THREE-DIMENSIONAL COVERAGE CONTROL FOR MULTI-UNMANNED AERIAL VEHICLE

SIOW WEN HAO

UNIVERSITI TUNKU ABDUL RAHMAN

THREE-DIMENSIONAL COVERAGE CONTROL FOR MULTI-UNMANNED AERIAL VEHICLE

SIOW WEN HAO

A project report submitted in partial fulfilment of the requirements for the award of Bachelor of Mechatronics Engineering with Honours

Lee Kong Chian Faculty of Engineering and Science
Universiti Tunku Abdul Rahman

May 2024

DECLARATION

I hereby declare that this project report is based on my original work except for citations and quotations which have been duly acknowledged. I also declare that it has not been previously and concurrently submitted for any other degree or award at UTAR or other institutions.

Signature : Siow

Name : Siow Wen Hao

ID No. : 2003860

Date : 15/05/2024

APPROVAL FOR SUBMISSION

I certify that this project report entitled **"THREE-DIMENSIONAL COVERAGE CONTROL FOR MULTI-UNMANNED AERIAL VEHICLE"** was prepared by **SIOW WEN HAO** has met the required standard for submission in partial fulfilment of the requirements for the award of Bachelor of Mechatronics Engineering with Honours at Universiti Tunku Abdul Rahman.

Approved by,

Signature	:	
Supervisor	:	Dr. Shalini A/P Darmaraju
Date	:	16 May 2024

Signature	:	
Co-Supervisor	:	Dr. Kwan Ban Hoe
Date	:	16 May 2024

ACKNOWLEDGEMENTS

I would like to thank everyone who had contributed to the successful completion of this project. I would like to express my gratitude to my research supervisor, Dr. Shalini A/P Darmaraju and Dr. Kwan Ban Hoe for their invaluable advice, guidance and their enormous patience throughout the development of the research.

In addition, I would also like to express my gratitude to my loving parents and friends who had helped and given me encouragement and unwavering support throughout the journey of conducting this study.

Without the support from Dr. Shalini, Dr. Kwan, parents, and friends, this journey would have been far more daunting. Their encouragement has fueled my determination and strength to oversom obstacles and complete the research.

ABSTRACT

Multi-Agent Systems (MAS) facilitate complex systems by enabling self-governing agents to collaborate towards common goals. Within MAS, coverage control plays a vital role in optimizing agent deployment for area coverage. Hence, this research addresses the development and evaluation of a Three-Dimensional (3D) coverage control algorithm for Multi Unmanned Aerial Vehicle (MUAV). This project aims to model and simulate a group of quadrotors for area coverage in 3D space with the objectives of reviewing existing MUAV coverage methods, algorithm development, and assessing computational load, convergence, path length, coverage quality, and scalability in both Two-Dimensional (2D) and 3D spaces. The study begins with a literature review categorizing coverage problems into barrier, blanket, and sweeping coverage, alongside different coverage control strategies such as centralized, decentralized, and hybrid approaches. The transition from 2D to 3D algorithms is highlighted as a contemporary trend, as 2D coverage algorithms are sufficient for simple coverage tasks but lack the ability to handle complex coverage tasks. The Multi-step Broadcast Control (MBC) scheme emerges as a key reference due to its effectiveness in scalability and computational efficiency. This study proposes a 3D coverage control algorithm to address complex real-world scenarios, wherein the 3D control algorithm will be developed based on the 2D MBC scheme. The performance of the transitioned 3D algorithm is evaluated against the original 2D MBC algorithm in terms of computational load, convergence analysis, path length, coverage quality, and scalability. Results indicate that while the 3D algorithm exhibits higher computational load, it surpasses the 2D algorithm in convergence analysis and path length, maintaining consistent coverage quality and scalability. Objectives of this project are achieved by reviewing existing coverage control methods, developing, and evaluating the proposed 3D coverage control algorithm, demonstrating its effectiveness, and potential applicability in real-life scenarios.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF SYMBOLS / ABBREVIATIONS

MAS	Multi Agent System
UGVs	Unmanned Ground Vehicles
UUVs	Unmanned Underwater Vehicles
UAVs	Unmanned Aerial Vehicles
USVs	Unmanned Surface Vehicles
2D	Two-dimensional
3D	Three-dimensional
MUAV	Multi Unmanned Aerial Vehicle
UAV	Unmanned Aerial Vehicle
MARL	Multi Agent Reinforcement Learning
CPP	Coverage Path Planning
BC	Broadcast Control
PBC	Pseudo-perturbation-based Broadcast Control
MBC	Multi-step Broadcast Control
BRKGA	Biased Random Key Genetic Algorithm
ROS	Robotics Operating System

LIST OF APPENDICES

CHAPTER 1

INTRODUCTION

1.1 General Introduction

Multi-Agent Systems (MAS) offer a robust framework for designing and modelling complex systems, where multiple self-governing agents interact and cooperate to achieve mutual objectives (Wooldridge, 2011). Within a MAS, each agent possesses unique knowledge, skills, and decision-making capabilities, empowering them to function both independently and collaboratively to resolve challenges and navigate in dynamic environments.

Agents within a MAS can display a wide range of behaviours, communication patterns, and strategies, enabling them to adapt and interact with other agents in a flexible and dynamic manner. This adaptability renders MAS particularly suitable for real-world applications, where uncertainties, complexities, and the necessity for cooperation and coordination are prevalent. As MAS research advances, it has led to the development of sophisticated algorithms, negotiation protocols, and decision-making mechanisms, fostering agent collaboration, information sharing, and collective problem-solving. Additionally, the study of MAS provides valuable insights into the emergent behaviours and system-level phenomena that arise from the interactions of individual agents, illuminating complex systems that may be challenging to comprehend using traditional reductionist methods.

The versatility and efficacy of MAS have made it a favoured approach for addressing a wide spectrum of problems in diverse domains, including robotics, artificial intelligence, economics, transportation, and social sciences (Jennings, 2000). By capitalizing on distributed intelligence and coordination among agents, MAS is equipped to tackle tasks that may surpass the capabilities of individual agents or traditional centralized systems in term of efficiency, fault tolerance, scalability, enhanced problem solving, and flexibility as well as adaptability. In MAS, agents can separate tasks and work in parallel, leading to higher efficiency with shorter execution time since the system can settle numerous objectives simultaneously. Implementation of MAS can improve the robustness of unmanned vehicles by ensuring smooth operation flow, where the other agents will continue the function or replace the agents that encounter issues or fails. Besides, scalability of MAS allows the overall system to scale up to fit with the increase demand due to the complexity and enhanced problem solving, as increasing number of agents allow the system to explore different solutions, resulting in near optimal results. Multiple agents can collaborate and coordinate their actions in MAS, such as combining and coordinating agents with different abilities, allowing the system to handle complex tasks. Decentralization is discovered with the exploration of MAS, where multiple agents are in decentralized control, greatly reduce the computational burden of global central controller (Jiménez et al., 2018).

Usually, MAS will be implemented in unmanned vehicles. There are several types of unmanned vehicles, and each is designed for specific tasks and applications. The type of unmanned vehicle mainly can be categories as Unmanned Ground Vehicles (UGVs), Unmanned Underwater Vehicles (UUVs), Unmanned Aerial Vehicles (UAVs), and Unmanned Surface Vehicles (USVs).

