Coordination and Control of Multi-Robot Systems

Rachael N. Duca

A thesis submitted to the University of Malta
for the degree of Doctor of Philosophy by the Faculty Engineering

Department of Systems and Control Engineering

December 2023

L-Università ta' Malta

FACULTY/INSTITUTE/CENTRE/SCHOOL Faculty of Engineering

DECLARATION OF AUTHENTICITY FOR DOCTORAL STUDENTS

(a) Authenticity of Thesis/Dissertation

I hereby declare that I am the legitimate author of this Thesis/Dissertation and that it is my original work.

No portion of this work has been submitted in support of an application for another degree or qualification of this or any other university or institution of higher education.

I hold the University of Malta harmless against any third party claims with regard to copyright violation, breach of confidentiality, defamation and any other third party right infringement.

(b) Research Code of Practice and Ethics Review Procedure

I declare that I have abided by the University's Research Ethics Review Procedures. Research Ethics & Data Protection form code 4900_16042020.

☒ As a Ph.D. student, as per Regulation 66 of the Doctor of Philosophy Regulations, I accept that my thesis be made publicly available on the University of Malta Institutional Repository.

□ As a Doctor of Sacred Theology student, as per Regulation 17 (3) of the Doctor of Sacred Theology Regulations, I accept that my thesis be made publicly available on the University of Malta Institutional Repository.

□ As a Doctor of Music student, as per Regulation 26 (2) of the Doctor of Music Regulations, I accept that my dissertation be made publicly available on the University of Malta Institutional Repository.

□ As a Professional Doctorate student, as per Regulation 55 of the Professional Doctorate Regulations, I accept that my dissertation be made publicly available on the University of Malta Institutional Repository.

Copyright Notice

Preface

During the course of this research I have co-authored the following related peer-reviewed publications:

Journal Articles:

1. R. N. Duca, M. K. Bugeja, "A task allocation scheme for heterogeneous robot teams in coverage control applications", IEEE Access, submitted September 2023.

Conference Papers:

2. R. N. Duca, M. K. Bugeja, "A multi-robot allocation scheme for coverage control applications with multiple areas of interest", *9th International Conference on Control, Decision and Information Technologies*, July 2023.

3. R. N. Duca, M. K. Bugeja, "Multi-robot energy-aware coverage control in the presence of time-varying importance regions", *1st Virtual IFAC World Congress*, July 2020.

4. R. N. Darmanin, M. K. Bugeja, "A review on multi-robot systems categorised by application domain", *25th Mediterranean Conference on Control and Automation*, Valletta, Malta, July 2017.

Abstract

Modern technology facilitates the versatile use of robotic systems in multiple facets of life. The increasing need for efficient automation propels research in multi-robot systems (MRSs), essential for tasks like surveillance and search and rescue missions. Optimally covering an environment requires a framework that drives the robotic team to cover multiple distinct areas in the environment, be capable of reacting to time varying features in the environment and in the team itself, and optimally exploit the heterogeneity in the robot team at all times.

The work presented in this thesis is a step in this direction. More specifically, this thesis presents the design, implementation and validation of a holistic framework tailored for the coverage control application using an MRS. Current literature on this research area focuses on solving one singular aspect that affects the coverage control of the MRS, such as introducing energy awareness in the control algorithm, or tracking the time varying features of the environment. However, to the best of our knowledge, to date no work in literature simultaneously addresses multiple factors that typify most real-life coverage control applications.

To this end, the novel modular framework proposed in this thesis exploits the heterogeneity in the robot team by optimally allocating robots to particular regions of relatively higher importance in the environment, according to the requirements of each region and the capabilities of the robots. Furthermore, the environment is smartly segmented into cells according to the energy reserves of the robot, its sensors' performance and its maximum velocity, and each robot is made to cover one cell. Finally, a novel energy-aware controller that accurately and efficiently drives the robots to their respective cells and keeps tracking them is designed. The proposed control law is shown to converge the MRS to a centroidal voronoi tessellation (CVT). All these algorithms are combined into a non-trivial modular framework that addresses various practical aspects for the coverage control of a given environment. Finally, comprehensive and realistic simulations are employed to test and validate each module and the complete framework.

Acknowledgements

Firstly, I would like to sincerely express my gratitude towards my supervisor, Dr Ing. Marvin Bugeja. A paragraph or a couple of sentences are nowhere near enough to express my thanks for his mentorship, assistance and unwavering interest in my research. He walked with me every step of the way, through a great number of hours-long meetings and in-depth discussions about the subject. Without his excellent supervision and mentorship, this work surely would not have been possible and I would not be where I am today.

Secondly, once again I cannot express enough thanks towards my husband, Adrian. I am lucky to have found the best partner in life, and also as my support system throughout my journey. He has not only provided me with the much needed emotional and psychological support, but I am also extremely grateful for the technical discussions we had throughout the years about my research. Such technical discussions have been eye-opening and they have enriched my work. For this, I am eternally grateful.

It would also be impossible not to mention and express my gratitude towards my family, especially my mother and my sister who always supported me in every step of the way, even when things got rough and the road ahead was unclear. My sincerest thanks extends to my in-laws and friends who have shared this journey with me for a number of years.

Finally, I would like to thank my colleagues and friends, Sanchia and Rosanne. There is no comparison to having colleagues who become friends and who are able to perfectly relate and support your Ph.D. journey. I would also like to express my sincerest thanks to my Head of Department, Dr Alexandra Bonnici, for her support, especially during the last months of my thesis, as well as all my colleagues from the Department of Systems and Control Engineering for their continuous understanding and support. To you all, I am grateful.

Contents

List of Figures

List of Tables

List of Acronyms

CBF	control barrier function	36
CVT	centroidal voronoi tessellation	iv
GNN	graph neural network	47
HRI	human-robot interaction	3
ILP	integer linear programming	52
MILP	mixed integer linear programming	52
MRS	multi-robot system	iv
MRTA	multi-robot task allocation	5
SRS	single robot system	2
UAV	unmanned aerial vehicle	6
UGV	unmanned ground vehicle	45
USV	unmanned surface vehicle	7

Mathematical Definitions

Notation	Description
p_i	vector robot position
V_i	Voronoi cell (polygon) as generated by the Voronoi diagram
M_{V_i}	scalar mass of Voronoi cell
C_{V_i}	vector centre of mass (centroid) of a Voronoi cell
$J_{V_{i,p_i}}$	polar moment of inertia around the seed generator p_i of a Voronoi cell
q	two-dimensional vector coordinate point in the environment, (q_x, q_y)
Q	bounded and given environment
$\parallel q - p_i \parallel$	Euclidean distance between points q and p_i
$\phi(q)$	probability density function representing a region of importance in Q, known a priori
$\mathcal{H}$	a scalar cost function that is minimized by Lloyd's algorithm
$\dot{p_i}$	vector robot speed, first derivative of p_i with respect to time
u_i	vector control input that drives the robots to their target destination
k_{prop}	proportionality constant in a proportional control law
w_i	weight of robot i used in a power diagram
E_i	scalar energy reserve (for example, fuel or battery level) of robot i
$E_{i(max)}$	scalar maximum energy capacity of robot i
$E_{i(min)}$	scalar minimum energy threshold by which robot i is considered depleted of energy
V_i^e	Voronoi cell generated using an energy aware environment segmentation module
C_{ij}	the scalar cost of assigning robot i to a probability density function j in the multi-robot task allocation algorithm
x_{ij}	selector vector in the Integer Linear Programming formulation of selecting the allocation of robot i to density j in the final solution
s_k	scalar sensor performance rating set by the user
$v_{i(max)}$	scalar maximum velocity of robot i
n_r	total number of healthy robots making up the team
n_d	total number of density functions (regions of interest) in the environment
$k_{i,e}$	tunable scalar constant parameter used in the proposed energy aware control law

To my husband

Chapter 1

Introduction

1.1 Overview

With the current advancements in technology, robotic systems have taken centre stage in our daily life. What was once academic research in robotic exploration and mapping is nowadays found in the mainstream robot vacuum-mop cleaners that are found in many households. Additionally, one can also see the shift to robotic automated systems in industrial environments, in the military, and even in the entertainment scene [2–4]. The demand to automate tasks using robotic systems highlights the importance to further the ongoing research in this field. Such academic research is then facilitated with the improvement in technology that makes these robotic systems feasible and easily attainable by the end customers.

One specific area of interest in the field of robotics is that of multi-robot systems (MRSs). Such systems are composed of a number of robots that may or may not be different in their physical composition and in the number and types of sensors that they have on-board. If the team consists of robots of the same type and having the exact same sensors on-board, then that team is called a homogeneous team, while if there are differences in these capabilities, the team is called a heterogeneous team. The applications of **NRS!**s (**NRS!**s) are vast and may span from a simple warehousing operation to a complex search and rescue mission. Using an MRS for tasks that are generally carried out by a team of humans has various benefits. Primarily, if the task is a dangerous one, automating it with an MRS reduces the potential risks and hazards posed to the human team. Additionally, the task may be completed more efficiently and in a more optimal

manner than if executed by a human team. This is because an MRS may be deployed on the scene very quickly and with the appropriate, state of the art algorithms on-board, optimization of the task may be achieved better and quicker.

This thesis focuses on a specific application of MRSs. The research topic is coverage control of a given environment. Coverage control is the algorithm applied in most surveillance or search and rescue applications. The aim of coverage control using an MRS is to spread the robots in the team across a given environment as much as possible, in an attempt to optimally cover this environment. For instance, in a search and rescue operation the environment might have multiple sub-areas within it that have a higher probability of hosting a victim. Therefore, the coverage algorithm steers an appropriate subset of the robot team to each of these sub-areas to maximise the mission success. Additionally, these sub-areas of higher importance might be moving in space or their importance might vary with time. For instance, an importance region in the sea during a search and rescue operation is susceptible to move by time due to the effect of sea currents and waves. Therefore, the algorithm must be smart enough to react to these variations to retain optimality.

The coverage control problem requires coordination and control algorithms such that the multi-robot team is able to cover a given environment accurately and efficiently. This task becomes more complex and less trivial when environment and system constraints are present. Such constraints may be manifested as the limited energy levels of the robots, the limitations in the sensory performance of the robots, having multiple sub-areas in the environment that are considered of higher importance than others, as well as the variations of these multiple sub-areas in the environment with time, among others. All of these aspects need to be factored into the design of the coordination algorithm that drives the robots to complete the coverage task.

1.2 Multi-Robot Systems

Any robotic system may be categorized into either a single robot system (SRS) or a multi-robot system (MRS). SRSs are generally less complex in nature in that they do not require any coordination or task sharing among the peers in the team. The complexity in such systems arises from the actual applications themselves, as well as the robot's

interaction with the environment, and possibly the human-robot interaction (HRI). On the other hand, MRSs have the benefit of redundancy and cooperation, but this makes the overall design more complex. An MRS requires a good coordination scheme, which allows the robots in the team to share the tasks among themselves in an efficient and effective manner. Coordination schemes involve two main aspects in an MRS: *task decomposition* and *task allocation* [5–8]. Task decomposition is the process of defining the details of the sub-tasks, which must be allocated to the robots in the team. Task decomposition must be effective such that the overall main task is performed efficiently and the process of task allocation is also optimal [7, 9]. Task allocation is a scheme that allows the team of robots to decide which sub-task is performed by which robot. This technique must be efficient and effective in that it must exploit the full capabilities of the individual robots for the benefit of the whole team and of the main task [10, 11].

Furthermore, good coordination requires good communication and hence, a solid communication infrastructure must be in place [12, 13]. Before choosing any communication scheme, the system's designer must decide on the architecture of the system. A multi-robot system architecture may be *centralized* or *distributed* (*decentralized*) [7, 10, 14]. In a centralized system, one of the robots in the team, or else a fixed station, acts as the master that instructs and distributes the tasks to the other members in the team. In such a case, communication is fairly simple since it a one-way communication between the master robot and each of the slave robots. In a *distributed* system, each robot in the team must decide which task to carry out and at which point in time. This makes the communication scheme more complex since all the robots in the team must be able to somehow communicate with one another with the scope of fulfilling the requirements of the overall task. Such communication may either be *implicit* communication or else *explicit* communication [7, 15, 16]. *Implicit* communication is when the robots communicate with one another by leaving some form of message to each other in the environment. This is inspired from natural colonies such as bees and ants, who communicate among each other by leaving informative traces in the environment. Furthermore, communication may be *explicit*, which involves a direct communication between the members of the team. This requires the setting up of a good communication network such that the robots are always within range of each other [7, 15, 16]. Furthermore, the strength of the coordination among the robots may vary across different applications [7, 17]. A team of robots

may be strongly coordinated [18–21], weakly coordinated [22–24] or not coordinated . This is designed in the coordination and communication protocol deployed on the MRS.

The type of environment in which the MRS operates may be either *cooperative* [24–26] or *competitive* [21, 27]. In *cooperative* environments, the robots work together in order to accomplish some global task. On the other hand, in a *competitive* or adversarial environment, the robots compete against each other (or teams of robots compete against one another) with the scope of fulfilling their own self-interest. For example, object manipulation and transportation requires cooperation among the team of robots while a robotic football match such as RoboCup is based on a competitive environment [7].

1.3 Multi-Robot Systems application domains

In early works, researchers observed natural systems, such as a swarm of bees, ants and even humans, to study how a group of individual entities can work together to perform a given task. The multidisciplinary nature of these early studies, eventually led to MRS being applied in several different application domains such as surveillance, search and rescue, foraging, exploration, cooperative manipulation, and transportation of objects, among others. This section reviews a number of prominent and recent research works that aim to address various problems appertaining to six main application domains of MRS. The review presented in this section has been published in [28], as follows:

It is good to note that the majority of review papers on multi-robot systems (MRSs) focus on classifying the most fundamental aspects of an MRS, such as coordination and communication. In [7] Farinelli et al. classify these MRS features into two dimensions. The first, termed the coordination dimension, deals with the different classes of cooperation schemes, such as whether the system is centralized or decentralized, strongly cooperative (i.e. following a strict protocol), or weakly cooperative, among others. The system dimension classifies the existing types of communication schemes and team decomposition attributes. Similarly, in [10] Parker classifies MRS according to their architecture, the heterogeneity in the team, the type of communication scheme adopted, and the different

types of task allocation schemes. In this work, Parker also briefly reviews some works according to their application domain. However, the latter is not an extensive literature review of such works. A similar reviewing approach that categorises works based on the foundation topics of MRS, namely coordination, task allocation and cooperation, is also adopted in [8, 14, 29]. Furthermore, in contrast to these works, in [30] Gerkey and Matarić focus on one particular aspect of MRS, namely multi-robot task allocation (MRTA), and propose a taxonomy of task allocation schemes. The same taxonomy is also used in [31].

Alternatively, a literature review may also focus on particular schemes forming part of the main components in an MRS. In [32], Bernardine Dias et al. focus on classifying what they call *market-based coordination approaches*, which is a type of MRTA scheme. Such coordination schemes require the robots to bid for the tasks that they are able to perform. Furthermore, detailed reviews in the area of swarm robotics are properly surveyed in [33–35], while those reviewing biologically-inspired research include [36]. The interested reader is encouraged to refer to these works for more details.

In recent years, considerable attention has been given to human-robot interaction (HRI). In this light, Goodrich et al. [37] provide an extensive review of the history of HRI, as well as a review of works that introduced HRI to the field of MRS. In this work the author identifies the main topics in the field of HRI and provides a list of challenges that could shape the future of this interesting research area. Similarly, Chen et al. [38], propose a review of the human factors in systems exhibiting HRI, focussing on human supervision of multiple robots, and maintaining the human operator's situation awareness while retaining authority in the decision-making process. Both Goodrich et al. and Chen et al. provide indicators to the current open challenges in this area.

Moreover, some review papers focus on specific problems in MRS, such as formation control of robots in a team. Guanghua et al. [39] review such works in light of the type of formation architecture, and the existing formation control strategies employed. In the latter classification the authors classify the reviewed approaches according to the following categories: behaviour-based approaches, leader-follower approaches, virtual structure approaches, artificial potential functions or graph-based approaches. In [40], the authors review the mathematical problem of flocking, together with existing flocking control strategies. Other problem-specific reviews focus on patrolling algorithms [41], robotic urban search and rescue [42], and autonomous underwater vehicles [35, 43]. The review

papers referred to in this section have been grouped in six broad categories in Table 1.1.

Table 1.1: Summary of review papers

MRS Main Principles	[7, 8, 10, 14, 29–32]
Swarm Robotics	[33–35]
Biologically-Inspired Works	[36, 44]
Human-Multi-Robot Interaction	[37, 38]
Problem-Specific Works	[39–42]
Autonomous Underwater Vehicles	[35, 43]

In contrast to the survey works mentioned above, in this section briefly reviews recent and notable works in MRS categorised according to the most prominent application domains of the field, as follows.

1.3.1 Surveillance and, search and rescue

Surveillance and, search and rescue applications have attracted considerable attention from the MRS community over the years. This is due to the relevance of such applications in daily life. Surveillance applications were initially reserved to patrolling or surveying indoor areas [45]. However, with the introduction of unmanned aerial vehicles (UAVs), researchers have expanded their study to include the surveillance of outdoor areas, such as areas far out at sea or along river banks [46–49]. One of the main challenges in these applications is that of *persistent surveillance* due to the fact that one-time coverage and exploration algorithms cannot be used directly to continuously patrol and monitor the same area [50]. Nigam et al. [50] propose a novel control policy for persistent surveillance that maintains optimal performance through a formally-derived and scalable heuristic method. In this work, the environment is divided into grid cells, where each cell is attributed with an age and the goal is to minimize the overall age of all the cells. A control policy called the *Multi-Agent Reactive Policy* is proposed to control the unmanned aerial vehicles (UAVs) performing surveillance. A similar approach is adopted in [51] where the authors solve the task of persistent surveillance through the *Vehicle Routing Problem*.

Task allocation is another challenge in such applications because the solution to this problem must be time-efficient. In [52], the authors make use of a market-based strategy where the robots bid for locations that must be surveyed. In contrast, Jeon et al. [53]

calculate costs for a set of tasks, and allocate the mission tasks to the robots according to these costs. The concept of multi-robot task allocation (MRTA) is reviewed in detail in Chapter 2.

Over the years, disasters such as the Fukushima nuclear accident in 2011, have enabled researchers to deploy advanced MRS in real-life applications, mostly for search and rescue. For example, in [48] the authors propose a cooperative scheme for a multi-robot team for the surveillance of shipwreck survivors at sea. Using satellite imagery the user can plan the mission waypoints, which are then followed by an unmanned surface vehicle (USV) carrying a UAV. When the USV arrives at the designated way-point, the UAV takes off and uses a grid-like search pattern and image processing to localize survivors.

Another example is that of Gregory et al. [54], who address a humanitarian assistance and disaster relief applicaton. In this work the authors focus on simultaneously evaluating the damage done to the environment, and localising the victims according to two types of goals, namely, goals established from prior maps, and dynamic goals established according to the sudden detection of victims. The novelty of this work lies in addressing unreliable autonomy and communication by modelling unknown travel costs in a dynamic variant of the *Capacitated Team Orienteering Problem* [55].

Moreover, in [56], the authors also address the humanitarian relief problem through an implementation of a heterogeneous robotic system. This system features land and marine mobile stations that are responsible for coordinating and supporting UAVs and fast-speed land or marine robots. The authors propose solutions to two coordination technical challenges. The first entails the localization and landing of the UAVs —achieved through the use of visual SLAM—and UAV battery replacement to mitigate the limited energy constraint, which is a very common problem in MRS. In this work the authors also propose a collaboration scheme for the team of UAVs based on dynamic communication, target identification and triangulation.

1.3.2 Foraging and flocking

The task of foraging is often synonymous with swarm robotics, which is inspired by natural colonial systems such as those of bees and ants. This is because very often, the decentralized team only requires implicit communication to cooperatively collect randomly distributed objects and transport them to a “home" location. In [57] Parker proposes

ALLIANCE, a fault-tolerant framework that assigns tasks to robots according to their motivation and capabilities —modelled through a behaviour set—to do these tasks. This framework was applied to the foraging task of cleaning up hazardous waste, where the task allocation and coordination among the team involved assigning the robots to move the waste or report back to the base station. In another behaviour-based approach [58], Schneider-Fontán and Matarić assign segments from a territory to each robot for clean-up and object collection, as opposed to the task assignment used in [57]. One challenge in this domain is the optimal sharing of navigational space. In [59] Lein and Vaughan propose an algorithm that reduces mutual spatial interference and exploits a non-uniform distribution of robots during foraging. To achieve this, the authors propose a technique named *adaptive bucket brigade foraging*, where the robots remain within a variable space distribution within the environment. Furthermore, since foraging is a task associated with natural systems, a number of works adopt *particle swarm optimization* (PSO). Particularly, Couceiro et al. [60] study the *robotic Darwinian PSO* under communication constraints in the team [60]. From these works one can conclude that in general, foraging is not implemented using complex explicit communication schemes. The preferred choice of architecture is often decentralized, in order to allow the team members to achieve the task with minimal interference between them.

The task of flocking, also called *shepherding*, is considered to be a trait of swarm robotics. Some works make use of behaviour-based models and the *generation of safe zones*, such that the members in the MRS can follow a direction and stay in line with the flock to maintain cohesion, but at the same time maintain enough distance between them to avoid collision [61]. In these cases, a decentralized architecture is often adopted. Moreover, in [62] Sakai et al. propose a novel flocking algorithm that treats all detected objects as obstacles, irrespective of whether they are truly obstacles or form part of the team. They argue that such a method limits the amount of information handled by the team since velocity information on neighbouring robots is not required. On the contrary, in [63] Gu and Wang propose a *leader-follower flocking* technique, which requires the followers to communicate with their neighbours to exchange information about the estimated position of the flocking centre. Moreover, in [64] the authors apply *reinforcement learning* together with flocking control to enable a decentralized MRS to learn how predators should be avoided while maintaining the connectivity required for flocking. Additionally,

in [65] the authors propose a control algorithm that allows a flock to navigate around obstacles. Similar to foraging, in flocking a trend is noticed in the use of a decentralized architecture. However, the complexity of the task increases in flocking applications since the team must not only coordinate to avoid spatial interference, but also to maintain connectivity among the entities.

1.3.3 Formation and exploration

In formation applications, the team of robots must maintain some strict arrangement while at the same time avoid obstacles in its path. This problem becomes more complex than flocking since an obstacle must be collectively avoided without any of the team members leaving the formation for a long while. A common solution is the *leader-follower approach* where a trajectory-planning algorithm is implemented on the leader robot and formation constraints—distances from the leader—restrict the followers to maintain formation around the leader [66]. Other works adopting leader-follower approaches include [67–69]. Recent research is also adopting *computational intelligence methods* in formation applications. In [70], Wang et al. solve the optimal formation problem by using a recurrent neural network. Shape theory is used to generate a set of feasible formations and the proposed optimal formation solution chooses the one that has the minimum distance from the initial formation. Alternatively, the work reported in [71] adopts fuzzy logic to achieve formation control. Additionally, the work in [72] adopts control schemes, such as *Model Predictive Control*, in order to establish formation in the team.

In contrast to formation, in exploration the robots in a team must distribute themselves in an unknown environment in order to explore the area effectively. The coordination of such a system brings about many challenges, particularly those related to connectivity and battery-life problems. In [73] Banfi et al. address the problem of communication constraint by proposing an exploration strategy under recurrent connectivity. The robots only need to connect to the base station if new observations are made, hence allowing the members to disconnect for long periods of time until new information is obtained. Moreover, Cesare et al. [74] address the problems of communication and battery life. They propose a state-based approach in which a robot explores and shares information only if it is within the communication range and its battery levels are above a certain threshold. If the robot does not have enough energy to continue exploring, it waits for

another robot to meet it, and then uses its remaining energy to relay the information to base. This *rendezvous solution* to overcome communication limitations is also used in [75].

Other works in this domain focus on exploration strategies that are tailor-made for a particular MRS. For instance, in [76] the authors propose a *circle partitioning method* that segments the environment into sections and assigns each robot to a particular sub-region. In [77] the authors use a *flooding algorithm* that aims at reducing the exploration time and minimize the overall distance travelled by the robots during exploration. A number of works even propose exploration strategies that emerge from a *graph-based approach*, such that optimal coordination can be achieved when having a known number of robots exploring an area [78].

1.3.4 Cooperative manipulation

The *box-pushing* problem has become synonymous with MRS and it has been studied in several early works [79–81]. In one of these works, namely that by Brown and Jennings [82], the authors implement a *pusher/steerer system* where the object is moved from one place to another by small mobile robots. The steerer robot is pre-programmed with a trajectory and the pusher robot exerts a force onto the object, such that the steerer can follow its programmed path by setting its heading. Hence, communication is evidently absent from such a system. Alternatively, in [83] Sieber et al. propose a novel linear state feedback controller to surround an object with a formation of robots in order to transport said object. In this work the authors propose a suboptimal control law similar to the linear quadratic regulator (LQR) approach, which is used to regulate the way that a group of robots form an assembly around an object to transport it. Other works which adopt this *formation control for object manipulation* and transportation include [84] and [85].

More recently, Amanatiadis et al. [86] propose the system AVERT which is used to extract and transport vehicles from a specific location. The novelty of this work is focussed on lifter mobile robots used in a system that applies the concept of trajectory planning (using the D* Lite method), obstacle detection, and makes use of intercommunication in order to exchange control and trajectory information with a command base. As opposed to the previously mentioned works, the approach proposed in the AVERT project requires a stable communication among the team members, which is often missing in traditional

pusher-steerer or formation-based cooperative manipulation algorithms. In view of the solutions presented in these works, one may also think of *foraging* as another solution to cooperative object transportation and manipulation, since this involves the collection and transportation of items to some base location.

1.3.5 Heterogeneous capabilities in a robot team used in applications

Heterogeneity in a team of mobile robots enables the team to handle complex tasks more efficiently and effectively by exploiting the benefits of the diverse capabilities of its members. In this work, heterogeneity is analysed from two perspectives, namely, *human-robot heterogeneity* and *heterogeneous robot teams*.

A **HCI!** (**HCI!**) may be present in a system where the human is supervising and commanding a team of multiple robots. In [87] Rossi et al. propose a scheme where the human operator communicates with a whole team of robots to assign tasks and specify the members which must perform each task through speech utterances. In this case this study focuses on how speech may be segmented to simultaneously address multiple robotic recipients. Similarly, Cacace et al. [88] apply an HRI to a search and rescue application. In their proposed scheme they exploit human gestures and speech to select a desired robot for a task in a nonverbal and implicit manner. Robot selection is estimated by a probability that a particular command given by the user infers a set of capabilities which the robot possesses. For instance, the command “take off" shall probably be directed to an aerial vehicle and therefore, given more information, the algorithm proposed by Cacace et al. can select which aerial vehicle needs to take off. In [89], the authors also propose a scheme that seeks to establish an effective cooperation between a human and a team of robots during navigation.

The element of heterogeneity in a robot team is even evident in the earlier works of MRS. The framework ALLIANCE, proposed by Parker [57], supports and exploits the inherent differences in a team of diverse robot platforms. This is achieved by tailoring the motivational behaviour model and behaviour sets according to the capabilities of the robots. Similarly, Gerkey and Matarić [90] propose a market-based planner system named MURDOCH for task allocation in a robotic team. In MURDOCH task messages are published over a network with a subject relating to the capabilities required from

a robot to perform that task, leaving only those capable robots to subscribe to these messages. In the framework ASyMTRe [91], Tang and Parker propose an algorithm that decomposes a general task into sub-tasks according to pre-defined schema which reflect the different capabilities of the robots. These schema are then connected to assign the subtasks to the team members. Furthermore, Jeon et al. [53] propose a scheme with leader/follower roles for robots which are meant to survey an area and act in the event that an intruder is detected. These roles are defined according to the ability of the robot. A mission is decomposed into tasks, which are then assigned to the robots whose capabilities make them the most adequate to perform them. Similarly, in [88] and [92] we see search and rescue MRSs, where robots with different capabilities, such as to provide aerial views [88], or extinguish fires and transporting victims [92], are assigned to specific tasks. Additionally, the element of heterogeneity is strong in ground-to-aerial vehicle cooperation as seen in [93–95].

1.3.6 Adversarial environments

In 1997 RoboCup was founded with the aim of promoting research in robotics and artificial intelligence. This competition has shifted some of the research attention onto MRS in adversarial environments, such as those found in soccer competitions or battlefields. In [96] Weigel et al. propose a novel approach that tracks the ball and the adversarial players, and at the same time it strategically coordinates the team. In the mission to win against the other team, a specific role requiring a number of skills, is assigned to each robot such that, each team member can then adopt adequate behaviour from a behaviour set. A similar approach is also adopted in [97], where Browning et al. propose a hierarchical architecture of *Skills*, *Tactics* and *Plays* to execute low-level actions, decide on the skills to use, and coordinate the activity among the team. Furthermore, this problem has also been solved by using *reinforcement learning* techniques, as has been done in [98] and [99]. Alternatively, MRS have also been involved in different adversarial environments, such as in battlefields. For instance, in [100] Zhang et al. use a *genetic algorithm* to enable robots to learn not to enter their adversaries' defence region, where they may be "killed". Similarly, this problem has also been studied and solved in [101] and [102]. Using a somewhat different approach, in [103] the authors look at the adversarial environment as being inherent to an auction system, where each member bids for a task to perform in a

mission.

1.4 Coverage control problem

As previously explained, the coverage control problem lends itself well in a variety of applications, such as in surveillance and search and rescue applications. This field of research has been vastly studied from different aspects that practically affect the coverage algorithm. However, such works mostly focus on one aspect in isolation. For example, there are works that focus only on how to preserve the energy reserves of the robots in a single module [1, 104–109], to address the differences in the health and actuator capabilities of the robots [110–113], to address the time-varying features of the environment [114–119], or to address the problem of having multiple sub-areas of higher importance in the environment [120–122]. However, to date, there is no framework that attempts to bring together all of these aspects into one framework that may be deployed on a real robotic team.

This thesis aims to address this open problem and propose a holistic framework that accounts for various practical factors in a coverage control problem, with the main aim of maximizing area coverage subject to several constraints. To do this, the coordination framework first allocates each robot to a specific area in the environment, according to the capabilities that *that* area requires. Then it segments the environment into cells, according to the current positions of the robots and according to the robots' energy levels, their hardware capabilities. Using a non-energy-aware controller, the robots are then driven to the centre of mass of these cells, such that coverage is maximized at all times even in the presence of faults. This process is repeated for the duration of the operation. The framework repeatedly checks the energy levels of the robots, such that if a robot is depleted of energy, the remaining robots in the team are re-allocated to the specific areas of interest in the environment to take up the workload of the robot that has stopped. The design of such a framework is modular to allow for modifications in the algorithms according to a specific application.

To this aim, the main novel contributions of this thesis may be summarized as follows:

1. The design, implementation and validation of the framework's architecture that integrates multiple practical aspects that typify the coverage control problem. This

architecture allows the framework to exploit the heterogeneity in the robot team (in terms of their energy levels and hardware capabilities). Furthermore, the framework strives to preserve the energy reserves of the robots as much as possible both in the area segmentation stage, as well as in the control algorithm that drives the robots to their target areas. The architecture of the framework also caters for the time varying features of the environment. The novelty in this framework is that there is currently no work in literature that combines all of these aspects together in one modular framework tailored for the coverage control problem.

2. The design and implementation of a novel algorithm that optimally allocates the robots to the specific sub-areas in the environment, according to the specific requirements of each sub-area subject to the capabilities of the robots in the team.

3. The design and implementation of a novel environment segmentation module that divides the environment into cells, where each cell is to be covered by a specific robot. The generation process of these cells is multi-faceted, meaning that various factors such as the robots' energy levels and their hardware capabilities are considered in this environment segmentation. These factors may be expanded and modified according to the needs of the specific coverage application.

4. The design and implementation of a novel energy-aware tracking controller whose aim is to drive the robots toward its allocated sub-area and track its movement with time, while at the same time attempt to preserve the energy content of each robot.

1.5 Conclusions

The rest of this thesis is organized as follows. This chapter introduced the vast area of multi-robot systems (MRSs) and provided preliminary detail about the main topic of research in this thesis. Chapter 2 provides an in-depth review of the existing works in the literature of coverage control. Additionally, Chapter 2 analyses the current existing gaps in this research area and lists a number of open problems that this work sets out to address. Chapter 3 discusses the design and implementation details of the novel allocation algorithm. This algorithm allocates robots to the specific sub-areas of higher

importance according to the specific requirements of each sub-area subject to the capabilities of the available robots in the team. Furthermore, Chapter 4 details the design and implementation of the module that segments the environment into smaller cells. Environment segmentation is a vital part of the coverage control problem since each cell that is generated for each robot represents the area that that robot is covering. By repeatedly segmenting the environment into these cells, and moving each robot to the centre of mass of its cell, one maximizes the coverage of the environment. However, in contrast to other works, the design of this module considers factors such as the energy levels of the robots, their actuation capabilities as well as their sensors' performance, among others, to ensure that a robot is able to cover well the area generated for it. Together, the allocation module and the environment segmentation module strive to exploit the heterogeneity in the robot team as well as preserve the energy reserves of the robots.

Chapter 5 presents the details of the design and implementation of the novel controller that drives the robots to the centre of mass of their generated cell. This controller is designed as a tracking controller to address the concept of having a time-varying environment, where the centre of mass of each cell is therefore also time-varying. In addition, this controller is aware of the energy levels of the robots such that it aims to preserve the energy reserves of the robots. Reducing the energy consumption of the robots contributes to a higher overall endurance of the team which is always of benefit in coverage control missions. In each of Chapters 3, 4 and 5, a set of realistic test case scenarios are presented and their results are analysed to validate the design and implementation of each of the modules proposed.

Finally, Chapter 6 combines all of these modules together into one general framework and provides the details of implementation of this framework. A set of test case scenarios are simulated to validate the overall design of the framework. Finally, Chapter 7 concludes this thesis and includes a discussion of how the presented work has pushed the boundaries of research in the field of coverage control using MRS.

Chapter 2

Literature review

2.1 Overview

The research documented in this thesis deals with the practical application of the optimal coverage of some environment using a multi-robot system (MRS). This chapter introduces the reader to the general problem of coverage control and the related mathematical formulations. To explain such a problem better, it is best to provide a context, such as the following. Natural disasters have gripped the world ever since its existence. The worst consequence of such events is most often rooted in the aftermath; the search for victims and survivors, and obtaining information regarding the disaster. The search and rescue problem is often time-critical and an effective solution is one that covers the entire environment efficiently, especially in areas with a larger probability of hosting victims or survivors.

Coverage control with an MRS is an effective way of dispersing autonomous robot agents in areas which are inaccessible, potentially hazardous or simply too vast for humans to cover. During the process, such algorithms constantly factor in time-varying features in the environment and in the robot team, as well as certain constraints in the team, such as limited communication between the MRS and the human operator, and sensor and battery limitations, among others. Such algorithmic solutions most often provide an optimal or a sub-optimal solution which is more effective than the strategies that a human team may come up with during a time of crisis. Even if the area of crisis is accessible to humans, deploying such an MRS equipped with a coverage control scheme that is aware of its surroundings and the limitations of each team member, may be more effective and

efficient than a 'manual' man-made strategy. This is because the robots may be equipped with numerous sensors that may detect subjects and objects undetectable to humans, and the autonomous algorithms are able to quickly optimize or re-optimize the tasks among the robots that are deployed on the scene.

For this purpose, this work focuses on finding a coverage control solution that is able to handle changes in the environment through a time-varying model of the important regions in the environment, while at the same time account for the energy, the sensors' and physical limitations of each robot within the multi-robot team. Furthermore, one of the most important benefits of such a solution is that it is robust to disturbances both internal and external to the MRS.

The rest of the chapter shall serve as a detailed technical survey of the techniques used in coverage control applications. Additionally, this chapter shall also present a critical literature review of the existing works in this field of research, highlighting the gaps in knowledge during the discussion of these works. This critical appraisal shall lay the foundations for the novel contributions proposed in this thesis.

2.2 Facility Location Problem

The facility location problem (FLP) is an optimization problem where a number of facilities must be optimally and strategically placed in a given environment such that some cost function is minimized. The aim of such an optimization problem is that the cost between these facilities and their customers is minimized while at the same time ensuring that these customers are maximally served [123–127]. Locational optimization problems find their application in various different disciplines; from the biological sphere to study and characterize the territorial behaviour of animals [125] to simply placing mailboxes in a city [126]. In robotics, the robots that must be placed in a given environment represent the facilities, and the cost that they must minimize is similarly structured to the cost of an FLP. This type of locational optimization problem is often called coverage control using a multi-robot system (MRS) [123–125, 128].

The fundamental problem of coverage control with a number of agents/robots is to optimally segment the environment such that coverage is maximized. Over the years, several researchers formally introduced map segmentation techniques that segment an

environment according to some criterion. In 1942 Stone and Tukey [129] formally present the "Ham Sandwich Theorems" where they state that given n objects which are floating in an n-dimensional space, there is a hyperplane that segments each of the objects in equal volume portions. Theodore Hill [130] introduced the "fair division" concept in 1983, which is a mathematical model that focuses on segmenting a piece of land fairly among a number of countries, subject to the constraint that each allocated share of land must be adjacent to the country it is allocated to. However, perhaps the oldest and most well-known map segmentation technique is that of Voronoi Tessellations, whose informal introduction has been traced back to Descartes in 1644 [131]. This technique was formally formulated and named after Georgy Feodosievych Voronoy in 1908 [132]. The general concept of the Voronoi Diagram is to cluster points around a seed point (also called a generator), such that all the points clustered around that seed point are closer to that point than to any neighbouring seed point. This results in an environment being segmented into regions, called Voronoi cells, where each region is composed of environment points closer to the seed site than any other seed. Such a map segmentation technique becomes fundamental in the formulation of the locational optimization problem [133].

2.2.1 Generalised Voronoi diagram

Due to its relevance and importance in multiple disciplines, Okabe et al. [128] present numerous variations of the main concept of the Voronoi Diagram in their book. To understand the relevance of Voronoi diagrams to the coverage control problem that is being addressed in this thesis, the reader is first familiarised with the general concept of Voronoi diagrams. Okabe et al. [125, 128] apply this concept to the facility location problem. The aim is to determine the location of a set of seed points such that that average cost to the nearest points is minimized [123, 125, 127, 134]. For example, suppose that a team is composed of n robots that must be distributed in a given environment Q. Each robot must be entrusted to cover a region which is the closest to it than any other robot, and thus minimize the distance travelled by each robot. In [128] Okabe et al. propose a solution to this problem by using generalized Voronoi diagrams to partition the environment Q. The generalized Voronoi partitioning can be described by Definition 2.1 [125, 135].

Definition 2.1. A Voronoi diagram is the segmentation of an area into polytopes called

Voronoi cells, $\mathcal{V} = \{V_1, ..., V_n\}$, computed as follows

$$\mathcal{V}(p_i) = \{q \in \mathbb{R} | f(\| q - p_i \|) \leq f\left(\| q - p_j \|\right), \forall j \neq i\} \tag{2.1}$$

where p_i represents the coordinates of the seed point, $\mathcal{V}(p_i)$ is the Voronoi region around that seed point, q represents each discrete point in the environment Q, p_j represents the seed point neighbour to seed i, and $f(\| q - p_i \|)$ is a general cost, called the performance function, that must be minimized through the clustering of points.

In the standard Voronoi diagram, this general cost is set as the Euclidean distance between each seed point and its closest points in the environment, that is $\| q - p_i \|$. For each seed point p_i there is an associated Voronoi cell such that all points in that cell are closer to that seed point p_i than any other seed point in the environment, as defined in (2.1). For example, in Figure 2.1, the discrete point q is found in the Voronoi cell of the seed point p_j since it is closer to p_j than it is to p_i.

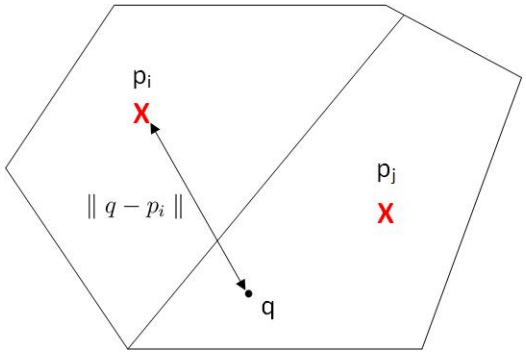


Figure 2.1: A simple example of Voronoi partitioning

The generalized Voronoi diagram may be modified to create special cases of Voronoi diagrams, whose importance becomes prominent when implementing particular features in coverage control. Sun et al. [136] propose a modification to the generalized Voronoi diagram where the performance function, $f(\| q - p_i \|)$, is set to the minimum time taken for the i^{th} robot to move from its position p_i to a point q in the environment. For this reason, the function f is replaced with $\frac{\| q - p_i \|}{v_{i\_max}}$, where v_{i_max} is the maximum velocity attainable by the i^{th} robot. In this work, Sun et al. argue that by segmenting the environment using this metric, they impose time constraints in the segmentation of the environment such that each robot is able to reach the area it is in in the shortest time possible. Therefore, the performance function f may be modified to give rise to variations

to the standard Voronoi diagram, such as *weighted Voronoi diagrams*, *power diagrams* and *range-limited Voronoi diagrams*. Each of these modifications have their own segmentation characteristics, that may not be suitable for all applications of coverage control.

2.2.2 Weighted Voronoi diagram

A weighted Voronoi diagram is one where the cost by which each point in the environment is evaluated against the seed point, does not only depend on the distance between q and p_i but also on an additive and/or multiplicative weight [137, 138]. Formally, this is described by Definition 2.2 [1, 135, 137].

Definition 2.2. A weighted Voronoi diagram has weights added to and/or multiplied with the distance between points q and the seed site.

$$\mathcal{V}(p_i) = \{q \in \mathbb{R} | w_1 \parallel q - p_i \parallel - w_2 \leq w_1 \parallel q - p_j \parallel - w_2, \forall j \neq i\} \tag{2.2}$$

where w_1 is called the multiplicative weight and w_2 is called the additive weight. Including a multiplicative weight in the generation of Voronoi cells may lead to Voronoi cells which are non-convex and which may have disconnected boundaries or holes within them. Additionally, an additive weight may result in Voronoi cells which are star-shaped and/or empty, as well as boundaries which may be hyperbolic arcs [1]. Thus, using such Voronoi diagrams may result in a higher computational load since the algorithm no longer necessarily deals with convex polytopes as regular Voronoi cells [1]. Nevertheless, Kim et al. [139] utilise multiplicative weights such that the Voronoi regions are segmented according to the maximum speeds of the robots, in a coverage control application. Such a solution is effective but may prove complicated and resource-heavy to implement in a general framework that requires analysis of the Voronoi cell boundaries. This is because the resulting environment segmentation ends up with arcs for boundaries instead of straight line polytopes. Similarly, in [140, 141] the authors adopt multiplicatively-weighted Voronoi diagrams to address the coverage empty spaces which emerge due to the limited ranges of the robots' sensors. In [142] Wang et al. also use multiplicatively-weighted Voronoi diagrams to create a network of mobile sensors that are restricted by their communication infrastructure. Similar to [139], the solutions in [140–142] also result in non-convex polytopes.

2.2.3 Power diagram

Similar to a weighted Voronoi diagram, a power diagram evaluates points according to the power distance between the seed point and a general point q. Formally, this is described by Definition 2.3 [1, 135].

Definition 2.3. A power diagram, also called the Voronoi diagram in Laguerre geometry, evaluates the power distance $\| q - p_i \|^2 - w$ between the points to generate the cells.

$$\mathcal{V}(p_i) = \{q \in \mathbb{R} | \| q - p_i \|^2 - w_i \leq \| q - p_j \|^2 - w_j, \forall j \neq i\} \tag{2.3}$$

where the weight w represents the footprint of the seed site. When the value of w increases positively, the region associated with the seed site that has that weight shall be bigger than of the other seeds with smaller weights. Therefore, the larger the value of w, the bigger the region generated will be. In the case of coverage control, the weight w represents the footprint of the robot, which may be associated to the physical limitations of a robot, such as its energy levels and actuator limitations. Power diagrams result in convex polytopes as Voronoi cells, however, the seed sites may not end up in their region [1]. Arslan et al. [143] employ power diagrams to combine the problem of collision avoidance and the coverage limitations in the robots' sensors. Similarly, Kantaros et al. [144] employ power diagrams to address the problem of sensor range limitations as well as the communication limitations in the multi-robot system (MRS).

2.2.4 Range-limited Voronoi diagram

A rather particular case of the Voronoi diagram is one that considers the robots to have a limited sensory footprint. In the other Voronoi diagram types, as described in Sections 2.2.1, 2.2.2 and 2.2.3, a robot has sensing capabilities that deteriorate with distance. As the distance between the seed site (i.e. the robot position) and a discrete point in the environment increases, the performance cost increases as well. Generally, this performance cost is directly related to the deterioration of the sensing capability of that robot. However, in all of these cases, the robot is expected to have an unlimited, albeit diminishing, sensing range. In [145], Cortés et al. introduce the effect of having a limited sensing range within the Voronoi diagram itself. They do so by considering the intersection between a sensing circle of some radius, R_i, with the environment when computing the

Voronoi diagram [135]. Furthermore, Li and Liu [146] focus on a scenario where the robots do not have the same sensing abilities. Hence, the maximum ranges of their sensors would not allow them to cover and sense the whole Voronoi partition from their current location p_i. The authors propose a modified partitioning approach to take these limitations into account to enhance coverage performance.

Similar approaches that use range-limited Voronoi diagrams for limited range sensors include [147–149]. Using a similar concept, Papatheodorou et al. [150] propose a distributed control law that takes into account the uncertainty in the robots' localization. This uncertainty is modelled as a limited sensing range around the robot. The result of this study is a Guaranteed Voronoi diagram, where the area is partitioned such that the points in a cell are guaranteed to be closer to p_i and not to p_j, depending on the uncertainty of the robots' locations. Therefore, if the robots have a high uncertainty in their location, they will have a smaller partition in which they are guaranteed to be in it. Although having range-limited environment segmentation ensures that the areas that the robots are operating in may be fully covered by their sensors, having a range-limited Voronoi diagram shall result in areas which are not accounted for in a coverage problem [135]. This may lead to gaps or holes in the environment that are not attended by any of the robots in the team.

2.3 Coverage control using a multi-robot system

The coverage control problem is an application of the locational optimization problem [133, 135]. To maximise coverage and place the robots optimally in the environment, Voronoi diagrams are used. The coverage control problem is typically formulated as follows [123,126]. Suppose that a bounded environment Q is defined by a convex polytope in a two-dimensional space. A probability density function $\phi(q)$ represents the probability that some event occurs over $q \in Q$. The location of the robots is given by the set $P = \{p_1, ..., p_n\}$. A function $f(\| q - p_i \|)$ is used as a measure of the sensing performance at point q from the robot having position p_i. This function deteriorates with the distance $\| q - p_i \|$. Moreover, the environment Q must be partitioned into n regions, given by the set of polytopes $\mathcal{V} = \{V_1, ..., V_n\}$, where each cell is assigned to its respective robot. Hence, the locational optimization problem is tasked with segmenting the environment

into n regions while minimizing the cost function in Equation 2.4. This optimization function $\mathcal{H}$, shall be minimized with respect to the robot positions P, as well as the assignment of the Voronoi regions V. When the local minimum of $\mathcal{H}$ is found, optimal coverage is achieved.

$$\mathcal{H}\left(P, \mathcal{V}\right) = \sum_{i=1}^{n} \int_{V_i} f\left(\| \, q - p_i \, \|\right) \phi(q) \, \mathrm{d}q \tag{2.4}$$

This means that the robots will be distributed optimally over the area of the environment that is highly influenced by the dominant region, $\phi(q)$. Cortés et al. argue that the optimal partitioning of Q is the Voronoi diagram, $\mathcal{V}(P) = \{V_1, ...V_n\}$ which is generated by the cell generators given by the robot positions $\{p_1, ..., p_n\}$. This is because inherently, the Voronoi Diagram partitions the discrete points in the environment, $q \in Q$, such that the distance between each point and the seed site is the minimum distance, compared to the other seed sites. The aim of the locational optimization problem is to minimize the cost $\mathcal{H}$. This goes in tandem with how the Voronoi diagram is partitioning the environment, since the same cost function is being used to cluster the points q in the environment segmentation. Therefore, using the Voronoi diagram shall contribute to the minimization of $\mathcal{H}$.

Cortés et al. [126] argue that the locational optimization function $\mathcal{H}$ shall have local minimum points that are not necessarily unique. However, these local minimum points shall coincide with the centroids of the Voronoi cells when a condition called *Centroidal Voronoi Tessellation* is achieved. Further treatment requires the following definitions with respect to a Voronoi region.

Definition 2.4. The mass of a Voronoi cell depends on the importance associated with each point in the Voronoi cell, according to the probability density function ϕ. This is given by

$$M_{V_i} = \int_{V_i} \phi(q) \, \mathrm{d}q \tag{2.5}$$

Definition 2.5. The centroid of a Voronoi cell is the center of mass of each cell, given by

$$C_{V_i} = \frac{1}{M_{V_i}} \int_{V_i} q\phi(q) \, \mathrm{d}q \tag{2.6}$$

Definition 2.6. The polar moment of inertia around the seed generator p_i of a Voronoi cell is given by Equation 2.7. Furthermore, by the parallel axis theorem, (2.7) may be

re-written as (2.8).

$$J_{V_{i,p_i}} = \int_{V_i} \| q - p_i \|^2 \, \phi(q) \, \mathrm{d}q \tag{2.7}$$

$$J_{V_i,p_i} = J_{V_i,C_{V_i}} + M_{V_i} \| p - C_{V_i} \|^2 \tag{2.8}$$

Theorem 2.1. *According to Cortés et al. [126] the local minimum points of $\mathcal{H}$ are found when the centroids of the Voronoi cells coincide with the robot location p_i. The location that acts as a site generator of a Voronoi cell, becomes equivalent to the centroid of that same cell. This means that the trajectories of the robots shall asymptotically converge to the set of critical points of $\mathcal{H}$ [135]. This is given by*

$$C_{V_i} = \arg\min_{p_i} \mathcal{H}(P) \tag{2.9}$$

Proof. If $f(\| q - p_i \|)$ in (2.4) is assumed to be equal to $\| q - p_i \|^2$, then using Definitions 2.4, 2.5 and 2.6, Equation 2.4 may be written as

$$\mathcal{H}(P, \mathcal{V}) = \sum_{i=1}^{n} J_{V_i,C_{V_i}} + \sum_{i=1}^{n} M_{V_i} \| q - p_i \|^2 \tag{2.10}$$

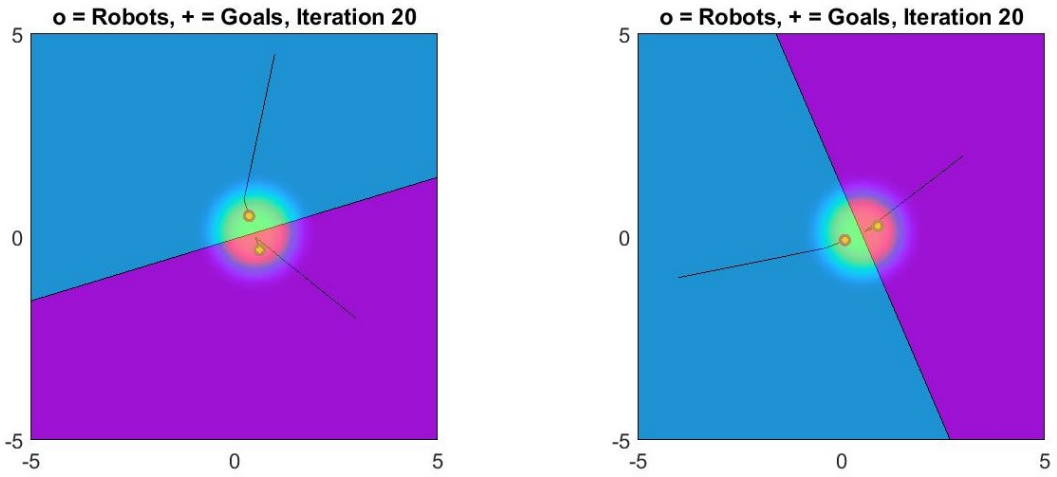
Since $J_{V_i,C_{V_i}}$ does not depend on the seed generator point p_i, then the partial derivative of $\mathcal{H}(P,V)$ with respect to p_i evaluates to

$$\frac{\partial \mathcal{H}}{\partial p_i} = 2M_{V_i} \left(p_i - C_{V_i} \right) \tag{2.11}$$

By evaluating $\frac{\partial^2 \mathcal{H}}{\partial p_i^2} = 2M_{V_i}$, it may be observed that the second partial derivative is positive. This means that the stationary point of (2.11) is a minimum value. By equating (2.11) to 0, it is evident that at the local minimum points, $p_i = C_{Vi}$. At this convergent stage, the final configuration is known as the centroidal voronoi tessellation (CVT), since the Voronoi cell seed site and the centroid of that same cell are the same. □

In addition, Cortés et al. [126] and Du et al. [133] also argue that the local minimum points are not necessarily unique. This is because the final Voronoi configuration always depends on the initial positions of the seed sites p_i [133]. This means that for a given density function, running the same algorithm multiple times with different initial robot positions will create different final CVT configurations, as shown in Figure 2.2. The dependency of the final CVT configurations over the initial positions of the robots may not affect coverage of an environment having a single density function, but it shall become

a bigger issue when the environment has multiple, distinct density functions. This is discussed further in Sections 2.4.4 and 2.6.



(a) Initial conditions Set 1 (b) Initial conditions Set 2

Figure 2.2: Different CVTs for two different sets of initial robot positions

2.3.1 The Lloyd's Algorithm

The final Voronoi configuration is achieved by following Lloyd's algorithm [151], as described by Cortés et al. in [126]. Lloyd's algorithm is an iterative algorithm that is a practical solution to Theorem 2.1. Suppose that the robots in the team have initial positions p_{i_0}. Then for each time step, Lloyd's algorithm computes the Voronoi diagram and generates the centroids of each Voronoi cell, according to Definition 2.5. Each robot position p_i acts as the seed site for the Voronoi diagram. Therefore, the centroid C_{V_i} is the centroid of the Voronoi cell that each robot is in. During the transient phase of Lloyd's algorithm, these centroids become the next destination of each robot, and hence the robot controller should be tuned in a way to drive the robot to the centroid of its Voronoi cell in time. By repeatedly computing the Voronoi diagram and driving the robots towards the respective centroids of their Voronoi cells, a centroidal voronoi tessellation (CVT) is achieved [126, 133, 152–154].

The number of iterations required to reach CVT may either be pre-set, or else a threshold is used to check by how much the robot poses change from one iteration to the next. If the change in the pose is below the threshold, the robots may be assumed stationary and hence that CVT is considered achieved. It is important to note that the

Lloyd's algorithm does not require the implementation to have a controller that drives the seed site to the centroid. Rather, this may be achieved by directly moving the seed site to the centroid in each iteration. In the case of coverage control with an MRS, since the seed is the location of a physical robot, it is fundamental to account for the dynamics of this robot. For the purpose of using the simplest form of robot dynamics, single integrator dynamics are considered. The controller is then necessarily required to drive each robot to its designated centroid according to the given dynamics. The pseudocode of Lloyd's algorithm in light of coverage control, is presented in Algorithm 2.1.

Algorithm 2.1. Lloyd's algorithm for coverage control

```
1: for num_iterations = 1: total_iterations do
2:     procedure VORONOI_BOUNDED(p_i, environment_boundaries)
3:         Generate the Voronoi cells according to (2.1).
4:         return voronoi_cells
5:     end procedure
6:
7:     procedure GENERATE_CENTROIDS(voronoi_cells)
8:         Compute the centroid of each Voronoi cell using (2.6)
9:         return centroids
10:    end procedure
11:
12:    procedure DRIVE_ROBOTS(centroids)
13:        Drive the robots towards their centroids using a controller of choice
14:        return final_robots_positions
15:    end procedure
16: end for
```

To show that using a particular controller the dynamic robots are able to converge to a CVT, Cortés et al. assume single integrator dynamics as follows,

$$\dot{p}_i = u_i \tag{2.12}$$

The controller used in [126] is a proportional controller given in (2.13), where k_{prop} is some positive gain. This means that the robot location p_i shall follow a gradient descent. By considering $\mathcal{H}$ as a Lyapunov function, the multi-robot system (MRS) stabilizes to a local minimum of $\mathcal{H}$ via dissipative control.

$$u_i = -k_{prop}(p_i - C_{V_i}) \tag{2.13}$$

Proposition 2.1. *Using a gradient descent control law, and for the closed loop system in (2.13), the location of the robots converges asymptotically to the set of centroids described by Theorem 2.1, such that CVT may be achieved.*

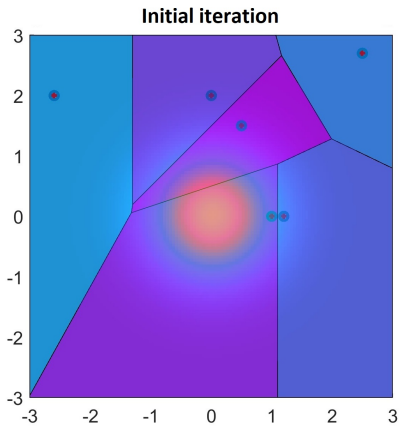
Proof. Consider $\mathcal{H}(P, V)$ as a candidate Lyapunov function. Then, under the control law in Equation 2.13 and by substituting (2.11), it can be shown that

$$\begin{aligned} \frac{d}{dt}\mathcal{H}(P,V) &= \sum_{i=1}^{n} \frac{\partial \mathcal{H}}{\partial p_i}\dot{p}_i \\ &= -2k_{prop}\sum_{i=1}^{n} M_{V_i} \parallel p_i - C_{V_i} \parallel^2 \end{aligned} \tag{2.14}$$

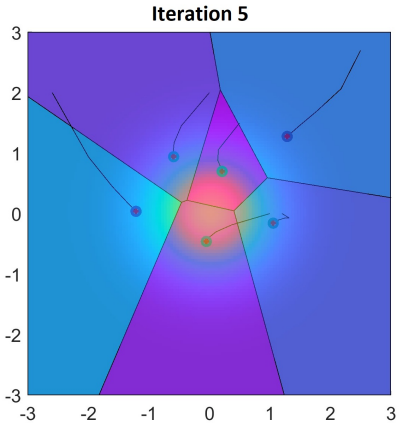
By LaSalle's Invariance Principle, the robot locations shall converge to the largest invariance set which are precisely the set of points $p_i = C_{Vi}$ at which the cost function $\mathcal{H}(P, V)$ has its local minimum.

