hotspot detection and localization: a gamma camera with high FOV and poor AR or a gamma camera with low FOV and good AR? A gamma camera with higher FOV and poor AR is more suitable for exploration and radioactive hotspot localization. Note that we made these conclusions based on one or two radioactive sources. Certainly, the behaviours of some graphs would change if the number of sources increased. As a result, for the time being, the conclusion about the best choice of gamma camera is valid only for environments with a low density of radioactive sources. Further experiments will be needed to investigate the best choice in environments with many radioactive sources.

4.7 Conclusion

In this chapter, we presented heuristic and MCDM exploration methods for the autonomous mapping and localization of elements in an environment after a nuclear disaster. These new methods were developed for mobile robots that use gamma cameras for discovering radioactive hotspots. As the limitations of gamma camera technologies (long acquisition time and poor AR) were two main challenges for available autonomous exploration methods, we developed and implemented new effective exploration algorithms to overcome these limitations.

From our results, we conclude that MCDM is an effective method for rescue robots. In particular, MCDM reported significant improvements in more complex situations (e.g., environment B) where MCDM can demonstrate its superiority by making good decisions. We also applied the new method (Algorithm 1) to discover the locations of real radioactive hotspots among many candidates. Moreover, we have also shown that the gamma camera with higher FOV and poor AR is more suitable rather than a gamma camera with lower FOV and good AR in environments with low density of radioactive sources.

As a result of the promising results, we could produce the maps and update the world model by fusing together the approved elements with other sensory data. In the next phase, we plan to apply the map and world model for computing a construction plan and then building protective walls. The search phase is a key part in reducing the time, and it can also improve the performance of AFAC by providing an accurate world model.

5 Computing the construction plan[1]

5.1 Introduction

After a terrible disaster, such as an earthquake or a nuclear accident, rescue operations begin immediately to find victims and isolate them from hazards. One of the most important considerations is making sure the post-disaster environment is safe for human responders. The existence of potential dangers such as radioactive sources or collapsing walls restricts human rescuers from performing rescue tasks and securing post-disaster environments. In these cases, rescue robots can assist in securing an environment by building protective structures. They can also be used to stabilize large structures or protecting the victims by performing construction tasks.

Post-disaster environments are usually unstructured, cluttered, and unknown, which is a big challenge when attempting to perform autonomous construction (Section 2.3.1). Now, the question is how will robots be able to autonomously build protective structures in post-disaster environments within a limited time? As we have already mentioned, AFAC is a suitable solution that drives robots for functional and adaptive construction, allowing the robots to build adaptive structures based on a defined purpose (e.g., block radioactive sources) without having any blueprints or prior knowledge of the environment.

In our scenario, we assume here this is an environment with radioactive sources, such as a nuclear laboratory incident. The goal of the rescue operation is to decrease victims' exposure by building protective walls. Therefore, we need to identify where the robots can build protective walls efficiently. In fact, we deal with two questions: where should the robots build walls, and in what order should they build them? Answering these questions provides a construction plan that drives robots for an effective construction for rescue operation.

[1] Most parts of this chapter were previously published in [7].

The goal of this chapter is to develop new methodologies for computing an effective construction plan based on the requested goals for AFAC. As shown in Figure 5.1, the `construction plan unit` starts to autonomously produce a construction plan based on the built maps and world model received from the exploration box; then, the robot builds the protective walls based on either a complete or incremental construction plan. In the case of an incremental construction plan, the robots explore, compute the plan, and build when a portion of the environment has been covered.

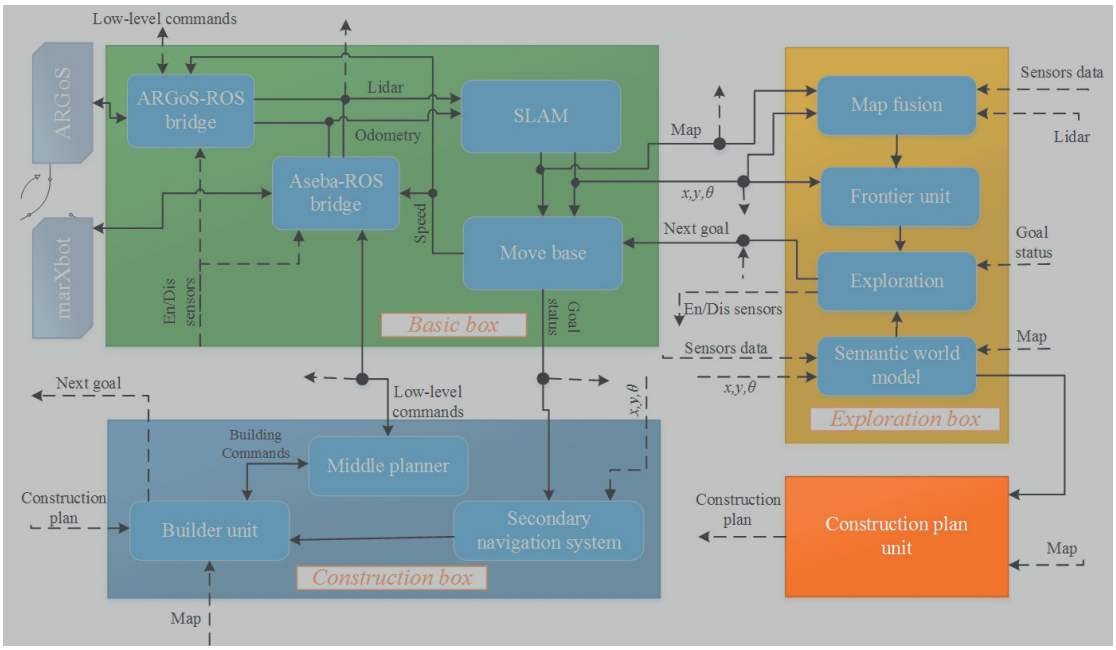


Figure 5.1 – The architecture of the autonomous construction system that directs robots to build protective structures. The construction plan unit autonomously produces a construction plan by receiving the built map and semantic world model.

The radioactive dose is one of the important factors in our scenario. Radiation can damage body tissues and organs depending on the effective amount of radiation absorbed by the body. Moreover, some electronic components, such as semiconductors, are very sensitive to radiation. Therefore, a construction plan must satisfy the following objectives: *victim safety*, *rescuer safety*, and *robot safety*. In other words, the goal of this construction plan is to minimize cumulative exposure to radiation for victims, rescuers, and robots. We thereby frame the construction planning problem as an optimization of these three objectives.

In order to achieve optimization, we applied a genetic algorithm (GA), which is an evolutionary algorithm based on an iterative search. GAs have the potential to find good approximate solutions for problems that are not able to be solved with standard optimization methods. Moreover, they can be straightforwardly applied to problems in which the objective function is non-differentiable, complex, discontinuous, or highly non-linear. The strength of GA is its

use of stochastic search to cover the solution space, combined with meta-heuristics that focus on promising regions in the space. However, GAs do not guarantee that an optimal solution will be found; we may consider their application to be an approximation method [117]. In this work, we benefit from the ability of GA and characterize our non-trivial problem to find a best solution. Our method may subsequently serve as a benchmark in comparing the performance of other construction plan methods.

Providing and optimizing the construction plan using GA will be the first contribution of this research. The second contribution will be to present the trade-offs involved between the three objectives. Finally, we will analyze how the objectives are sensitive to the effects of physical complexity (e.g., an increased number of victims or sources).

In Section 5.2, we review the literature on robotic exploration in terms of their construction plans. Section 5.3 provides a description of the problem and assumptions. Section 5.4 explains our proposed method to compute a construction plan based on a GA. In Section 5.5, we discuss the results. Finally, in Section 5.6, we conclude the research and propose future work.

5.2 Related work

We primarily review studies that develop algorithmic construction plans; although, most previous work on autonomous construction has focused on other aspects (Chapter 2). In terms of the construction plan, previous research on autonomous construction can be classified as follows: 1) using an entirely prespecified construction plan; 2) using a simple construction plan (e.g., wall in a straight line) with the influence of environmental features; and 3) using a construction plan influenced by environmental features without any prespecified shape to satisfy the abstract objectives.

Some researchers present construction of specific structures, in which the shape has been fully prespecified and requested by a user [14]. Werfel [50] demonstrated a simple simulation in which a group of robots tried to rearrange blocks into a 2D shape based on a high-level geometric shape. This shape was provided by the user, and the robots tried to build the structure. Moreover, Werfel et al. [51] presented a 3D collective construction in which the system received a high-level representation of a desired structure and translated it into some rules, based on collective robotic behaviours. Similarly, a team of quad-rotors were controlled to build 3D structures based on relatively complicated pre-specified shapes (e.g., block tower, truss) using blocks or rods [13].

Some construction algorithms take input from both prespecified shapes and the influence of environmental features. Melhuish et al. [118] reported a simple 2D wall constructed by robots, built over linear strips pre-placed on the floor. The robots used the linear strips as markers

and deposited materials at a certain distance away from them. In [119], an organizer robot coordinated the building activities of robots by generating a light-field pattern that varied in space and time. This light-field pattern was used as a template for robots to build a loose linear wall. Soleymani et al. [30] developed a construction system in which robots built a protective barrier that filled in a rectangular area.

Conversely, some research showed that construction plans do not need to be fully specified in advance, so they can allow a certain amount of dynamic flexibility during the construction process. Werfel et al. [54] proposed algorithms involving amorphous materials that enable robots to build arbitrary shapes. One of the ideas described is building an adaptive ramp, where robots fill a valley and provide access to the other side. In this case, a compilation procedure efficiently encodes a ramp structure for arbitrary shapes with a relatively small number of markers. This method allows robots to build ramps by locally reacting to markers. Werfel et al. [54] proposed an adaptive structure algorithm in which no predefined shape is considered, and environmental features (e.g., chemical spill shape) will conduct construction. In other words, a team of robots will build a protective barrier of a given thickness around a chemical spill. They will work reactively and will not analyze environmental features based on desired objectives to obtain an effective construction plan.

In conclusion, these three classifications span the range from entirely pre-specified construction plans to those whose shapes are determined by environmental features. In this thesis, we present an intelligent and robust construction system (AFAC) that works without any prespecified plan. AFAC applies two approaches: the local approach based on reactive behaviours and the global approach which is beyond the state-of-the-art. In the global approach, the computation algorithm takes into account global representations of the world and analyzes important environmental features to provide and optimize an effective plan for AFAC. In this chapter, we therefore address new methodologies for computing the effective plans required for the global approach of AFAC.

5.3 Scenario

5.3.1 Problem statement

We consider rescuers who want to employ robots for rescue in an environment that has toxic radiation. The existence of toxic sources makes the rescue operation dangerous and limits the actions of rescuers. In this situation, an instantaneous rescue effort might result in compounded problems. Therefore, post-disaster areas should be scanned and secured to avoid the problems associated with an unsafe environment. Robots can be useful in rescue operations, gathering information about the disaster area and then reducing the risks. For

instance, they can build protective walls to stabilize large structures or isolate victims from radioactive waves. In this research, we focus on building protective walls that minimize the radiation doses for victims and rescuers. Building these walls can also provide safe paths for victims and rescuers; rescuers can safely access victims to retrieve them and administer treatment.

Due to time limits and safety issues, a plan is necessary to guide construction tasks and meet objectives in an efficient way. The objectives are victim safety, rescuer safety, and robot safety. Victims are the first priority because cumulative toxic radiation exposure can seriously damage body tissue and organs. Rescuers need safe paths to perform rescue tasks; although, providing completely safe paths requires more time and robotic effort. Finally, some electronic components, such as semiconductors, are very sensitive to radiation, and this can cause failure in robots' performance. Therefore, in addition to enabling the rescue operation to be performed in the least time possible, the plan must also efficiently achieve these objectives. We benefit from GA in providing and optimizing the construction plan based on these objectives.

5.3.2 Assumptions

We assume there will be a rectangular flat environment surrounded by walls. It may also have internal walls. The environment is represented by an array of cells. Victims and sources are randomly placed in the environment. We assume that the toxic radiation sources are point-like and emit rays from their centres. we furthermore assume that they have the same intensity and emit linear and continuous rays.

According to the inverse square law for radiation fields, the radiation dose of a toxic source is inversely proportional to the square of the distance from the toxic source. As a result, the radiation dose of each point is established by adding the decreased doses of radiation sources [120], as can be seen in Equation 5.1:

$$g(x, y) = \sum_{i=1}^{N_r} \frac{I_i}{\sqrt{(x - X_{r_i})^2 + (y - Y_{r_i})^2}} \tag{5.1}$$

Where

x, y indicate the cell of the map

$g(x, y)$ gives the dose of the cell (x,y)

I_i is the intensity of the radiation source for the ith source

N_r is the number of radiation sources

X_{r_i}, Y_{r_i} is the position of the radiation source for the ith source.

In Equation 5.1, we compute cell dosages for each radiation source that is connected to the cell by using an imaginary straight line that does not intersect any walls or obstacles. Victims are represented by filled squares that are of the same size and shape. The doors are connected to safe regions, as illustrated in Figure 5.2. The robotic construction task consists of four steps: moving toward the safe region, picking up a block from the depot, moving toward the target place, and dropping the block, as shown in Figure 5.2.

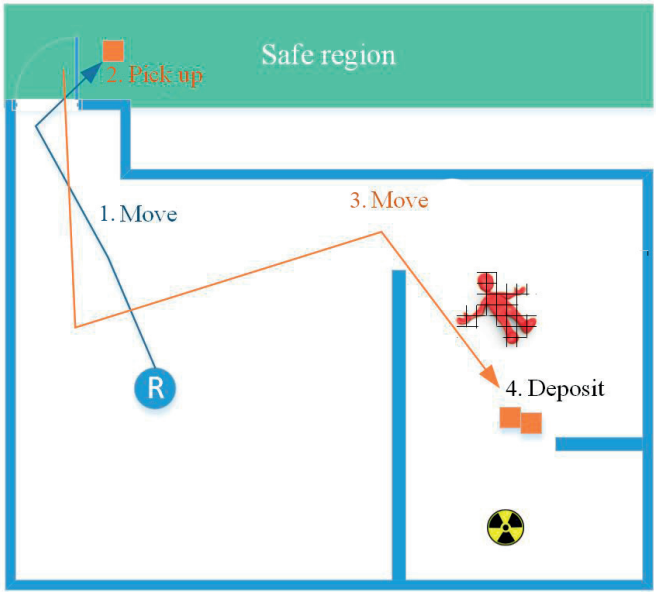


Figure 5.2 – A schematic of an environment with four robotic activities: 1. moving toward the safe region; 2. picking up the block; 3. moving towards the construction site; 4. dropping the block.

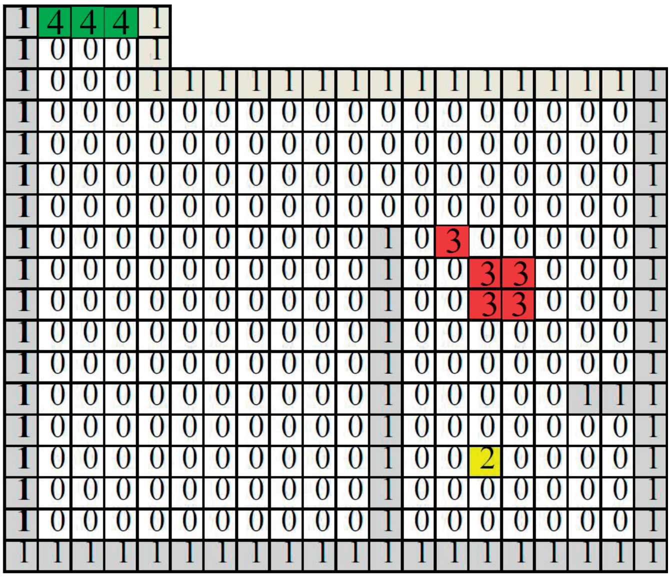
In our construction experiments,[2] the robot spends significantly more time placing blocks compared to moving; thus, we discount exposure to radiation for the robot during movement. The robot's exposure is also zero while picking up blocks because the depots are placed in a safe area. Therefore, the robot's exposure to radiation is only considered while the robot is dropping a block.

Calculating rescuer and robot exposure depends on having paths between victims and the safe area. Rescuers need a computed path to access victims, so we need to determine the shortest path between the victims and the nearest safe region before defining wall positioning. In accordance with the path-finding methods in the literature, we apply the wavefront path-finding method, which is commonly used in grid maps to find the shortest path [121].

[2] The experiments in Chapter 6.

5.4 Construction plan based on GAs

As already mentioned, we apply GA to compute and optimize the construction plan. First, we need to create a new map based on the geometric map in order to evaluate individuals in the population of GA. This map is an array of cells with a user-defined resolution. A specific value in each cell reflects which entity is placed in the cell. We call this map an entity map. The entity map was made from the environment represented in Figure 5.2, and is depicted in Figure 5.3. The value of each cell indicates which entity (wall, door, victim, or radioactive source) occupies it.



1	4	4	4	1															
1	0	0	0	1															
1	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
1	0	0	0	0	0	0	0	0	0	0	1	0	3	0	0	0	0	0	1
1	0	0	0	0	0	0	0	0	0	0	1	0	0	3	3	0	0	0	1
1	0	0	0	0	0	0	0	0	0	0	1	0	0	3	3	0	0	0	1
1	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1
1	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1
1	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1	1	1
1	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1
1	0	0	0	0	0	0	0	0	0	0	1	0	0	2	0	0	0	0	1
1	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1
1	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Figure 5.3 – A representation of an entity map, that reflects the environment in Figure 5.2; the number 0 represents a free cell, and the numbers 1, 2, 3, and 4 denote a cell being occupied by a wall (or obstacle), toxic source, victim, or door, respectively. This encoded environment will be used by GA to compute fitness values of individuals.

The geometric map is a probabilistic map; the value of each cell has to be converted to a binary value for the entity map. Thus, we define a threshold to consider cells less than the threshold as free cells and higher than it as wall cells.

In the next step, we need to encode protective walls to *chromosome* for use in GA. Each protective wall is supposed to be a segment. This segment is a straight line defined by its start and end positions. Both the start and end positions are included in an array of integers that makes up a chromosome (Figure 5.4). Each chromosome contains *genes* that are either start position (x_{s_j}, y_{s_j}) or end positions (x_{e_j}, y_{e_j}) of a segment. The chromosome k is of the

following form:

$$ch_k = [x_{s_1}, y_{s_1}, x_{e_1}, y_{e_1}, x_{s_2}, y_{s_2}, x_{e_2}, y_{e_2}, ..., x_{s_n}, y_{s_n}, x_{e_n}, y_{e_n}] \tag{5.2}$$

where

$$x_{s_j}, y_{s_j}, x_{e_j}, y_{e_j} \in \mathbb{N}, \ \forall j \in \{1, ..., n\}$$

The chromosome k consists of n genes, so each segment (or protective wall) has been defined by four genes of this chromosome.

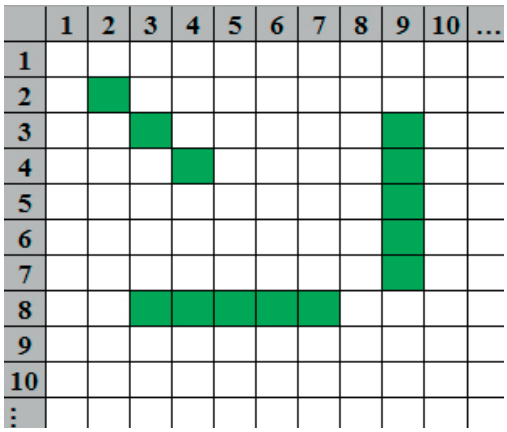


Figure 5.4 – A representation of the segments as protective walls; these protective walls are encoded to the chromosomes on an entity map, as follows: [2,2,4,4,3,9,7,9,8,3,8,7].

Now, we need a fitness function to evaluate chromosomes. We define the fitness function as a multiplicative form based on the three objectives that computes a fitness value for each chromosome. We multiply the objectives by the associated weights to produce a single fitness value. The fitness function is defined as follows:

$$fitness\_function(ch_k) = (1 + w_1 \times VE(ch_k) + w_2 \times RE(ch_k) + w_3 \times RESE(ch_k))^{-1} \tag{5.3}$$

$$w_1 + w_2 + w_3 = 1$$

where

w_1, w_2, w_3 are corresponding weights,

VE is the cumulative dose of radiation (i.e., gray[3] units) absorbed by victims during the whole process,

RE is the radiation dose absorbed by robots during construction,

$RESE$ is the radiation dose absorbed by rescuers.

The weights are arbitrary values that are usually determined according to user needs. For instance, if a robot is shielded from the radiation or the radiation intensity does not cause any trouble for the robot, its weight can be lower than the other weights or even set to zero. In Equation 5.3, the exposure terms (VE, RE, $RESE$) are inversely proportional to the discussed objectives; for instance, the *victim safety* objective is equal to *1/(victim cumulative exposure)*.

In each iteration, GA computes the fitness of all new chromosomes. The fitness function evaluates a chromosome (a set of segments) by laying genes (i.e., segments) of the chromosome on the entity map. In other words, we need to simulate the building of protective walls based on each chromosome in order to compute the fitness value. We measures all required parameters during the virtual building of segments, as one example, victims' radioactive exposure over time, to achieve the fitness value. The direction and order of each segment also affect the fitness value. After computing all fitnesses, GA exploits particular operations to produce a new generation. The operations of GA are as follows: selection, crossover, and mutation. First, all chromosomes are ranked from highest fitness to lowest fitness. Then, only a fraction of the population is selected to survive while the rest have to be replaced with a new generation of chromosomes. Fitness proportionate selection (roulette wheel selection) is used to select chromosomes in the population. We then assign a probability for every chromosome to breed the next generation. The probability assigned to a chromosome is proportional to its fitness, such that individuals with higher fitnesses have a greater chance of surviving than weaker ones. In the next step, two chromosomes selected from survival chromosomes produce two new offspring using a crossover operator. The crossover operator combines them based on a random crossover point; the parent chromosomes break into two parts based on the random point and then they combine to each other. By using mutation operators, we replace a certain percentage of the genes in the list of chromosomes. The initial population is generated at random. The maximal number of generations is set manually by the user, and was defined in an iterative way.

Table 5.1 – The values are assumed for the parameters of GA

Selection rate	Mutation rate	Population size
0.4	0.011	100

[3]The gray unit (Gy) is defined as the absorption of one joule of radiation energy by one kilogram of matter.

As mentioned, the fitness function simulates the construction processes for each chromosome (a set of protective walls) to compute the required values, such as victim and robot *exposure over time*. Hence, we need to assume user-defined values for the construction speed and rescue speed to compute the terms. Without a losing generality, we assume that construction speed (including all four tasks in Figure 5.2) is *one block per time unit* (e.g., $one\ block/minute$) and that rescuer speed is also equal to *1 cell per time unit* (we envision the environment as being composed of grid-cells, so the speeds are defined based on cells per time unit) and rescuers start after construction. Of course, if we change the construction speed, the range of cumulative exposure for victims and robots will change proportionally. Similarly, since the range of cumulative exposure for rescuers is proportional to the speed of the rescuers, it will decrease with an increase in rescuers' speed. Because the relationship between time spent and speeds is linear, we can scale the computed terms up or down according to the ratio between real speeds and what we assumed to be speeds.

5.5 Experiment evaluations

5.5.1 Experimental setup

We considered two environments with the same map but with two different configurations of elements: the first one is an environment with a victim, a toxic source, and two doors, which we considered a simple environment (Figure 5.6a); the second is an environment with two victims, two radiation sources, and two doors, which we considered a complex environment (Figure 5.6b). The maximum iteration numbers of GA for the simple and complex environments are 1000 and 2000 respectively. Other assumptions were presented in Section 5.3.2.

Ten trials were carried out for the selected weights configuration (each point inside the triangle of Figure 5.5 shows one weight configuration), and elements were scattered randomly in each trial.

5.5.2 Performance of GA for the construction plan computation

We demonstrated the detailed results of one selected trial that used the following weights: $w_1 = 0.3$, $w_2 = 0.35$, $w_3 = 0.35$. Figure 5.6 shows two construction plans made using GA. The protective walls are illustrated by a *green* line on the environment. The paths, sources, victims, and entry doors to the safe areas are illustrated by *blue*, *yellow*, *red*, and *brown*, respectively. For this sample, the protective walls did not completely protect the path, but the victims are completely protected. If a longer wall is considered in order to completely protect the path, the cumulative exposure of the robot will increase. This correlation shows that these two these two objectives are competing.