Coverage control is a fundamental concept in robotics and autonomous systems, aiming to optimize the deployment and movement of agents to achieve efficient and comprehensive coverage of a given area or region (Cortés et al., 2004). The objective of coverage control is to control an agent or multiple agents to ensure the entire target space is covered to perform exploration, monitoring, and network coverage supply tasks in a coordinated manner. Coverage controls are widely used nowadays in various domains such as unmanned vehicles, environmental and disaster monitoring, robotics, agriculture, and surveillance.

The research and development of coverage control began in the early 2000s (Tan and Zheng, 2013). It has undergone significant advancements lately and become a pivotal area of research in robotics and autonomous systems. The challenges of the research fall in coverage control falls in various factors such as complexity of the target area, capabilities and mobility of agents, real-time adaptation to dynamic environmental conditions, as well as the presence of static and dynamic obstacles. To this day, fellow researchers are still trying their best to developed sophisticated algorithms to improve its optimization and coordination abilities to achieve comprehensive coverage efficiently.

1.2 Importance of the Study

In this era of industry revolution 4.0, the relentless advancement of technology has led to remarkable breakthroughs in various industries, especially in automation and robotics. As a result of the significant advances made in automation and robotics, Unmanned Aerial Vehicles (UAVs) can accomplish a variety of tasks in a more effective, efficient, and secure manner (Jayanthi et al., 2023). Hence, research on the control of UAVs is widely done by the researcher and is still ongoing due to its profound impact on various applications that required data gathering, surveillance, and monitoring. Not only that, fellow researchers also further improve the control strategy by developing algorithms that allow controllers to autonomously control multiple UAVs.

Coverage control is a control strategy and coordination mechanism for directing single or multiple agents to cover a given space with the optimum performance, ensuring the space or area of interest is fully under covered by the agents (muro, n.d.). The coverage control can be applied to either Two-dimensional (2D) or Three-dimensional (3D) regions in space. This technique can be implemented in various field such as environmental monitoring, agriculture, disaster response, infrastructure inspection, and more. Multi Unmanned Aerial Vehicle (MUAV) equipped with sensors can efficiently perform task such as surveillance, data gathering, and monitoring.

As mentioned earlier, MUAV can efficiently perform a variety of tasks, but without a proper and robust control algorithm, MAS cannot optimize coordination among agents to perform required tasks. Thus, a 2D algorithm might not be as robust as a 3D algorithm for MAS in most coverage tasks. A 3D coverage control enables MUAV that are equipped with multiple sensors to capture data with varying altitudes and providing a more holistic view of ecological control. In a search and rescue operations, a 3D algorithm allows MUAV to search through complex terrains, hazardous environments, assess the level of destruction due to disaster as well as to locate the survivors. The ability to cover with different altitudes minimizes the risk of rescuers and improve the success probability while reducing response time. Moreover, a 3D algorithm can adjust the altitude of MUAV for data collection from multiple layers of environments, providing a comprehensive understanding of ecosystems that allows researchers to study the patterns of ecosystems. Furthermore, coverage control in 3D space provides the room for improvement, as the real-life situation is in 3D space. Hence, a 3D coverage control will comply with the situation tackled, whereas 2D coverage control is limited.

In short, developing a coverage control algorithm in 3D space enhances data collection and decision-making by providing deeper insights, improving operational efficiency, and enabling the tackling of complex tasks and environments.

1.3 Problem Statement

Multi Unmanned Aerial Vehicle (MUAV) face challenges during Two-dimensional (2D) static coverage tasks, as coverage control to achieve optimal path planning in a 2D environment can be complex. This involves efficient coverage patterns, kinematic constraints of the agents, and coordination among MUAV to avoid collisions.

The data processing of a MUAV system is important as it transforms raw sensor data collected by the Unmanned Aerial Vehicles (UAVs) into actionable information and insights. The process of data processing involves various steps such as filtering, fusion, analysis, and interpretation to extract valuable information from the collected data. Hence, real-time data processing impacts the output quality, while slow data processing will affect the robustness of the MUAV system in performing coverage tasks. Speed in decision-making is challenging as it can perform flight path adjustion to quickly avoid collisions and overcome the limitation in maneuverability of UAVs. Hence, it is important to ensure the algorithm developed will have a lower computational cost as well as higher convergence value in optimization as it will causes the system to have a slow reaction time and low position accuracy which will lead to inequality in coverage control.

Furthermore, an algorithm for coverage control that uses 2D as the environment boundary is not realistic, as the real world is in Three-dimensional (3D). Transitioning from a 2D algorithm to a 3D algorithm for multi-UAV coverage control is essential, as it could solve several challenges that are specific to 2D scenarios. For example, MUAV's view coverage is restricted to the horizontal plane in 2D space, limiting obstacle avoidance to the horizontal plane. Transitioning from 2D to 3D allows MUAV to navigate more efficiently, safely, and smoothly in cluttered environments with varying object altitudes.

Next, transitioning from 2D to 3D provides more comprehensive environmental sensing. 2D restricted the capabilities to detect certain environmental features or hazards such as complex structures, terrains, forests, buildings, and mountains. 3D environment provides adaptability to cover complex environment for coverage control and it allows the UAVs to have the ability to detect object with full freedom of movement instead of just object with a horizontal motion. Lastly, the transition from a 2D algorithm to a 3D one is crucial, as it will reduce the overlapping paths that cause redundant coverage. 3D environment provides platform for the MUAV to optimize their trajectories and altitudes to reduce overlap issue and improve coverage efficient. The algorithm cannot be said to be complete and precise if it is not model in 3D environment due to the distance travel will not be Euclidean distance which is the optimal distance for an agent to travel from initial spot to targeted spot. When the 2D coverage control algorithm is implemented in real world applications, the agents will have to depart with raising altitude to certain height then only starts to perform coverage task to travel to the assigned location. This against the objective for Multi Agent System (MAS) in coverage control which the agents will perform task in an optimal way. The route for agents to travel is not the shortest pathway. Modeling and developing the algorithm in 3D will solve the issues that arise, as 3D allows the agents to optimize their route travel in an optimal way, which is based on the Euclidean distance.

Moreover, a 2D algorithm cannot be as efficient as a 3D algorithm when implemented in a real-life application. If a 2D algorithm for MUAV coverage control is implemented in real life, the MUAV system will require all the agents to raise their altitude at the same time in the form of a horizontal plane. To raise the MUAV to a desired altitude, a large and empty open area are required for the departure of MUAV without any obstacle when rising altitude. This is due to the control limitation of 2D algorithm which does not have the ability to adjust the dimensions of 3 axis. Therefore, a 3D algorithm is important, as it provides better coverage control for the MUAV system, allowing the MUAV to depart from complex environments where obstacles exist.

Besides, there are many limitations to a 2D coverage control algorithm, as it limits the details of sensor capabilities and the ability to deal with complex and challenging environments. A 2D algorithm decreases the adaptability of the MUAV system to changing conditions, whereas a 3D algorithm can be extended by adjusting the MUAV's altitude for better efficiency and stability. A 2D algorithm will have a higher deviation when implemented in a real-life application, as the model used for the 2D algorithm does not exactly replicate the circumstances of real-world operations. Dynamic tasks are limited in 2D, as the MUAVs are unable to adjust their altitude to perform tasks of varying importance in the environment. Lowering the altitude of UAVs allows for detailed inspection, while some other UAVs with higher altitude cover a broad area.