Figure 2.3 illustrates the process the robots go through to arrive to CVT when one static density function is considered.

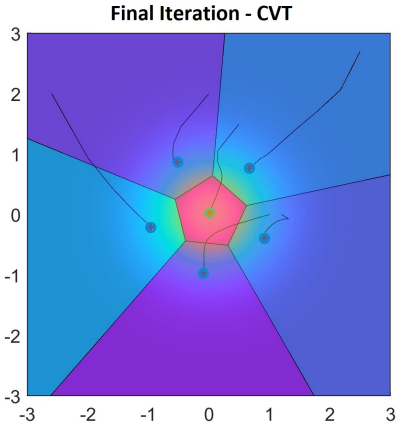
Through Lloyd's algorithm, if the controller ensures that the robots are able to arrive at the centroids as the target destination, then eventually CVT is achieved [155]. However, the proof as presented in [126] does not simply show that the controller is able to drive the robots to their centroids (hence achieving CVT), but it further shows that given the stated dynamics and with the proposed controller, the MRS is able to converge to a set of stationary points that minimize the coverage cost function, $\mathcal{H}(P, V)$. Other works [114, 116, 155, 156] design their controllers such that the robots are always driven



(a) Initial iteration



(b) Mid-trial at iteration 5



(c) Final iteration when CVT is reached

Figure 2.3: CVT achieved with Lloyd's algorithm using a proportional controller

to the centroids of their Voronoi cells. These works do not prove CVT by evaluating $\frac{\mathrm{d}\mathcal{H}}{\mathrm{d}t}$. Rather, they assume CVT is always reached as long as the controller always drives p_i to C_{V_i}. Therefore, these works argue that since the robots are repeatedly driven to these centroids, then a CVT is achieved. Although both approaches arrive at the same conclusions, the proofs presented in [114, 116, 155, 156] are based on the assumption and original proof [126, 133] that if a CVT is achieved, then the cost function $\mathcal{H}(P, V)$ is minimized. However this is not laid out in detail as it is presented in the proof by Cortés et al. [126]. The works in [114, 116, 155] shall be discussed in the succeeding sections.

2.4 Variations of the coverage control problem

In practical scenarios, distributing a number of robots in a given environment is not sufficient since robot and sensor limitations as well as time-variations in the environment shall impact the overall optimal coverage solution [126]. Hence, it is vital that such constraints are accounted for in the area coverage solution itself. One of the most pressing issues in the field of robotics is the limited battery life of the mobile robots [1, 105, 155, 157–159]. This requires the design of the overall coverage algorithm to be energy-aware in terms of how a given environment is segmented, as well in the control inputs given to each robot in order to execute its navigation task. Similarly, robots may also be burdened with actuator limitations [160] which may be intrinsic to the robot or may occur while the robot is performing the coverage task [113]. Intrinsic actuator limitations include weak motors that lead to velocity limitations [139] as well as friction losses in the gear train of the mobile robot [160]. Furthermore, a robot may be in the middle of executing a task when internal disturbances such as an actuator fault or running out of energy, may occur. Such disturbances must be immediately dealt with such that the rest of the fleet is able to carry on with the global task, making the coverage control solution fault-tolerant [57, 113]. In addition, each robot may be equipped with different sensors [161, 162], giving it different capabilities which may be suited to cover different parts of the given environment. In practice, these sensors are not infallible and hence, a more complete algorithm should be dynamic and fault-tolerant in order to deal with the changes occurring in the capabilities of each robot in real-time as well.

Faults occurring to a robot in a team are not the only disturbances that may affect the overall coverage task. A given environment may also undergo changes at the same time that a robot team is covering it. For example, during an off-shore search and rescue operation, the search area may easily change due to new knowledge about the possible location of the survivors, inclement weather, undercurrents or variations in the wind force and direction. Furthermore, a given environment may also have multiple separate areas which are considered as having a higher coverage importance. For instance, in a search and rescue operation following an earthquake, several separate areas known to have housed a school or a hospital or similar buildings, might be considered as having higher probability of hosting survivors than the surroundings. These areas may be modelled through multiple, distinct probability density functions [120]. The probability value represents the coverage

importance of the underlying area and therefore its 'weight' in the coverage problem formulation. Additionally, these areas of relatively higher importance might also change with time [116]. Therefore, the coverage framework must be equipped to track such time-varying modifications in the environment [115, 156, 163, 164].

A holistic and practical solution to the area coverage control problem should consider and cater to the above-mentioned practical constraints. For this purpose, a fault-tolerant framework is required such that the heterogeneous multi-robot team is able to optimally cover a given environment which has multiple areas of distinct importance that are also time-varying in nature. This framework is ideally a modular framework, such that it caters for advances in technology and the unique requirements of different applications. Sections 2.4.1, 2.4.2, 2.4.3 and 2.4.4 review existing works that address singular aspects of the coverage control problem as presented here. This critical appraisal lays the foundations for the proposed novel framework that holistically deals with an algorithm that enables a team of mobile robots to optimally cover a given environment, while accounting for all the practical factors outlined in this sub-section.

2.4.1 Energy-aware coverage control

The current energy reserve of each robot should be one of the main concerns of the coverage control algorithm. A practical coverage control algorithm should be aware of the energy levels of each robot both when segmenting the environment, as well as when generating the actuator signals that drive the robots towards the centroids of the respective Voronoi cells. In this thesis, the term *energy level* refers to the level of autonomy a robot has to continue moving. For example, an energy level may be the battery level of a robot or its remaining fuel. This is so because the type and energy source of robots in this work are kept generic. Having energy-aware algorithms is important because the energy level of a robot should be used with caution such that the robot energy source (typically finite in nature, for instance batteries or fuel) last longer during the coverage task [105].

Algorithmically, energy preservation in coverage control problems can be addressed on two levels. Firstly, the environment can be segmented such that a robot with low energy levels is not assigned to a large Voronoi region to cover. This is based on the assumption that the larger the area, the more energy the assigned robot is expected to spend to cover

the area. Secondly, a robot with low energy levels should not be given large control inputs (for instance high velocities) since these are directly related to the energy dissipation of the robot.

On the other hand, works such as [107–109] incorporate the notion of energy awareness in coverage and surveillance applications through a planning algorithm that optimally decides when to send the robots back to base for a battery recharge. However, this literature review is restricted to a survey of works that incorporate energy awareness in the design of the MRS control framework itself, rather than the planning algorithm that recharges the robots.

Kwok and Martínez [1] propose a novel approach that is energy aware both when segmenting the environment, as well as in the generation of the control inputs. Firstly, they choose to implement *power diagrams* such that the environment segmentation process is weighted according to the robots' energies, as described in Section 2.2.3. The authors choose to set the weight of the power diagram $w = E_i - E$, where E_i and E are the current and maximum energy level of the i^{th} robot, respectively. By setting such weights, the authors ensure that robots having lower energy levels are given a portion of the environment with a smaller Voronoi cell area. This follows from the fact that the power diagram ensures that if a point q is equidistant from two robots p_i and p_j that point shall fall in the region of dominance of the robot with the higher energy level. This is in contrast to what happens on a normal Voronoi diagram where the choice of place a point in the region of dominance of a robot depends solely on the distance of that point from the robots' positions. Furthermore, the authors propose the cost function $\mathcal{H}_e$ where the distance between the points $q \in Q$ and the robot position now has energy considerations.

$$\mathcal{H}_e(P, \varepsilon, \mathcal{V}^e) = \sum_{i=1}^{n} \int_{V_i^e} \Big(\| \, q - p_i \, \|^2 - (E_i - E) \Big) \, \phi(q) \, \mathrm{d}q \tag{2.15}$$

where ε is the set of energy-aware weights associated with the energy levels of each robot, $\mathcal{V}^e$ is the set of Voronoi cells generated according to these energy considerations, and $E_i - E$ is directly related to the amount of energy consumed by a robot. Kwok and Martínez [1] explain that if the maximum energy capacity of a robot is substantially large, then the amount of energy left in the reserve after movement would be comparable to the maximum level. This means that a robot would use a small amount of energy when comparing this to the maximum energy capacity. Therefore, if the robot has not used a

lot of its reserves during its movement for a finite amount of time, then the consumption of energy $E - E_i$ would become insignificant. This leads to the energy-aware cost in 2.15 becoming approximately the same as the non-energy aware cost in 2.4 and therefore minimizing both equations becomes almost equivalent. Hence, the bigger the value of E, the more the energy-aware cost approximates the non-energy aware cost. In addition, the scheme proposed by Kwok and Martínez [1] approach yields a final Centroidal Voronoi Tessellation which is more balanced in terms of the energy levels of the robots in the team. This means that the expenditure of energy among the robots is more balanced in an energy-aware algorithm.

Additionally, Kwok and Martínez also propose an energy-aware control law that drives the robots to the target centroids, as follows:

$$u = -k(E_i)(p_i - C_{V_i^e}) \tag{2.16}$$

In Equation 2.16 the function $k(E_i) = \frac{E_i}{E}$ may be regarded as a varying gain that is directly dependent on the ratio of the robots' energy level to their maximum energy capacity. This ensures that as the battery life of the robot diminishes, the control effort is restrained in order to preserve the remaining energy levels. Furthermore, Kwok and Martínez opt to model the energy dynamics, of robot i as follows:

$$\dot{E}_i = -u^2 = -k(E_i)^2 \parallel p_i - C_{V_i^e} \parallel^2 \tag{2.17}$$

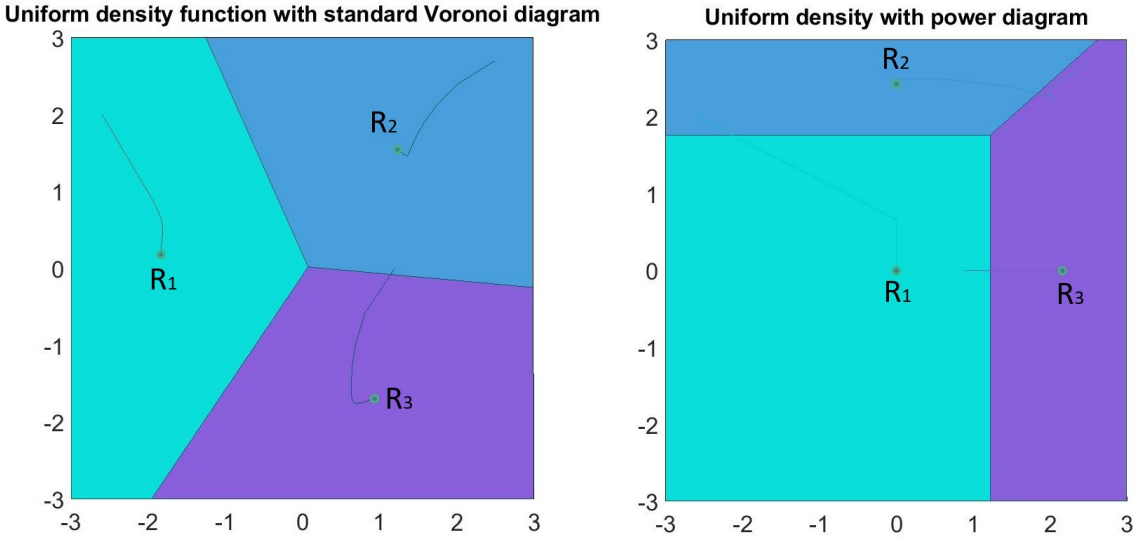
One may observe that the rate at which energy is spent is directly related to the amount of kinetic energy consumed while the robots are moving in the environment. This is because the rate of change of E_i is directly dependent on the velocity of the robot. In addition, this work [1] does not take into account time-varying importance regions, as it only considers a single, static, density function.

In the studies undertaken in this thesis, the algorithm proposed by Kwok and Martínez [1] was simulated. To simplify the analysis, a uniform density function is considered and three robots are used to cover this area. For this instance, the maximum energy level of all the robots is set to 10 units of energy. The robots' initial positions and initial energy levels are set according to Table 2.1.

To be able to analyse the effect of the energy aware algorithm in [1], the same ex-

Table 2.1: Initial conditions for comparison analysis experiment

Robot	Initial position	Initial energy level, in units of energy
R_1	(-2.6, 2)	10
R_2	(2.5, 2.7)	4
R_3	(1.2, 0)	6



(a) Standard Voronoi diagram for three robots covering a uniform density

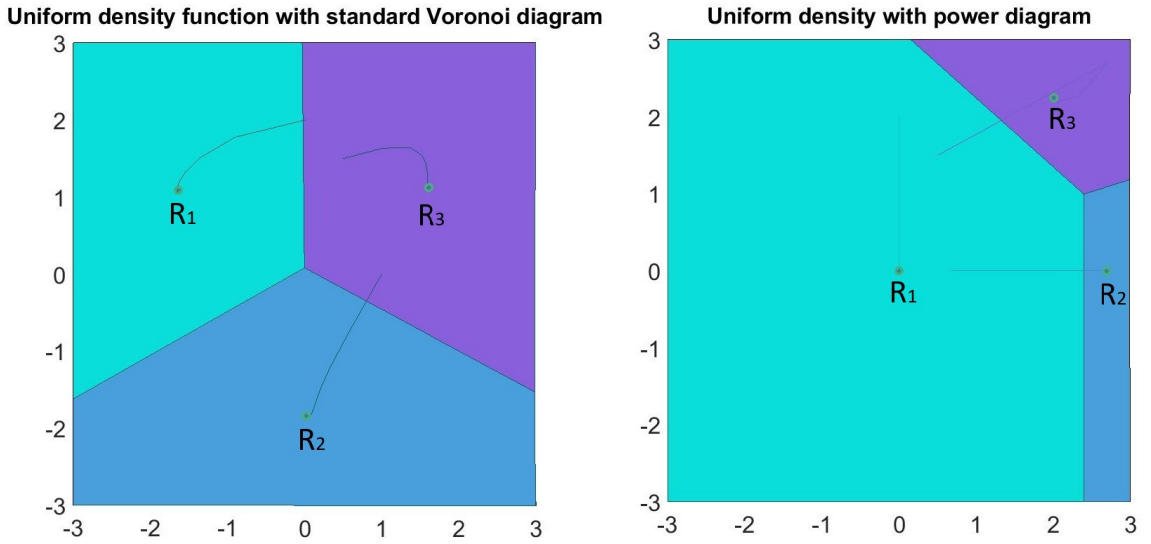
(b) Running Kwok et al.'s work for three robots covering a uniform density

Figure 2.4: CVT achieved with two different algorithms

periment was generated using Lloyd's algorithm with the standard Voronoi diagram as the environment segmentation algorithm and the proportional controller as the controller that drives the robots to their centroids. The constant of proportionality in the controller $k_{prop} = 1$ for this experiment. The resulting CVTs for each algorithm are illustrated in Figure 2.4. In addition, the areas of each robot cell at CVT, is recorded in Table 2.2. For the experiment that is running Kwok et al.'s algorithm [1], the final energy level of each robot at CVT is also tabulated since this is the metric that affects the area calculated at CVT.

Table 2.2: Resulting cell areas with the standard Voronoi diagram and with the algorithm proposed by Kwok et al. [1]

Robot	$E_i(T)$	Standard Voronoi area	$E_i(T)$	Kwok et al. algorithm areas
R_1	9.02	13.402	6.48	20.144
R_2	3.34	11.079	3.8	6.072
R_3	5.13	11.519	5.85	9.784



(a) Standard Voronoi diagram for three robots covering a uniform density

(b) Running Kwok et al.'s work for three robots covering a uniform density

Figure 2.5: CVT achieved with two different algorithms

By visually inspecting the CVT in Figure 2.4b, it is already evident that robot 1 that had a final energy level of 6.48 energy units has a larger area at CVT than the other robots when running the algorithm by Kwok et al. [1]. This is, of course, not the case in Figure 2.4a since the standard Voronoi diagram only segments the environment using distance between points as a metric. Hence, it is already evident that for better energy balancing of an environment, using a power diagram as done in [1] is beneficial, as opposed to using the standard Voronoi diagram. This is also confirmed through the calculation of polygon areas tabulated in Table 2.2. While with the standard Voronoi diagram one can observe that the areas generated are approximately equal, this is not the case when using the power diagram. In fact, when the power diagram was used, the areas generated are reflective of the final energy levels of the robots in the team. The same effect can be noticed when the robots depart from different initial positions as shown in Figure 2.5 and as tabulated in Table 2.3. This second set of results shows that the same effect is achieved irrespective of the initial positions of the robots.

Table 2.3: Resulting cell areas with the standard Voronoi diagram and with the algorithm proposed by Kwok et al. [1]

Robot	**Standard Voronoi area**	$E_i(T)$	**Kwok et al. algorithm areas**
R_1	11.245	9.26	30.131
R_2	13.524	3.6	2.468
R_3	11.227	4.25	3.402

Another observation made when using power diagrams is that since the more capable robots (i.e. having a larger weight w as defined in Section 2.2.3) have a larger generated cell, they end up intersecting with larger portions of the denser areas of the density function. This phenomenon is observable when a non-uniform probability density functions are used. A robot is considered to be more capable if, for example, it has a larger energy level than its team members. Therefore, a more capable robot is expected to have a larger weight in the power diagram such that its generated Voronoi cell is larger than that of its team members. Iteratively this pushes the more capable robots towards the denser parts of the regions. This is a beneficial side effect in coverage control because with this phenomenon, there is a higher chance that the more capable robots (such as having the highest energy reserve) end up in denser areas of the environment. This is beneficial because with the coverage control algorithm, reaching CVT is usually not the end goal. Rather, and especially in environments with a static density function, as the robots arrive at CVT, they would potentially need to perform a task while they are in their final generated cell. Hence, having the more capable robots in a denser area would mean that the task that they need to perform would be executed better.

Furthermore, Kwok and Martínez [104] further expand their work and incorporate the notion of energy levels in a range-limited Voronoi diagram scenario, as opposed to their previous work which focuses on segmenting the environment using power diagrams [1]. In the range-limited scenario, Kwok and Martínez directly use E_i to generate a sphere centered around p_i, with a radius E_i. Then, they use the presumption that each point within that sphere may be reached with the available energy that the robot has. The Voronoi diagram is thus created such that the admissible area of a robot in its Voronoi cell is that which intersects with the sphere of radius E_i. This shall then result in Voronoi cells which may be disconnected. This is a typical consequence of using range-limited Voronoi diagrams, as discussed in Section 2.2.4.

Conversely, Mei et al. [105] focus on the problem of deploying a team of robots by optimizing the total number of robots that should be deployed for the coverage task, as well as finding the appropriate initial locations of these robots in order to minimize the total energy consumption of the task. Furthermore, their approach focuses on preserving the energy of each robot by analysing the power consumed by each robot at different speeds. Their proposed speed management method maximises the distance travelled under energy and timing constraints. Moreover, their method is more focused on optimizing the number of robots in the deployed team as well as their initial locations. This is in contrast to preserving the energy of the robots while they are performing the coverage task. Furthermore, Zuo et al. [106] proposed a control law that considers energy constraints by creating upper and lower bounds on the control effort spent by the robots. In both these works, Mei et al. [105] and Zuo et al. [106] account for the energy limitations of the robots in terms of the control effort only.

Unlike the works proposed in [1, 104–109], energy considerations may also be embedded in a constraint optimization problem formulation [155, 159, 165]. Santos et al. [155] employ constrained optimization with control barrier functions (CBFs) [155, 166, 167] to minimize the control effort u. They propose the use of a CBF as the constraint function that ensures that the task of driving the robot to its target location C_{V_i} is performed. To be able to define a CBF, consider a dynamical system in control affine form as in (2.18). A definition of a CBF is thus given below.

$$\dot{x} = f(x) + g(x)u \tag{2.18}$$

where $x \in \Re^n$, $u \in U$ and the functions $f(x)$ and $g(x)$ are Lipschitz continuous vector fields.

Definition 2.7. A control barrier function (CBF) is a function $h(x)$ that satisfies the following condition:

$$L_f h(x) + L_g h(x)u + \alpha(h(x)) \geq 0, \forall x \in \Re^n \tag{2.19}$$

where $L_f h(x)$ and $L_g h(x)$ are the Lie derivatives of $h(x)$ in the directions of f and g, respectively, and $\alpha(h(x))$ is a locally Lipschitz extended class κ function [168].

Before providing details of how control barrier functions may be used in an energy-

aware coverage problem, a theoretical background of this concept is presented. Control barrier functions become particularly useful in safety-critical controllers [166]. A collection of works in literature [1, 105, 111, 126, 135, 145] focus primarily on obtaining asymptotic stability and use a Lyapunov function to show that a system is asymptotically stable. In control theory, asymptotic stability may not be sufficient without a component that enforces safety.

For example, in an automatic cruise control (ACC) application, a vehicle equipped with an asymptotically stable controller shall converge to a fixed cruising speed, which is expressed as the asymptotic stabilization of a set of speeds. This, however, does not mean that the vehicle is safe at this fixed cruising speed. If there is another vehicle whose distance from the cruising vehicle is too small, a stable cruising speed of the vehicle equipped with ACC may lead it to crash in the vehicle in front of it. Hence, the stable set of speeds must be further reduced to a subset of safe speeds which shall not allow the cruising vehicle to collide [166, 167, 169].

The use of CBFs in constrained optimization problems allow for this safe set of values to be enforced on a safety-critical controller, hence ensuring the survivability, that is the ability to remain active, of a robot [159, 165, 170]. Hence, the aim of CBFs is to add a constraint to the optimization control problem to avoid undesirable controller outputs.

The theory behind finding the safe set originates from Nagumo's Invariance Principle [171]. Suppose that C is a set of values of a state variable $x \in \Re^n$ for which a function $h(x) \geq 0$, that is, C is called the superlevel set of a continuously differentiable function $h(x)$,

$$C = x \in \Re : h(x) \geq 0 \tag{2.20}$$

Theorem 2.2. *Nagumo's invariance principle states that if a state variable x is on the boundary of the superlevel set of $h(x)$, then x shall remain in this set. Hence at the boundary of this set, C, $\dot{h}(x, u) \geq 0$.*

However, Theorem 2.2 is restrictive for values along the boundary of the safe set. Xu et al. [168] and Ames et al. [166, 167] extend this theorem to the cases where the state variable x is within the set. This gives rise to Theorem 2.3.

Theorem 2.3. *Given a dynamic system as described in (2.18), where $x \in \Re^n$ are the system states, $u \in \Re^m$ are the system inputs, f and g are locally Lipschitz, and given a set C as described in (2.20), then any Lipschitz continuous control input, u, such that*

(2.19) holds, makes the set C forward invariant and asymptotically stable [167, 168].

Santos et al. [155] summarize this as follows:

$$\begin{aligned} x(0) \in C \Rightarrow x(t) \in C, \quad \forall t \geq 0 \\ x(0) \notin C \Rightarrow x(t) \rightarrow \in C, \quad as\, t \rightarrow \infty \end{aligned} \tag{2.21}$$

Santos et al. [155] use the theory of CBFs and apply it to the coverage control problem where the probability density functions are time-varying and the resulting control algorithm is energy aware. They start from defining a function $J(x)$ whose stationary points are the CVT configurations, that is $p_i = C_{V_i}$. In [155], the natural choice of a function whose stationary points are the centroidal Voronoi tessellations, is as follows:

$$J(x) = \sum_{N}^{i=1} \frac{1}{2} \parallel p_i - C_{V_i} \parallel^2 \tag{2.22}$$

In their work, Notomista and Egerstedt [165] show that finding some control input u that minimizes $J(x)$ is equivalent to (2.23).

$$\begin{aligned} \min_{u,\delta} \parallel u \parallel^2 + |\delta|^2 \\ \text{s.t} - \frac{\partial h}{\partial x} u \geq -\alpha\left(h(x)\right) - \delta \end{aligned} \tag{2.23}$$

The solution of the optimization problem in (2.23), where $h(x) = -J(x)$ shall lead to finding a state x as a stationary point of the cost function $J(x)$. In [165] they also extend their approach as a decentralized scheme where each robot i has a set of neighbours defined by the set $\mathcal{N}_i$. For the decentralized case, they reformulate the optimization problem as follows:

$$\begin{aligned} \min_{u_i,\delta_i} ||u_i||^2 + |\delta_i|^2 \\ \text{s.t} - \frac{\partial J_i}{\partial x_i} u_i \geq -\alpha\left(-J_i(x)\right) + \frac{\partial J_i}{\partial t} - \delta_i \end{aligned} \tag{2.24}$$

where $J_i(x,t) = \sum_{j \in \mathcal{N}_i} J_{ij}(||x_i - x_j, t)$, α is an extended class κ function. Note that $J_{ij}(\parallel x_i - x_j \parallel, t) = \frac{1}{2} \parallel (x_i - x_j) - C_{V_i}||^2$, where the summation of $J_i(x,t)$ across all the robots represents the cost function that is minimized in a decentralized manner if each robot executes the control input in Equation 2.24. In this work Santos et al. [155] ensure that with a minimum control effort the robots may safely arrive to the centroids of

their Voronoi cells in a stable manner. Furthermore, if the robots satisfy the constrained optimization problem in Equation 2.24, the robots are able to achieve a CVT and also minimize the cost function J in a decentralized manner.

Similarly, Notomista et al. [159] expand on this work where they propose an energy-aware task allocation framework. In their work, Notomista et al. propose a framework that is able to generate minimum control efforts subject to task completion constraints. An optimization formulation evaluates the features of the robots and assigns tasks to these robots according to their capabilities. This task-to-robot mapping is encoded via CBFs where the cost function to be minimized is the control effort u_i, and the optimization constraints represent the different tasks that may be assigned to the robot. In this approach, Notomista et al. are merging the energy-awareness of the algorithm with the task allocation process. They claim that the proposed framework is said to be resilient because it runs in real-time and the task allocation is robust against changes in the robot capabilities. Furthermore, Notomista et al. present this formulation as a mixed-integer quadratic program, which is necessary in cases where the function to be optimized is a quadratic function. This means that without an approximation, this problem cannot be solved in polynomial time, hence, causing the formulation to form part of the NP set of decision problems [172].

Moreover, the framework proposed by Notomista et al. [159] does not tailor for some of the most practical requirements in a coverage control problem. For instance, the framework does not address the need to create the subtasks (the individual Voronoi cells to be covered by the robots) according to the capabilities of the robots themselves. In a coverage control problem, this may be achieved using power diagrams designed to segment the environment according to the overall capabilities of the robots. Furthermore, the framework in [159] is not modular, in that the generation of the control input and the allocation of the tasks themselves is formulated as one optimization problem with constraints. Therefore, such a framework suffers from the rigidity of a non-modular system. For instance, if an application requires a different allocation algorithm, the overall framework would need to be re-designed. Finally, the practical implementation of the algorithm proposed in [159] is not straightforward to implement as it requires the computation of complicated derivatives such as $\frac{\partial C_{V_i}}{\partial p_i}$. Additionally, since a complex optimization problem is being solved on every iteration to compute the control effort, the

high computational complexity of this framework leads to long computation time.

2.4.2 Time-varying regions of importance in coverage control

A region of importance may move with time, such that the robots would need to track it in order to maintain the CVT. For example, in an aquatic scenario, the aquatic area to be monitored and surveyed is often dependent on underwater currents and wind. This time variance in the environment is modelled via a time-varying density function. The time-dependency of $\phi(q,t)$ is also reflected in the cost function in (2.25) since $\phi(q,t)$ forms part of this cost. The cost (2.25) is being minimized as part of the locational optimization problem.

$$\mathcal{H}(P,t) = \sum_{i=1}^{n} \int_{V_i} \parallel q - p_i \parallel^2 \phi(q,t)\,\mathrm{d}q \tag{2.25}$$

By reformulating (2.25) using the parallel axis theorem, Kennedy et al. [114] show that by finding the derivative of (2.26), additional terms appear in the derivative that were not present in the derivative $\frac{\mathrm{d}\mathcal{H}}{\mathrm{d}t}$ when the cost $\mathcal{H}$ is not time-varying. These additional terms no longer ensure that this derivative is always negative, unlike the derivative $\frac{\mathrm{d}\mathcal{H}}{\mathrm{d}t}$ in (2.14) when a proportional controller is used. Hence, there is no longer a guarantee on the convergence of the closed-loop system. For this purpose, Kennedy et al. [114] argue that a new control law must be devised.

$$\mathcal{H}(P,t) = \sum_{i=1}^{n} \int_{V_i} \parallel q - C_{V_i} \parallel^2 \phi(q,t)\,\mathrm{d}q + M_{V_i} \parallel p_i - C_{V_i} \parallel^2 \tag{2.26}$$

One of the first works to address time-varying density functions in coverage control is that proposed by Cortés et al. [115]. Assuming first order dynamics as in Equation (2.12), they propose the feedback plus feedforward control law in (2.27).

$$\dot{p}_i = \dot{C}_{V_i} - \left(k + \frac{\dot{M}_{V_i}}{M_{V_i}}\right)\left(p_i - C_{V_i}\right) \tag{2.27}$$

This control law is inspired by the feedback plus feedforward closed loop system shown in Figure 2.6.

Following the block diagram in Figure 2.6, one observes:

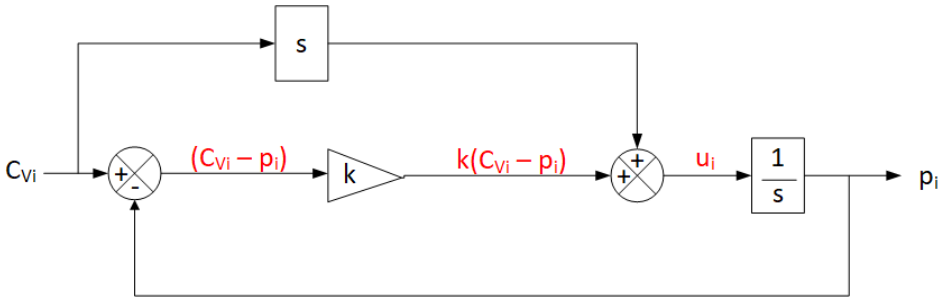


Figure 2.6: A block diagram of the feedback plus feedforward closed loop system

$$\begin{aligned} \dot{p_i} =& u \\ =& k(C_{V_i} - p_i) + \dot{C}_{V_i} \\ =& \dot{C}_{V_i} - k(p_i - C_{V_i}) \end{aligned} \tag{2.28}$$

Cortés et al. [115] modify this control law by introducing the term $\frac{\dot{M}_{V_i}}{M_{V_i}}$ in the proportional gain of the controller. This results in,

$$\dot{p_i} = \dot{C}_{V_i} - \left(k + \frac{\dot{M}_{V_i}}{M_{V_i}}\right)(p_i - C_{V_i}) \tag{2.29}$$

By introducing this term, Cortés et al. [115] ensure that the closed loop behaviour in (2.30) is obtained:

$$\frac{\mathrm{d}\mathcal{H}}{\mathrm{d}t} = \sum_{i=1}^{n_r} \frac{\mathrm{d}}{\mathrm{d}t} J_{V_i, C_{Vi}} - k M_{V_i} \parallel p_i - C_{V_i} \parallel^2 \tag{2.30}$$

To derive the expected closed loop behaviour, Cortés et al. [115] first use the parallel axis theorem to reformulate (2.25) as follows,

$$\mathcal{H}(P,t) = \sum_{i=1}^{n_r} J_{V_i, C_{Vi}} + M_{V_i} \parallel p_i - C_{V_i} \parallel^2 \tag{2.31}$$

Then to get the desired closed loop behaviour, the derivative of $\mathcal{H}(P,t)$ with time is computed as follows,

$$\frac{\mathrm{d}\mathcal{H}(P,t)}{dt} = \sum_{i=1}^{n_r} \frac{\mathrm{d}}{\mathrm{d}t} J_{V_i, C_{Vi}} + \dot{M}_{V_i} \parallel p_i - C_{V_i} \parallel^2 + M_{V_i} \left(p_i - C_{V_i}\right)^T \left(\dot{p_i} - \dot{C}_{V_i}\right) \tag{2.32}$$

By substituting the control law in (2.29), that includes the $\frac{\dot{M}_{V_i}}{M_{V_i}}$ incorporated with the gain, the closed loop behaviour becomes that in (2.30).

In their work Cortés et al. [115] assume that the density function ϕ is quasi-static (in practice, it is slow moving) and hence, they are able to show that with their controller, $\mathcal{H}(P,t)$ is minimized and the system converges to the centroidal Voronoi tessellations. This is so because under their assumption $\frac{\mathrm{d}}{\mathrm{d}t}J_{V_i,C_{Vi}} \approx 0$ and therefore, $\frac{\mathrm{d}\mathcal{H}(P,t)}{\mathrm{d}t} \leq 0$. By LaSalle's Invariance Principle, the largest invariant set of $\frac{\mathrm{d}\mathcal{H}(P,t)}{\mathrm{d}t}$ is $p_i - C_{V_i} = 0$, which are the Centroidal Voronoi Tessellations. Lee et al. [116, 117] and Diaz-Mercado et al. [163] argue that the assumption made by Cortés et al. might be too restrictive. To address this shortcoming in [116] they propose the control law (2.33), where $p = [p_1, \dots p_n]$ is the set of positions of the robots in the team, $c = [c_1, \dots c_n]$ is the set of centroids relative to each robot, and I is an $n \times n$ identity matrix.

$$\dot{p} = \left(I - \frac{\partial c}{\partial p}\right)^{-1} \frac{\partial c}{\partial t} \tag{2.33}$$

In [117] Lee et al. compare their proposed control law to that proposed in [115] and to the Lloyd's algorithm proportional control law 2.13. They show that for a slow moving density function, the difference between the three approaches is minimal, which conforms with the assumption that a density is quasi-static in [115]. However, as the speed of the density increases, the discrepancy between the approaches becomes more pronounced. Furthermore, although the works in [116–119] seek to address the limitation in [115], they still do not provide any guarantees of the convergence of the robot team when time-varying densities are present in the environment.

The control law in [116] was shown to exponentially converge to a CVT in [163], for time-varying densities in a centralized system. With this formulation, the authors point out a number of problems in the computation of the inverse matrix in (2.33), stating that this centralized method will not scale well. To improve on their approach, they extend this to a decentralized case. Due to the complexity in the computation of the matrix inverse, the authors approximate (2.26) with the truncated Neumann series. Moreover, this is all based on the assumption that there is knowledge of how the density varies with time, that is, $\frac{\partial \phi}{\partial t}$. Computing all the proposed line and area integrals may be computationally heavy and thus, a static domain of interest must be assumed to be able to compute $\frac{\partial C_{V_i}}{\partial t}$.

Xu and Diaz-Mercado [156] propose an improvement to the time-varying decentralized

algorithm proposed by Diaz-Mercado et al. [163] by adding terms to the approximation of $\frac{\partial c}{\partial t}$ to be able to capture the evolution of the time-varying domain. They also test and analyse the scalability of their algorithm by analysing its computational complexity. Moreover, the work in [156] still requires significant processing power in order to be able to compute the partial derivative of $\frac{\partial C_{V_i}}{\partial t}$. Furthermore, this work analyses the surface and line integrals present in the control law under uniform density, therefore there is no knowledge of how the approach would be affected with a generic density function.

The work proposed by Santos et al. in [155] also considers that the density function is time-varying. The authors argue that the approach proposed by Lee et al. in [116] cannot apply to a decentralized system because of the inverse matrix computation. They argue that by encoding the time-varying notion in the design of the CBF as it is incorporated in the constrained optimization problem, no approximations or assumptions on $\phi(q,t)$ or its rate of change are required. They consider that the cost that is minimized, $J(x,t)$ is time-varying since this depends on C_{V_i} that is inherently time-varying. By setting the CBF function $h(x,t) = -J(x,t)$, and by ensuring that the constraint in (2.24) holds, they ensure that if each robot executes the control input returned by the optimization problem, then $J(x,t)$ is minimized in a decentralized manner.

2.4.3 Robot constraints in coverage control

As discussed in Sections 2.2.2 and 2.2.3, a special case of the Voronoi diagram may be used for specific segmentation of the environment. Pavone et al. [110] employ power diagrams for equitable partitioning of the environment based on the workload of the robots. They formulate their problem such that the power diagram weights described in Definition 2.3 are expressed in terms of the workload of the robots in the team. Similarly, Pierson et al. [111] encode the robots' actuator health in the weights of a power diagram. These weights follow an *adaptation law* such that they are constantly updating according to the variations in actuator performance. The control law proposed by Pierson et al. [111] is also affected by the actuation errors of each robot Δ_i.

Furthermore, Marier et al. [112, 113] exploit the strength of the generalized Voronoi Diagram to segment the environment according to the robot health. The regions covered by the robots are generated according to the communication capabilities of the robots, their sensors health and whether a vehicle has become inactive enough such that it needs

to leave the team. Marier et al. define the cost J in (2.34) where they include the sensor health $h_{s,i}$ in the performance function $f(\| q - p_i \|)$.

$$\mathcal{J}(P, \mathcal{V}) = \sum_{i=1}^{n} \int_{V_i} h_{s,i} \| q - p_i \|^2 \phi(q) \, \mathrm{d}q \tag{2.34}$$

Marier et al. then make use of this cost as feedback to dynamically recompute the regions because the cost is used to generate these regions. For instance, if a robot has a sensor that developed a fault, the scenario is modelled to increase the cost due to that degraded sensor. The boundaries of the cells of the agent with the degraded sensor are pushed to shrink the cell overall. This shall in turn minimize the overall cost. As explained in Defintion 2.2, using the sensor-weighted performance function $h_{s,i} \| q - p_i \|^2$ to generate the Voronoi diagram shall make the boundaries of the Voronoi cells move according to the degradation of the sensor health.

Similarly, Sun et al. [136] employ a generalised Voronoi diagram where the performance function used to generate the Voronoi cells is a time metric that encodes the time taken for the robot at p_i to arrive to each point in the environment q. As previously discussed in Section 2.2.1, in this manner Sun et al. [136] ensure that the Voronoi regions are generated in such a way that the robot is able to reach each part of the region in minimum time. Similarly, this is also reflected in the work by Kim et al. [139] who devise an algorithm that segments the environment according to the different maximum speeds of the robots in the team. To do so, Kim et al. make us of multiplicatively weighted Voronoi diagrams. In this work, Kim et al. shift the focus from optimally covering an environment spatially to optimally covering an environment in terms of time. The main focus of this work is adjusting the coverage problem to become time-sensitive for time-critical applications. The drawback of such an implementation is that the Voronoi cells generated are not guaranteed to be convex polytopes and rather the Voronoi cells end up with arcs for boundaries. This makes the generation of Voronoi cells more difficult to compute and even to work with computationally in a general coverage control framework.

2.4.4 Multiple regions of interest in coverage control

The research field of coverage control is mostly studied for applications where there is only one area of importance, and hence one density function. To cover such an environment,

the multi-robot system (MRS) may consist of either homogeneous or heterogeneous robots [173, 174]. Advancements in technology and the introduction of different types of robots gave rise to in-depth research on how to exploit any heterogeneity in a team to further improve coverage of an area [74, 135, 144, 161, 175].

Abbasi et al. [120] argue that in a heterogeneous team of robots, the dynamical differences inherent to the team need to be addressed. They argue that existing works that use heterogeneous teams for coverage control [176–179] do not take into account the dynamical differences between UAVs and unmanned ground vehicles (UGVs). Not accounting for these dynamical differences may create challenges and problems in coverage control. For example, due to their nature and dynamics UAVs may be deployed and reach a destination faster than however UGVs may then be equipped with higher-performing sensors than the UAVs are able to support. Abbasi et al. [120] claim that the inherent differences between the robots in the team need to be reflected as well in the region of interest that they are assigned to in an environment where different regions of interest require different robot capabilities. Therefore the re-grouping of the robots into smaller teams should be based on the density function to which they are assigned, their dynamics and sensing capabilities. In this regard, Abbasi et al. propose the control law (2.35) which takes into account the potential differences in the coverage importance functions of two neighbouring robots,

$$\dot{p}_i = \frac{K_i}{2M_{V_i}} \left(2M_{V_i}(C_{V_i} - p_i) - \gamma_i\right) \tag{2.35}$$

where γ_i is computed as follows:

$$\gamma_i = \sum_{p_j \in \mathcal{N}_{p_i}} \left[\left(\frac{N_{ij}(N_{ij})^{\mathsf{T}} - I_2}{\parallel p_j - p_i \parallel} \right) \left(\frac{p_j + p_i}{2} \right) + \frac{1}{2} N_{ij} \right] \mathcal{G}_{ij} - \left(\frac{N_{ij}(N_{ij})^{\mathsf{T}} - I_2}{\parallel p_j - p_i \parallel} \right) \mathcal{L}_{ij} \tag{2.36}$$

and:

$$\begin{aligned} \mathcal{G}_{ij} &= \int_{V_{ij}^B} \parallel q - p_i \parallel^2 \left(\phi_i(q) - \phi_j(q)\right) \mathrm{d}q \\ \mathcal{L}_{ij} &= \int_{V_{ij}^B} \parallel q - p_i \parallel^2 q \left(\phi_i(q) - \phi_j(q)\right) \mathrm{d}q \\ N_{ij} &= \frac{p_j - p_i}{\parallel p_j - p_i \parallel} \end{aligned} \tag{2.37}$$

One must note that V_{ij}^{B} represents the border between two neighbouring regions V_i and V_j, p_j is the neighbour of p_i, ϕ_i is the density function associated with the robot at p_i, and similarly ϕ_j is the density associated with the neighbouring robot at p_j. In (2.36), $\mathcal{N}_{p_i}$ represents the set of robots which are neighbours to the robot at p_i, while N_{ij} is the vector normal to the shared boundary V_{ij}^{B}.

With the additional term γ_i in (2.36) Abbasi et al. [120] show that the robots converge to their respectively associated density functions. The authors also present simulation results that show the convergence of the cost function in (2.38).

$$\mathcal{H}(P, \mathcal{V}) = \sum_{i=1}^{n} \int_{V_i} \| \, q - p_i \, \|^2 \, \phi_i(q) \, \mathrm{d}q \tag{2.38}$$

In their approach, Abbasi et al. associated the density functions to the individual robots heuristically. They claim that their approach allows the heterogeneous team of robots to divide the environment among themselves based on their capabilities. This is based on a prior knowledge of which density each robot needs to cover during the operation. The novel concept of forcing a robot to be assigned to a particular density proposed in [120] has two main advantages. The first is that the team heterogeneity is truly exploited in that the robots drive towards the density that mostly requires their capabilities. Without such a concept, having a multi-modal single density function leaves the coverage of the environment totally dependent on the initial positions of the robots. This may result in some of the peaks of the multi-modal density being poorly covered. The second advantage of this approach is that it forces robots to move towards certain density functions, hence allowing all areas in the environment to be better served and covered than if the algorithm is heavily dependent on the initial robots' positions. An in-depth discussion about this concept can be found in Section 2.6.

Moreover, Abbasi et al. [120] claim that in (2.35) the first term drives the robots towards their respective centroids while the γ_i term takes into account the differences between the density functions especially for neighbouring robots. However, it is unclear why the influence of the neighbouring density functions should affect the movement of a robot through its controller. This is especially so since the authors emphasize the need to treat the densities as individual densities assigned to individual robots due to their heterogeneous capabilities. Having a robot be influenced by the density function of its neighbouring robot counteracts the necessity of having it focused solely on its own density

function, which is the novel concept introduced in [120].

Furthermore, the control law in [120] is proposed when static density functions are used. Additionally, the approach proposed by Abbasi et al. [120] does not take into account the energy limitations of the heterogeneous team. Finally, although the approach by Abbasi et al. is shown to drive the robots towards their assigned density functions, this work does not investigate the computational complexity and time complexity of this approach. Equation (2.36) is rather complicated and complex to implement and therefore may not scale up well with an increase in densities and robots.

Gosrich et al. [121] adopt graph neural networks (GNNs) as an effective tool to drive the robots to distinct areas of higher importance in the environment. The focus of their work is to explicitly share and communicate information about the environment among the robot team. Inter-robot communication about the environment leads to better coverage such that robots learn more about their environment and are able to reach the multiple regions of interest. The work proposed by Gosrich et al. [121] brings about an important question. They claim that with Lloyd's algorithm, one cannot achieve an effective coverage solution in a multi-density scenario, since robots may cluster around one importance area only and leave the other areas of interest uncovered. The question that arises revolves around the reason why with Lloyd's algorithm, the robots do not gravitate towards the individual areas of interest in the environment. This happens because the way that the environment is segmented when using Lloyd's algorithm is dependent on the initial positions of the robots, as shown in Figure 2.2.

During each iteration of the algorithm, the target centroid of the robot would be that within its Voronoi cell. If the initial positions of the robot are in such a way that the centroids of the robots pull them towards one of the density functions in the environment, the team ends up covering just that density. The rest of the environment shall not be effectively covered. This brings out the importance of assigning and associating a density function with a robot a priori, such that the centroid calculation of the robot is always relative to the density function that it should be covering.

A different approach to the conventional coverage control with multiple regions is that adopted by Li et al. [122]. Li et al. propose an approach that aims to rectify the research question posed by Gosrich et al. [121], i.e. that the conventional coverage control problem is dependent on the initial conditions of the robots and hence, coverage of an environment

that has multiple importance regions may not be efficient since some importance regions may be only serviced by one or no robots at all. The approach proposed by Li et al. [122] first decomposes the environment into weighted square cells, where each cell represents a task. This decomposition is then translated into a node graph in preparation for a combinatorial optimization-based task allocation algorithm. They assume that the robots in the team are heterogeneous in terms of their velocities and therefore, a robot with a higher velocity should be allocated more tasks. After they allocate the robots to the set of tasks, the algorithm proposed by Li et al. [122] performs a cyclic region-growing operation through a set of single-objective combinatorial optimization task allocation algorithm. In other words, the environment is segmented into cells each representing a task, and through an objective function, these cells are optimally grouped together for each robot, according to its capabilities. This operation shall be executed iteratively, since the initial execution may lead to a region of robots assigned with too many tasks and other regions with too few tasks.

Although the authors show that their algorithm is able to work in an environment that is segmented into 300 by 300 cells, with 300 operating robots, the proposed approach needs a fine-tuning algorithm after every run of the cyclic-growth algorithm. The authors claim that this fine tuning algorithm is necessary due to the random initial placement of the robots in the environment practically this is the same problem mentioned by Gosrich et al. [121]. However this approach may be very inefficient when implemented on a practical system. For example, if an application is time-critical, having to grow the regions using multiple single-objective combinatorial optimization problems takes time, and having to fine-tune the result of this cyclic process would add to the overhead of the scheme. This shows that although this approach attempts to steer away from the dependency on the initial positions of the robots in coverage control, it does so with an algorithm that requires iterative steps to arrive to an optimal placement solution. Additionally, this approach does not consider that the different weighted square cells in the environment may require specific robot capabilities, and only addresses the robots' heterogeneity in terms of their maximum velocities.

2.5 General framework for coverage control and task allocation

In general, from the works presented in literature, it is clear that a complete coverage control scheme must consist of a number of modular components. An important component in coverage control is of course, the intelligent environment segmentation module. Moreover, if the environment consists of multiple regions of interest, the robots need to be assigned to these density functions intelligently, possibly based on their capabilities and the robots' characteristics. This asks for an algorithm that is able to match robots to density functions according to the heterogeneous capabilities of the robots and the requirements of the densities. Following this process, the environment is then segmented appropriately, using a method such as Voronoi diagrams. After segmenting the environment and generating the target locations (coinciding with the centroids of each Voronoi cells), the dynamic controller must then come into play to drive the robot towards its centroid. The processes of segmenting the environment and also driving the robots towards their centroids need to address a number of factors, such as the team's energy limitations, sensor capabilities and the robot's type. In this light, a completed modular framework for a coverage control problem is a neat and effective approach to optimally distribute a heterogeneous team of robots in a time-varying environment with multiple regions of interest. To the best of the author's knowledge, frameworks of this kind, that address coverage control in presence of the mentioned practical factors simultaneously, have not yet been proposed in literature. In the next section, the existing frameworks that attempt to solve other specific problems in the field of multi-robot systems (MRSs) are reviewed.

2.5.1 Multi-robot task decomposition frameworks

Applications that adopt multi-robot systems (MRSs) require good coordination among the members of the robotic team. One way to enforce this coordination is through task decomposition and task allocation [180]. The aim of decomposing tasks into sub-tasks in preparation for task allocation has been researched for different applications [181–184]. The main objective of these works is to start with one global task and then segment it into smaller local sub-tasks. These sub-tasks then need to be allocated to the different members in the team.

If this structure is applied to the coverage problem, the global task would be the overall coverage of the given environment, while the local sub-tasks can be 'optimal' coverage of the different regions in the same environment, each having its own specifications and requirements. To perform task decomposition in robotics, Tang and Parker propose a framework called ASyMTRe [91]. This framework decomposes each robot's capabilities into four schemas: environmental schemas, perceptual schemas, communication schemas and motor schemas. Depending on the available schemas on each robot, the algorithm proposes solutions as to how these schemas should be grouped and connected in order to decompose a global task into smaller tasks that can be handled by the individual robots according to their abilities. This framework was later extended to IQ-ASyMTRe where this coalition among the robots also factors in the sensor constraints [185]. Furthermore, in many works task decomposition frameworks are moulded for particular applications, such as coverage of an area or surveillance. In such cases, the global task would be to survey or monitor an area, and this task is typically decomposed into smaller areas which are assigned to each robot as a sub-task [181, 182, 184, 186].

2.5.2 Multi-robot task allocation frameworks

After decomposing a global task into smaller, and more manageable sub-tasks, the system must match these sub-tasks to the appropriate robots in the team. In literature, this problem is referred to as multi-robot task allocation (MRTA). In MRTA schemes, the sub-tasks generated by the task decomposition algorithm are assigned to the possibly heterogeneous members in the team, in a way that fully exploits each robot's capabilities [187, 188]. The MRTA problem might also be prone to situational and environmental factors, such as a fault developing in one of the robots, or a change in the environment importance regions. This makes the decision process a dynamic problem that must be solved iteratively with time [189]. The problem of assigning T tasks to R robots, according to the capabilities of the robots, may be modelled using different methods.

Khamis et al. [31] categorize these models into four main categories. The first category, is the Discrete Fair Division problem [190]. In such model, N robots perform a share of the set of tasks S, such that the amount of work required by the set S is divided among N robots fairly. To ensure discreteness in the sharing of the tasks, the robots shall either secretly bid for a particular task (method of sealed bids) [191] or else use

the method of markers [192]. In the method of markers, the tasks are first lined up and then each robot segments the tasks into what they consider to be a "fair share" from their perspective, by placing markers that divide the tasks into shares. The algorithm then starts to allocate shares to the robot that has the leftmost marker placed in the task line-up. On allocation, that robot is removed from the set and the algorithm repeats by allocating the next fair share to the robot whose marker was placed in the leftmost place.

Another modelling approach for MRTA is the optimal assignment problem, where robots are assigned to tasks such that some profit objective function is maximized [193]. A third modelling approach is the Multiple Travelling Salesman Problem (mTSP) [194,195]. Similar to optimal assignment problem, the mTSP follows a combinatorial optimization formulation where in this case, an objective cost function is minimized, subject to a number of constraints. Finally, Khamis et al. [31] present the ALLIANCE Efficiency Problem (AEP) as a fourth modelling category for MRTA. AEP is a singular-objective optimization problem [196]. With this method, a robot, having a number of skills and capabilities, tries to conquer a task that through its resources shall contribute to the robot's survival. The alliance aspect of this method stems from the fact that robots must form alliances to be able to improve their skills and resources [196].

Parker [57] proposed a framework entitled 'ALLIANCE' that attempts to use behavioural skills and resources to allocate tasks to robots. Parker [10] argues that complex tasks are better handled by a team of robots which can distribute sub-tasks among the agents. Performing such sub-tasks in parallel could also lead to more efficient problem-solving techniques, and having multiple robots increases the robustness and fault tolerance of a system through redundancy. ALLIANCE [57] is one of the earlier task allocations frameworks that attempts to address fault tolerant coordination in a multi-robot team. This is a behaviour-based, fault-tolerant architecture where the robots possess several behaviour sets that are activated according to whether they are motivated to perform a particular task or not. This motivation is triggered by a metric called the *motivational behaviour* that depends on the robot's impatience to perform a task, and the robot's acquiescence. This work is limited by the fact that robots must maintain strong communication links among them, and the fact that the behaviour parameters must be known a priori. The latter limitation has been addressed by the framework L-ALLIANCE that allows the robots to learn their own behaviour sets [197]. Other frameworks that seek

to be fault tolerant in performing multi-robot tasks by employing behaviour models include [198–200].

MRTA schemes and formulations vary according to the nature of the problem. The problem may consist of a single task or of multiple tasks that must be executed in parallel; having a singular robot or multiple robots. It may also be defined according to whether the assignment of tasks is instantaneous, and does not consider dependencies between tasks, or time-extended, such that it considers all the available tasks and robots, together with their requirements and capabilities, [201]. Furthermore, given the scheme characteristics, two most commonly used MRTA approaches are market-based approaches and optimization-based approaches.

A market-based approach is an economically-inspired approach that involves explicit communication among the team members while they participate in an auction for tasks. Each robot in the team places a bid on a task and the negotiation process the seeks to optimize an objective function based on the robots' capabilities for the tasks available [202]. Gerkey and Matarić [90] propose a market-based framework called MURDOCH whereby robots bid to perform particular sub-tasks. In this framework, tasks are allocated to the agents in the team according to their capabilities. Like ALLIANCE, MURDOCH relies heavily on stable communication links between the robots in the team. Similarly, the framework called M+, proposed by Botelho and Alami [203] is a framework that employs actions that can be performed by each robot to perform a task that they negotiate for with the other team members. Other market-based MRTA approaches include [204–208].

On the other hand, optimization-based approaches perform the robot-to-task assignment by minimizing some cost function, subject to a number of constraints [31]. Generally speaking, MRTA optimization-based approaches do not scale up well in terms of the number of inputs to the problem against the computational complexity of the problem execution [209, 210]. Such optimization-based approaches may be deterministic — such as integer linear programming (ILP) or mixed integer linear programming (MILP) formulations [211, 212], or stochastic — such as meta-heuristic algorithms like Particle Swarm Optimization [213, 214], genetic algorithms [215, 216], and ant-colony optimization [66, 146, 217]. The rest of this section presents the reader with a detailed review of deterministic optimization-based MRTA approaches, because they are the most related to the novel task allocation scheme proposed in a subsequent chapter.

The field of combinatorial logic offers tools which are very well suited for the task allocation problem. This is substantiated through the use of combinatorial logic formulations for processor and resource allocation in computer systems [218, 219], as well as through its implementation in power management strategies [220]. For instance, depending on the application, one may adopt a variation of the knapsack problem [221] that assumes that different knapsacks represent the different tasks to be matched to the robots in the team. Inspired by such approaches, some multi-robot task allocation (MRTA) frameworks [222–228] are formulated as an integer linear programming (ILP) problem. More specifically, ILP is the optimization problem formulated as:

Definition 2.8. The integer linear programming (ILP) is an optimization problem where the decision vector variable $\boldsymbol{x}$ is an integer value in task allocation. The decision variable decides upon whether a candidate is chosen or not as part of the final problem solution. In ILP, the objective function is minimized or maximized subject to a number of constraints. The objective function is made up of a summation of costs $\boldsymbol{c}$ multiplied with the integer decision variable $\boldsymbol{x}$.

$$\begin{aligned} \text{maximize } & \boldsymbol{c}^{\mathsf{T}}\boldsymbol{x} \\ \text{subject to } & A\boldsymbol{x} \leq b \\ & \boldsymbol{x} \geq 0 \\ & \boldsymbol{x} \in \mathbb{Z}^n \end{aligned} \tag{2.39}$$

where $\boldsymbol{c} \in \mathbb{R}^n$ is a flattened vector that groups the cost values of each decision, and $b \in \mathbb{R}^m$ and $A \in \mathbb{R}^{m \times n}$ are the constraints vector and matrix, respectively.