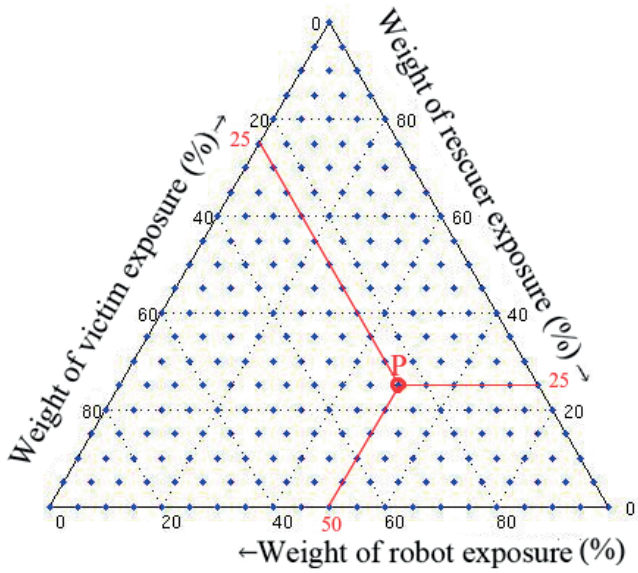
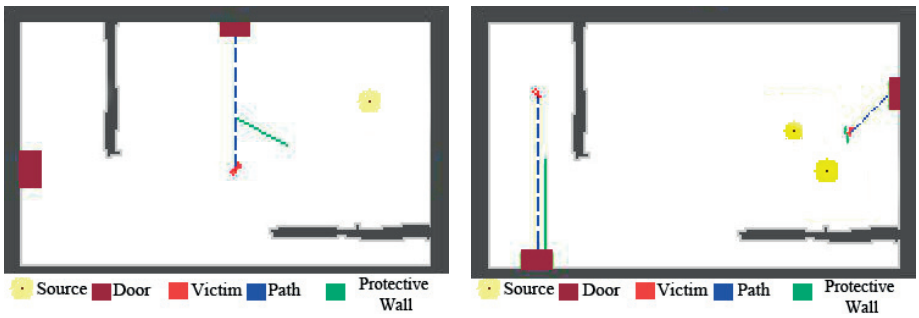


Figure 5.5 – The weight-balancing triangle, in which the blue points reflect weights used for the experiments, and point **P** shows a sample for the corresponding weights of each point. In this case, point **P** shows 0.25, 0.5, and 0.25 for w_1, w_2, and w_3, respectively.

Figure 5.7a illustrates the performance of GA in producing the construction plan. It shows that the fitness quickly reaches a steady state. The graphs shown in Figure 5.7b depict the exposure to radiation for victims versus time in the simple and complex environments. The graphs show that the construction plan aims to decrease the cumulative exposure of the victims (integral of victims exposure graph over time) quickly. Because the victims are exposed to radiation, the robot must try to protect them quickly. Otherwise, cumulative exposure will increase if the robots perform unnecessary or time-consuming tasks. The graphs shown in Figure 5.7c illustrate the exposure to radiation for robots versus time elapsed. The graphs show that the radiation exposure on the robot tends to decline over time as the robot builds protective walls that prevent further exposure. The line in the graphs are not smooth because the robots drop blocks in several places that have different radiation intensities.

5.5.3 Objectives trade-off

It is interesting to study the trade-offs involved between competing objectives (victim safety, robot safety, and rescuer safety). As described in Equation 5.3, the fitness function consists of three terms and corresponding weights. By changing the weights, we can manipulate the balance of the objectives. For instance, point P in Figure 5.5 shows 0.25, 0.5, and 0.25 for w_1, w_2, and w_3, respectively, so this construction plan keeps the robots safer than victims and rescuers.



(a) A post-disaster environment with a victim and a toxic source (simple environment)

(b) A post-disaster environment with two victims and two toxic sources (complex environment)

Figure 5.6 – Two construction plans computed by GA in which the *green* color depicts protective walls, the *blue*, *yellow*, *red*, and *brown* colors depict the paths, toxic sources, victims, and entry doors to the safe areas, respectively.

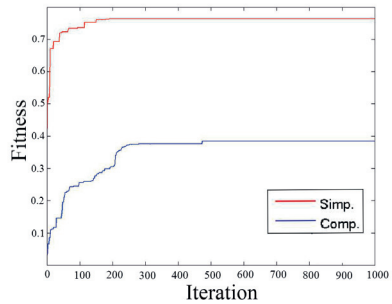
Therefore, the trade-off of the objectives is done by changing the corresponding weights. To represent the trade-offs between objectives, we create a particular triangle to plot all possible weights. Each point inside this triangle gives us the value of the corresponding weights. Each edge represents a variation of the weight between 0 and 1. Each edge is divided into 20 sections, and then this triangle provides 231 points of possible weights, as can be seen in Figure 5.5.

The result of this weight balancing is shown in Figure 5.8. For each point, the trials were carried out ten times, each time with a random arrangement of elements (victims, doors, sources). we assume two configurations: simple (one victim, two doors, one toxic source) and complex (two victims, two doors, two toxic sources).

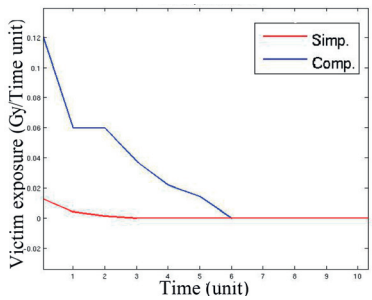
The interesting points of Figure 5.8 are as follows:

- The cumulative exposure does not have a uniform dispersion with respect to the weights. For instance, as seen in the average rescuer cumulative exposure for the complex configuration in Figure 5.8 (C2), if the rescuer weight equals 0.2 and we increase the robot weight (or decrease the victim weight), the cumulative rescuer exposure increases despite the constant rescuer weight. In other words, robots limit their presence in the environment when we enhance robot's importance by increasing the associated weight. Thus, the building of protective walls is decreased, which causes weaker protection. The dispersion of the cumulative exposure in respect to the weights shows that these objectives are competing.

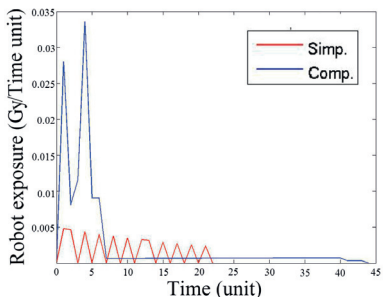
- A comparison of simple and complex configurations shows that exposure dispersion patterns stay the same. This is an interesting point because it means we can potentially define a particular region inside the triangles to guarantee the behaviour of the con-



(a) Fitness value versus iteration to show performance of GA for the selected situation



(b) Victim radiation exposure versus time during the execution of the construction plan



(c) Robot radiation exposure versus time during the execution of the construction plan

Figure 5.7 – The graphs of construction plan performance using GAs for a set of arbitrary weights ($w_1 = 0.3, w_2 = 0.35, w_3 = 0.35$) in two selected situations; the red line shows the simple environment and the blue line shows the complex environment.

struction plan. We call this region the *reliable area*; this area gives users a set of reliable weights. These weights provide a construction plan in which cumulative exposure to victims, robots, and rescuers will be less than 80% of their maximum values. The cumulative exposure will be its maximum value if the corresponding weight is equal to zero.

- Our results also show that the objectives have different impacts on fitness values. For instance, the construction plan immediately protected the victims for even low associated weights [Figure 5.8(A2)] while the paths were not well-protected for low weights [Figure 5.8(C2)].

- In addition to the victims' and rescuers' exposure amounts, Figure 5.8(D) shows the relationship between protective wall lengths and the weights. The dispersion pattern of the total protective wall length changed by increasing their complexity. We expect the wall lengths to increase by a factor of four because the number of victims and number of sources were multiplied by two. However, the result indicates that the wall lengths less

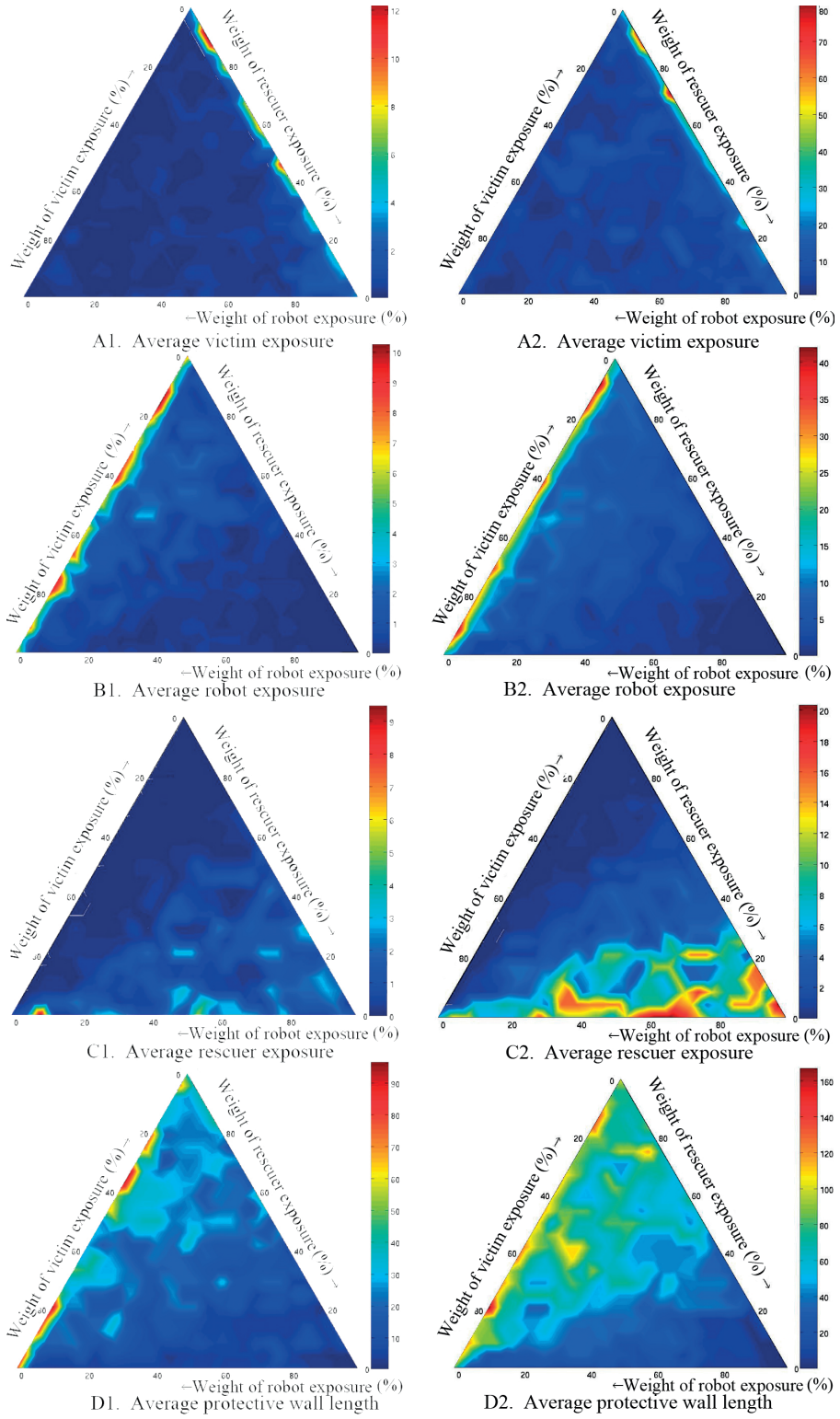


Figure 5.8 – Objectives' trade-off results. The triangles on the left belong to the simple environment configuration and the triangles on the right belong to the complex environment configuration. The triangles were made from many points that were the average of ten trials.

than doubled. This shows that there is some overlap between the protective walls or that some parts of the protective walls are able to secure wider regions of the environment than expected.

5.6 Conclusion

In this thesis, we present an autonomous construction system (AFAC) that is based on a global approach and that can build protective structures and save victims. Because saving human lives in the least amount time without imposing additional threats is an important issue in the rescue operation, an effective construction plan is required to lead AFAC in a secure and efficient way.

Here, we applied GA to compute and optimize the construction plan required for the global approach of AFAC. The fitness function of GA consists of three terms linked to main objectives (victim safety, robot safety, and rescuer safety) and the corresponding weights; with this, GA evaluates the proposed plan to choose a final plan with the highest fitness value. We also encoded the environment, knowledge of the world, and protective wall candidates for use by GA.

By studying the trade-offs between these objectives, we have shown the impact of the chosen weighting approach, highlighting, for instance, that there are correlations between victim and robot safety. We also determined a set of reliable weights that ensure 80% of the maximum cumulative exposure value for the three objectives.

One problem with GAs is that they typically require significant hardware capacity and computation time, especially in the case of a large environment with many radioactive sources and victims. We will speak about an algorithm for finding quick solutions using less computations in Chapter 7.

6 Portable and autonomous robotic construction[1]

6.1 Introduction

Developments in robotics sciences have recently led to the use of various robotic platforms to achieve construction automation objectives, although fully automated construction is still a dream of civil engineers. Robotic developments have shown that robots could potentially perform construction tasks where human presence is impossible, undesirable, or intensively expensive. Construction in hazardous areas after natural or man-made disasters or construction under difficult physical conditions (e,g., outer space) are situations that are not readily accessible to humans or that at least require initial structures to be prepared for human arrival.

Robots can be used to build required structures for these types of particular situations with various levels of autonomy, without explicit human intervention or with some level of planning interaction conducted with a human supervisor. A robot performing autonomous construction has to adapt itself to the sensed environment, make decisions regarding the execution of its task, and re-plan when its task is not executable. Mobile robots represent one type of robotic system that can be used for autonmous construction. Applying mobile robots to construction introduces new capabilities to the field. For instance, building large structures without being confined by dimensions is a challenge for current technologies; we might need huge and expensive fixed-base fabrication systems (e.g., contour crafting [23]) to build giant structures. Capabilities of mobile robots, on the other hand, allow them to create objects without fixed-base system constraints (e.g., the size of a printer's frame).

In contrast to these advantages, mobile robots, by nature, do not have a fixed referential point and their positioning systems are not as accurate as fixed-based systems. Existing construction processes, like many other additive manufacturing processes, are mostly based on precise positioning, which is achieved by machines that have a fixed mechanical link to the con-

[1] Most parts of this chapter were previously published in [41].

struction and rely on absolute positioning. Therefore, mobile robots have to compensate for this weakness by incorporating technologies or methods for better precision in construction processes. Equipping robots with external tracking systems (e.g., GPS, camera) may provide an accurate positioning system, but we aim to implement and study localization methods based on a self-positioning system that allows for autonomous handling of construction tasks, especially where external tracking systems are difficult to have installed (e.g., outer space). However, the accuracy of current self-positioning systems that are not linked to the built structure are not sufficient to handle construction processes in an unknown environment [3].

In this chapter, we combine a global positioning system with a short-range relative localization system (we called it SRL) to compensate the required precision for the construction of structures that are spatially separate from one another, which we refer to here as separated artifacts. The aim is for a mobile robot to build coherent artifacts using both SLAM and SRL, and based on a human-prescribed blueprint. The robot employs a laser range scanner to autonomously map an environment and to determine its current position using SLAM. The precision of SLAM as a self-positioning system is not sufficient to support construction activities, but it is well-suited to globally determine the location of the robot. SRL serves as supplementary system for construction, to compensate for the inaccuracies of SLAM. In SRL, the robot's gripper, which is equipped with many IR sensors, sense previously placed blocks for dropping next blocks adjacent to them. Further to implementing the construction system for building separated artifacts, we need to assess the performance of our approach by measuring the built artifacts. The assessment greatly benefits us in improving the autonomous construction system (AFAC) that used SLAM and SRL.

The scenario, assumptions, and control system are provided in Section 6.2. The results and discussion are presented in Section 6.3. Finally, in Section 6.4, we conclude this study.

6.2 System description

6.2.1 Scenario and assumptions

As presented in the introduction, the robot has to build separate artifacts based on a given blueprint in an unknown environment. We assume that the initial position of the robot is the location of a block repository. We put a new block in the repository for each step of construction. The robot has to detect the block and align itself with the block to pick up the block with the gripper in the correct position.

The arena is a 200 cm× 100 cm rectangle with a flat surface that contains a few obstacles. Note that the environment is unknown for the robot, and SLAM is used to inform the robot of its position, and to map the environment for the path planning as well. The artifacts are simple

polystyrene blocks with an attached strip of ferromagnetic metal on the lower part of the body (Figure 3.15). Each block is 6 cm in length, 6 cm in width, and 18 cm in height, and each one weighs approximately 20 g. The size and weight of the block was chosen to satisfy the requirements of the robot's gripper and laser range scanner. Specifically, the block must be light enough to be held by the gripper without falling. In addition, the minimum height of the block must be such that the laser range scanner is able to detect it.

After grasping the block, the robot moves towards the destination point. The first block of the artifact will be dropped after the robot fine-tunes its location. The robot returns and takes a new block from the repository. Now, the robot is ready to drop the second block of the artifact. The robot drops the second block beside the first block using SRL.

6.2.2 Robot platform

We used the marXbot robot (Figure 3.5), which was introduced in Section 3.2. It is a miniature robot that consists of five modules such as the gripper.

6.2.3 Control architecture

The required algorithms and packages for building the artifacts are implemented inside the `construction box` of Figure 6.1. In fact, the `construction box` is designed for AFAC, so it has to act based on the flexible construction plan computed by the `construction plan unit`. However, we aim to evaluate the precision of built artifacts by comparing them with the desired plan; the robot builds artifacts based on a predefined plan (i.e., blueprint) given by a human user instead of the construction plan. The `construction box` and `basic box` are only needed to perform construction based on the predefined plan.

We have already described all packages of `basic box` in Chapter 3. The robot navigation is implemented through a collection of ROS packages, including navigation (`move base`[2]) and SLAM (`Hector mapping`[3]) packages. The `move base` package receives a goal and directs the robot to the requested point. The `SLAM` unit provides the location of the robot. The `ARGoS-ROS Bridge` and `Aseba-ROS Bridge` units are middle tools that connect other units to the simulated or real robot.

As shown in Figure 6.1, the `construction box` is based on three subcomponents: `builder unit`, `middle planner`, and `secondary navigation system`. Below, we describe each

[2] `move base` is a 2D navigation stack that receives information from sensors and then directs the mobile robot by determining a safe speed and reliable paths to the target position; see Section 3.5 for more information.

[3] `Hector mapping` is a kind of SLAM algorithm that uses a laser range scanner; see Section 3.5 for more information.

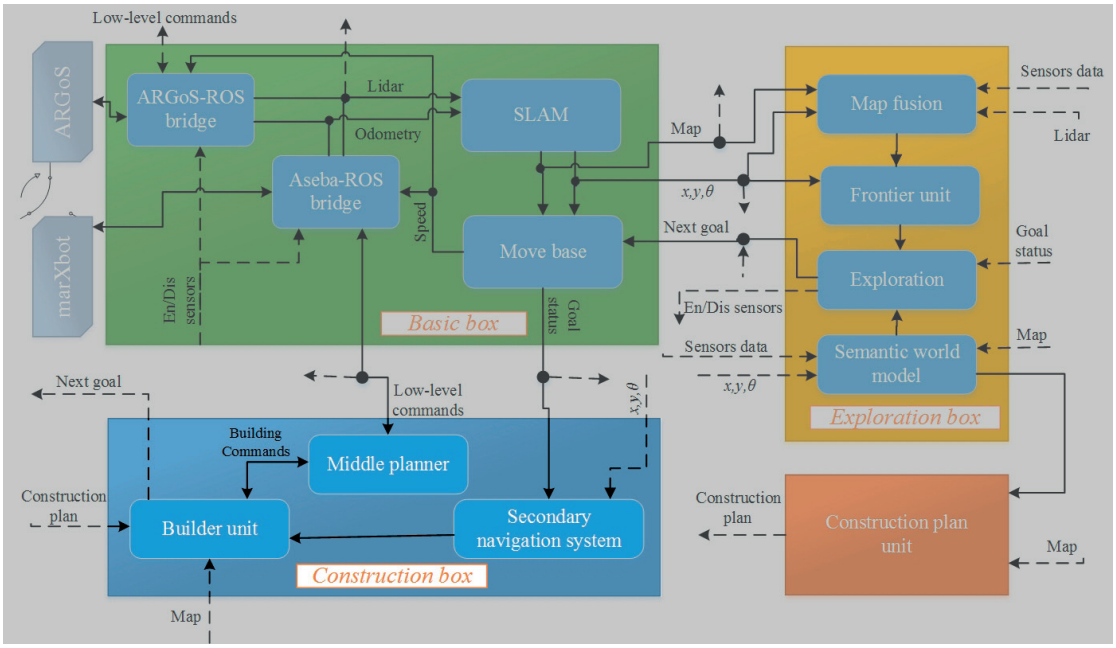


Figure 6.1 – The `construction box` is represented as one part of the AFAC architecture used for the rescue operation. It consists of three subcomponents: `builder unit`, `middle planner`, `secondary controller`.

subcomponent.

6.2.3.1 Builder unit

The `builder unit` executes overall construction of separated artifacts by generating a sequence of high-level behaviours. These behaviours either take the form of target poses for robot movement, which are then delegated to the `move base` package of `basic box`, or block manipulation behaviours (such as *pickup* and *placing* behaviours), which are then delegated to the `middle planner` package.

6.2.3.2 Secondary navigation system

This package operates as an auxiliary system for the navigation system. This means that the `move base` drives the robot to reach a region where it is very close to the goal position, and then, the `secondary navigation system` slowly moves the robot to precisely reach the goal position. Because this auxiliary system is time-consuming, we apply it where we require a high level of precision, for instance, in placing the first block of the artifact.

The `secondary navigation system` is made up of two PID controllers that are used to improve the robot's position. The first one is used to adjust the robot in ${}^0X$ or ${}^0Y$ positions

(*translational controller*); the second one is used to place the robot at the determined yaw angle (*rotational controller*). Figure 6.2 shows a block diagram of the PID controller used by either the translational controller or the rotational controller. The robot estimates its position using SLAM and then gracefully moves to improve its location.

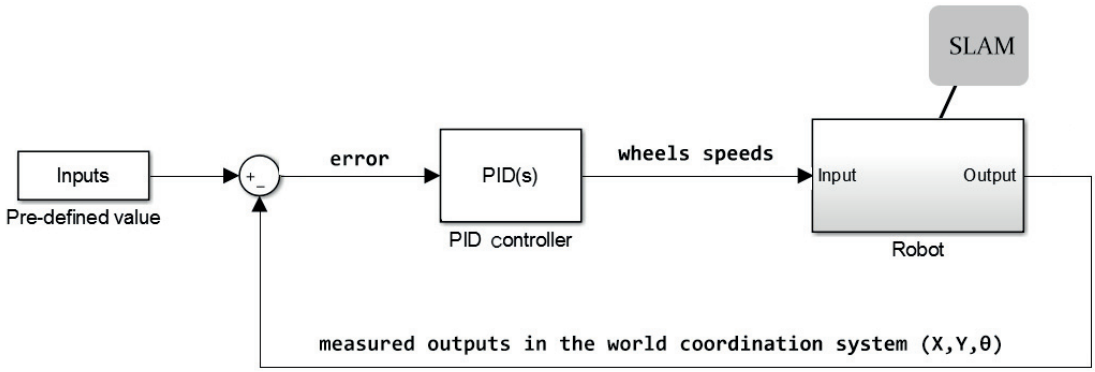


Figure 6.2 – A block diagram represents the PID controller is used by the translational controller or rotational controller. SLAM provides the robot's position (${}^0X$, ${}^0Y$, yaw angle) in the world coordination system.

More specifically, the `secondary navigation system` works as follows:

1) After receiving a success message from the `move base` package, the robot uses the rotational controller to rotate in place and to align with ${}^0X$-axis. Then the robot activates the translational controller to gracefully move along ${}^0X$-axis. 2) Similarly, the robot rotates in place to align with ${}^0Y$-axis. Then the robot moves along ${}^0Y$-axis to accurately reach the expected position. 3) In the last step, the robot rotates in place to establish the correct yaw angle. Performing these three steps results in the robot improving its location in respect to the target point.

The rotational and translational controllers are established based on the PID controller. Table 6.1 show the gains used for the translational and rotational controllers. The rotational controller has applied several gains in three different conditions. When the $\delta\theta_{error}$ is high, we need to decrease the rotational speed with decreasing gains. In contrast, for a small amount of $\delta\theta_{error}$, it is usually better to increase the gains, because the robot does not move properly for a very low error value. Fine-tuning the gains allows the robot to have a continuous rotation.

6.2.3.3 Middle planner

The `middle planner` is an intermediate package between the high-level construction commands and low-level subtasks. It translates high-level commands into a set of low-level subtasks for block manipulation. Each of these low-level subtasks is implemented by a finite-

Table 6.1 – The PID gains of the translational and rotational controllers

Controller	Condition	K_p	K_i	K_d
Rotational	$\delta\theta_{error} >= \pi/4$	0.1	0.0	0.1
	$\pi/20 < \delta\theta_{error} < \pi/4$	0.1	0.5	0.1
	$\delta\theta_{error} <= \pi/20$	1.6	1.0	0.1
Translational	-	0.4	0.1	0.1

state machine controller that at 5Hz senses its surroundings by using the gripper's infrared sensors and actuates the treel and gripper motors. Because of sensory inaccuracies and environmental imperfections, the low-level subtasks are not deterministic in their effects. For instance, a brief mismeasurement of infrared distances caused by fluctuations in ambient lighting could cause the *align-gripper-angle* controller to over-rotate the gripper such that the robot loses sight of the block with which it is aligning. When such unintended effects occur, the robot's conditional plan gracefully recovers by selecting the next appropriate behaviour [122]. In essence, the conditional plan creates complex and dynamic sequence of subtasks that are theoretically guaranteed [123] to eventually allow the robot to accomplish its manipulation goal.

As illustrated in Figure 6.3, the green blocks show high-level building commands that come from the `builder unit`. Moving down this hierarchy, the `middle planner` interprets each building command and breaks it into subtasks such as finding a block (or cube), approaching the block, and so forth. A building command is accomplished when a sequence of these subtasks is successfully finished. These subtasks are also fed from sensor data, allowing the `middle planner` to know the latest state of a subtask.

The bilateral interaction between the subtask and module levels allows troubleshooting and fault correction of actions on the construction behaviours to be performed. If the expected effects are not produced, the robot has to infer that a fault has occurred (e.g., the robot has not successfully grasped a block) and what type of fault occurred (e.g., the magnet of the gripper did not work well and infrared sensors can no longer see a block). The `middle planner` will then make a new decision at the subtask level to correct for faults (e.g., *find-cube*) [122].