1.4 Aim and Objectives

Multi Unmanned Aerial Vehicle (MUAV) team are beneficial for number of tasks including providing coverage for monitoring and mapping. This project aims to model and simulate a group of Unmanned Aerial Vehicles (UAVs) to perform an area coverage task with smooth motion planning to overcome the kinematic constraints of UAVs under 3D space. The UAVs should intelligently allocate their efforts evenly while maintaining collision-free trajectories. The simulation should demonstrate the UAVs’ team coordination in a preset 3D environment, smooth motion of the MUAV while moving to the allocated spot. To review MUAV coverage control methods in Two-dimensional (2D) space and Three-dimensional (3D) space.

- To review existing MUAV coverage control methods.
- To develop a 3D coverage control algorithm for static coverage task.

- To evaluate the performance of the algorithm in 2D and 3D space in terms of computational load, convergence analysis, path length, coverage quality, and scalability.

1.5 Scope and Limitation of the Study

To study the control algorithm to perform static coverage task for Multi Unmanned Aerial Vehicle (MUAV) in Three-dimensional (3D) environment is the scope of this study. The performance of the coverage control in both Two-dimensional (2D) and 3D will be evaluated based on the computation cost and the distance travelled by each agent. Since there will be missing dimension for 2D when compared to 3D, the distance travelled by each agent in 2D will include the rising altitude distance to ensure that the comparison is done fairly under the same circumstances.

Constructing a Multi Agent System (MAS) to conduct the experiment is very challenging; this project will mainly focus on the development of the coverage control algorithm and simulate the operation using MATLAB from a defined departure point to the assigned locations for each agent. Since this project will mainly focus on the development of coverage control algorithm for MUAV in 3D space, the MAS will be designed as a homogeneous system instead of a heterogeneous system where all the agents involved will be the same, which is same characteristics of MUAV with quadrotor as the type of vehicle. This is because a heterogeneous system involved multiple various type of unmanned mobile vehicles cooperating with each other, which will increase the complexity and difficulty of this project. Additionally, the simulation environment is assumed to be an empty open-air space where there will be no static or dynamic obstacles. The environment is a self-defined environment which it is assumed to be a known environment where the departure points and the target locations for the MUAV are defined before the operation of coverage starts. Since it is assumed to be an empty open-air space, obstacle avoidance for the MUAV will be ignored during the algorithm development phase. However, the developed coverage control algorithm shall have the ability to perform smooth operation of coverage task without collision between agents.

1.6 Contribution of the Study

This project outlines existing coverage control methods and briefly describes how they are categorized. The existing coverage control methods will be evaluated, and a control algorithm will be selected as the main reference for developing Three-dimensional (3D) coverage control algorithms. The aim of this project is to develop a 3D coverage control algorithm that maintains or exceeds the performance benchmarks set by the original algorithm. The developed 3D coverage control algorithm will serve as a framework for other researchers to implement additional features related to coverage in 3D, enriching the algorithm for deployment in real-life scenarios.

1.7 Outline of the Report

In this report, there will be 5 chapters of content in this final year project report. The focus of Chapter 1 will be the introduction of Multi Agent System (MAS) in coverage control. Chapter 2 will comprise the literature review, encompassing an examination of existing research to gain insights into the coverage problem, coverage control, and methods for performing coverage tasks. Chapter 3 will provide a detailed explanation of how the selected method achieves the project's objectives. Chapter 4 will focus on the performance evaluation of the developed Three-dimensional (3D) algorithm, with a brief discussion on its application. Chapter 5 will serve as the conclusion for the report, where future work that can be implemented in the project will also be discussed.

CHAPTER 2

LITERATURE REVIEW

2.1 Introduction

Multi Unmanned Aerial Vehicle (MUAV) have gained significant attention in recent years for their diverse applications, including surveillance, monitoring, search and rescue, agriculture, and environmental data collection. While a single Unmanned Aerial Vehicle (UAV) can undoubtedly offer valuable capabilities, it inevitably encounters limitations when tasked with complex missions and extensive coverage requirements. To address these constraints and enhance mission effectiveness, the deployment of MUAV becomes imperative. When considering the deployment of a substantial number of vehicles concurrently, it becomes evident that this arrangement forms a Multi Agent System (MAS). While the concept of leveraging numerous agents holds immense potential for various applications, it introduces numerous sets of significant challenges related to the efficient and cost-effective control of such a fleet.

The main problem when dealing with MAS is the coordination and control of multiple agents operating within the system simultaneously. It is an intricate task to control each individual agent in a well-coordinated manner, ensuring that the agent achieve the intended objectives while maintaining optimal resource allocation. The complexity of this task escalates as the number of agents within the MAS increases, handling more agents is harder than handling fewer. Scaling up in MAS does not equate to being able to handle more agents (Durfee, 2004). There is increment in complexity as the inherent trade-off between the number of agents deployed and the control overhead required for their effective operation. The scalability of the control algorithms and decision-making processes becomes an essential issue in the context of a large-scale MAS. Some developed control algorithms might not be suitable for large scale MAS as when the number of agents increases, the computation cost and efficiency will vary differently according to different algorithm. Besides, the communication within the MAS is important to ensure efficient operation. Establishment of a robust communication networks and real-time data exchange mechanism can greatly reduce the computational cost of a MAS.

A coverage control system for MAS places its emphasis on efficiently organizing the movement of multiple dynamic agents to ensure adequate sensor coverage within a specific bounded area(Kennedy et al., 2019). According to (Kennedy et al., 2019), the primary objective of the optimal coverage problem is to strategically position these agents in a manner that minimizes a designated coverage cost. Typically, this coverage cost is determined with reference to a density function, which is employed to steer the network towards desired arrangements. Cooperative in MAS can be classified into homogeneous and heterogeneous and the cooperative between agents is important as it shows the cumulative impact of combining 1 and 1 exceeds the value of 2.

In the world of MAS, homogeneous groups of agents represent agents that have the same actions available, and it does not care which agent will perform the action given, but only concern on how many agents will perform the action given (Pedersen and Dyrkolbotn, 2013).

In conjunction, heterogeneous groups of agents indicates that the system achieve its objectives in a way that the tasks are done collaboratively from different agents having different roles (Lee and Shim, 2022).

To simplify further, a MAS will be classified based on the agents' abilities. However, In the realm of Multi-Agent Reinforcement Learning (MARL) systems, it's important to note that MAS can exhibit both homogeneity and heterogeneity. According to (Fernandez et al., 2021), research delves into the consequences of employing both homogeneous and heterogeneous strategies within a MARL framework. In the context of MARL, the agents initially possess uniform capabilities, but they adapt to become heterogeneous. The results obtained from employing either uniform or diverse learning strategies are markedly distinct from those of agents adhering to fixed strategies, and these distinctions are precisely outlined through analytical definitions.

In the realm of coverage control, 2 distinct approaches have emerged, each with its own set of methodologies and characteristics. These 2 primary types of coverage control include Coverage Path Planning (CPP) and static coverage control. Static coverage control which is one of the fundamental strategies for optimizing the spatial coverage of autonomous systems, can be further categorized into 2 category which is centralized and decentralized coverage control. These classifications represent how the static coverage control strategy is executed and how decisions are made within the system.

2.2 Coverage Control Methods

Coverage control can be classified as Coverage Path Planning (CPP) and static coverage control. Static coverage control can be further classified into centralized coverage control and decentralized coverage control. The details of the control approaches used in the coverage task for different types of coverage control will be evaluated in this subsection.