ILP problems may be expanded as mixed integer linear programming (MILP) problems, such that there is a restriction on some, but not all, components of $\boldsymbol{x}$ to be integers. If one considers the knapsack problem as an example, this would mean that part of an object is allocated to a knapsack while other objects are allocated as a whole to the same knapsack. This mixed-integer notion is encoded within the MILP constraints. Furthermore, ILP-based MRTA formulations attempt to minimize some objective function, which is usually tied to the costs of having each robot performing a particular task, subject to certain constraints. Such problems usually set the optimization variable as binary, meaning that a robot-to-task mapping is either accepted or rejected by the solver.

In their work, Carlo and Kumar [222] develop a cost matrix with each element representing the cost of having some robot i performing some task j. The authors then propose an objective function as the summation of the total time T taken for the robot to finish the task, and the amount of energy E dissipated for that task. This objective function is then minimized subject to the following constraints: there is only one task allocated to each robot (the first constraint in (2.40)), a robot is assigned to complete one task only (the second constraint in (2.40)) and that the value of the task allocation decision variable $\boldsymbol{x}$ is binary (the third constraint in (2.40)). In this manner, the MRTA problem is represented by the ILP formulation in (2.40)).

$$\begin{aligned} \text{maximize } & \sum_{i=1}^{n}\sum_{j=1}^{n}(T_{ij}+E_{ij})x_{ij} \\ & \text{subject to:} \\ & \sum_{j=1}^{n} x_{ij} = 1 \text{where } 1 \leq j \leq n \\ & \sum_{i=1}^{n} x_{ij} = 1 \text{where } 1 \leq i \leq n \\ & x_{ij} \in 0,1 \text{where } 1 \leq i,j \leq n \end{aligned} \tag{2.40}$$

Similarly, in [226–228], the authors formulate an approach that they titled the *Beam Search Approach*. In these works, a multi-robot system (MRS) is used to perform surveillance of an industrial area, with the aim that these robots are able to collect information that proves useful in case of an industrial disaster. The MRTA is modeled as a multiple Travelling Salesman Problem (mTSP). The MRTA is formulated as an MILP, that decides which action each robot must take in the inspection mission, and what information each robot must collect. Moreover, it is know that mTSP is an NP-Hard problem. If the formulated MILP is a complicated one, such a system may not scale up well and the decision making process may take a long time, or infinite time, to finish its execution. Currently, these works do not analyse the computational complexity or the time complexity of the respective approaches. Such an analysis shall provide an idea of the size of the robot team that the MILP can handle, as well as the number of tasks that need to be distributed among the team. Korsah et al. [229] formulate the multi-robot multi-task allocation similar to [222].

In contrast to the novel task allocation scheme proposed in this thesis, these works assume that there are as many tasks as there are robots. Consequently, this kind of formulation is not suitable for a coverage control application with multiple regions of interest. For such a case, there will surely be tasks (density functions) that should be covered by more than one robot at a time. This is because multiple robots may be assigned to the same density, but only one density may be allocated to each robot. Furthermore, the cost function to be minimized in [222] is also not suitable to address several other practical factors that a coverage control application typically requires, such as the performance of the sensors.

A formulation similar to the above-mentioned works is proposed by Ye et al. in [223] where the authors consider a multi-stage manufacturing application. At each manufacturing stage, the ILP problem is solved in an attempt to distribute the tasks involved. In this work, as well as in [224], the ILP is solved in a centralized manner. Lippi and Marino [225] expand on such works by considering different human and robot abilities and allocate tasks to robots as well as to human-robot teams, bearing in mind that human abilities deteriorate as they get tired and lose concentration. Lippi and Marino [225] formulate this as an MILP problem, which allows the assignment of more than one agent to the same task, unlike the works in [222–224]. Similarly, in [230] the authors also formulate the task allocation problem as an MILP, where collision avoidance and connectivity are encoded as constraints to the optimization problem.

Similar to the works proposed in [222–225, 229, 230], Notomista et al. [159] also propose an optimization-based MRTA scheme, but in contrast employ a non-linear objective function. The authors propose a particular task allocation framework based on quadratic programming constrained optimization. The authors argue that such a framework needs to take into account the robot capabilities that each task requires as well as the features and capabilities that each robot may offer. Features of a robot refer to their sensing characteristics, while capabilities refer to the robots' algorithmic capabilities to carry out specific jobs, such as navigation. Hence, the task is created by collecting a number of the available robot capabilities according to the task requirements set by the designer. Through the use of hypergraphs, the authors encode a capabilities-to-task mapping, a robot-to-feature mapping, a feature-to-capability mapping, and a direct mapping between robots and capabilities. The final formulation of the tasks is then encoded

as a control barrier function (CBF), as done in [231]. Notomista et al. claim that their approach minimizes energy. This is because, similar to [155], the optimization minimizes the control effort u, while adhering to the constraints that the tasks need to be completed with an allowed amount of slacking in completing the tasks. This approach is resilient to situational changes, however it is not modular. The concept of energy awareness, the control input generation, and the idea of task completion are all encoded within the same mathematical formulation. Consequently it is not possible to separate these tasks to apply different and independent solutions, which might be imperative for certain applications. This renders the approach less general and hinders its adaptability.

The work proposed by Li et al. [122] attempts to solve the problem of having heterogeneous robots with different maximum velocities and therefore, must be assigned a subarea that is representative of their capability. As discussed in Section 2.4.4, the cell in which each robot is operating grows in a cyclic manner through a market-based approach. In this scheme small tasks representing a square cell in a region are traded between robots, such that each robot finally ends up with a collection of small tasks making up the region. The process involved in this work [122] is shown to have linear complexity in the first initial region-growing stage, however this excludes the tuning algorithm that is required to refine the partitions. Furthermore, the region-growing and task allocation in this work are merged together into one scheme, similar to the way that Notomista et al. [159] have merged the task allocation and control effort generation into one design problem. This shall create a tight dependency on how the regions are segmented and how the tasks are allocated, and would require a complete re-design of the formulation if one needs to be different than the other. Hence, the scheme in [122] suffers from the same lack of modularity as that in [159]. Additionally, the work proposed by Li et al. [122] does not address the notion of having time-varying environments. Rather, it assumes that the environment is static and known a priori and that it can be decomposed into a number of fixed, weighted square cells representing a task. Having a rapidly-changing environment would require this framework to recompute the embedded task allocation and region-growing scheme at the same rate as the changing environment.

2.6 Gaps in knowledge and contributions

In light of the reviewed literature, one can observe a number of gaps in the field of coverage control using a multi-robot team. In literature there are several frameworks that are designed for various specific applications in the field of multi-robot systems (MRSs). However, to the best of the authors' knowledge, there does not exist a modular framework specific for the coverage control problem. Individual works on coverage control mostly address one aspect of the coverage control problem, such as having time-varying areas of interest in the environment [114–117], including energy and sensor constraints in the problem [1,111], or else by considering multiple regions of interest in the same environment [120]. Some works attempt to address the robots' energy limitations together with the time-varying environment [155, 156, 163]. However, even in cases where multiple factors making up a practical coverage scenario are considered, such approaches are neither complete nor modular. One of the benefits of modularity is that each module is easily upgraded or replaced with new algorithms as they become available in literature as a result of technology advancements. Another benefit of modularity in such frameworks is that one module is easily deactivated for different applications. For example, let's suppose that in a particular application the environment is static and does not require a tracking control law. With a modular framework, one can easily disable the time-dependent control law and activate a time-invariant control law.

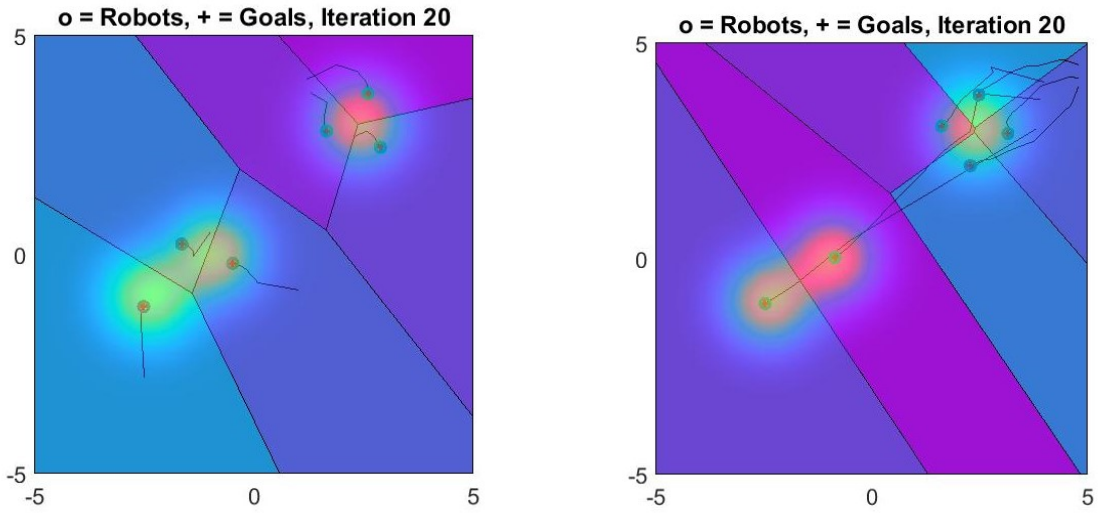
Dealing with multiple regions of interest (and hence multiple density functions) in one environment is also an area of research that requires attention. To the best of the author's knowledge, the only works in literature that attempt to address this problem are those by Abbasi et al. [120], Gosrich et al. [121] and Li et al. [122]. These works, as well as an understanding of how Voronoi diagrams are created, address the need to treat multiple density functions as individual and distinct, rather than as one multi-modal density function as attempted in [139]. Li et al. [122] move away from the conventional coverage approach that uses Voronoi diagrams to segment the environment, however, they still address the notion of spreading the robots across the environment and create coverage regions for the robot teams depending on the robots' capabilities. As discussed in Section 2.3, a CVT is dependent on the initial positions of the robots [133]. Hence, the final configuration of the Voronoi regions and how the environment is segmented depends on where the robots started the process.

Furthermore, the work proposed in [120] does treat density functions individually, however this work assigns the robots to these densities heuristically. This does not address the practical need to assign robots to densities according to the heterogeneous capabilities. Additionally, the controller proposed by Abbasi et al. [120] seems to assume that the robots positions in the environment shall not be affected just by the density that they are assigned to, but also by the densities assigned to their neighbouring robots. Furthermore, the controller proposed by Abbasi et al. is complicated to implement due to the computation of γ_i, and may not be practical in real-life systems with limited computational resources, especially if dealing with a small-to-medium sized multi-robot system (MRS). As shown in this thesis, an easier novel approach may be adopted to drive robots to their designated density functions without the algorithmic complications of Equation (2.36). Moreover, the work proposed by Abbasi et al. does not address the concept of having time-varying density functions, and the fact that the robots must be able to track these individual density functions accurately.

In light of the arguments above, one major gap in literature is that an application may have multiple areas that require individual attention from a specific set of robots in the team. If there is no association between the robots and the density functions, and the multiple density peaks are considered as one multi-modal density function, there is no control over which areas the robots shall cover, because this shall depend on from where they start converging towards a CVT. For example, in [139], Kim et al. propose the use of a density function that is the sum of two bivariate normal distributions. In such a case, the proposed algorithm is not addressing the concept of driving particular robots to particular densities. Therefore, there is no handle on which robots gravitate towards which density as long as all the densities are covered. However, the outcome of such an algorithm is heavily dependent on where the robots have been deployed in the environment.

Figure 2.7 demonstrates the difference between using a summation of multiple density functions as opposed to treating these density functions separately. Figure 2.7a shows three robots around the two peaks located in centre-left of the environment, while three other robots are covering the other peak of the density function. For the same exact environment but with different initial robot positions, there is only one robot on each of the centre-left peaks while the remaining robots cover the top right peak in Figure 2.7b.

This means that with an algorithm that considers the a summation of the individual peaks as one multi-modal density function, the coverage of the environment is entirely dependent on where the robots are first deployed.



(a) Initial conditions set 1 (b) Initial conditions set 1

Figure 2.7: Different CVTs for two different sets of initial robot positions using a scheme that does not utilise the allocation scheme to multiple, distinct density functions

Another reason in favour of pre-allocation is that there might be situations where certain areas of the environment need to be covered by robots which are equipped with specific sensors. For example, a density function may be modelling an aquatic area of interest that may only be covered by unmanned aerial vehicle (UAV) that are carrying visual cameras. If such robots are not associated with that density function and the final positions of the robots in the environment is dependent solely on their initial conditions, one might end with a ground vehicle trying to cover an aquatic area. This is not a practical solution and hence, definitely not optimal. Existing works in coverage control do not explicitly tackle the problem of allocating densities to the robots in the team, in a multi-density environment. The importance of such an allocation is twofold: first, by associating a density with a robot it means that there is complete control over which areas specific robots need to cover, secondly, it can be ensured that all important areas of the environment have robots that will cover them. Without such pre-association between robots and densities, there may be cases where robots do not manage to reach specific densities that are far away because they hover around the density closest to them.

Although this concept was addressed by Gosrich et al. [121] and Li et al. [122], these

works still leave a number of topics unaddressed. The work by Gosrich et al. [121] attempts to reduce the dependency on the initial robot positions by having the robots communicate more about their surroundings, hence encouraging the robots in the team to move to areas that have higher density peaks, thus increasing coverage of these areas. This, however, does not account for the optimal placement of these robots according to their capabilities and also suffers from communication link strength and data sharing over this link. On the other hand, the work proposed by Li et al. [122] does take into account the heterogeneity of the robots velocities when growing the subareas (synonymous to the Voronoi cells) and allocating the tasks in the areas. However, this work does not address the potential time variation properties of the environment, and it also merges the task allocation and environment segmentation problems into one formulation. Therefore, this work suffers from a lack of modularity and would require a complete redesign of the problem if for example, the environment segmentation objective function would need to be modified or simply, be different to the objective function of the task allocation problem.

When it comes to the segmentation problem, the works found in literature are mostly limited to addressing a single aspect. For instance in [1] the authors only address the energy factor when segmenting the environment, in [111] the authors consider the robots' actuation health only, the works in [112, 113] only consider the robots' health degradation and finally, the works in [136, 139] use time efficiency in the segmentation process. In addition, works such as [136, 139–142, 146–150] segment the environment using either a multiplicatively weighted Voronoi diagram or range-limited Voronoi diagram. The practical issues with such implementations would be that a multiplicatively weighted Voronoi diagram would result in areas that are segmented by parabolic arcs, that make it more difficult for the general framework to process and implement. Furthermore, using range-limited Voronoi diagrams may lead to areas that are unaccounted for when creating the Voronoi diagram.

For this purpose, the algorithm that segments the environment should be multi-faceted in addressing multiple, practical factors that affect the segmentation process. Additionally, the algorithm should ensure that all of the environment is accounted for such that no areas remain uncovered by the team. Finally, for ease of implementation, the areas generated must be convex polytopes such that the individual modules of the framework are able to easily manipulate these Voronoi cells.

When one looks at the robot control algorithms proposed in the literature of coverage control, the ones that are able to handle time-varying environments are either not energy-aware [114–116], or have the control law embedded with the task allocation and completion metric, making the algorithm non-modular [155, 159]. Besides being more computationally intensive, controllers such as those proposed in [114, 116] also suffer from complex implementation problems such as the existence and computation of matrix inverses. Furthermore, these controllers are derived from the need to find a controller that allows the robots to track the moving centroids in the environment. In this regard, a simple-to-implement tracking controller that is also aware of the energy consumption of the robots becomes beneficial to the overall modular framework that addresses the practical aspects of a coverage control application.

This thesis focuses on a number of problems that have not yet been addressed in the area of coverage control. Particularly, this thesis presents a holistic energy-aware solution to the coverage control problem where an environment has multiple, time-varying regions of interest and the robot team is heterogeneous in its capabilities. The approach proposed in this thesis addresses the dependency of the coverage control problem on the initial positions of the robots, especially when the environment has multiple importance regions, while at the same time, the proposed approach focuses on exploiting the heterogeneous capabilities of the robots in all aspects of the coverage control algorithm. Each region of importance is modelled as a probability density function that is assumed to be known a priori. These probability density functions are mathematical equations set by the system designer based on some knowledge of the important features found in the environment.

This approach is designed to be energy efficient by accounting for the energy consumption of the robots in the environment segmentation as well as in the generation of the control efforts of the robots. In the subsequent chapters of this thesis, the energy levels of the robots represent the energy reserve of a robot. In practical terms, this energy reserve may be expressed as the percentage of the battery charge that a robot may have left to move autonomously, or the amount of fuel left in the robot's tank. In this thesis, the term *energy level* will be used such that there is no loss of generality. It is also assumed that the maximum capacity of the amount of kinetic energy that a mobile robot may use, together with the threshold at which a robot is assumed to be completely depleted of energy (and hence, it becomes inactive), are known. Following the critical

appraisal presented in this chapter, the main novel contributions presented in this thesis can be summarized as follows:

1. This thesis proposes a general multi-robot framework that is particularly tailored for the coverage control problem. Unlike the work proposed in [122], this is a modular framework that builds on various works that address individual practical aspects in coverage control. This framework consists of three main modules: the robot-to-density allocation module, the multi-faceted environment segmentation module, and the robot motion controller module. In this framework, the algorithm continuously checks the robots' energy levels and use the robot-to-density allocation module to re-assign the tasks in the environment, if required, such as in the case of robot malfunctions. The environment segmentation module also runs continuously such that on every iteration, the Voronoi cells generated for each robot are representative of the current states (such as energy levels) of the robots. Finally, the proposed motion controller is also energy aware, such that it is able to conserve energy and allow the robots to run for a longer period of time. Furthermore, this framework considers the coverage problem to have multiple, distinct areas of interest. A block diagram of the proposed framework is provided in Figure 2.8.

2. A novel multi-robot to multi-density allocation algorithm is proposed to address the need to optimally assign robots to densities according to the robots' capabilities, energy levels, and the densities' requirements. Unlike other works [1, 117, 126, 155], an environment is considered to be composed of multiple regions of importance, with each region of importance requiring specific capabilities from the robots in the team. This is one of the three pillar modules of the proposed general framework. This contribution addresses the need to embed the robot heterogeneity in a scenario where multiple areas of importance dominate the environment. For example, an unmanned aerial vehicles (UAVs) would be indifferent to a density function that requires an unmanned surface vehicles (USVs) since this would be operating in a body of water. In such a case, it would not make sense that the UAV is attracted to the density function requiring the USV. Furthermore, this module contributes towards the optimal coverage of the environment since the allocation of robots to densities shall be done in an optimal manner. For the purpose of this thesis, the criteria used for optimal allocation shall include the current energy reserve of the

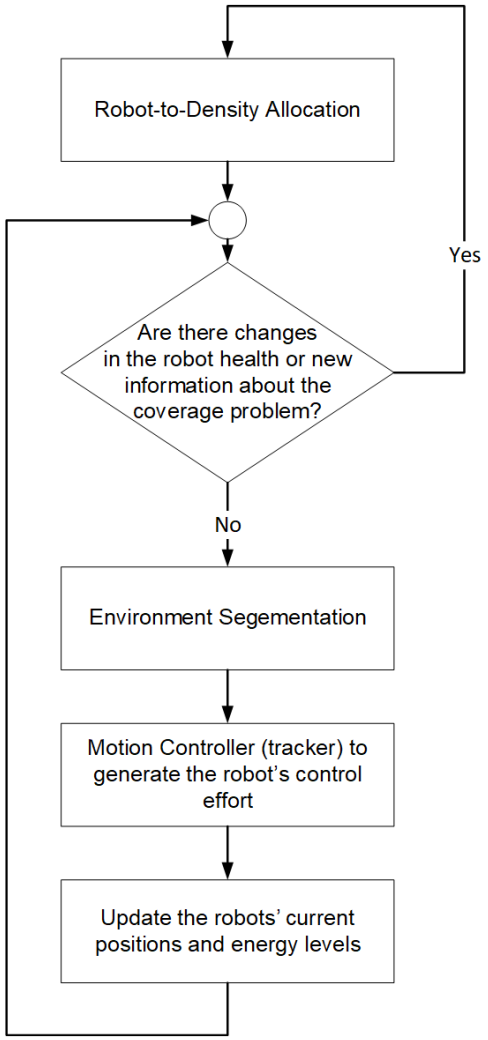


Figure 2.8: A block diagram of the proposed framework

robots, the performance of the sensors onboard the robots, the maximum velocities of the robots as well as the time taken for the robots to arrive at each density function. In addition, this module is also computationally efficient and can be applied for real-time applications for small to medium-sized teams of robots. This is seen through an in-depth study of the time complexity of this algorithm. The details of this design and implementation are documented in Chapter 3.

3. The environment segmentation algorithm is designed to be multi-faceted. This means that it shall consider multiple practical factors when segmenting the environment using power diagrams. For the purpose of this work, the main factors addressed in this module mirror those used in the proposed multi-robot task allocation module, namely, the current energy reserve of the robots, the performance

of the sensors onboard the robots and the maximum velocities of the robots. This enhances the uniformity between the different modules and the computational efficiency of the framework. This environment segmentation algorithm is a contribution to the field since to the best of the authors' knowledge to date, there is no algorithm that considers multiple practical factors when segmenting the environment during coverage control. The details of this design and implementation are documented in Chapter 4.

4. The controller used in the general framework is based on the tracking controller proposed by Cortés et al. [115]. However, in this thesis, this tracking controller was modified to become an energy aware motion tracker. Once again, this contributes to the end goal of having a framework that strives to stretch the energy reserves of the robots with the aim of allowing the overall system to operate for a longer period of time. The details of this design and implementation are documented in Chapter 5. Furthermore, in Chapter 6 this controller shall be implemented in the general framework where multiple density functions are considered. This controller is shown to drive the robots to their assigned density functions. Unlike other controllers proposed in literature, this energy-aware tracking controller allows the robots to achieve arbitrarily good tracking performance at the expense of energy. This is achieved by the tracker tuning parameter that allows the trade-off between tracking performance and energy consumption. Finally, with this tracking controller, the framework is shown to achieve convergence to CVT through a proof presented in Appendix A.

2.7 Conclusions

Coverage of an environment is fast becoming an important problem in several applications, from search and rescue, to surveillance and even extending to specific applications such as agricultural tasks. Using a multi-robot system (MRS) to execute the coverage task is potentially the most efficient way of doing this. However, a practical multi-robot system (MRS) comes with several factors that affect the coverage control problem. Some of these factors are the energy limitations of the robots in the team, their maximum actuator capabilities (such as their maximum velocities) and the performance of their

on-board sensors. Apart from these factors that must be accounted for in the coverage problem, the environment itself can also have its own practical features that must be considered. Particularly, the environment can have multiple, specific areas of higher importance (modelled by probability density functions) and such areas can also vary with time. The current works in literature tend to focus on just one of these practical aspects that affect the coverage control problem. For instance, in [1,104–109] the authors consider just the energy limitations of the robots in the problem, while in [114–119] the authors only consider the time varying probability density functions in the environment.

Although there exist many works in literature that focus on individual aspects that affect the coverage control problem, to the best of our knowledge, none of these works propose one holistic framework that incorporates all of these practical aspects. To this effect, this thesis proposes a novel framework that, through its architecture, is able to address multiple facets that typify a practical implementation of the coverage control problem with a multi-robot system (MRS). Among these aspects, the novel framework considers the energy limitations of the robots, their maximum velocities, the performance of their on-board sensors, as well as the environmental aspect of having multiple, time-varying higher regions of importance. This proposed framework has a modular architecture such that it is easily modified and adapted to different coverage control applications. In addition to the novel framework, this thesis also proposes novel contributions in each of the individual modules that make up this framework.

Chapter 3

Robot-to-density allocation

3.1 Overview

A given environment composed of multiple regions of importance with specific requirements requires a smart robot allocation scheme. Such a concept becomes more important in applications where each region requires a different set of robot capabilities. For example, in a specific application a team of robots needs to survey and monitor marine species travelling in packs (such as jellyfish blooms). The same environment may consist of different species located in different areas. Hence, each region may require different types of sensors onboard the robots to monitor the different species. Such applications motivate the design of a smart multi-robot multi-density allocation scheme. In this chapter this robot-to-density allocation problem is formulated as a combinatorial optimization problem.

Although the proposed scheme is of a generic nature, it was primarily designed for a coverage control algorithm that requires the heterogeneous team of robots to be assigned to multiple, distinct areas of interest. Within the coverage control framework this robot-to-area allocation process is executed prior to the environment segmentation process. The proposed novel allocation scheme takes into account the diverse capabilities of the robots, their current positions, their sensors' performance and their energy levels. These factors are merely a subset of what such a scheme can consider, and the list of factors under consideration may be easily adapted to cater for different applications.

Additionally, the scheme also considers the particular requirements of the areas or tasks that must be serviced. Using combinatorial logic, the proposed algorithm optimally

assigns robots to tasks (areas to be covered) by minimizing a specifically formulated cost function, subject to a number of constraints. Moreover, the proposed scheme is designed to be able to re-assign robots on the fly during the coverage task. Re-allocation becomes important when there are significant changes in the heterogeneous team of robots, for example when a robot's sensor malfunctions or when a robot ends up without energy. Re-allocation might also be required if there are significant changes in the environment. Such situational changes would mean that a re-evaluation and a new solution to the allocation problem is required to maintain optimal operation. Hence, the proposed scheme is designed to have robustness to situational changes. This chapter introduces two variants of the task allocation algorithm. Algorithm 1.0 seeks to have all the robots' capabilities match exactly the requirements of the densities, while Algorithm 2.0 relaxes this rigidity in favour of maximising the use of the available robotic resources.

The robot allocation problem addressed in this chapter is formulated as an integer linear programming (ILP) problem. This ILP formulation considers all the feasible robot-to-density mappings. In contrast to [222], the generality of the proposed method is that the algorithm does not require that the number of robots matches the number of tasks. Particularly, the proposed scheme evaluates a cost value for all the possible robot-to-density mappings, subject to the constraints that each task must be serviced by at least one robot, and that each robot must be assigned once. These cost values are evaluated and updated periodically during the coverage process, making the scheme robust to situational changes and to sudden robot and sensor faults. The cost value of each mapping reflects how well the requirements of a task are serviced by each robot. To reflect an infeasibility of a mapping due to a mismatch between the requirements of a task and the robots' capabilities, an arbitrarily large value is assigned to the cost of the mapping.

The nature of the proposed scheme is similar to that proposed by Notomista et al. [159], in that it solves the robot-to-task allocation problem. However, unlike [159], our approach separates the task allocation problem from the optimization problem that generates the control input. Hence, the scheme proposed in this chapter has a modular structure. Modularity has several advantages in a coverage control application. Primarily, having a modular framework allows for more than one algorithm to be plugged into each stage of the coverage process. Advancements in technology and algorithms shall then

be easier to include into the established framework. Furthermore, the proposed scheme allows the user to forcefully set a robot-to-task mapping prior to the optimization stage and in a sense allowing a level of human-machine collaboration.

Finally, the two variations of the proposed scheme provide flexibility to the task allocation module, in that the module may either consider only those robots that perfectly match the requirements of the density, or else relax this rigidity by allowing the robot capabilities to partially match the requirements. Most of the concepts introduced in this chapter are detailed in the following sections that present each of the two allocation algorithm variants.

The main problem formulation as well as the details and results of Algorithm 1.0 have been published in [232], while all the details of the formulation, implementation and validation results for both Algorithm 1.0 and Algorithm 2.0 have been submitted for publication in [233].

R. N. Duca, M. K. Bugeja, "A multi-robot allocation scheme for coverage control applications with multiple areas of interest", 9th International Conference on Control, Decision and Information Technologies, July 2023.

R. N. Duca, M. K. Bugeja, "A task allocation scheme for heterogeneous robot teams in coverage control applications", IEEE Access, [Manuscript submitted for review, October 2023].

As discussed in Chapter 2, the relevance and the importance of using an allocation algorithm such as that proposed in this chapter is twofold. Firstly, having robots optimally allocated to the multiple, distinct importance regions in an environment according to their capabilities ensures that these areas are being served by the robot capabilities that they require. In a practical application this allows the end user to optimally benefit, for example, from the diverse sensors onboard the robots in the team, according to the needs of the application. Secondly, as suggested in [120–122], treating the density peaks as separate density functions, as opposed to one single multi-modal density function allows for better coverage of these peaks by reducing the dependency on the initial positions of the robots. In the grand scheme of the general framework proposed in this thesis, a particular focus is placed on the tight association between a robot and the density function it is allocated to. Hence, the focus of the framework would be to have these robots service only those densities that they are associated with.

3.2 Problem formulation

As mentioned previously, the proposed MRTA scheme is formulated as an ILP problem with a set of specifically designed constraints that reflect the particular conditions of this problem. The scope of the proposed algorithm is to optimally assign n_r robots to n_d distinct tasks. Without any loss of generality, the rest of this chapter assumes that the term *tasks*, as used in a general MRTA problem description, refers to the coverage of each area of interest (modelled as a probability density function) in the coverage control problem.

In this context, the notion of *tasks* and *densities* shall be used interchangeably. Each density function represents an area of higher importance than others (a region of interest) in an environment Q. Each of these density functions has a number of requirements listed in terms of the robot type and the sensors that it requires from a robot. Furthermore, each robot in the team is described by its physical configuration, i.e. unmanned ground vehicle (UGV), unmanned aerial vehicle (UAV) or unmanned surface vehicle (USV), and it also has a number of sensor capabilities, i.e. LiDAR, visual camera and/or thermal sensor. These robot types and sensors are merely examples chosen for this study, and may generally be expanded and adapted according to the specific application.

In the proposed scheme, the algorithm first checks whether the type and sensor capabilities of each robot in the team match the requirements of any of the density functions. This matching process renders a robot 'a candidate' for a particular allocation, if it results in the positive. The actual matching process, or 'candidacy test', varies between Algorithm 1.0 and 2.0. In Algorithm 1.0, for a robot to become a candidate for a particular density, it must have the physical configuration required by the density, and it must possess *at least all* the capabilities required by that density. In this case the particular robot-to-density mapping is considered as feasible. The rigidity of this 'candidacy test' is relaxed in Algorithm 2.0, where a robot is still considered a feasible candidate if it has the physical configuration required by the density, and if it possesses *at least one* of the sensors required. Further details about these candidacy tests are presented in Sections 3.3 and 3.4.

In the proposed scheme, a number of constraints are imposed to ensure optimal coverage of the environment. Firstly, each candidate robot-to-density allocation may either be chosen, or not. This makes the selection variable a binary variable, hence

rendering the formulation a *binary integer programming*, or *zero-one linear programming* problem [234]. Secondly, each robot must be assigned to one density only. Finally, each density must be serviced by at least one robot. Furthermore, if there are no feasible matches between a robot and any of the densities, that particular robot is removed from the optimization problem a priori. Similarly, if a density is not matched by any capabilities of the available robots, then that density is also removed. The user may also intentionally force a robot-to-density mapping a priori. These forced assignments are treated as further constraints by the optimization algorithm, and the rest of the feasible matches are optimized accordingly.

In light of these requirements, the proposed algorithm first draws up a cost matrix, with each cell corresponding to each robot-to-density mapping. The cost of each mapping is calculated by the algorithm itself. If a robot-to-density mapping is not feasible according to the previously described candidacy test of Algorithm 1.0 or Algorithm 2.0, then an arbitrarily large number is assigned for that particular mapping. For readability, in this thesis this arbitrarily large number is represented by ∞. Alternatively, if a mapping is deemed feasible (i.e. the candidacy test is positive), the corresponding cost value is calculated. This cost value is composed of a number of factors that reflect the demands of the application. In this thesis, the cost value is based on four factors that affect the coverage control problem, namely: the distance that a robot must travel to arrive to a given density, the speed at which a robot can reach a density, the current energy level of the robot, and its sensors' performance rating. One must note that although the proposed scheme considers the four above-mentioned factors, the allocation algorithm is flexible, in that each factor can be modified, removed or new ones added, to suit better the requirements of the specific application. Finally, the $(i,j)^{th}$ element $C_{i,j}$ of the cost matrix $\boldsymbol{C}$, representing the total cost of robot i servicing density j, is computed by

$$C_{ij} = \beta_1\gamma_d + \beta_2\gamma_t + \beta_3\gamma_e + \beta_4\gamma_s \tag{3.1}$$

where γ_d, γ_t, γ_e, and γ_s are cost values normalized between 0 and 1 and introduced to respectively account for: the Euclidean distance between robot i and a reference point on density j; the shortest time taken for robot i to reach density j depending on its maximum velocity $v_{i(max)}$; the current energy level E_i of the robot; and the robot sensors' performance rating. More specifically, γ_d and γ_t are computed using (3.2) where

γ_σ represents a generic normalized cost value, $\gamma_{u,\sigma}(i,j)$ is that cost's un-normalized value corresponding to the $(i,j)^{th}$ robot-density pairing, and $\boldsymbol{\gamma_\sigma}$ is the vector containing all $\gamma_\sigma(i,j)$ values.

$$\gamma_\sigma = \frac{\gamma_{u,\sigma}(i,j) - \min(\boldsymbol{\gamma_\sigma})}{\max(\boldsymbol{\gamma_\sigma}) - \min(\boldsymbol{\gamma_\sigma})} \tag{3.2}$$

The normalized energy cost γ_e, which is effectively the remaining energy as a fraction of the total available energy $E_{i(max)} - E_{i(min)}$, is computed by

$$\gamma_e = \frac{E_{i(max)} - E_i}{E_{i(max)} - E_{i(min)}} \tag{3.3}$$

$E_{i(max)}$ and $E_{i(min)}$ represent the maximum and minimum energy level that a robot may have. These are set by the designer according to the application. The mean sensor performance rating γ_s is computed using (3.4), where n_s is the total number of sensors on robot i that are required by density j, and s_k is the performance cost of each sensor, that varies between 0 and 1 (a lower cost indicates a better sensor).

$$\gamma_s = \frac{1}{n_s} \sum_{k=1}^{n_s} s_k \tag{3.4}$$

The reasoning behind the choice of the cost components in (3.1), in the context of coverage control, is as follows. The distance between each robot and each density is considered such that if two robots are identical in their characteristics but one is closer to a density than the other, the closer one is preferred for that allocation. Generally speaking in a group of similar robots this leads to higher time and energy efficiencies. The maximum robot velocity is also included in the overall cost. Having a robot that can reach a density in a shorter amount of time may prove to be indispensable for certain time-critical applications. To promote energy preservation, and therefore improve the overall energy efficiency, an energy dependent factor is included. Finally, including the sensor performance in the cost value, rewards robots with better-performing sensors in the optimization process. All these factors combined in an ILP shall lead to an optimal allocation of robots to densities, given the limitations of the resources available for coverage.

In (3.1), β_1 to β_4 denote positive weights of the individual cost elements. These are

adjusted by the user to change the character of the optimization process, by having the possibility of putting more weight on any one (or more) of the factors relative to the others. Furthermore, although the probability density functions may take the form of any function, typically these are allowed to be Gaussian. Therefore, in this algorithm their peaks are taken as the reference points when calculating γ_d. Moreover, since the total cost presented in (3.1) is additive in nature, additional cost factors may be easily included. The resulting cost matrix $\boldsymbol{C}$ is considered resilient to changes in both the robots and their environment since the components in (3.1) are constantly being updated to reflect the current state of the system. For instance if the robot's sensor rating degrades because of damages that may occur to the sensor, the robot may no longer be a valid candidate for a particular density, and hence, by setting its cost arbitrarily large, one ensures that that mapping is not selected.

Based on all the above assumptions and definitions, the proposed robot optimal allocation scheme is formulated as an ILP problem as follows:

$$\begin{aligned} &\min_{\boldsymbol{x}} \sum_{i=1}^{n_r} \sum_{j=1}^{n_d} C_{ij} x_{ij} \\ &\text{subject to:} \\ &x_{ij} \in \{0,1\} \\ &\sum_{i=1}^{n_r} x_{ij} \geq 1 \text{ for } j = \{1,2,...,n_d\} \\ &\sum_{j=1}^{n_d} x_{ij} = 1 \text{ for } i = \{1,2,...,n_r\} \end{aligned} \tag{3.5}$$

The optimization variable $\boldsymbol{x}$, is a binary assignment vector of length $i \times j$. This means that if the ILP solution yields $x_{23} = 1$, Robot R_2 is assigned to Density 3. Furthermore, the first constraint ensures that a mapping is either selected ($x_{ij} = 1$) or not ($x_{ij} = 0$). The second constraint in (3.5) ensures that each density is serviced by at least one robot, while the third constraint ensures that each robot is allocated to only one density.

The implementation of the proposed scheme is based on a centralized architecture programmed in MATLAB®. Each of the algorithms presented in Sections 3.3 and 3.4 are

programmed as a MATLAB® function that takes the following input arguments: the robot capabilities, the density requirements, the robot positions, the reference points on each density, the sensors costs s_k, the robots' maximum velocities, the robots' energy levels, and whether there are any user-forced pairings between robots and densities. Implementation details of each of the two algorithm variations, together with a set of validation test cases and results, are presented in the following two sections.

One must note that in the following descriptions, the density functions are assumed to be Gaussian. The formulation of these density functions is based on prior human knowledge of the environment, as discussed in Section 2.6. For the purpose of the presented validation tests, the means and variances of the Gaussian density functions are set arbitrarily and the mean is chosen to represent the density's reference point in the allocation algorithm. However, in the general case the user can tie the density's reference point to any other measure of choice to fit the particular application. Additionally, referring to the terms defined in Chapter 2, p_i refers to the i^{th} robot's position, E_i is its energy level (i.e. energy reserve) and $v_{i(max)}$ is the maximum velocity that a robot may reach.

3.3 Algorithm 1.0

3.3.1 Implementation

The design of Algorithm 1.0 is based on the premise that the application requires that the candidate robot matches *all* the requirements (configuration and sensors) of the paired task (density function). If a robot has at least one missing capability, such as a missing sensor, with respect to a density's requirements, then an arbitrarily large number is assigned as the cost of that pairing. Otherwise, i.e. if the robot has all the capabilities and therefore is a good candidate for a task, the actual cost value of that pairing is computed through Equation (3.1). The reasoning behind this design is to ensure that a particular task is fully serviced by the same robot, which might be an important requirement in certain scenarios.

This method is outlined clearly by the pseudo-code provided in Algorithm 3.1. Note that $forced\_pairings$ is a matrix where each row corresponds to a robot and each column corresponds to a density. A value of one in any of this matrix's elements represents a user-forced mapping between the corresponding robot (row) and density (column).

Additionally: $index_{j\_forced}$ is the index that identifies those densities with which a robot i has a forced pairing; n_r and n_d are the total number of robots and densities in the problem; $robot.capabilities$ is a structure that holds all the capabilities (type and sensors) of each robot; and $density.requirements$ is a structure that holds all the requirements (type and sensors) of each density. Furthermore, one must also consider that Algorithm 3.1 relies heavily on the MATLAB® language and therefore the unfamiliar reader is referred to the appropriate literature [235–238]. One function of particular importance is the `intlinprog` function [238], which is the ILP MATLAB® solver. This function uses the *Branch and Bound* algorithm to efficiently find an optimal solution [238].

3.3.2 Validation tests

In this section, a series of test cases are presented to investigate and validate the effectiveness of Algorithm 1.0. These test cases are conducted in MATLAB® simulations and aim to replicate realistic scenarios. Each scenario is tailored to evaluate a specific aspect of the algorithm's functionality. All tests assume that the physical configuration of the robot can be categorised as either a UGV, a UAV, or a USV. In addition, the robots may be equipped with one or more of the following sensors: a LiDAR, a visual camera, and a thermal sensor. It should be noted that these configurations and sensor options can be easily expanded to suit different applications. It is important to note that the number of robots and the number of densities in these tests have been kept intentionally low to ease the verification of the results.

Test case 1

In the first test case, there are no predetermined user-forced assignments of robots to densities. The test case includes three density functions (i.e. representing different tasks or regions to cover in a coverage control problem) and four robots. The requirements and 2D location of each density function, as well as a summary of the robots' capabilities and the costs associated with each sensor, are tabulated in Tables 3.1 and 3.2. In addition, Table 3.3 contains information about the properties of each robot, including its current energy level, maximum speed and its current position.

After evaluating the cost matrix using Equation (3.1) and taking into account the given constraints, the ILP solver generates the result shown in Table 3.4. As expected

Algorithm 3.1. Algorithm 1.0

procedure NORMALIZED_QUANTITIES
    Compute the normalized quantities, γ_d, γ_t, γ_e, and γ_s using (3.2), (3.3) and (3.4).
end procedure

▷ The outer loop (with variable i) iterates through all the available robots while the inner loop (with variable j) iterates through all the density functions.
for $i = 1 : n_r$ **do**
    for $j = 1 : n_d$ **do**
                ▷ Checking for any user-forced mappings
        if $any(forced\_pairings(i, :)) == 1$ **then**
            $index_{j\_forced} = find(forced\_pairings(i, :))$
            if $j == index_{j\_forced}$ **then**
                $C_{ij} = 0$
            else
                $C_{ij} = \infty$
            end if
        else
        ▷ Checking for matches between robot capabilities and density requirements.
            if $robot.capabilities(i) == density.requirements(j)$ **then**
                Calculate C_ij using (3.1)
            else
                $C_{ij} = \infty$
            end if
        end if
    end for
end for
                ▷ Generate the constraints matrices and vectors
$\boldsymbol{A} = -1 * repmat(eye(n_d), 1, n_r)$
$\boldsymbol{b} = -1 * ones(n_d, 1)$
$\boldsymbol{Ac} = repmat(ones(1, n_d), 1, n_r)$
$\boldsymbol{Aeq} = blkdiag(Ac:)$
$\boldsymbol{beq} = ones(n_r, 1)$

$\boldsymbol{x} = intlinprog(\boldsymbol{C}, \boldsymbol{A}, \boldsymbol{b}, \boldsymbol{Aeq}, \boldsymbol{beq})$ ▷ Executing the ILP solver
return $\boldsymbol{x}$

Table 3.1: Density function parameters

	Density functions		
	$\boldsymbol{D_1}$	$\boldsymbol{D_2}$	$\boldsymbol{D_3}$
Requirements	UGV	UGV	UGV
	LiDAR	Thermal	Visual
	Visual		
Reference point on density	(1, 2)	(2, 1.5)	(3, 4)

Table 3.2: Robot capabilities and their sensors' costs

	Robot capabilities		
$\boldsymbol{R_1}$	UGV	LiDAR, cost=0.5	Visual, cost=0.2
$\boldsymbol{R_2}$	UGV	Visual, cost=0.2	Thermal, cost=0.3
$\boldsymbol{R_3}$	UGV	Thermal, cost=0.7	
$\boldsymbol{R_4}$	UGV	Visual, cost=0.2	

Table 3.3: Robot parameters

	$\boldsymbol{E_i}$	$\boldsymbol{v_{i(max)}}$	$\boldsymbol{p_i}$
$\boldsymbol{R_1}$	10	10	(5, 4.5)
$\boldsymbol{R_2}$	6	2	(4.8, 4)
$\boldsymbol{R_3}$	8	5	(5, 4.2)
$\boldsymbol{R_4}$	7	4	(3.8, 3)

in this test case, R_1 is the only robot that perfectly satisfies the requirements of D_1 and is consequently assigned to it. An interesting observation is that R_2 could be assigned to either D_2 ($C = 6.354$) or D_3 (leading to a total cost C of 5.185); however, the optimization algorithm chooses the latter as this yields a lower total cost value of 5.185 as opposed to the alternative of 6.354.

Table 3.4: Algorithm 1.0 test case 1 results

Cost Matrix			
	$\boldsymbol{D_1}$	$\boldsymbol{D_2}$	$\boldsymbol{D_3}$
$\boldsymbol{R_1}$	1.487	∞	0.427
$\boldsymbol{R_2}$	∞	2.289	1.117
$\boldsymbol{R_3}$	∞	2.016	∞
$\boldsymbol{R_4}$	∞	∞	0.565

Optimal allocations	
R_1	D_1
R_2	D_3
R_3	D_2
R_4	D_3

In a similar experiment, the same density functions (Table 3.1), robot capabilities and positions are used, but the sensors' performance ratings, the robots' energy levels and their maximum velocities are changed to the values reported in Tables 3.5 and 3.6. The aim of this experiment is to study the sensitivity of the algorithm to different parameters.

Table 3.5: Robot capabilities and their sensors' costs

	Robot capabilities		
$\boldsymbol{R_1}$	UGV	LiDAR, cost=0.1	Visual, cost=0.5
$\boldsymbol{R_2}$	UGV	Visual, cost=0.9	Thermal, cost=0.1
$\boldsymbol{R_3}$	UGV	Thermal, cost=0.2	
$\boldsymbol{R_4}$	UGV	Visual, cost=0.6	

Table 3.6: Robot parameters

	$\boldsymbol{E_i}$	$\boldsymbol{v_{i(max)}}$	$\boldsymbol{p_i}$
$\boldsymbol{R_1}$	4	5	(5, 4.5)
$\boldsymbol{R_2}$	10	12	(4.8, 4)
$\boldsymbol{R_3}$	2	2	(5, 4.2)
$\boldsymbol{R_4}$	3	7	(3.8, 3)

The results of the optimal allocations together with the cost matrix are presented in Table 3.7. From this table, one can observe that R_2 to D_2 allocation yields the lowest cost. This makes sense because the thermal sensor that R_2 uses to serve D_2 has a significantly lower cost than the visual sensor that R_2 uses to serve D_3, as shown in the table of normalized cost components in Table 3.8. In this light, a change in these parameters has resulted in a change in the allocations of the robots to the same density functions used in the previous experiment. This shows that the proposed algorithm is sensitive to variations in the robots' parameters.

Table 3.7: Algorithm 1.0 test case 1 with parameter variations results

Cost Matrix			
	$\boldsymbol{D_1}$	$\boldsymbol{D_2}$	$\boldsymbol{D_3}$
$\boldsymbol{R_1}$	2.284	∞	1.463
$\boldsymbol{R_2}$	∞	0.896	1.051
$\boldsymbol{R_3}$	∞	2.694	∞
$\boldsymbol{R_4}$	∞	∞	1.329

Optimal allocations	
R_1	D_1
R_2	D_2
R_3	D_2
R_4	D_3

Table 3.8: Algorithm 1.0 test case 1 normalized cost components

Normalized component	R_2 **to** D_2 **candidate**	R_2 **to** D_3 **candidate**
γ_d	0.7197	0.1511
γ_t	0.0763	0
γ_s	0.1	0.9
γ_e	0	0
C_{ij}	0.896	1.0511

Test case 2

For the second test case, the same exact inputs of Test Case 1 are used, but a pairing between R_1 and D_3 is forced a priori. In this case, the algorithm returns the optimal allocations tabulated in Table 3.9.

Table 3.9: Algorithm 1.0 test case 2 results

Cost matrix			
	D_1	D_2	D_3
R_1	∞	∞	0
R_2	∞	2.289	1.117
R_3	∞	2.016	∞
R_4	∞	∞	0.565

Optimal allocations	
R_1	D_3
R_2	D_3
R_3	D_2
R_4	D_3

This test case shows that density D_1 is not served by any of the robots. This is because, due to the user-forced allocation of R_1 with D_3, none of the other robots have the necessary capabilities to satisfy the requirements of D_1. Consequently, D_1 is excluded from the assignment problem from the outset.

Test case 3

In this particular test case, the inputs remain identical to those in Test Case 1. However, there is a slight change in the robot capabilities, which are shown in Table 3.10. As a result of these changes, one of the robots does not have the necessary capabilities to be allocated to any of the three densities.

The optimal allocations generated by the proposed algorithm in this case, along with the corresponding cost matrix, are presented in Table 3.11. It can be observed that R_3 is once again assigned to D_2, as it remains the only robot that possesses the capabilities necessary to fully satisfy the necessary requirements. Another noteworthy observation is that R_1 does not appear in any of the allocations generated by the algorithm. This is

Table 3.10: Robot capabilities and the sensors' costs

	Robot Capabilities		
$\boldsymbol{R_1}$	UAV	LiDAR, cost=0.5	Visual, cost=0.2
$\boldsymbol{R_2}$	UGV	Visual, cost=0.2	Thermal, cost=0.3
$\boldsymbol{R_3}$	UGV	Thermal, cost=0.7	
$\boldsymbol{R_4}$	UGV	Visual, cost=0.2	

because R_1 was automatically excluded from the pool of available robots by the candidacy test before the optimisation process even started, as none of the densities require the use of a UAV. Understandably, this resulted in D_1 being left unattended, as none of the other robots have the required capabilities.

Table 3.11: Algorithm 1.0 test case 3 results

Cost matrix			
	$\boldsymbol{D_1}$	$\boldsymbol{D_2}$	$\boldsymbol{D_3}$
$\boldsymbol{R_1}$	∞	∞	∞
$\boldsymbol{R_2}$	∞	2.289	1.117
$\boldsymbol{R_3}$	∞	2.016	∞
$\boldsymbol{R_4}$	∞	∞	0.565

Optimal allocations	
R_2	D_3
R_3	D_2
R_4	D_3

Test case 4

In this particular test case, the robot fleet is increased to seven robots and the number of densities is set to three. This is done in order to simulate a more realistic and more complex scenario. It is important to note that there are no user-forced pre-assignments of robots to densities in this scenario. The parameters used for this test case are listed in the Tables 3.12, 3.13, and 3.14.

Table 3.12: Density function parameters

	Density functions		
	$\boldsymbol{D_1}$	$\boldsymbol{D_2}$	$\boldsymbol{D_3}$
Requirements	UGV	UAV	UGV
	LiDAR	Visual	Visual
	Thermal		
Reference point on density	(1, 2)	(2, 1.5)	(4, 3)

After performing the candidacy test described in Section 3.3.1, the optimization algorithm generates the cost matrix and robot-to-density allocations listed in Table 3.15.

Table 3.13: Robot capabilities and the sensors' costs

	Robot capabilities			
$\boldsymbol{R_1}$	UGV	LiDAR, cost=0.5	Visual, cost=0.2	Thermal, cost=0.4
$\boldsymbol{R_2}$	UGV	Visual, cost=0.1	Thermal, cost=0.3	
$\boldsymbol{R_3}$	UAV	Visual, cost=0.7		
$\boldsymbol{R_4}$	UAV	Visual, cost=0.1		
$\boldsymbol{R_5}$	UAV	LiDAR, cost=0.1	Visual, cost=0.6	
$\boldsymbol{R_6}$	UGV	LiDAR, cost=0.2	Visual, cost=0.3	Thermal, cost=0.8
$\boldsymbol{R_7}$	UAV	Visual, cost=0.2	Thermal, cost=0.4	

Table 3.14: Robot parameters

	$\boldsymbol{E_i}$	$\boldsymbol{v_{i(max)}}$	$\boldsymbol{p_i}$
$\boldsymbol{R_1}$	10	10	(5, 4.5)
$\boldsymbol{R_2}$	6	2	(4.8, 4)
$\boldsymbol{R_3}$	8	5	(5, 4.2)
$\boldsymbol{R_4}$	7	4	(3.8, 3)
$\boldsymbol{R_5}$	10	7	(1, 2)
$\boldsymbol{R_6}$	10	7	(2.5, 3)
$\boldsymbol{R_7}$	5	10	(1.5, 3)

In this test case, robots R_3, R_4, R_5 and R_7 could only be paired to density D_2 because they are all UAVs equipped with at least a visual camera, as required by D_2. This shows that setting ∞ for the infeasible pairings has the desired effect of eliminating them from the solution. Moreover, robots R_1 and R_6 competed for the same densities. The result produced by the algorithm ensures that the chosen pairings (R_1 paired with D_3 and R_6 paired with D_1) yield the lowest total cost for the whole team. Furthermore, this result also shows that the algorithm satisfies the condition that each density must be serviced by at least one robot, if there are feasible pairings. This property is shown by the fact that as the algorithm does not assign R_1 and R_6 to D_3, as this would mean that density D_1 remains unserviced.

3.4 Algorithm 2.0

3.4.1 Implementation

Algorithm 2.0 relaxes the rigidity found in Algorithm 1.0 that for a robot to be a good candidate for a given density, it needs to possess *all* the requirements (type, and all

Table 3.15: Algorithm 1.0 test case 4 results

Cost Matrix			
	$\boldsymbol{D_1}$	$\boldsymbol{D_2}$	$\boldsymbol{D_3}$
$\boldsymbol{R_1}$	1.67	∞	0.666
$\boldsymbol{R_2}$	∞	∞	1.078
$\boldsymbol{R_3}$	∞	2.136	∞
$\boldsymbol{R_4}$	∞	1.176	∞
$\boldsymbol{R_5}$	∞	0.911	∞
$\boldsymbol{R_6}$	1.002	∞	0.718
$\boldsymbol{R_7}$	∞	1.119	∞

Optimal allocations	
R_1	D_3
R_2	D_3
R_3	D_2
R_4	D_2
R_5	D_2
R_6	D_1
R_7	D_2

sensors) associated with that density (task). Furthermore, Algorithm 2.0 still has the same main features of Algorithm 1.0, namely, the notion that a robot-to-density mapping may be forced, and that the mappings are formulated by minimizing the overall cost of the ILP in (3.1). However, in Algorithm 2.0, the robot's physical configuration (type) must still always match that required by the density function. However, through the cost matrix calculation process described by the flowchart in Figure 3.1, a robot may contribute to the servicing of a density through *one or more* of the sensors required by the density, and not necessarily through *all* of them. Note that the flowchart in Figure 3.1 shows how each of the elements in C_{ij} is calculated. Moreover, Algorithm 2.0 has the additional feature that a density (task) may optionally request a subset of its sensor requirements to appear together on the candidate robot. This subset of sensors is called a *Forced Sensor Group*. In addition to a forced sensor group, a density may also have other individual sensor requirements that require no special grouping and therefore are treated in the regular manner (as in Algorithm 1.0). Hence, in Algorithm 2.0 a robot of the correct physical configuration (type) may contribute to the servicing of a density function (be a good candidate) through one of the following options:

- It has just the sensors requested in the forced group,
- It does not have the sensors requested by the forced group, but it has one or more of the other sensors that do not form part of the forced group.

The process presented by the flowchart of Figure 3.1 can be clarified further by the following example. Consider that a particular task (density) requires a UGV as the physical configuration of the robot. This UGV must be equipped with both a thermal sensor and LiDAR simultaneously (forced group), while at the same time, the density also requires a

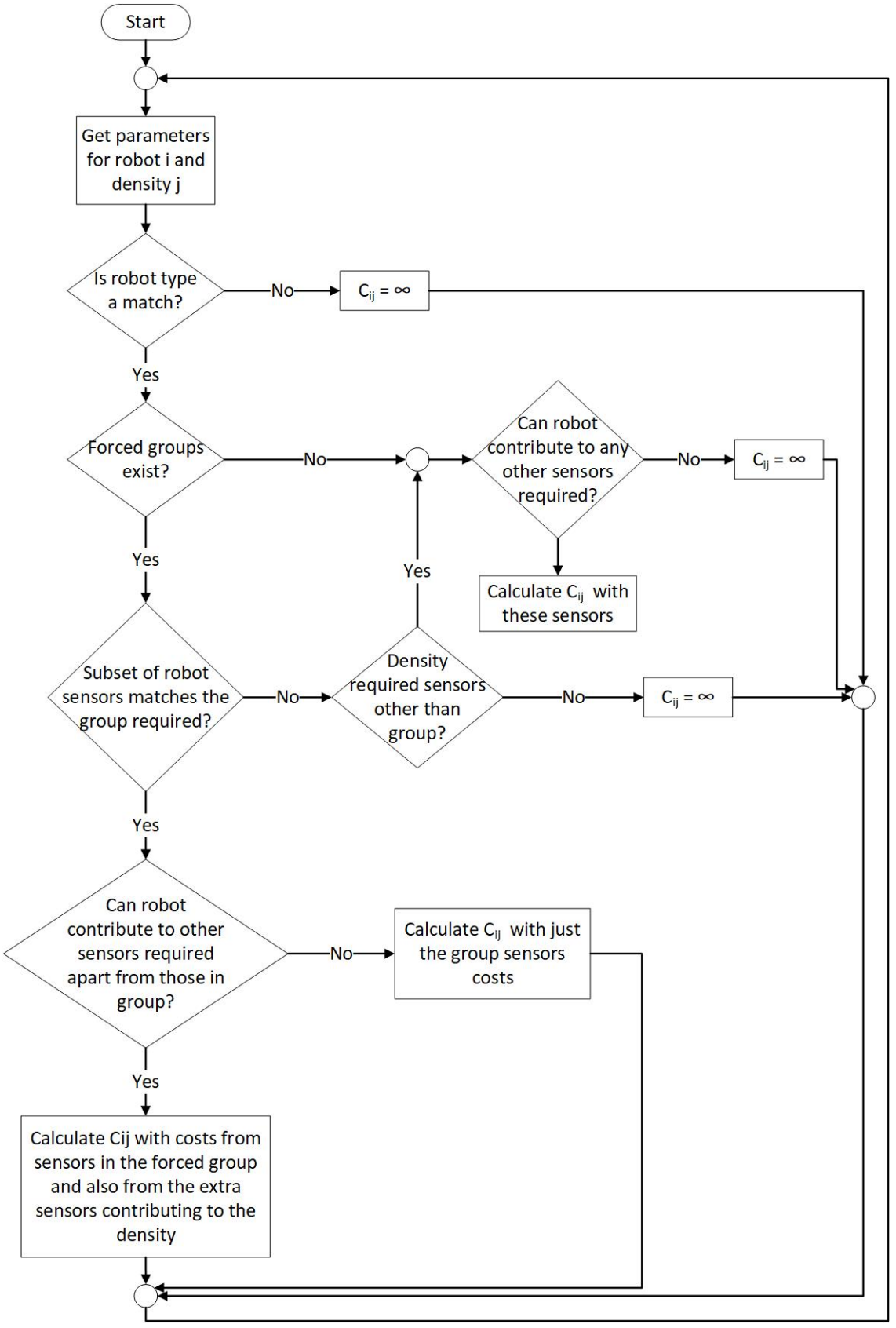


Figure 3.1: Algorithm 2.0 cost matrix calculation process flowchart

visual camera from either the same robot or from any other robot. Robot R_1 is a UGV equipped with a thermal sensor only, robot R_2 is equipped with a thermal sensor and a LiDAR, while robot R_3 is equipped with a visual camera. In this example, R_1 is not a good candidate to service the density because it does not have all the sensors required in the forced group and neither does it have the visual camera to contribute to the density. R_2 is a valid candidate because it possesses *all* of the forced group sensors, while R_3 is also a candidate because it may service the density through the use of its visual camera. Moreover, if neither R_3 nor R_2 form part of the team, one may end up with a situation where the density function is not fully serviced with all the sensors it requires. This variation to Algorithm 1.0 is motivated by the idea of maximizing the team resources by allocating as many of the robots as possible, even if this means that one or more of the densities end up being only partially serviced. It is interesting to note that if all of the sensors required by each density function are placed in a forced group, Algorithm 2.0 effectively converges to Algorithm 1.0. A full comparison of the two algorithms is included in Section 3.5.1.