6.2.4 Behaviours for the building

The `builder unit` sends high-level commands for carrying out construction based on the defined plan. These high-level commands are translated subtasks and low-level commands by the `middle planner`. The high-level commands are as follows:

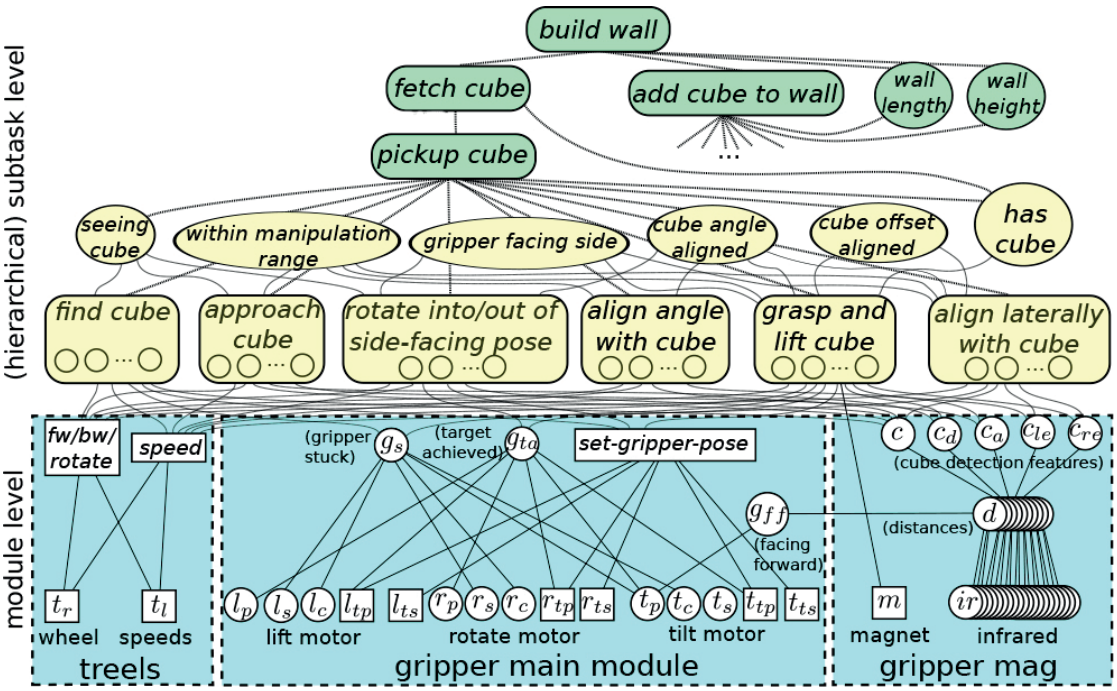


Figure 6.3 – A hierarchical representation used in [122] shows interactions between high-level building commands, low-level subtasks, and module levels. A similar structure is used for implementation of construction behaviours (e.g., *pick-up*) in our systems.

6.2.4.1 Pick-up

The *pick-up* function has to enable an accurate grasp on blocks because a misaligned block will inflict errors upon all subsequent operations. As illustrated in Figure 6.4, the `middle planner` generates a sequence of movements. First, the robot turns to face the repository. The `secondary navigation system` helps the robot to precisely align itself at the right yaw angle. The `middle planner` starts with the *find_block* subtask; it looks for the nearest block (1). Then, the robot moves slightly towards the detected block and uses the gripper's front infrared sensors of the gripper to fine-tune its movement, by calling *approach-cube* subtask (2). The next subtask is *rotate-around-gripper*, where the robot rotates 90 degrees to the right, and the gripper rotates in the opposite direction (3). This is where the robot needs to align itself laterally. Using the front infrared sensors on the magnetic gripper helps the robot contact the block with the centre of the gripper; the *align-laterally* subtask is performed during this part (4). The *rotate-around-gripper* subtask is called when the block is centred on the gripper's edge; the robot performs a 90-degree leftward rotation while the gripper rotates in the opposite direction (5). It then moves forward and rotates a few degrees in order to touch the block at the corner of where the magnet is located, whether on the right or left side of gripper (6). When the robot touches the block, it grasps it and lifts the gripper; the last processes is done by the *grip-and-carry-cube* subtask (7).

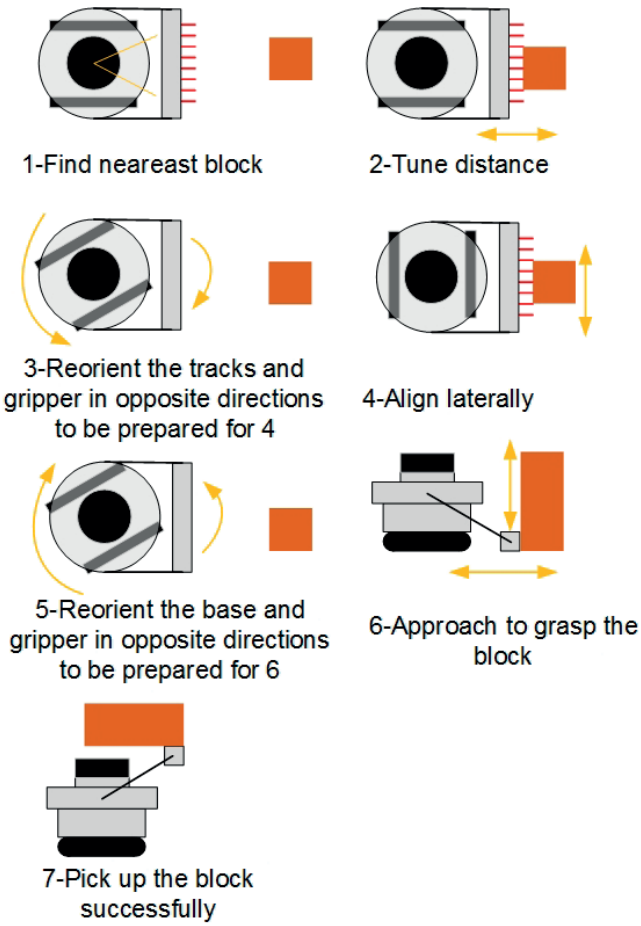


Figure 6.4 – The sequence of subtasks for picking up a block.

6.2.4.2 Traveling

This behaviour consists of raising the gripper in order to avoid blinding detectors such as the laser scanner and camera when the robot moves.

6.2.4.3 Drop/Place-adjacent-to

This behaviour consists of dropping a block either in the first position for the artifact or directly adjacent to existing blocks. To begin to build an artifact, the robot has to drop its first block. To do this, it simply lowers the gripper and disengages its magnetic switchable device. It then lifts its manipulator slightly and moves back a short distance. The *place-cube* subtask performs this processes.

For the remaining blocks of the artifact, the `middle planner` activates *find-block*, by which the robot moves toward the artifact (1). Then by use of the *approach-cube* subtask, the robot

starts to scan the wall using the infrared sensors of the gripper. The robot computes the distance to the artifact and moves accordingly towards this location; the *align-with-wall-edge* subtask performs this role (2). The next subtask is *rotate-around-gripper*, in which where the robot rotates 90 degrees, and while the gripper rotates in the opposite direction with the same angular speed. The `middle planner` calls the *align-laterally* subtask, by which the robot aligns itself laterally; and then, it calls the *align-with-wall-edge* subtask to find the left edge[4] of the wall and to drop a new block exactly beside the wall (4). After receiving a success message for the current subtask, the `middle planner` performs the *rotate-around-gripper* subtask; the robot rotates a 90 degrees to the left while the gripper rotates in the opposite direction (5). The final subtask is *place-cube*, which functions as follow: when the robot moves forward, it lowers and rotates the gripper by a few degrees to avoid collision with the other blocks (6). It then lowers its gripper slightly, disengages the magnet, and moves back a short distance. Finally, the robot tilts the gripper and moves forward to push and line up the block (7).

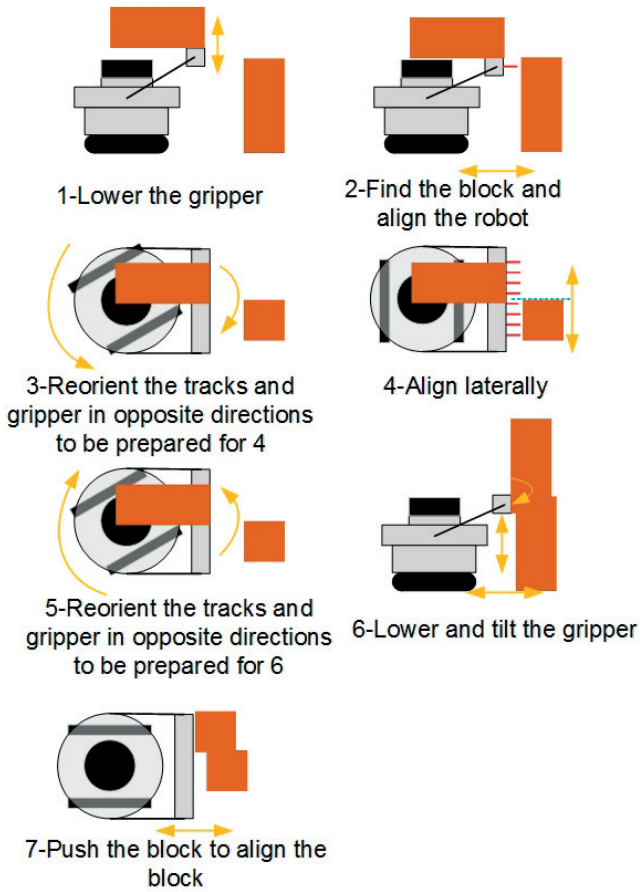


Figure 6.5 – The sequence of subtasks for dropping a new block adjacent to the existing one.

[4] Or the right edge, depending on whether the block was grasped on the right or left side

6.2.5 Robotic activities for construction

In this section, we briefly describe how the robot performs construction tasks to build artifacts. The robot has to first come back to the depot site to pick up a new block. When it reaches the depot site, the `secondary navigation system` rotates the robot to precisely achieve the goal yaw angle. Then, the `construction unit` package makes the request to run the *pick-up* behaviour from the `middle planner`. The `middle planner` calls a sequence of subtasks to pick up the block. If the robot succeeds in grasping it, the `middle planner` sends a success feedback.

Now, the robot has picked up a block and is ready to move towards the next construction point. If this is the first block, the robot drops the block after fine-tuning its position by use of the `secondary navigation system`. Otherwise, the robot drops the block beside the built wall with the help of SRL. If the robot succeeds in finishing the task, it will return to the depot site for the next block. The pseudo-code, which is illustrated in Algorithm 4, demonstrates the building algorithm; it is implemented in the `construction unit`.

6.3 Experiments

The marXbot is employed to build coherent separated artifacts based on two approaches: using both SLAM (as a self-positioning system) and SRL or by using only SLAM. Furthermore, to implement the construction system for building separated artifacts based on a given blueprint, we assess the performance of our approaches by comparing the built artifacts with the desired outcome. The results show how well the robots can build the artifacts.

6.3.1 Description

The desired arrangement of blocks is illustrated in Figure 6.6. This arrangement includes three artifacts that were made with 3, 4, 3 blocks. The robot starts from the indicated location with (0,0) and picks up a new block from the depot site (the yellow region marked with a D). Ideally, the robot has to drop off all ten blocks at the desired locations. The numbers on the blocks indicate their numerical order for construction.

We performed two types of experiments for artifacts building. First, the robot placed three blocks based on given positions; this was done to illustrate the precision of SLAM. It only dropped blocks B1, B4, and B8 (the first block of each artifact). In the second experiment, the robot employed both SLAM and SRL to build several separated artifacts, each made up of several blocks. Again, the robot employed the laser range scanner for SLAM. For SRL, the robot used infrared sensors to align itself to the blocks that were already placed as part of the artifact.

Algorithm 4 Pseudo-code for automated building

```
function MAIN()
    while (next_point is not NULL) do
        if (mode == Idle) then
            {next_point, side, first_block} ← NEXT-CONSTRUCTION-POINT()
            MOVE(depot_point)
            mode ← toward_depot_site
            state ← Start
        else if (mode == toward_depot_site) and (state == Complete) then
            PICK-UP()
            mode ← pick_up
            state ← Start
        else if (mode == pick_up) and (state == Complete) then
            MOVE(next_point)
            mode ← toward_construction_site
            state ← Start
        else if (mode == toward_construction_site) and (state == Complete) then
            if (first_block) then
                DROP-ALONE()
            else
                DROP-ADJACENT-TO(side)                ▷ side is either left or right
            mode ← Idle
            state ← Start
        else if (state == Fail) then
            RECOVERY(mode, state, fail_reason_code)
            mode ← recovery
            state ← Start
    return

function GET-FEEDBACK(feedback) ▷ This function is activated since other nodes send
new feedback
    if feedback == success then
        state ← Complete
        return success_reason_code
    else if feedback == Fail then
        state ← Fail
        return fail_reason_code
```

Then, the robot dropped the new block by activating the *place-adjacent-to* behaviour. Ten trials were carried out for each type of experiment. After the artifact construction was finished, we took a photo to measure the precision of construction. We extracted the dark color of the blocks' tops through image processing methods in order to compare the position of each built artifact with the desired arrangement of blocks. We matched the processed images of the built

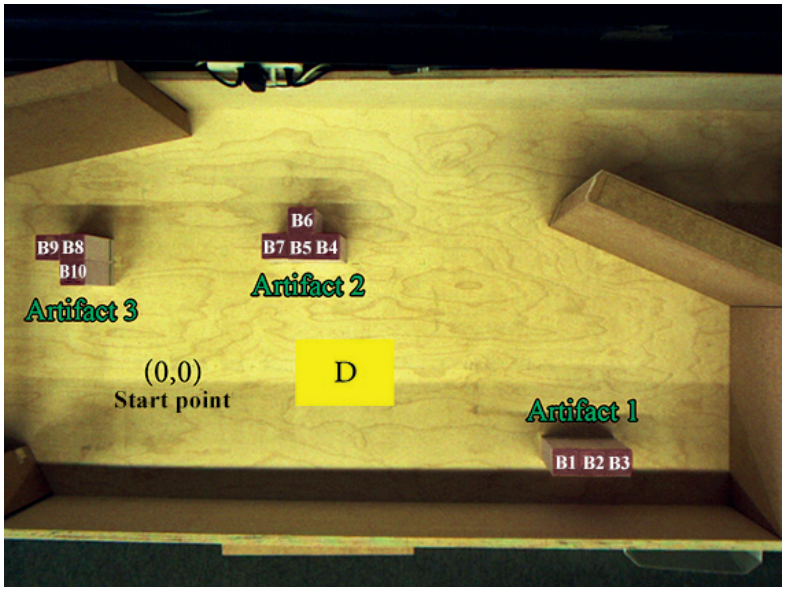


Figure 6.6 – The desired arrangement of the blocks is shown for the three artifacts. A comparison of the position of built artifacts with the desired arrangement demonstrates how precise the construction was. The robot starts from the (0,0) point to take a block from the repository. The numbers on the blocks indicate their numerical order for construction.

artifacts with the images of the desired artifacts to measure the translational and rotational errors of the blocks' positions on a 2D plane. As illustrated in Figure 6.7b, the distance between the centres of two corresponding blocks indicates the translational error; the angle between the same edges of the corresponding blocks also shows rotational error.

6.3.2 Results and discussions

In the first type of experiment, the robot only used SLAM to drop blocks B1, B4, and B8 (the first block of each artifact). The error of placement based on using only SLAM, one third of the block size, shows that we cannot use this positioning technique to build artifacts made of several blocks; if the robot employs only SLAM, the positioning error will cause collisions among blocks or create gaps between the blocks in the artifacts. Nevertheless, the robot can find the approximate position for construction using SLAM, but it must apply other methods (e.g., SRL) to successfully accomplish artifact construction.

(a) A representation of gaps between the blocks

(b) The comparison between a placed block and its desired location

Figure 6.7 – The left image shows the built artifact in respect to the defined blueprint. The brown squares are real blocks, and the white squares represent the desired location of the blocks. The g4 and g5 show the gaps between two adjacent blocks. The right image shows processed image of the second artifact and measure the translational and rotational errors for the fourth block (B4); d shows the distance between the centre of the squares, and θ shows the angle between the lower edges of the squares. The green color also depicts the overlapping area between the fourth block (B4) and the desired artifact.

In the second experiment, the robot employed both SLAM and SRL to build three artifacts (Figure 6.8). Because the artifacts are not fixed to the ground, dropping a new block may cause a displacement of the existing blocks. In other words, the locations of the first blocks of the artifacts (B1, B4, and B8) may be changed by the dropping of neighbouring blocks.

This explains why we might not expect the same graphs for the first blocks of artifacts (B1, B4, and B8) in Figure 6.10 and Figure 6.9; although, only SLAM was used for both of them. If the artifacts were fixed to the ground, the robots could use force sensors to push the blocks without influencing the precision of blocks that have been already dropped in place.

We measured the surface of occupied by placed blocks on each desired artifact. Ideally, the overlap percentage between the real blocks and the desired blueprint should be 100%. Figure 6.11 shows the overlap percentages for two construction types: single-block construction and multiple-block artifact construction. Note that the average percentage in the single-block construction (SLAM) is 64.41%, but it is 73.53% in artifact construction (SLAM and SRL). This increase performance does not mean that multiple-block artifacts are placed more precisely. Adjacent blocks could compensate for the positioning error regarding the global coverage of the artifact surface. This increase in performance shows that SRL may enable robots to build more coherent artifacts.

For a better understanding of the extent to which the blocks could be more coherent, we demonstrate the average size of gaps created between adjacent blocks. In Figure 6.7a, we

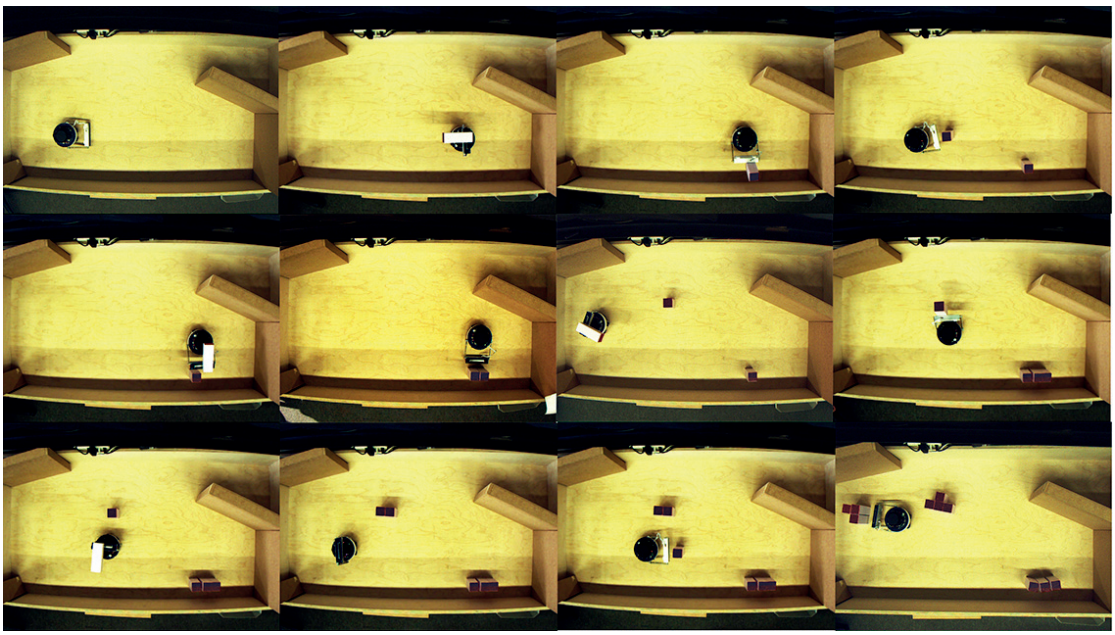
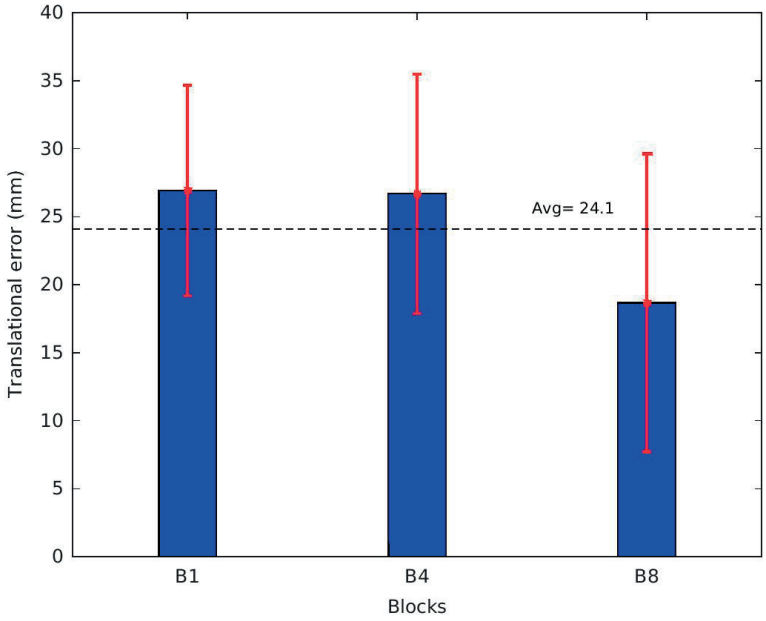


Figure 6.8 – Image sequence of artifact construction using both SLAM and SRL.

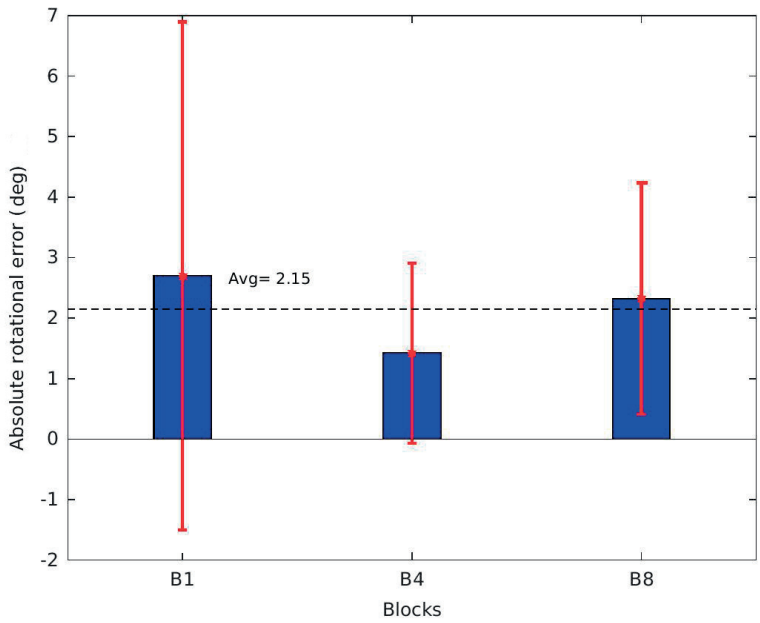
noted two gaps (g4 and g5) made in the second artifact; for instance, g4 is the minimum distance between two adjacent blocks B5 and B6. The chart in Figure 6.12 shows that the average of the gaps is about 4.608 mm; however, the translational error is about 21.89 mm as seen in Figure 6.10. The robot could reduce the gaps that could appear in the absence of SRL. In other words, if we build the artifacts based on the use of SLAM only, the gaps might be much wider than 4.608 mm because the average of the transitional error is around 24.1 mm; or conflicts between blocks might prevent the successful completion of construction.

Building coherent structures is a significant factor influences the performance of AFAC for rescue applications. For instance, isolating radioactive sources is a rescue task that can be supported by AFAC, but protective walls in front of the sources have to be coherent, or else dangerous waves will pass through gaps in the walls.