2.2.1 Coverage Path Planning

Coverage Path Planning (CPP) aims to cover the total area of interest with minimum overlapping (Fevgas et al., 2022). Besides, (Galceran and Carreras, 2013) mention that CPP can be described as the task of finding a route that covers all specified points within a defined area or volume, all while navigating around obstacles effectively. This algorithm that drives the CPP plays a crucial role in numerous robotic applications, spanning a wide range of domains. These encompass a wide range of applications, including vacuum-cleaning robots, painting robots, autonomous underwater vehicles dedicated to creating image mosaics, demining robots, lawn mowing machines, automated harvesters, window-cleaning devices, and the inspection of complex structures, among others. The effectiveness of a CPP algorithm is typically assessed based on several key metrics, including the overall coverage ratio, the time required to complete the task, the total distance traveled, and the count of turns made during the path planning (Khan et al., 2017). In short, CPP produces a seamless and uninterrupted path that spans a designated area of interest, all while effectively steering clear of obstacles. (Galceran and Carreras, 2013).

CPP can be performed in multiple regions with multiple agents as well. A large-scale region may be presented in the form of disjoint sub-regions and its usually apply in cases like disaster management, numerous surveillance area and more (Kumar and Kumar, 2023).

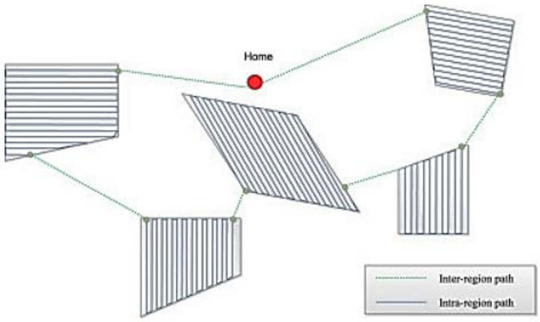

Figure 2.1: Multiple Region Coverage Task (Kumar and Kumar, 2023)

Grid-based CPP is a foundational approach in the field of robotics and autonomous systems. Grid-based methods simplify coverage in a defined area of interest by employing cellular decomposition by overlaying a grid structure onto the area, streamlining the coverage process. (Cabreira et al., 2019).

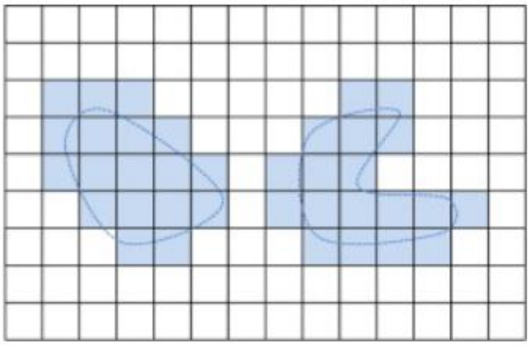

Figure 2.2: Grid that is Implemented on the Area of Interest (Galceran and Carreras, 2013).

The concept of utilizing a grid-based representation was first introduced by (Moravec and Elfes, 1985) for the purpose of mapping indoor environments with implementation of a sonar ring mounted on a mobile robot to fulfill the grid-based mapping approach. In this approach, each grid cell is assigned with a value that indicates the presence of obstacles or the absence of obstacles and the value representing whether the cell corresponds to occupied or free space. Typically, the value that are assigned to each grid are either probability or binary (Elfes, 1987). In most cases, the grid will be a square, yet it does not limit to square only, a polygon shape such as triangle and convex shapes can also be used which (Oh et al., 2004) introduced grid-based technique in a control algorithm which implement triangular boundary cells but with a higher resolution compared to rectangular boundary cells.

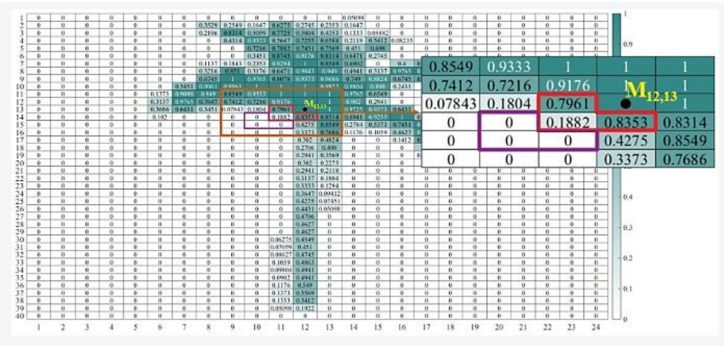

Figure 2.3: Implementation of Grid-Based Technique on a Tree (Ghaddar and Merei, 2020).

Figure 2.3 shows the example of implementation of grid-based technique on tree with probability value attached which the boundary cell with ‘0’ value indicates a free space, while the boundary cell with ‘1’ value indicates that boundary falls within the obstacle area. Grid-based method are usually ‘resolution complete’ which the completeness of the grid is greatly dependent on the resolution of the grid map and the grid will only provide an approximation of the target area and the obstacle within the area of interest. Hence, Grid-based method is classified as approximate cellular decompositions (Choset, 2001). The grid size and resolution can significantly affect the accuracy of path planning, but a fine grid might cause excessive computational load. When the complexity of the environment increases with constant grid resolution, grid-based tend to experience exponential growth in memory requirement (Thrun, 1998). Grid-based approach might face limitations in the flexibility when dealing with dynamic obstacles as it is an approximate approach. However, the simplicity and ease of implementation making it accessible for researchers as it’s advantageous for rapid prototyping and deployment.

The implementation of grid-based into CPP was first proposed by Zelinsky et al by using wavefront algorithm in grid-based CPP (Galceran and Carreras, 2013). According to (Zelinsky et al., 2007), start point and goal point for the grid map will be pre-determined and a distance will be labelled propagates a wavefront from goal point to start point which the goal will be marked as ‘0’, the surrounding cells will be marked as ‘1’ and the neighboring cells to ‘1’ will be marked as '2’. This process repeated from goal point to start point by the wavefront as shown in Figure 2.4.

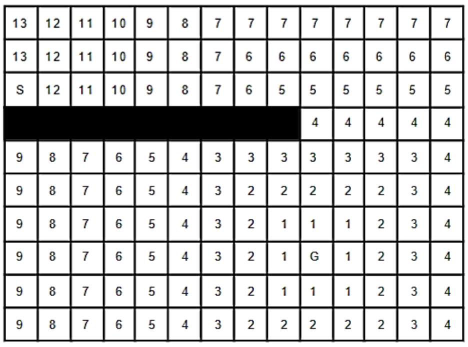

Figure 2.4: Wavefront Distance Transform from the Goal Point to Start Point (Zelinsky et al., 2007).

The path can be generated once the distance transform is done labelled. The path will be generated according to the value assigned to the cell by starting with the start point and selecting the neighboring cells that are labelled with the highest value that are unselected before. If there are 2 or more cells that are with the same value and unselected, the system will select the cell randomly which this type of CPP is similar to pseudo-gradient descent (Galceran and Carreras, 2013).