3.4.2 Validation tests

Algorithm 2.0 was validated by subjecting it to a number of different test case scenarios in simulation. It is important to note, that unlike the test cases presented in the case of Algorithm 1.0, the validation process of the optimization cost evaluation, that involves the manual computation all the possible optimization combinations, has now become too complex and arduous. Hence, for the test cases used for Algorithm 2.0, the algorithm relies on the optimization validation process carried out for Algorithm 1.0, where the cost matrix is much simpler for the designer to evaluate manually. Additionally, the following test cases seek to validate whether each robot is appropriately allocated to a density according to its capabilities and according to the requirements and restrictions of the densities, rather than the ILP optimization as such. Moreover, to make the results more readable, in the cost matrices presented in this section, a value of ∞ is assigned to infeasible mappings. One must note that in practice this value is set to an arbitrarily large value.

Test case 1

In this test case, there are no forced robot-to-density mappings and the problem has three density functions and four robots. Table 3.16 tabulates the requirements and location of each density function, where shaded cells mark sensors that are forced in a group. Table 3.17 summarizes the robot capabilities together with the cost associated with each sensor. Finally, Table 3.18 tabulates the properties (current energy level, maximum velocity, and current position) of each robot.

Table 3.16: Density function parameters

	Density functions		
	$\boldsymbol{D_1}$	$\boldsymbol{D_2}$	$\boldsymbol{D_3}$
Requirements	UGV	UGV	UGV
	LiDAR	LiDAR	LiDAR
	Visual	Visual	Thermal
	Thermal		
Reference point on density	(1, 2)	(2, 1.5)	(3, 4)

Table 3.17: Robot capabilities and the sensors' costs

	Robot capabilities			
$\boldsymbol{R_1}$	UGV	LiDAR, cost=0.2		
$\boldsymbol{R_2}$	UGV	LiDAR, cost=0.2	Visual, cost=0.2	
$\boldsymbol{R_3}$	UGV	LiDAR, cost=0.2	Visual, cost=0.2	Thermal, cost=0.2
$\boldsymbol{R_4}$	UGV	Visual, cost=0.2		

Table 3.18: Robot parameters

	$\boldsymbol{E_i}$	$\boldsymbol{v_{i(max)}}$	$\boldsymbol{p_i}$
$\boldsymbol{R_1}$	10	10	(5, 4.5)
$\boldsymbol{R_2}$	6	2	(4.8, 4)
$\boldsymbol{R_3}$	8	5	(5, 4.2)
$\boldsymbol{R_4}$	7	4	(3.8, 3)

Since all the densities require a UGV and all the available robots are in fact UGVs, then the robot type requirement for each density is satisfied by all the robots. By inspecting the available robots' capabilities and matching them to the requirements of the densities, one can draw up the following points:

- R_1 does not match any of the sensors in the forced group of D_1 or D_3, but it can contribute in part to the requirements of D_2 through its visual camera.

- R_2 has all the sensors required by the forced group of D_1, and it may also service D_2.
- R_3 is able to service all of the densities.
- R_4 does not match all of the sensors required by the forced group of D_1, however it is able to service D_2 with its visual camera.

The allocations generated by the algorithm are tabulated in Table 3.19. From these results, one can notice that since R_3 was the only robot that could satisfy the forced group of D_3, it was allocated as such. This is because of the constraint that a density must be serviced by at least one robot. If R_3 did not form part of the team, D_3 would have remained unserviced. With R_3 out of the picture, R_2 is the only robot that can satisfy the forced group of D_1. Finally, R_1 and R_4 are both allocated to D_2. In this case, due to the limited resources available in the team and due to the constraint that densities must be serviced by at least one robot, the system ends up with D_1 not being fully serviced by all the sensors it requested. However, this was done in order to increase the servicing of other densities.

Table 3.19: Algorithm 2.0 test case 1 results

Optimal allocations	
R_1	D_2
R_2	D_1
R_3	D_3
R_4	D_2

Test case 2

Similarly, in this test case there are no forced robot-to-density mappings. The density requirements and parameters are tabulated in Table 3.20. Furthermore, Tables 3.21 and 3.22 tabulate the robot capabilities (and sensor costs) and the robot parameters, respectively.

After considering the robots' capabilities against the requirements of the densities, one can make the following observations:

- R_1 is able to service D_2 through its LiDAR sensor, while it is able to service D_3 through its visual sensors.

Table 3.20: Density function parameters

	Density functions		
	$\boldsymbol{D_1}$	$\boldsymbol{D_2}$	$\boldsymbol{D_3}$
Requirements	UAV	UGV	UGV
	LiDAR	LiDAR	LiDAR
		Thermal	Visual
			Thermal
Reference point on density	(1, 2)	(2, 1.5)	(3, 4)

Table 3.21: Robot capabilities and the sensors' costs

	Robot capabilities			
$\boldsymbol{R_1}$	UGV	LiDAR, cost=0.2	Visual, cost=0.2	
$\boldsymbol{R_2}$	UGV	LiDAR, cost=0.2	Thermal, cost=0.2	
$\boldsymbol{R_3}$	UAV	LiDAR, cost=0.2		
$\boldsymbol{R_4}$	UAV	Visual, cost=0.2	Thermal, cost=0.2	
$\boldsymbol{R_5}$	UAV	LiDAR, cost=0.2	Visual, cost=0.2	Thermal, cost=0.2
$\boldsymbol{R_6}$	UGV	LiDAR, cost=0.2	Visual, cost=0.2	Thermal, cost=0.2
$\boldsymbol{R_7}$	UGV	LiDAR, cost=0.2	Visual, cost=0.2	

Table 3.22: Robot parameters

	$\boldsymbol{E_i}$	$\boldsymbol{v_{i(max)}}$	$\boldsymbol{p_i}$
$\boldsymbol{R_1}$	10	10	(5, 4.5)
$\boldsymbol{R_2}$	6	2	(4.8, 4)
$\boldsymbol{R_3}$	8	5	(5, 4.2)
$\boldsymbol{R_4}$	7	4	(3.8, 3)
$\boldsymbol{R_5}$	10	7	(1, 2)
$\boldsymbol{R_6}$	10	7	(2.5, 3)
$\boldsymbol{R_7}$	5	10	(1.5, 3)

- R_2 is able to service D_2 using all of its sensors, while it is able to service D_3 since it has all of the sensors required by the forced group of D_3.
- R_3 is only able to service D_1 since it is a UAV and has the LiDAR sensor requested by D_1.
- Although R_4 is a UAV (as requested by D_1), it does not have the necessary sensors to service this density. Since its robot type does not match the types requested by the other densities, then R_4 is unable to form part of the team and is therefore eliminated prior to the optimization step.
- R_5 is able to service D_1 using its LiDAR sensor.

- R_6 is able to service either D_2 or D_3 because it has all the sensors required, especially the sensors in the forced group of D_3.
- R_7 is able to service D_2 because it has all of the sensors requested by D_2, while it is able to service D_3 by using its visual sensor only.

The allocations generated by the algorithm are tabulated in Table 3.23. From these results, one can notice that the algorithm utilised the majority of the team, even if some of the robots are only servicing a density through one of their sensors. Maximising the size of the team is beneficial since this directly contributes to better coverage of the environment. This in turn leads to better balancing of the workload among the team of robots, hence the coverage operation can go on for a prolonged time since the robots would not be overworked. Additionally, having more robots in the team results in more diverse capabilities and versatility that might become useful halfway through the operation (for example, deciding to use a currently-unused sensor on a particular robot mid-operation). Moreover, having more robots means a greater redundancy in the team which is beneficial in case of robot failures.

Table 3.23: Algorithm 2.0 test case 2 results

Cost matrix			
	$\boldsymbol{D_1}$	$\boldsymbol{D_2}$	$\boldsymbol{D_3}$
$\boldsymbol{R_1}$	∞	1.297	0.7331
$\boldsymbol{R_2}$	∞	2.278	1.409
$\boldsymbol{R_3}$	1.797	∞	∞
$\boldsymbol{R_4}$	∞	∞	∞
$\boldsymbol{R_5}$	0.2	∞	∞
$\boldsymbol{R_6}$	∞	0.640	0.511
$\boldsymbol{R_7}$	∞	1.119	1.176

Optimal allocations	
R_1	D_3
R_2	D_3
R_3	D_1
R_5	D_1
R_6	D_3
R_7	D_2

Test case 3

In this test case, the algorithm is further validated by increasing the number of densities that need to be serviced as well as the robots available. For this particular test case, it is assumed that the robots are deployed from the same initial positions, such that the effect of the distance between a robot and a density is now diminished. Furthermore, this test case shows how the algorithm operates when the densities are not subjected to any forced sensor groups. The density requirements and parameters are tabulated in Table 3.24.

Furthermore, Tables 3.25 and 3.26 tabulate the robot capabilities (and sensor costs) and the robot parameters, respectively.

Table 3.24: Density function parameters

	Density functions				
	$\boldsymbol{D_1}$	$\boldsymbol{D_2}$	$\boldsymbol{D_3}$	$\boldsymbol{D_4}$	$\boldsymbol{D_5}$
Requirements	USV	UGV	UAV	UGV	UAV
	Visual	LiDAR	LiDAR	LiDAR	LiDAR
	Thermal	Visual		Thermal	Visual
		Thermal			Thermal
Reference point on density	(1, 2)	(5,4)	(3, 1)	(1,4)	(2,1)

Table 3.25: Robot capabilities and the sensors' costs

	Robot capabilities			
$\boldsymbol{R_1}$	UAV	LiDAR, cost=0.5	Thermal, cost=0.1	
$\boldsymbol{R_2}$	UGV	Visual, cost=0.3	Thermal, cost=0.6	
$\boldsymbol{R_3}$	UGV	LiDAR, cost=0.1	Visual, cost=0.1	Thermal, cost=0.2
$\boldsymbol{R_4}$	USV	Visual, cost=0.6	Thermal, cost=0.5	
$\boldsymbol{R_5}$	UGV	Visual, cost=0.2		
$\boldsymbol{R_6}$	UAV	Visual, cost=0.1	Thermal, cost=0.3	
$\boldsymbol{R_7}$	USV	LiDAR, cost=0.1	Visual, cost=0.2	Thermal, cost=0.4
$\boldsymbol{R_8}$	UGV	LiDAR, cost=0.2		
$\boldsymbol{R_9}$	UAV	LiDAR, cost=0.2	Visual, cost=0.4	Thermal, cost=0.6
$\boldsymbol{R_{10}}$	UGV	Thermal, cost=0.2		

Table 3.26: Robot parameters

	$\boldsymbol{E_i}$	$\boldsymbol{v_{i(max)}}$	$\boldsymbol{p_i}$
$\boldsymbol{R_1}$	5	10	(3, 3)
$\boldsymbol{R_2}$	4	2	(3, 3)
$\boldsymbol{R_3}$	10	5	(3, 3)
$\boldsymbol{R_4}$	8	4	(3, 3)
$\boldsymbol{R_4}$	8	7	(3, 3)
$\boldsymbol{R_4}$	9	7	(3, 3)
$\boldsymbol{R_4}$	10	10	(3, 3)
$\boldsymbol{R_4}$	10	2	(3, 3)
$\boldsymbol{R_4}$	6	3	(3, 3)
$\boldsymbol{R_4}$	7	4	(3, 3)

The resulting robot-to-density mappings are tabulated in Table 3.27. When analysing these results, one can observe that the allocation of robots to densities respects the capabilities of the robots with respect to the density requirements. It is interesting to

note that D_5 is not receiving the service of a LiDAR sensor because the algorithm only allocated R_6 to this density. R_6 can only contribute its visual camera and thermal sensor to D_5. It is important to note that there were no forced groups of sensors placed on the sensors of D_5, and that the cost values are tabulated in Table 3.27. R_9 would have appeared to be a better fit for D_5 since it would have been able to fully service it with all of its required sensors. In this case, the cost values primarily depend on the robot's maximum velocity, sensor performance rating and energy levels. In this light, the overall minimum cost of the ILP was achieved when R_6 was paired with D_5 and R_9 was paired with D_3. To guarantee that a density is fully serviced with all of the sensors that it requires, the forced groups of sensors in Algorithm 2.0 should be used.

Table 3.27: Algorithm 2.0 test case 3 results

Cost Matrix					
	$\boldsymbol{D_1}$	$\boldsymbol{D_2}$	$\boldsymbol{D_3}$	$\boldsymbol{D_4}$	$\boldsymbol{D_5}$
$\boldsymbol{R_1}$	∞	∞	1.01	∞	1.836
$\boldsymbol{R_2}$	∞	3.062	∞	3.212	∞
$\boldsymbol{R_3}$	∞	1.403	∞	1.419	∞
$\boldsymbol{R_4}$	2.095	∞	∞	∞	∞
$\boldsymbol{R_5}$	∞	1.534	∞	∞	∞
$\boldsymbol{R_6}$	∞	∞	∞	∞	1.432
$\boldsymbol{R_7}$	1.326	∞	∞	∞	∞
$\boldsymbol{R_8}$	∞	2.2	∞	2.2	∞
$\boldsymbol{R_9}$	∞	∞	1.117	∞	2.402
$\boldsymbol{R_{10}}$	∞	1.897	∞	1.897	∞

Optimal allocations	
R_1	D_3
R_2	D_2
R_3	D_2
R_4	D_1
R_5	D_2
R_6	D_5
R_7	D_1
R_8	D_4
R_9	D_3
R_{10}	D_4

3.5 Discussion

3.5.1 Comparison of Algorithm 1.0 to Algorithm 2.0

Two variants of the same MRTA formulation are presented in this chapter in an attempt to propose an overarching MRTA that suits different applications. While neither Algorithm 1.0 nor Algorithm 2.0 can be thought of superior to the other, this section compares the main benefits and drawbacks of each of these two algorithms. The choice of the MRTA variation presented in this chapter depends highly on what the designer of the application sets out to achieve.

The main benefit of Algorithm 1.0 is that it ensures that a density is always fully

serviced with *all* of the robot capabilities, as long as there is at least one robot that matches its requirements. This may be very beneficial in specific applications where the data from all of the sensors required by the density function is expected to be available. For instance, an application in which the robots are running a SLAM (simultaneous localization and mapping) algorithm can possibly have density functions where the data required should come from a single robot. However, one main limitation of this approach is that one or more healthy and useful robots in the team, may be completely unused because they do not have a whole set of sensors required by any of the tasks, on-board. Algorithm 2.0 improves on this by allowing a density function to be serviced with the required sensors through different robots in the same team. This maximises the use of the available resources, such that the coverage workload is divided among the available robots better. This is highly relevant when a smaller team of robots is used. Generally speaking the workload burden consumes the energy of each robot faster than when the same burden is shared among a larger team.

Another benefit of Algorithm 2.0 is the feature of forced groups. This allows an application to ensure that a certain group of sensors are always found on a singular robot. However, as already explained, a limitation of Algorithm 2.0 is that a density may end up not being serviced by all the sensors it requires from the robot team. This is of course a limitation that is directly tied to the limitations of the capabilities of the robots in the team. If all of the robots are all fully equipped with all of the potential sensors that a density may require, Algorithm 2.0 would not suffer from the above-mentioned limitation. Arguably, the limitations of both Algorithms 1.0 and 2.0 come about due to the practical, and non-idealistic limited resources and capabilities that typify a heterogeneous team of robots.

To bring out the main differences between the proposed algorithm variants, consider the conditions presented in Test Case 2 in Section 3.4.2. If both algorithms are run with these conditions, the results in Table 3.28 are obtained. From these results one can observe that with Algorithm 2.0, only one robot ended up not being assigned at all, and this is because its capabilities do not match any of the densities' requirements. However, when running Algorithm 1.0 with the same conditions, another two robots (R_1 and R_7) end up not forming part of the team. This is due to the aforementioned rigidity of requiring that all sensors come from the same robot. The choice of which features of

which algorithm to use is highly dependent on the application.

Table 3.28: Comparing Algorithm 1.0 and 2.0 using the conditions of Section 3.4.2

Optimal Allocations		
	Algorithm 1.0	Algorithm 2.0
R_1	—	D_3
R_2	D_2	D_3
R_3	D_1	D_1
R_4	—	—
R_5	D_1	D_1
R_6	D_3	D_3
R_7	—	D_2

3.5.2 Computational load

It is a known fact that in general, an ILP problem is an NP-Hard problem [221, 239]. For this reason, there are no known algorithms that are able to solve all the problems formulated as ILPs in polynomial time. Moreover, there are special instances where the ILP may be heuristically solved in polynomial time, using heuristic methods such as the Branch and Bound technique. For example, by considering four robots and two densities, one can easily find the optimal allocation in polynomial time because the solution to this ILP is quite simple. This, however, does not mean that all instances of an ILP are like this, and therefore, in general ILP is considered as an NP-Hard problem.

In this light, this section investigates whether the two proposed algorithms may be optimally solved in polynomial time as the number of inputs (robots and densities) to the ILP increases. Specifically, the increase in the time complexity is investigated as the inputs increase, to address the time complexity issue in small-to-medium sized multi-robot systems. Moreover, the ILP solver used to obtain the results presented in this chapter, namely the MATLAB® function `intlinprog`, uses the *Branch and Bound* algorithm [238]. The branch and bound algorithm is an algorithm that is widely used to generate exact solutions for NP-hard optimization problems such as the ILP [240–243]. This algorithm inspects all possible solutions by representing all the possible combinations as a tree search space. Internally, the algorithm generates *bounds* that determine whether a tree node should be included as part of a candidate solution or not. At each tree level, the node with the best bound is evaluated such that the optimal solution is rapidly found.

It is important to observe that the time complexity of the branch and bound algorithm varies from one problem to another. This also depends on the branching strategy used as well as the pruning techniques adopted in the algorithm [240]. This means that the time complexity of any branch and bound algorithm is related to the branching factor, b used to create the search tree and the depth of the search tree, d. Therefore, at the worst-case running time, the branch and bound algorithm operates with a time complexity of $O(Mb^d)$, where M is a bounded period of time required to investigate one of the candidate sub-problems in the search tree [240]. Furthermore, in the MATLAB®function `intlinprog`, the performance of the branch and bound algorithm depends on the branching rule used [244]. The branching rule chosen for this algorithm is 'reliability', that has slower branching method than other branching rules but has potentially fewer branch and bound iterations [244]. Furthermore, this branching rule is observed to save execution time overall [244].

In this light, the proposed algorithms are examined and tested for their time complexity by profiling the execution of the proposed algorithms as they are solved using the branch and bound algorithm. In doing so, the limitation of how many robots and densities may be included in the problem such that the solution is still given in a reasonable amount of time is analyzed. In other words, in this experiment, the time complexity of the proposed algorithms are evaluated. The time complexity is measured as the time taken for the whole algorithm to perform the candidacy test as well as to solve the ILP problem.

For this experiment, all the densities require ground vehicles equipped with all the sensors (LiDAR, visual camera and thermal sensor), and all robots are UGVs equipped with all the necessary sensors. The robots' energy levels and maximum velocities shall be randomly generated between some maximum and minimum bounds, which are the same for all robots. The reference points on the densities, and the robots' positions are also randomly generated. Finally, the sensors' performance costs are randomly generated values between 0 and 1. In this experiment, there are no forced robot-to-density mappings and the densities shall not have any forced sensor groups. The experiment involves increasing the number of robots and densities, and noting down the time taken for the algorithm to complete. These tests were run on an office laptop with an Intel®Core^{TM} i7-6500U CPU operating at 2.5GHz, with 20GB RAM installed. To time the algorithms, the MATLAB®*Profiler* tool [245] was used. The same experiment was repeated ten

times for each algorithm, to account for variations in the system load and other factors that may affect the measurements. An average of the time taken for each algorithm to execute in each case, is then calculated. The averaged measurements are tabulated in Table 3.29 and used to plot the time complexity of both algorithms in Figure 3.2. More specifically, in Figure 3.2, the time taken for the algorithm to execute is plotted against the number of robot-to-density combinations (i.e. the size of the search space in the cost matrix). The flattened size of the cost matrix is considered as the number of inputs in the decision problem. The cost matrix is flattened by reshaping it into a column vector using row-major order.

Table 3.29: Experiment to test computational load of the algorithms

Number of Robots	**Number of Densities**	**Number of inputs to the ILP**	**Average Execution time for Algorithm 1.0 (s)**	**Average Execution time for Algorithm 2.0 (s)**
5	2	10	0.015	0.063
100	5	500	0.044	1.69
200	5	1000	0.086	3.352
300	5	1500	0.119	4.762
250	8	2000	0.139	6.349
250	10	2500	0.173	7.965
300	10	3000	0.204	9.744
350	10	3500	0.24	11.334
400	10	4000	0.277	13.052
450	10	4500	0.319	14.645
200	25	5000	0.31	15.795
220	25	5500	0.352	17.461
240	25	6000	0.399	18.994
260	25	6500	0.417	20.583
280	25	7000	0.461	22.019
300	25	7500	0.481	23.866
320	25	8000	0.511	25.198
360	25	9000	0.596	28.959
400	25	10000	0.662	30.872

From Figure 3.2, one can observe that as the number of elements in the cost matrix increases, the amount of time taken for the algorithms to execute increases linearly. It is important to note that this cannot be extended to all ILP formulations, and generally depends on the complexity of the problem. This is because, in general, an ILP is classified as an NP-Hard problem [239, 240, 246]. The linear relationship between the number of inputs to the ILP and the time taken to achieve the optimal allocations may be explained

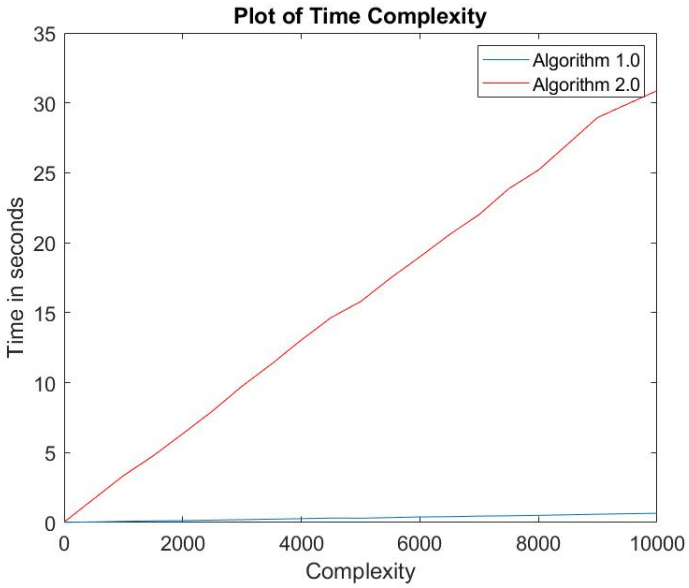


Figure 3.2: Time complexity plot for both algorithms

as follows. On further analysis, the linear time complexity may be attributed to the formulation of the constraints in (3.5). These constraints allow the branch and bound algorithm to prune away several candidate sub-problems where a robot is allocated twice or if a density is not assigned with any robots. This reduces the search space and thus, facilitates the efficiency with which the branch and bound algorithm reaches an optimal solution. Additionally, the greater the number of densities in the problem, the smaller the percentage of admissible sub-problems. This can be seen if one draws up all the candidate sub-problems manually. For instance, with four robots and two densities, the total number of candidate sub-problems is 4^2. Out of these 16 candidates, only 14 of them are admissible since in the other two sub-problems, one of the constraints in (3.5) is violated. However, with three robots and three densities, out of a total number of 27 candidate sub-problems, only nine are admissible due to the violation of one or more of the candidates. Therefore, this pruning away of inadmissible candidate solutions contributes to the linear relationship illustrated in Figure 3.2. Moreover, it is not the purpose of the proposed work to mathematically derive the time complexity of the proposed formulation. Rather, this study aims to present simulation results and identify the relationship between the number of inputs to the problem and the time taken to finish the execution in a practical manner.

To push the algorithm to its limits, the experiments conducted in this study consider

significantly large teams of robots and density sets that are not typical in real MRTA scenarios. For example, using the proposed Algorithm 2.0 to find an optimal solution to the MRTA problem for a team of fifty robots that need to cover around ten to fifteen distinct areas of interest would take approximately two seconds, which is still feasible in a practical application. In addition to this, in a real application the MRTA process would only need to run before the coverage process starts (to allocate the robots to their tasks before they start) and occasionally to adjust to situational changes, in a scenario that merits a re-allocation of the robots to the available densities. Therefore, the short amount of time that the MRTA would need to execute shall not inhibit the real-time coverage process.

Additionally, these results also highlight the difference in complexity between Algorithm 1.0 and Algorithm 2.0. Due to its simplistic nature, Algorithm 1.0 is better in scaling up in terms of the number of inputs than Algorithm 2.0. For larger teams of robots, and a larger number of tasks, a relaxation method, such as Lagrangian relaxation [247,248], should be applied to the ILP. Alternatively, if the problem must handle a large number of inputs, an offline computation of the allocations needs to be implemented using non-deterministic methods such as genetic algorithms [215, 216]. The computation in this case is said to execute offline since it takes a longer time to finish its execution and hence cannot run in real-time while the coverage control process is running. Using non-deterministic approaches such as genetic algorithms, however, may yield sub-optimal allocations. The presented experiment shows that the proposed ILP formulation may be optimally solved using the branch and bound method in `intlinprog`, within a very reasonable amount of time, given very humble computing resources.

3.6 Conclusions

The proposed robot-to-density allocation algorithms are specially formulated to work as a standalone module. Such a module may also be executed on its own, as a separate entity from the general framework. With minor modifications to the objective function, such a formulation may also be adapted for other allocation applications. Nevertheless, these algorithms were primarily designed to address the problem of allocating robots to multiple areas of interest in the environment for a coverage control application. This module is the

first module that needs to executed in the coverage control framework. The new approach presented in this work differs from previous multi-robot task allocation (MRTA) methods discussed in the existing literature. It takes into account a variety of factors related to the environment and the robots involved, and it remains effective even when these factors change. This effectiveness is achieved by regularly re-evaluating and updating the cost matrix in real time. Also, unlike previous approaches, this method does not assume a one-to-one mapping between robots and density. Another, a notable feature of this approach is that the user can retain a degree of control over the solution by assigning robots to specific areas in advance. To the best of the author's knowledge, this feature is not present in existing solutions. Furthermore, unlike [159], the proposed algorithm is modular and easily adaptable. It can be adapted to applications that require the same integer linear programming (ILP) formulation but different configurations. The allocation algorithm and the coverage control algorithm are separate, which is advantageous as these two problems are different and typically require different approaches. Both variants of the proposed scheme have been tested using realistic simulations and it is tested further as an integral part of the proposed framework in Chapter 6.

Chapter 4

Environment segmentation

4.1 Overview

After optimally allocating the robots to the multiple importance areas in the environment, the environment must be segmented optimally as well. As discussed in Chapter 2, one may choose to segment an environment just by focusing on the spatial metric of the whole team by using the standard Voronoi diagram computed according to (2.4), where $f(\| q - p_i \|) = \| q - p_i \|$. This means that the Voronoi diagram is computed depending only on the distance between each discrete point in the environment and the robots' positions (that act as seed points for the Voronoi diagram). Although this may prove sufficient in applications where factors such as energy consumption, actuator and sensor capabilities are inconsequential, this is not so in coverage control where typically these factors are at the forefront of the optimization problem. Hence, in cases where these extra non-spatial factors need to be addressed in the segmentation of the environment, the generalised Voronoi diagram, such as a *weighted Voronoi diagram*, should be adopted.

This module is based on a particular weighted Voronoi diagram called the *Power Diagram*. This power diagram employs the power metric described in Definition 2.3. The power metric was chosen over multiplicatively weighted and additively weighted metric because in contrast, it does not suffer from non-convex Voronoi cells, from having disconnected boundaries and holes in the segmentation process, or from having star-shaped or empty regions [1, 128]. For instance, the solution proposed by Kim et al. [139] results in Voronoi cells that have arcs for boundaries. In the grand scheme of a general framework, these arcs prove difficult to analyse and compute, especially when the general algorithm

requires the analysis of neighbouring robots. In cases where the boundaries between the neighbouring robots need to be computed, it would be a simpler procedure to evaluate the points on a straight line rather than on an arc. Hence, computing a power diagram shall facilitate the process of dealing with convex polytopes. In contrast to what happens in standard Voronoi diagrams, in power diagrams the seed position may end up being out of its region after computing the new boundaries. This drawback is less significant in the coverage control problem, since the robots are always driven towards the centroid of the Voronoi region, and hence, with a good controller, the robots shall be placed within the region that they are assigned to.

This chapter outlines the novel design of a power diagram tailored for a coverage control application that considers energy limitations, as well as the robot actuator performance and the on-board sensors' capabilities. To demonstrate the effectiveness and the significance of the proposed environment segmentation scheme, it is numerically compared to standard Voronoi segmentation. As a validation process, a new metric against which the environment segmentation is evaluated is proposed. This is done to ensure that the regions generated make sense with respect to the current status of the respective robots.

4.2 Design and implementation

Recalling w_i in (2.3), the design of the power diagram revolves around how this weight w_i is set,

$$\mathcal{V}(p_i) = \{q \in \mathbb{R} | \parallel q - p_i \parallel^2 - w_i \leq \parallel q - p_j \parallel^2 - w_j, \forall j \neq i\}$$

In an attempt to make the proposed framework more generic and easier to implement, some of the factors that are considered in the multi-robot multi-density allocation module are incorporated in the segmentation process as well. For this purpose, the weight of the power diagram w_i is made to depend on the current energy consumption of each robot, the robot's maximum velocity and the robot's sensor capabilities. By making use of the same factors, it is ensured that the environment segmentation module supports and strengthens the main factors that a coverage control application typically needs to consider.

In a similar fashion, as an energy-aware approach, Kwok and Martínez [1] adopt power

diagrams and adjust the weight to reflect the robot's energy consumption. However, unlike Kwok and Martínez, our approach extends the weight design to include other factors. The difference between the design of the power diagram weights proposed in [1] and the weights designed in this thesis is as follows. In [1], the weight depends only on one factor which is the energy consumption, whereas in this thesis, the weight depends on multiple factors that are reflecting the main goal of the overall framework and they are also reflected in the design of the other framework modules. Furthermore, since multiple factors are included in the weight parameter w_i, each component is normalized between 0 and -1 so that the different factors are comparable and easy to tune. The normalizing range is between 0 and -1 such that when all the components that make up w_i have a value of zero, the weight is not factored into the segmentation process and the environment is segmented according to the distance between the discrete points in space q and the seed point p_i. When $w_i = 0$, the power diagram becomes equivalent to the standard Voronoi diagram.

The overall weight that is used in the proposed power diagram performance function is given by a summation of the individual factors w_e, w_v and w_s as follows,

$$w_i = \alpha_1 w_e + \alpha_2 w_v + \alpha_3 w_s, \tag{4.1}$$

where w_e is a component that represents the energy consumption, w_v represents the maximum velocity of a robot, w_s represents the on-board sensors' performance, and α_1, α_2, and α_3 are positive design parameters used to prioritize one weight component over the others. When α_1, α_2, or α_3 are increased, more importance is given to weight w_i over the physical Euclidean distance between each discrete point in the environment q, and the seed point, p_i in (2.3). The effect of each design parameter over the individual weight components is as follows. By increasing either α_1, α_2, or α_3 an emphasis on the respective component is placed. For instance, by increasing α_1, the effect of the energy consumption is more heavily pronounced on the environment segmentation such that robots with a lower energy level are given a smaller area to cover.

Additionally, the more negative a weight is, the larger the "distance" $\left(\| q - p_i \|^2 - w_i\right)$ is. If a point q is equidistant from p_i and p_j, that point q will be clustered with the robot that has a less negative weight. In this example if $w_j > w_i$, then q shall belong to the region of p_j. Therefore, the more negative w_i is, the smaller the area of the i^{th} robot that is generated in the power diagram. The significance of each factor in w_i shall now

be explained.

One component that affects how the environment is segmented is the current energy levels of the robots. In this thesis, one must note that the energy level E_i of a robot is unitless since in general it does not have to represent actual energy in $Joules$. Rather, E_i is related to the energy reserve of the i^{th} robot in terms of, for example, how much battery a mobile robot has left, which represents a level of its autonomy and activeness.

The more energy consumed, the more negative w_e is and therefore, the smaller the area generated for that robot should be. This follows the reasoning expressed by Kwok and Martínez [1]. If a robot is low on its energy reserves, its coverage workload should be reduced. In this work this is done in two ways: by assigning that robot with a smaller area to cover, and by limiting its acceleration through its controller. The first is achieved by incorporating the energy awareness in the environment segmentation module. Additionally, $w_e = [0, -1]$, meaning that when $w_e = 0$ the robot has full energy and when $w_e = -1$, the robot has no energy left. For this purpose, the energy-dependent, normalized weight component proposed is given by,

$$w_e = \frac{E_i - E_{i(max)}}{E_{i(max)} - E_{i(min)}}. \tag{4.2}$$

Using the same definitions as in (3.3), $E_{i(max)}$ and $E_{i(min)}$ represent the maximum and minimum energy level of the i^{th} robot, and E_i is the current energy level (i.e. the energy reserve) of the i^{th} robot.

Another factor that is made to affect the size of the area generated is the actuator metric. This stems from the premise that a robot is more able to cover a larger area if its actuators perform better than others. For example, if a robot has a higher maximum velocity than other robots, it can typically cover a given area faster. Hence, it may be given a larger area to cover. The actuator metric chosen for this purpose is the robot's velocity, and hence, the velocity-dependent, normalized weight component is given by,

$$w_v = \frac{v_i - \max(\boldsymbol{v})}{\max(\boldsymbol{v}) - \min(\boldsymbol{v})}, \tag{4.3}$$

where v_i is the maximum velocity of robot i, $\boldsymbol{v}$ is the vector containing all the maximum velocities of all the robots in the team, $\max(\boldsymbol{v})$ is the highest maximum velocity among the team of robots, and $\min(\boldsymbol{v})$ is the smallest maximum velocity among the team. In

addition, for any robot if $w_v = 0$ it means that that particular robot is the fastest in the team, while if $w_v = -1$ it is the slowest robot.

Finally, the last factor considered to affect the environment segmentation in the proposed module is the effect of the sensors' performance parameters. Some works encode this effect as its own individual range-limited Voronoi diagram [146–149]. The disadvantage of such approaches is that the regions are usually generated as the intersection of the Voronoi partition with a disc whose radius depends on the sensor range. This may leave parts of the environment empty and unassigned to any of the robots. Although such a notion works well for stationary sensors, this becomes a limiting factor in a mobile multi-robot system (MRS).

It is important to note that even though an area may exceed the range of the robot's sensors, that robot may still reach all parts of its region by moving around. By including the sensors' performance as a weight component in a power diagram, this ensures that the portion of environment generated for a robot is in accordance to its sensors capabilities. Furthermore, since the environment segmentation is done after the robot-to-density allocation, only the performances of the sensors used by that robot to cover that density are considered. The reasoning behind this is that sensors which shall not be used by the robot to cover the allocated region have no bearing on the overall size of the region. For this purpose, the proposed sensors-dependent, normalized weight component is given by,

$$w_s = -\frac{1}{n_s} \sum_{k=1}^{n_s} s_k. \tag{4.4}$$

where n_s and s_k are the number of robots in the team and the performance cost of each sensor that is in use, as used in (3.4). If $w_s = 0$ it means that the sensors of that particular robot have the best performance, and if $w_s = -1$ it means that that robot's sensors have the worst performance.

It is important to note that all parameters affecting the above-mentioned weight components, namely E_i, v_i and s_k, are assumed to be deterministic and known at all times. Hence, it is beyond the scope of this work to consider these parameters as stochastic. Such variation may be considered in potential extensions of this work.

Furthermore, the cost function that can be used to measure how well an environment

is covered, based on the proposed power diagram, becomes,

$$\mathcal{H}_w\left(P, E, \mathcal{V}\right) = \sum_{i=1}^{n_r} \int_{V_i} \left(\parallel q - p_i \parallel^2 - w_i \right) \phi_i(q) \, \mathrm{d}q \tag{4.5}$$

where $E = \{E_1, E_2, \ldots E_i\}$ is the set of energy levels for robots $\{1, 2, \ldots i\}$, respectively, and ϕ_i is the density allocated to robot i and

$$w_i = \alpha_1 \left(\frac{E_i - E_{i(max)}}{E_{i(max)} - E_{i(min)}} \right) + \alpha_2 \left(\frac{v_i - \max(\boldsymbol{v})}{\max(\boldsymbol{v}) - \min(\boldsymbol{v})} \right) - \frac{\alpha_3}{n_s} \sum_{k=1}^{n_s} s_k \tag{4.6}$$

One must note that (4.5) is minimized through the iterative Lloyd's algorithm as presented in Algorithm 2.1 in Chapter 2.

In effect, the resulting power diagram clusters the discrete environment points q based on: the distance that depends on the physical distance of each point q from the seed point p_i, as well as the distance encoded as the summation of an energy-dependent component, a velocity-dependent component and a sensor-dependent component. If a discrete point q is physically closer to p_i rather than to p_j, but the energy level of the robot at p_j is higher than that at p_i, the sensors of robot j perform better than those of robot i and robot j has better actuators than robot i, then that discrete point shall be allocated to robot j located at p_j.

4.3 Comparison to standard Voronoi segmentation

When the environment is segmented using a standard Voronoi diagram, the process considers just the distance between each discerete point in the environment q, and the seed point (i.e. the robot position p_i in the case of coverage control). Hence, there is no consideration for other factors that actively and practically affect the performance of the multi-robot system (MRS) as it performs coverage of an environment. As has been argued previously, by using a power diagram a number of practical factors that are not related to the robot's position may be included in the process of the environment segmentation. In our opinion, a coverage control framework benefits greatly from this consideration, since all the modules in the framework are designed for a common goal, that of optimizing coverage, subject to a number of situational constraints.

To demonstrate the benefit of using a power diagram over the standard Voronoi diagram in the proposed framework, consider the initial conditions and parameters in Table 4.1. Lloyd's algorithm is first executed with the standard Voronoi diagram as the environment segmentation module, and then, it is once again executed using the power diagram proposed in this chapter. The resulting robot areas at CVT, corresponding to each experiment are tabulated in Table 4.2 and illustrated in Figure 4.1.

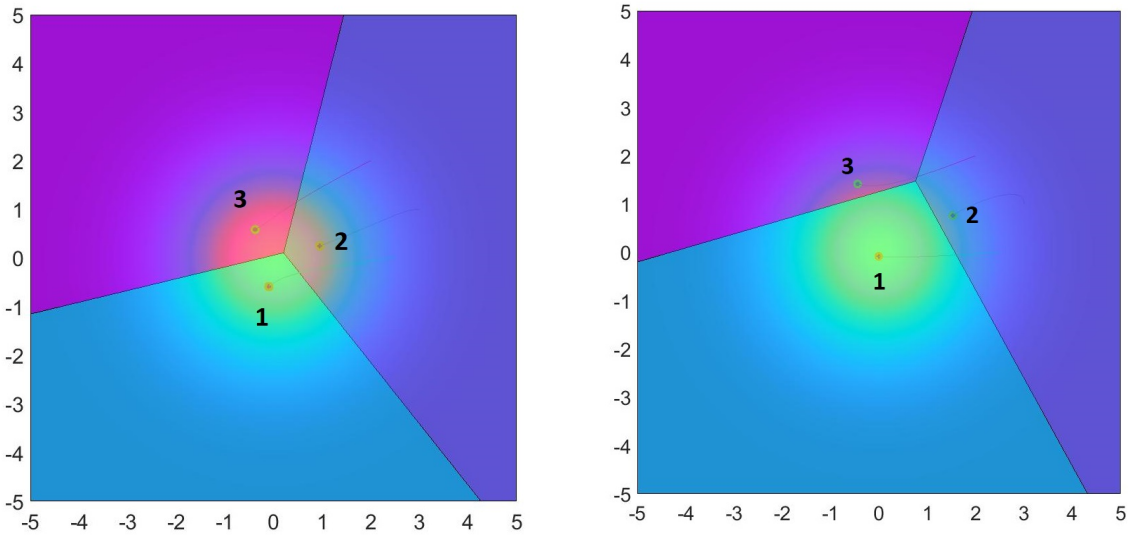
Table 4.1: Robot parameters during the environment segmentation process

Robot	**Initial position**	**Initial Energy levels**	**Maximum Velocity,** $\mathrm{m\,s^{-1}}$	**Overall sensors performance (**w_s **in (4.4))**
R_1	(2.5, 0)	20	10	-0.1
R_2	(3, 1)	15	3	-0.8
R_3	(2, 2)	15	5	-0.7

Table 4.2: Results - area of each Voronoi cell

Robot	**Standard Voronoi diagram cell area in** $\mathrm{m^2}$	**Power diagram cell area in** $\mathrm{m^2}$
R_1	33.67	41.904
R_2	34.51	29.79
R_3	31.8	28.31

The first significant difference to note in these results is how balanced the areas are generated when using the standard Voronoi diagram. By using the standard Voronoi diagram, the CVT configuration of the Voronoi cells results in comparable regions. This is most definitely unlike the power diagram, where, as can be seen in Table 4.2 and in Figure 4.1b, the areas of the Voronoi cells are un-balanced among the robots in the team. This occurs since through the weights, the power diagram generates the largest area for the most capable robot. Tied to this observation, the second aspect to notice about this result is that, as expected, when using the standard Voronoi diagram, the more capable robot R_1 is not being given a larger area (and hence a bigger workload). This means that the less capable robots would have to work as hard as R_1. This is not beneficial for a



(a) CVT generated using the standard Voronoi segmentation

(b) CVT generated using proposed power diagram segmentation

Figure 4.1: Comparison of a CVT achieved with the standard Voronoi diagram and with the proposed power diagram

framework that strives to optimize and balance the workload among the robots according to their capabilities. Hence, using a standard Voronoi diagram in the proposed framework works against the holistic design of the other modules that value energy, sensor and actuator-awareness. In contrast to Figure 4.1a, one can observe that the power diagram in Figure 4.1b allows robot R_1 to have the largest area and as a consequence, cover a bigger section of the higher important region. Hence, in the power diagram result of Figure 4.1b, R_1 has a larger Voronoi cell mass than the other robots, and also a larger mass when compared to itself when a standard Voronoi diagram is used. This is because in this experiment the parameters of R_1 were designed to make it the most capable robot in the team.

4.4 Validation of the proposed approach

The purpose of this section is to show that the proposed power diagram is achieving its objectives. In this light one set of tests aims to show that the environment is truly being segmented according to the current capabilities of the robots, while the second set aims to validate the effect of the priority gains of the individual weight components on the segmentation process. It is important to note that in these validation test cases,

the positions of the robots also affect the overall environment segmentation due to the $\| q - p_i \|^2$ component in (2.3).

These validation tests require the pre-setting of the following robot parameters: initial energy level, maximum velocities, overall sensor performance, and initial positions. Note that the energy quantities are unitless since the validation process is only concerned with the relative levels of energy among the robots in the team. Using the set energy levels and maximum velocities, the weights w_e and w_v are calculated according to (4.2) and (4.3) respectively. For simplicity, and for easier human-verification, (4.4) is pre-computed and tabulated in the robots' parameters tables. Lloyd's algorithm with a simple proportional controller as described by (2.13), is then executed until a centroidal voronoi tessellation (CVT) is achieved. The resulting area for each Voronoi cell allocated to each robot is then calculated using the `polyarea` function in MATLAB®. The units of the generated area once again depend on the application. For the following validation test cases, an area of 10 metres by 10 metres is considered and hence the area calculated is in m^2. In addition to the area, the mass of each Voronoi cell M_{V_i} is computed for comparison using Equation (2.5). For the purpose of these validation simulations, one static density function is considered as the only importance region of the environment, and therefore $\phi(q)$ is the only density that shall be covered by the multi-robot team. In this case, the allocation algorithm detailed in Chapter 3 is not in effect.

Two sets of simulations are considered in this study. In the first set of simulations, the effect of the weight w_i on the Voronoi cell polygon area and mass is evaluated through a single trial analysis, a multiple trial analysis and a general Monte Carlo analysis. In the second set of simulations, the effects of the priority gains α_1, α_2 and α_3 on the Voronoi cell polygon area and mass are evaluated.

4.4.1 Analyzing the effect of w_i using simulations

For this experiment, the first test case uses a specific set of parameters (initial energy levels, maximum velocities and overall sensors performance cost) with known pre-set initial robot positions. The objective of this test is to validate the design of the weights used in the proposed power diagram. More specifically it aims to ensure that for a relatively larger weight, the power diagram generates a corresponding larger Voronoi cell. The second test considers the same set of robot parameters except for the initial positions.

Ten trials with randomly generated initial positions are investigated individually to analyse the effect of different initial positions on the generation of the CVT Voronoi cells in more detail. The third test considers the same set of robot parameters, but employs a Monte Carlo simulation where the initial positions of the robots are generated randomly. The aim of this validation test is to reduce the effect of the robot positions on the test results. For this validation process consider $\alpha_1 = \alpha_2 = \alpha_3 = 1$, such that all the individual weight components have equal priority. In the following test results, the size of the Voronoi regions is evaluated at CVT and a single, static Gaussian density function is used as the importance function. The controller of choice for this experiment is the simple proportional controller in (2.13) with gain $k_{prop} = 0.3$.

Test Case 1 - Single trial analysis

For this single-trial analysis, consider three robots having the parameters tabulated in Table 4.3. These parameters are the initial conditions before the robots start engaging in Lloyd's algorithm and move around the environment to achieve CVT. All three robots have a maximum energy level $E_{i(max)} = 20$ and a minimum energy level $E_{i(min)} = 0.5$.

Table 4.3: Robot parameters during the environment segmentation process

Robot	**Initial position**	**Initial Energy levels**	**Maximum Velocity,** $\mathrm{m\,s^{-1}}$	**Overall sensors performance (**w_s **in (4.4))**
R_1	(2.5, 0)	20	10	-0.1
R_2	(3, 1)	15	3	-0.8
R_3	(2, 2)	15	5	-0.7

Consequently, the initial weights calculated (prior to the execution of Lloyd's algorithm), after computing (4.2) - (4.1) are as tabulated in Table 4.4.

Table 4.4: Initial weights calculated prior to the execution of Lloyd's algorithm

Robot	$\boldsymbol{w_e(0)}$	$\boldsymbol{w_v(0)}$	$\boldsymbol{w_s(0)}$	$\boldsymbol{w_i}$
R_1	0	0	-0.1	-0.1
R_2	-0.2564	-1	-0.8	-3.082
R_3	-0.2564	-0.714	-0.7	-2.696

The test was designed such that R_1 is the robot that remains with the highest weight throughout the duration of the trial, hence at CVT, R_1 is expected to have the largest area. It is important to note that as the robots move in the environment they consume energy, and therefore the overall weight is continuously changing. In an attempt to reduce this effect of, robot R_1 was set to have the highest maximum velocity and best sensors performance, since these two parameters do not vary as the robots move to a CVT. Furthermore, the initial positions of the robots are set close to one another such that the robots more or less consume similar levels of energy as they achieve CVT. This allows an easier human-validation process. Additionally, the Voronoi cell mass M_{V_i} is computed and compared since as a consequence of having a large cell area, the mass of the cell is expected to be large as well. To evaluate this result, the mass of each Voronoi cell is evaluated according to the Voronoi mass equation in (2.5).

The resulting mass of each Voronoi cell as well as the overall polygon area calculated for each robot at CVT are tabulated in Table 4.5. As expected, robot R_1 has the largest generated area. Since the parameters of R_2 and R_3 were set similar to one another, as expected their resulting generated areas are similar as well, based on the presumption that the robots started from a similar initial position and hence consumed a similar amount of energy to get to CVT. Additionally, since R_1 remained the robot with the highest weight throughout the trial, its intersection with the higher probability areas of the density function was always larger than that of the robots. As a consequence, the centroid of R_1 was always pushed towards the densest part of the Voronoi cell, resulting in R_1 having the largest Voronoi cell mass at CVT.

Table 4.5: Results - area of each Voronoi cell

Robot	**Final weight**	**Mass of Voronoi cell,** M_{V_i}	**Area of Voronoi cell in** m^2
R_1	-0.15	2.93	44.01
R_2	-2.07	0.08	28.78
R_3	-1.72	0.14	27.21

Figure 4.2 illustrates the environment at CVT at the end of the trial. As can be seen, R_1 has the largest computed area as well as the the largest intersection with the higher probability areas of the density function. This conforms with the final weights at CVT, where R_1 is seen to have the largest weight, i.e. the weight that is closest to 0.

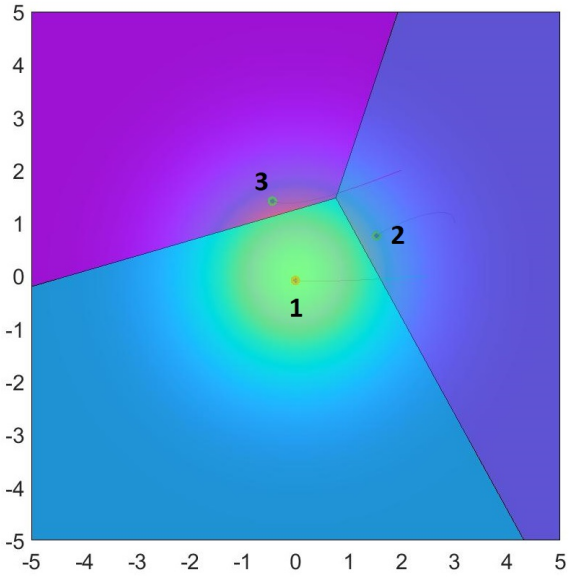


Figure 4.2: A plot of the environment at CVT showing that R_1 is the robot with the largest (and most important) generated area.

Test Case 2 - Multiple trial analysis

In this test, ten simulations are analysed and the individual results from each simulation are recorded. The purpose of this test is to compare the resulting areas and Voronoi cell masses among the robots in the team when the robots start the simulation trials from randomly generated initial positions. The weights in Table 4.4 are used in this test as the initial conditions of each trial. The same experimental procedure followed in Test 1

is used. Table 4.6 tabulates the robots' initial positions and the respective final area generated for each robot for the 10 trials.

Table 4.6: Multiple trial analysis - area of each Voronoi cell

Trial	**Robot**	**Initial position**	**Final weight**	**Mass of Voronoi cell,** M_{V_i}	**Area of Voronoi cell in** m^2
2.1	R_1	(3.15, 4.13)	-0.58	2. 86	44.66
	R_2	(4.06, 1.32)	-2.32	0.11	27.34
	R_3	(-3.73, -4.02)	-1.99	0.17	27.99
2.2	R_1	(-2.22, 4.65)	-0.52	2.89	40.82
	R_2	(0.47, -3.42)	-2.23	0.11	30.9
	R_3	(4.58, 4.71)	-2.02	0.14	28.28
2.3	R_1	(4.57, -3.58)	-0.44	2.85	41.24
	R_2	(-0.15, -0.78)	-2.08	0.12	30.38
	R_3	(3, 4.16)	-1.86	0.16	28.38
2.4	R_1	(2.92, -4.64)	-0.62	2.84	42.68
	R_2	(4.59, 3.49)	-2.32	0.11	29.5
	R_3	(1.56, 4.34)	-1.96	0.195	27.81
2.5	R_1	(1.79, -1.08)	-0.21	2.93	37.31
	R_2	(2.58, 1.55)	-2.13	0.08	29.71
	R_3	(2.43, -3.29)	-1.75	0.13	32.98
2.6	R_1	(2.06, -4.54)	-0.52	2.87	42.16
	R_2	(-4.68, -4.03)	-2.35	0.09	26.55
	R_3	(-2.23, 3.23)	-1.88	1.18	31.29
2.7	R_1	(1.95, -4.66)	-0.4	2.88	44.46
	R_2	(-1.83, -0.61)	-2.13	0.11	26.94
	R_3	(4.5, -1.18)	-1.87	0.15	28.6
2.8	R_1	(2.66, -0.1)	-0.25	2.94	32.48
	R_2	(2.95, -0.54)	-2.17	0.07	31.29
	R_3	(-3.13, 1.46)	-1.78	0.13	36.22
2.9	R_1	(2.09, 1.8)	-0.27	2.93	40.93
	R_2	(2.55, 1.55)	-2.19	0.08	30.36
	R_3	(-2.24, -3.37)	-1.82	0.13	28.71
2.10	R_1	(-3.81, -1.6)	-0.31	2.91	45.08
	R_2	(-0.01, 0.85)	-2.08	0.11	27.75
	R_3	(4.6, -2.76)	-1.87	0.13	27.16

These results show that in nine out of ten trials R_1 had the largest operating area at CVT, when compared to the other two robots. This is expected, since R_1 was set out to be the most capable robot from the onset. However, in Trial 2.8, one can note that

R_1 did not end up with the largest operating area at CVT, even though it still ended up with the highest area mass. It is important to note that this is because the power diagram metric does not only depend on the weights associated with each robot. The metric also depends on the distance between each discrete point q and the position of the robot p_i, that is $\| q - p_i \|^2$. This distance metric had a bigger affect on the environment segmentation than the weights of the robots. Nevertheless, since R_1 started as the robot with the highest weight, it was initially exposed to a higher importance area, and this led Lloyd's algorithm to push R_1 to the denser parts of the density function. Consequently, at CVT one can observe that R_1 has the largest Voronoi cell mass. In this trial, one observes that although R_1 is now responsible of covering a denser region (in terms of importance), its polygon area is made smaller in an attempt to reduce its overall workload in terms of navigation work.

This result highlights the effectiveness of this weight-based design since such situation-dependent behaviour allows the multi-robot system (MRS) to adapt itself to various scenarios, such as a heavier energy consumption due to longer distances travelled. Additionally, R_2 and R_3 always had comparable areas in the final CVT as their overall weights and capabilities were always comparable. The differences in their resulting areas and cell masses are attributed to their different initial positions and hence, to the small discrepancies in weight due to the different amounts of energy consumed. Furthermore, one must note that the distance factor in (2.3) would take precedence if the weights of two neighbouring robots are similar. To ensure that the weight w_i always takes precedence over the distance factor $\| q - p_i \|^2$, the priority gains α_1 to α_3 should be set such that the w_i has a bigger value than the expected largest distance between the points. Setting the priority gains sufficiently high, shall always ensure that the weights override the distance factor.

Test Case 3 - Monte Carlo general results

To further analyse the behaviour of the power diagram for different initial robots positions, a Monte Carlo simulation of 100 trials is performed. The initial conditions in Table 4.3 are employed in this experiment, except for the initial positions of the robots, which are randomly generated in the given environment. The density function used in these simulations is the same static, Gaussian probability density function used in Tests 1 and 2. In each

trial, the final area and the final weights of the robot at CVT are recorded. A boxplot of the final areas and another of the final weights at CVT are shown in Figures 4.3 and 4.4, respectively.

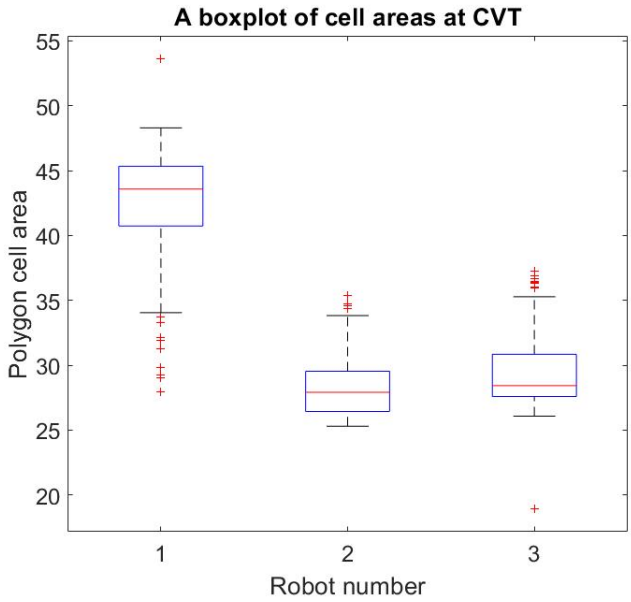


Figure 4.3: A boxplot of the final areas at CVT for each robot for 100 trials

As can be seen from Figure 4.3, for the particular pre-set parameters, R_1 generally has the largest area to operate in when CVT is reached, even when the initial positions of the robots are totally random. This conforms with the values of the weights as presented in the boxplot in Figure 4.4, that illustrates the final weights of each robot at CVT for each trial. From Figure 4.4 it is evident that robot R_1 has the largest weight, R_2 had the smallest weight and R_3 lied in between. If one corresponds these weights to the final areas in Figure 4.3, one observes that R_1 generally has the largest area, R_2 has the smallest area and R_3 is in between the two, as expected. This ranking is also reflected in the mean mass of the Voronoi cells of each of the robots. This is further shown in Table 4.7 that tabulates the mean weight, area and mass at CVT for every robot for the 100 trials.