In our experiment, the translation error for both approaches was close to one-hundredth of the diagonal of the environment in a static environment, where we applied a miniature robot with a low-cost laser range scanner. It is difficult to evaluate how this error will scale in a real environment because SLAM accuracy depends on numerous factors. The quality and quantity of landmarks impacts the estimation of position by the SLAM algorithm. A dynamic environment can result in a loss of precision because dynamic obstacles can hide interesting landmarks. Finally, the quality of the sensor for distance measurement impacts the whole system directly. There is an obvious need for further developments to the sensory system and SLAM algorithms for building complex artifacts in real and large environments. Despite a

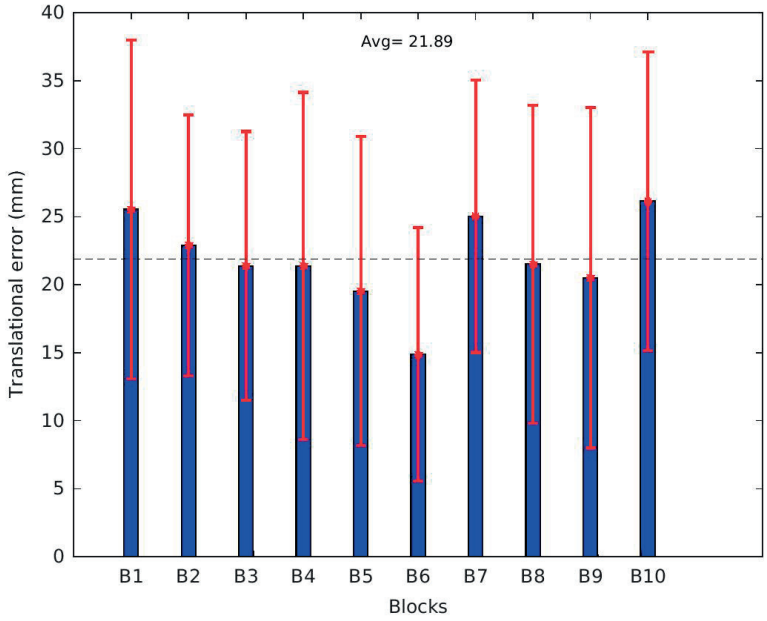


(a) Translational error

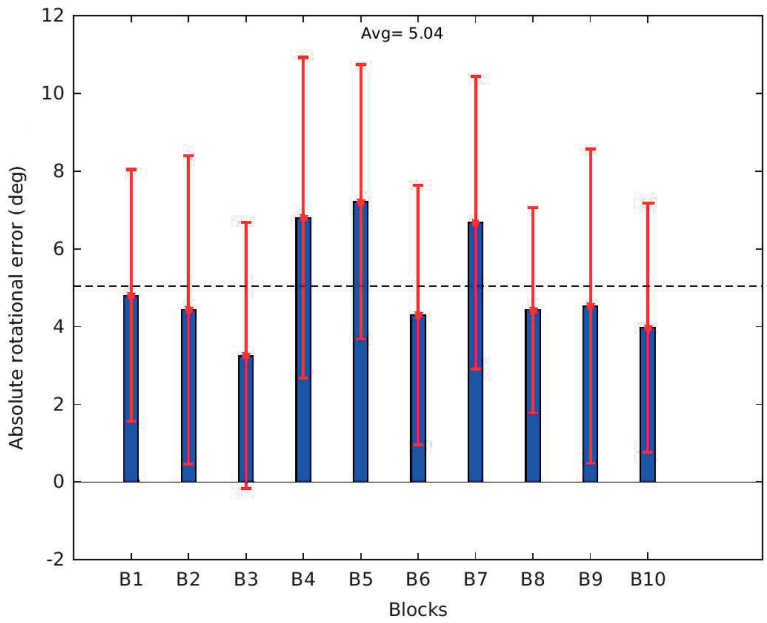


(b) Rotational error

Figure 6.9 – The chart on the left shows the translational error and the chart on the right shows the rotational error in the robot's placement of the first, fourth, and eighth blocks using only SLAM.

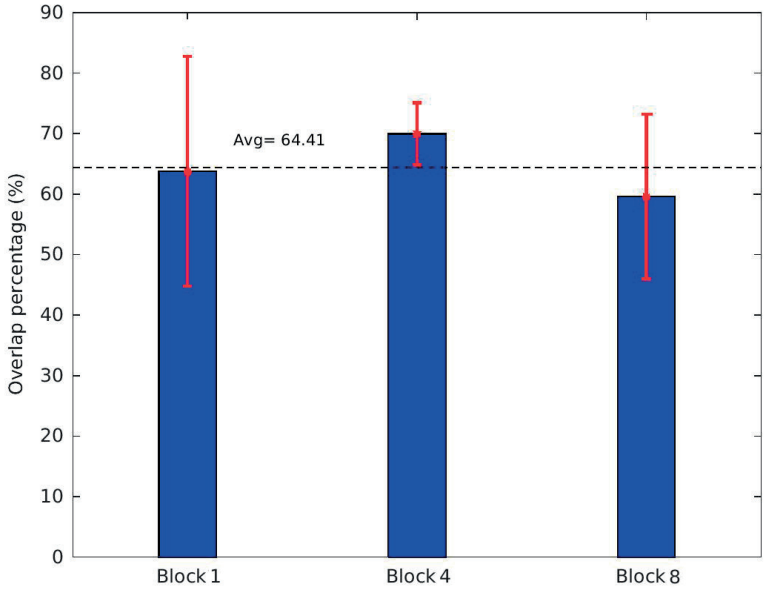


(a) Translational error

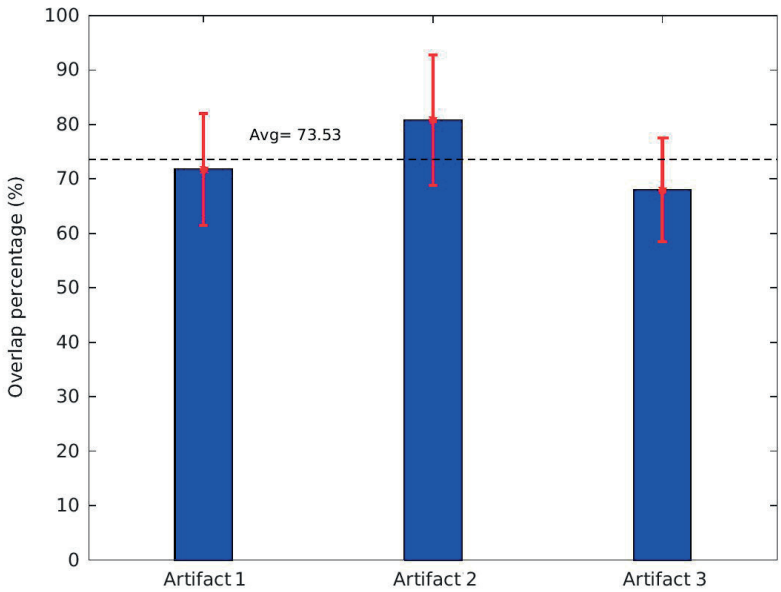


(b) Rotational error

Figure 6.10 – The chart on the left shows the translational error, and the chart on the right shows the rotational error when the robot builds an artifact using SLAM and SRL.



(a) SLAM



(b) SLAM and SRL

Figure 6.11 – The left chart depicts the overlap percentage for dropping blocks using SLAM, and the right chart depicts the overlap percentage for building the artifact based on use of SLAM and SRL.

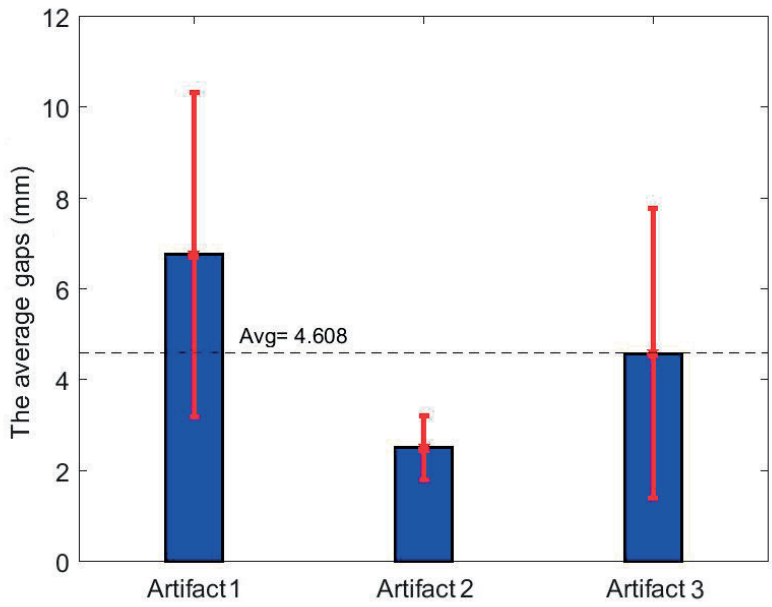


Figure 6.12 – The average gaps left between blocks of the artifacts.

great deal of progress in the past few decades in the SLAM field, high-precision applications are still a challenge. For the time being, SLAM could help to find approximate construction sites, and then, the robot could use other methods to improve construction, such as SRL, as was used in this research.

In this research, we applied SRL based on a pure IR-sensing system. Today, companies are designing and manufacturing prefabricated components to increase construction speed and efficiency. New prefabricated components could be designed and manufactured for robotic use in automated construction. For example, components with male–female connectors allow for automatic assembly in a more robust way [5]. Developing construction methods based on mechanical-relative localization or that use force-sensing systems could allow blocks to be placed in a more reliable and precise way.

Autonomous construction is a complex application in which many failures can occur. These failures can propagate from one step to another; for instance, if the robot incorrectly grasps a block, it could ruin the built structures. Thus, it is important to detect and correct faults. We performed fault detection in the `middle planner`, which is designed for construction commands (*pick up*, *drop*, *place-adjacent-to*). However, we could improve the builder planner, allowing it to take high-level decisions for failures created during construction.

6.4 Conclusion

This chapter has presented an autonomous construction system for building separated artifacts with simple blocks. We used a miniature mobile robot that autonomously mapped the environment using the SLAM algorithm and then manipulated blocks to build the desired and separated artifacts. Our approach was based on the combination of two approaches: a self-positioning system (SLAM) to find the construction site in an unknown environment, and SRL to build coherent artifacts. The control system allowed the robot to perceive and pick up a block, move towards the construction site in an unknown environment, and drop the block off based on a human-prescribed blueprint. We observed that the positioning using SLAM alone is not precise enough to build the artifacts. SRL could help the robot in the creation of coherent structures. The results and analysis in this chapter show that SRL can significantly reduce the gaps in the built walls by employing a short-relative sensing system.

The results of this chapter can be used to improve AFAC for the rescue purposes, which will be looked at more in Chapter 7. The protective walls must be as coherent as possible to protect the human victims from toxic sources, as one example. Thus, both SRL and SLAM are needed to minimize gaps in the walls. In addition, the `construction plan unit` computes an effective and accurate plan but real robots are unable to precisely drop blocks at requested locations. We need to modify the construction plan based on rotational and translational errors to compensate for the inaccuracies that come from a real-world setup.

7 Autonomous Construction

7.1 Introduction

In nature, there are several animals that employ different mechanisms to build adaptive and functional structures. The structures' shapes vary even within the same species. Animal architects often adapt building processes to local environmental conditions and try to achieve expected functionalities in their nest building. Termites build structures to satisfy required functionalities and to adapt to environmental conditions [124]. For instance, a ventilation system has to capture wind energy for exchanging gas and heat trapped inside the mound; their mound has to adapt to the topography of the environment and availabe material types [125].

In this thesis, we mainly address AFAC for rescue applications compared to a structure based on a blueprint. AFAC is a new method that can tackling current challenges in autonomous construction. The general idea of AFAC is to relax the constraints coming of blueprints and allow the robot to be smarter and more flexible regarding construction. In other words, the thoughts and expectations that can shape human construction can be formulated for autonomous robots, help them to perform autonomous construction.

A rescue operation is a good example of facing an unknown environment where it is impossible to plan in advance. This application is well suited to show how to formulate and implement this new construction method. Overall, rescue is an interesting application in the robotic field in which robots can be used as assistants for human rescue teams in hazardous and restricted environments. Isolating humans from the hazard is necessary because of potential harm and even death.

A nuclear accident, for example, might destroy city infrastructures, potentially hurt a number of people, and can contaminate an environment. The main goal of a search and rescue

operation is to find and isolate victims from hazards in the shortest amount of time and then to provide safe regions by isolating the toxic sources. Limiting the time spent near the source of radiation or shielding the victims from these sources by building protective walls are two possible ways to minimize the radiation exposure. One of the rescue services that mobile robots can contribute is autonomously building particular structures to reduce radiation exposure. However, the robots' vulnerability to radiation restricts its freedom to move into unsafe regions. Radioactive radiation might cause physical damage to the robot, especially to its semiconductors; and, long-term damage will eventually disable the robot completely while short-term exposure increases sensory noise for the robot. Because of these concerns, the associated objectives are defined to shape AFAC for rescue operations as follows: decreasing exposure on victims, decreasing exposure on rescuers, and decreasing exposure on robots.

AFAC is developed based on two different approaches: the local approach and the global approach. In the first approach, we employ robots to build protective walls inspired by the stimulus–response method used in nature. This is a powerful method that allows robots to respond very quickly to changes in an environment [126]. In this approach, robots do not require any internal representation or knowledge of the world; they act based on defined rules. Robots build protective walls in front of victims or radioactive sources based on sensed environmental features and then, within a given time, act immediately based on defined rules. We call this type of construction *reactive construction*.

In contrast to a local approach (reactive construction), we develop a new global approach to build structures. This global approach consists of three phases: the first phase is exploration, which was presented in Chapter 4. A robot explores the environment to map it and find important elements (e.g., victims). In the second phase, which was described in Chapter 5, the robot uses the collected data and world models to autonomously compute an effective construction plan. In the last phase, the robot constructs structures based on the computed construction plan. Because robots build the structures based on the construction plan, we call this approach of AFAC *plan-based construction*.

Both construction types for rescue, plan based and reactive, are implemented to develop AFAC and to demonstrate an achievable use of AFAC in new classes of construction. The results and analysis of extensive experiments in the simulation and on a real robot shows the properties and performance of the two approaches of AFAC. In summary, this chapter provides the following outcomes:

- We develop AFAC from scratch to a level at which the robots can autonomously build for real applications.
- We demonstrate autonomous construction can be used for the rescue operation.

- We show how efficiently the approaches secure the environment and the victims' situations.
- Rescuers can understand the trade-off between the gain and cost for the rescue operation if they employ robots to build protective walls.

In Sections 7.2 and 7.3, the plan-based and reactive construction systems are discussed respectively. The experimental setup was presented in Chapter 3. In Section 7.4, the experimental methodology and assumptions are presented for both approaches. The results and discussions are presented in Section 7.5. Finally, we conclude with Section 7.6.

7.2 Plan-based construction

One approach of AFAC for rescue is to autonomously compute a construction plan based on built maps and world models; then, the robot has a plan with which to build the protective walls. In this section, we describe how AFAC is designed and implemented for the plan-based approach. In Section 7.2.1, we provide an overview of the architecture of AFAC based on the global approach. GAs typically require significant hardware capacity and computation time, especially when there is a big environment with many radioactive sources and victims; thus, another method is designed to find a simple solution that requires less computations and time. A new method for computing a construction plan is introduced in Section 7.2.2. Section 7.2.3 introduces the method by which a construction plan is translated from segments to target points; the construction plan consists of segments but the robot act based on points.

7.2.1 Architecture

All the parts of the AFAC architecture are now combined while being based on the global approach used for rescue (Figure 7.1). As we have already described, plan-based construction (AFAC based on the global approach) consists of three phases: the first phase is exploration. The `exploration box`, which was described in Chapter 4, directs the robot to efficiently explore an unknown environment and concurrently find victims and radioactive hotspots. In Chapter 4, we presented the heuristic and MCDM exploration methods for mobile robots that use gamma cameras (with the long acquisition time and poor AR limitations) for discovering radioactive hotspots. However, the heuristic method is applied for this phase.[1] In the second phase, the `construction plan unit` provides the construction plan based on required inputs, such as the semantic world model and geometric map. GAs can be applied to provide

[1] We first implemented AFAC based on the heuristic exploration method, and then we designed MCDM to decrease the exploration time. That is why we used the heuristic method for this phase despite fact that the MCDM shows the better performances.

the construction plan, as was explained in Chapter 5. An alternative algorithm for computing construction plan will be explained in the next section. In the last step and after computing

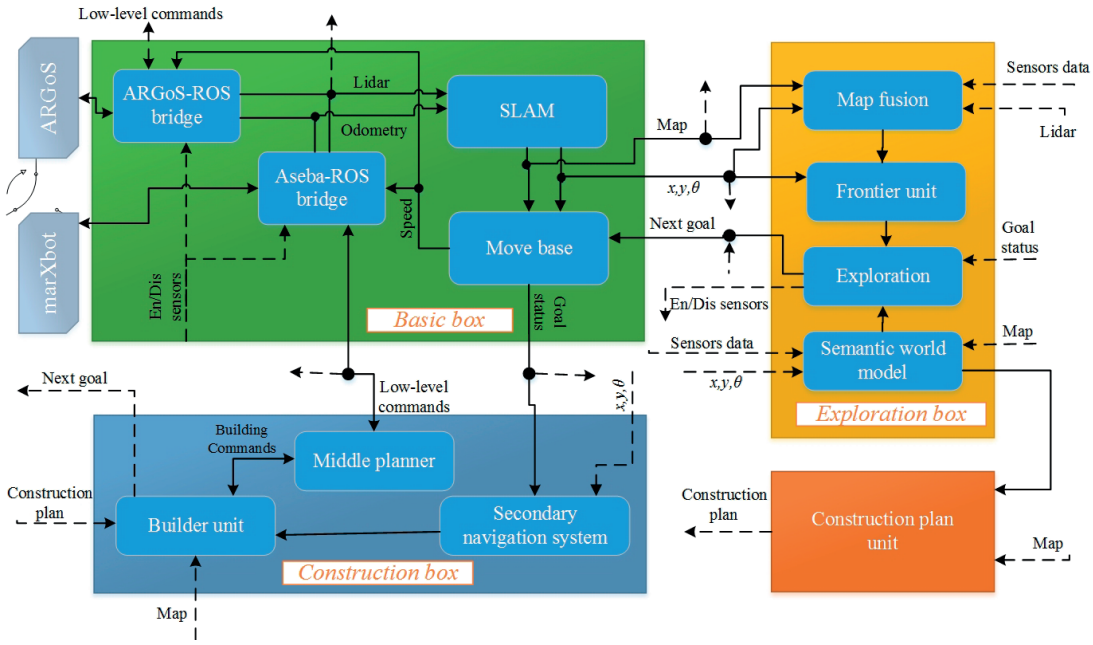


Figure 7.1 – AFAC architecture based on the plan-based approach for rescue.

a construction plan, the `construction box` directs robot to build the protective walls. In Chapter 6, we described how the construction box works on a predefined blueprint. Here, the `construction box` applies the plan computed by the `construction box` instead of the predefined blueprint provided by a human user; other aspects of the `construction box` stay the same.

The `basic box` includes common packages needed for other boxes. The navigation system, SLAM, and middle units are embedded in this box. The middle units (`ARGoS-ROS Bridge` and `Aseba-ROS Bridge`) are designed for exchanging data (e.g., messages, commands) between the robot (marXbot) or simulator (ARGoS) and other units. SLAM provides the robot locations. The `move base` unit is a navigation system used that directs the robot from its current location to a target point.

7.2.2 Second algorithm for computing a construction plan

In Chapter 5, GAs were used to provide and optimize a construction plan. One problem with GAs is that they typically require significant hardware capacity and computation time, especially in the case of a large environment with many radioactive sources and victims. Thus, we would have to design a simple algorithm for finding quick solutions using less

computations.

The *arc algorithm (ArcA)* is a new algorithm that works based on very simple principles. It renders a circle around each radioactive hotspot as a barrier, and then removes redundant parts that don't protect anything. Finally, it sorts the remaining arcs from the initial circles based on their importance (e.g., protected victims earlier). With utilizing this method, we can potentially protect victims and provide secure paths between victims and rescuers.

More specifically, ArcA draws an imaginary circle around each radioactive source. The length of the radius is an arbitrary variable, but it, of course, can depend on the intensity of the radiation and the existence of a possible victim near the radioactive source. In the next step, ArcA must remove the redundant parts. After identifying the shortest paths between the victims and safe regions (exit doors), we connect the radioactive sources to the victims and paths with imaginary lines. The parts of the circles that do not have any intersect these imaginary lines are removed. In the next step, the arcs of the circles that are overlapping each other are removed. An arc may be able to isolate more than one source at the same time or protect nothing, so ArcA removes the useless walls (or parts of walls). The next step is to convert the arcs into a series of segments. Finally, we sort the segments based on importance. For a better understanding, a sample solution made by ArcA as a schematic representation is shown in Figure 7.2.

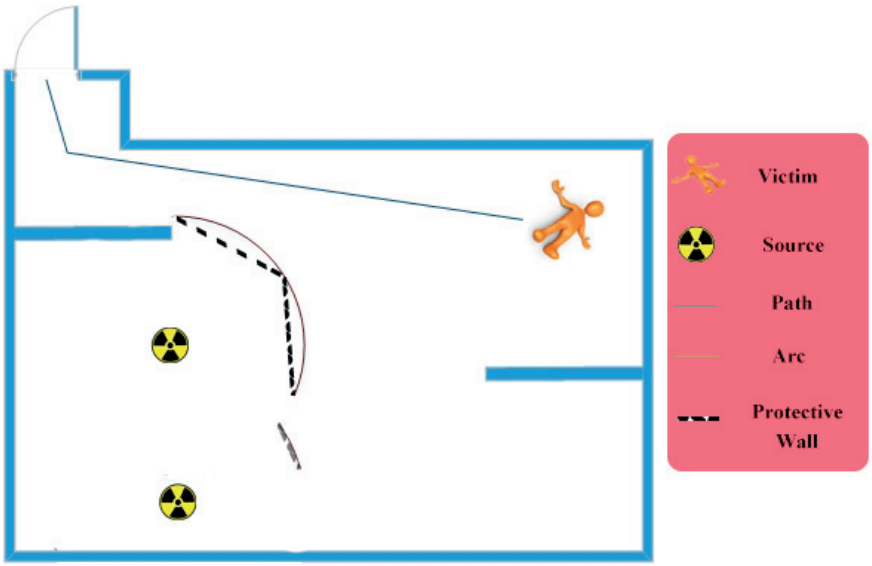


Figure 7.2 – A schematic representation of the walls developed by ArcA.

For sorting the segments, we look for those segments protected the victims earlier, or that

can decrease the cumulative exposure to victims more quickly. For this step, we employ the fitness function used by GAs in Chapter 5. We evaluate each segment of ArcA with the fitness function; and then, we sort the segments by their fitness values. All of these processes of ArcA are completed inside the `construction plan unit`.

To explain why ArcA needs less computation effort and time, we used the fitness function only for a few segments of ArcA in one step but GA exploits the fitness function for each individual of a population in each iteration.

7.2.3 Construction points production and modification

In the plan-based construction, the robot has to build protective walls based on the construction plan which consists of a set of segments but the robot places bricks based on points. Thus, we simply need to convert the segments into a series of points. Each segment that represents a protective wall has to be divided by the block size to some sections; the center of each section will be considered as a construction point. The segments of the construction plan are ranked in order of importance, so we need to keep the same order for points. The construction points will follow the expected goal if the points is assigned based on the same order of the segments.

Section 6.3 showed that real robots are unable to drop blocks precisely at requested locations. Thus, we need to modify the construction plan based on rotational and translational errors to compensate for the inaccuracies encountered in the real-world setting. According to the results presented in Section 6.3, the length of each segment must increase (add construction points) from both sides to at least half of the block size (3 cm) to compensate for these inaccuracies of the real-world.

Furthermore, these errors cause that wall directions diverge or displace walls; which they cause the walls are not exactly built on the desired segments. As a result of deviations from the desired plan (e.g., detour on a wall direction), the robot again needs to investigate possible collisions between a new block and obstacles. The robot must also investigate the construction site in advance in order to find a suitable place for itself to drop the block. Figure 7.3 shows a real robot that is building protective walls. The robot builds protective walls on segments which are shown by green lines. The question is where the robot must go for the next block to avoid collision with obstacles, as it needs enough space around the construction point to drop the new block. We define an orange polygon depicted in Figures 7.3 and 7.5; it shows a contour of the robot, in which we assume the robot has levered its gripper to drop a block. This polygon is used to investigate the robot's next position around the target position in order to find collision-free places. More specifically, we first consider an inflation map, which is a map based on the geometric map but with a safe margin around occupied cells (Figure 7.4). Then eight states of the polygon which are shown in Figure 7.5, are overlaid on the inflation

Figure 7.3 – The real robot is building protective walls to provide safe regions for a victim and rescuers; a victim and two radioactive sources are represented by the red color and yellow color, respectively. Green line show the desired protective wall made by the `construction plan unit`. The orange polygon shows a possible next location for the robot to move.

map around the target point. The states of the polygons that have a collision are removed, and the remaining state closest to the current position of the robot is selected.

Figure 7.4 – An inflation map of the environment represented in Figure 7.3.

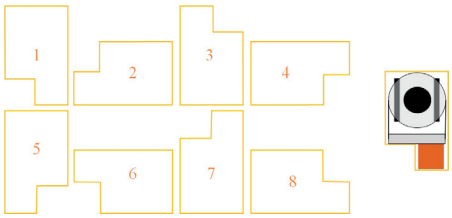


Figure 7.5 – The robot needs to check a region around the target point to find possible collision-free places. This figure shows eight states of the polygon (the contour of the robot with a grasped block) that are used to check the region around the target point. For example, the orange polygon which is shown in Figure 7.3 is state two.

7.3 Reactive construction

In contrast to plan-based construction, which is based on a global approach, we employ robots to build protective walls based on a local approach. In this approach, robots do not require any internal representation or knowledge of the world; they act based on some defined rules. Creating and updating a global representation is a time-consuming task; so by removing all dependencies that are related to global representations, AFAC based on the local approach is intrinsically fast and react quickly in environmental changes. These competencies (e.g., fast response to environmental changes) are very useful in highly dynamic and unpredictable situations such as in the rescue operation [127]. We already called AFAC based on a the local approach reactive construction. Thanks to local approach capabilities, such as a fast response to changes, the robots can be employed to perform reactive construction for rescue operations.

Moreover, the robot will not be able to use a gamma camera in reactive construction; the long acquisition time of the gamma camera is a big challenge that avoids the robot to have a fast response. Therefore, we employ the robot with an array of non-directional radioactive sensors[2] around it instead of using the gamma camera.

7.3.1 Architecture

In this section, we describe the architecture of a system designed for reactive construction. As described before, reactive construction does not require any world representation; instead, the robots locally sense environmental features and then act immediately according to defined rules. Figure 7.6 presents the architecture of reactive construction, which includes several packages.