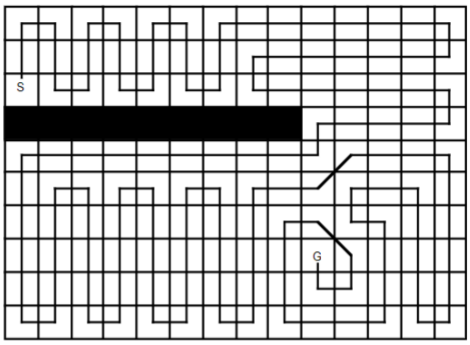

Figure 2.5: The coverage path form by selecting the highest value neighboring cells (Zelinsky et al., 2007).

Moreover, Spiral-Spanning Tree Coverage algorithm with grid-based approach was introduced by (Gabriely and Rimon, 2002). For Spiral-Spanning Tree Coverage algorithm, each grid cells are divided into 4 subsection which quarter of the cell size must be larger than the agent's size.

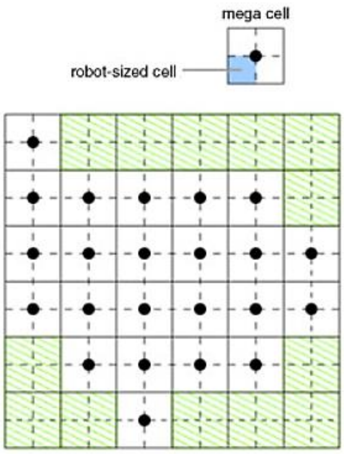

Figure 2.6: The Grid Cells are divided into 4 Subsections (Galceran and Carreras, 2013).

The coverage begins by starting from the starting point and select the new path from the small subsection of a grid cell according to anti-clockwise direction. From this point, the agents will never visit the subsection of the grid cell twice which minimize the coverage time.

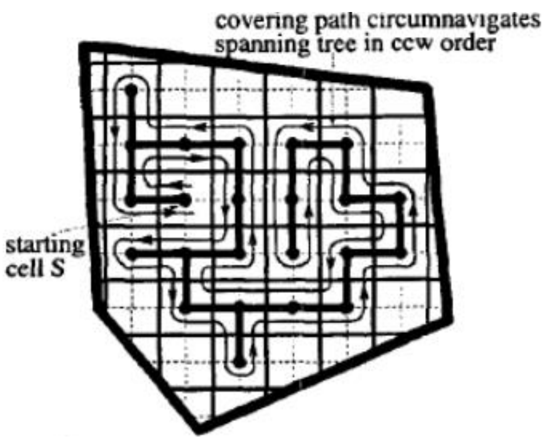

Figure 2.7: The Coverage Path is Formed according to Anti-clockwise Direction (Gabriely and Rimon, 2002).

Battery endurance, camera sensing, and more are the limitations of Unmanned Aerial Vehicle (UAV). Hence, due to the limitations of UAV, it is difficult to obtain efficient coverage using grid-based methods as there will be an increase in the computational time as the number of grids cells increase (Cho et al., 2021). As the area of interest widen, single UAV will find it struggle to cover the targeted area, so multiple UAV operating simultaneously is essential as it greatly reduces the completion time and efficiency (Barrientos et al., 2011; Modares et al., 2017; Li et al., 2018). Hence, the CPP that with multiple agents involved in the system can be defined as dynamic multi agent coverage which agents travel continuously throughout the environment so that every single point within the space is intermittently observed (Patel et al., 2020). The implementation of MAS into CPP is a very strategic control method that can perform the tasks efficiently with overcoming the constraints of UAV.

2.2.2 Static Coverage Control

Static Coverage Control aims to develop a control law that drive groups of agents from an initial position and distributed into another position such that each of the agents are fully cover the given domain (Atınç et al., 2020). In conjunction with the cooperative of Multi Agent System (MAS) in Coverage Path Planning (CPP), MAS can also be implemented into static coverage control and the combination of MAS and static coverage control is named static multi agent coverage (Patel et al., 2020). The simplest way in differentiate the difference between the MAS in CPP and Static coverage control is the use in different applications. For example, static coverage is widely used for perimeter monitoring that required to monitor every single point of the given space and perform it continuously respect to time (Gupta et al., 2019). The difference between CPP and static coverage is CPP requires every point in the environment to be swept as frequently as possible according to the resources allocated (Wong et al., 2002).

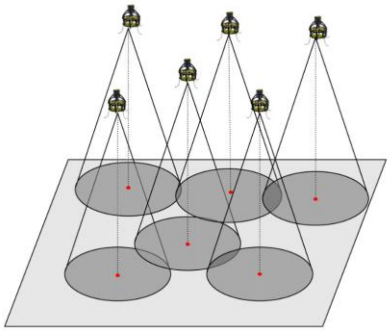

Figure 2.8: Example of Static Coverage Control (Wang and Li, 2013).

Figure 2.8 shows that the agents are deployed to a location and perform static coverage by maintaining the position continuously to perform the assigned task which is to monitor the area of interest. The characteristics of the static coverage are the agents' positions are fixed and the adaptability is limited. This type of coverage is favorable for scenario that the coverage requirements are relatively stable, yet the coverage algorithms are developed to the extent that the limitation of static coverage can be minimized which collision avoidance, obstacle avoidance, and changing of environmental are taken into account by the researchers.

2.2.2.1 Voronoi Based Voverage Control

The concept of Voronoi partition, also known as the Voronoi diagram or Voronoi Tessellation, has its roots dating back to as early as 1644, when philosopher Rene Descartes contemplated it. However, it is commonly associated with the name of the Russian mathematician George Voronoi, who is credited for its development and formalization. (University of Bristol, n.d.). Voronoi partition is the action that forming Voronoi diagram, the diagram can be formed by scattering initial points randomly on a Euclidean plane. After the points are defined, the plane is then divided into cells which known as tessellating polygons and these cells encompass the areas of the plane that are closer to that point than to any other (University of Bristol, n.d.). There are 3 characteristics for a Voronoi partition which is Voronoi cell, Voronoi edge, and Voronoi vertex. Voronoi cell refers to the point that is associated with a Voronoi cell which the region of space closest to that seed point while a Voronoi edge is the edge of Voronoi cell which segments of lines or curve lie exactly equidistant between the neighboring points. Voronoi vertex refer to the point where three or more Voronoi edges intersect, and it is essential when forming a Voronoi diagram structure.

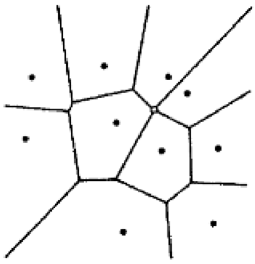

Figure 2.9: Planar Ordinary Voronoi Diagram (Okabe et al., n.d.).

Voronoi partition has been widely used by the researchers in developing the coverage control algorithm. According to (Hayashi et al., 2015), the event-triggered control techniques for addressing the centroidal Voronoi coverage problem is explores. In this technique, individual agents will update their control inputs when discrepancy happen between their current state and the centroid of their Voronoi cell, compared to the state at the last triggering event, surpasses a predefined threshold (Hayashi et al., 2015). Through this technique, centroidal Voronoi coverage can be achieve and the agent's triggering interval will remain positive with the constraint of the velocity of the centroid of its Voronoi cell is adequately low. The usage of data processing will reduce due to agents can ignore uninterrupted update of the input control.