Table 4.7: Mean of the weights and areas of the Monte Carlo study

Robot	**Mean weight**	**Mean area**	**Mean mass**
R_1	-0.18	42.09	2.89
R_2	-2.12	28.41	0.09
R_3	-1.74	29.5	0.16

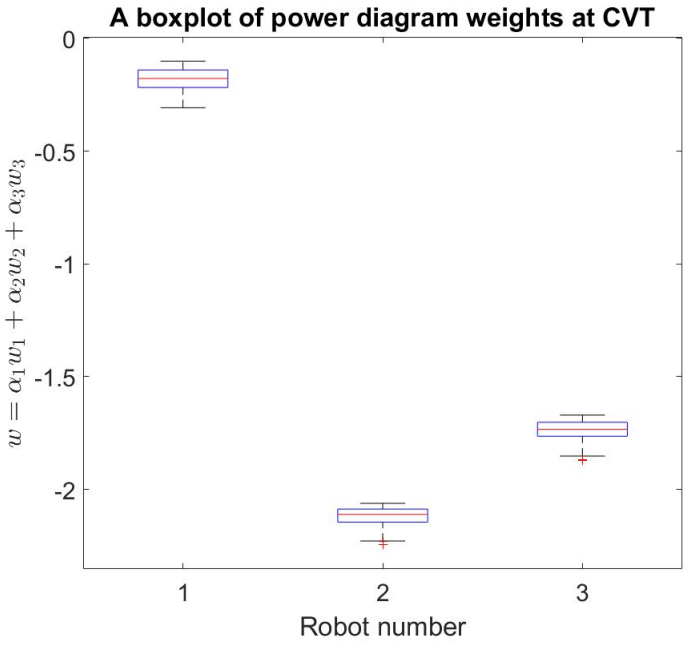


Figure 4.4: A boxplot of the final weights at CVT for each robot for 100 trials

4.4.2 Effect of the weight priority gains α_1, α_2 and α_3

This test case shall analyse the effect of the priority gains of Equation (4.1) over the segmentation process. The test parameters tabulated in Table 4.8 are used, together with $E_{i(max)} = 20$ and $E_{i(min)} = 0.5$ for all $i = \{1, \ldots 4\}$. For this test case, four robots are placed equidistantly from the centre of mass of the density function that they must cover. This was done such that the robots follow a similar trajectory to get to a CVT, and hence consume a similar amount of energy during their transit.

Table 4.8: Robot parameters during the environment segmentation process

Robot	**Initial position**	**Initial Energy levels**	**Maximum Velocity,** $\mathrm{m\,s^{-1}}$	**Overall sensors performance (**w_s **in (4.4))**
R_1	(-4, 4)	20	8	-0.5
R_2	(4, 4)	10	5	-0.9
R_3	(4, -4)	15	10	-0.1
R_4	(-4, -4)	10	5	-0.5

The first step of this experiment is to run the environment segmentation module

where all the priority gains are equal. This allows for a fair comparison of results when the priority gains are changed to prioritize one of the weight components. In order to reduce the overall effect of the distance factor ($\| q - p_i \|^2$) in (2.3), initially the priority gains are set as: $\alpha_1 = \alpha_2 = \alpha_3 = 5$. This ensures that the weight w_i reduces the influence of the distance factor in the environment segmentation process of the power diagram, as argued previously. The initial weights computed during the first iteration are tabulated in Table 4.9.

Table 4.9: Initial weights of robots when $\alpha_1 = \alpha_2 = \alpha_3 = 5$

Robot	$\boldsymbol{w_e(0)}$	$\boldsymbol{w_v}$	$\boldsymbol{w_s}$	$\boldsymbol{w_i(0)}$
R_1	0	-0.4	-0.5	-4.5
R_2	-0.51	-1	-0.9	-12.06
R_3	-0.26	0	-0.1	-1.782
R_4	-0.51	-1	-0.5	-10.06

Under these conditions, the resulting areas at CVT are as tabulated in Table 4.10, where t_f is the final time stamp at the end of the trial, and $E_i(t_f)$ and $w_i(t_f)$ are the final energy level and weight values of the i^{th} robot at CVT. By examining the weights $w_i(t_f)$ one can observe that robot R_3 has the highest overall weight at -2.85 and therefore the area of its Voronoi cell is the largest area among the team. Since the area is significantly larger than that of the other robots, there is a larger overlap with the points q that have higher importance. This results in the largest Voronoi cell mass across the whole team. One may observe that even though R_1 has the highest energy level E_i it does not have the largest Voronoi cell area. This is because of the influence of the other components that make up its weight w_i.

Table 4.10: Results - area and mass of each Voronoi cell

Robot	$\boldsymbol{E_i(t_f)}$	$\boldsymbol{w_i(t_f)}$	**Mass of Voronoi cell** M_{V_i}	**Area of Voronoi cell in** m^2
R_1	16.499	-5.4	0.05	24.46
R_2	7.94	-12.59	1.59×10^{-5}	13.93
R_3	10.83	-2.85	3.09	45.57
R_4	7.68	-10.66	1.22×10^{-4}	16.03

The second phase of the experiment involves prioritizing one of the weight components. In this test case, the energy component is prioritized, such that its weight gain α_1 is set to 20. The values of α_2 and α_3 remain the same at 5 units. The initial weights of each robot in this scenario are tabulated in Table 4.11.

Table 4.11: Initial weights of robots when $\alpha_1 = 20$ and $\alpha_2 = \alpha_3 = 5$

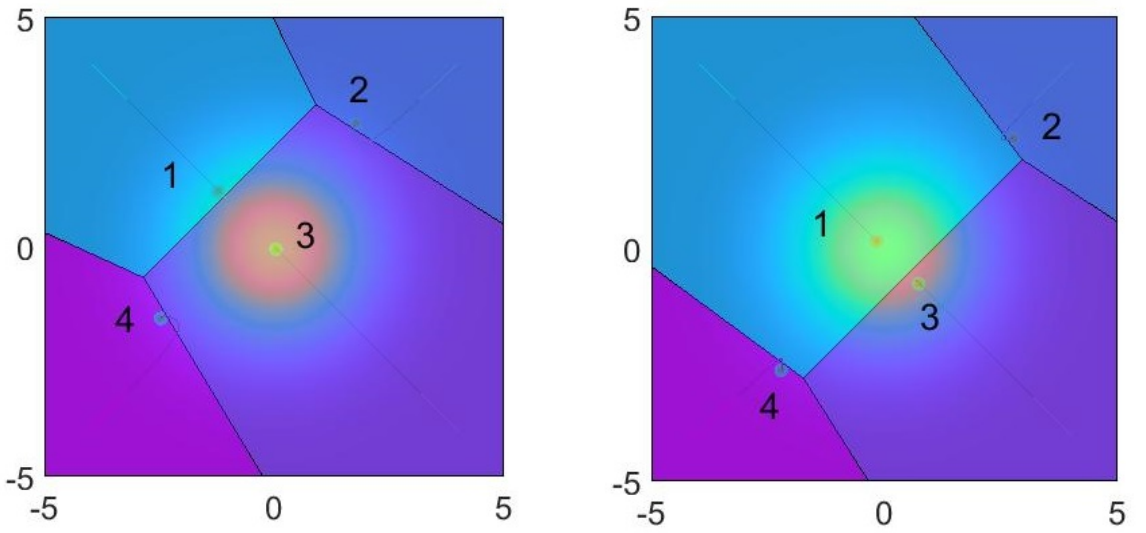
Robot	$\boldsymbol{w_e(0)}$	$\boldsymbol{w_v}$	$\boldsymbol{w_s}$	$\boldsymbol{w_i(0)}$
R_1	0	-0.4	-0.5	-4.5
R_2	-0.51	-1	-0.9	-19.76
R_3	-0.26	0	-0.1	-5.63
R_4	-0.51	-1	-0.5	-17.76

The resulting areas and Voronoi cell masses for each robot are tabulated in Table 4.12. By placing an emphasis on the energy component, one observes that even though the energy consumed by each robot is comparable to the energy consumed in the previous experiment, the robot that has the largest Voronoi cell area is now R_1. This is because the larger value of α_1 is penalizing lower energy levels of the robots with more aggression than in the previous experiment. Therefore, since R_3 has a lower energy level at CVT (of 11.27) than R_1, it is now penalized more and therefore gets a smaller Voronoi cell area when compared to its Voronoi cell in the previous experiment. In the previous experiment, R_3 had an area of 45.57 while in this experiment it has an area of 32.74, while the final energy level of this robot remain comparable at around 11 units in both experiments. This shows that in the second experiment, R_3 is being more aggressively penalized for its lower energy level when compared to R_1 such that its overall weight now yields a smaller area than that of R_1.

Table 4.12: Results - area and mass of each Voronoi cell

Robot	$\boldsymbol{E_i(t_f)}$	$\boldsymbol{w_i(t_f)}$	**Mass of Voronoi cell** M_{V_i}	**Area of Voronoi cell in** m^2
R_1	16.01	-8.59	2.67	43.44
R_2	8.5	-21.29	7.14×10^{-7}	11.13
R_3	11.27	-9.45	0.472	32.74
R_4	8.28	-19.52	4.54×10^{-6}	12.69

Figure 4.5 illustrates the difference in the two test scenarios described above. This further illustrates that by increasing α_1 to 20, robot R_3 was penalized more for its lower energy component and hence, is given a smaller area to cover. In conclusion, the larger the value of the priority gain, the more cautious the segmentation becomes about its corresponding component, thus prioritizing the importance of that component in comparison to the other components.



(a) Equal priority gains

(b) Priority placed on the energy component w_e

Figure 4.5: Comparing CVTs with equal priority gains and unequal priority gains

In both experiments, robots R_2 and R_4 both had comparable weights and therefore, their resulting Voronoi cell areas are also comparable. The weights of both R_2 and R_4 were significantly smaller than those of R_1 and R_3 and therefore the final cell areas and masses at CVT are significantly smaller than those of R_1 and R_3.

4.5 Conclusions

This chapter details the design of the power diagram environment segmentation scheme designed for the proposed coverage control framework. A weighted Voronoi diagram, such as the power diagram, is necessary in such a framework since it contributes to the energy, actuator and sensor capability awareness of the framework. Of course, these are mere examples of the factors that the framework may be made to account for. In fact,

the proposed segmentation scheme is straightforward enough to be modified and account for other factors that might be required in a different application. Additionally, each factor that the power diagram accounts for may be prioritized over the other factors in the scheme, depending on the application. For example, a coverage operation that is estimated to last for a long time must aggressively prioritize energy conservation of the robots over the other factors that affect the power diagram weight w_i, such that the multi-robot system (MRS) is able to contribute effectively to the operation for longer.

The weighted Voronoi diagram of choice is the power diagram, as it generates convex polygons that are easy to handle in a coverage control application. The factors of choice that affect the generation of the regions are embedded in the overall weight used in the power diagram. This particular design considers the energy consumption, the maximum velocity, and the sensors' performance of the robot, as the factors of interest. The motivation for this choice stems from the fact that a more capable robot is considered as one that has a lower energy consumption, a higher maximum velocity and better performing sensors. Together these factors allow the robot to cover a larger area (and thus, taking a larger workload) better. Moreover this selective segmentation allows weaker robots to conserve their energies and remain operational for longer.

This chapter addresses the importance of using a power diagram over a standard Voronoi diagram in a framework that seeks to optimize the practical restrictions of the robot team. A study with both the standard Voronoi diagram and the designed power diagram shows that balancing the workload among robots with different capabilities is only achievable through a power diagram. Additionally, this chapter presented a number of tests that study and validate the proposed design. A number of important conclusions may be drawn from these tests. Firstly, one can observe that if a robot is superior in its capabilities and energy in relation to its team members, its final area allocated at CVT is almost always bigger than that of the other robots. Of course, this is not always the case. If a robot must travel much longer distances, it ends up consuming a larger amount of energy and at CVT it can no longer be the most capable robot. In such a case, this robot does not end up with the largest area. Since the weight of the power diagram depends on a component that varies with time, one can observe that this environment segmentation is constantly being updated to represent the current status of the robots.

Secondly, it is important to note that the segmentation of the environment does not

only depend on the capabilities that affect the weight w_i. It also depends on the current position of the robot p_i. If the application deems it necessary that the weights must be the driving factor in the segmentation process, then the priority gains α_1, α_2 and α_3 must be set large enough to make the $\| q - p_i \|$ term relatively insignificant. Thirdly, if one factor must be given priority over the others, the priority gains need to be adjusted such that the component that must be prioritized has a gain α that is larger than that of the other factors.

To the best of the author's knowledge, such an approach has not been adopted in existing works on area segmentation for coverage control. Kwok and Martínez [1] use a similar approach with power diagrams, however, they limit their design to just using the energy consumption as the only factor that affects the weight w_i. Although this appears simpler and easier to evaluate, it excludes other factors that the general framework is designed to account for. Furthermore, on including other components in the overall weight, it is important to normalize these components such that neither one has any numerical advantage over the other. Other works in literature [105–107, 111–113, 136, 146–149, 155] do not consider this multiple-factor optimization in the segmentation process of the environment. This makes the proposed approach novel as it addresses the practical need to include multiple factors effectively to cater better for practical applications in the process of environment segmentation in coverage control.

Chapter 5

Energy-aware coverage control with time-varying densities

5.1 Overview

A truly efficient coverage control framework ideally requires a robot controller that considers both the energy restrictions of the robot, as well as the time-variations in the environment in which the multi-robot team is operating. This chapter presents an energy-aware tracking controller that contributes to the energy efficiency of the proposed framework. The control algorithm attempts to conserve as much of the energies of the robots as possible. This is expected to yield better coverage in typical realistic scenarios where robots have limited energy for the task in question. Having an energy-aware controller means that the robots in the team are able to remain in operation (not have their energy levels depleted) for a longer time than non-energy-aware algorithms. This allows the team to have better overall coverage of the environment throughout the entire operation. Additionally, the energy-aware controller must also be a tracking controller such that the robots are able to accurately track the time-varying centroids determined by the environment segmentation module.

This chapter provides a detailed description of the energy-aware coverage controller designed to track the time-varying densities. Moreover, a number of simulation results validating the proposed energy-aware controller are presented. In these simulations, a single time-varying density function was used. This scheme is compared to non-energy-aware coverage schemes in a single trial experiment, and the benefits of its energy-efficient

features are investigated in depth. A more in-depth analysis is also done on the endurance of the multi-robot team when it runs the proposed energy-aware controller as opposed to the non-energy-aware algorithms. The proposed algorithm is designed to preserve the energies of the robots, hence allowing the robots to endure a longer operation. This is analysed in detail and compared to scenarios where non-energy-aware controllers are used. Moreover, a Monte Carlo simulation involving 100 trials is employed to validate and compare the proposed algorithm to a typical non-energy-aware coverage algorithm subject to a time-varying density function. Statistical hypothesis testing is then used to analyse the statistical significance from the Monte Carlo results. Our results show that in a realistic scenario, the energy-aware coverage control algorithm proposed in this thesis performs better than the algorithms found in literature. Additionally, the effect of the tuning parameter within the proposed controller is also investigated, and a set of results are presented that validate its function and inclusion. Finally, the proposed scheme is also compared to one of the most recent methods reported in literature, namely the energy-efficient time-varying control scheme proposed by Santos et al. [155].

Parts of the study presented in this chapter have been published in [249]:

5.2 An energy-efficient coverage control scheme

As discussed in relation to the other framework modules, the concept of energy awareness and awareness of the robot's environment, can also be taken into account in the design of the coverage control algorithm. One should note that there are only a limited number of works in literature that address coverage control in the presence of time-varying density functions. Moreover, only very few of these works introduce energy awareness in the coverage control algorithm. For instance, Santos et al. [155] propose a controller that is able to track a single time-varying density function, and at the same time minimize the control effort. However, this controller is computationally intensive due to the calculation of a particular derivative term $\frac{\partial C_i}{\partial p_i}$ and the optimization step that occurs on every iteration. Additionally, the scheme proposed by Santos et al. does not provide any form of tuning

mechanism. In contrast the algorithm proposed in this chapter has a design parameter $k_{i,e}$ that the designer can use to trade off energy preservation for the tracking performance.

More specifically, this work considers an environment Q, where the single density function denoting the importance region is time-varying $\phi(q,t)$, and the robots have limited energy. The energy-aware coverage control scheme discussed in this chapter uses the energy-aware power diagram proposed by Kwok et al. [1]. Specifically, w_i of the power diagram is set to $w_i = E_i - E_{i(max)}$. In [1], the design of the power diagram with these weights contributes to the minimization of (4.5) whereby in this chapter, the proposed power diagram contributes to the minimization of (5.1). In contrast to (5.1), in (4.5) ϕ is treated as time-varying. Inspired by the work in [1], and by factoring in the time-varying aspect of the density function, a new energy-aware coverage cost function $\mathcal{H}_{e,t}$ is proposed as follows,

$$\mathcal{H}_{e,t} = \sum_{i=1}^{n_r} \int_{V_i^e} \left(\| \, q - p_i \, \|^2 + (E_{i(max)} - E_i) \right) \phi(q,t) \, \mathrm{d}q \tag{5.1}$$

To manoeuvre the robots to the Voronoi centroids, after each environment segmentation computation, a novel energy dependent tracking control law, given in (5.3) is proposed. In form, this control law resembles that in (2.27), originally proposed by Cortés et al. [115], with the difference that the feedback gain k is replaced by $k_{i,e}\frac{E_i}{E_{i(max)}}$. This ensures that the constant of proportionality is auto-adjusted in real-time according to the energy of the respective robot due to the ratio $\frac{E_i}{E}$. The parameter $k_{i,e}$ serves as a tunable gain, for this energy dependent ratio. Similar to Figure 2.6, Figure 5.1 illustrates the block diagram that represents the feedforward plus feedback tracking control law that is proposed in this thesis. Following the block diagram in Figure 5.1, we get,

$$\begin{aligned} \dot{p}_i =& u_i \\ =& k_{i,e}\frac{E_i}{E_{i(max)}}(C_{V_i^e} - p_i) + \dot{C}_{V_i^e} \\ =& \dot{C}_{V_i} - k_{i,e}\frac{E_i}{E_{i(max)}}(p_i - C_{V_i^e}) \end{aligned} \tag{5.2}$$

Similar to the modification done by Cortés et al. [115], the term $\frac{\dot{M}_{V_i}}{M_{V_i}}$ is included in the proportional term of the controller to get the desired closed loop dynamics. This yields

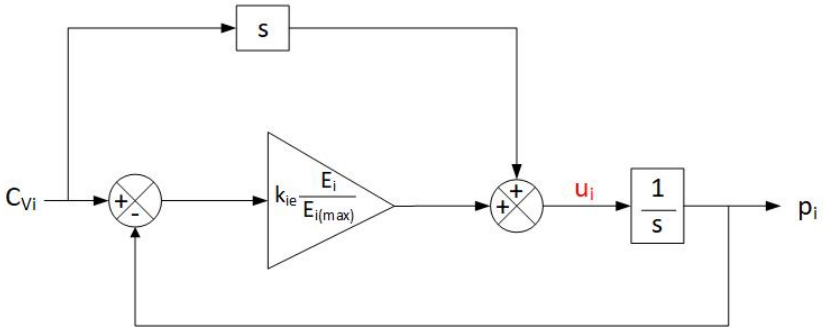


Figure 5.1: A block diagram of the feedback and feedforward energy-aware closed loop control

the following control law,

$$u_i = \dot{C}_{V_i^e} - \left(k_{i,e} \frac{E_i}{E_{i(max)}} + \frac{\dot{M}_{V_i^e}}{M_{V_i^e}} \right) \left(p_i - C_{V_i^e} \right), \tag{5.3}$$

In this control law V_i^e refers to the i^{th} robot energy-weighted Voronoi cell. Such a controller seeks to maintain tracking of the time-varying density functions but at the same time, seeks to preserve the energy levels of the robots through the energy-dependent term $\frac{E_i}{E_{i(max)}}$. This design allows the system designer to adjust the degree of energy awareness of the controller through $k_{i,e}$. Increasing $k_{i,e}$ makes the robots more willing to spend more energy in an attempt to improve the tracking error. However, the main aim of the controller is to preserve the energy levels of the robots while at the same time allow the robots to track the time-varying densities while energy is still available. By tuning $k_{i,e}$ the system designer would be tuning the overall feedback gain that affects how quickly the robot energy is spent, even though the controller remains energy-aware through the term $\frac{E_i}{E_{i(max)}}$. In contrast, in the work proposed by Cortés et al. [115], a constant gain k is set and the control law would not have any awareness of the energy expenditure.

The proposed formulation makes our coverage control framework energy conscious on two counts. Firstly, energy awareness is included in the segmentation of the environment by using power diagrams, and secondly, the control input of the robots $\dot{p}_i$, which is effectively their velocity, is restricted according to their current energy levels. Therefore, if the robots are low on energy, they are assigned smaller areas and they conserve energy by driving slower towards the target centroids.

The concept behind this formulation is that in an environment where the important

features are constantly moving, the team's energy should be conserved as much as possible in order to complete the task successfully and efficiently. As time passes, the energy of these robots starts to dissipate, to a point where some robots might end up with no energy at all. More specifically, when the energy of a robot reaches a preset minimum threshold $E_{i(min)}$, that particular robot is withdrawn from the team (considered completely inactive) and the environment needs to be covered by the rest of the team from that point in time onwards. This means that the number of robots n_r in the coverage control problem changes with time as some of the robots may eventually stop with depleted energy levels. Using an energy-aware tracking controller, such as the one proposed in this chapter, allows the robots in the team to operate for a longer period of time, hence making the overall coverage solution more efficient.

5.3 Comparison of the proposed energy-aware coverage control scheme to non-energy-aware coverage control schemes

In our simulation study, the proposed novel energy-aware coverage control scheme is evaluated and compared to the published algorithm in [115] as well as to the baseline Lloyd's algorithm [126] that uses a proportional controller with no energy considerations. The three different control laws used in this comparative analysis are:

$$\begin{aligned}
u_i &= -k_{prop}(p_i - C_{V_i}) && \text{Proportional controller} \\
u_i &= \dot{C}_{V_i} - \left(k_{TV} + \frac{\dot{M}_{V_i}}{M_{V_i}}\right)\left(p_i - C_{V_i}\right) && \text{Cortés's tracking controller} \\
u_i &= \dot{C}_{V_i^e} - \left(k_{i,e}\frac{E_i}{E_{i(max)}} + \frac{\dot{M}_{V_i^e}}{M_{V_i^e}}\right)\left(p_i - C_{V_i^e}\right) && \text{Proposed controller}
\end{aligned} \tag{5.4}$$

In our work it is assumed that the robots behave according to the holonomic model given in Equation (5.5), and have the energy dynamics described in Equation (5.6). Note that the energy dynamics in (5.6) are proportional to the kinetic energy of the robot. This is because the rate of change of the current energy level $\dot{E}_i$ is dependent on the kinetic energy of the robot. Therefore, the more the robot moves, the more energy it consumes

and therefore, the less the energy reserve E_i will become.

$$\dot{p_i} = u_i, \tag{5.5}$$

$$\dot{E_i} = \begin{cases} -m_i u_i^2 & \text{if } E_i \geq 0, \\ 0 & \text{if } E_i = 0, \end{cases} . \tag{5.6}$$

An analytical stability analysis showing that the proposed control law in (5.3) allows the system to converge to CVT, is presented briefly in Chapter 6, and fully included in Appendix A. This stability analysis proves that under certain assumptions, the proposed control law in (5.3) minimizes the framework cost function as presented in Chapter 6.

For the purpose of the following simulations, $k_{prop} = 1$, $k_{TV} = 1$ and $k_{i,e} = 1, \forall i = \{1, ..., n_r\}$. This is such that the capability and characteristics of each controller are compared fairly. One must also note that n_r varies during a simulation, depending on how many (if any) of the robots have stopped without energy. Additionally, the time-varying probability density function used in these simulations is defined in Equation (5.7). Note that q_x and q_y represent the (x, y) coordinate of an arbitrary point q in space, t is time, and $\frac{1}{\tau}$ is the frequency of the sinusoid. This probability density function is effectively a Gaussian function that moves across the environment in a sinusoidal fashion. This is also the reason why the coverage cost over time has a cyclic nature.

$$\phi(q, t) = e^{-\left(q_x - 2\sin\frac{t}{\tau}\right)^2 - q_y^2} \tag{5.7}$$

For the purpose of these simulations, the power diagram used to segment the environment when the proposed energy-aware controller is used, is designed based on the work in [1]. In this case, the weights $w_i = E_i - E_{i(max)}$. The actuator metric and sensors performance is not included in this study. This is done such that the energy awareness of the algorithm is more pronounced and validated in isolation. Validation of the power diagram design proposed in Chapter 4 was presented in Section 4.4. In this light, the energy-aware algorithm with a time-varying density function is provided in pseudocode in Algorithm 5.1.

First, a single trial analysis of the three schemes being compared is presented. These

three schemes are: 1) Lloyd's algorithm that uses a standard Voronoi diagram and a proportional controller, 2) the scheme proposed by Cortés et al. [115] that uses a standard Voronoi diagram and a tracking controller, and 3) the scheme proposed in this thesis that uses an energy-dependent power diagram and an energy-aware tracking controller. In this study, the overall energy-aware coverage cost $\mathcal{H}_{e,t}$ in (5.1) and the non-energy-aware coverage cost $\mathcal{H}_{c,t}$ in (5.8) are analysed. It is important to note that these costs factor in all the active robots from $i = 1$ until $i = n_r$. Therefore, any robots that have stopped without energy are not considered in the calculation of these costs. In addition, the energy consumption of all the team, as well as the error between the final robot position and the target destination centroid are analysed.

$$\mathcal{H}_{c,t} = \sum_{i=1}^{n_r} \int_{V_i^e/V_i} \left(\| q - p_i \|^2 \right) \phi(q,t) \, \mathrm{d}q \tag{5.8}$$

In the second part of the study, the endurance of the three schemes is investigated. In this simulation, the initial energy levels of the robots and the energy dynamics of these robots are modified in order to study for how long the robots are able to operate while covering an environment. The aim of this study is to show that using the proposed energy-aware controller, the robots are able to remain active for a longer time than when the non-energy-aware controllers are used. This concept of endurance is fundamental in a coverage control problem since allowing the robots to remain active for longer (by preserving their energies) means that a more effective coverage is achieved throughout the duration of the operation. Particularly, in this study the non-energy-aware coverage cost is compared for all of the three schemes to show that the endurance of the robot team is vital for efficient coverage.

In the third part of this simulation study, the non-energy-aware controller proposed by Cortés et al. [115] is compared to the energy-aware controller proposed in this thesis using Monte Carlo trials. To render the simulation study more powerful and realistic, the simulation is executed for a number of trials, each time varying a number of system parameters randomly. More specifically, for each trial the initial energy of the robots is set with random values between $E_{i(max)}$ and $E_{i(min)}$. For each trial run, the masses of the robots themselves are also varied in order to observe how different teams of robots would fare in the same environment. The proposed comparative Monte Carlo study compares

Algorithm 5.1. Single trial simulation code of the proposed energy-aware coverage control scheme.

procedure (p_i, Q)
  $n_r = 5$ ▷ number of robots n_r varies according to the number of active robots in the team
  $E_{max} = 2$
  $E_{min} = 0.1$
  $E_{i,all} = (E_{max} - E_{min}) \cdot rand(n_r, 1) + E_{min}$ ▷ random initial energies
  $weights = E_{i,all} - E_{max}$ ▷ power diagram weights
  $mass_{max} = 5$
  $mass_{min} = 1$
  $m_i = (mass_{max} - mass_{min}) rand(n_r, 1) + mass_{min}$ ▷ random robot mass
  $t_{final} = 125$
  $k_{prop} = 0.3$

  for $t = 0 : 0.5 : t_{final}$ **do**
    Compute values for ϕ for time t
    $[v, r] =$ PowerDiagram$(p_i, Q, weights)$ ▷ Computes energy-aware power diagram
    for $i = 1 : n_r$ **do**
      $C_{V_i^e} =$ PolyCentroid$(v(r_i), Q, \phi(q))$
      ▷ Simulate movement of robots and update the system states
      $\dot{p}_i = \dot{C}_{V_i^e} - \left(k_{i,e} \frac{E_i}{E} + \frac{\dot{M}_{V_i^e}}{M_{V_i^e}} \right) \left(p_i - C_{V_i^e} \right)$ ▷ robot closed-loop dynamics
      $\dot{E}_i = - \left(\dot{C}_{V_i^e} - \left(k_{i,e} \frac{E_i}{E} + \frac{\dot{M}_{V_i^e}}{M_{V_i^e}} \right) \left(p_i - C_{V_i^e} \right) \right)^2$ ▷ robot energy dynamics

      $p_{i_{new}} =$ ODEsolver $(\dot{p}_i, p_i)$
      $E_{i_{new}} =$ ODEsolver $(\dot{E}_i, E_i)$
      if $E_i \geq E_{min}$ **then**
        update $p_i = p_{i_{new}}$
        update $E_i = E_{i_{new}}$
        update $weight_i = E_i - E_{max}$
      else
        $\dot{p}_i = 0$ ▷ Stop movement of the robot
        $\dot{E}_i = 0$
      end if
    end for
  end for
  return p_i, E_i
end procedure

the cost in Equation (5.1) for the proposed energy-aware algorithm, for a time-varying density function $\phi(q,t)$, with an algorithm that does not account for the limited energy of the robots, namely the algorithm proposed in [115]. This algorithm represents a class of coverage algorithms (including [114, 116, 118, 119, 163]) which in one way or another consider a time-varying density function, but are not sensitive to the energy limitations of the robots.

For each of the above-mentioned studies, the movement of the robots to the target locations $C_{V_i^e}$, as well as the energy dissipation of the robots are computed by solving the differential equations in Equations (5.5) and (5.6) using an ODE solver. This robot movement simulation is not required in a practical implementation of the coverage control scheme on real robots. In such a case, a navigation algorithm would be adopted to drive the robots to their target destinations. Furthermore, if after each time step, the robots still have enough energy above the minimum threshold, they will form part of the team in the next time step. Otherwise, they would no longer be considered part of the team, and are left out of the coverage problem from that point in time onwards. This algorithm is repeated for a preset amount of time.

5.3.1 Single trial experiment results

The first part of this results section considers a typical single trial. The main scope of this simulation is to compare coverage control schemes that are not-energy-aware (such as Lloyd's algorithm and Cortés's algorithm) to the proposed scheme that is energy-aware on two fronts, i.e. in the environment segmentation process as well as in the controller used. In this study five robots were used to cover and track the density function in (5.7) with $\tau = 20\text{s}$. The environment to be covered is a rectangular environment bounded by the vertices (-3, -3), (-3, 5), (5, 5) and (5, -3). The initial conditions used in this simulation are tabulated in Table 5.1. Additionally, the maximum energy of each robot $E_{i(max)} = 10$, while the minimum energy of each robot $E_{i(min)} = 0.5$.

Figure 5.2 shows the energy-aware instantaneous cost in (5.1) against time during this experiment, for the three controllers under test. It is expected that the cost $H_{e,t}$, is higher for proportional controller and for the non-energy-aware tracker proposed by Cortés et al. [115], than that of the proposed controller that considers the robots' energies as well. This is because in the cost function of Equation (5.1), the aim is to minimize the

Table 5.1: Initial conditions for time-varying single trial experiment

	Initial position $p_i(0)$	**Initial energy level $E_i(0)$**
$\boldsymbol{R_1}$	(-2.6, 2)	10
$\boldsymbol{R_2}$	(2.5, 2.7)	10
$\boldsymbol{R_3}$	(1.2, 0)	10
$\boldsymbol{R_4}$	(0, -1)	10
$\boldsymbol{R_5}$	(-2, -2.5)	10

energy cost *as well as* the coverage cost. Since it is expected that both the non-energy-aware controllers use up more energy, then the energy cost across all the team is expected to be higher. In fact, Figure 5.2 shows that the energy-aware instantaneous cost over time $H_e(t)$, is consistently higher for the non-energy-aware algorithms than for the controller proposed. The RMS value of the energy-aware cost for the proportional controller is in fact 4.32, that of the non-energy-aware tracker is 3.98, and that of the proposed energy-aware tracking controller is 2.47. As expected, this further shows that an energy-aware algorithm performs better than the non-energy-aware algorithms, in a realistic scenario where energy conservation is always important.

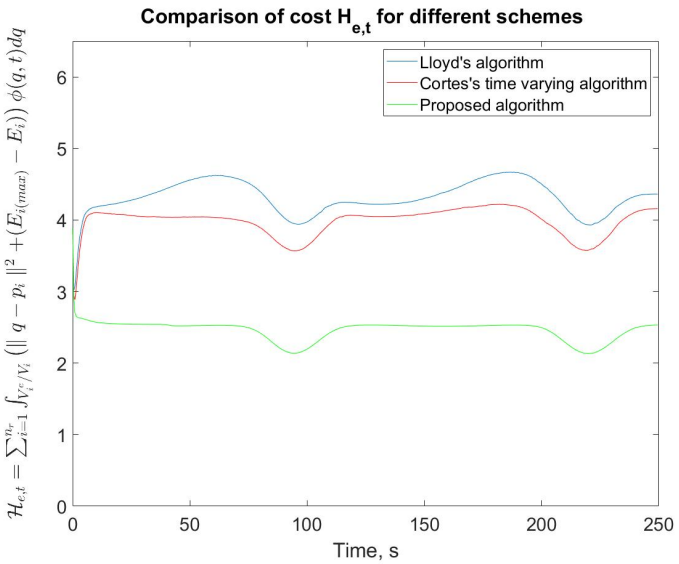


Figure 5.2: Plot of the instantaneous cost $H_{e,t}$.

Additionally, one would expect that since the proposed energy-aware scheme is trying to conserve energy, then the coverage of the environment by that scheme shall somewhat suffer when compared to those algorithms that are not energy-aware. This is because an energy-aware algorithm would be more reluctant to allow robots to spend their energies

freely. Rather, it would attempt to strike a balance between energy expenditure and maximising coverage of the environment. To show this, the instantaneous cost $H_c(t)$ in (5.8) is plotted for all three algorithms for the same simulation trial.

Moreover, the algorithms using the proportional controller and the controller proposed by Cortés et al. [115], are segmenting the environment using the standard Voronoi diagram, and therefore, these two algorithms are expected to have a lower coverage cost than that of the proposed algorithm since their sole aim is to minimize the cost in (5.8). On the other hand, the proposed algorithm is segmenting the environment using a power diagram that segments the environment with the aim of maximizing coverage *as well as* conserving the energy through the size of the generated Voronoi areas, as argued in Chapter 4. Figure 5.3 illustrates that, as expected, the proposed energy-aware algorithm has the largest overall coverage cost, while the non-energy-aware algorithms have comparable coverage costs since they are using an environment segmentation process that does not account for low energy values.

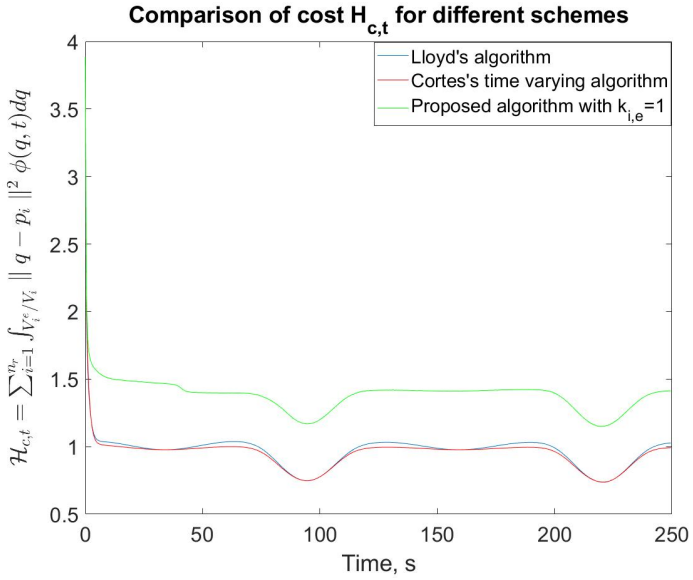


Figure 5.3: Plot of the instantaneous coverage cost $H_{c,t}$.

However, it should also be emphasized that in a realistic scenario, where the robots' energies are limited, the proposed scheme might even lead to lower coverage cost $H_c(t)$ than its competition in a longer trial period. This is because its energy conserving nature allows the robots to operate for a longer period of time and hence, continue to contribute to the team to lower the coverage cost. In contrast, with the non-energy-aware algorithm,

the robots take less time to be depleted of their energy and hence, might stop earlier on during the operation. This sacrifices coverage from that point onwards since fewer robots are now covering the environment. This argument stems from the fact that perfect coverage (zero cost) may be obtained with an infinite number of robots. Hence, decreasing the size of the team (due to robots stopping without energy), increases the overall cost. Moreover, there is a higher likelihood that the size of the team diminishes quickly with non-energy-aware algorithms than with the proposed energy-aware algorithm. This element of endurance when using the proposed energy-aware controller is investigated in Section 5.3.2.

Using environment segmentation and control algorithm that is not energy-aware, one expects the energy consumption to be greater than that of an algorithm which is energy-aware. This is because the non-energy-aware algorithm is not concerned with how the energy of the team is spent, but rather, the aim is focused on dispersing the robots as much as possible over the environment. The total energy consumed for this trial simulation, for every time instance, is calculated as follows,

$$E_{consumed}(t) = \sum_{i=1}^{n_r} E_{i(max)} - E_i(t) \tag{5.9}$$

Although it might be obvious that an energy-aware algorithm will always outperform a non-energy-aware one, this can actually depend on the chosen scenario. If the maximum energy of each robot $E_{i(max)}$, is very high for the given mission, the non-energy-aware algorithm can have an advantage since it can be more greedy in its energy consumption, achieve better coverage than the energy-aware algorithm and still leave the robots with ample energy. For this reason, in these simulations $E_{i(max)}$ is set to a fair and realistic value. Figure 5.4 shows that as expected the energy-aware algorithm using the proposed controller and the proposed energy-aware power diagram, consumes the least energy when compared with the other non-energy-aware algorithms, for which both the controllers and the environment segmentation are non-energy-aware. The energy consumption seems to reach a plateau not because the robots stop moving, but because the density function they follow is slow and hence the rate at which energy is being consumed is also low.

As expected from a scheme that is trying to preserve energy, the energy consumed by the scheme proposed in this thesis is less than that of the two other non-energy-aware schemes. However, a question may also arise whether such a preservation of energy

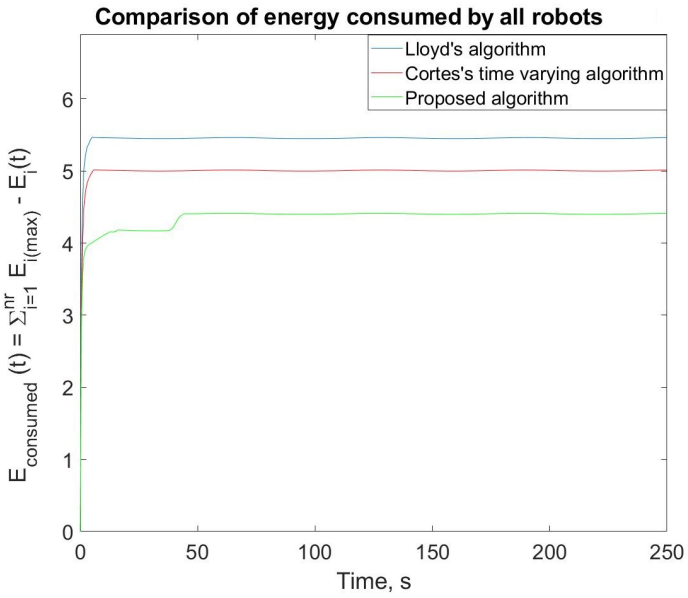


Figure 5.4: Plot of the instantaneous energy consumed by the team of robots.

would affect the overall centroid tracking performance. For this purpose, the accumulated tracking error according to (5.10) is also compared across the three schemes. The aim of this experiment is to understand better the effect of the energy-dependent gain on the tracking ability of the system. The same single trial experiment described above was run for 800 seconds. The resulting accumulated errors are illustrated in Figure 5.5.

$$e_{total} = \int_0^t \sum_{i=1}^{n_r} \| \, p_i - \dot{C}_{V_i^e} \, \|^2 \tag{5.10}$$

In Figure 5.5 one may observe that the scheme proposed by Cortés et al. [115] provides the best tracking. The very small increase in the error by time, is due to the spatial-quantization of the environment. More noticeable is the increase in the error of the Lloyd's algorithm and the proposed scheme. Firstly, Lloyd's algorithm does not use a tracking controller and this contributes to the ever-increasing tracking error due to the time-varying density function. However, one must note that in this case, increasing the gain k_{prop} would lead to a reduction in the tracking error. Nevertheless, this would be detrimental on the energy consumption. In contrast, the energy-aware tracking controller proposed in this thesis seeks to strike a trade off between tracking performance and energy preservation. Although the tracking of the proposed scheme is not as good as it is in [115], the rate of error increase is not as high as that of the Lloyd's algorithm. This is because

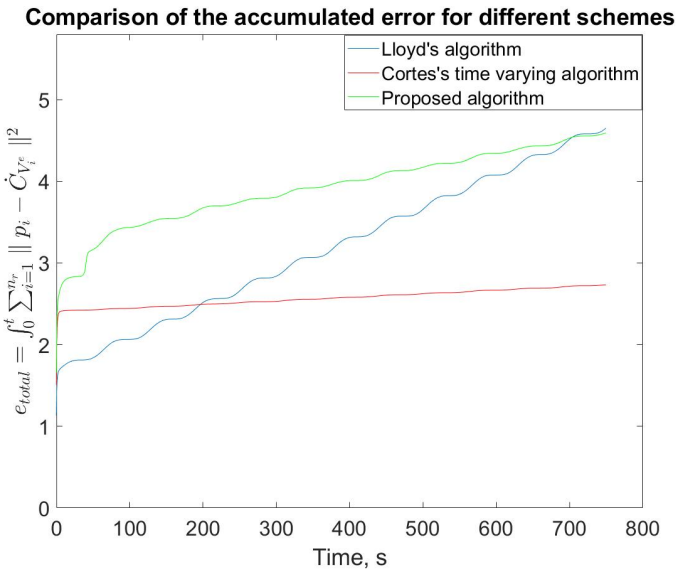


Figure 5.5: Comparison of the accumulated centroid-tracking error among the three schemes

unlike the proportional controller, the proposed tracker is not just reactive to the present error, but it also foresees the rate of change of the target centroid. At the same time, the proposed controller is also working towards preserving the energy of the robots, and hence, its gain $k_{i,e}\frac{E_i}{E_{i(max)}}$ is mindful of the rate at which energy is being depleted. Hence, the higher the energy consumption, the slower the robots are driven. To summarize, the priority of the proposed controller is not to provide the least tracking error, but rather to strike a user-tunable balance between perfect centroid-tracking and energy preservation.

5.3.2 Multi-Robot System endurance test

As has been discussed in the previous sections, one important aim of having an energy-aware controller is for the team to be able to operate for a longer time during a coverage operation. In this section, a simulation is presented to assess the endurance of the proposed energy-aware controller when compared to the non-energy-aware controllers. For this purpose, the initial conditions in Table 5.2 were used in this test. In this experiment, $E_{i(max)} = 4$ and $E_{i(min)} = 1$.

For the purpose of this test, the energy dynamics were modified slightly such that the robots were made to spend energy at a faster rate than in the previous tests. Particularly, the energy dynamics used in this experiment are as follows, where m_i in (5.6) is now set

Table 5.2: Initial conditions for a single trial endurance test

	Initial position $p_i(0)$	**Initial energy level $E_i(0)$**
$\boldsymbol{R_1}$	(-2.6, 2)	4
$\boldsymbol{R_2}$	(2.5, 2.7)	3
$\boldsymbol{R_3}$	(1.2, 0)	3.5
$\boldsymbol{R_4}$	(0, -1)	4
$\boldsymbol{R_5}$	(-2, -2.5)	3

to 2.

$$\dot{E}_i = \begin{cases} -2u_i^2, & \text{if } E_i \geq 0, \\ 0, & \text{if } E_i = 0 \end{cases} \tag{5.11}$$

Additionally, the same time-varying density function used in the previous experiments is employed in this test as well. For this endurance test, the Lloyd's algorithm using a proportional controller is compared to the algorithm proposed by Cortés et al. [115] (that uses a tracking controller), and to the algorithm proposed in this thesis that uses an energy-aware tracker. As has been explained in Section 5.3, for the non-energy-aware algorithms, the standard Voronoi is used as the segmentation module, while for our proposed algorithm, the power diagram proposed in [1] is used.

For each of the three algorithms, the robots were completely depleted of energy at the times indicated in Table 5.3. From Table 5.3 one observes that robot R_4 was the only robot that did not stop during this trial.

Table 5.3: The times at which the robots were depleted of energy across the different algorithms

Algorithm	$\boldsymbol{R_1}$	$\boldsymbol{R_2}$	$\boldsymbol{R_3}$	$\boldsymbol{R_4}$	$\boldsymbol{R_5}$
Lloyd's Algorithm	2.5s	1.5s	193.5s	–	3s
Cortés's tracker	6.5s	2s	194.5s	–	12s
Energy-aware proposed algorithm	142s	53s	198s	–	96s

From the result in Table 5.3, it is immediately evident that since the robots are using more energy when using the proportional controller in Lloyd's algorithm, they tend to stop much earlier than in the other approaches. In fact, after just 3s, the team is already operating with two robots only. Compartively, using the tracking controller proposed by Cortés et al. [115], the team was reduced to just two robots after 12s. However, both of these algorithms are non-energy-aware. When the multi-robot system (MRS) was running the proposed energy-aware controller, one can observe that the first robot was depleted

of energy after 53s. Therefore, the team had been operating with the maximum number of agents for a longer time than either of the two other algorithms. In fact, it was only after 142s that the multi-robot time running our proposed algorithm ended up with two robots. This clearly shows that the proposed energy-aware algorithm allowed the robots to continue operating for a longer time than in the non-energy-aware algorithms. Therefore, the endurance of the multi-robot team running our proposed algorithm is much higher than that of the non-energy-aware algorithms.

In addition, one may also note that R_3 remains active for a longer period of time in all three cases, despite the fact that it started with relatively a low energy level of 3.5 units. The reason for this is that its initial position is relatively close to the density function being covered, and therefore, it does not need a lot of energy to get to its CVT position.

Moreover, in a coverage control application we require the robots to last longer (in terms of energy) because effective and continuous coverage of an environment is better achieved with more robots. If most of the robots have their energies depleted early on in the operation, the coverage of the environment will not be as good as it had been with a full team. This is because the smaller the number of robots, the more spread out they shall be over the environment. To analyse this effect, the coverage cost in (5.8) is plotted for the duration of this endurance test for all three algorithms. This is illustrated in Figure 5.6.

Since many of the robots running the non-energy-aware algorithms stopped early on, while those running the energy-aware algorithm did not, the coverage cost of the proposed energy-aware algorithm is now lower than that of the non-energy-aware algorithms. This is in contrast to what has been seen in Figure 5.3, where the proposed energy-aware algorithm obtained a worse coverage cost than the non-energy algorithms. In Figure 5.3, however, none of the robots had been depleted of their energies and therefore, the team was operating with the maximum number of robots across the three algorithms. In this endurance test, Figure 5.6 shows that by allowing the robots to spend less energy during the operation, they are able to run for longer and therefore, they are more able to cover the environment effectively when compared to non-energy-aware algorithms. Additionally, at around 142s the robot team in all of the three algorithms now consisted of just two robots. Therefore one can observe that the coverage cost of the energy-aware algorithm is now higher than that of the non-energy-aware algorithms. This part of the experiment

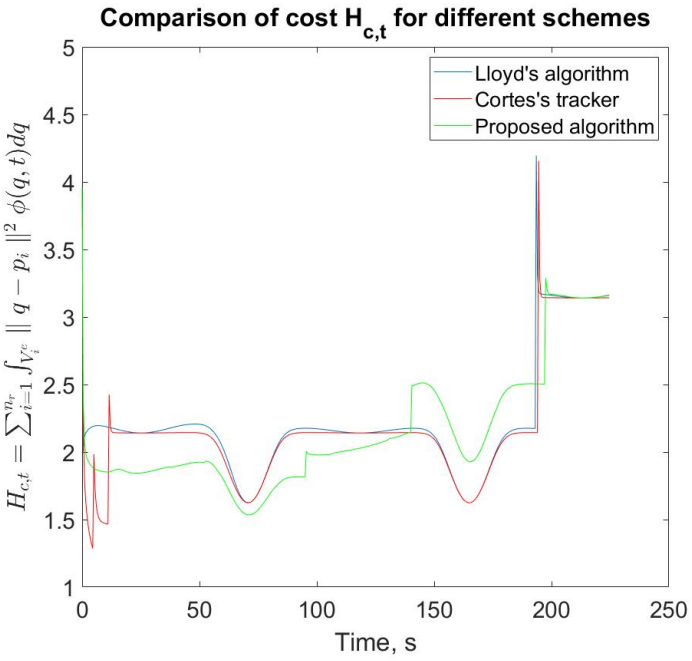


Figure 5.6: Coverage cost for the endurance test for the three algorithms

conforms with the result in Figure 5.3.

5.3.3 Monte Carlo simulations

To further show that the hypothesis that an energy-aware algorithm performs better than the non-energy-aware algorithm, proposed by Cortés et al. [115], in a typical coverage control scenario where the robots' energies are limited, 100 Monte-Carlo simulations were conducted. For each trial the RMS value of the cost in Equation 5.1, $H_{e,t}$, is recorded for both algorithms. For this purpose, the initial energy level as well as the mass of each robot, were randomly generated for each trial. The mass of each robot affects the rate of energy consumption, so this also makes our simulation more realistic. For each trial, the initial energy levels and the robot masses were the same for both algorithms. All other parameters were left constant across the different trials and across the two algorithms. Furthermore, the density function $\phi(q,t)$ could have an affect on the rate of energy consumption depending on the value of τ. This is because the smaller the value of τ, the faster the movement of the environment and hence the higher the reference speeds of the robots. Since speed directly effects the rate of energy consumption, then one would expect this rate to be higher for smaller values of τ. In the current study, τ

was also kept constant.

The average RMS value of the cost $H_{e,t}$ (across all trials) for the energy-aware algorithm is 5.03, while that of the non-energy-aware algorithm is 5.46. The mean and variance of the two data sets are tabulated in Table 5.4.

Table 5.4: Table of mean and variance of the two cost data sets

Algorithm	Mean Cost	Variance of Cost
Not Energy-aware	5.46	0.65
Energy-aware	5.03	0.72

One can already see that there is a considerable difference among the mean cost of these two algorithms, however, a statistical test can show whether this difference is statistically significant or not. The RMS values data set of the two algorithms, were first tested for normality by analysing the histograms of the two data sets and ensuring that the data is normally distributed. This revealed that the collected data in both data sets is normally distributed. This is further indicated in the boxplot shown in Figure 5.7. To confirm that there is a statistical difference between the two sample means, a two-sample t-test was used to test the *null hypothesis* that the difference between the mean cost of the two algorithms is due to chance and not due to the intrinsic differences between the algorithms themselves. The *alternative hypothesis* is that the difference between the means of the two algorithms is statistically significant and not due to chance. The result of the statistical t-test strongly rejected the null hypothesis, with a level of significance of 0.05. The two-tailed p-value recorded was 0.000335. Hence, this shows that the difference between the performance of the two algorithms is statistically significant, meaning that the proposed scheme truly leads to lower overall costs.

5.4 Analysis of the effect of the gain $k_{i,e}$

5.4.1 Varying $k_{i,e}$ of one robot

In the following experiment, robot R_1 is assumed to be a robot that is able to spend more energy and hence, its constant $k_{i,e} = 5$, while for all the other robots $k_{i,e} = 1$. Since R_1 now has a larger proportionality constant when compared to the other robots, it is expected to move faster towards its goal, at the detriment of consuming energy quicker.

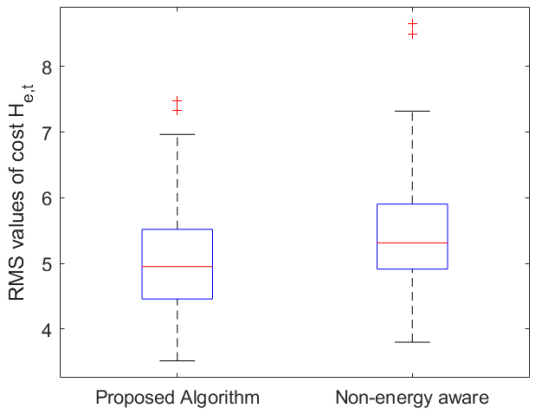


Figure 5.7: Boxplot of the datasets of our proposed algorithm and the non-energy-aware algorithm

In turn, the overall cost $H_{e,t}$ is also expected to be higher than the cost when $k_{i,e} = 1$ for all the robots. To show this, during this single trial experiment six robots are considered and all of them start from a maximum energy of 10 units. Furthermore, the duration of the trial was set to 120 seconds.

To be able to analyse these results, the same conditions used in Section 5.3.1 were employed. This study consists of three trials. In the first trial, $k_{1,e}$ of robot R_1 is set to 5 while the $k_{i,e}$ of the remaining robots was set to 1. In the second trial, the gain $k_{i,e}$ was set to 1 for all the robots in the team. For the first and second trial experiments, the power diagram segmentation algorithm used in Section 5.3.1 is employed and the proposed energy-aware controller is used. In the third trial, the tracker proposed by Cortés et al. [115] is used together with the standard Voronoi environment segmentation algorithm. For this third trial, the tracker's proportionality constant k in (2.29) was set to 1. Table 5.5 tabulates all the results together with the initial positions of each robot, which were chosen arbitrarily. The results include the total energy consumed by each robot, for each individual trial. The total energy consumed is calculated as $E_{i(max)} - E_{i(final)}$.

As expected, R_1 consumed the most energy during Trial 1, because it was tuned to use more energy via a higher $k_{i,e}$. This shows that with the proposed controller an individual robot may be allowed to use more energy, for example to achieve higher speeds. When comparing the energy consumed for Trials 2 and 3, one can note that almost all robots perform better (in terms of energy consumption) when the energy-aware controller is used, as opposed to that proposed by Cortés et al. [115]. This, of course, confirms the

Table 5.5: Results of the single trial experiment to analyse the effect of $k_{i,e}$

Robot	Initial position	Trial 1: Total energy consumed when $k_{1,e} = 5$ and $k_{i,e} = 1$ $\forall i = \{2, ..., 6\}$	Trial 2: Total energy consumed when $k_{i,e} = 1$ $\forall i = \{1, ...6\}$	Trial 3: Total energy consumed when using Equation 2.27
R_1	(-2.5, -2.8)	4.898	1.362	1.502
R_2	(-1, 0.5)	0.203	0.202	0.202
R_3	(1, -0.8)	0.223	0.223	0.233
R_4	(2.3, 2.7)	1.732	1.733	1.863
R_5	(1.3, 3.7)	1.723	1.722	1.843
R_6	(1.2, 4)	2.053	2.053	2.314

results previously presented in the Monte Carlo simulations in Section 5.3.3.

5.4.2 Cost analysis when varying $k_{i,e}$

For a more in-depth analysis of the proposed energy-aware algorithm, the overall cost is analysed for various $k_{i,e}$ settings $\forall i = \{1, \dots n_r\}$. Five robots are considered to form the team in this study, with the initial conditions as tabulated in Table 5.6.

Table 5.6: Initial conditions for time-varying single trial experiment for different $k_{i,e}$

	Initial position $p_i(0)$	**Initial energy level $E_i(0)$**
$\boldsymbol{R_1}$	(-2.6, 2)	10
$\boldsymbol{R_2}$	(2.5, 2.7)	10
$\boldsymbol{R_3}$	(1.2, 0)	10
$\boldsymbol{R_4}$	(0, -1)	10
$\boldsymbol{R_5}$	(-2, -2.5)	10

The energy-aware cost in (5.1), the energy consumption in (5.9) and the accumulated tracking error in (5.10) are plotted in Figures 5.8, 5.9 and 5.10, respectively.