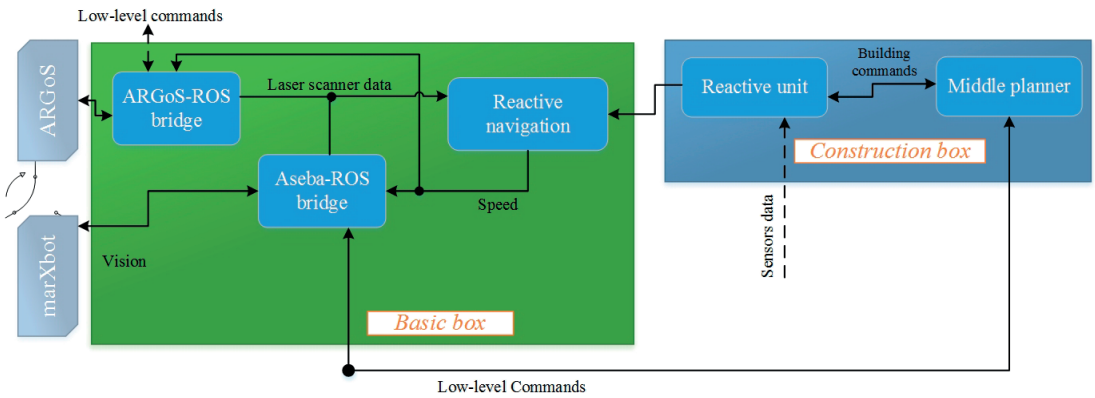


Figure 7.6 – AFAC architecture based on the reactive approach used for rescue.

Although there is no particular package for mapping, the robot must come back to depot site to pick up a new block. The `reactive navigation` package is based on a reactive navigator that allows the robot to move back to the depot site without using a global map. It was designed in the ROS[3] for wheeled mobile robots. The basic idea of this navigator is to transform a robot of any shape and kinematic restriction into a new space [trajectory parameters space (TP-Space)] where well-known obstacle avoidance methods are applicable. In other words, the TP-Space transformation is defined in such a way that the robot can be considered a free-flying point, but its shape and kinematic restrictions are already embedded into the transformation process. Moreover, TP-Space provides a space for existing obstacle avoidance methods to perform better detection of obstacles and avoid them [128].

The rules and construction behaviours are defined within the reactive unit. The reactive unit

[2]As presented in Section 4.2.2, non-directional radioactive sensors are almost fast and they are used for applications such the monitoring.

[3]The `mrpt_reactivenav2d` package in ROS

is responsible for performing the construction processes; it sends high-level commands to other packages, such as the *middle planner*, and changes the task based on the feedback. The *middle planner* is an intermediate package between the high-level construction commands and low-level subtasks. It translates high-level commands into a set of low-level subtasks for block manipulation. The `basic box` also includes the middle units (`ARGoS-ROS Bridge` and `Aseba-ROS Bridge`) that are designed for exchanging data (e.g., messages, commands) between the robot (marXbot) or simulator (ARGoS) and other units.

7.3.2 Behaviours

In this section, we present the behaviours that are used inside the *reactive unit*. As shown in Figure 7.7, the reactive package consists of both deliberative and purely reactive low-level tasks.

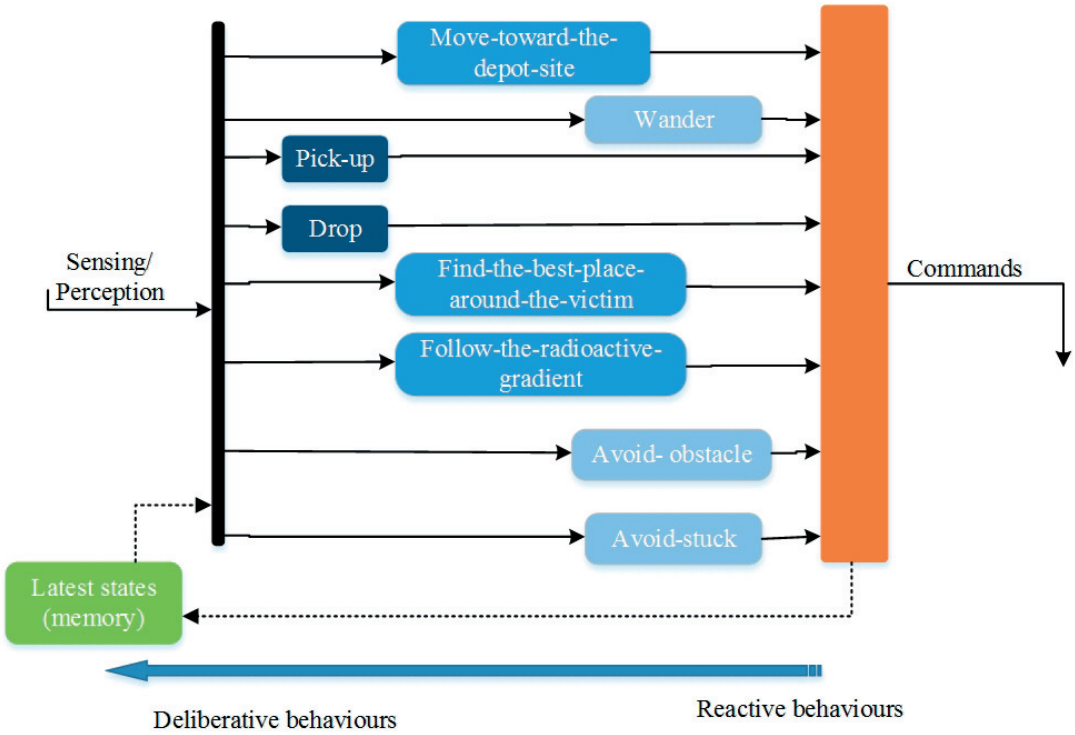


Figure 7.7 – A schematic of the control system used in the `reactive unit`.

Designed reactive construction is generally reactive; although, it consists of both deliberative and reactive low-level behaviours (Figure 7.7). The behaviours will be triggered if the perception and latest states meet the conditions specified inside them, even if they are deliberative behaviours. For instance, the *pick-up* is a behaviour that needs a planner[4] to lead the subtasks;

[4]The middle planner

but the robot will triggered this behaviour when the perception data and states reach the requirements defined inside it.

The role of the *latest states* (Figure 7.7) is to retain the latest values of a few parameters for the sake of behaviour activation. The execution of some behaviours depends on previous behaviours that have already been performed by the robot. For instance: the wander needs to know whether or not a block was grasped; the *find-the-best-place-around-the-victim* will be executed if the *latest states* affirms that a block has been grasped and a victim has been found.

We define the behaviours below.

7.3.2.1 Wander

This behaviour generates random values for linear and rotational velocities of the robot. It causes the robot to move randomly. We also define three rules within this behaviour as follows:

1. If the robot sees any victim, it updates the memory (latest states) of the construction system by announcing that a possible victim has been detected.
2. If the robot detects a radioactive dose of more than a user-defined threshold, it updates the memory of the construction system by announcing that radioactive intensity has reached the first threshold.
3. The most important goal of rescue is to find and save the victim, so the robot randomly ignores the second rule and does not care about radioactive dose. This rule will increase the chance of the victim detection.

Algorithm 5 shows the pseudo-code of the *wander* behaviour. First, it gets the latest states in order to know what to do. Then, two random values are determined for linear and rotational velocities. The robot has to move for a certain amount of time (`excution_time`) according to the specified velocities' values. According to the first and second rules, if the robot sees any victim or detects a radioactive dose more than a first user-defined threshold, `states.victim` and `states.exposure` of the memory will change to `Yes` and `Medium` respectively. By changing one of these state parameters, another behaviour will be activated.

In an environment with a strong radioactive source that is close to the depot point, for instance, the robot will always detect the high radioactive dose, so this keeps the robot from detecting victims or at least the chance of victim detection decreases. To solve radioactive trap problems, we apply the third rule. The `states.victim_priority` is provided by a number that is either 0 or 1; if it is equal to one, it means that finding the victim is the first priority. In this case, the robot ignores radioactive exposure while sensing and looking for a victim. If the robot cannot

Algorithm 5 Pseudo-code shows the wander behaviour

```
function WANDER()
    states ← GET-STATES()                    ▷ The set of states includes block, victim, ...
    if (states.block == grasped) and (state.victim == No) and (state.exposure == Low) then
    ▷ The function-randomizer(type,l,h)- determines a random value (float, int,...) between l and h.
        linear_velocity ← RANDOMIZER(float, −0.10, +0.10)                ▷ m/s
        rotational_velocity ← RANDOMIZER(float, −0.12, +0.12)            ▷ rad/s
        excution_time ← RANDOMIZER(float, 10, 40)    ▷ A robot moves over this time
        if (states.victim_priority == 0) then  ▷ The states.victim_priority is given by a random number (0 or 1)
            exposure_value ←CHECK-RADIOACTIVE-EXPOSURE()
            if (exposure_value) > first_threshold then
                states.exposure ← Medium
            else if (exposure_value) < first_threshold then
                states.exposure ← Low
    if FIND-VICTIM() then
        states.victim ← Yes
    else
        states.victim ← No
    if states.victim_priority == 1 then
        victim_counter ← victim_counter + 1
    if victim_counter > 20 then
        victim_counter ← 0
        states.victim_priority ← 0
    UPDATE-LATEST-STATE(states)
    return
```

find any victims after a certain amount of time, the `states.victim_priority` will switch to zero. If `states.victim_priority` is equal to zero, then both cases (finding a victim and detecting considerable radioactive dose) have the same priority.

7.3.2.2 Move-toward-the-depot-site

The robot needs to come back to the depot site to pick up a new block. This behaviour drives the robot based on the paths produced by the reactive navigation unit[5]. The absence of any world representation in reactive construction means not having the robot's position in the global coordination system. However, the existence of an attractor at the depot site or a chain of landmarks, can help the robot find the depot site.

We solve this problem by putting an emitter on the depot site that sends signals to robots. we

[5]See Section 7.3.1.

put an emitter on the depot site. By means of the emitter (or chain of emitters) the robot will know its position in respect to the main emitter placed at the depot site. Both the emitter and robot are equipped with a range and bearing system. As shown in Figure 7.8, the emitter sends a message, including the IDs, range, and angle of a observed robot with respect to itself via the range and bearing system. The observed robot matches the ID of the received message with its internal ID; if the signals are the same, the robot computes its position in the emitter's coordination system (${}^{E}X$ and ${}^{E}Y$ axes).

For example, the position of a robot with an ID equal to 2 will be as follows:

$$\begin{aligned} {}^{E}x_{robot_2} &= d_1 \times cos(\theta_{01}) \\ {}^{E}y_{robot_2} &= d_1 \times sin(\theta_{01}) \\ \theta_{robot_2} &= f(\theta_{20}, \theta_{01}) = 180 + (\theta_{20} - \theta_{01}) \end{aligned} \tag{7.1}$$

Where

$\theta_{20}, \theta_{02} < 0$ and $\theta_{01} > 0$

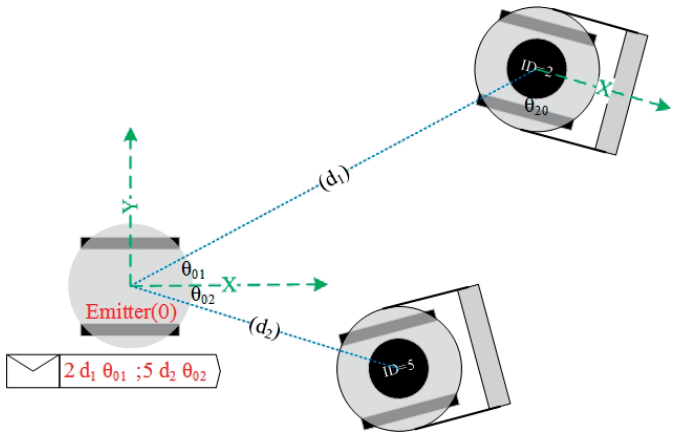


Figure 7.8 – A schematic of the range and bearing system used for finding the depot site.

Finally, the robot can move back to the depot site by using the *move-toward-the-depot-site* behaviour that acts based on the reactive navigator explained earlier and the position of the robot established by the help of the range and bearing system.

One issue still remains: the robot and emitter have to see each other in a sight line[6] when they use the range and bearing system. Otherwise, the obstacles reflect the infrared waves of the range and bearing system, so a false emitter will appear somewhere else in the direction of the last reflected wave. This is not an accurate positioning system, the robot can follow

[6]Sight line is an unobstructed line between an observer and an object.

the infrared waves to reach the main source (or depot site). However, for complex and large environments, a chain of emitters is necessary.

7.3.2.3 Find-the-best-place-around-the-victim

Finding and protecting the victims are the main concerns of a rescue operation. When a robot successfully finds a victim, the question is where the block should be placed to efficiently protect the victim. For instance, the best place could be somewhere on the straight line between the source and victim. This behaviour allows the robot to place a block in a suitable place where the block can decrease the victim's radioactive exposure as much as possible. However, a sequential process is needed to perform this behaviour, as shown in Figure 7.9: 1) The robot rotates and then moves towards the victim. It stops when it reaches a location that is close to the victim. 2) The robot rotates 90 degrees in place either left or right and then revolves around the victim until the robot meets an obstacle or the initial place again. 3) The robot returns to the location where it detected the highest radioactive exposure. If the radiation sensors only recorded the background dose, the robot will return to the place where it started its revolution. In these movements, the robot benefits from the odometry system in finding its relative location. In addition, the laser range scanner helps to estimate the distance between the robot and the detected victim.

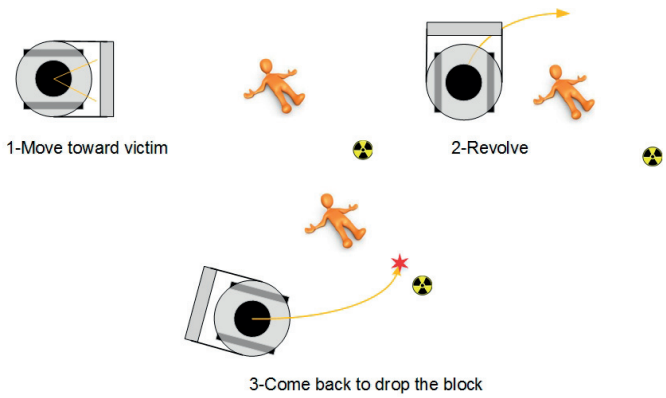


Figure 7.9 – The sequence of movements to find the best place around the victim.

7.3.2.4 Follow-the-radioactive-gradient

One goal of this rescue operation is to secure the environments for rescuers' activities. If the robot drops blocks closer to radioactive sources, it protects more regions. This behaviour directs the robot towards radioactive sources by tracking the radioactive gradients. If the robot detects a radioactive dose more than the first user-defined threshold in the *wander* behaviour, it will start to follow the highest radioactive dose. `states.exposure` of the memory will

change to `High`, if the robot senses a radioactive dose higher than a second user-defined threshold.

7.3.2.5 Avoid-obstacle

The robot must be able to avoid obstacles. This section explains the collision avoidance behaviour that allows the robot to avoid a collisions with any number of obstacles in the environment. It employs the 24 proximity sensors around the robot to detect obstacles in any direction. We implement the obstacle avoidance based on potential fields. We normalize the value of each proximity sensor to the common scale [0,1] as a repulsive force; then, we split the resultant force vectors into X and Y directions of the robot's coordination system. Finally, the right and left speeds of the robot are changed based on the combined corresponding components of repulsive forces in order to avoid collisions with obstacles. Algorithm 6 shows what was implemented in the marXbot for obstacle avoidance behaviour using potential fields.

Algorithm 6 The algorithm of obstacle avoidance that was implemented in the marXbot using potential fields

function AVOID OBSTACLE()
  $proximity \leftarrow$ **GET-PROXIMITY**() ▷ Read the proximity data (intensity)
  for $(i = 0\ to\ 23)$ **do**
    $angle \leftarrow 2 \times \pi - \frac{2 \times \pi \times i}{24} + \frac{\pi}{48}$
    $bufferX \leftarrow bufferX + \frac{proximity[i] - low\_intensity}{high\_intensity - low\_intensity} \times cos(angle)$
    $bufferY \leftarrow bufferY + \frac{proximity[i] - low\_intensity}{high\_intensity - low\_intensity} \times sin(angle)$
    $left\_speed\_collision \leftarrow 0$
    $right\_speed\_collision \leftarrow 0$
    if $(bufferX > 1)$ or $(bufferY > 1)$ or $(bufferX \times bufferY > 0.5)$ **then**
      $left\_speed\_collision \leftarrow \frac{(bufferX + bufferY) \times max\_linear\_speed \times -1}{6}$
      $right\_speed\_collision \leftarrow \frac{(bufferX + bufferY) \times max\_linear\_speed \times -1}{6}$
  return 0

.
.
.
▷ Commands below are executed in the main body of the `reactive unit`
$left\_speed \leftarrow left\_speed + left\_speed\_collision$
$right\_speed \leftarrow right\_speed + right\_speed\_collision$

7.3.2.6 Avoid-stuck

In this behaviour, the robot must first recognize a potentially trapping situation and then avoid getting stuck in the trap. Hence, this behaviour consists of two parts: recognizing the potential to be stuck and avoiding it. The first component works based on the investigation of sensory data, such as wheel sensors that can help it understand whether or not the robot should stop. If the robot guesses it might be trapped, then the second part randomly moves the robot. The robot repeats the random movements until it escapes from the trap.

7.3.2.7 Pick-up

The pick-up behaviour directs the robot to grasp a block and then raise the gripper. See Section 6.2.4.1 to read about this.

7.3.2.8 Drop

This behaviour is designed to drop a block, which is similar to the drop task discussed in Section 6.2.4.3 (but only steps 6 and 7 will be performed). For reactive construction, an additional subtask is required. Before the robot drops a block, it has to check for possible collisions with other blocks or walls. The robot needs a free space in front of itself to be able lever the gripper down.[7] In the case of an occupied space, the robot must move back to create the space required and then drop the block. If it cannot find sufficient space after moving back a certain distance, the robot will ignore this behaviour and switch to another behaviour such as *wander*.

7.3.3 Robotic activities for construction

In this section, we briefly describe how the robot performs reactive construction to build in a rescue application. First, the robot has to come back to the depot site to pick up a new block using the `reactive navigation` package and the *move-toward-the-depot-site* behaviour. When it reaches the depot site, the construction box make a request to run the *pick-up* behaviour from the `middle planner`. The `middle planner` activates a sequence of subtasks to pick up the block. If the robot successfully grasps the block, the `middle planner` sends a success feedback to the `reactive unit`. Afterwards, the robot wanders. If the robot finds a victim or reaches the place where radioactive exposure exceeds the first certain threshold, it will start to follow the radioactive dose gradient. If it senses a radioactive dose higher than the second threshold, it will stop to drop the block and then immediately return to the depot site for a new block.

[7] The robot uses the laser range scanner to investigate the front space.

7.4 Experimental methodology and assumptions

We test AFAC for rescue applications in simulation and a real-world setup. The environment is similar to an office space, but the victims and sources are not at real scales. Figure 3.16 presents the environment used for simulation and physical experiments.

As explained in Chapter 3, the marXbot and ARGoS were used for the real experiments and simulation respectively. In simulation, the sources are lights and victims are virtual models. In the real-world setup, the sources are other marXbots and their range and bearing systems are used as radioactive hotspots. The front camera of the marXbot is used to detect the victims, distinguished by the color red in the real-world setup. The front camera is not simulated in simulation, so the victims are detected as long as the coverage area of the imaginary sensor sweeps through the positions of the victims. All sources emit radiation with the same intensity and rate, so an object which is close to a source will absorb $0.01\frac{Gy}{s}$ radiation. The customized gamma camera used for hotspot detection in plan-based construction. In reactive construction, an array of non-directional radioactive sensors around the robot are used to find the radioactive intensity and direction. In the real-world setup, the range and bearing system is used as non-directional radioactive sensors; in simulation, the non-directional radioactive sensors are the light sensors. The parameters of Table 3.2 show specifications of the customized camera. The parameters of the victim camera and the laser range scanner parameters are also in Table 3.2.

The following configurations are considered in studying the performance of AFAC. The environment's shape is the same for all experiments. The number of victims is one or two, the number of sources is one or two, and the environment is neat or cluttered. Therefore, there are eight different configurations. For each configuration, we generate 15 environments in which the source(s) and victim(s) are randomly placed. Hence, 120 different trials are performed for each approach: plan-based construction using GAs, plan-based construction using ArcA, and reactive construction. In total, 360 experiments are totally performed in simulation.

In the plan-based construction using GAs, the fitness function (Equation 5.3) of GA consists of three terms: victim, rescuer, and robot terms, along with their corresponding weights. We show the values of the weights in Table 7.1. The number of iterations is 1000 and other GA settings are the same as what we used in Chapter 5 such as Table 5.1.

Table 7.1 – The values are assumed for the weights of the fitness function

w_{victim}	w_{robot}	$w_{rescuer}$
0.35	0.3	0.3

In the plan-based construction using ArcA, we considered two values for the radius of the

imaginary circle around each radioactive source. If the minimum distance between a source and the victim to the source is less then 1 m, the radius of the circle will be this minimum distance; otherwise, the radius value is a constant value and is equal to one meter.

In the reactive construction, we need to define the first and second threshold that were used in *wander* and follow-the-radioactive-gradient behaviours. In this construction system, if the robot reaches a location where radioactive exposure from the second threshold, it will stop to drop a block. The threshold value is equal to 0.0025 $\frac{Gy}{s}$[8]. The first threshold is also equal to 0.0005 $\frac{Gy}{s}$.

Experiments on the real robot were performed for eight configurations. We choose only one trial for each configuration to validate the experiments performed in the simulation. For the real experiments, we decide to run eight trials for plan-based construction using GAs and eight trials for reactive construction.

The robot always starts the mission at the same point, considered (0,0) on the global map. The depot site is also a small region on this point. The exit door (the green point in Figure 3.16) is placed behind the start point.

In some practical experiments, there are hardware and software failures: the robot losses connection with the Wi-Fi network; the motor for tilt rotation of the gripper fails; the grasped block falls; or other errors occur. In these cases, the robot is restarted based on latest state and its location. In other instances, the map or robot's location may jump to somewhere else in SLAM because of the laser range scanner's inaccuracies or insufficient landmarks; so we would manually reset the SLAM system and consider the previous position of the robot as a new position.

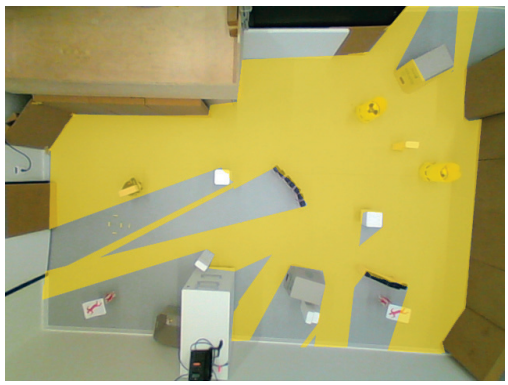
7.5 Results and discussions

Figure 7.10 presents scenarios with neat and cluttered environments after plan-based and reactive construction. The variations in performance is obvious. In the plan-based construction approach, the robot built two protective walls based on the plan. In contrast, in the reactive approach, the robot placed blocks without an overview of the environment, so the final arrangement of blocks was not as organized as in the plan-based construction.

In Figure 7.10d, there are two victims, but the robot built protective walls of a different density of blocks around the victims. There are five blocks for the bottom-left victim and three blocks for the bottom-right victim. It shows that the locations of victims and radioactive sources are important factors in reactive construction for the rescue operation, meaning some victims

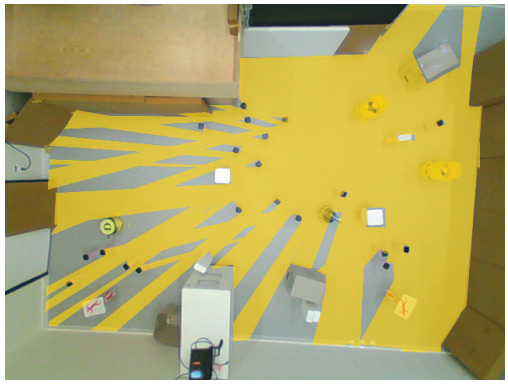
[8] It is radioactive intensity on a place where it is 1 meter far from a radioactive source.

(a) Plan-based construction in the neat environment



(b) Plan-based construction in the cluttered environment

(c) Reactive construction in the neat environment



(d) Reactive construction in the cluttered environment

Figure 7.10 – The robot has completed the building of protective barriers; two sources (yellow objects) and two victims (red objects) were randomly placed in each environment. The yellow shading shows the contaminated areas.

may not be completely covered or the density of the blocks are different[9] The building based on the reactive approach is sensitive to the configuration of the elements and the shape of the environment.

In the following sections, we study in detail the different aspects of AFAC used for the rescue operation.