In the early stage of using Voronoi partition, Voronoi partition are typically used as a centralized coverage control which is a control strategy that achieve coverage task by multiple agents under supervision of a central coordinator or global controller. The coordinator is responsible for making decisions and providing instructions to the agents which will helps in minimizing the coverage objective function. The coordinator will receive information from each of the agents such as positions and state of the agents as well as environmental data. Data collection and data cleaning will be done in global controller and the decision made will be sent to every individual agent with one-to-one communication. Basically, centralized coverage control will be a one-to-one communication which it will causes a limitation where there will be a high demand for the data usage. Hence, decentralized coverage control or also known as distributed coverage control is introduced. The difference between both is the centralized coverage control communicate between coordinator and agents while decentralized communicate between agents. In decentralized coverage control, the agents in Multi Agent System (MAS) will spread out over an environment and aggregated in the areas of interest as a result which is the same as what centralized coverage control did. However, the agents do not know beforehand the information of environment and they learn the information by communicating between themselves instead of communication between agent and controller (Schwager et al., 2009). The traditional Voronoi partition is modified through merging with an adaptive decentralized coverage control such that the data gathered by sensor can be learnt and shared among agents (Schwager et al., 2009). Voronoi partition with distributed coverage is justified as a convex region is covered by Multi Unmanned Aerial Vehicle (MUAV) (Chen et al., 2017). According to (Chen et al., 2017), the algorithm repeatedly updates and refines the Voronoi partition by exchanging local information with neighboring elements and subsequently adjusts its movement based on the calculated direction.

Lloyd's algorithm often paired with Voronoi partition by researchers as Lloyd's algorithm is an iterative technique used to improve the quality and regularity of a Voronoi tessellation. This combination is verified by (Bhattacharya et al., 2013) as an improved control law based on Voronoi and Llyod are generated and implemented to a non-Euclidean metric space with non-polygonal obstacles. Besides, detailed proof is provided to support the convergence of the Voronoi and Llyod control law (Bhattacharya et al., 2014). However, the results shown does not clearly mention the possibility in extending the control law to a generalized Three-dimensional (3D) map. With the short sensing capabilities of MUAV, the MUAV might not be able to detect the whole given environment with the given number of Unmanned Aerial Vehicle (UAV) resulting some loss of information due to not entire information of the environment is gathered. Hence, a modified Voronoi partition is proposed to enhance the coverage performance by making it dynamic environment which the agents will move gradually according to the motion of area of interest (Li and Liu, 2017).

For Voronoi partition, the agents might not be distributed evenly even though the cell of Voronoi diagram is evenly distributed. This problem is due to the seed points in every cell of the Voronoi diagram is not located in the center of each cell. Thus, further modification on the Voronoi partition is presented by (Du et al., 2006). The Voronoi tessellations are modified into centroidal Voronoi tessellations which is a bounded geometric domain such that the seed points of the Voronoi partition are also the centers of the cells with respect to a given density function (Du et al., 2006). The results show that the centroidal Voronoi tessellations guarantee the seed point is in the center of every Voronoi cell which indicates that the agents are evenly distributed.

2.2.2.2 Potential Field Coverage Control

Potential field coverage control is a form of coverage control that is based on the concept of potential functions. It uses artificial potential fields to guide mobile agents to perform coverage task while avoiding obstacle and collisions. This technique is inspired by the concept of how objects interact in physical field such as gravity and magnetic fields. Potential field coverage usually will implement a grid-based technique by dividing the area to be covered into grid cells and the grid cells will later form a waypoint for the motion of Multi Unmanned Aerial Vehicle (MUAV). Each of the cells will be assigned with a virtual potential field value which can also be known as gradient value and this value indicates which the agents should move to. Uncovered areas have a higher potential value and a lower potential value for the covered areas. The value can also be representing the respective grid cells are an obstacle.

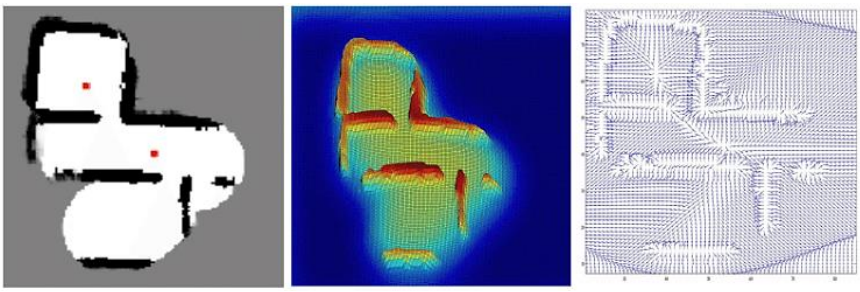

Figure 2.10: Potential Field for a Covered Area with Obstacle (Julia et al., 2011).

Figure 2.10 shows examples of how global potential field and gradient generated are taking in account of the occupancy grip map. Obstacles that are assigned with potential value will form a repulsive force that will repel the agents from getting nearby. The higher the value, the higher the repulsive force and the closer proximity to an obstacle will causes a stronger repulsive force. With this repulsive force, the repulsive force between agents will increase and lead to the agents move in a direction that is not covered (Miao et al., 2021). The agents will detect the current location's potential value and move to a direction based on negative gradient with steepest descent direction to minimize the objective function (Huang et al., 2018). In shorts, the potential field method is categorized into decentralized coverage control as there exists a repulsive force between agents indicates that there is a communication between agents.

The implementation of potential field in coverage control is first proposed by (Howard et al., 2002) where the potential field was generated so that each agent repelled by each other (dynamic obstacle) as well as obstacles (static obstacle). According to (Wang and Guo, 2008), potential field approach is implemented to control the deployed mobile sensors to achieve coverage goal with maximum coverage and total communication distance minimized. Besides, the decentralized control law exhibits the characteristics to avoid potential collision between agents (Wang and Guo, 2008).

A distributed welfare game with a designed potential function is developed by (Marden and Wierman, 2008) and the method for global controller to distribute global welfare to the players are investigated. In this case, each of the agents will be represented by the players. The effectiveness of the distribution control law is measured based on two criteria which is does pure Nash equilibrium exist and the efficiency of Nash equilibria compared to global optimum across various scenarios (Marden and Wierman, 2008).

Potential field can be modified and hybridized by combining two different approaches into one method to achieve the advantages from both approaches. The decentralized adaptive solution for coverage problems, achieved by combining Lie bracket trajectory approximation with a potential game algorithm, operates effectively without the necessity for agents to possess prior information regarding event distribution or detection probabilities. (Dürr et al., 2011).

2.2.2.3 Broadcast Coverage Control

As mentioned earlier, the coverage control is distributed into centralized and decentralized coverage. Starting from centralized coverage which all the agents are controlled by a global coordinator and found it extensively required more computational load when the Multi Agent System (MAS) is scaled up. Besides, centralized coverage is highly dependent on the global controller which when there is any malfunction for central controller or is compromised, the entire MAS will not be in control and results in failure. To overcome this, decentralization for coverage control is developed. Nowadays, many existing control techniques and algorithms developed for coordinate the motion of MAS with the aim of achieving specific tasks primarily emphasize the necessity of enabling communication among all the agents involved (Ota, 2006; Xie and Liu, 2017; Cao et al., 2013). However, decentralized coverage might not be perfect in solving the coverage problem. Although changing from centralized to decentralized can be more scalable, but this method has no information or limited information is given before it performs the task. The lack of information might lead to sub-optimal coverage due to the agents are not aware of the global state of coverage.