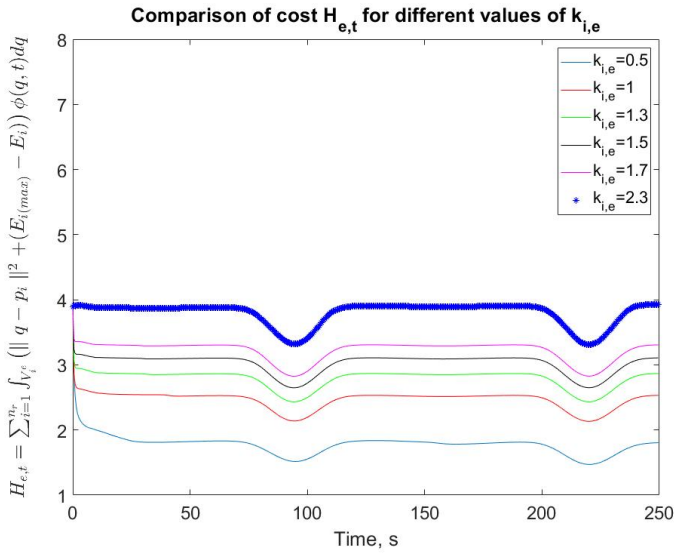


Figure 5.8: Comparison of the energy-aware cost $H_{e,t}$ for different $k_{i,e}$

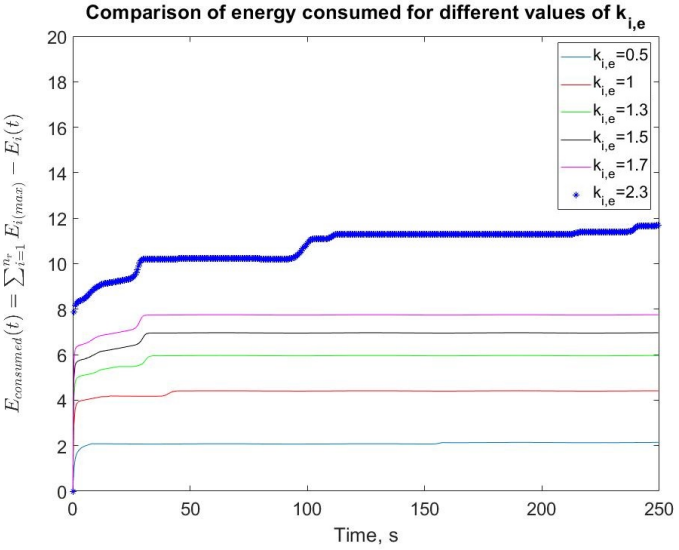


Figure 5.9: Comparison of the energy consumed for different $k_{i,e}$

As can be seen in Figure 5.8, when $k_{i,e}$ is increased $\forall i = \{1, \dots n_r\}$ the energy-aware cost $H_{e,t}$ increases as well. This is because the $(E_{i(max)} - E_i)$ component increases since more energy is being consumed, as illustrated in Figure 5.9. The increased energy consumption directly contributes to the cost $H_{e,t}$. Additionally, as the energy consumption increases, the environment segmentation generates cells V_i^e that increases the coverage cost aspect of the energy-aware cost, given by $\| q - p_i \|^2$. This is because as more energy is consumed, the lower the robots' energies levels E_i are, and hence, the higher the term

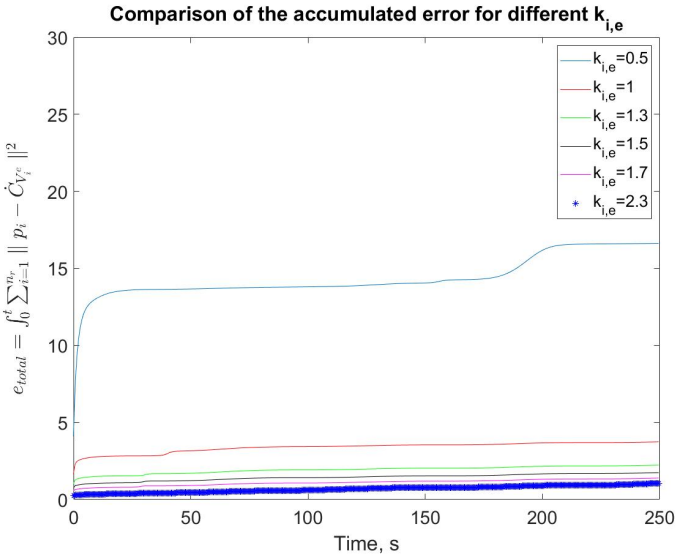


Figure 5.10: Comparison of the accumulated tracking error for different $k_{i,e}$

$(E_{i(max)} - E_i)$ is. These two factors contribute to the increase in $H_{e,t}$. Furthermore, as $k_{i,e}$ increases the accumulated error in Figure 5.10 is seen to decrease. This is because the robots are being allowed to have a larger control effort u, and therefore, the tracking error decreases as expected.

5.5 Comparison of the proposed energy-aware scheme to the main energy-aware schemes recently proposed in literature

Santos et al. [155] also present an energy-aware algorithm that allows the robots in the team to track time-varying density functions. The work presented in [155] was simulated and compared to the energy-aware scheme proposed in this chapter. The main differences between the proposed scheme and that in [155] can be summarized as follows:

- The scheme proposed by Santos et al. segments the environment using the Delaunay triangulation called the Delaunay graph. The Delaunay graph of a discrete set of points corresponds to the bounded Voronoi diagram of that discrete set of points. The circumcenters of the Delaunay triangles in the graph correspond to the vertices of the Voronoi diagram. This means that the environment segmentation is not

energy-aware like the proposed power diagram.

- The controller proposed by Santos et al. is an optimization-based controller that aims to minimize the control input in an effort to make the scheme energy-aware. However, this leads to significantly higher execution times than those of the proposed energy-aware controller in (5.3). This is because the algorithm in [155] solves a constrained-optimization problem during every iteration.

- The proposed controller in (5.3) has the tuning parameter k_{ie}. The controller as presented in [155] does not have a similar tuning parameter that adjusts the level of energy awareness as described above.

In this experiment five robots formed part of the team that first ran the proposed energy-aware algorithm in a number of trials and in one of the trials the team employed the algorithm proposed in [155]. The initial conditions of this experiment are tabulated in Table 5.6. The time-varying density function used in this experiment is given by (5.12). For a more in-depth analysis, the tuning parameter of the proposed scheme $k_{i,e}$ was set to 0.3, 0.5 and 1 in three separate trials. The response of the proposed scheme is then compared to the scheme simulating the design in [155].

$$\phi(q,t) = e^{-\left(q_x - 2\sin\frac{t}{\tau}\right)^2 - q_y^2} \tag{5.12}$$

The resulting energy-aware cost $H_{e,t}$ is shown in Figure 5.11. From this plot, one can observe that as the tuning parameter $k_{i,e}$ is decreased, the cost of the proposed scheme becomes lower than that produced by the scheme proposed by Santos et al. [155]. This is also reflected in the plot of the energy consumption (Figure 5.12) where for low values of $k_{i,e}$, the total energy consumed by the robot team is less than that consumed by the team running the scheme in [155].

When $k_{i,e} = 1$ the cost $H_{e,t}$ of the proposed scheme is comparable to that by Santos et al. In fact, the RMS value of cost of the proposed scheme when $k_{i,e} = 1$ is equal to 2.47 while the RMS value of the cost of the scheme proposed in [155] is 2.38. For such a comparable cost, the energy consumed by the scheme proposed in this chapter is larger than that consumed by the scheme in [155] as seen in Figure 5.12.

However, Figure 5.13 shows that the accumulated error for our scheme is less than that achieved by the Santos et al. scheme for this setting of $k_{i,e}$. This means that for

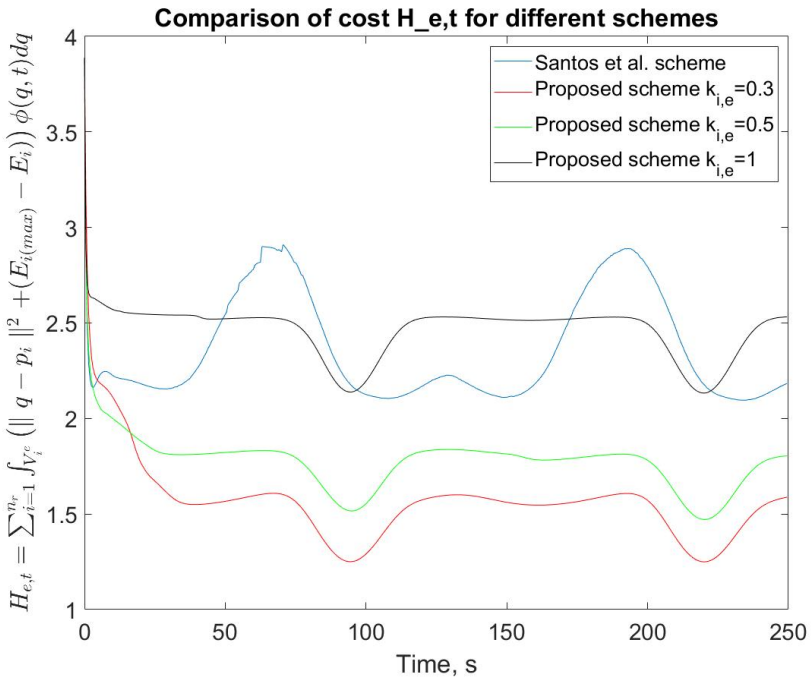


Figure 5.11: Comparison of the energy-aware cost $H_{e,t}$ for different schemes

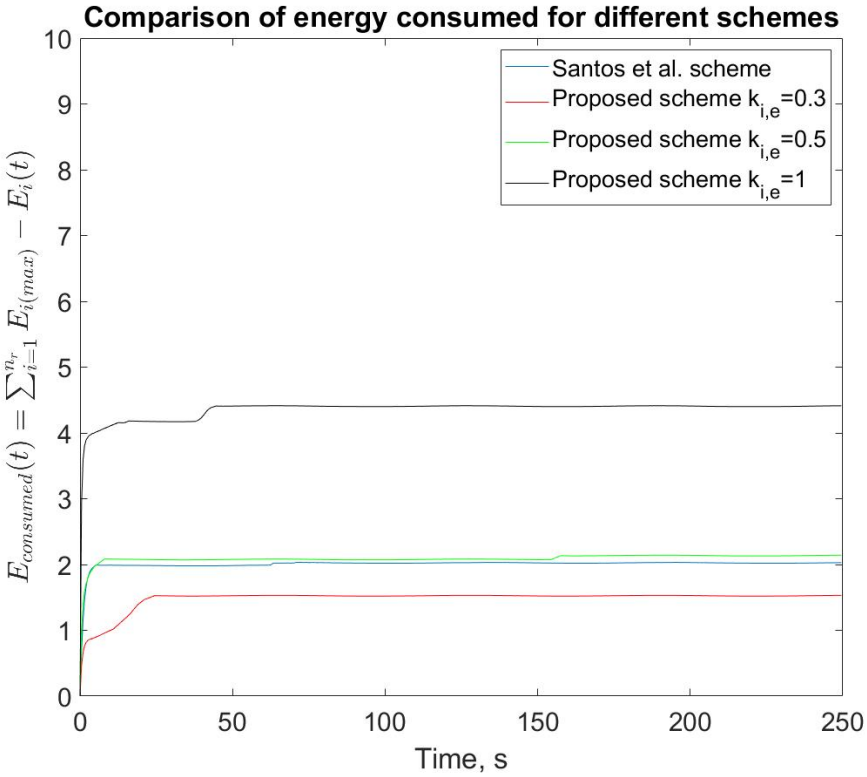


Figure 5.12: Comparison of the energy consumed for different schemes

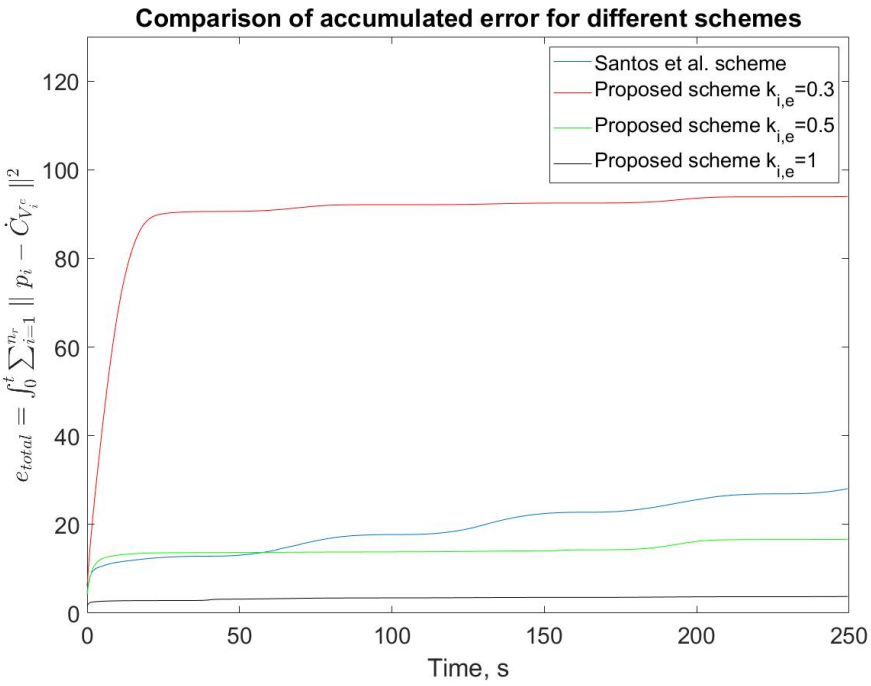


Figure 5.13: Comparison of the accumulated error for different schemes

a comparable overall energy-aware cost, our scheme is less energy efficient than that by Santos et al., however, it offers better tracking. When $k_{i,e} = 0.5$ one can observe that the two approaches fare similarly in terms of the amount of energy consumed and tracking error. However, the tracking error by Santos et al. has a steeper gradient than that of our proposed scheme. In this scenario, the energy-aware cost $H_{e,t}$ is lower for our proposed scheme than it is for the scheme by Santos et al.

In conclusion, the main benefit of the proposed scheme over that proposed in [155] is that the proposed scheme has the tuning parameter that allows the user to trade off energy efficiency for better tracking error. With specific settings, a similar performance (in terms of the energy consumed and the tracking error) may be obtained, with a lower value of the cost $H_{e,t}$. By tuning the parameter $k_{i,e}$ the end-user is able to tailor to the needs of the application. This feature is not present in the state-of-the-art scheme presented in [155]. In addition, for the same number of time steps, the simulation of the scheme by Santos et al. took 591 seconds, while the total execution time of the proposed approach is 300 seconds. This shows that the proposed scheme is much less computationally heavy than that proposed in [155]. This significant discrepancy between the execution times (almost twice as fast) may be attributed to the fact that the scheme

by Santos et al. executes an optimization algorithm in each iteration to compute the control efforts of the robots. This can be another determining factor in fast real-time implementations.

5.6 Conclusions

This chapter introduced the concept of energy awareness that is ingrained in the proposed design. In particular, the energy awareness of the framework is twofold: it is achieved by including the energy component in the power diagram segmentation and in the proposed energy-aware centroid-tracking controller. In this chapter, the proposed energy-aware scheme is compared to non-energy-aware schemes. From this study, the benefits of having an energy-aware scheme become very evident since the energy-aware scheme consumes less energy than the non-energy-aware schemes and still retains good coverage performance.

This is especially seen when the robot team was subject to an endurance test to see how much each robot is able to preserve its energy and therefore, remain active for a longer period of time when running the energy-aware algorithm. The designed energy-aware tracking controller allows the robots in the team to consume less energy while they are tracking the time-varying centroids, when compared to when they are running the non-energy-aware algorithms. In the endurance study presented in this chapter, it is shown that when robots are using the energy-aware algorithm they are able to remain active for a longer time than when running the non-energy-aware algorithm and this directly contributes to the efficiency of the coverage problem. In fact, this chapter shows that when most of the robots in the team stop without energy during the operation, the multi-robot system (MRS) obtains a higher non-energy-aware coverage cost than when the robots remain active for longer. In conclusion this means that when the energy-aware controller is used, the robots are able to cover the environment more effectively by remaining active for a longer time. This is a fundamental benefit that the energy-aware controller has over the non-energy-aware controllers.

Furthermore, the proposed energy-aware scheme performs better than the non-energy-aware scheme in terms of the energy-aware cost $H_{e,t}$. To further show that this is not the case for some singular particular conditions, 100 Monte Carlo simulations are performed,

each time-varying the initial energy levels of the robots and the masses of the robots themselves. Across these 100 trials, it is shown that there is a statistical difference between the mean RMS values of the costs of the two algorithms. This shows that that the energy-aware algorithm generally performs better than that which is not energy-aware, when the environment has a time-varying density function and the robots have limited energies, as is the case in typical practical coverage control applications.

Additionally, the effects of the tuning parameter of the proposed energy-aware controller $k_{i,e}$ was also investigated, and a set of simulations were presented. This study shows that if an application can afford to be less energy efficient, by setting $k_{i,e}$ to a larger value, the tracking error would also decrease. This shall, however, increase the energy consumption. Hence, the tuning parameter $k_{i,e}$ allows the user to trade-off energy consumption for better tracking performance. This was further investigated when the proposed scheme was simulated and compared to the recent work proposed by Santos et al. [155]. The proposed scheme may obtain a better performance and the robots may consume less energy by setting the value of $k_{i,e}$ appropriately for each robot. With a similar energy consumption and similar tracking error, the proposed scheme is seen to perform better than the scheme proposed by Santos et al. in terms of the energy-aware cost $H_{e,t}$ that is being minimized.

In conclusion, the proposed energy-aware controller and the inclusion of the energy consumption in the power diagram contribute to the energy efficiency of the framework. This energy efficiency may be adjusted accordingly through the tuning parameter $k_{i,e}$.

Chapter 6

Combined coverage control framework

6.1 Overview

As discussed in the previous chapters, a practical implementation of a coverage control multi-robot system (MRS) is one that addresses a number of practical factors such as the sensory capabilities of the robot team, the energy levels of the robots, as well as any features and changes in the environment. To account for all these factors, the novel proposed framework was designed around three main modules, namely: an optimal multi-robot multi-task allocation module, an environment segmentation module that is dependent on the robots' health and capabilities, and an energy-aware tracking controller that allows the team to accurately track time-varying features of the environment. The previous chapters of this thesis have presented each of these modules individually, by first detailing the design of each module, and then by providing various tests to validate and study each module independently.

This chapter investigates how these modules fit together to form a modular framework that can be used in a real coverage control application with a heterogeneous team of robots, in the presence of multiple, time-varying regions of interest. The rest of the chapter presents the design of the framework together with a stability analysis of the coverage control algorithm. Additionally, a set of realistic simulation results to validate this novel framework are presented and discussed.

6.2 Coverage control framework design

The proposed framework is modular since each individual component can be directly replaced with a similar algorithm, as long as the correct inputs, outputs, and function of each module are adhered to. This is in contrast to the framework recently proposed by Notomista et al. [159] which embeds the control input generation and the task-allocation in one optimization problem. The main advantage of a modular framework is that each module can be easily replaced or modified to suit specific requirements of the particular coverage application. For example in the proposed framework, one can seamlessly switch the MRTA algorithm between Algorithm 1.0 and 2.0 to meet different objectives as explained in Chapter 3. Moreover, the interface between the modules must be clearly designed and the stability of the multi-robot system (MRS) should be analysed.

The framework is designed to operate in an environment with multiple, time-varying regions of interest. As argued previously, the mathematical models of the regions of interest are probability density functions assumed known by the system designer. To achieve its goal, the framework first obtains a list of all the robot-to-density allocations as generated by the MRTA module. The segmentation of the environment is not directly dependent on the density function associated with each robot, since as detailed in Section 2.2.3 the Voronoi cells are generated according to:

$$\mathcal{V}(p_i) = \{q \in \mathbb{R} | \parallel q - p_i \parallel^2 - w_i \leq \parallel q - p_j \parallel^2 - w_j, \forall j \neq i\} \tag{6.1}$$

where

$$w_i = \alpha_1 \left(\frac{E_i - E_{i(max)}}{E_{i(max)} - E_{i(min)}} \right) + \alpha_2 \left(\frac{v_i - \max(\boldsymbol{v})}{\max(\boldsymbol{v}) - \min(\boldsymbol{v})} \right) - \frac{\alpha_3}{n_s} \sum_{k=1}^{n_s} s_k \tag{6.2}$$

However, the calculation of the centroid of each cell is directly dependent on the specific density function that is associated with a particular robot. This means that the robot is driven towards the centroid of its current polygon Voronoi cell depending on the intersection of that polygon with the respective density function. In this light, the calculation of the Voronoi cell mass M_{V_i} and the centroid C_{V_i} are calculated as follows, as detailed in Section 2.3:

$$\begin{aligned} M_{V_i} &= \int_{V_i} \phi_i(q,t)\,\mathrm{d}q \\ C_{V_i} &= \frac{1}{M_{V_i}} \int_{V_i} q\phi_i(q,t)\,\mathrm{d}q \end{aligned} \tag{6.3}$$

where $\phi_i(q,t)$ is the time-varying density function associated with robot i. This method aims to achieve the same objectives as that in [120], i.e. drive the robots towards the density functions that they have been associated with. However, Abbasi et al. [120] achieve this through the complex and computationally-heavy controller shown in (2.35) and (2.36). Abbasi et al. argue that their controller drives the robots towards their centroids while the γ_i in their controller factors in the effects of the neighbouring regions on the robot's movements, i.e. by addressing the difference in the neighbouring density functions at the boundary between two neighbouring robots. Additionally, Abbasi et al. also comment on the significantly large amount of data that has to be shared over the communication link such that the robots are able to individually calculate their own control effort u_i. Moreover, the controller proposed by Abbasi et al. does not address time-varying density functions, and hence tracking accuracy is not ensured. To this end, our approach is to utilise the tracking controller proposed in Chapter 5 and drive the robots towards the centroids of their Voronoi cells that heavily depend on the density function that they are associated with. Section 6.4.1 shows that using this approach, the robots are able to cluster around their respective density functions, without requiring the controller proposed in [120]. Before addressing the framework as a whole, the inputs and outputs of each module are analysed. Table 6.1 lists the inputs and outputs required by each module accordingly. Additionally, Figure 6.1 illustrates the interaction among the main modules in the framework.

Table 6.1: List of inputs/outputs for each module

Module	**Inputs**	**Outputs**
Multi-robot task allocation	▪ Robot capabilities (physical type and sensor capabilities) ▪ Density requirements ▪ User-forced robot-to-density pairings ▪ Values for β_1 to β_4 ▪ Normalized values of γ_d and γ_t for each candidate robot-to-density pairing ▪ Normalized values of γ_e ▪ The sensors' performance ratings for each robot	▪ Robot-to-density allocation list
Power diagram-based environment segmentation	▪ Environment boundaries ▪ Current robot position, p_i ▪ Robot weight calculated according to (4.1)	▪ List of vertices of all the polygon Voronoi cells ▪ List of grouped vertices for each polygon Voronoi cell
Centroid-tracking robot controller	▪ Current robot position, p_i ▪ Current robot energy level, E_i ▪ Centroid C_{V_i} for the robot's generated Voronoi cell ▪ Pre-calculated values of $\dot{C}_{V_i}$ and $\dot{M}_{V_i}$	▪ Robot position after movement ▪ Robot energy level after movement

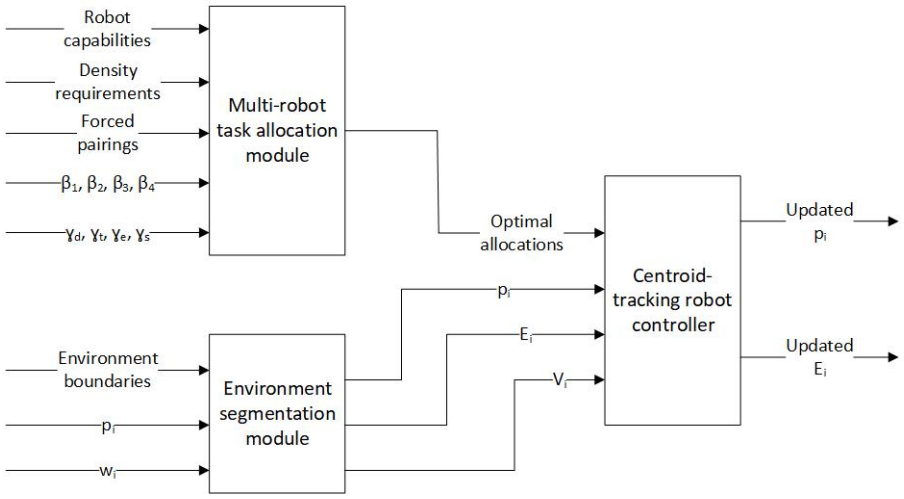


Figure 6.1: Block diagram depicting the proposed framework modules.

Before starting its execution, the framework pre-computes the normalized values of the Euclidean distance γ_d, the shortest travel time γ_t and the current energy level indicator γ_e for all the potential robot-to-density candidate allocations. This facilitates the online calculation of C_{ij} when performing the candidacy test in the allocation algorithm. Furthermore, γ_s is computed during the candidacy test itself, such that only the performance ratings of the sensors matched with the density requirements are considered in the calculation. The framework is then segmented into two phases. The first phase is the process of robot-to-density allocation using the algorithms proposed in Chapter 3. During this phase, the coverage process has not yet been initiated. The list of optimal allocations generated by the optimal allocation module is then passed to the coverage process, such that the robots can start their coverage operation.

During the second phase, the framework first checks the robots' energy reserves. If it finds that one of the robot's energy has been depleted (i.e. E_i falls below a particular minimum threshold $E_{i(min)}$), the framework removes that robot from the problem and re-allocates the available robots among the given densities. Otherwise, if a robot has an energy level above $E_{i(min)}$, it is considered as part of the team and is allocated to a density function.

Re-allocation is done by running the same allocation algorithm that has been run in the first phase, and by re-computing the normalized γ_d, γ_t, γ_e and γ_s in real-time. After this part, the framework has a list of all the robots participating in the coverage operation, together with their necessary parameters (position, maximum velocity, energy levels and

sensor performance ratings), and their optimal allocations to the densities to be covered. The framework then progresses to the next step which is the environment segmentation process.

This process utilises and modifies some of the parameters computed for the allocation algorithm, namely: w_e in (4.2) and w_s in (4.4). These weights are effectively the negated values of γ_e and γ_s, such that they are changed from a cost to a rewarding factor. During this process, the design proposed in Chapter 4 uses the robots' positions, the environment boundary and the computed weights, to segment the environment using the power diagram algorithm.

Once each robot has a Voronoi cell generated for it, the framework computes the centroids of each cell as a target location. The centroid is calculated according to (6.3), i.e. according to the intersection of the Voronoi cell generated by the power diagram, with the density function assigned to a robot. This calculation enables the robots to drive towards the higher importance areas of the densities that they are allocated to. This design leads the group of robots to a final configuration where each sub-team covers the density function that they were allocated to and at the same time maximize the coverage of the overall environment through the adoption of Algorithm 2.1.

In the proposed work it is important to note that the generation of the centroids and the robot's controller, ignore the effects of the neighbouring density functions that intersect with the Voronoi cell of the robot in question. The reason for this design choice is that the design of this framework prioritizes the gathering of the robots around the densities that they have been allocated to.

This leads to the final stage of the framework, which is that of using the controller proposed in Chapter 5 to drive the robots towards their centroids. In the simulation environment, the holonomic robot first order dynamics are formulated as a linear ordinary differential equation that is solved by the `ode45` solver in MATLAB®. Once the simulated trajectory is generated, the final robot position is updated in the framework. In a real-life application of the framework on a robotic team, the simulated trajectory step is replaced with a navigation algorithm that physically drives the robots to their target destination. The framework then returns back to its first step (checking the robots' energies) and continues looping through these steps for the duration of the time set by the designer. A flowchart of the overall framework as described in this section is in Figure 6.2.

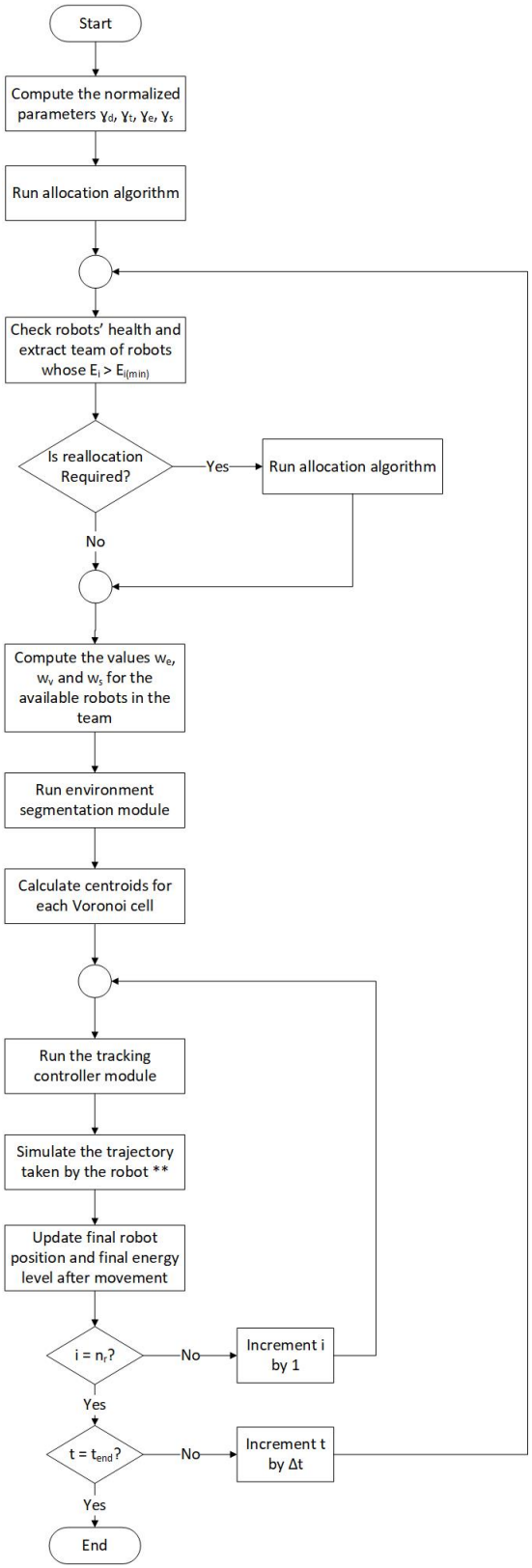


Figure 6.2: Flowchart of the proposed coverage control flowchart. Note that the step (marked with **) that simulates the robots' trajectories is not required in an actual implementation with real robots. This step is only included for the simulation studies.

6.3 Stability analysis of the proposed framework

In their work, Cortés et al. [126] show that the controller they propose leads the robots to converge to the largest invariant set, which in practice is the set of centroidal Voronoi tessellations. This means that the controller proposed by the authors drives the robots to their centroids, meaning that CVT is achieved, and therefore the cost function in (2.4) is minimized. The mathematical proof is provided in Section 2.3. A similar approach to prove that the proposed gradient-descent controller leads to CVT is also adopted in [1, 111, 250–252]. It is important to note that in all these works the coverage control formulation considers only static (time-invariant) density functions.

In [115], Cortés et al. use the same method to show that perfect tracking is obtained in that the team of robots is able to perfectly converge and track the moving (time-varying) CVT. However, to prove this, Cortés et al. assume that the density function is quasi-static, even though they propose a tracking controller for a time-varying environment. On the other hand, in [114, 116, 155, 156] the authors design controllers on the premise that the robots shall always arrive at the centroids, which in turn means that CVT is achieved. These works do not make the 'quasi-static density' assumption of Cortés et al. [115], but on the other hand, they do not present the same kind of proof that ascertains convergence to a CVT.

In this thesis the same approach used by Cortés et al. [115] is adopted to prove convergence, as well as the same assumption that the density functions, ϕ_i, for all the robots are quasi-static, in relation to the transient speeds of the robots. The aim of the following proof is to show that the proposed coverage controller drives the robots to the allocated centroids, and effectively lead them to a CVT. It is important to note that in the following derivation, p_i, C_{V_i} and u_i are $(n \times 1)$ vectors, whereas $\phi_i(q,t)$, M_{V_i}, E_i and $H_w(P, \mathcal{V}, t)$ are scalars.

Assumption 6.1. *The density functions $\phi_i(q,t)$, for $i = \{1, 2, \dots n_r\}$, are quasi-static relative to the movement of the robots. This means that:*

$$\begin{aligned} \dot{C}_{V_i} &\approx 0 \\ \dot{M}_{V_i} &\approx 0 \end{aligned} \tag{6.4}$$

Assumption 6.2. *The design parameters α_1 and $k_{i,e}$, $\forall i = \{1, \dots n_r\}$, are chosen such*

that $\alpha_1 k_{i,e} < 2(E_{i(max)} - E_{i(min)})$.

Theorem 6.1. *Under Assumptions 6.1 and 6.2, and subject to the non-linear dynamics in (6.5), the proposed energy-aware controller in (6.6) is a gradient descent algorithm for the coverage cost function* $\mathcal{H}_w$ *in (6.7), and that it drives the robots to a CVT, where:*

$$\begin{aligned} \dot{p_i} &= u_i \\ \dot{E_i} &= - \parallel u_i \parallel^2 = -u_i^\intercal u_i \end{aligned} \tag{6.5}$$

$$u_i = \dot{C}_{V_i} - \left(k_{i,e} \frac{E_i}{E_{i(max)}} + \frac{\dot{M}_{V_i}}{M_{V_i}} \right) \left(p_i - C_{V_i} \right) \tag{6.6}$$

$$\mathcal{H}_w \left(P, E, \mathcal{V}, t \right) = \sum_{i=1}^{n_r} \int_{V_i} \left(\parallel q - p_i \parallel^2 - w_i \right) \phi_i(q,t) \, \mathrm{d}q \tag{6.7}$$

and:

$$w_i = \alpha_1 \left(\frac{E_i - E_{i(max)}}{E_{i(max)} - E_{i(min)}} \right) + \alpha_2 \left(\frac{v_i - \max(\boldsymbol{v})}{\max(\boldsymbol{v}) - \min(\boldsymbol{v})} \right) - \frac{\alpha_3}{n_s} \sum_{k=1}^{n_s} s_k \tag{6.8}$$

Proof. Consider (6.7) to be a candidate Lyapunov function. $\mathcal{H}_w \left(P, E, \mathcal{V}, t \right)$ is a valid candidate since the weights w_i are negative, making $\mathcal{H}_w \left(P, E, \mathcal{V}, t \right)$ positive definite. Furthermore, $\mathcal{H}_w \left(P, E, \mathcal{V}, t \right)$ is also continuously differentiable. As discussed before, the robots are assumed to be holonomic with first order dynamics, given in (6.5). By applying the proposed control law in (6.6) to these dynamics, one can observe:

$$\dot{p_i} = \dot{C}_{V_i} - \left(k_{i,e} \frac{E_i}{E_{i(max)}} + \frac{\dot{M}_{V_i}}{M_{V_i}} \right) \left(p_i - C_{V_i} \right) \tag{6.9}$$

$$\dot{E_i} = -\left(\dot{C}_{V_i} - \left(k_{i,e} \frac{E_i}{E_{i(max)}} + \frac{\dot{M}_{V_i}}{M_{V_i}} \right) \left(p_i - C_{V_i} \right) \right)^\intercal \left(\dot{C}_{V_i} - \left(k_{i,e} \frac{E_i}{E_{i(max)}} + \frac{\dot{M}_{V_i}}{M_{V_i}} \right) \left(p_i - C_{V_i} \right) \right) \tag{6.10}$$

To show that (6.9) and (6.10) make $\mathcal{H}_w \left(P, E, \mathcal{V}, t \right)$ in (6.7) a gradient descent, the time derivative of $\mathcal{H}_w$ in the direction of p_i and E_i is computed:

$$\frac{\mathrm{d}\mathcal{H}_w}{\mathrm{d}t} = \sum_{i=1}^{n_r} \frac{\partial \mathcal{H}_w}{\partial p_i}\dot{p}_i + \frac{\partial \mathcal{H}_w}{\partial E_i}\dot{E}_i \tag{6.11}$$

By applying the convention that the partial derivative of a scalar by a column vector leads to a row vector, and by using (6.7) and (6.8) and deriving $\mathcal{H}_w$ with respect to vector p_i, it can be noted that:

$$\frac{\partial \mathcal{H}_w}{\partial p_i} = 2M_{V_i}\left(p_i - C_{V_i}\right)^{\intercal} \tag{6.12}$$

where M_{V_i} is the mass scalar. The result in (6.12) returns $(1 \times n)$ vector. Similarly, using (6.3), (6.7) and (6.8) and deriving $\mathcal{H}_w$ with respect to scalar E_i yields,

$$\frac{\partial \mathcal{H}_w}{\partial E_i} = \frac{-\alpha_1}{E_{i(max)} - E_{i(min)}} M_{V_i} \tag{6.13}$$

where $\frac{\partial \mathcal{H}_w}{\partial E_i}$ is a scalar since both $\mathcal{H}_w$ and E_i are also scalars. Additionally, under Assumption 6.1, then,

$$\begin{aligned} \dot{p}_i &= -\left(k_{i,e}\frac{E_i}{E_{i(max)}}\right)\left(p_i - C_{V_i}\right) \\ \dot{E}_i &= -\left(k_{i,e}\frac{E_i}{E_{i(max)}}\right)^2 \parallel p_i - C_{V_i} \parallel^2 \end{aligned} \tag{6.14}$$

By substituting (6.12), (6.13) and (6.14) in (6.11), one can observe:

$$\begin{aligned} \frac{d\mathcal{H}_w}{dt} = \sum_{i=1}^{n_r} & -2M_{V_i}k_{i,e}\left(\frac{E_i}{E_{i(max)}}\right) \parallel p_i - C_{V_i} \parallel^2 \\ & + \frac{\alpha_1 M_{V_i}}{E_{i(max)} - E_{i(min)}}\left(k_{i,e}\left(\frac{E_i}{E_{i(max)}}\right)\right)^2 \parallel p_i - C_{V_i} \parallel^2 \end{aligned} \tag{6.15}$$

This leads to,

$$\frac{d\mathcal{H}_w}{dt} = \sum_{i=1}^{n_r} M_{V_i} \parallel p_i - C_{V_i} \parallel^2 k_{i,e}\left(\frac{E_i}{E_{i(max)}}\right)\left(-2 + k_{i,e}\left(\frac{E_i}{E_{i(max)}}\right)\left(\frac{\alpha_1}{E_{i(max)} - E_{i(min)}}\right)\right) \tag{6.16}$$

Since $0 < \frac{E_i}{E_{i(max)}} \leq 1$, then taking the extreme case when $\frac{E_i}{E_{i(max)}} = 1$ means that for $\frac{\mathrm{d}\mathcal{H}_w}{\mathrm{d}t} \leq 0$, the design parameters α_1 and $k_{i,e}$ must be chosen such that

$$\alpha_1 k_{i,e} < 2\left(E_{i(max)} - E_{i(min)}\right) \quad \forall i = \{1, \ldots n_r\}$$

Therefore, under the above condition, $\frac{\mathrm{d}\mathcal{H}_w}{\mathrm{d}t}$ is negative semi-definite. The set of points of p_i and E_i for which $\frac{\mathrm{d}\mathcal{H}_w}{\mathrm{d}t} = 0$ is given by,

$$p_i - C_{V_i} = 0 \qquad \text{or} \qquad E_i = 0 \tag{6.17}$$

By LaSalle's Invariance Principle, the location of the robots converges to the largest invariant set contained in $\frac{\mathrm{d}\mathcal{H}_w}{\mathrm{d}t} = 0$, which corresponds to $p_i - C_{V_i} = 0$ given that $E_i \neq 0$. This means that the proposed control law drives the robots to CVT. □

Remark 6.1. It is important to note that in this derivation, it is assumed that the neighbouring density functions do not have an effect on the movement of the robots, since by the design of this algorithm, the robots must move towards the centroids governed by their assigned density functions. A full derivation of this proof is found in Appendix A.

6.4 Validation experiments

In this section, a set of test cases are used to validate the complete proposed framework. Firstly, in Section 6.4.1 the novel concept of driving the robots towards their assigned density dependent centroids is tested. This ensures that for static density functions, the robots are able to cluster around the density functions that they have been allocated to. In this section, the framework with multiple, static density functions is also validated against an algorithm that assumes a single, multi-modal density function. Furthermore, in Section 6.4.2 the individual density functions are set to be time-varying. In this section, each module in the framework is analysed and validated, such that the expected operation of the framework is observed. In Section 6.4.3 a particular case study is analysed such that the framework is validated also in the presence of robot malfunctions. This ensures that as designed, the framework is able to deal with situational changes as they occur.

6.4.1 Multi-density coverage

The concept of treating the density functions as individual density functions is novel and necessary in a framework that seeks to exploit the heterogeneity of the robot team. This section analyses the performance of the framework in an application where the environment consists of three distinct and static density functions, that must be covered by ten robots of various capabilities. For this purpose, the controller module uses the proportional controller in (2.13) that is typical in Lloyd's algorithm, since the densities in operation are not time-varying and therefore, a tracking controller is not required. Swapping the controller proposed in Chapter 5 with the controller in (2.13), without affecting the overall operation of the framework, demonstrates the benefits of a modular framework. Additionally, for this test case, Algorithm 1.0 is chosen as the allocation algorithm. As discussed in Chapter 3, this may be easily swapped with Algorithm 2.0 accordingly.

The individual density functions (ϕ_1 to ϕ_3) used in this experiment are chosen as follows: ϕ_1 by (6.18), ϕ_2 by (6.19) and ϕ_3 by (6.20), where q_x and q_y represent the x and y coordinates of a discrete point q in the environment.

$$\phi_1(q) = e^{-\left((q_x+1)^2+q_y^2\right)} \tag{6.18}$$

$$\phi_2(q) = e^{-\left((q_x-2)^2+(q_y-3)^2\right)} \tag{6.19}$$

$$\phi_3(q) = e^{-\left((q_x+3)^2+(q_y+1)^2\right)} \tag{6.20}$$

Tables 6.2, 6.3 and 6.4 tabulate the initial conditions and all the parameters chosen for this simulation. For this particular test case, $\beta_1 = \beta_2 = \beta_3 = \beta_4 = 1$ in the cost function in (3.1) of the allocation algorithm, and $\alpha_1 = \alpha_2 = \alpha_3 = 1$ for the overall weight function in (4.1) of the power diagram weights.

Table 6.2: Density function parameters

	Density functions		
	ϕ_1	ϕ_2	ϕ_3
Requirements	UGV	UAV	UGV
	LiDAR	LiDAR	LiDAR
	Visual	Thermal	Visual
			Thermal
Reference point on density	(-1, 0)	(2, 3)	(-3, -1)

Table 6.3: Robot capabilities and the sensors' costs

	Robot Capabilities			
$\boldsymbol{R_1}$	UAV	LiDAR, cost=0.1	Thermal, cost=0.3	
$\boldsymbol{R_2}$	UAV	LiDAR, cost=0.1	Visual, cost=0.5	Thermal, cost=0.3
$\boldsymbol{R_3}$	UGV	LiDAR, cost=0.1	Visual, cost=0.5	
$\boldsymbol{R_4}$	UGV	LiDAR, cost=0.1	Visual, cost=0.5	Thermal, cost=0.3
$\boldsymbol{R_5}$	UGV	LiDAR, cost=0.1	Visual, cost=0.5	
$\boldsymbol{R_6}$	UAV	LiDAR, cost=0.1	Visual, cost=0.5	Thermal, cost=0.3
$\boldsymbol{R_7}$	UGV	LiDAR, cost=0.1	Visual, cost=0.5	Thermal, cost=0.3
$\boldsymbol{R_8}$	UAV	LiDAR, cost=0.1	Thermal, cost=0.3	
$\boldsymbol{R_9}$	UGV	LiDAR, cost=0.1	Visual, cost=0.5	Thermal, cost=0.3
$\boldsymbol{R_{10}}$	UGV	LiDAR, cost=0.1	Visual, cost=0.5	Thermal, cost=0.3

Table 6.4: Robot parameters

	$\boldsymbol{E_i}$	$\boldsymbol{E_{i(max)}}$	$\boldsymbol{E_{i(min)}}$	$\boldsymbol{v_{i(max)}}$	$\boldsymbol{p_i}$
$\boldsymbol{R_1}$	20	20	0.2	5	(-1.5, -2)
$\boldsymbol{R_2}$	15	15	0.2	5	(-0.5, -1.5)
$\boldsymbol{R_3}$	20	20	0.2	5	(3, -1.5)
$\boldsymbol{R_4}$	20	20	0.2	5	(1.5, -3)
$\boldsymbol{R_5}$	15	15	0.2	5	(-3, 2)
$\boldsymbol{R_6}$	10	10	0.2	5	(1, 0)
$\boldsymbol{R_7}$	15	15	0.2	5	(-2, 3)
$\boldsymbol{R_8}$	20	20	0.2	5	(-2.7, 2.5)
$\boldsymbol{R_9}$	20	20	0.2	5	(-3, -3)
$\boldsymbol{R_{10}}$	20	20	0.2	5	(-1.5, -3)

During the first phase of the framework, the allocation algorithm (Algorithm 1.0 in this case), is run to create the list of optimal allocations. For this test case, the optimal allocations returned by the allocation algorithm are tabulated in Table 6.5.

Table 6.5: Result returned by the allocation algorithm during the first stage of the framework

Optimal allocations	
R_1	ϕ_2
R_2	ϕ_2
R_3	ϕ_1
R_4	ϕ_1
R_5	ϕ_1
R_6	ϕ_2
R_7	ϕ_1
R_8	ϕ_2
R_9	ϕ_3
R_{10}	ϕ_3

Since robots R_1, R_2, R_6 and R_8 are the only UAVs in the team, then as expected,

Algorithm 1.0 allocated these robots to density ϕ_2 as it required UAVs that are equipped with a LiDAR sensor and a thermal sensor. Furthermore, R_9 and R_{10} are both UGVs and they are equipped with all of the sensors required by ϕ_1 and ϕ_3. However, since they are initially closer to ϕ_3, the allocation algorithm assigned them to ϕ_3. Finally, according to the evaluated cost matrix, the algorithm allocated R_3, R_4, R_5 and R_7 to ϕ_1.

During the second stage of the framework, according to the flowchart in Figure 6.2, the framework checks the current energy levels of the robots. Throughout this trial, it was noticed that with the initial energy levels, and the maximum energy levels set, none of the robots suffered from depleted energy for the duration of the experiment.

Figure 6.3 shows a spatial plot of the density functions in the given environment. Additionally, Figure 6.4 depicts three instances during this trial. These correspond to: the beginning of the trial at $t = 0s$, the progression of the robots towards their allocated density functions during at $t = 6.5s$, and the final CVT reached at the end of the trial at $t = 25s$. The latter shows that the robots end up grouped around the density functions that they were assigned to.

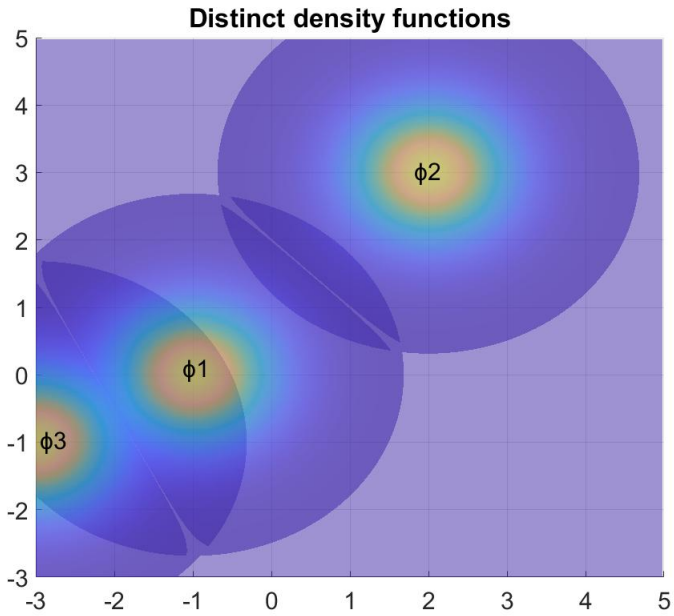
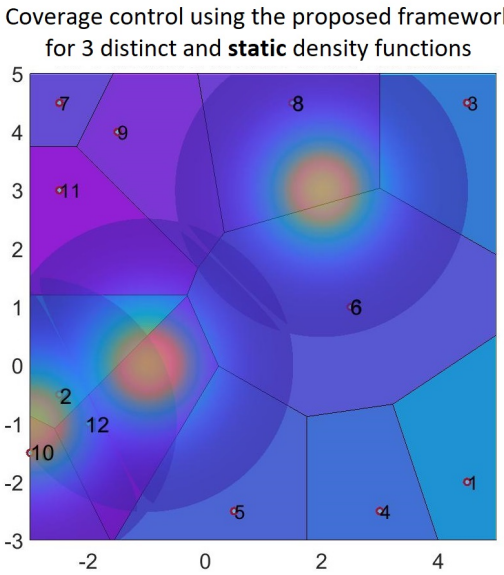


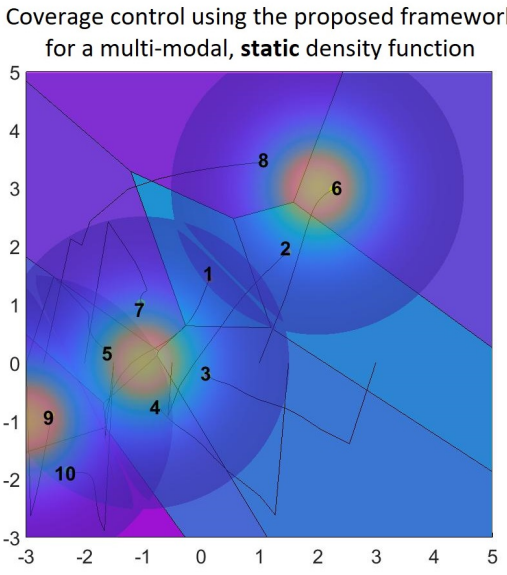
Figure 6.3: Three distinct density functions used during this trial. Warmer colours represent a higher importance value in the density functions.

The film-still shown in Figure 6.4c corresponds to the time when CVT is reached, since a few seconds before and after this moment, the robots may be observed to be

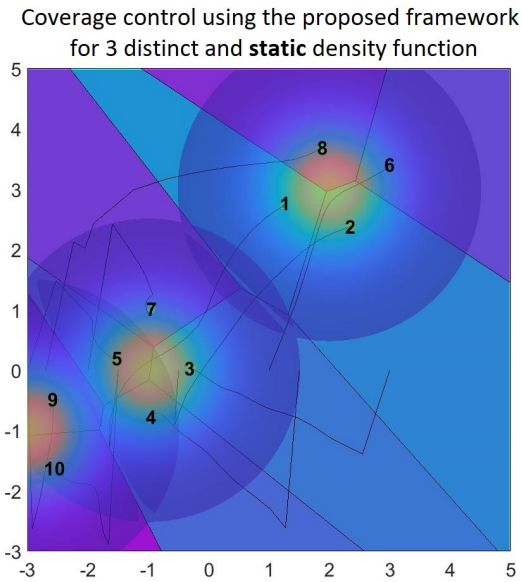
approximately in the same positions. This means that their current position is at the centroid of the respective Voronoi cell, weighted by their density function. In fact, from Figure 6.4c one observes that R_1, R_2, R_6 and R_8 end up covering ϕ_2, R_3, R_4, R_5 and R_7 end up covering ϕ_1, while R_9 and R_{10} end up covering ϕ_3. Since there were no re-allocations during the trial, then the original robot-to-density optimal allocations tabulated in Table 6.5 are expected to hold throughout the whole trial.



(a) Start of trial at $t = 0$



(b) Mid-way through trial at $t = 6.5s$



(c) End of trial when CVT is reached at $t = 25s$

Figure 6.4: Three instances until CVT is reached while using the framework

To show the effectiveness of the proposed approach that treats each density function as an individual density (as opposed to one multi-modal density), the same experiment is carried out by combining the density functions in (6.18), (6.19) and (6.20) into one multi-modal density function in (6.21). This multi-modal density function, having three peaks, is illustrated in Figure 6.5.

$$\phi(q) = e^{-\left((q_x+1)^2+q_y^2\right)} + e^{-\left((q_x-2)^2+(q_y-3)^2\right)} + e^{-\left((q_x+3)^2+(q_y+1)^2\right)}. \tag{6.21}$$

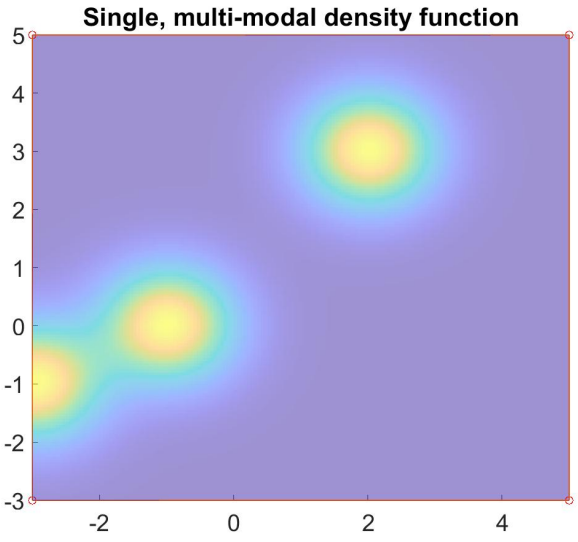


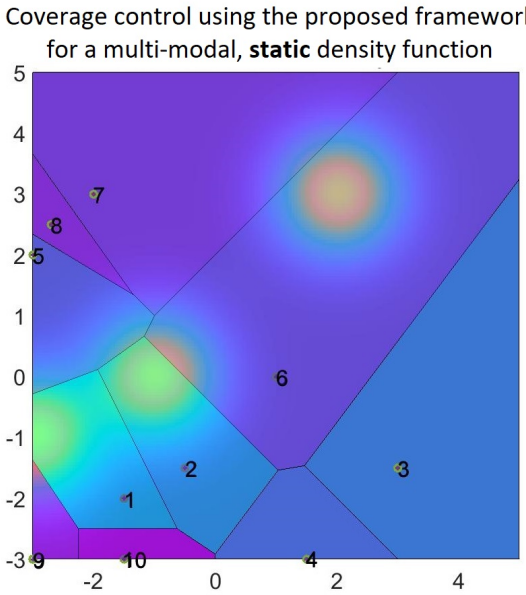
Figure 6.5: The multi-modal density function with three peaks used during this trial

Similar to the previous experiment with three distinct density functions, three instances were recorded when a multi-modal density function is adopted: the beginning of the trial at $t = 0s$, the progression of the robots towards their allocated density functions during mid-trial at $t = 6.5s$, and finally the final CVT reached at the end of the trial at $t = 25s$. These results are illustrated in Figure 6.6.

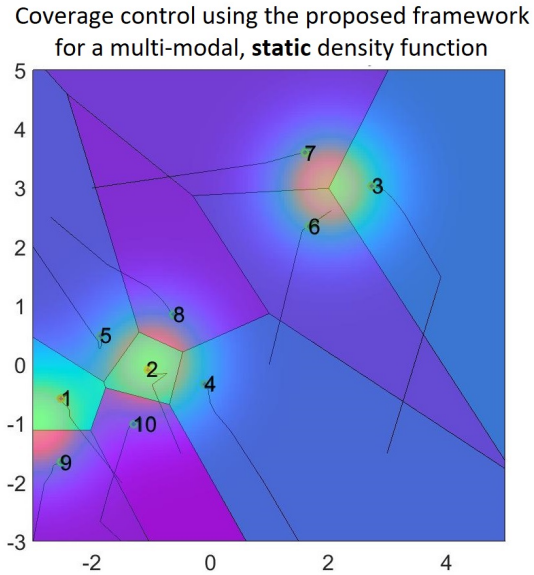
As can be seen in Figure 6.6, the robots moved towards the density peak that is closest to their initial position. Due to this, the final CVT ends up with robots R_1 and R_9 covering the peak of ϕ_3, R_2, R_4, R_5, R_8, and R_{10} covering the peak of ϕ_1, while R_3, R_7 and R_6 are covering the peak of ϕ_2. From Table 6.2 it is known that ϕ_2 requires a UAV to cover it, possibly because this area is inaccessible for UGVs. By using a multi-modal density function and having the coverage of the area rely solely on the initial robot positions, the system ends up with two UGVs, R_3 and R_7, trying to access that area.

In practice, this would result in a failed operation if the area being described by ϕ_2 is inaccessible to UGVs.

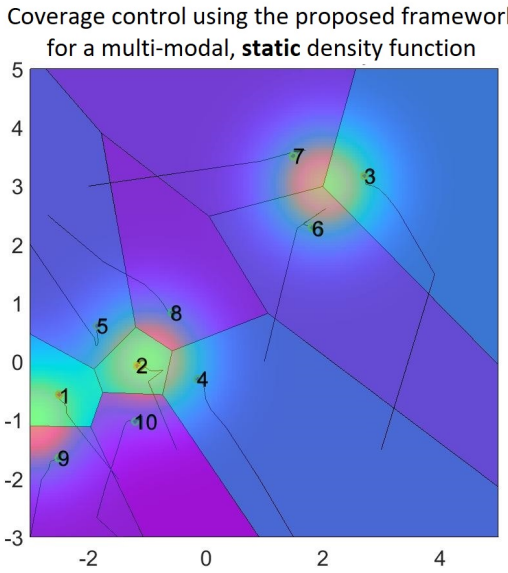
Furthermore, if the robots are deployed from different initial positions, different robots would end up covering the different peaks, as can be illustrated in Figure 6.7. Using a single, multi-modal density function may be sufficient for homogeneous teams where the robots in the team are all of the same type and all equipped with the same sensing capabilities. However, this shall still leave the coverage control problem dependent on the initial locations of the robots and hence, removing any degree of freedom over where the robots' placement in the environment.