7.5.1 Time of completion

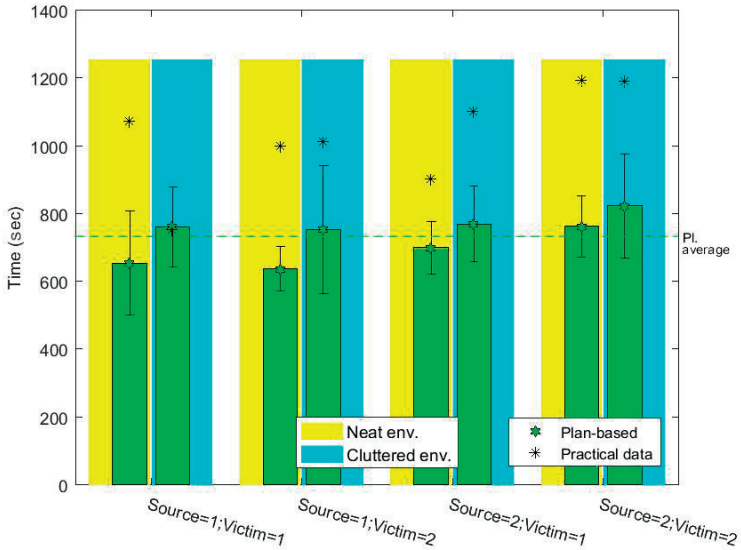
One of the biggest issues in a rescue operation is finding and saving victims within the first few hours (called golden time) after a disaster. Thus we study the completion time of plan-based and reactive construction. As mentioned, plan-based construction consists of three phases: exploring the post-disaster environment to map and find elements, the time to compute the construction plan,, and building protective walls based on the construction plan. Therefore, the time of completion includes the exploration time, computing the construction plan time, and the time required for building. The bars in the exploration graphs (and other graphs in this section) were made based on simulation experiments; the start points, labelled as practical data, show data of the corresponding experiments that were performed in the real-world setup.

The first phase is exploration, which was presented in Chapter 4. A robot explores the environment to map it and find important elements (e.g., victim). As described above, the heuristic method was used for this phase. Figure 7.11a shows the exploration time. The exploration time slightly increases when the number of sources changes from one to two. According to Figure 7.12, which shows the number of stops required for the gamma camera to obtain a radioactive image, the robot stops more often where the density of the sources is higher. In other words, these graphs show that exploration time is more sensitive to the number of radioactive sources than to the number of victims.

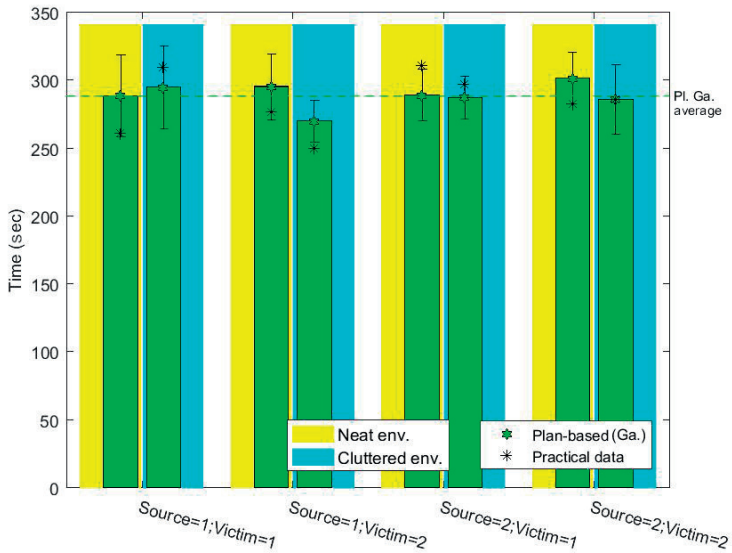
The second phase of AFAC is to compute the construction plan. In Chapter 5, we presented the construction plan unit, whose role was to produce a construction plan based on GAs. The parallel computing capability of GAs allowed us to distribute the computation efforts between many processors. Hence, we divided the computation efforts between all the cores of a CPU. The average time reached from about 800 sec to 290 sec when there were five cores instead of one core.[10] The computation time could be drastically decreased if we produced construction plans using parallel computers, especially where the environment is large and complex. Decreasing the resolution of the encoded map is another way to produce a

[9] The `states.victim_priority` is considered in Algorithm 5 to reduce the negative effect of this aspect of reactive construction for the victims' protection.

[10] Intel core i7-4700 2.4 GHz was used.

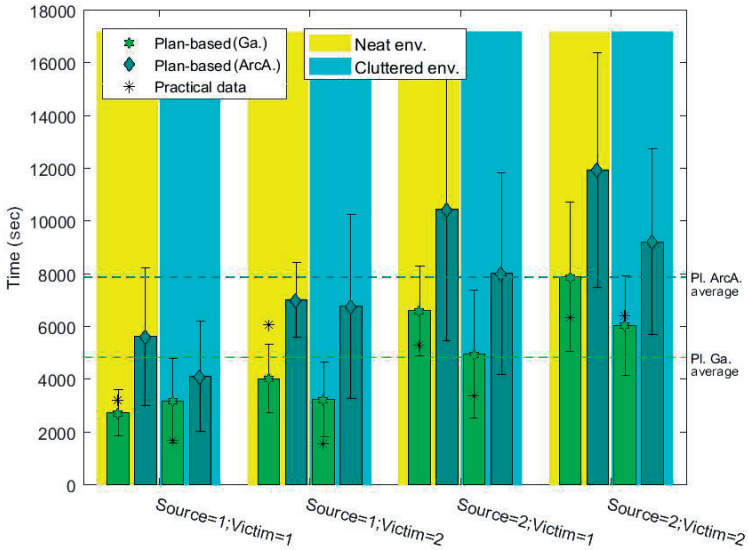


(a) The exploration time based on the heuristic method



(b) The computing construction plan time based on GAs

Figure 7.11 – The graphs show the components of the total time required for plan-based construction, which consists of three phases: exploring the post-disaster environment to map and find elements, computing a construction plan, and building protective walls based on the construction plan.



(c) The building time

Figure 7.11 – The graphs show the components of the total time required for plan-based construction, which consists of three phases: exploring the post-disaster environment to map and find elements, computing a construction plan, and building protective walls based on the construction plan (cont.).

construction plan based on GAs in a shorter time.

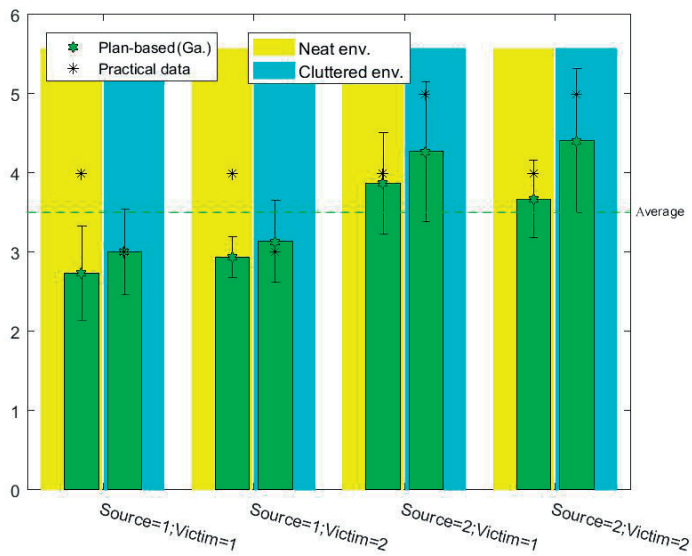


Figure 7.12 – The number of stops required for the gamma camera to obtain radioactive data using the heuristic method.

In Section 7.2.2, we introduced ArcA, which is a simple and quick way to compute the con-

struction plan. The time for computing a plan based on ArcA is negligible. However, Figure 7.11c illustrates that the protective walls building based on ArcA took more time than it did by using GAs. In addition, by looking at Figure 7.13, we see that plan-based construction based on GAs required less time to complete the rescue operation.

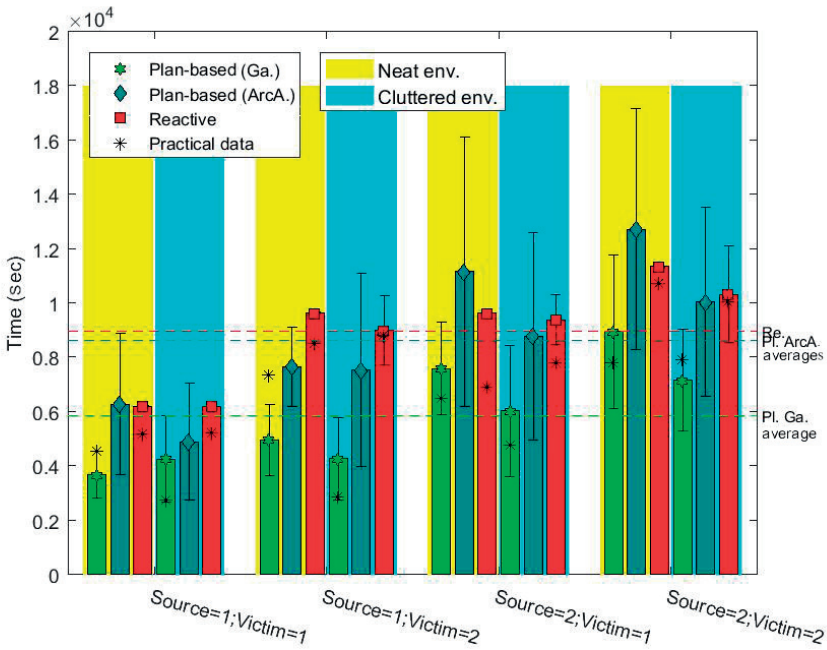


Figure 7.13 – Comparison of the total times for three different approaches applied in construction.

Because we have not yet evaluated other aspects of the different construction methods, the total time will not provide any interesting assessment of these three methods. To answer the question regarding which approach performs better in the rescue operation, we need to evaluate other important objectives, such as the victim safety. However, for a fair comparison between these methods, we must consider an equal elapsed time for all the methods. For instance, we can make a judgement the cumulative exposure on victim if it is measured within the same time. Thus, the minimum time between the three total times is considered the same for evaluating and comparing the methods. We call this minimum time *the least time*. In other words, we consider construction in three methods until they reach *the least time*. Below, we present our results based on *the least time* instead of the total time to provide a fair evaluations of the methods.

The graphs that are illustrated in Figure 7.14 show the elapsed time per consumed blocks ($\frac{time}{blocks}$ (within *the least time*)). The point about these graphs is that the construction speed decreases if the environment changes in terms of tidiness. In the cluttered environments, the

robot needs more time to navigate or to find victims.

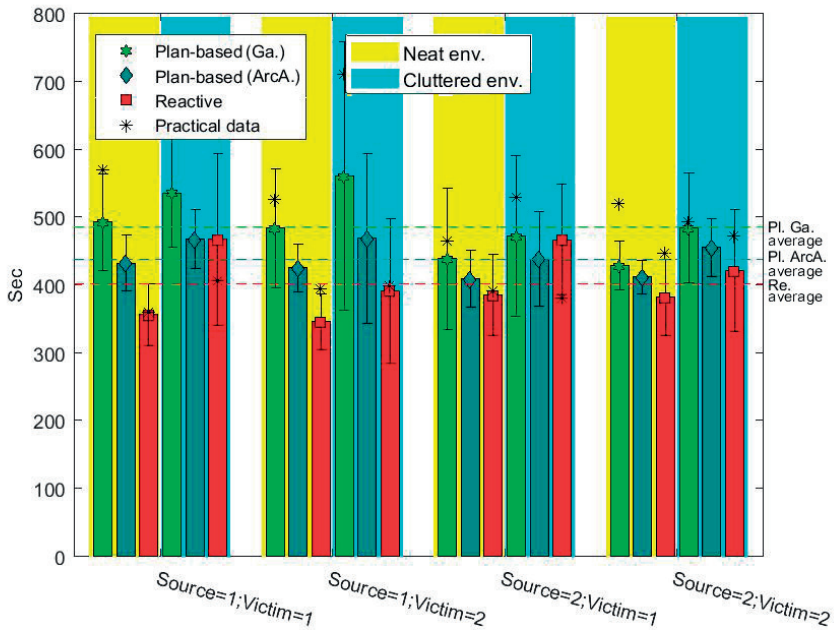


Figure 7.14 – The average amount of time is required to drop a block in the three methods.

Our results also show that, on average, 12.5 percent of *the least time* was spent exploring the environments; 5 percent of *the least time* was needed to compute the construction plan based on GAs, and the rest of *the least time* was used for building. This amount of time is an important challenge for plan-based construction, especially when the post-disaster area is large and complex. Improving the exploration methods or finding more effective methods for computing the construction plan can reduce the initial time required and improve AFAC performance. For instance, Section 4.6.1 showed that the MCDM method can complete the exploration earlier and miss much fewer radioactive hotspots in respect to the heuristic method.

7.5.2 Victim exposure

The purposes of AFAC are to protect the victim, robot, and rescuer in the rescue operation. In this section, we study how AFAC could satisfy the victim safety objective and which approach worked better. Figures 7.15 and 7.16 present victim exposure over time.[11] Interestingly, victim exposure immediately started to drop in the reactive construction; although, it never reached the zero value at the end. In the other approach, the robot first spent a certain amount of time exploring and computing the construction plan; afterwards, the robot started to isolate the

[11] To avoid repetition, we did not use other graphs; these graphs represent similar behaviours.

victims. However, the plan-based lines have steeper slopes than the reactive construction, so they did reach the zero value.

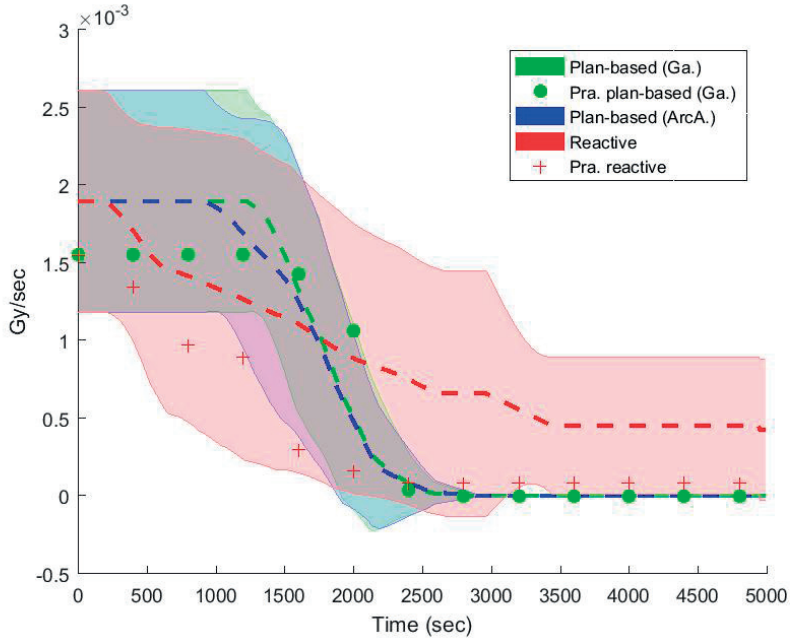


Figure 7.15 – Victim exposure over time: one victim, one source, and a neat environment. The lines and shaded areas around the lines show the mean and standard deviation, respectively. The practical data are indicated with pluses and circles.

Total cumulative exposure of the victims (Figure 7.18) is the integral of victims' exposure over time; it is calculated until the end of *the least time*. The cumulative exposure versus environmental complexity (increasing the victim or radioactive sources) has the same pattern at the beginning as the victim exposure (Figure 7.17). In the state *source=1; victim=1*, the cumulative exposure of the victims in the cluttered environment is higher than the uncluttered one; in contrast, in the other states, the cumulative exposure in the neat environment is higher. In addition, on average, the difference between cumulative exposure in neat and cluttered environments for the state *source=1; victim=1* is minor, while the difference for the other states is significant. The mentioned points about cumulative exposure are true for victim exposure in the beginning as Figure 7.17 illustrates. As a result, we can conclude that victim exposure in the beginning has an important role in the cumulative exposure of the victim.

Figure 7.19 presents the remaining victim exposure at the end of *the least time*. The value for the remaining victim exposure for reactive construction could not reach the zero value within a given time. The haphazardly placed blocks obstructed the robot's path so that the robot could barely detect or approach the victims; radioactive waves passed through the gaps between the blocks. In the plan-based construction using ArcA, the victims were also not completely

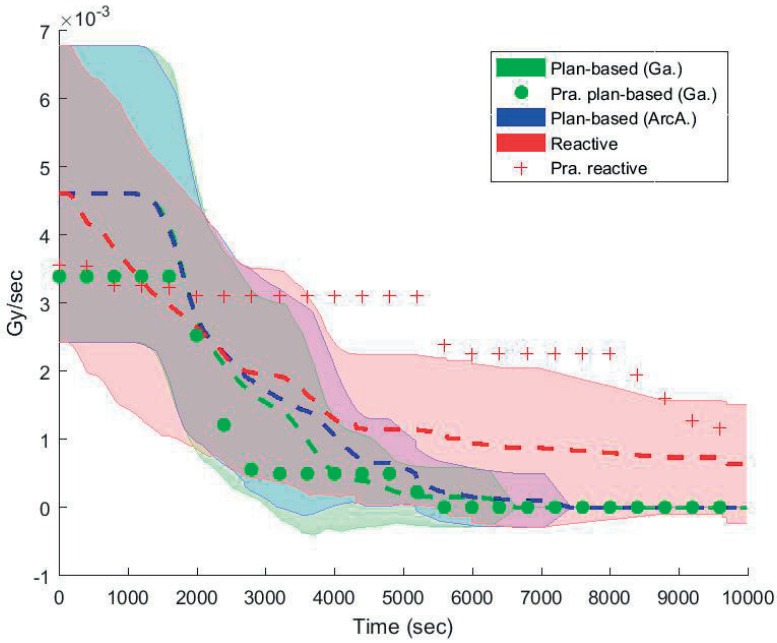


Figure 7.16 – Victim exposure over time: two victims, two sources, and cluttered environment. The lines and shaded areas around the lines show the mean and standard deviation, respectively. The practical data are indicated with pluses and circles.

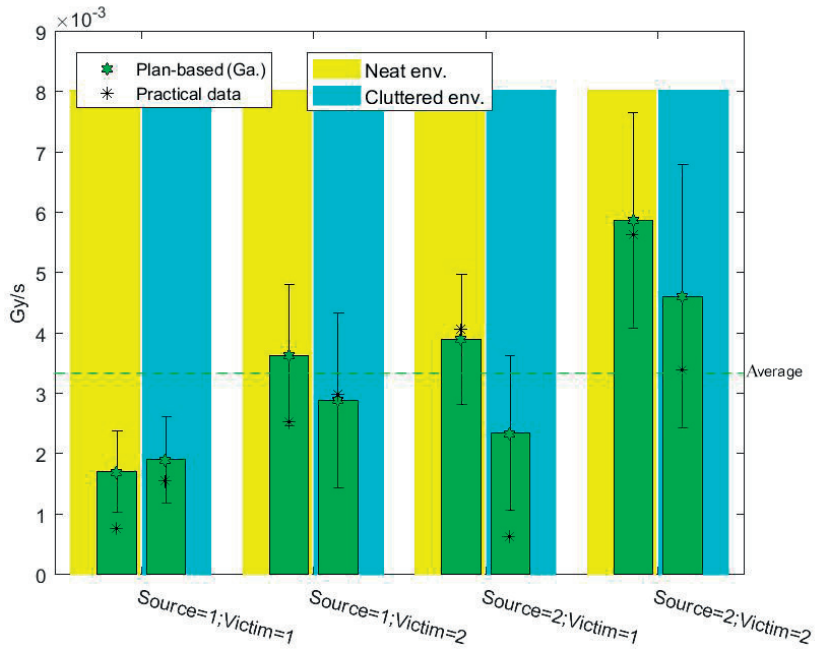


Figure 7.17 – Victim exposure in the beginning.

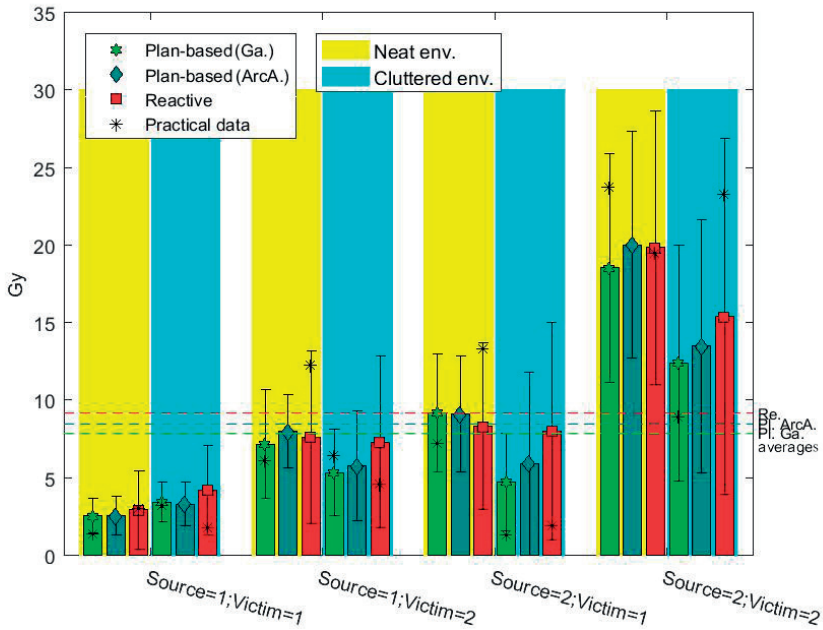


Figure 7.18 – Total cumulative exposure of the victims.

protected. There were some small gaps that appeared when the arcs were converted into linear segments. However, if the victims are not immediately retrieved, the remaining victim exposure will impose the worse conditions on victims and increase the cumulative exposure they absorbed.

7.5.3 Rescuer safety

Another objective of the rescue operation is to provide safe paths or regions for rescuers to retrieve victims or treat them on-site. In this section, the shortest paths between victims and exit doors are determined. The results show that what portion of paths have been secured in performing the construction tasks. We define this metric to evaluate the rescuer objective as follows:

$$Path\ ratio = \frac{length\ of\ secured\ paths}{length\ of\ total\ paths} \times 100 \tag{7.2}$$

In Figure 7.20, the plan-based construction using GAs covered more paths than the other two approaches. In contrast, reactive construction secured about half of the total paths on average. The plan produced by the construction plan unit considers rescuer safety. In the reactive approach, there is no particular plan to protect the rescuers; the robot simply drops the blocks when it senses a considerable amount of radiation.

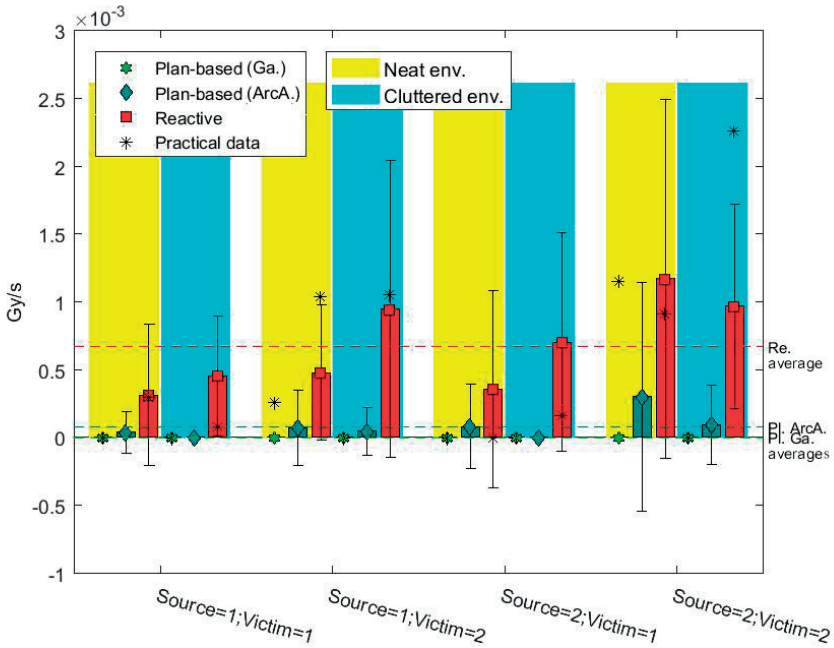


Figure 7.19 – Victim exposure at the end of *the least time*.

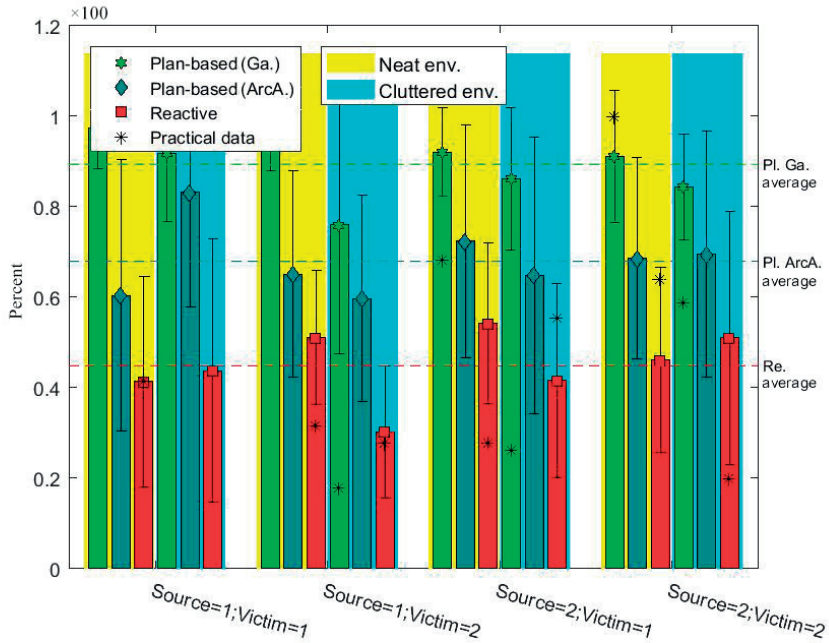


Figure 7.20 – Safety ratio of secured paths to whole paths.