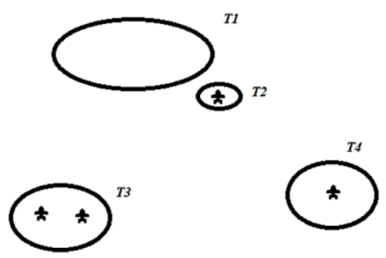

Figure 2.11: Sub-optimal coverage with Decentralized Coverage (Huang et al., 2018).

Figure 2.11 shows the sub optimal coverage that will be resulted by decentralized coverage and many decentralized coverage algorithms converge to a local minimum solution which is sub-optimal in a global sense, and it has a huge negative impact in the application (Huang et al., 2018). Although decentralized coverage is able to handle large number of agents in MAS, but there still exist scalability limitations. When the number of agents is large to an extent, the complexity of maintaining coordination among them are prohibitive as each agent will receive information from its neighboring agent and perform calculations before the decision-making process (Darmaraju et al., 2019). The characteristics of decentralized coverage in communicating and receive information with neighboring agents can significantly increase the computational load when number in a MAS increase due to highly complex network of communication formed (Darmaraju et al., 2019).

Hence, to perfectly minimized the problem raises from centralized and decentralized coverage, Broadcast Control (BC) algorithms is a unique approach that combine both the advantages from centralized and decentralized coverage. In the framework of BC algorithms, there will be no communication between individual agents except an overall update is received while not the individual performance (Darmaraju et al., 2019). The behavior of the BC algorithm shows one to all communication.

The BC was first applied in an artificial cellular actuator system by (Ueda et al., 2007). The BC framework is later been applied into MAS by (Azuma et al., 2013). BC method is widely used in traffic control due to its effectiveness. A modified version from the standard BC framework is introduced to solve the instability in MAS during the motion coordination task. In the proposed algorithm, the deterministic move of the agents is set with a gain with limited value (Nor et al., 2017). The instability of agents in MAS are mostly due to the big gap of value in the updates for deterministic move, since the deterministic move is set with limited value, the deterministic movement is restricted.

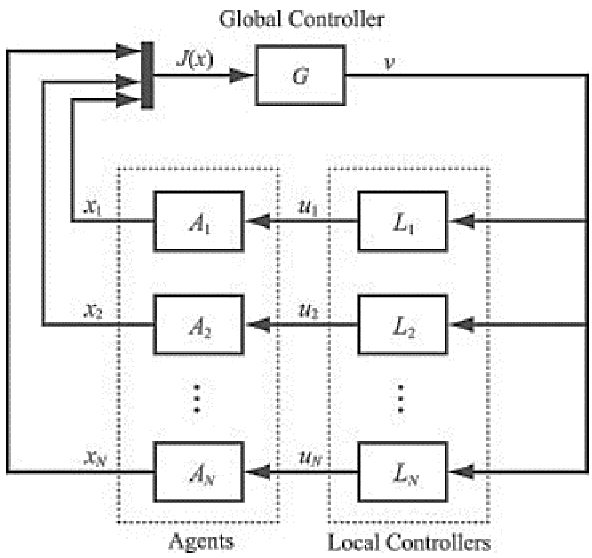

Figure 2.12: Framework of a Coverage Broadcast Control (Azuma et al., 2013).

From Figure 2.12, it shows that the BC system consists of multi-agents, global controller, and local controller. The performance of the agents will be evaluated by global controller and broadcast the global behavior signal to the local controller to make decision to control the action of agents.

While integrating BC into MAS has reduced hardware and communication requirements, it comes with inherent challenges related to broadcasting. One of these challenges is that agents may take unpredictable actions because BC relies on stochastic optimization. This randomness in agent movements can undermine the effectiveness of the control law, leading to subpar coordination performance and potentially resulting in undesirable configurations (Darmaraju et al., 2022). The introduction of Pseudo-perturbation-based Broadcast Control (PBC) law is the solution for this drawback. PBC is modified version of BC law by applying single step agent movement with predictive move instead of random movement (Ito et al., 2020). 2 steps of random movement in even time steps and deterministic movement in odd time steps from BC is evolved into 1 step which the step included 2 movement as BC but with a predictive movement instead of random movement. This change has improved the performance of traditional BC proposed by (Azuma et al., 2013) in terms of decrease in convergence time taken and reduce the total distance travelled by agents.

The evolution of BC continues with the name of Multi-step Broadcast Control (MBC) proposed by (Darmaraju et al., 2022). Traditional BC even PBC rely on a single-step perspective of the environment, and they do not promptly acknowledge the fluctuating distribution density of the environment as perceived by the agents. This phenomenon will cause the coverage algorithm to converge into a suboptimal performance result. The forestall this, MBC has successfully developed by the researchers from the BC schemes. Instead of calculating the varying distribution density of the environment one step ahead, MBC uses a predictive multi-step view by calculating the varying distribution density multiple steps ahead of time. Weighted averaging technique is apply into the local controller output and a higher weight is assigned to immediate steps (Darmaraju et al., 2022). Addition of this technique will contribute to the increase of accuracy as the number of steps increase. MBC is justified and is compared with the traditional BC in term of efficiency and converge time.

2.3 Three-Dimensional Coverage

As quadrotors are the agents that work in this project which are related to the Three-dimensional (3D) environment. However, the methods to perform coverage task for Multi Unmanned Aerial Vehicle (MUAV) mention earlier are conducted in a Two-dimensional (2D) preset environment and it shows that most of the works concentrate only on 2D, and this limited the behaviors of agents with only working on a planar surface or the height for each iteration are consider as constant to achieve a 2.5-dimensional work (Yang et al., 2016). According to (Yang et al., 2016), the real environment that the coverage control is going to implemented are unstructured and full of uncertainties which proof that 3D algorithms are urgently needed nowadays and a simple 2D algorithm will not be qualified in dealing with the real-world complex situations. Most of the existing static and dynamic coverage control algorithms consider only 2-dimensional field and the literature lacks a proper analysis of coverage control in 3D field even though the agents used in the proposed method assume the agents are at fixed altitude with planar sensing footprint, yet the capabilities of the agents might not be fully utilized (Elmokadem and Savkin, 2021).

Coverage algorithm can be classified into centralized, decentralized and both but a coverage problem can be classified into static and dynamic. In another war, coverage problem can also be divided into blanket coverage, barrier coverage, and sweeping coverage (Elmokadem and Savkin, 2021). Blanket coverage is done by covering the area in static formation to maximize the detection rate and coverage is formed according to the surface of the environment. Barrier coverage is done by covering the area of interest with maximizing the detection range and minimizing intrusions which the agents are located at a constant plane. Sweeping coverage is done by forming a dynamic arrangement across the area of interest to explore along the area. Typically, Sweeping can only be done with a basic of barrier coverage or blanket coverage as a foundation which the agents will have to form up blanket or barrier coverage before the sweeping process initiated.