(a) Start of trial at $t = 0$



(b) Mid-way through trial at $t = 6.5s$



(c) End of trial when CVT is reached at $t = 25s$

Figure 6.6: Three instances until CVT is reached when using a single, multi-modal density function

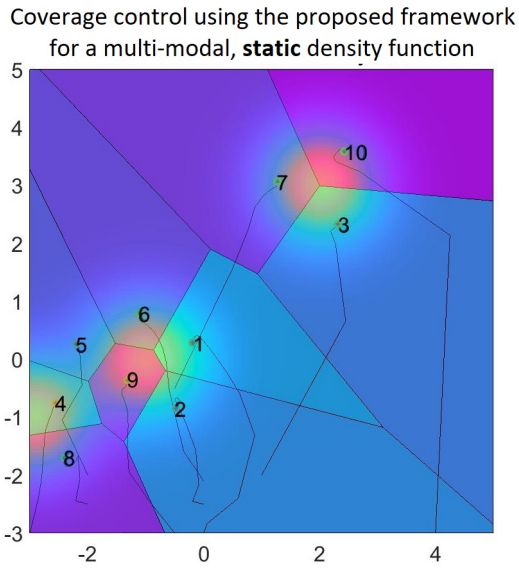


Figure 6.7: Different initial positions resulting in a different CVT with a multi-modal density function

6.4.2 Complete framework in operation

In this test case, the framework operates with three distinct density functions that are set to move in space with time, as follows,

$$\phi_1(q) = e^{-\left(\left(q_x - 2\sin\left(\frac{t}{\tau}\right)+1\right)^2 + q_y^2\right)} \tag{6.22}$$

$$\phi_2(q) = e^{-\left(\left(q_x - 2\sin\left(\frac{t}{\tau}\right)-2\right)^2 + \left(q_y - 3\right)^2\right)} \tag{6.23}$$

$$\phi_3(q) = e^{-\left(\left(q_x - 2\sin\left(\frac{t}{\tau}\right)+3\right)^2 + \left(q_y + 1\right)^2\right)} \tag{6.24}$$

For this experiment, the density requirements and robot capabilities are tabulated in Tables (6.6), (6.7) and (6.8). For this test case, $\beta_1 = \beta_2 = \beta_3 = \beta_4 = 1$ for the cost function in (3.1) of the allocation algorithm. Additionally, in this particular case, a priority is given to the energy reserves of the robots when it comes to segmenting the environment using the power diagram module. For this purpose, $\alpha_1 = 10$, while $\alpha_2 = \alpha_3 = 1$. Finally, it is assumed that $k_{i,e} = 1$ for all the robots in the team.

Table 6.6: Density function parameters

	Density functions		
	ϕ_1	ϕ_2	ϕ_3
Requirements	UGV	USV	UAV
	LiDAR	Visual	Thermal
	Visual		
Reference point on density	(-1, 0)	(2, 3)	(-3, -1)

Table 6.7: Robot capabilities and the sensors' costs

	Robot Capabilities			
R_1	UGV	LiDAR, cost=0.1	Visual, cost=0.3	
R_2	UAV	Visual, cost=0.3	Thermal, cost=0.5	
R_3	USV	Visual, cost=0.3		
R_4	UGV	LiDAR, cost=0.1	Visual, cost=0.3	Thermal, cost=0.5
R_5	UGV	LiDAR, cost=0.1	Visual, cost=0.3	Thermal, cost=0.5
R_6	USV	Visual, cost=0.3		
R_7	UGV	LiDAR, cost=0.1	Visual, cost=0.3	
R_8	USV	Visual, cost=0.3		
R_9	USV	Visual, cost=0.3		
R_{10}	UAV	Thermal, cost=0.5		
R_{11}	UGV	LiDAR, cost=0.1	Visual, cost=0.3	
R_{12}	UAV	Visual, cost=0.3	Thermal, cost=0.5	

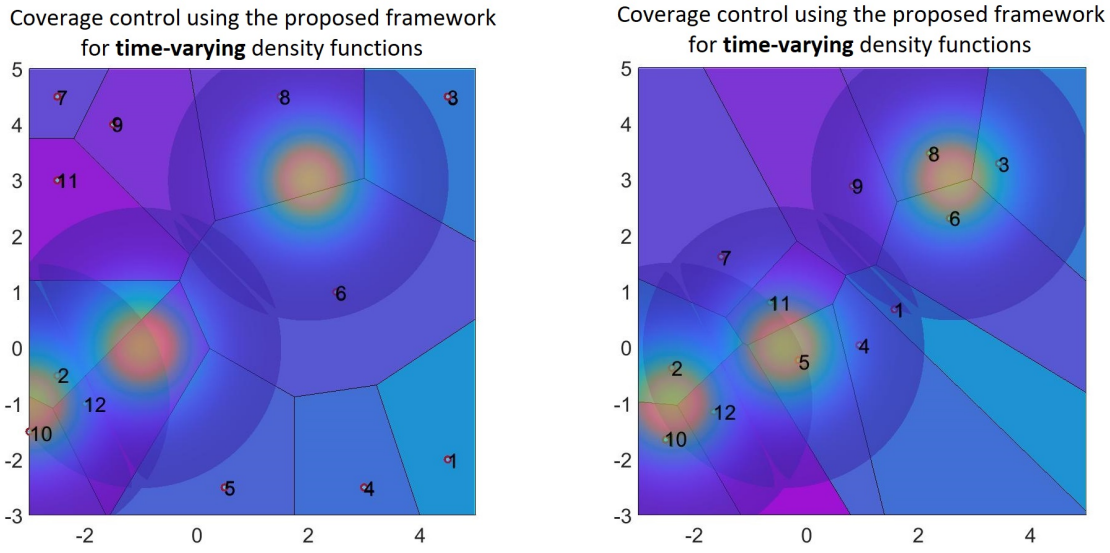
Table 6.8: Robot parameters

	E_i	$E_{i(max)}$	$E_{i(min)}$	$v_{i(max)}$	p_i
R_1	20	20	0.2	5	(4.5, -2)
R_2	20	20	0.2	5	(-2.5, -0.5)
R_3	20	20	0.2	5	(4.5, 4.5)
R_4	20	20	0.2	5	(3, -2.5)
R_5	20	20	0.2	5	(0.5, -2.5)
R_6	20	20	0.2	5	(2.5, 1)
R_7	20	20	0.2	5	(-2.5, 4.5)
R_8	20	20	0.2	5	(1.5, 4.5)
R_9	20	20	0.2	5	(-1.5, 4)
R_{10}	20	20	0.2	5	(-3, -1.5)
R_{11}	20	20	0.2	5	(-2.5, 3)
R_{12}	20	20	0.2	5	(-2, -1)

The optimal allocation algorithm allocates the robots in the team as tabulated in Table 6.9. This is as expected since ϕ_1 can only accept robots of the UGV type, and the robots that are not allocated to it are either UAVs or USVs. Similarly, all the robots allocated to ϕ_2 are USVs, unlike the other robots in the team, while those allocated to ϕ_3 are the only UAVs in the team. Furthermore, all the robots allocated to each density are equipped with the necessary sensors. Figures 6.8 and 6.9 illustrate the progression of the trial as the densities move in the environment and each robot in the team tracks the density function it has been assigned to.

Table 6.9: Result returned by the allocation algorithm during the first stage of the framework

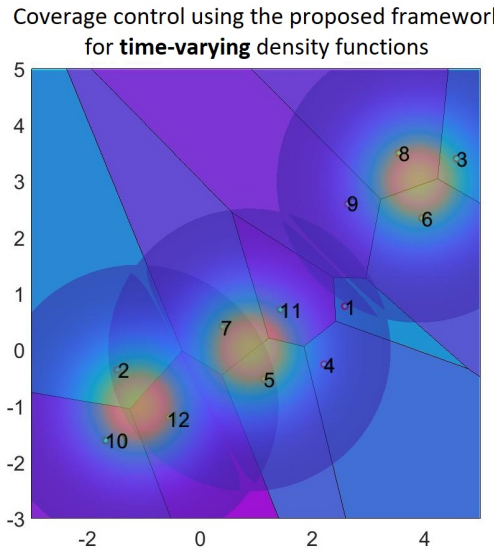
Optimal allocations	
R_1	ϕ_1
R_2	ϕ_3
R_3	ϕ_2
R_4	ϕ_1
R_5	ϕ_1
R_6	ϕ_2
R_7	ϕ_1
R_8	ϕ_2
R_9	ϕ_2
R_{10}	ϕ_{10}
R_{11}	ϕ_1
R_{12}	ϕ_3



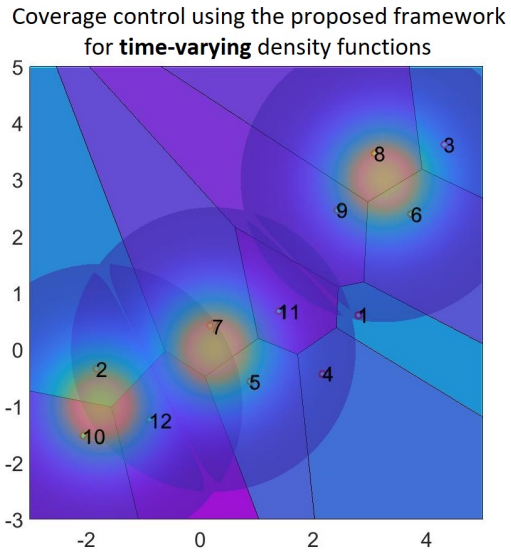
(a) Start of trial at $t = 0$

(b) Trial at $t = 6.5s$

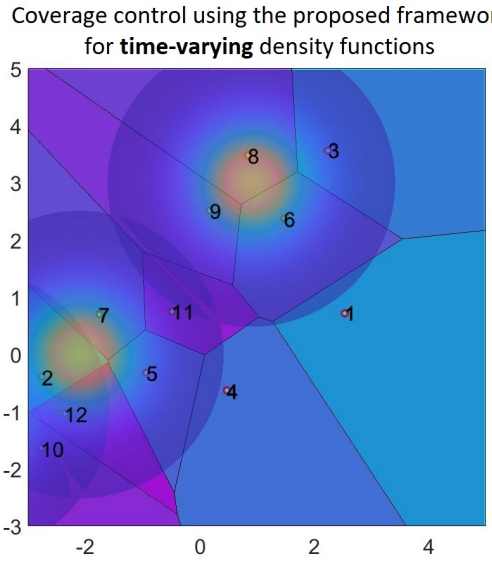
Figure 6.8: Coverage control using the proposed frameworks for three, distinct time-varying density functions



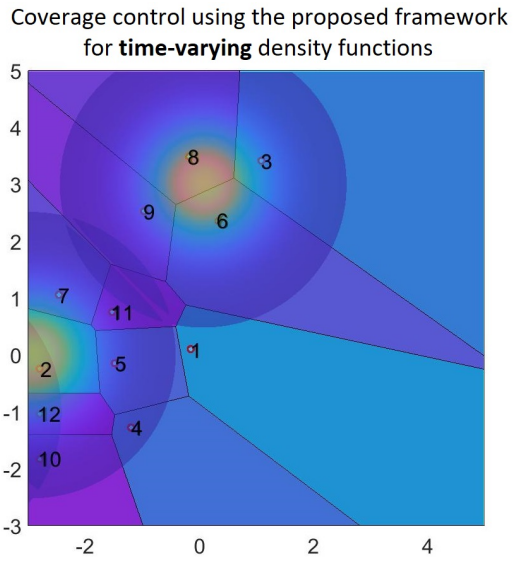
(a) Trial at $t = 25s$



(b) Trial at $t = 50s$



(c) Trial at $t = 75s$



(d) End of trial at $t = 100s$

Figure 6.9: Coverage control using the proposed frameworks for three, distinct time-varying density functions

In Figures 6.8a and 6.8b, the robots have just been deployed and are still seen to be in transit towards the density functions that they have been assigned to. As can be noted, especially from Figures 6.9a and 6.9b, the robots eventually cluster around their allocated density function. As these density functions move around in the environment, the robots may be seen to be tracking their movement while keeping their designated assignment to their density. During this trial, none of the robots ran out of energy while tracking the

density functions, and so, re-allocation was not necessary. In Figures 6.9c and 6.9d ϕ_3 no longer appears in the environment. This is because due to its nature (determined by (6.20)), at these moments, the higher importance parts of the density have moved out of the environment. Nevertheless, the respective robots are still seen to track its movement.

To analyse the effect of setting $\alpha_1 = 10$, i.e. setting a priority of the energy component in the environment segmentation process, the energy levels of the robots are stored for analysis. For simpler analysis, consider one timestamp during the overall trial such that the effect of the energy levels on the environment segmentation can be analysed. Consider the iteration at $t = 50s$, that is illustrated in Figure 6.9b. For each robot, the energy levels, the resulting Voronoi cell areas and Voronoi cell masses are tabulated in Table 6.10.

Table 6.10: Robot parameters

Robot	$\boldsymbol{E_i}$ **at** $t = 50s$	**Voronoi cell area**	**Voronoi cell mass**
$\boldsymbol{R_1}$	15.25	2.95	9.45×10^{-4}
$\boldsymbol{R_2}$	19.9	8.85	1.12
$\boldsymbol{R_3}$	18.34	2.44	0.27
$\boldsymbol{R_4}$	16.31	8.44	0.03
$\boldsymbol{R_5}$	17.52	2.69	0.71
$\boldsymbol{R_6}$	18.98	4.7	0.92
$\boldsymbol{R_7}$	17.86	0.23	1.49
$\boldsymbol{R_8}$	19.21	2.41	1.42
$\boldsymbol{R_9}$	18.34	12.3	0.49
$\boldsymbol{R_{10}}$	19.86	16.87	1.1
$\boldsymbol{R_{11}}$	17.06	2.33	0.22
$\boldsymbol{R_{12}}$	19.87	17.22	0.76

The analysis of the Voronoi cell area and mass should take into account the dependence of each robot on the density function that it is covering. For this purpose, analyzing the Voronoi cell areas across all the robots in the team does not do justice to the effectiveness of the power diagram design, as described in Section 2.4.1. Therefore, for this purpose one should consider the mass of the Voronoi cell as the better metric for comparison. Additionally, since the robots are tied to a particular density function, one needs to

consider each sub-team of robots for comparison. If the focus is placed on the sub-team of robots that is covering ϕ_1 (i.e. R_1, R_4, R_5, R_7 and R_{11}), one can see that the largest mass of the Voronoi cell, equal to 1.49, is that of R_7 because R_7 is the robot with the highest energy level in that sub-team. Alternatively, R_1 is the robot with the lowest energy level in that sub-team, and this is reflected in the mass of its Voronoi cell of 9.45×10^{-4}. If one focuses on the sub-team of robots that is covering ϕ_2, (i.e. R_3, R_6, R_8, and R_9), one can observe that R_8 is the robot with the largest mass of its Voronoi cell, equal to 1.42, because it is the robot with the highest energy level in that sub-team. Alternatively, for this sub-team, R_3 and R_9 are the robots with the lowest energy level and hence, they have the smallest Voronoi cell mass of 0.49.

Finally, when considering the sub-team that is covering ϕ_3 (i.e. R_2, R_{10} and R_{12}), R_2 is the robot with the highest Voronoi cell mass of 1.12, as opposed to the lowest Voronoi cell mass of 0.76 that belongs to R_{12}. Note that the energy levels of R_{10} and R_{12} are very similar and therefore, the Voronoi cell mass shall then depend on other factors, such as the clustering of the discrete points q in the power diagram. One must also note that when considering the framework as a whole, using the Voronoi cell mass as a comparison metric is more useful. This is because a larger Voronoi cell mass indicates a higher probability of an event occuring in that cell. This may be translated to having a larger workload since a robot might have more work to do in a higher likelihood area.

6.4.3 Case study: robot malfunction

One important feature of the proposed framework is that it is tolerant to situational changes. In the current implementation, the situational changes that the framework reacts to are those related to the energy levels of the robots. If the robots' energy levels fall below a certain threshold, $E_{i(min)}$, then a 'malfunction' in their operation would have occurred, and they are removed from the solver. In this case re-allocation of the robot-to-density pairings needs to be done such that the density that has been covered by the energy-depleted robot is re-allocated to another. On every discrete time iteration, the framework is checking the energy levels of the robots. If at the time of checking, a robot is found to be depleted of energy, its parameters are not included in the set of robot parameters that make up the team. In this same iteration, the framework re-runs the allocation algorithm to get a new set of optimal allocations, based on the newly reduced

team of robots. The framework then continues to operate the rest of the modules with the newly reformed team and the new optimal allocations.

Naturally, if there is only one robot from the team that is able to cover a specific density, and that robot stops without energy, then at that point that density function will remain unserviced due to the lack of resources in the team. It is important to note that although in the current implementation this robot health check corresponds to an energy level check, this may easily be expanded to include other health checks, such as checking whether the sensors onboard a robot are still operational, or whether some form of malfunction has occurred in the robots' actuators. These checks might become very useful in certain practical implementations of this framework.

To validate this feature, a specific test was designed, where one of the robots is forced to stop at a known point in time. In this experiment, Algorithm 2.0 is used as allocation algorithm. Seven robots form part of the team that must cover an environment that has three density functions. The density functions are governed by (6.22), (6.23) and (6.24). Additionally, in this experiment robot R_1 is forced to have its energy supply drain to $E_{i(min)}$ at $t = 30s$. For this experiment, the density requirements and robot capabilities are tabulated in Tables (6.11), (6.12) and (6.13). In this experiment, ϕ_3 has a group of sensors (marked by gray cells in Table 6.11) that are forced and must be all on-board the same robot for it to be a candidate. For this test, $\beta_1 = \beta_2 = \beta_3 = \beta_4 = 1$ for the cost function in (3.1) of the allocation algorithm. Furthermore, the priority gains of the power diagram are set as $\alpha_1 = \alpha_2 = \alpha_3 = 1$. Once again, for this experiment, it is assumed that $k_{i,e} = 1$ for all the robots in the team.

Table 6.11: Density function parameters

	Density functions		
	$\boldsymbol{\phi_1}$	$\boldsymbol{\phi_2}$	$\boldsymbol{\phi_3}$
Requirements	UGV	UGV	UGV
	LiDAR	Visual	LiDAR
	Thermal		Visual
			Thermal
Reference point on density	(-1, 0)	(2, 3)	(-3, -1)

Table 6.12: Robot capabilities and the sensors' costs

	Robot Capabilities			
$\boldsymbol{R_1}$	UGV	LiDAR, cost=0.1	Visual, cost=0.3	Thermal, cost=0.5
$\boldsymbol{R_2}$	UGV	LiDAR, cost=0.1	Thermal, cost=0.5	
$\boldsymbol{R_3}$	UGV	LiDAR, cost=0.1	Visual, cost=0.3	Thermal, cost=0.5
$\boldsymbol{R_4}$	UGV	LiDAR, cost=0.1	Visual, cost=0.3	
$\boldsymbol{R_5}$	UGV	Visual, cost=0.3		
$\boldsymbol{R_6}$	UGV	Visual, cost=0.3	Thermal, cost=0.5	
$\boldsymbol{R_7}$	UGV	LiDAR, cost=0.1	Thermal, cost=0.5	

Table 6.13: Robot parameters

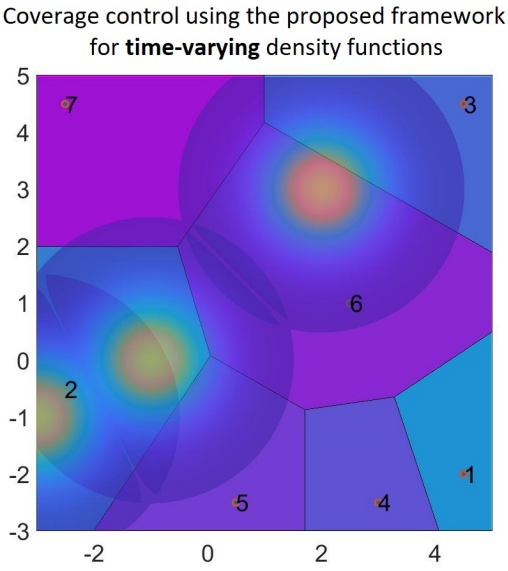
	$\boldsymbol{E_i}$	$\boldsymbol{E_{i(max)}}$	$\boldsymbol{E_{i(min)}}$	$\boldsymbol{v_{i(max)}}$	$\boldsymbol{p_i}$
$\boldsymbol{R_1}$	20	20	0.2	5	(4.5, -2)
$\boldsymbol{R_2}$	20	20	0.2	5	(-2.5, -0.5)
$\boldsymbol{R_3}$	20	20	0.2	5	(4.5, 4.5)
$\boldsymbol{R_4}$	20	20	0.2	5	(3, -2.5)
$\boldsymbol{R_5}$	20	20	0.2	5	(0.5, -2.5)
$\boldsymbol{R_6}$	20	20	0.2	5	(2.5, 1)
$\boldsymbol{R_7}$	20	20	0.2	5	(-2.5, 4.5)

As can be seen from Table 6.12, robots R_1 and R_3 are the only robots that are able to service ϕ_3 since they are the only ones that have all the sensors required in the forced group of sensors of ϕ_3. Furthermore, the initial robot-to-density optimal allocations according to Algorithm 2.0 are tabulated in Table 6.14. From the initial optimal allocations, it may be observed that ϕ_3 is only serviced by R_1. Hence, when R_1 forcibly runs out of energy at $t = 30s$, re-allocation is absolutely necessary since otherwise ϕ_3 ends up without any robot covering it. In this specific scenario, one would expect the allocation algorithm to allocate robot R_3 to ϕ_3 since it is the only other robot that can service it.

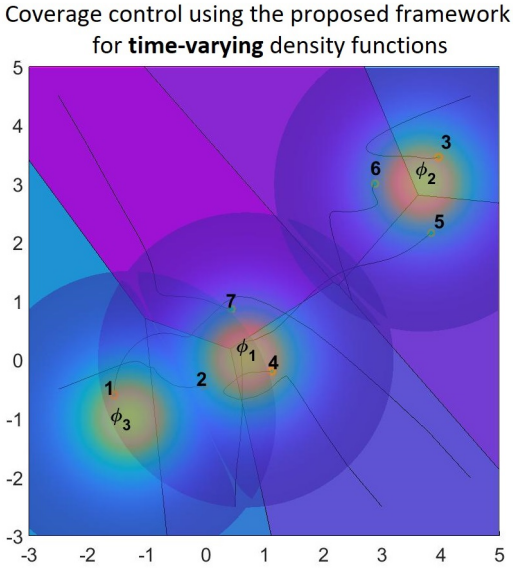
Table 6.14: Initial optimal allocations before R_1 is completely depleted of its energy reserves

Optimal allocations	
R_1	ϕ_3
R_2	ϕ_1
R_3	ϕ_2
R_4	ϕ_1
R_5	ϕ_2
R_6	ϕ_2
R_7	ϕ_1

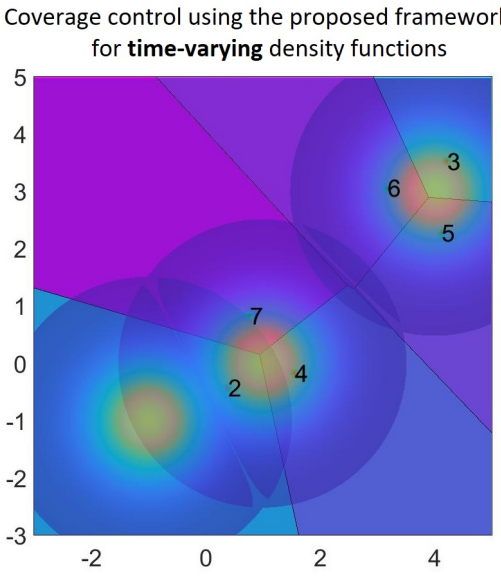
Figure 6.10 illustrates a number of steps during the coverage control operation. At $t = 0s$ the robots are still in their initial position, but as time goes by they may be seen moving towards their assigned densities according to the optimal allocations in Table 6.14. This is illustrated in Figure 6.10, where it may be observed that R_3, R_5 and R_6 are covering ϕ_2, while R_2, R_4 and R_7 are covering ϕ_1 and finally, R_1 is covering ϕ_3.



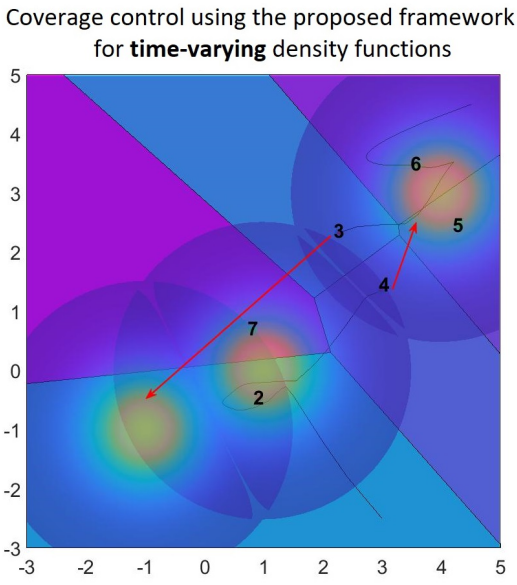
(a) Trial at $t = 0s$



(b) Trial at $t = 20s$



(c) Trial at $t = 30s$



(d) End of trial at $t = 35s$

Figure 6.10: Coverage control progression when a robot malfunctions at $t = 30s$

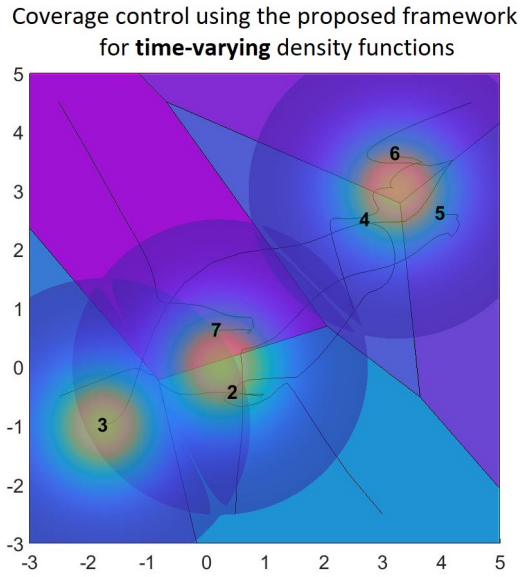


Figure 6.11: Trial at $t = 50s$ - Coverage control progression when a robot malfunctions at $t = 30s$

At $t = 30s$, as illustrated in Figure 6.10c, R_1 runs out of energy. In fact, it may be observed that at this point in time, it has been removed from the overall problem, and consequently, ϕ_3 is not being covered by any robot. At this moment, re-allocation of the robots occurs and R_3 is re-allocated to ϕ_3. This is because it is the only other robot in the team that is able to service ϕ_3 through its capabilities. The ILP constraint that all densities must be serviced ensures that this happens. The result of the re-allocation process is tabulated in Table 6.15. It can be noted that during this re-allocation, R_4 was re-allocated to ϕ_2. These optimal re-allocations yield the lowest overall cost from all the possible allocation pairings. Figure 6.10d illustrates R_3 leaving the sub-team that is covering ϕ_2 to make its way towards ϕ_3, while R_4 is also seen leaving ϕ_1 to make its way to ϕ_2. Figure 6.11 shows R_3 arriving at its designated destination to cover ϕ_3 in the absence of R_1.

Table 6.15: Initial optimal allocations before R_1 is completely depleted of its energy reserves

Optimal allocations	
R_2	ϕ_1
R_3	ϕ_3
R_4	ϕ_2
R_5	ϕ_2
R_6	ϕ_2
R_7	ϕ_1

The aim of this experiment was to validate the framework's tolerance to malfunctions in the robot team. From this experiment, one can observe that the framework reacted swiftly to a robot's fault (depleted energy) and immediately re-allocated another equally-capable robot to the density that it was covering. After re-allocation took place, one observes that the newly-reduced team having robots with new roles, continues to track the moving density functions as expected.

6.5 Conclusions

In this chapter, the novel framework for coverage control applications proposed in this thesis was detailed and validated via simulation. The proposed framework is novel on many counts. First, in contrast to existing frameworks for multi-robot systems (MRSs), it focuses particularly on the coverage control problem. Additionally, the design of this framework is also modular, hence, allowing the easy replacement of each module with alternative components. For example, in the set of results presented in Section 6.4.1, it may be observed that the controller was easily changed from the tracking controller to the time-invariant proportional controller. Similarly, in Section 6.4.2, the allocation algorithm of choice was Algorithm 1.0, while in Section 6.4.3, the allocation algorithm of choice was Algorithm 2.0. This further shows that every module in the framework is easy replaced with an alternative algorithm.

Besides these novel features, the proposed framework expands on an important concept in coverage control applications when faced with an environment composed of mul-

tiple regions of interest. This concept is based on the importance of allocating robots to these multiple, distinct regions of interest and ensuring that the main goal of each robot is to cover the region of interest that it is assigned with. The thought process behind this design is to exploit the team heterogeneity as much as possible. For instance, if a density requires an unmanned surface vehicle (USV) in an aquatic region, it would be pointless to have an unmanned ground vehicle (UGV) attempt to approach the density that is covering the aquatic area.

In addition to these features, all of which were tested in realistic test scenarios, in this Chapter, a stability analysis of the coverage scheme is also presented. Under the assumption that the density functions are quasi-static, relative to the movement of the robots, this analysis shows that the multi-robot system (MRS) converges to a CVT and is able to track the movement of the density functions. This is also shown through simulations as the robots are seen to track the moving density functions.

Chapter 7

Conclusions

The use of single and multi-robot systems (MRSs) in everyday life is exponentially increasing as advancements in technology render commonplace robotic solutions more conceivable. What usually begins as a desired fictional robotic system, soon develops into a prototype through research, and through technological growth it often becomes economically feasible to be introduced to the market. This was the case with the robotic vacuum cleaner, which in a matter of a few decades was developed into an affordable robo tic system mainly relying on the cutting edge technology of visual or LiDAR-based simultaneous localization and mapping (SLAM) [253], as well as various area coverage algorithms.

As happened to single robot systems (SRSs), multi-robot systems (MRSs) are also gaining traction in their use in real-life applications. A multi-robot team is capable of solving complex tasks more efficiently than a single robot, especially when these tasks are spread over a large geographical area. For instance, this becomes very evident in service robots [254] that are tasked with performing timely and accurate actions in search and rescue missions [42, 48], surveillance [45–49], and object transportation applications [79–83]. An MRS is also beneficial in taking over from a human team in hazardous situations. Incidents such as the Fukushima nuclear accident in 2011, as well as natural disasters such as the 2004 Indian Ocean earthquake and tsunami, have triggered researchers into developing efficient MRSs that replace a human team to identify and locate victims in dangerous disaster zones. In a more recent case, during the COVID-19 pandemic, service robots were deployed to disinfect an environment for a number of days [255]. Having a human team perform such tasks puts the human beings at risk.

In particular, one very versatile application of MRSs is the optimal coverage control of a given environment. Coverage control may be required in various scenarios ranging from search and rescue, surveillance missions, as well as warehousing operations. When using an MRS to solve such coverage problems, a number of challenging constraints must typically be adhered to. These include, among others: the ability to conserve the robots' energy to allow the operation to last longer, the exploitation of the heterogeneity in the capabilities of the team members, and the consideration to the typically time-varying nature of the environment. The team of robots is often expected to operate autonomously and be able to react to changes occurring within the team itself (such as when a robot runs out of energy) or to changes occurring in the environment (such as the movement of shipwreck victims due to sea currents, as the robots are attempting to locate them).

In the literature, works on coverage control using MRSs mostly focus on a singular aspect or complexity of the coverage affecting the overall operation [105, 111, 113, 115–117, 120, 126, 163, 164]. Such works tend to focus either on the time-varying nature of the environment [114–119, 156], the energy or health awareness of the robots [1, 105–109], the effect of robot hardware constraints (such as actuator variations) on the coverage problem [110–113, 136, 139], or the multiple regions of interest that make up the environment [120–122]. Our argument is that these factors are all equally important, and must be simultaneously considered when designing an MRS that is meant to be used for generic coverage control. This is because an environment might be composed of multiple, time-varying regions of interest that an energy-limited heterogeneous team of robots must cover. However, each of these factors cannot be accounted for and studied in isolation since their effects are often interdependent and present at the same time.

Therefore, covering a typical environment with an MRS generally requires a good framework that is capable of addressing multiple factors often found in real-life scenarios. Unlike existing task allocation frameworks [57, 90, 197–200], a coverage control framework of the proposed kind aims to address the challenges of real-life coverage problems. For instance, the proposed coverage control framework starts by allocating robots to each area of interest in the environment, depending on the different capabilities of the robots. These allocations may need to be updated by time, if a change in the robots' status occurs or as a reaction to a change in the environment's requirements. Simultaneously, the framework is capable of establishing effective coverage of the environment by segmenting it into

robot-dependent cells, and driving the robots accurately and continuously to each of these cells. The manner in which these cells are segmented considers factors such as (but not limited to) the current energy levels of the robots, their sensory capabilities and also their maximum velocity. By using the novel energy-aware tracking controller, the robots are driven to and track the centre of their generated cells as they move by time. This controller leads to accurate tracking which maximises the coverage of the environment, and at the same time conserve the robots' energy as much as possible.

Such a holistic framework for coverage control is novel and has not been previously proposed in literature. Nonetheless this work builds upon previous research [1, 115, 120, 126] on various aspects of coverage control considered in isolation. The design of such framework is not trivial since *all* of the individual modules contribute to the goal of covering the environment in the presence of multiple constraints. The modularity of the framework is also beneficial as it allows the system designer to easily replace modules with alternative algorithms that might be better suited for some given specific coverage application.

The structure of the proposed framework is first introduced in Chapter 2. The framework itself is composed of three main modules. These modules together are able to disperse robots to specific regions of importance in the environment according to the requirements of each region and the capabilities of each robot in the team. In this manner, the framework is able to exploit the heterogeneity of the robotic team. More specifically, the first module that is executed is the multi-robot task allocation (MRTA) module that is detailed in Chapter 3. This module is designed to allocate robots to each region of importance within the given environment. The allocation algorithm first analyses the sensory and robot type requirements of each density function and then it allocates the robots by minimizing a particular cost function that factors in the robot capabilities as well as the densities' requirements. This optimization process is formulated as an integer linear programming problem, where the optimal allocations of robots to densities is chosen as the set that minimizes the overall cost. This cost function is dependent on four factors: the current energy levels of the robots, their maximum velocity, their sensors' performance and the robots' distance from the density functions. The design of this cost function allows for these factors to be modified according to the specific needs of the application. Additionally, in this thesis, two variants of the allocation scheme have been

proposed. The first variant, Algorithm 1.0, ensures that the robot capabilities fully match the requirements of the density functions. To relax this rigidness, and in an attempt to exploit better the capabilities of the robots, the second variant, Algorithm 2.0, allows the robots to partially match the requirements of the density functions. In both cases, however, an optimal allocation according to a specifically-designed multi-factor cost function, is achieved.

Unlike any other work in literature, this part of the framework strives to exploit the heterogeneity in the robot team when dispersing the robots about the environment. This means that specific areas shall have the optimal placement of robots according to what that area requires in terms of robot type and capabilities. This allocation algorithm, in conjunction with the other modules, drastically reduces the dependency of the coverage control solution on the initial positions of the robots. This is because the framework first allocates robots to specific density functions and then through the controller module drives them to the matched area. Hence, such a formulation eliminates the risk of having a large group of the robotic team servicing a single density function simply because they were initially deployed close to it. In addition, through the allocation algorithm, this framework also reduces the risk of having inappropriately-typed robots servicing a given density function. For instance, by solely relying on the initial position of the robots, one might end up having a ground vehicle trying to service an aquatic area, which is obviously an incorrect assignment. Therefore, the proposed allocation algorithm improves on the effectiveness of the framework by ensuring that the tasks allocated to the robots make sense and are truly optimal. A set of realistic simulation studies are presented in Chapter 3 to validate the proposed allocation algorithm.

Another stage in coverage control involves the segmentation of the environment into sub-areas (the Voronoi cells) such that the robots are moved to the centre of mass of these areas. By repeatedly moving the robots to the centroids of these generated Voronoi cells, the MRS is able to achieve maximum coverage of the environment. To further exploit the heterogeneity in the team, and to conserve the robots' energy as much as possible, the environment segmentation module in our framework considers factors similar to those in the cost function of the allocation algorithm. Chapter 4 details the design and implementation of this module. Basically, this module segments the environment according to the energy levels, the maximum velocities and the sensors' performance of

the robots in the team. The idea behind it is that the area generated for the robot that has the highest energy level, the best sensors' performance and the best maximum velocity, should be the largest area. In such a manner, this module matches the more capable robots with larger tasks, in an attempt to exploit the available resources better. This module is also designed to allow the designer to tune the priority of each factor that contributes to the segmentation process. This use of a power diagram that smartly segments the environment is also novel. In contrast, existing segmentation schemes consider only the distance or up to one factor, such as energy [1] or actuator limitations [111]. Chapter 4 presents a set of results that demonstrate the significance of this design as opposed to the standard generation of Voronoi cells that simply depend on the distance between the robot and each discrete point in the environment.

After allocating robots to the appropriate densities and segmenting the environment accordingly, the robots must then be driven to the centroids of the generated Voronoi cells. The framework is designed to cater for environments where the density functions move by time. For this purpose, a novel energy-aware tracking controller is proposed, to ascertain accurate tracking of the time-varying centroids. The design and implementation of this controller is presented in Chapter 5. This tracking controller is designed to account for the current energy reserve of the robot when issuing velocity commands. In this manner, the controller has dual goals, namely: to achieve good tracking and to conserve energy. Moreover, the design of this controller allows the designer to trade off between the energy awareness and tracking performance.

In light of the above, the main novel contributions of this thesis can be summarized as follows:

1. The coverage control modular framework proposed in this thesis is an architecture that allows a team of robots to cover multiple, time-varying regions of importance in an environment, by exploiting the heterogeneity of the team, and ensuring that energy is conserved as much as possible. The main novelty of this framework lies in the way it simultaneously addresses multiple factors that typify practical coverage applications.

2. An optimization-based robot-to-density allocation algorithm is proposed to assign robots to densities by minimizing a multi-faceted cost function. The designed cost function penalizes robots that have lower energy levels, poorer sensors' performance,

a longer distance to a density function and lower maximum velocities. This allocation algorithm is novel since it is the first of its kind that exploits the heterogeneity in the robot team in the presence of multiple importance areas that each require a specific set of robot capabilities.

3. The novel manner in which the environment is segmented according to practical factors like the robot's energy and capabilities. Unlike existing segmentation schemes, the proposed approach segments the environment according to multiple factors.

4. A novel energy-aware tracking controller module for coverage control is proposed. The novelty of this controller is that it strives to preserve the energy levels of the robots, which once again, contributes one of the main features of the framework. The proposed energy-aware controller also ascertains that CVT is reached, as discussed in the proof presented in Appendix A.

The modularity aspect of the framework allows for potential improvement, modification and replacement of each module, to address better the various priorities of different applications. Hence, the proposed framework inherently facilitates the inclusion of future work. One such modification that may be explored is in the allocation algorithm. This module may be modified such that rather than having robots being allocated to densities, the problem is reformulated to have *specific robot capabilities* being allocated to densities. This would allow more flexibility in the formulation of the constraints. For instance, the constraints of the optimization problem may be reformulated to request a specific number of LiDAR sensors required for a particular density function.

Another potential modification to the framework may be done when checking the health of the robots. This concept may be expanded to check not just the energy levels of the robots, but also the health of the sensors on-board the robots. An additional advancement of this study includes a set of experimental results using physical robots when they are applied to a real coverage control application. For instance, this may be achieved by deploying a real-life team of drones and unmanned surface vehicles (USVs) whose aim is to perform surveillance of a beach. Furthermore, the novel control law (5.3) proposed in this thesis can be modified to account for the limited control inputs of all real-life actuators.

In conclusion, this thesis has detailed the architecture, implementation and validation

of a novel multi-robot framework tailored for the coverage control problem. This goal is achieved through the interaction of various individual modules that were all designed to achieve the same aim. This aim is to have a multi-robot system (MRS) that is able to optimally cover an environment that is typified by multiple, time-varying regions of importance, while simultaneously exploiting the heterogeneity of the robot team and conserving the energy levels of the robots as much as possible. To the best of the author's knowledge such a holistic framework is the first of its kind in the research field of coverage control.

Appendix A

Proof of convergence of the proposed system to CVT

To prove that the proposed system converges to CVT, the following assumptions are made. It must be noted that in this derivation, p_i, C_{V_i} and u_i are $(n \times 1)$ vectors, whereas $\phi_i(q,t)$, M_{V_i}, E_i and $H_w(P,\mathcal{V},t)$ are scalars.

Assumption A.1. *The density functions $\phi_i(q,t)$, for $i = \{1,2,\dots n_r\}$, are quasi-static relative to the movement of the robots. This means that:*

$$\begin{aligned} \dot{C}_{V_i} &\approx 0 \\ \dot{M}_{V_i} &\approx 0 \end{aligned} \tag{A.1}$$

Assumption A.2. *The design parameters α_1 and $k_{i,e}$, $\forall i = \{1,\dots n_r\}$, are chosen such that $\alpha_1 k_{i,e} < 2(E_{i(max)} - E_{i(min)})$.*

Theorem A.1. *Under Assumptions A.1 and A.2, and subject to the non-linear dynamics in (A.2), the proposed energy-aware controller in (A.3) is a gradient descent algorithm for the coverage cost function $\mathcal{H}_w$ in (A.4), and that it drives the robots to a CVT, where:*

$$\begin{aligned} \dot{p}_i &= u_i \\ \dot{E}_i &= -\parallel u_i \parallel^2 = -u_i^{\intercal} u_i \end{aligned} \tag{A.2}$$

$$u_i = \dot{C}_{V_i} - \left(k_{i,e} \frac{E_i}{E_{i(max)}} + \frac{\dot{M}_{V_i}}{M_{V_i}} \right) \left(p_i - C_{V_i} \right) \tag{A.3}$$

$$\mathcal{H}_w\left(P,\mathcal{V},E,t\right)=\sum_{i=1}^{n_r}\int_{V_i}\left(\parallel q-p_i\parallel^2-w_i\right)\phi_i(q,t)\,\mathrm{d}q \tag{A.4}$$

and:

$$w_i=\alpha_1\left(\frac{E_i-E_{i(max)}}{E_{i(max)}-E_{i(min)}}\right)+\alpha_2\left(\frac{v_i-\max(\boldsymbol{v})}{\max(\boldsymbol{v})-\min(\boldsymbol{v})}\right)-\frac{\alpha_3}{n_s}\sum_{k=1}^{n_s}s_k \tag{A.5}$$

All the terms in this derivation have been defined throughout this thesis, specifically in Section 2.2, Section 4.2, Section 5.2 and Section 6.3.

Proof. Consider (A.4) to be a candidate Lyapunov function. $\mathcal{H}_w\left(P,E,\mathcal{V},t\right)$ is a valid candidate since the weights w_i are negative, making $\mathcal{H}_w\left(P,E,\mathcal{V},t\right)$ positive definite. Additionally, $\mathcal{H}_w\left(P,E,\mathcal{V},t\right)$ is also continuously differentiable. In order to prove Theorem A.1, the time derivative of $\mathcal{H}_w$ in the direction of p_i and E_i is computed. To compute $\frac{d\mathcal{H}_w}{dt}$, consider,

$$\frac{\mathrm{d}\mathcal{H}_w}{\mathrm{d}t}=\sum_{i=1}^{n_r}\frac{\partial\mathcal{H}_w}{\partial p_i}\dot{p_i}+\frac{\partial\mathcal{H}_w}{\partial E_i}\dot{E_i} \tag{A.6}$$

First, $\frac{\partial\mathcal{H}_w}{\partial p_i}$ is computed as follows,

$$\begin{aligned}\frac{\partial\mathcal{H}_w}{\partial p_i}=\frac{\partial}{\partial p_i}\Bigg[&\left(\int_{V_i}\left(\parallel q-p_i\parallel^2\right)\phi_i(q,t)\,\mathrm{d}q\right)-\alpha_1\int_{V_i}\left(\frac{E_i-E_{i(max)}}{E_{i(max)}-E_{i(min)}}\right)\phi_i(q,t)\,\mathrm{d}q\\&-\alpha_2\int_{V_i}\left(\frac{v_i-\max(\boldsymbol{v})}{\max(\boldsymbol{v})-\min(\boldsymbol{v})}\right)\phi_i(q,t)\,\mathrm{d}q+\frac{\alpha_3}{n_s}\int_{V_i}\left(\sum_{k=1}^{n_s}s_k\right)\phi_i(q,t)\,\mathrm{d}q\Bigg]\end{aligned} \tag{A.7}$$

Using the parallel axis theorem, the following section of the cost may be re-written as follows [126],

$$\int_{V_i}\left(\parallel q-p_i\parallel^2\right)\phi_i(q,t)\,\mathrm{d}q=J_{V_i,C_{Vi}}+M_{V_i}\parallel p_i-C_{Vi}\parallel^2 \tag{A.8}$$

where $J_{V_i,C_{Vi}}$ is the polar moment of inertia of a region V_i about its centroid C_{V_i}, and it is computed by,

$$J_{V_i,C_{Vi}}=\int_{V_i}\parallel q-C_{V_i}\parallel^2\phi_i(q,t)\,\mathrm{d}q$$

Then,

$$\begin{aligned}\frac{\partial \mathcal{H}_w}{\partial p_i} =& \frac{\partial}{\partial p_i}\left(J_{V_i,C_{Vi}} + M_{V_i} \parallel p_i - C_{Vi} \parallel^2 \right) \\ &+ \frac{\partial}{\partial p_i}\left(-\alpha_1 \int_{V_i} \left(\frac{E_i - E_{i(max)}}{E_{i(max)} - E_{i(min)}} \right) \phi_i(q,t)\, \mathrm{d}q \right) \\ &+ \frac{\partial}{\partial p_i}\left(-\alpha_2 \int_{V_i} \left(\frac{v_i - \max(\boldsymbol{v})}{\max(\boldsymbol{v}) - \min(\boldsymbol{v})} \right) \phi_i(q,t)\, \mathrm{d}q \right) \\ &+ \frac{\partial}{\partial p_i}\left(\frac{\alpha_3}{n_s} \int_{V_i} \left(\sum_{k=1}^{n_s} s_k \right) \phi_i(q,t)\, \mathrm{d}q \right)\end{aligned} \tag{A.9}$$

By evaluating each component and using the convention that the partial derivative of a scalar by a column vector leads to a row vector, it can be noted that:

$$\begin{aligned}&\frac{\partial}{\partial p_i}\left(J_{V_i,C_{Vi}} + M_{V_i} \parallel p_i - C_{Vi} \parallel^2 \right) = 2M_{V_i}(p_i - C_{Vi})^\intercal \\ &\frac{\partial}{\partial p_i}\left(-\alpha_1 \int_{V_i} \left(\frac{E_i - E_{i(max)}}{E_{i(max)} - E_{i(min)}} \right) \phi_i(q,t)\, \mathrm{d}q \right) = 0 \\ &\frac{\partial}{\partial p_i}\left(-\alpha_2 \int_{V_i} \left(\frac{v_i - \max(\boldsymbol{v})}{\max(\boldsymbol{v}) - \min(\boldsymbol{v})} \right) \phi_i(q,t)\, \mathrm{d}q \right) = 0 \\ &+ \frac{\partial}{\partial p_i}\left(\frac{\alpha_3}{n_s} \int_{V_i} \left(\sum_{k=1}^{n_s} s_k \right) \phi_i(q,t)\, \mathrm{d}q \right) = 0\end{aligned} \tag{A.10}$$

where M_{V_i} is the mass scalar. This is because $J_{V_i,C_{Vi}}$, E_i, v_i and s_k are not functions of p_i. Therefore,

$$\frac{\partial \mathcal{H}_w}{\partial p_i} = 2M_{V_i}\left(p_i - C_{V_i}\right)^\intercal. \tag{A.11}$$

where $\frac{\partial \mathcal{H}_w}{\partial p_i}$ returns a $(1 \times n)$ vector. Similarly, $\frac{\partial \mathcal{H}_w}{\partial E_i}$ is computed as follows,

$$\begin{aligned}\frac{\partial H_w}{\partial E_i} =& \frac{\partial}{\partial E_i}\left(J_{V_i,C_{Vi}} + M_{V_i} \parallel p_i - C_{Vi} \parallel^2 \right) \\ &+ \frac{\partial}{\partial E_i}\left(-\alpha_1 \int_{V_i} \left(\frac{E_i - E_{i(max)}}{E_{i(max)} - E_{i(min)}} \right) \phi_i(q,t)\, \mathrm{d}q \right) \\ &+ \frac{\partial}{\partial E_i}\left(-\alpha_2 \int_{V_i} \left(\frac{v_i - \max(\boldsymbol{v})}{\max(\boldsymbol{v}) - \min(\boldsymbol{v})} \right) \phi_i(q,t)\, \mathrm{d}q \right) \\ &+ \frac{\partial}{\partial E_i}\left(\frac{\alpha_3}{n_s} \int_{V_i} \left(\sum_{k=1}^{n_s} s_k \right) \phi_i(q,t)\, \mathrm{d}q \right)\end{aligned} \tag{A.12}$$

It can be noted that:

$$\begin{aligned}
&\frac{\partial}{\partial E_i}\Bigg(J_{V_i,C_{Vi}} + M_{V_i} \parallel p_i - C_{Vi} \parallel^2 \Bigg) = 0 \\
&\frac{\partial}{\partial E_i}\Bigg(-\alpha_2 \int_{V_i} \left(\frac{v_i - \max(\boldsymbol{v})}{\max(\boldsymbol{v}) - \min(\boldsymbol{v})} \right) \phi_i(q,t)\,\mathrm{d}q \Bigg) = 0 \\
&+ \frac{\partial}{\partial E_i}\Bigg(\frac{\alpha_3}{n_s} \int_{V_i} \left(\sum_{k=1}^{n_s} s_k \right) \phi_i(q,t)\,\mathrm{d}q \Bigg) = 0
\end{aligned} \tag{A.13}$$

This is because none of these terms are a function of E_i and therefore the derivative with respect to the scalar variable E_i is 0. Hence, consider the following,

$$\begin{aligned}
\frac{\partial \mathcal{H}_w}{\partial E_i} =& \frac{\partial}{\partial E_i}\Bigg(-\alpha_1 \int_{V_i} \left(\frac{E_i - E_{i(max)}}{E_{i(max)} - E_{i(min)} } \right) \phi_i(q,t)\,\mathrm{d}q \Bigg) \\
=& \frac{\partial}{\partial E_i}\Bigg(\frac{-\alpha_1}{E_{i(max)} - E_{i(min)}} \int_{V_i} E_i \phi_i(q,t)\,\mathrm{d}q \Bigg) + \frac{\partial}{\partial E_i}\Bigg(\frac{\alpha_1}{E_{i(max)} - E_{i(min)}} \int_{V_i} E_{i(max)} \phi_i(q,t)\,\mathrm{d}q \Bigg)
\end{aligned} \tag{A.14}$$

Note that $\frac{\partial}{\partial E_i}\Bigg(\frac{\alpha_1}{E_{i(max)} - E_{i(min)}} \int_{V_i} E_{i(max)} \phi_i(q,t)\,\mathrm{d}q \Bigg) = 0$ since $E_{i(max)}$ is a constant value. Hence,

$$\begin{aligned}
\frac{\partial \mathcal{H}_w}{\partial E_i} =& \frac{-\alpha_1}{E_{i(max)} - E_{i(min)}} \int_{V_i} \phi_i(q,t)\,\mathrm{d}q \\
=& \frac{-\alpha_1}{E_{i(max)} - E_{i(min)}} M_{V_i}
\end{aligned} \tag{A.15}$$

where $\frac{\partial \mathcal{H}_w}{\partial E_i}$ is a scalar since both $\mathcal{H}_w$ and E_i are both scalars. To be able to compute $\frac{d\mathcal{H}_w}{dt}$ consider, from (A.2) that:

$$\begin{aligned}
\dot{p}_i &= \dot{C}_{V_i} - \left(k_{i,e} \frac{E_i}{E_{i(max)}} + \frac{\dot{M}_{V_i}}{M_{V_i}} \right) \left(p_i - C_{V_i} \right) \\
\dot{E}_i &= -\Bigg(\dot{C}_{V_i} - \left(k_{i,e} \frac{E_i}{E_{i(max)}} + \frac{\dot{M}_{V_i}}{M_{V_i}} \right) \left(p_i - C_{V_i} \right) \Bigg)^{\mathsf{T}} \Bigg(\dot{C}_{V_i} - \left(k_{i,e} \frac{E_i}{E_{i(max)}} + \frac{\dot{M}_{V_i}}{M_{V_i}} \right) \left(p_i - C_{V_i} \right) \Bigg)
\end{aligned} \tag{A.16}$$

Under Assumption A.1, then,

$$\begin{aligned} \dot{p}_i &= -\left(k_{i,e}\frac{E_i}{E_{i(max)}}\right)\left(p_i - C_{V_i}\right) \\ \dot{E}_i &= -\left(k_{i,e}\frac{E_i}{E_{i(max)}}\right)^2 \parallel p_i - C_{V_i} \parallel^2 \end{aligned} \tag{A.17}$$

Equations A.11, A.15 and A.17 are substituted in equation A.6 resulting in the following,

$$\begin{aligned} \frac{d\mathcal{H}_w}{dt} = \sum_{i=1}^{n_r} -2M_{V_i}k_{i,e}\left(\frac{E_i}{E_{i(max)}}\right) \parallel p_i - C_{V_i} \parallel^2 \\ + \frac{\alpha_1 M_{V_i}}{E_{i(max)} - E_{i(min)}}\left(k_{i,e}\frac{E_i}{E_{i(max)}}\right)^2 \parallel p_i - C_{V_i} \parallel^2 \end{aligned} \tag{A.18}$$

This leads to,

$$\frac{d\mathcal{H}_w}{dt} = \sum_{i=1}^{n_r} M_{V_i} \parallel p_i - C_{V_i} \parallel^2 k_{i,e}\left(\frac{E_i}{E_{i(max)}}\right)\left(-2 + k_{i,e}\left(\frac{E_i}{E_{i(max)}}\right)\left(\frac{\alpha_1}{E_{i(max)} - E_{i(min)}}\right)\right) \tag{A.19}$$

Since $0 < \frac{E_i}{E_{i(max)}} \leq 1$, then taking the extreme case when $\frac{E_i}{E_{i(max)}} = 1$ means that for $\frac{\mathrm{d}\mathcal{H}_w}{\mathrm{d}t} \leq 0$, the design parameters α_1 and $k_{i,e}$ must be chosen such that

$$\alpha_1 k_{i,e} < 2\left(E_{i(max)} - E_{i(min)}\right) \quad \forall i = \{1, \ldots n_r\}$$

Therefore, under the above condition, $\frac{\mathrm{d}\mathcal{H}_w}{\mathrm{d}t}$ is negative semi-definite. The set of points of p_i and E_i for which $\frac{\mathrm{d}\mathcal{H}_w}{\mathrm{d}t} = 0$ is given by,

$$p_i - C_{V_i} = 0 \qquad \text{or} \qquad E_i = 0 \tag{A.20}$$

By LaSalle's Invariance Principle, the location of the robots converges to the largest invariant set contained in $\frac{\mathrm{d}\mathcal{H}_w}{\mathrm{d}t} = 0$, which corresponds to $p_i - C_{V_i} = 0$ given that $E_i \neq 0$. This means that the proposed control law drives the robots to CVT. □

References

[1] Andrew Kwok and Sonia Martínez. Energy-balancing cooperative strategies for sensor deployment. In *IEEE Conference on Decision and Control*, pages 6136–6141, 2007.

[2] P A Harsha Vardhini and K Murali Chandra Babu. Iot based autonomous robot design implementation for military applications. In *2022 IEEE Delhi Section Conference (DELCON)*, pages 1–5, 2022.

[3] Ho Seok Ahn, Chandan Datta, I-Han Kuo, Rebecca Stafford, Ngaire Kerse, Kathy Peri, Elizabeth Broadbent, and Bruce A. MacDonald. Entertainment services of a healthcare robot system for older people in private and public spaces. In *2015 6th International Conference on Automation, Robotics and Applications (ICARA)*, pages 217–222, 2015.

[4] Katalin Ferencz and József Domokos. Rapid prototyping of iot applications for the industry. In *2020 IEEE International Conference on Automation, Quality and Testing, Robotics (AQTR)*, pages 1–6, 2020.

[5] Lynne E Parker. *Heterogeneous Multi-Robot Cooperation*. PhD thesis, Massachusetts Institute of Technology, 1994.