7.5.4 Robotic activity

The last important objective of the rescue operation is to reduce the robot's cumulative exposure during the rescue operation. Figure 7.22 shows that the plan-based construction method exposed the robot to a higher level of radiation than the reactive method did. The cumulative exposure from the plan-based approach is the sum of the radiation absorbed in the exploration and in the construction steps. In plan-based construction, the robot spent a significant amount of time (approximately 60 seconds) precisely dropping a block where the closest radioactive sources were most likely to be (Figure 7.21). Moreover, in the exploration

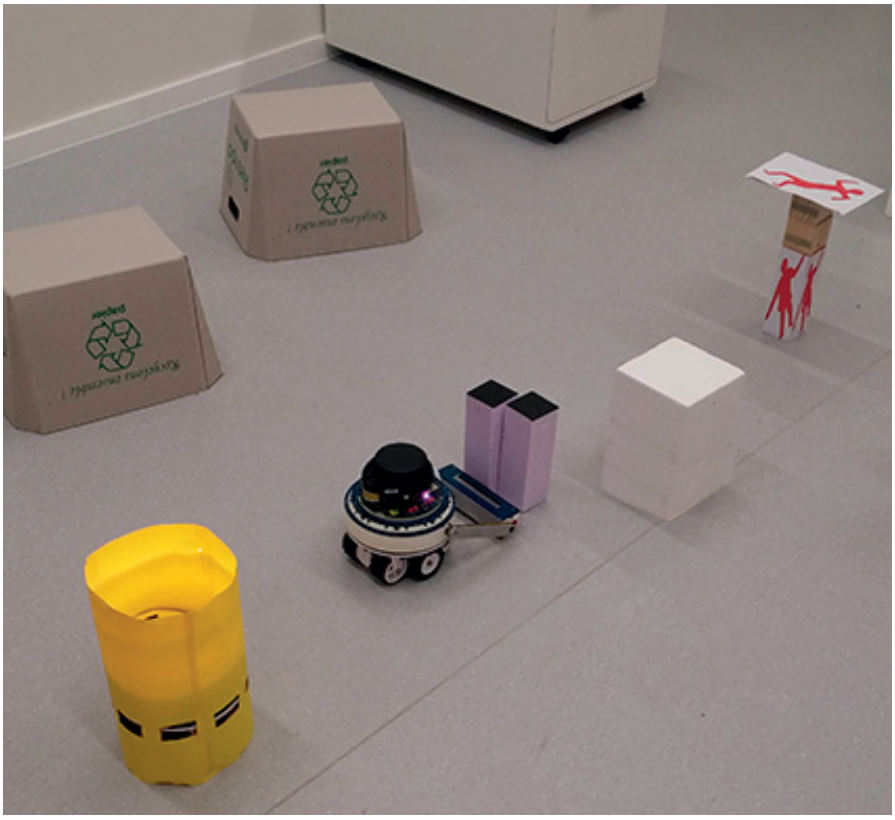


Figure 7.21 – A representation of the protective wall building. The marXbot drops the second block where it is close to a radioactive source in plan-based construction. The yellow object and red human symbol show the source and victim respectively.

phase, the robot regularly stopped for a few seconds (30 seconds), sometimes very close to a radioactive hotspot. According to the designed exploration algorithm, the robot must go near a radioactive source and then stop to check whether the source is real or not. Staying near a radioactive source exposed the robot to a considerable amount of radiation. This caused the more cumulative exposure in plan-based construction.

As noted in Chapter 5, GAs seek an effective plan based on victim, robot, and rescuer objectives.

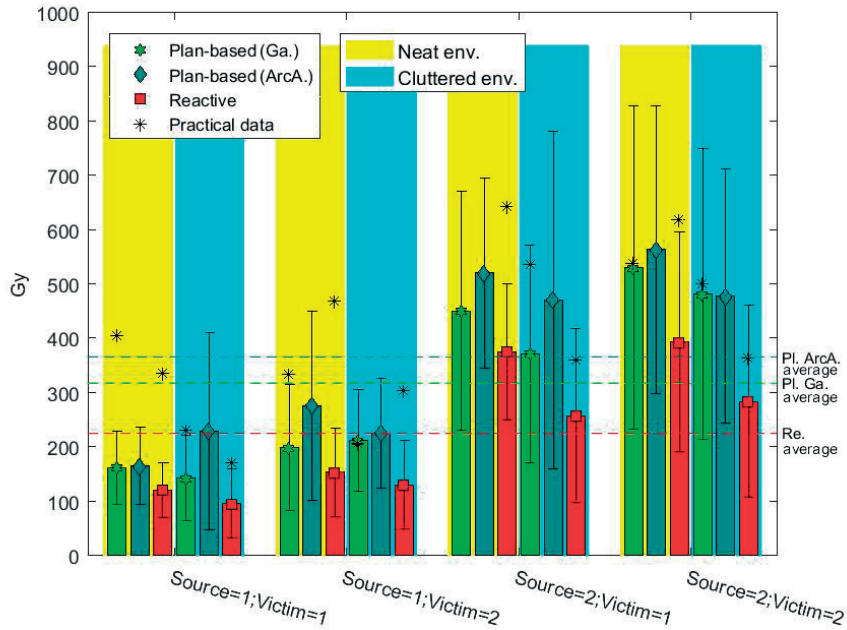


Figure 7.22 – Cumulative exposure on the robot after performing all tasks.

The graphs above have shown that plan-based construction using GAs had a better performance regarding protection of the victims and providing safe paths for rescuers; however, the robot was exposed to more radioactive waves than when it worked in reactive construction. Balancing the weights in the fitness function (Equation 5.3) could decrease robot exposure, but the other two objectives might lose ground. In Section 5.5, we showed that these objectives are competing.

Furthermore, the robot's cumulative exposure increases versus complexity (increasing the victim or source numbers) for all cases because it depends on the robot's presence in the environment for finding the victims or sources.

7.5.5 Validation

Most of the experiments in this study were performed in simulation (ideal conditions). The aim of this section is to assess how the results might differ when done in a real-world setup. We used a statistical method to evaluate important aspects. Symmetric mean absolute percentage error (SMAPE) is this statistical method based on the percentage errors and is used to assess validation aspects. It is defined as follows [129, 130]::

$$SMAPE = \frac{\sum_{i=1}^{n} |S_i - P_i|}{\sum_{i=1}^{n} (S_i + P_i)} \times 100 \tag{7.3}$$

where

S_i and P_i are simulation data and practical data respectively.

The numerator of Equation 7.3 consists of both the plan-based and reactive construction. Of course, for a few aspects (e.g., exploration time), we do not have the reactive data. In Figure 7.23, we illustrate some important aspects for evaluating the differences between the simulation and physical data. As mentioned, we performed a practical experiment for each configuration state; thus, the eight practical experiments for the plan-based (based on GAs) and eight experiments for reactive construction were carried out.

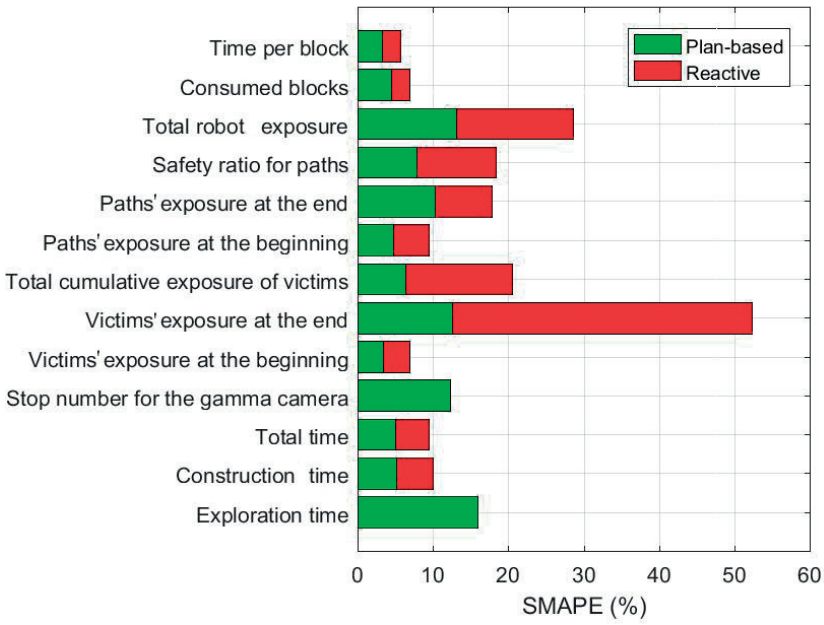


Figure 7.23 – Symmetric mean absolute percentage error

The first aspect we consider is time. SMAPE values for the total, construction, and exploration times are 9.5%, 10%, and 16%, respectively. These differences could be from movements of real robot. Because of the inaccuracies of the laser range scanner in measuring distance, the robot can sometimes lose its position. The robot then automatically starts the recovery behaviours to find its position again. Performing recovery behaviours affects the operation time for plan-based construction (the exploration and building times). In addition, the robot sometimes repeated the subtasks of *pick-up* or *drop* behaviour because of sensory inaccuracies and environmental imperfections. For instance, the robot could not successfully grasp a block, so it must repeat *find_block* subtask of the *pick-up* behaviour.

The exploration time also includes the time required to obtain the radioactive image which

can be increased by the gamma camera's limitations. Because to inaccuracies in the detection and positioning systems, the robot guesses the source candidates with less precision. This means that the robot needs to check more possible sources in order to find the real hotspots, so it will need to stop more.[12]

The victim exposure and path exposure at the beginning are items that are independent of the construction processes. The non-zero values for the related SMAPE indicate that the experiments did not have exactly the same conditions at the beginning. The elements' locations, such as the victims' locations, were not exactly the same in the two types of experiments because we physically arranged the environment within 10 cm precision in the real-world. In addition, the sources that were used in the real-world setup did not act the same as what we used in simulation because of uncertainties such as noise.

Because the positioning system works based on SLAM, it is inaccurate, especially for real-world use. Thus, the real robot places a block in the determined goal locations with less accuracy in plan-based construction. In reactive construction, the robot uses its odometry system, which has more errors in a real-world situation. The robots will not be able to drop the block exactly in the desired place as well as in the in the simulation. Accordingly, victim aspects, such as total cumulative exposure and exposure at the end, have considerable differences in simulation and in the real-world. The denominator of SMAPE for victim exposure at the end is a very small value, so it is highly sensitive to even minor differences.

Robot exposure is the integral of instantaneous radioactive intensity over time. As the robot moves along the different trajectories during different experiments, the radiation intensity varies between experiment to another. Thus, the SMAPE value for the total cumulative robot exposure is a considerable value. In addition, the range and bearing system that is used to model the radioactive source is not as accurate as the light system used in simulation.

7.6 Conclusion

In this thesis, we first developed AFAC principles, taking inspiration from nature. Here, we designed an achievable application to develop AFAC from scratch to a level at which robots can autonomously build real artifacts. The rescue application was well suited for studying how to formulate and implement AFAC. The purpose of using AFAC for a rescue operation was to build protective walls for decreasing radioactive exposure to victims, rescuers, and robots. The robot could build structures based on desired functions and adapt them to environmental conditions without using any blueprint.

After extensive experiments in simulation and on real robots, we uncovered promising results

[12] See section 3.4.1.

for rescue operations. AFAC was designed based on two different approaches: global and local. Comparisons between them revealed how efficient they were. Although AFAC based on a global approach (plan-based construction) required time upfront to explore the environment and compute a construction plan (in the GAs method), it could better protect victims and secure paths. However, the robot was exposed to more radiation during construction. Ultimately, GAs could not produce a better construction plan for the robot's cumulative radioactive exposure than reactive construction. On the other hand, reactive construction was quicker and simpler, but it did not protect as well as plan-based construction in terms of other rescue objectives, such as decreasing exposure to victims and rescuers.

One suggestion based on the discussed graphs of victim exposure is the development of a hybrid construction approach to improve AFAC for rescue purposes. For instance, two or more robots can work together, some of them performing reactive construction and others exploring the environment. As soon as the construction plan was produced, all of robots can start to build protective walls in the plan-based construction system. Another ideas is that a robot drops a block when it explores an environment or it computes a construction plan.

8 Conclusion

8.1 Final conclusion

In this thesis, we developed and studied a new method for autonomous construction, taking inspiration from animals to address existing challenges. We called this method AFAC. AFAC offers an intelligent and robust construction system with which to tackle the problems caused by various environmental conditions. The goal of autonomously build structures for particular functions instead of using a predefined construction plan (i.e., blueprint) and adapting the structures to fit the environmental conditions. AFAC does not use any predefined construction plan; forcing autonomous robots to build a structure based on a predefined plan decreases a robot's manoeuvrability in dynamic and unpredictable situations, or could construction defects. In our method, the robot conducted both building and architectural jobs to achieve an intelligent and robust construction system.

We have demonstrated that AFAC can be used for applications such as rescue. The idea for the rescue application was to secure an environment suffering from nuclear radiation and protect any victims inside this zone. Thus, we illustrated our approach by a mobile robot applying the AFAC method to build protective walls based on defined objectives. The first objective was to reduce the victims' cumulative radioactive exposure; meanwhile, the environment had to be secured for rescuers, and the robot had to minimize its exposure to radiation. These three objectives formed the functions required for AFAC.

Two different approaches were considered for AFAC. In the local approach, a robot built structures based on reactive behaviours without using any global representation or knowledge of the world. In contrast, in the global approach, a robot uses knowledge of the world to perform construction. Construction consists in three phases: exploration, computing a construction plan, and building. Comparisons between the local and global approaches revealed how efficient they were. Although AFAC based on a global approach spent extra times

to explore the environment and compute a construction plan, it could better protect victims and secure paths. In contrast, the robot was exposed to more radioactive radiation during construction. AFAC based on the local approach was quicker and simpler, and it exposed the robot to less radioactive exposure; but it did not satisfy other rescue objectives, such as decreasing exposure to victims and rescuers as well as AFAC based on the global approach.

The exploration and computation times of the first and second phases could significantly impact the performance of AFAC in the global approach, so we studied these phases independently of each other. In the exploration phase, we designed and studied effective exploration algorithms that fall under specific sensor limitations. The first limitation of the detection system (gamma camera) is the relatively long acquisition time for recording data. The second limitation is the poor angular resolution. Both limitations significantly degrade the performance of the exploration in terms of completion time and accuracy. Therefore, we presented the heuristic exploration and MCDM methods to autonomously map and localize the elements. These new methods were developed for mobile robots that used gamma cameras for radioactive hotspot detection and localization. We observed that the MCDM method can complete the exploration earlier and miss much fewer radioactive hotspots (target elements) in respect to the heuristic method.

In the computing a construction plan phase, we applied GAs to compute and optimize a construction plan based on collected data and defined objectives. Because of time limitations and safety issues when saving lives in disaster areas, we need to perform AFAC fast and safe; therefore, GAs were used to optimize a construction plan based on three objectives: victim safety, robot safety, and rescuer safety.

8.2 Contributions

This work provides the following contributions to the state-of-the-art:

- We have developed, implemented, and studied AFAC for use in the new class of construction applications. It provides an intelligent and robust construction system for building functional structures in different environmental conditions. We presented two approaches: the local one, which is based on reactive behaviours and has been already addressed in the literature. Second approach is the global approach, which is beyond the state-of-the-art; conversely, it uses global representations of the world to drive a robot for effective construction, even in dynamic and unforeseen environments.
- We have demonstrated for the first time that autonomous construction can be applied to rescue applications. The robot can stabilize large structures or protect the victims by performing construction tasks in a prototype environment. Our study also shows how

efficiently the approaches secure the environment and the victims' situations.

- We have developed new and effective exploration algorithms that can work within the limitations imposed by highly constraints sensory system (e.g., gamma camera). The limitations that we studied are the long acquisition time to record the required data and poor angular resolution. Both limitations degrade exploration in terms of completion time and accuracy, but our methods have demonstrated the ability to mitigate the unfavourable effects of the limitations.
- Another contribution of this research is the study of GAs for the providing and optimization of a functional construction plan. We have also evaluated the trade-offs between the three objectives to study how the objectives are competing and what sort of compromise can be found.

8.3 Outlook

In this work, we demonstrated that our autonomous construction approach (AFAC) can be applied to rescue applications. Our rescue experiments were performed in a small and artificial room that was far from a post-disaster situation. For sure the validation of the concepts developed in this thesis should be conducted in more realistic conditions. Robots must perform construction tasks in a challenging, uncluttered, and complex environments representative of real situations.

One strategy of AFAC is based on a global approach; it needs models and representations of the world to perform construction. Thus, we developed and implemented two different exploration methods (MCDM and heuristic) that must overcome the limitations of gamma cameras (the long acquisition time and poor angular resolution). However, these methods could be improved and optimized. For instance, one could study the weights that allow to perform MCDM exploration in an optimal way. Another possible research path on exploration methods would consist in using a team of robots, combining their contribution in a centralized way and following the approaches we have explored.

In this work, we applied GAs to compute and optimize a construction plan required for the global approach of AFAC. The fitness function of GA consists of the three terms that are multiplied by the corresponding weights. However, future studies could explore new methods. For instance, the multi-objective optimization (MOO) could be another approach to find a set of non-dominated solutions (Pareto front) [117] and it may provide better construction plans. The development and implications of simple computation algorithms (e.g., ArcA) could be also studied within our theoretical framework.

Another limitation of this work is in the study of 2D constructions. In Chapter 6, we demon-

strated an autonomous construction system for building 2D artifacts. Studying the precision of the built structure, which was based on a predefined blueprint, allowed us to modify the construction plan and compensate for construction errors. However, more efforts are needed in the case of 3D construction on an uneven surface. It would be interesting to broaden the scope of this work by using different types of blocks (e.g., male and female blocks) and methods.

Finally, a larger application field could be explored. In this work, we implemented AFAC for the rescue application. Further developments of AFAC could explore large complex structures evolved during the construction process. In the future, we can think to revolutionize the autonomous construction by offering all construction steps to an intelligent system. One could drop robots off at a site and come back several months later to see a enormous and fantastic building based on user-defined objectives. Although this is by now just a dream, we hope that our study will open a new avenue for future research going in this direction.

Bibliography

[1] J. Gould and C. Gould, *Animal Architects: Building and the Evolution of Intelligence*. Basic Books, 2012.

[2] M. Hansell, *Built by animals: the natural history of animal architecture*. OUP Oxford, 2007.

[3] H. Ardiny, S. Witwicki, and F. Mondada, "Are autonomous mobile robots able to take over construction? a review," *International Journal of Robotics, Theory and Applications*, vol. 4, no. 3, pp. 10–21, 2015.

[4] "Construction.", *Oxford English Dictionary*. Oxford University Press. [Online]. Available: http://www.oxforddictionaries.com/

[5] K. S. Saidi, J. B. O'Brien, and A. M. Lytle, *Robotics in Construction*. springer, 2008, ch. 47, pp. 1079–1099.

[6] L. Cousineau and N. Miura, *Construction Robots: The Search for New Building Technology in Japan*. ASCE, 1998.

[7] H. Ardiny, S. Witwicki, and F. Mondada, "Providing and Optimizing a Robotic Construction Plan for Rescue Operations," in *The 2015 IEEE International Workshop on Advanced Robotics and its Social Impacts (ARSO 2015)*, 2015.

[8] R. P. Hoyt, J. I. Cushing, J. T. Slostad, G. Jimmerson, T. Moser, G. Kirkos, M. L. Jaster, and N. R. Voronka, "Spiderfab: An architecture for self-fabricating space systems," *Am. Inst. Aeronaut. Astronaut*, p. 5509, 2013.

[9] B. Khoshnevis, M. Bodiford, K. Burks, E. Ethridge, D. Tucker, W. Kim, H. Toutanji, and M. Fiske, "Lunar contour crafting-a novel technique for isru-based habitat development," in *43rd AIAA Aerospace Sciences Meeting and Exhibit*, 2005, p. 538.

[10] N. P. Hack, W. V. Lauer, F. M. Gramazio, M. Kohler, and N. Blank, "Method of fabricating a 3-dimensional structure, mesh formwork element for fabricating a 3-dimensional structure, and method of fabricating the same," 2014, uS Patent App. 14/916,475.

[11] Balaguer Carlos and Abderrahim Mohamed, "Trends in Robotics and Automation in Construction," in *Robotics and Automation in Construction*, Balaguer Carlos and Abderrahim Mohamed, Eds. InTech, 2008.

[12] V. Helm, S. Ercan, F. Gramazio, and M. Kohler, "Mobile robotic fabrication on construction sites: Dimrob," in *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on*. IEEE, 2012, pp. 4335–4341.

[13] J. Willmann, F. Augugliaro, T. Cadalbert, R. D'Andrea, F. Gramazio, and M. Kohler, "Aerial Robotic Construction Towards a New Field of Architectural Research," *International Journal of Architectural Computing*, vol. 10, pp. 439–460, 2012.

[14] S. Wismer, G. Hitz, M. Bonani, A. Gribovskiy, and S. Magnenat, "Autonomous construction of a roofed structure: Synthesizing planning and stigmergy on a mobile robot," in *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on*, 2012, pp. 5436–5437.

[15] R. A. Rihani and L. E. Bernold, "Computer integration for robotic masonry," *Computer-Aided Civil and Infrastructure Engineering*, vol. 9, no. 1, pp. 61–67, 1994.

[16] G. Pritschow, M. Dalacker, J. Kurz, and M. Gaenssle, "Technological aspects in the development of a mobile bricklaying robot," *Automation in Construction*, vol. 5, no. 1, pp. 3 – 13, 1996.

[17] "Tiger stone website," 2015, [Online; accessed 20-Nov-2015]. [Online]. Available: http://www.tiger-stone.nl/

[18] "Construction robotics website," 2015, [Online; accessed 20-Nov-2015]. [Online]. Available: http://construction-robotics.com/

[19] M. A. Goodrich and A. C. Schultz, "Human-robot interaction: A survey," *Found. Trends Hum.-Comput. Interact.*, vol. 1, no. 3, pp. 203–275, 2007.

[20] Q. Lindsey, D. Mellinger, and V. Kumar, "Construction of cubic structures with quadrotor teams," in *Robotics: Science and Systems*, H. F. Durrant-Whyte, N. Roy, and P. Abbeel, Eds., 2011.

[21] F. Nigl, S. Li, J. Blum, and H. Lipson, "Structure-reconfiguring robots: Autonomous truss reconfiguration and manipulation," *Robotics Automation Magazine, IEEE*, vol. 20, no. 3, pp. 60–71, 2013.

[22] S. Shepherd and A. Buchstab, "Kuka robots on-site," in *Robotic Fabrication in Architecture, Art and Design 2014*, W. McGee and M. Ponce de Leon, Eds. Springer International Publishing, 2014, pp. 373–380.

[23] B. Khoshnevis, "Automated construction by contour crafting—related robotics and information technologies," *Automation in Construction*, vol. 13, no. 1, pp. 5–19, 2004.

[24] F. Gramazio and M. Kohler, *Made by robots: challenging architecture at a larger scale*. John Wiley & Sons, 2014.

[25] S. Jokic, P. Novikov, S. Maggs, D. Sadan, S. Jin, and C. Nan, "Robotic positioning device for three-dimensional printing," *arXiv preprint arXiv:1406.3400*, 2014.

[26] P. Rétornaz, "Construction par procédé additif réalisée avec des robots mobiles autonomes hétérogènes," Ph.D. dissertation, 2015.

[27] M. B. Griggs, "Robots can now 3d-print steel bridges," 2015, [Online; accessed 08-Dec-2015]. [Online]. Available: http://www.popsci.com/now-we-can-3d-print-steel-bridges

[28] N. Napp and R. Nagpal, "Robotic Construction of Arbitrary Shapes with Amorphous Materials," in *Conf on Robotics and Automation (ICRA)*, 2014.

[29] N. Napp, O. Rappoli, J. Wu, and R. Nagpal, "Materials and mechanisms for amorphous robotic construction," in *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on*, 2012, pp. 4879–4885.

[30] T. Soleymani, V. Trianni, M. Bonani, F. Mondada, and M. Dorigo, "Bio-inspired construction with mobile robots and compliant pockets," *Robotics and Autonomous Systems*, vol. 74, pp. 340–350, 2015.

[31] R. Fujisawa, N. Nagaya, S. Okazaki, R. Sato, Y. Ikemoto, and S. Dobata, "Active modification of the environment by a robot with construction abilities," *ROBOMECH Journal*, vol. 2, no. 1, p. 1, 2015.

[32] J. Werfel, K. Petersen, and R. Nagpal, "Designing collective behavior in a termite-inspired robot construction team," *Science*, vol. 343, no. 6172, pp. 754–758, 2014.

[33] Y. Terada and S. Murata, "Automatic Modular Assembly System and its Distributed Control," *I. J. Robotic Res.*, vol. 27, no. 3-4, pp. 445–462, 2008.

[34] K. Galloway, R. Jois, and M. Yim, "Factory floor: A robotically reconfigurable construction platform," in *Robotics and Automation (ICRA), 2010 IEEE International Conference on*, 2010, pp. 2467–2472.

[35] N. Gershenfeld, M. Carney, B. Jenett, S. Calisch, and S. Wilson, "Macrofabrication with digital materials: Robotic assembly," *Architectural Design*, vol. 85, no. 5, pp. 122–127, 2015.

[36] J. Hiller and H. Lipson, "Design and analysis of digital materials for physical 3d voxel printing," *Rapid Prototyping Journal*, vol. 15, no. 2, pp. 137–149, 2009.