[6] Lynne E Parker. Current Research in Multi-Robot Systems. *Journal of Artificial Life and Robotics 7*, 7(1):1–5, 2003.

[7] Alessandro Farinelli, Luca Iocchi, Daniele Nardi, and Abstract Multirobot. Multirobot Systems : A Classification Focused on Coordination. *IEEE Transactions on Systems, Man, and Cybernetics —Part B: Cybernetics*, 34(5):2015–2028, 2004.

[8] Yifan Cai and Simon X Yang. A Survey on Multi-robot Systems. In *World Automation Congress 2012*, pages 1–6, 2012.

[9] Hejin Zhang, Zhiyun Zhao, Ziyang Meng, and Zongli Lin. Experimental verification of a multi-robot distributed control algorithm with containment and group dispersion behaviors: The case of dynamic leaders. *IEEE/CAA Journal of Automatica Sinica*, 1(1):54–60, 2014.

[10] Lynne E. Parker. *Multiple Mobile Robot Systems*, pages 921–941. Springer Berlin Heidelberg, Berlin, Heidelberg, 2008.

[11] Yu Zhang and Lynne E Parker. Task Allocation with Executable Coalitions in Multirobot Tasks. In *IEEE International Conference on Robotics and Automation*, pages 3307–3314, Saint Paul, Minnesota, USA, 2012.

[12] Stephanie Gil, Swarun Kumar, Mark Mazumder, Dina Katabi, Daniela Rus, Stephanie Gil, Mark Mazumder, Dina Katabi, and Daniela Rus. Guaranteeing Spoof-Resilient Multi-Robot Networks. In *Proceedings of Robotics: Science and Systems*, Rome, Italy, 2015.

[13] James Stephan, Jonathan Fink, Vijay Kumar, and Alejandro Ribeiro. Hybrid Architecture for Communication-Aware Multi-Robot Systems. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 5269–5276, Stockholm, Sweden, 2016.

[14] Yongcan Cao, Wenwu Yu, Wei Ren, and Guanrong Chen. An overview of recent progress in the study of distributed multi-agent coordination. *IEEE Transactions on Industrial Informatics*, 9(1):427–438, 2013.

[15] Avinash Gautam and Sudeept Mohan. A review of research in multi-robot systems. In *IEEE 7th International Conference on Industrial and Information Systems (ICIIS)*, pages 1–5, 2012.

[16] Rajesh Doriya, Siddharth Mishra, and Swati Gupta. A brief survey and analysis of multi-robot communication and coordination. *International Conference on Computing, Communication and Automation, ICCCA 2015*, pages 1014–1021, 2015.

[17] Jacques Ferber and Gerhard Weiss. *Multi-agent systems: an introduction to distributed artificial intelligence*, volume 1. Addison-wesley Reading, 1999.

[18] Reid G Simmons, David Apfelbaum, Wolfram Burgard, Dieter Fox, Mark Moors, Sebastian Thrun, and Håkan L S Younes. Coordination for Multi-Robot Exploration and Mapping. In *Proceedings of the Seventeenth National Conference on Artificial Intelligence and Twelfth Conference on Innovative Applications of Artificial Intelligence*, pages 852–858, 2000.

[19] Anthony Barrett, Gregg Rabideau, Tara Estlin, and Steve Chien. Coordinated continual planning methods for cooperating rovers. In *2001 IEEE Aerospace Conference Proceedings (Cat. No. 01TH8542)*, volume 1, pages 1–133. IEEE, 2001.

[20] Ratnesh Kumar and James A Stover. A behavior-based intelligent control architecture with application to coordination of multiple underwater vehicles. *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans*, 30(6):767–784, 2000.

[21] Manuela M. Veloso and Daniele Nardi. Special Issue on Multirobot Systems. *Proceedings of the IEEE*, 94(7):1253–1256, 2006.

[22] Tucker Balch and Ronald C Arkin. Behavior-based formation control for multirobot teams. *IEEE transactions on robotics and automation*, 14(6):926–939, 1998.

[23] Hiroki Takeda, Yasuhisa Hirata, Zhi-Dong Wang, and Kazuhiro Kosuge. Collision avoidance algorithm for two tracked mobile robots transporting a single object in coordination based on function allocation concept. In *Distributed Autonomous Robotic Systems 5*, pages 155–164. Springer, 2002.

[24] Israel A Wagner, Yaniv Altshuler, Vladimir Yanovski, and Alfred M Bruckstein. Cooperative cleaners: A study in ant robotics. *The International Journal of Robotics Research*, 27(1):127–151, 2008.

[25] Paul Rybski, Sascha Stoeter, Chris Wyman, and Maria Gini. A cooperative multi-robot approach to the mapping and exploration of mars. In *AAAI/IAAI*, pages 798–799, 1997.

[26] Marios M Polycarpou, Yanli Yang, and Kevin M Passino. A cooperative search framework for distributed agents. In *Proceeding of the 2001 IEEE International Symposium on Intelligent Control (ISIC'01)(Cat. No. 01CH37206)*, pages 1–6. IEEE, 2001.

[27] Jong-Hwan Kim and Prahlad Vadakkepat. Multi-agent systems: a survey from the robot-soccer perspective. *Intelligent Automation & Soft Computing*, 6(1):3–17, 2000.

[28] Rachael N. Darmanin and Marvin K. Bugeja. A Review on Multi-Robot Systems Categorised by Application Domain *. In *25th Mediterranean Conference on Control and Automation (MED)*, pages 701–706, 2017.

[29] Yan Zhi, Nicolas Jouandeau, and Arab Ali Cherif. A Survey and Analysis of Multi-Robot Coordination. *International Journal of Advanced Robotic Systems, INTECH*, 10(12):399, 2013.

[30] Brian P Gerkey and Maja J. Mataric. A formal analysis and taxonomy of task allocation in multi-robot systems. *International Journal of Robotics Research*, 23(9):939–954, 2004.

[31] Alaa Khamis, Ahmed Hussein, and Ahmed Elmogy. *Multi-robot Task Allocation: A Review of the State-of-the-Art*. Springer, 2015.

[32] M. Bernardine Dias, Robert Zlot, Nidhi Kalra, and Anthony Stentz. Market-based multirobot coordination: A survey and analysis. In *Proceedings of the IEEE*, volume 94, pages 1257–1270, 2006.

[33] M Brambilla, E Ferrante, and M Birattari. Swarm robotics : A review from the swarm engineering perspective. In *Swarm Intelligence*, volume 7, pages 1–41. 2012.

[34] Marco Dorigo, Dario Floreano, Luca Maria Gambardella, Francesco Mondada, Stefano Nolfi, Tarek Baaboura, Mauro Birattari, Michael Bonani, Manuele Brambilla, Arne Brutschy, Daniel Burnier, Alexandre Campo, Anders Lyhne Christensen, Antal Decugniere, Gianni Di Caro, Frederick Ducatelle, Eliseo Ferrante, Alexander Forster, Javier Martinez Gonzales, Jerome Guzzi, Valentin Longchamp, Stephane Magnenat, Nithin Mathews, Marco Montes De Oca, Rehan O'Grady, Carlo Pinciroli, Giovanni

Pini, Philippe Retornaz, James Roberts, Valerio Sperati, Timothy Stirling, Alessandro Stranieri, Thomas Stutzle, Vito Trianni, Elio Tuci, Ali Emre Turgut, and Florian Vaussard. Swarmanoid: A novel concept for the study of heterogeneous robotic swarms. *IEEE Robotics and Automation Magazine*, 20(4):60–71, 2013.

[35] Benjamin T. Champion and Matthew A. Joordens. Underwater swarm robotics review. *10th System of Systems Engineering Conference, SoSE 2015*, pages 111–116, 2015.

[36] Poorva Agrawal. A Review on Multi Robot Cooperation Using Bio Inspired Neural Networks. *International Journal of Soft Computing, Mathematics and Control (IJSCMC)*, 2(4):15–24, 2013.

[37] Michael A. Goodrich and Alan C. Schultz. Human-Robot Interaction: A Survey. *Foundations and Trends in Human-Computer Interaction*, 1(3):203–275, 2007.

[38] Jessie Y C Chen and Michael J Barnes. Human – Agent Teaming for Multirobot Control : A Review of Human Factors Issues. *IEEE Transactions on Human-Machine Systems*, 44(1):13–29, 2014.

[39] Wang Guanghua, Li Deyi, Gan Wenyan, and Jia Peng. Study on Formation Control of Multi-Robot Systems. In *3rd International Conference on Intelligent System Design and Engineering Applications*, pages 1335–1339, 2013.

[40] Yang Mao, Gan Gui Yan, and Yan Tao Tian. A review of studies in flocking for multi-robot system. In *International Conference on Computer, Mechatronics, Control and Electronic Engineering, CMCE 2010*, volume 4, pages 28–31, 2010.

[41] David Portugal and Rui Rocha. A Survey on Multi-robot Patrolling Algorithms. In Luis M. Camarinha-Matos, editor, *Technological Innovation for Sustainability: Second IFIP WG 5.5/SOCOLNET Doctoral Conference on Computing, Electrical and Industrial Systems, DoCEIS 2011*, pages 139–146. Springer Berlin Heidelberg, 2011.

[42] Yugang Liu and Goldie Nejat. Robotic urban search and rescue: A survey from the control perspective, 2013.

[43] Daniel Cook, Andrew Vardy, and Ron Lewis. A survey of AUV and robot simulators for multi-vehicle operations. In *IEEE/OES Autonomous Underwater Vehicles, AUV 2014*, pages 1 – 8, 2014.

[44] Ji-Hwan Son and Hyo-Sung Ahn. Cooperative Reinforcement Learning: Brief Survey and Application to Bio-insect and Artificial Robot Interaction. In *Science And Technology*, pages 71–76, 2008.

[45] Andrei Stancovici, Mihai V Micea, and Vladimir Cretu. Cooperative Positioning System for Indoor Surveillance Applications. In *International Conference on Indoor Positioning and Indoor Navigation, (IPIN)*, number 1, pages 1–7, 2016.

[46] Supreeth Achar, Bharath Sankaran, Stephen Nuske, Sebastian Scherer, and Sanjiv Singh. Self-supervised segmentation of river scenes. In *2011 IEEE International Conference on Robotics and Automation*, pages 6227–6232, 2011.

[47] Stephen Nuske, Sanjiban Choudhury, Sezal Jain, Andrew Chambers, Luke Yoder, Sebastian Scherer, Lyle Chamberlain, Hugh Cover, and Sanjiv Singh. Autonomous exploration and motion planning for an unmanned aerial vehicle navigating rivers: Autonomous exploration and motion planning for a uav navigating rivers. *Journal of Field Robotics*, 32, 06 2015.

[48] Ricardo Mendonça, Mario Monteiro Marques, Francisco Marques, André Lourenço, Eduardo Pinto, Pedro Santana, Fernando Coito, Victor Lobo, and José Barata. A Cooperative Multi-Robot Team for the Surveillance of Shipwreck Survivors at Sea. In *MTS/IEEE Monterey OCEANS*, pages 2–7, 2016.

[49] Farshid Abbasi, Afshin Mesbahi, and Javad Mohammadpour Velni. A New Voronoi-Based Blanket Coverage Control Method for Moving Sensor Networks. *IEEE Transactions on Control Systems Technology*, pages 1–9, 2017.

[50] Nikhil Nigam, Stefan Bieniawski, Ilan Kroo, and John Vian. Control of multiple UAVs for persistent surveillance: Algorithm and flight test results. *IEEE Transactions on Control Systems Technology*, 20(5):1236–1251, 2012.

[51] Ethan Stump and Nathan Michael. Multi-robot persistent surveillance planning as a vehicle routing problem. In *IEEE International Conference on Automation Science and Engineering*, pages 569–575, 2011.

[52] Alejandro Pustowka and Eduardo F. Caicedo. Market-based task allocation in a multi-robot surveillance system. In *Brazilian Robotics Symposium and Latin American Robotics Symposium, SBR-LARS 2012*, pages 185–189, 2012.

[53] Seohyun Jeon, Minsu Jang, Seunghwan Park, Daeha Lee, Young Jo Cho, and Jaehong Kim. Task allocation strategy of heterogeneous multi-robot for indoor surveillance. In *8th International Conference on Ubiquitous Robots and Ambient Intelligence*, pages 169–173, 2011.

[54] Jason Gregory, Jonathan Fink, Ethan Stump, Jeffrey Twigg, John Rogers, Nick Fung, and Stuart Young. Application of Multi-Robot Systems to Disaster-Relief Scenarios with Limited Communication. *Springer Tracts in Advanced Robotics*, 113:639–653, 2016.

[55] C Archetti, D Feillet, a Hertz, and M G Speranza. The capacitated team orienteering and profitable tour problems. *Journal of the Operational Research Society*, 60(6):831–842, 2009.

[56] Marco A. Gutierrez, Suraj Nair, Rafael E. Banchs, Luis Fernando D Haro Enriquez, Andreea I. Niculescu, and Aravindkumar Vijayalingam. Multi-robot collaborative platforms for humanitarian relief actions. In *IEEE Region 10 Humanitarian Technology Conference, R10-HTC 2015 - co-located with 8th International Conference on Humanoid, Nanotechnology, Information Technology, Communication and Control, Environment and Management, HNICEM 2015*, 2016.

[57] Lynne E Parker. ALLIANCE : An Architecture for Fault Tolerant Multirobot Cooperation. *IEEE Transactions on Robotics and Automation*, 14(2):220–240, 1998.

[58] Miguel Schneider-Fontán and Maja J. Matarić. Territorial multi-robot task division. *IEEE Transactions on Robotics and Automation*, 14(5):815–822, 1998.

[59] Adam Lein and Richard T. Vaughan. Adapting to non-uniform resource distributions in robotic swarm foraging through work-site relocation. *IEEE/RSJ International Conference on Intelligent Robots and Systems, (IROS)*, pages 601–606, 2009.

[60] Micael S. Couceiro, Rui P. Rocha, Carlos M. Figueiredo, J. Miguel A. Luz, and N. M. Fonseca Ferreira. Multi-Robot Foraging based on Darwin's Survival of the Fittest. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 801–806, 2012.

[61] Fernando J Mendiburu, Marcos R A Morais, and Antonio M N Lima. Behavior coordination in multi-robot systems. In *IEEE International Conference on Automatica (ICA-ACCA)*, pages 1–7, 2016.

[62] Daito Sakai, Hiroaki Fukushima, and Fumitoshi Matsuno. Flocking for Multirobots Without Distinguishing Robots and Obstacles. *IEEE Transactions on Control Systems Technology*, pages 1–9, 2016.

[63] Dongbing Gu and Zongyao Wang. Leader-follower flocking: Algorithms and experiments. *IEEE Transactions on Control Systems Technology*, 17(5):1211–1219, 2009.

[64] Hung Manh La, Ronny Lim, and Weihua Sheng. Multirobot cooperative learning for predator avoidance. *IEEE Transactions on Control Systems Technology*, 23(1):52–63, 2015.

[65] Seoung Kyou Lee, Daniel Kim, Dong Suk Shin, Taeho Jang, and James Mclurkin. Distributed Deformable Configuration Control for Multi-Robot Systems. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 5347 – 5354, 2016.

[66] Jianping Wang, Yuesheng Gu, and Xiaomin Li. Multi-robot task allocation based on ant colony algorithm. *Journal of Computers*, 7, 09 2012.

[67] Juan Antonio Vazquez Trejo, Damiano Rotondo, Didier Theilliol, and Manuel Adam Medina. Observer-based quadratic boundedness leader-following control for multi-agent systems. *International Journal of Control*, 96(5):1314–1323, 2023.

[68] Juan Vazquez Trejo, Didier Theilliol, Manuel Adam Medina, Carlos Garcia-Beltrán, and Marcin Witczak. Leader-following formation control for networked multi-agent systems under communication faults/failures. In *14th International Conference on Diagnostics of Processes and Systems, DPS 2020*, 09 2020.

[69] B. Madhevan and M. Sreekumar. Tracking algorithm using leader follower approach for multi robots. *Procedia Engineering*, 64:1426–1435, 2013. International Conference on Design and Manufacturing (IConDM2013).

[70] Yunpeng Wang, Long Cheng, Zeng-guang Hou, and Min Tan. Optimal Formation of Multi-Robot Systems Based on a Recurrent Neural Network. *IEEE Transactions on Neural Networks and Learning Systems*, 27(2):1–11, 2016.

[71] Rattanapol Phanichnitinon, Tanakrit Hemwarangkoon, Jumpol Polvichai, Thanathep Boonpromma, and Kasin Jarutekumporn. Multi modular robots maneuver for geometry formation control. In *IEEE 7th International Workshop on Computational Intelligence and Applications, (IWCIA) 2014 - Proceedings*, pages 195–200, 2014.

[72] Seung Mok Lee, Hanguen Kim, Hyun Myung, and Xin Yao. Cooperative coevolutionary algorithm-based model predictive control guaranteeing stability of multirobot formation. *IEEE Transactions on Control Systems Technology*, 23(1):37–51, 2015.

[73] Jacopo Banfi, Alberto Quattrini Li, Nicola Basilico, Ioannis Rekleitis, and Francesco Amigoni. Asynchronous multirobot exploration under recurrent connectivity constraints. In *IEEE International Conference on Robotics and Automation*, volume 2016-June, pages 5491–5498, 2016.

[74] Kyle Cesare, Ryan Skeele, Soo-hyun Yoo, Yawei Zhang, and Geoffrey Hollinger. Multi-UAV Exploration with Limited Communication and Battery. In *IEEE International Conference on Robotics and Automation (ICRA)*, number 1, pages 2230–2235, 2015.

[75] Julian De Hoog, Stephen Cameron, and Arnoud Visser. Autonomous Multi-Robot Exploration in Communication-Limited Environments. In *11th Conference Towards Autonomous Robotic Systems*, 2010.

[76] Upma Jain, Ritu Tiwari, Samriddhi Majumdar, and Sanjeev Sharma. Multi Robot Area Exploration Using Circle Partitioning Method. In *International Symposium on Robotics and Intelligent Sensors*, volume 41, pages 383–387, 2012.

[77] Flavio Cabrera-Mora and Jizhong Xiao. A flooding algorithm for multirobot exploration. *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, 42(3):850–863, 2012.

[78] Peter Brass, Andrea Gasparri, Flavio Cabrera-mora, and Jizhong Xiao. Multi-robot Tree and Graph Exploration. In *IEEE International Conference on Robotics and Automation*, pages 2332–2337, 2009.

[79] Lynne E Parker. Multi-Robot Team Design for Real-World Applications. In *Distributed Autonomous Robotic Systems 2*, number 1, chapter 4, pages 91–102. Springer Japan, 1996.

[80] Maja J. Matarić, Martin Nilsson, and Kristian T Simsarian. Cooperative Multi-Robot Box-Pushing. In *IEEE/RSJ International Conference onIntelligent Robots and Systems 95. 'Human Robot Interaction and Cooperative Robots'*, number Figure 1, pages 556–561, 1995.

[81] Arnab Ghosh, Avishek Ghosh, Amit Konar, Senior Member, and R Janarthanan. Multi-Robot Cooperative Box-pushing Problem Using Multi-objective Particle Swarm Optimization Technique. In *World Congress on Information and Communication Technologies, (WICT)*, pages 272–277, 2012.

[82] R.G. Brown and J.S. Jennings. A pusher/steerer model for strongly cooperative mobile robot manipulation. In *IEEE/RSJ International Conference on Intelligent Robots and Systems. Human Robot Interaction and Cooperative Robots*, pages 562–568, 1995.

[83] Dominik Sieber, Frederik Deroo, and Sandra Hirche. Formation-based approach for multi-robot cooperative manipulation based on optimal control design. In *IEEE International Conference on Intelligent Robots and Systems*, pages 5227–5233, 2013.

[84] Weiwei Wan, Rui Fukui, Masamichi Shimosaka, Tomomasa Sato, and Yasuo Kuniyoshi. Cooperative manipulation with least number of robots via robust caging. In *IEEE/ASME International Conference on Advanced Intelligent Mechatronics, AIM*, pages 896–903, 2012.

[85] Andrea Zambelli Bais, Sebastian Erhart, Luca Zaccarian, and Sandra Hirche. Dynamic load distribution in cooperative manipulation tasks. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2380–2385, 2015.

[86] Angelos Amanatiadis, Christopher Henschel, Bernd Birkicht, Benjamin Andel, Konstantinos Charalampous, Ioannis Kostavelis, Richard May, and Antonios Gasteratos. AVERT : An Autonomous Multi-Robot System for Vehicle Extraction and Transportation. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 1662–1669, 2015.

[87] Alessandra Rossi, Mariacarla Staffa, and Silvia Rossi. Supervisory Control of Multiple Robots through Group Communication. *IEEE Transactions on Cognitive and Developmental Systems*, 8920(c):1–12, 2016.

[88] Jonathan Cacace, Alberto Finzi, and Vincenzo Lippiello. Implicit Robot Selection for Human Multi-Robot Interaction in Search and Rescue Missions. In *25th IEEE International Symposium on Robot and Human Interactive Communication (RO-MAN)*, pages 803 – 808, 2016.

[89] Joan Saez-pons, Lyuba Alboul, and Jacques Penders. Experiments in Cooperative Human Multi-Robot Navigation. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 1 – 4, 2011.

[90] Brian P Gerkey and Maja J Mataric. Sold !: Auction Methods for Multirobot Coordination. *IEEE Transactions on Robotics and Automation*, 18(5):758–768, 2002.

[91] Fang Tang and Lynne E Parker. ASyMTRe : Automated Synthesis of Multi-Robot Task Solutions through Software Reconfiguration. In *IEEE International Conference on Robotics and Automation*, number April, pages 1501 – 1508, Barcelona, Spain, 2005.

[92] Ivana Budinská and Štefan Havlik. Task allocation within a heterogeneous multi-robot system. *2016 Cybernetics and Informatics, K and I 2016 - Proceedings of the 28th International Conference*, 2016.

[93] Ayush Dewan, Aravindh Mahendran, Nikhil Soni, and K. Madhava Krishna. Heterogeneous UGV-MAV exploration using integer programming. *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 5742–5749, 2013.

[94] Paolo Stegagno, Marco Cognetti, Lorenzo Rosa, Pietro Peliti, and Giuseppe Oriolo. Relative localization and identification in a heterogeneous multi-robot system. *Proceedings - IEEE International Conference on Robotics and Automation*, pages 1857–1864, 2013.

[95] El Houssein Chouaib Harik, Frédéric Guinand, Hervé Pelvillain, François Guérin, and Jean François Brethé. A decentralized interactive architecture for aerial and ground mobile robots cooperation. In *Proceedings - 2015 International Conference on Control, Automation and Robotics, ICCAR 2015*, pages 37–43, 2015.

[96] Thilo Weigel, Jens Steffen Gutmann, Markus Dietl, Alexander Kleiner, and Bernhard Nebel. CS Freiburg: Coordinating robots for successful soccer playing. *IEEE Transactions on Robotics and Automation*, 18(5):685–699, 2002.

[97] B Browning, J Bruce, M Bowling, and M Veloso. STP: skills, tactics, and plays for multi-robot control in adversarial environments. *IEEE Journal of Control and Systems Engineering*, 219(1):33–52, 2004.

[98] Peter Stone, Richard S Sutton, and Gregory Kuhlmann. Reinforcement Learning for RoboCup-Soccer Keepaway. *Adaptive Behavior*, 13(3):165–188, 2005.

[99] Shivaram Kalyanakrishnan, Yaxin Liu, and Peter Stone. Half Field Offense in RoboCup Soccer: A Multiagent Reinforcement Learning Case Study. *RoboCup 2006: Robot Soccer World Cup X*, pages 72–85, 2007.

[100] Dandan Zhang and Long Wang. Target Topology Based Task Assignment for Multiple Mobile Robots in Adversarial Environments. pages 5323–5328, 2007.

[101] Noa Agmon, Sarit Kraus, and Gal A. Kaminka. Multi-robot perimeter patrol in adversarial settings. In *Proceedings - IEEE International Conference on Robotics and Automation*, pages 2339–2345, 2008.

[102] Yaniv Shapira and Noa Agmon. Path planning for optimizing survivability of multi-robot formation in adversarial environments. *IEEE International Conference on Intelligent Robots and Systems*, 2015-Decem:4544–4549, 2015.

[103] Kao-Shing Hwang, Jin-Ling Lin, and Hui-Ling Huang. Cooperative patrol planning of multi-robot systems by a competitive auction system. *ICROS-SICE International Koint Conference 2009*, pages 4359–4363, 2009.

[104] Andrew Kwok and S Martínez. Deployment algorithms for a power constrained mobile sensor network. In *IEEE International Conference on Robotics and Automation, ICRA*, pages 140–145, 2008.

[105] Yongguo Mei, Yung Hsiang Lu, Y. Charlie Hu, and C. S George Lee. Deployment of mobile robots with energy and timing constraints. *IEEE Transactions on Robotics*, 22(3):507–522, 2006.

[106] Lei Zuo, Weisheng Yan, Rongxin Cui, Wei Chen, and Xiaoshan Bai. Coverage control of multiple ocean vehicles for environment monitoring with energy constraints. In *OCEANS 2014 - TAIPEI*, pages 1–6, 2014.

[107] Jason Derenick, Nathan Michael, and Vijay Kumar. Energy-aware coverage control with docking for robot teams. In *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3667–3672, 2011.

[108] Micha Rappaport. Energy-Aware Mobile Robot Exploration with Adaptive Decision Thresholds. In *47st International Symposium on Robotics (ISR)*, pages 1–8, 2016.

[109] Adam Seewald. Energy-aware coverage planning and scheduling for autonomous aerial robots, December 2021.

[110] Marco Pavone, Alessandro Arsie, Emilio Frazzoli, and Francesco Bullo. Equitable partitioning policies for robotic networks. In *IEEE International Conference on Robotics and Automation*, pages 2356–2361, 2009.

[111] A Pierson, L C Figueiredo, L C A Pimenta, and M Schwager. Adapting to performance variations in multi-robot coverage. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 415–420, 2015.

[112] Jean-samuel Marier, Camille-Alain Rabbath, and Nicolas Léchevin. Optimizing the Location of Sensors Subject to Health Degradation. In *American Control Conference*, pages 3760–3765, 2011.

[113] Jean Samuel Marier, Camille Alain Rabbath, and Nicolas Léchevin. Health-aware coverage control with application to a team of small UAVs. *IEEE Transactions on Control Systems Technology*, 21(5):1719–1730, 2013.

[114] James Kennedy, Airlie Chapman, and Peter M. Dower. Generalized coverage control for time-varying density functions. In *2019 18th European Control Conference (ECC)*, pages 71–76, 2019.

[115] Jorge Cortés, Sonia Martinez, Timur Karatas, and Francesco Bullo. Coverage control for mobile sensing networks: Variations on a theme. In *Mediterranean Conference on Control and Automation, MED*, 2002.

[116] Sung G. Lee, Yancy Diaz-Mercado, and Magnus Egerstedt. Multirobot Control Using Time-Varying Density Functions. *IEEE Transactions on Robotics*, 31(2):489–493, 2015.

[117] Sung G. Lee and Magnus Egerstedt. Controlled coverage using time-varying density functions. In *IFAC Workshop on Distributed Estimation and Control in Networked Systems*, pages 220–226, 2013.

[118] Luciano C.A. Pimenta, Mac Schwager, Quentin Lindsey, Vijay Kumar, Daniela Rus, Renato C. Mesquita, and Guilherme A.S. Pereira. Simultaneous Coverage and Tracking (SCAT) of moving targets with robot networks. *Springer Tracts in Advanced Robotics*, 57:85–99, 2010.

[119] D. Haumann, V. Willert, and K. D. Listmann. DisCoverage: From coverage to distributed multi-robot exploration. *IFAC Proceedings Volumes*, 4(Part 1):328–335, 2013.

[120] Farshid Abbasi, Afshin Mesbahi, and Javad Mohammadpour Velni. Coverage control of moving sensor networks with multiple regions of interest. In *American Control Conference*, pages 3587–3592, 2017.

[121] Walker Gosrich, Siddharth Mayya, Rebecca Li, James Paulos, Mark Yim, Alejandro Ribeiro, and Vijay Kumar. Coverage control in multi-robot systems via graph neural networks. *CoRR*, abs/2109.15278, 2021.

[122] Lin Li, Dianxi Shi, Songchang Jin, Ying Kang, Chao Xue, Xing Zhou, Hengzhu Liu, and XiaoXiao Yu. Complete coverage problem of multiple robots with different velocities. *International Journal of Advanced Robotic Systems*, 19(2):17298806221091685, 2022.

[123] Atsuyuki Okabe, Barry Boots, and Kokichi Sugihara. Nearest neighbourhood operations with generalized voronoi diagrams: a review. *International Journal of Geographical Information Systems*, 8(1):43–71, 1994.

[124] Zvi Drezner. *Facility Location: A Survey of Applications and Methods*. Springer Series in Operations Research and Financial Engineering. New York: Springer, 1995.

[125] Atsuyuki Okabe and Atsuo Suzuki. Locational optimization problems solved through Voronoi diagrams. *European Journal of Operational Research*, 98(3):445–456, 1997.

[126] Jorge Cortés, Sonia Martínez, Timur Karatas, and Francesco Bullo. Coverage Control for Mobile Sensing Networks. *IEEE Transactions on Robotics and Automation*, 20(2):243–255, 2004.

[127] Baoding Liu. *Facility Location Problem*, pages 157–165. Springer Berlin Heidelberg, Berlin, Heidelberg, 2009.

[128] Atsuyuki Okabe, Barry Boots, Kokichi Sugihara, Sung Nok Chiu, and D. G. Kendall. *Spatial Tessellations: Concepts and Applications of Voronoi Diagrams, Second Edition*. John Wiley and Sons Ltd., 2000.

[129] A. H. Stone and J. W. Tukey. Generalized "sandwich" theorems. *Duke Mathematical Journal*, 9(2):356 – 359, 1942.

[130] Theodore Hill. Determining a fair border. Technical report, American Mathematical Monthly, 1983.

[131] Atsuyuki Okabe, Barry Boots, Kokichi Sugihara, and Sung Nok Chiu. Spatial tessellations: concepts and applications of voronoi diagrams. 2009.

[132] Georges Voronoi. Nouvelles applications des paramètres continus à la théorie des formes quadratiques. premier mémoire. sur quelques propriétés des formes quadratiques positives parfaites. *Journal für die reine und angewandte Mathematik*, 133:97–178, 1908.

[133] Qiang Du, Vance Faber, and Max Gunzburger. Centroidal voronoi tessellations: Applications and algorithms. *SIAM Review*, 41(4):637–676, 1999.

[134] Miad Moarref and Hassan Sayyaadi. Facility location optimization via multi-agent robotic systems. In *IEEE International Conference on Networking, Sensing and Control, ICNSC*, pages 287–292, 2008.

[135] K. R. Guruprasad and Debasish Ghose. Coverage Optimization using Generalized Voronoi Partition. *Arxiv eprint arXiv:0908.3565*, 2011.

[136] Qihai Sun, Zhi-Wei Liu, Ming Chi, Ming-Feng Ge, and Dingxin He. Dynamic environment coverage control for heterogeneous robot networks. In *2022 IEEE International Conference on Unmanned Systems (ICUS)*, pages 13–18, 2022.

[137] F. Aurenhammer and H. Edelsbrunner. An optimal algorithm for constructing the weighted voronoi diagram in the plane. *Pattern Recognition*, 17(2):251–257, 1984.

[138] Günther Eder, Martin Held, Stefan de Lorenzo, and Peter Palfrader. On the recognition and reconstruction of weighted voronoi diagrams and bisector graphs. *Computational Geometry*, 109:101935, 2023.

[139] Soobum Kim, María Santos, Luis Guerrero-Bonilla, Anthony Yezzi, and Magnus Egerstedt. Coverage control of mobile robots with different maximum speeds for time-sensitive applications. *IEEE Robotics and Automation Letters*, 7(2):3001–3007, 2022.

[140] Hamid Mahboubi, Kaveh Moezzi, Amir G. Aghdam, Kamran Sayrafian-Pour, and Vladimir Marbukh. Self-deployment algorithms for coverage problem in a network of mobile sensors with unidentical sensing ranges. In *2010 IEEE Global Telecommunications Conference GLOBECOM 2010*, pages 1–6, 2010.

[141] Hamid Mahboubi, Jalal Habibi, Amir G. Aghdam, and Kamran Sayrafian-Pour. Distributed coverage optimization in a network of static and mobile sensors. In *2013 American Control Conference*, pages 6877–6881, 2013.

[142] Minghua Wang, Ran Ou, and Yan Wang. Multiplicatively weighted voronoi-based sensor collaborative redeployment in software-defined wireless sensor networks. *International Journal of Distributed Sensor Networks*, 18(3):15501477211069903, 2022.

[143] Omur Arslan and Daniel E. Koditschek. Voronoi-based coverage control of heterogeneous disk-shaped robots. In *IEEE International Conference on Robotics and Automation, ICRA*, pages 4259–4266, 2016.

[144] Yiannis Kantaros, Michalis Thanou, and Anthony Tzes. Distributed coverage control for concave areas by a heterogeneous Robot-Swarm with visibility sensing constraints. *Automatica*, 53:195–207, 2015.

[145] Jorge Cortés, Sonia Martinez, and Francesco Bullo. Spatially-distributed coverage optimization and control with limited-range interactions. *ESAIM: Control, Optimisation and Calculus of Variations*, 11(4):691–719, 2005.

[146] Xu Li, Zhengyan Liu, and Fuxiao Tan. Multi-robot task allocation based on cloud ant colony algorithm. In Derong Liu, Shengli Xie, Yuanqing Li, Dongbin Zhao, and El-Sayed M. El-Alfy, editors, *Neural Information Processing*, pages 3–10, Cham, 2017. Springer International Publishing.

[147] John Stergiopoulos and Anthony Tzes. Voronoi-based coverage optimization for mobile networks with limited sensing range - a directional search approach. In *2009 American Control Conference*, pages 2642–2647, 2009.

[148] K.R. Guruprasad and Prithviraj Dasgupta. Distributed voronoi partitioning for multi-robot systems with limited range sensors. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3546–3552, 2012.

[149] Lingxuan Kong, Qingchen Liu, and Changbin Brad Yu. Range-limited, distributed algorithms on higher-order voronoi partitions in multi-robot systems. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 6541–6546, 2019.

[150] Sotiris Papatheodorou, Anthony Tzes, and Yiannis Stergiopoulos. Collaborative visual area coverage. *CoRR*, abs/1612.0, 2016.

[151] Stuart Lloyd. Least squares quantization in pcm. *IEEE Transactions on Information Theory*, pages 129 – 137, 1982.

[152] Qiang Du, M. Emelianenko, and Lili Ju. Convergence of the lloyd algorithm for computing centroidal voronoi tessellations. *SIAM Journal on Numerical Analysis*, 44:102–119, 01 2006.

[153] Mac Schwager, James McLurkin, Jean-Jacques E. Slotine, and Daniela Rus. From theory to practice: Distributed coverage control experiments with groups of robots. In Oussama Khatib, Vijay Kumar, and George J. Pappas, editors, *Experimental Robotics*, pages 127–136, Berlin, Heidelberg, 2009. Springer Berlin Heidelberg.

[154] Andreas Breitenmoser, Mac Schwager, Jean Claude Metzger, Roland Siegwart, and Daniela Rus. Voronoi coverage of non-convex environments with a group of networked robots. In *IEEE International Conference on Robotics and Automation*, pages 4982–4989, 2010.

[155] María Santos, Siddharth Mayya, Gennaro Notomista, and Magnus Egerstedt. Decentralized minimum-energy coverage control for time-varying density functions. In *2019 International Symposium on Multi-Robot and Multi-Agent Systems (MRS)*, pages 155–161, 2019.

[156] Xiaotian Xu and Yancy Diaz-Mercado. Multi-agent control using coverage over time-varying domains. In *2020 American Control Conference (ACC)*, pages 2030–2035, 2020.

[157] Aydin Sipahioglu, Gokhan Kirlik, Osman Parlaktuna, and Ahmet Yazici. Energy constrained multi-robot sensor-based coverage path planning using capacitated arc routing approach. *Robotics and Autonomous Systems*, 58(5):529–538, 2010.

[158] Jong-Hyun Lee and Chang Wook Ahn. Improving Energy Efficiency in Cooperative Foraging Swarm Robots Using Behavioral Model. In *6th International Conference on Bio-Inspired Computing: Theories and Applications*, pages 39–44, 2011.

[159] Gennaro Notomista, Siddharth Mayya, Yousef Emam, Christopher Kroninger, Addison Bohannon, Seth Hutchinson, and Magnus Egerstedt. A resilient and energy-aware task allocation framework for heterogeneous multirobot systems. *IEEE Transactions on Robotics*, 38(1):159–179, 2022.

[160] Alyssa Pierson, Lucas C Figueiredo, Luciano CA Pimenta, and Mac Schwager. Adapting to sensing and actuation variations in multi-robot coverage. *The International Journal of Robotics Research*, 36(3):337–354, 2017.

[161] Nuttapon Boonpinon and Attawith Sudsang. Constrained coverage for heterogeneous multi-robot team. In *IEEE International Conference on Robotics and Biomimetics, (ROBIO)*, pages 799–804, 2007.

[162] María Santos, Yancy Diaz-Mercado, and Magnus Egerstedt. Coverage control for multirobot teams with heterogeneous sensing capabilities. *IEEE Robotics and Automation Letters*, 3(2):919–925, 2018.

[163] Yancy Diaz-Mercado, Sung G. Lee, and Magnus Egerstedt. *Human–Swarm Interactions via Coverage of Time-Varying Densities*, pages 357–385. Springer International Publishing, Cham, 2017.

[164] Sora Nishigaki, Kenta Hoshino, and Jun Yoneyama. Coverage control in moving regions with unmanned aerial vehicles. In *2019 IEEE/SICE International Symposium on System Integration (SII)*, pages 301–306, 2019.

[165] Gennaro Notomista and Magnus Egerstedt. Constraint-driven coordinated control of multi-robot systems. In *2019 American Control Conference (ACC)*, pages 1990–1996, 2019.

[166] Aaron D. Ames, Samuel Coogan, Magnus Egerstedt, Gennaro Notomista, Koushil Sreenath, and Paulo Tabuada. Control barrier functions: Theory and applications. In *2019 18th European Control Conference (ECC)*, pages 3420–3431, 2019.

[167] Aaron D. Ames, Xiangru Xu, Jessy W. Grizzle, and Paulo Tabuada. Control barrier function based quadratic programs for safety critical systems. *IEEE Transactions on Automatic Control*, 62(8):3861–3876, 2017.

[168] Xiangru Xu, Paulo Tabuada, Jessy W. Grizzle, and Aaron D. Ames. Robustness of control barrier functions for safety critical control. *IFAC-PapersOnLine*, 48(27):54–61, 2015.

[169] Wenceslao Shaw Cortez, Denny Oetomo, Chris Manzie, and Peter Choong. Control barrier functions for mechanical systems: Theory and application to robotic grasping. *IEEE Transactions on Control Systems Technology*, 29(2):530–545, 2021.

[170] Gennaro Notomista, Siddharth Mayya, Seth Hutchinson, and Magnus Egerstedt. An optimal task allocation strategy for heterogeneous multi-robot systems. In *2019 18th European Control Conference (ECC)*, pages 2071–2076, 2019.

[171] Mitio Nagumo. Über die lage der integralkurven gewöhnlicher differentialgleichungen. In *Proceedings of the Physico-Mathematical Society of Japan*, pages 551–559, 1942.

[172] Alberto Del Pia, Santanu S. Dey, and Marco Molinaro. Mixed-integer quadratic programming is in np, 2014.

[173] Efstathios Bakolas. Partitioning algorithms for homogeneous multi-vehicle systems with planar rigid body dynamics. In *53rd IEEE Conference on Decision and Control*, pages 5393–5398, 2014.

[174] Hamid Mahboubi, Kaveh Moezzi, Amir G. Aghdam, and Kamran Sayrafian-Pour. Distributed deployment algorithms for efficient coverage in a network of mobile sensors with nonidentical sensing capabilities. *IEEE Transactions on Vehicular Technology*, 63(8):3998–4016, 2014.

[175] Brian P Gerkey and Maja J Matari. Murdoch : Publish / Subscribe Task Allocation for Heterogeneous Agents. In *Proceedings of the fourth international conference on Autonomous agents, (ACM)*, pages 203–204, 2000.

[176] Luciano C. A. Pimenta, Vijay Kumar, Renato C. Mesquita, and Guilherme A. S. Pereira. Sensing and coverage for a network of heterogeneous robots. In *2008 47th IEEE Conference on Decision and Control*, pages 3947–3952, 2008.

[177] H. G. Tanner and D. K. Christodoulakis. Cooperation between aerial and ground vehicle groups for reconnaissance missions. In *Proceedings of the 45th IEEE Conference on Decision and Control*, pages 5918–5923, 2006.

[178] Herbert G. Tanner. Switched UAV-UGV cooperation scheme for target detection. In *Proceedings - IEEE International Conference on Robotics and Automation*, pages 3457–3462, 2007.

[179] Ben Grocholsky, James Keller, Vijay Kumar, and George Pappas. Cooperative air and ground surveillance. *IEEE Robotics and Automation Magazine*, 13(3):16–26, 2006.

[180] T. Arai, E. Pagello, and L.E. Parker. Guest editorial advances in multirobot systems. *IEEE Transactions on Robotics and Automation*, 18(5):655–661, 2002.

[181] Kaivan Kamali, Dan Ventura, Amulya Garga, and Soundar R T Kumara. Geometric task decomposition in a multi-agent environment. *Applied Artificial Intelligence*, 20(5):437–456, 2006.

[182] Jianping Chen, Yimin Yang, and Liang Wei. Research on the approach of task decomposition in soccer robot system. *Proceedings - 2010 International Conference on Digital Manufacturing and Automation, ICDMA 2010*, 2:284–289, 2010.

[183] Vitālijs Komašilovs and Egils Stalidzāns. Functional decomposition method reveals the number of possible specifications of multi-robot system. In *2011 IEEE 12th International Symposium on Computational Intelligence and Informatics (CINTI)*, pages 161–165, 2011.

[184] Hiroshi Kawano. Hierarchical Sub - task Decomposition for Reinforcement Learning of Multi - robot Delivery Mission. *International Conference on Robotics and Automation (ICRA)*, 2013.

[185] Yu Zhang and Lynne E Parker. IQ-ASyMTRe : Synthesizing Coalition Formation and Execution for Tightly-Coupled Multirobot Tasks. In *IEEE International Conference on Intelligent Robots and Systems*, 2010.

[186] Isai Roman-Ballesteros and Carlos F. Pfeiffer. A framework for cooperative multi-robot surveillance tasks. In *Proceedings - Electronics, Robotics and Automotive Mechanics Conference, CERMA 2006*, volume 2, pages 163–168, 2006.

[187] Kristina Lerman, Chris Jones, Aram Galstyan, and Maja J Matarić. Analysis of dynamic task allocation in multi-robot systems. *The International Journal of Robotics Research*, 25(3):225–241, 2006.

[188] Fang Tang and Lynne E. Parker. A complete methodology for generating multi-robot task solutions using asymtre-d and market-based task allocation. In *Proceedings 2007 IEEE International Conference on Robotics and Automation*, pages 3351–3358, 2007.

[189] Brian P. Gerkey and Maja J. Matarić. A framework for studying multi-robot task allocation. In *Computer Science*, 2003.

[190] Juan Camilo Gamboa Higuera and Gregory Dudek. Fair subdivision of multi-robot tasks. In *2013 IEEE International Conference on Robotics and Automation*, pages 3014–3019, 2013.

[191] Panagiota Vatsolaki and Alexios Tsalpatouros. Ewos: A sealed-bid auction system design and implementation for electricity interonnector capacity allocation. In *IISA 2013*, pages 1–6, 2013.

[192] D. Kim, Y. So, and S. Kim. Study of marker array list method for augmented reality service based smart home. 5:51–64, 01 2011.

[193] Changjoo Nam and Dylan A. Shell. Assignment algorithms for modeling resource contention and interference in multi-robot task-allocation. In *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2158–2163, 2014.

[194] Tolga Bektas. The multiple traveling salesman problem: an overview of formulations and solution procedures. *Omega*, 34(3):209–219, 2006.

[195] Omar Cheikhrouhou and Ines Khoufi. A comprehensive survey on the multiple traveling salesman problem: Applications, approaches and taxonomy. *Computer Science Review*, 40:100369, 2021.

[196] Valerio Lattarulo and Geoffrey T. Parks. A preliminary study of a new multi-objective optimization algorithm. In *2012 IEEE Congress on Evolutionary Computation*, pages 1–8, 2012.

[197] Lynne E Parker. L-ALLIANCE : Task-oriented multi-robot learning in behavior-based systems. *Advanced Robotics*, 11(4):305–322, 1998.

[198] Joel Esposito and James P Ostrowski. A Framework and Architecture for Multi-Robot Coordination. *The International Journal of Robotics Research*, 21(10-11):977–995, 2002.

[199] Anmin Zhu and Simon X. Yang. A framework for coordination and navigation of multi-robot systems. *2010 IEEE International Conference on Automation and Logistics, ICAL 2010*, pages 350–355, 2010.

[200] M. Tahir Khan, M. U. Qadir, F. Nasir, and C. W. De Silva. A framework for a fault tolerant multi-robot system. *10th International Conference on Computer Science and Education, ICCSE 2015*, (Iccse):197–201, 2015.

[201] Alaa Khamis, Ahmed Elmogy, and Fakhri Karray. Complex task allocation in mobile surveillance systems. *Journal of Intelligent and Robotic Systems*, 64:33–55, 10 2011.

[202] Robert Zlot and Anthony Stentz. Market-based multirobot coordination for complex tasks. *I. J. Robotic Res.*, 25:73–101, 01 2006.

[203] Silvia C. Botelho and Rachid Alami. M+: A scheme for multi-robot cooperation through negotiated task allocation and achievement. In *IEEE International Conference on Robotics and Automation*, number May, pages 1234–1239, 1999.

[204] M. Bernardine Dias and Anthony Stentz. A Free Market Architecture for Distributed Control of a Multirobot System. *Carnegie Mellon University Journal*, 1 2000.

[205] M.B. Dias and A. Stentz. Opportunistic optimization for market-based multirobot control. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, volume 3, pages 2714–2720 vol.3, 2002.

[206] M. Bernardine Dias. *TraderBots: A New Paradigm for Robust and Efficient Multirobot Coordination in Dynamic Environments*. PhD thesis, Carnegie Mellon University, Pittsburgh, PA, January 2004.

[207] Brian Coltin and Manuela Veloso. Mobile robot task allocation in hybrid wireless sensor networks. In *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2932–2937, 2010.

[208] Ahmed Hussein and Alaa Khamis. Market-based approach to multi-robot task allocation. In *2013 International Conference on Individual and Collective Behaviors in Robotics (ICBR)*, pages 69–74, 2013.

[209] J. K. Lenstra and A. H. G Rinnooy Kan. Computational complexity of discrete optimization problems. In P.L. Hammer, E.L. Johnson, and B.H. Korte, editors, *Discrete Optimization I*, volume 4 of *Annals of Discrete Mathematics*, pages 121–140. Elsevier, 1979.

[210] J. G. Martin, J. R. D. Frejo, R. A. García, and E. F. Camacho. Multi-robot task allocation problem with multiple nonlinear criteria using branch and bound and genetic algorithms. *Intelligent Service Robotics*, 14(5):707–727, Nov 2021.

[211] Marjorie Darrah, William Niland, and B. Stolarik. Multiple uav dynamic task allocation using mixed integer linear programming in a sead mission. *Infotech@Aerospace*, 09 2005.

[212] Nuzhet Atay and Burchan Bayazit. Mixed-integer linear programming solution to multi-robot task allocation problem. *Computer Science and Engineering*, 01 2006.

[213] Xun Li and Ma Hong-xu. Particle swarm optimization based multi-robot task allocation using wireless sensor network. *2008 International Conference on Information and Automation*, pages 1300–1303, 2008.

[214] Ayari Asma and Bouamama Sadok. Pso-based dynamic distributed algorithm for automatic task clustering in a robotic swarm. *Procedia Computer Science*, 159:1103–1112, 2019. Knowledge-Based and Intelligent Information and Engineering Systems: Proceedings of the 23rd International Conference KES2019.

[215] Alireza Tamaddoni-Nezhad and Stephen Muggleton. A genetic algorithms approach to ILP. In *Inductive Logic Programming*, pages 285–300, 2002.

[216] Milena Bogdanović. An ILP formulation and genetic algorithm for the maximum degree-bounded connected subgraph problem. *Computers and Mathematics with Applications*, 59(9):3029–3038, 2010.

[217] Ding Yingying, He Yan, and Jiang Jingping. Multi-robot cooperation method based on the ant algorithm. In *Proceedings of the 2003 IEEE Swarm Intelligence Symposium. SIS'03 (Cat. No.03EX706)*, pages 14–18, 2003.

[218] Andreas Ernst, Houyuan Jiang, and Mohan Krishnamoorthy. Exact solutions to task allocation problems. *Management Science*, 52:1634–1646, 10 2006.

[219] L. Puente-Maury, P. Mejía-Alvarez, and L. E. Leyva-del Foyo. A binary integer linear programming-based approach for solving the allocation problem in multiprocessor partitioned scheduling. In *8th International Conference on Electrical Engineering, Computing Science and Automatic Control*, pages 1–6, 2011.

[220] Essolizam Planté, Mylène Delhommais, Eric Bideaux, and Mathias Gérard. Offline power management strategy for fuel cell electric race carsizing using bi-level optimization based methodology. In *9th Internation Conference on Control, Decision and Information Technologies*, 2023.

[221] Martello, Silvano, and Paolo Toth. *Knapsack problems: algorithms and computer implementations*. John Wiley & Sons, Inc., 1990.

[222] Gian Carlo and Senthil Kumar. Dynamic Task Allocation for Mobile Robot Teams based on Linear Integer Programming. In *International Journal of Engineering Research and Technology (IJERT)*, 01 2021.

[223] Xin Ye, Wei Shen, Ilshat Mamaev, Thomas Bertram, Maximilian Bryg, Manuel Schwartz, Soeren Hohmann, Tamim Asfour, Bjoern Hein, Martin Kipfmueller, and Jan Kotschenreuther. Multi-level optimization approach for multi-robot manufacturing systems. In *ISR Europe 2022; 54th International Symposium on Robotics*, pages 1–8, 2022.

[224] Matthew Crosby, Anders Jonsson, and Michael Rovatsos. A single-agent approach to multiagent planning. In *Proceedings of the 21st European Conference on Artificial Intelligence (ECAI 2014)*, pages 237–242, 2014.

[225] Martina Lippi and Alessandro Marino. A mixed-integer linear programming formulation for human multi-robot task allocation. In *2021 30th IEEE International Conference on Robot and Human Interactive Communication*, pages 1017–1023, 2021.

[226] Meng-Tse Lee, Bo-Yu Chen, and Ying-Chih Lai. A hybrid tabu search and 2-opt path programming for mission route planning of multiple robots under range limitations. *Electronics*, 9(3), 2020.

[227] Hamza Chakraa, Edouard Leclercq, François Guérin, and Dimitri Lefebvre. A multi-robot mission planner by means of beam search approachand 2-opt local search. In *9th Internationa Conference on Control, Decision and Information Technologies*, 2023.

[228] Marwa Gam, Achraf Jabeur Telmoudi, and Dimitri Lefebvre. Hybrid filtered beam search algorithm for the optimization of monitoring patrols. *Journal of Intelligent and Robotic Systems*, 2023.

[229] G. Korsah, Anthony Stentz, and M. Dias. A comprehensive taxonomy for multi-robot task allocation. *International Journal of Robotics Research*, 32:1495–1512, 10 2013.

[230] Rubens J.M. Afonso, Marcos R.O.A. Maximo, and Roberto K.H. Galvão. Task allocation and trajectory planning for multiple agents in the presence of obstacle and connectivity constraints with mixed-integer linear programming. *International Journal of Robust and Nonlinear Control*, 30(14):5464–5491, 2020.

[231] Gennaro Notomista, Siddharth Mayya, Mario Selvaggio, Maria Santos, and Cristian Secchi. A set-theoretic approach to multi-task execution and prioritization. *CoRR*, abs/2003.02968, 2020.

[232] R. N. Duca and M. K. Bugeja. A multi-robot allocation scheme for coverage control applications with multiple areas of interest. In *9th International Conference on Control, Decision and Information Technologies*, 2023.

[233] Rachael N. Duca and Marvin K. Bugeja. A task allocation scheme for heterogeneous robot teams in coverage control applications. *IEEE Access*, 2023. Manuscript submitted for publication.

[234] Punyisa Kuendee and Udom Janjarassuk. A comparative study of mixed-integer linear programming and genetic algorithms for solving binary problems. In *2018 5th International Conference on Industrial Engineering and Applications (ICIEA)*, pages 284–288, 2018.

[235] Mathworks. repmat Documentation. `https://www.mathworks.com/help/matlab/ref/repmat.html`. Accessed: 09/08/2023.

[236] Mathworks. ones Documentation. `https://www.mathworks.com/help/matlab/ref/ones.html?searchHighlight=ones&s_tid=srchtitle_support_results_1_ones`. Accessed: 09/08/2023.

[237] Mathworks. blkdiag Documentation. `https://www.mathworks.com/help/matlab/ref/blkdiag.html?searchHighlight=blkdiag&s_tid=srchtitle_support_results_1_blkdiag`. Accessed: 09/08/2023.

[238] Mathworks. intlinprog Documentation. `https://www.mathworks.com/help/optim/ug/intlinprog.html?searchHighlight=intlinprog&s_tid=srchtitle_support_results_1_intlinprog`. Accessed: 09/08/2023.

[239] C.H. Papadimitriou and M. Yannakakis. The complexity of facets (and some facets of complexity). *Journal of Computer and System Sciences*, 28(2):244–259, 1984.

[240] David R. Morrison, Sheldon H. Jacobson, Jason J. Sauppe, and Edward C. Sewell. Branch-and-bound algorithms: A survey of recent advances in searching, branching, and pruning. *Discrete Optimization*, 19:79–102, 2016.

[241] Xavier Meyer, Bastien Chopard, and Paul Albuquerque. A branch-and-bound algorithm using multiple gpu-based lp solvers. In *20th Annual International Conference on High Performance Computing*, pages 129–138, 2013.

[242] Hsiao-Dong Chiang and Tao Wang. A novel trust-tech guided branch-and-bound method for nonlinear integer programming. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 45(10):1361–1372, 2015.

[243] Mahmoud M. Ismail, Waiel F. Abd El-Wahed, and Osama Abd El-Raoof. A parallel branch and bound algorithm for solving large scale integer multiobjective problems. In *2010 The 7th International Conference on Informatics and Systems (INFOS)*, pages 1–7, 2010.

[244] Mathworks. Branch and Bound algorithm in intlinprog Documentation. `https://www.mathworks.com/help/matlab/ref/profile.html`. Accessed: 14/08/2023.

[245] Mathworks. MATLAB profile Documentation. `https://www.mathworks.com/help/matlab/ref/profile.html`. Accessed: 11/08/2023.

[246] Michele Conforti, Gérard Cornuéjols, and Giacomo Zambelli. *Integer Programming*. Graduate Texts in Mathematics. Springer Cham, 2014.

[247] Harvey J. Everett. Generalized Lagrange Multiplier method for solving problems of optimum allocation of resources. *Operations Research*, 11:399–417, 1963.

[248] J. N. Hooker. *Integer programming: Lagrangian Relaxation Integer Programming: Lagrangian Relaxation*, pages 1667–1673. Springer US, Boston, MA, 2009.

[249] Rachael N. Duca and Marvin K. Bugeja. Multi-robot energy-aware coverage control in the presence of time-varying importance regions. *IFAC-PapersOnLine*, 53(2):9676–9681, 2020. 21st IFAC World Congress.

[250] Jose-Marcio Luna, Rafael Fierro, Chaouki Abdallah, and John Wood. An adaptive coverage control algorithm for deployment of nonholonomic mobile sensors. In *49th IEEE Conference on Decision and Control (CDC)*, pages 1250–1256, 2010.

[251] Mert Turanli and Hakan Temeltas. Power-aware adaptive coverage control with consensus protocol. *Journal of Automation and Control Engineering*, 4(6):413–418, 2016.

[252] Arezoo Ansari and Farzaneh Abdollahi. Coverage control for a group of uavs and ugvs with the effect of uavs altitude. In *2021 9th RSI International Conference on Robotics and Mechatronics (ICRoM)*, pages 112–119, 2021.

[253] Rachael N. Darmanin and Marvin K. Bugeja. Autonomous Exploration and Mapping using a Mobile Robot Running ROS. In *13th International Conference on Informatics in Control, Automation and Robotics*, 2016.

[254] Juan Angel Gonzalez-Aguirre, Ricardo Osorio-Oliveros, Karen L. Rodríguez-Hernández, Javier Lizárraga-Iturralde, Rubén Morales Menendez, Ricardo A. Ramírez-Mendoza, Mauricio Adolfo Ramírez-Moreno, and Jorge de Jesús Lozoya-Santos. Service robots: Trends and technology. *Applied Sciences*, 11(22), 2021.

[255] Chin-Sheng Chen, Feng-Chieh Lin, and Chia-Jen Lin. The energy efficiency multi-robot system and disinfection service robot development in large-scale complex environment. *Sensors*, 23(12), 2023.