[37] S. Magnenat, R. Philippsen, and F. Mondada, "Autonomous construction using scarce resources in unknown environments," *Autonomous Robots*, vol. 33, no. 4, pp. 467–485, 2012.

[38] A. Stroupe, A. Okon, M. Robinson, T. Huntsberger, H. Aghazarian, and E. Baumgartner, "Sustainable cooperative robotic technologies for human and robotic outpost infrastructure construction and maintenance," *Autonomous Robots*, vol. 20, no. 2, pp. 113–123, 2006.

[39] D. H. Jung, J. Park, and M. Schwartz, "Towards on-site autonomous robotic floor tiling of mosaics," in *Control, Automation and Systems (ICCAS), 2014 14th International Conference on*. IEEE, 2014, pp. 59–63.

[40] K. Dörfler, T. Sandy, M. Giftthaler, F. Gramazio, M. Kohler, and J. Buchli, "Mobile robotic brickwork-automation of a discrete robotic fabrication process using an autonomous mobile robot," *Robotic Fabrication in Architecture, Art and Design*, 2016.

[41] H. Ardiny, S. Witwicki, and F. Mondada, "Autonomous Construction of Separated Artifacts by Mobile Robots using SLAM and Stigmergy," in *Conference on Autonomous and Robotic Construction of Infrastructure*, 2015.

[42] D. Schmidt and K. Berns, "Construction site navigation for the autonomous excavator thor," in *Automation, Robotics and Applications (ICARA), 2015 6th International Conference on*. IEEE, 2015, pp. 90–97.

[43] J. Korb and K. E. Linsenmair, "The architecture of termite mounds: a result of a trade-off between thermoregulation and gas exchange?" *Behavioral Ecology*, vol. 10, no. 3, pp. 312–316, 1999.

[44] C. Detrain, J. L. Deneubourg, and J. M. Pasteels, *Information processing in social insects*. Springer Science & Business Media, 1999.

[45] W. R. Tschinkel, "The nest architecture of the florida harvester ant, pogonomyrmex badius," *Journal of Insect Science*, vol. 4, no. 1, p. 21, 2004.

[46] N. S. Estevez, "Functional blueprints: A dynamical approach to structure representation," Master's thesis, Cornell University, 2007.

[47] I. Navarro and F. Matía, "An introduction to swarm robotics," *ISRN Robotics*, vol. 2013, 2012.

[48] Y. Mohan and S. Ponnambalam, "An extensive review of research in swarm robotics," in *Nature Biologically Inspired Computing, 2009. NaBIC 2009. World Congress on*, 2009, pp. 140–145.

[49] M. Brambilla, E. Ferrante, M. Birattari, and M. Dorigo, "Swarm robotics: a review from the swarm engineering perspective," *Swarm Intelligence*, vol. 7, no. 1, pp. 1–41, 2013.

[50] J. Werfel, "Building blocks for multi-robot construction," in *Distributed Autonomous Robotic Systems 6*, R. Alami, R. Chatila, and H. Asama, Eds. Springer Japan, 2007, pp. 285–294.

[51] J. Werfel, K. Petersen, and R. Nagpal, "Distributed multi-robot algorithms for the termes 3d collective construction system," in *Proceedings of Robotics: Science and Systems VII*, 2011.

[52] C. A. C. Parker and H. Zhang, "Collective robotic site preparation," *Adaptive Behavior*, vol. 14, no. 1, pp. 5–19, 2006.

[53] J. Werfel and R. Nagpal, "Extended stigmergy in collective construction," *IEEE Intelligent Systems*, vol. 21, no. 2, pp. 20–28, 2006.

[54] J. Werfel, D. Ingber, and R. Nagpal, "Collective construction of environmentally-adaptive structures," *2007 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 2345–2352, 2007.

[55] C. Melhuish, J. Welsby, and C. Edwards, "Using templates for defensive wall building with autonomous mobile antlike robots," in *Proceedings of Towards Intelligent Mobile Robots (TIMR'99)*, 1999.

[56] B. Khoshnevis and G. Bekey, "Centralized sensing and control of multiple mobile robots," *Computers and Industrial Engineering*, vol. 35, no. 3, pp. 503 – 506, 1998.

[57] Y. U. Cao, A. S. Fukunaga, and A. B. Kahng, "Cooperative mobile robotics: Antecedents and directions," *Autonomous Robots*, vol. 4, pp. 226—-234, 1997.

[58] L. E. Parker, "Multiple mobile robot systems," in *Springer Handbook of Robotics*. Springer, 2008, pp. 921–941.

[59] B. Akinci, F. Boukamp, C. Gordon, D. Huber, C. Lyons, and K. Park, "A formalism for utilization of sensor systems and integrated project models for active construction quality control," *Automation in Construction*, vol. 15, no. 2, pp. 124–138, 2006.

[60] H. Lemaire, R. A. Khalil, K. Amgarou, J.-C. Angélique, F. Bonnet, D. De Toro, F. Carrel, O. Giarmana, M. Gmar, N. Menaa *et al.*, "Implementation of an imaging spectrometer for

localization and identification of radioactive sources," *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, vol. 763, pp. 97–103, 2014.

[61] Y. Calderón, M. Chmeissani, M. Kolstein, P. Casado *et al.*, "Design, development, and modeling of a compton camera tomographer based on room temperature solid statepixel detector," Ph.D. dissertation, 2014.

[62] Y. Kong, H. Brands, T. Glaser, C.-M. Herbach, L. Hoy, M. Kreuels, M. Küster, G. Pausch, J. Petzoldt, C. Plettner *et al.*, "A prototype compton camera array for localization and identification of remote radiation sources," *IEEE Transactions on Nuclear Science*, vol. 60, no. 2, pp. 1066–1071, 2013.

[63] H. O. Anger, "Scintillation camera," *Review of scientific instruments*, vol. 29, no. 1, pp. 27–33, 1958.

[64] F. Carrel, R. A. KHALIL, P. Blot, K. Boudergui, S. Colas, M. Gmar, F. Lemasle, N. Saurel, V. SCHOEPFF, and H. Toubon, "Gampix: a new generation of gamma camera for hot spot localisation," in *Proceedings of the ISOE conference, Cambridge*, 2010.

[65] S. Burion, "Human detection for robotic urban search and rescue," Tech. Rep., 2004.

[66] M. Andriluka, P. Schnitzspan, J. Meyer, S. Kohlbrecher, K. Petersen, O. Von Stryk, S. Roth, and B. Schiele, "Vision based victim detection from unmanned aerial vehicles," in *Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference on*. IEEE, 2010, pp. 1740–1747.

[67] S. Bahadori, L. Iocchi, D. Nardi, and G. P. Settembre, "Stereo vision based human body detection from a localized mobile robot," in *IEEE Conference on Advanced Video and Signal Based Surveillance, AVSS 2005*, 2005.

[68] A. Kleiner and R. Kummerle, "Genetic mrf model optimization for real-time victim detection in search and rescue," in *2007 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2007, pp. 3025–3030.

[69] J. Pissokas and C. Malcolm, "Experiments with sensors for urban search and rescue robots," in *Proceedings of the International Symposium on Robotics and Automation (ISRA'02)*, 2002.

[70] A. Kleiner, B. Steder, C. Dornhege, D. Höfer, D. Meyer-Delius, J. Prediger, J. Stueckler, K. Glogowski, M. Thurner, M. Luber *et al.*, "Robocuprescue-robot league team rescuer-obots freiburg (germany)," in *In RoboCup 2005 Team Description Paper, Rescue Robot League*, 2005.

[71] A. Maggu, K. Rana, M. Kumar, and M. D. S. Chaurasia, "Life detecting robot for rescue operations," *International Journal of Information Technology and Knowledge Management*, vol. 4, no. 2, pp. 653–657, 2011.

[72] K.-M. Chen, Y. Huang, J. Zhang, and A. Norman, "Microwave life-detection systems for searching human subjects under earthquake rubble or behind barrier," *IEEE transactions on biomedical engineering*, vol. 47, no. 1, pp. 105–114, 2000.

[73] I. R. Nourbakhsh, K. Sycara, M. Koes, M. Yong, M. Lewis, and S. Burion, "Human-robot teaming for search and rescue," *IEEE Pervasive Computing*, vol. 4, no. 1, pp. 72–79, 2005.

[74] M. Bonani, "Robotique collective et auto-assemblage," Ph.D. dissertation, EPFL, 2010.

[75] P. Rétornaz, S. Magnenat, and F. Vaussard, "Marxbot user manual, version 1.1," 2011, [Online; accessed 05-Jan-2017]. [Online]. Available: http://www.http://mobots.epfl.ch/data/robots/marxbot-user-manual.pdf

[76] S. Magnenat, "Software integration in mobile robotics, a science to scale up machine intelligence," Ph.D. dissertation, EPFL, 2010.

[77] K. Konolige, J. Augenbraun, N. Donaldson, C. Fiebig, and P. Shah, "A low-cost laser distance sensor," in *2008 IEEE International Conference on Robotics and Automation*, 2008, pp. 3002–3008.

[78] C. Pinciroli, V. Trianni, R. O'Grady, G. Pini, A. Brutschy, M. Brambilla, N. Mathews, E. Ferrante, G. Di Caro, F. Ducatelle, M. Birattari, L. M. Gambardella, and M. Dorigo, "ARGoS: a modular, parallel, multi-engine simulator for multi-robot systems," *Swarm Intelligence*, vol. 6, no. 4, pp. 271–295, 2012.

[79] ROS, "Ros introduction," [Online; accessed 10-Jan-2017]. [Online]. Available: http://wiki.ros.org/ROS/Introduction

[80] S. Kohlbrecher, J. Meyer, O. von Stryk, and U. Klingauf, "A flexible and scalable slam system with full 3d motion estimation," in *Proc. IEEE International Symposium on Safety, Security and Rescue Robotics (SSRR)*. IEEE, 2011.

[81] ROS, "Ros move base," [Online; accessed 10-Jan-2017]. [Online]. Available: http://wiki.ros.org/move_base

[82] W. H. Hallenbeck, *Radiation protection*. CRC Press, 1994.

[83] R. R. Murphy, S. Tadokoro, D. Nardi, A. Jacoff, P. Fiorini, H. Choset, and A. M. Erkmen, "Search and rescue robotics," in *Springer Handbook of Robotics*. Springer, 2008, pp. 1151–1173.

[84] R. R. Murphy, J. Kravitz, S. L. Stover, and R. Shoureshi, "Mobile robots in mine rescue and recovery," *IEEE Robotics & Automation Magazine*, vol. 16, no. 2, pp. 91–103, 2009.

[85] P. Rudol and P. Doherty, "Human body detection and geolocalization for uav search and rescue missions using color and thermal imagery," in *Aerospace Conference, 2008 IEEE*. IEEE, 2008, pp. 1–8.

[86] T. Tomic, K. Schmid, P. Lutz, A. Domel, M. Kassecker, E. Mair, I. L. Grixa, F. Ruess, M. Suppa, and D. Burschka, "Toward a fully autonomous uav: Research platform for indoor and outdoor urban search and rescue," *IEEE robotics & automation magazine*, vol. 19, no. 3, pp. 46–56, 2012.

[87] T. G. McGee and J. K. Hedrick, "Guaranteed strategies to search for mobile evaders in the plane," in *2006 American Control Conference*. IEEE, 2006, pp. 6–pp.

[88] N. R. Hoff III, "Multi–robot foraging for swarms of simple robots," Ph.D. dissertation, Citeseer, 2011.

[89] A. F. Winfield, "Towards an engineering science of robot foraging," in *Distributed Autonomous Robotic Systems 8*. Springer, 2009, pp. 185–192.

[90] S. Alers, D. Claes, K. Tuyls, and G. Weiss, "Biologically inspired multi-robot foraging," in *Proceedings of the 2014 international conference on Autonomous agents and multi-agent systems*. International Foundation for Autonomous Agents and Multiagent Systems, 2014, pp. 1683–1684.

[91] R. C. Arkin and J. Diaz, "Line-of-sight constrained exploration for reactive multiagent robotic teams," in *Advanced Motion Control, 2002. 7th International Workshop on*. IEEE, 2002, pp. 455–461.

[92] B. Yamauchi, "A frontier-based approach for autonomous exploration," in *Computational Intelligence in Robotics and Automation, 1997. CIRA'97., Proceedings., 1997 IEEE International Symposium on*. IEEE, 1997, pp. 146–151.

[93] W. Burgard, M. Moors, C. Stachniss, and F. E. Schneider, "Coordinated multi-robot exploration," *IEEE Transactions on robotics*, vol. 21, no. 3, pp. 376–386, 2005.

[94] S. Wirth and J. Pellenz, "Exploration transform: A stable exploring algorithm for robots in rescue environments," in *2007 IEEE International Workshop on Safety, Security and Rescue Robotics*. IEEE, 2007, pp. 1–5.

[95] H. H. Gonzalez-Banos and J.-C. Latombe, "Navigation strategies for exploring indoor environments," *The International Journal of Robotics Research*, vol. 21, no. 10-11, pp. 829–848, 2002.

[96] B. Tovar, L. Munoz-Gómez, R. Murrieta-Cid, M. Alencastre-Miranda, R. Monroy, and S. Hutchinson, "Planning exploration strategies for simultaneous localization and mapping," *Robotics and Autonomous Systems*, vol. 54, no. 4, pp. 314–331, 2006.

[97] F. Amigoni and A. Gallo, "A multi-objective exploration strategy for mobile robots," in *Proceedings of the 2005 IEEE International Conference on Robotics and Automation*. IEEE, 2005, pp. 3850–3855.

[98] N. Basilico and F. Amigoni, "Exploration strategies based on multi-criteria decision making for searching environments in rescue operations," *Autonomous Robots*, vol. 31, no. 4, pp. 401–417, 2011.

[99] H. Sugiyama, T. Tsujioka, and M. Murata, "Real-time exploration of a multi-robot rescue system in disaster areas," *Advanced Robotics*, vol. 27, no. 17, pp. 1313–1323, 2013.

[100] R. Cortez, H. Tanner, and R. Lumia, "Distributed robotic radiation mapping," in *Experimental Robotics*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009, vol. 54, ch. 10, pp. 174–156.

[101] N. S. Rao, M. Shankar, J.-C. Chin, D. K. Yau, S. Srivathsan, S. S. Iyengar, Y. Yang, and J. C. Hou, "Identification of low-level point radiation sources using a sensor network," in *Information Processing in Sensor Networks, 2008. IPSN'08. International Conference on*. IEEE, 2008, pp. 493–504.

[102] M. Chandy, C. Pilotto, and R. McLean, "Networked sensing systems for detecting people carrying radioactive material," in *Networked Sensing Systems, 2008. INSS 2008. 5th International Conference on*. IEEE, 2008, pp. 148–155.

[103] J. W. Howse, L. O. Ticknor, and K. R. Muske, "Least squares estimation techniques for position tracking of radioactive sources," *Automatica*, vol. 37, no. 11, pp. 1727–1737, 2001.

[104] H. E. Baidoo-Williams, "Novel techniques for estimation and tracking of radioactive sources," Ph.D. dissertation, 2014.

[105] J. Towler, B. Krawiec, and K. Kochersberger, "Radiation mapping in post-disaster environments using an autonomous helicopter," *Remote Sensing*, vol. 4, no. 7, pp. 1995–2015, 2012.

[106] A. A. R. Newaz, S. Jeong, H. Lee, H. Ryu, and N. Y. Chong, "Uav-based multiple source localization and contour mapping of radiation fields," *Robotics and Autonomous Systems*, vol. 85, pp. 12–25, 2016.

[107] Y. Du, Z. He, G. Knoll, D. Wehe, and W. Li, "Evaluation of a compton scattering camera using 3-d position sensitive cdznte detectors," *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, vol. 457, no. 1, pp. 203–211, 2001.

[108] D. Xu, Z. He, C. E. Lehner, and F. Zhang, "4-pi compton imaging with single 3d position-sensitive cdznte detector," in *Optical Science and Technology, the SPIE 49th Annual Meeting*. International Society for Optics and Photonics, 2004, pp. 144–155.

[109] C. E. Lehner, Z. He, and F. Zhang, "4π compton imaging using a 3-d position-sensitive cdznte detector via weighted list-mode maximum likelihood," *IEEE Transactions on Nuclear Science*, vol. 51, no. 4, pp. 1618–1624, 2004.

[110] C. Papadimitropoulos, I. Kaissas, C. Potiriadis, K. Karafasoulis, D. Loukas, and C. Lambropoulos, "Radioactive source localization by a two detector system," *Journal of Instrumentation*, vol. 10, no. 12, p. C12022, 2015.

[111] J. M. Jaworski and Z. He, "3d compton image reconstruction using a moving 3d-position-sensitive room-temperature cdznte detector array," in *Nuclear Science Symposium and Medical Imaging Conference (NSS/MIC), 2011 IEEE*. IEEE, 2011, pp. 976–981.

[112] B. Deb, J. F. Ross, A. Ivan, and M. J. Hartman, "Radioactive source estimation using a system of directional and non-directional detectors," *IEEE Transactions on Nuclear Science*, vol. 58, no. 6, pp. 3281–3290, 2011.

[113] T. M. Cover and J. A. Thomas, *Elements of information theory*. John Wiley & Sons, 2012.

[114] P. Fankhauser and M. Hutter, "A Universal Grid Map Library: Implementation and Use Case for Rough Terrain Navigation," in *Robot Operating System (ROS) – The Complete Reference (Volume 1)*, A. Koubaa, Ed. Springer, 2016, ch. 5. [Online]. Available: http://www.springer.com/de/book/9783319260525

[115] J. Meyer, "hector worldmodel," 2016, [Online; accessed 04-Dec-2016]. [Online]. Available: https://http://wiki.ros.org/hector_worldmodel

[116] J. Meyer, P. Schnitzspan, S. Kohlbrecher, K. Petersen, M. Andriluka, O. Schwahn, U. Klingauf, S. Roth, B. Schiele, and O. von Stryk, "A semantic world model for urban search and rescue based on heterogeneous sensors," in *RoboCup 2010: Robot Soccer World Cup XIV*. Springer, 2011, pp. 180–193.

[117] R. L. Haupt and S. E. Haupt, *Practical genetic algorithms*. Hoboken, N.J.: J. Wiley, 2004.

[118] C. Melhuish, J. Welsby, and C. Edwards, "Using templates for defensive wall building with autonomous mobile ant-like robots," in *Proceedings of Towards Intelligent Autonomous Mobile Robot*. Morgan Kaufmann Publishers, 1999, pp. 1495–1502.

[119] R. L. Stewart and R. A. Russell, "A distributed feedback mechanism to regulate wall construction by a robotic swarm." *Adaptive Behaviour*, vol. 14, no. 1, pp. 21–51, 2006.

[120] G. F. Knoll, *Radiation detection and measurement*, 2nd ed. New York: JOHN WILEY & SONS, 1989.

[121] R. Goutam and R. Khare, "A Comprehensive paper on mobile robot path planning using wave front approach," *Int. J. Tech*, vol. 1, no. 2, pp. 143–148, 2011.

[122] S. J. Witwicki and F. Mondada, "Circumventing robots' failures by embracing their faults: A practical approach to planning for autonomous construction." in *AAAI*, 2015, pp. 4298–4299.

[123] C. J. Muise, V. Belle, and S. A. McIlraith, "Computing contingent plans via fully observable non-deterministic planning." in *AAAI*. Citeseer, 2014, pp. 2322–2329.

[124] J. Korb, *Termite Mound Architecture, from Function to Construction*. Dordrecht: Springer Netherlands, 2011, pp. 349–373.

[125] J. S. Turner, "Architecture and morphogenesis in the mound of macrotermes michaelseni (sjöstedt)(isoptera: Termitidae, macrotermitinae) in northern namibia," *Cimbebasia*, vol. 16, pp. 143–175, 2000.

[126] M. J. Matari´c and F. Michaud, "Behavior-based systems," in *Handbook of Robotics*, B. Siciliano and O. Khatib, Eds. springer, 2008, ch. 38, pp. 891–909.

[127] R. Arkin, "Reactive robotic systems ronald c. arkin college of computing georgia institute of technology atlanta, georgia," 1995.

[128] J.-L. Blanco, J. González, and J.-A. Fernández-Madrigal, "Extending obstacle avoidance methods through multiple parameter-space transformations," *Autonomous Robots*, vol. 24, no. 1, pp. 29–48, 2008.

[129] L. Surhone, M. Tennoe, and S. Henssonow, *Symmetric Mean Absolute Percentage Error*. VDM Publishing, 2010.

[130] Wikipedia, "Symmetric mean absolute percentage error— Wikipedia, the free encyclopedia," 2016, [Online; accessed 25-Oct-2016]. [Online]. Available: https://en.wikipedia.org/wiki/Symmetric_mean_absolute_percentage_error

List of Acronyms

ABB ASEA Brown Boveri.

AFAC Adaptive and Functional Autonomous Construction.

AR Angular Resolution.

ArcA Arc Algorithm.

CAN Controller-Area Network.

CMOS Complementary Metal-Oxide-Semiconductor.

CO2 Carbon Dioxide.

CPU Central Processing Unit.

DOF Degree Of Freedom.

EPFL École Polytechnique Fédérale de Lausanne.

ETH Eidgenössische Technische Hochschule.

FOV Field Of View.

GA Genetic Algorithm.

GNSS Global Navigation Satellite System.

GPS Global Positioning System.

I2C Inter-Integrated Circuit.

ID Identification.

IDE Integrated Development Environment.

IMU Inertial Measurement Unit.

LiDAR Light Detection and Ranging.

LSRO Laboratoire de Systèmes Robotiques.

MAV Micro Aerial Vehicle.

MCDM Multi-criteria Decision Making.

MOO Multi-Objective Optimization.

MRS Multi-Robot System.

NASA National Aeronautics and Space Administration.

NBV Next Best View.

PC Personal Computer.

PhD Philosophiae Doctor.

PID Proportional Integral Derivative Controller.

PWM Pulse Width Modulation.

RAM Random Access Memory.

ROS Robot Operating System.

SAM Semi-Automated Masonry.

SAROP Search And Rescue Operation.

SLAM Simultaneous Localization And Mapping.

SMAPE Symmetric Mean Absolute Percentage Error.

SRL Short-Range Relative Localization.

TP-Space Trajectory Parameter Space.

UAV Unmanned Aerial Vehicle.

USB Universal Serial Bus.

VM Virtual Machine.

Wi-Fi Wireless Fidelity.

List of Symbols

E Energy.

H Entropy used in Shannon information.

I The intensity of the radiation source.

K_d Derivative gain of a PID controller.

K_i Integral gain of a PID controller.

K_p Proportional gain of a PID controller.

Pr A probability value to represent occupancy status of a cell.

P A set of points.

Sd Information gained in new discovered regions.

$\delta\theta_{error}$ Difference between the expected yaw angle and measured yaw angle.

π Ratio of circumference of circle to its diameter.

θ Angle.

ch A chromosome used in GA.

d_s Minimum distance between two separated sources.

d Distance between two objects.

pr Number of possible real elements.

tos The total of the *support*s.

Address: EPFL-LSRO-ME B3 30 (Bâtiment ME), Station 9, CH-1015 Lausanne-Switzerland
Telephone: 0041 787 212 134
Email: hadiardiny@yahoo.com

Hadi Ardiny

Personal Profile:

Gender: Male
Date of Birth: 1985
Place of Birth: Iran-Tehran
State of Marry: Married
Citizenship: Iranian

Education:

2012 – 2017 **Ph.D.** in Mobile Robotics, École Polytechnique Fédérale de Lausanne (EPFL), Lausanne, Switzerland

2008-2010 **M.Sc.** in Mechatronic Engineering, K.N. Toosi University of Technology, Tehran, Iran

2004-2008 **Bachelor** in Mechanical Engineering, K.N. Toosi University of Technology, Tehran, Iran

Professional Experience:

2008 – 2012 **R&D Engineer** Serat Noor Rezvan, Tehran, Iran

2007-2012 **Teacher**, Tehran, Iran

2007 – 2008 (3 months) **Engineer Intern**, Ventilation Industrial, Sarma Afarin, Ghazvin, Iran

Languages:

Persian	Native
English	B2-C1
Arabic	B1
French	A2

Publications:

1. **Hadi Ardiny**, Stefan Witwicki, and Francesco Mondada. *"Are Autonomous Mobile Robots Able to Take Over Construction? A Review." International Journal of Robotics, Theory and Applications* 4.3 (2015): 10-21.
2. **Hadi Ardiny**, Stefan Witwicki, and Francesco Mondada. *"Construction automation with autonomous mobile robots: A review." Robotics and Mechatronics (ICROM), 2015 3rd RSI International Conference on*. IEEE, 2015.
3. **Hadi Ardiny**, Stefan Witwicki, and Francesco Mondada. *"Providing and Optimizing a Robotic Construction Plan for Rescue Operations." The 2015 IEEE International Workshop on Advanced Robotics and its Social Impacts (ARSO 2015)*. No. EPFL-CONF-208859. 2015.
4. **Hadi Ardiny**, Stefan Witwicki, and Francesco Mondada. *"Autonomous Construction of Separated Artifacts by Mobile Robots using SLAM and Stigmergy." Conference on Autonomous and Robotic Construction of Infrastructure*. 2015.
5. **Hadi Ardiny**, Stefan Witwicki, and Francesco Mondada. "*Applying Automated Objective Construction Principles to Autonomously Secure an Environment After a Nuclear Disaster Using Mobile Robots*", 2nd Workshop on Robotics and Automation in Nuclear Facilities IEEE-RSJ IROS2015, Hamburg, Germany.