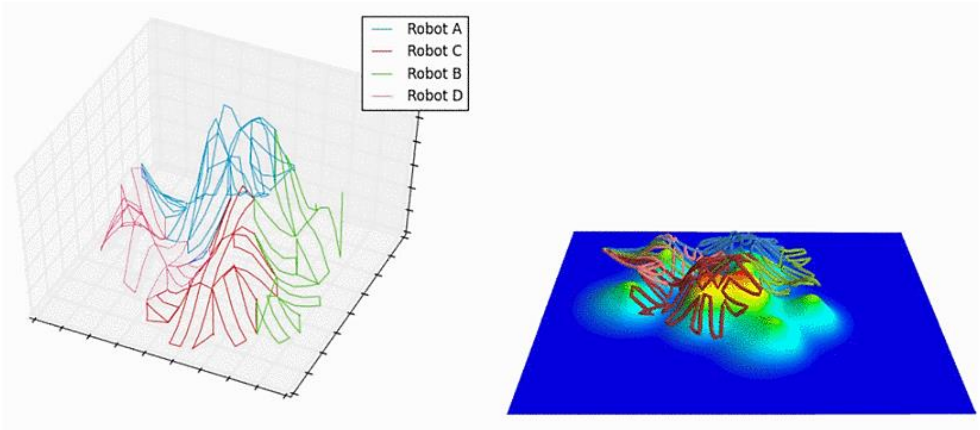

Figure 2.13: Sweeping Coverage with Blanket Coverage Basic (Perez-Imaz et al., 2016).

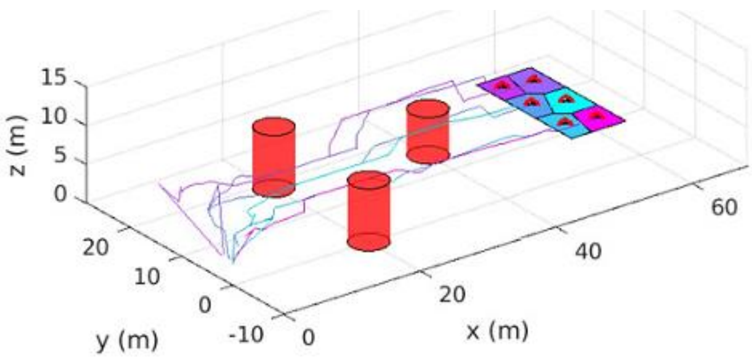

Figure 2.14: Sweeping Coverage with Blanket Coverage Basic (Elmokadem and Savkin, 2021).

A coverage control algorithm can be converts into a 3D coverage algorithm. According to (Elmokadem and Savkin, 2021), a 2D decentralized prioritized motion planning for MUAV has been converted into a coverage control that are able to function in a 3D environment which the algorithm used is a hybridized algorithm of A* algorithm with barrier functions-based method. The work is then further study for the feature of collision avoidance and the efficiency as well as the applicability of the conversion of coverage control algorithm is verified with simulation and experimental results using multiple quadrotors.

The current literature found for coverage control in 3D are mostly focusing on Coverage Path Planning (CPP) where CPP in 3D to perform inspection of complex 3D structures or formulation of unknown environment. Inspection of large scale and complex 3D structures were performed by (Jing et al., 2020) with sampling based CPP method. The sampling-based CPP method is tested on several complex 3D structures that are extracted from OpenStreetMap by combining the proposed method with modified Biased Random Key Genetic Algorithm (BRKGA). The results show an outstanding performance by reducing the path length up to 48% (Jing et al., 2020).

Forming a 3D coverage algorithm is just a framework or foundation to realize the idea from researchers. For example, collision avoidance is an important feature that cannot be ignored in developing the coverage control algorithm. Hence, a coverage control algorithm for 3D space should be constructed so that the function such as obstacle avoidance, varying importance of environmental, and method to overcome constraint of the agents can be implemented into the framework. Distributed coverage control has been a popular coverage control in 3D space recently. In (Hu et al., 2020) ‘s proposed work, an assignment switch scheme is embedded into a decentralized control algorithm to ensure that the asymptotic convergence is achieve without the occurrence of collision between agents. The results show the effectiveness of proposed control law and have been verified by (Hu et al., 2020) with Monte Carlo simulations as well as actual experiments in outdoor environment (3D space). Besides, implementation of 3D coverage control algorithms in disaster and rescue scenarios has been done by (Perez-Imaz et al., 2016).

According to (Perez-Imaz et al., 2016), MUAV for sure to have outperform advantage over other type of robots in disaster region where the MUAV can detect over large area in a short time with privileged view from above. Customized cell decomposition algorithm with regular hexagons is proposed by (Perez-Imaz et al., 2016) in a 3D area with MUAV and this approach has been evaluated in variety of disaster scenarios and the simulation is supported by the results conducted in an outdoor environment.

2.4 Summary

To summarize the literature review, there will be three coverage problem which is barrier coverage, blanket coverage, sweeping coverage where the three of them can be classified into static coverage or dynamic coverage. The coverage control can be divided into four category such as Coverage Path Planning (CPP), centralized coverage control, decentralized coverage control, and control that are with both characteristics. The coverage control can also be further classified into algorithm for Two-dimensional (2D) or Three-dimensional (3D). From the literature found, the implementation of distributed coverage control in 3D is now the trend and most of the 3D coverage algorithms are dealing with CPP as the coverage problem. After the study of literature, Multi-step Broadcast Control (MBC) is used to as a reference to the coverage control algorithm and the static barrier coverage will be the reference to the coverage problem in this project. MBC is selected as a reference to this project due to its advantages in dealing with the scalability as well as the effectiveness in low computational load and convergence time. 2D coverage algorithm are sufficient for a simple coverage task but it does not have the ability in dealing with complex coverage task. Hence, the development of 3D algorithms is crucial as it can implement in application to handle with the complex situation faced in real life. BC frameworks have no attempt in 3D environment, and it will be great if such a competitive coverage control algorithm can be converted to adapt with 3D environment and perform coverage task with perfect efficiency.

CHAPTER 3

METHODOLOGY AND WORK PLAN

3.1 Introduction

In this chapter, the methodology used to achieve the objectives for this project will be discussed in detail. Moreover, the additional information such as assumptions and environment setting will be stated in this chapter as well as the pseudocode and flow chart of the coverage task.

3.2 Proposed Multi Unmanned Aerial Vehicle Coverage Algorithm

Multi-step Broadcast Control (MBC) is the coverage algorithm chosen to work on to achieve the objectives of this project. MBC is selected due to it is neither decentralized nor centralized coverage, yet it combined the advantages of both decentralized and centralized. Besides, MBC is the latest and efficient coverage algorithm within the Broadcast Control (BC) scheme.

3.2.1 Multi-step Broadcast Control (MBC)

As mentioned earlier in chapter 2, the coverage control methods proposed by the researchers are mostly centralized or decentralized coverage method which the global task are known for every agent, and it means that all individual agents are acknowledge and store the information of other agents. This characteristic eventually raises the communication data volume. MBC is an evolutionary product from Broadcast Control (BC) schemes. Hence, it is important to understand how a BC works as it is a foundation for Multi-step Broadcast Control (MBC). The optimization methods applied in the BC scheme and its variant are stochastic optimization methods. Stochastic optimization is a process in minimizing or maximizing the objective functions when facing with stochastic problems which the formulation is attached with randomness or uncertainty (Cosma et al., 2017). BC scheme is a one to all communication which individual agents will not receive other agents' state of information and a global information is delivered to every individual agent (Darmaraju et al., 2022). Since the overall performance of the Multi Agent System (MAS) is observed by the global controller and the same information is received by every individual agent, the agents do not require to spare out the memory and energy for data transmission. Besides, the agents in BC does not have any idea of what the global task and they will only receive current global information sent by global controller. BC do have drawback as mentioned earlier and its variant, Pseudo-perturbation based Broadcast Control (PBC) has overcome the problem and shows a considerable enhancement in the results by reducing the convergence time by half.

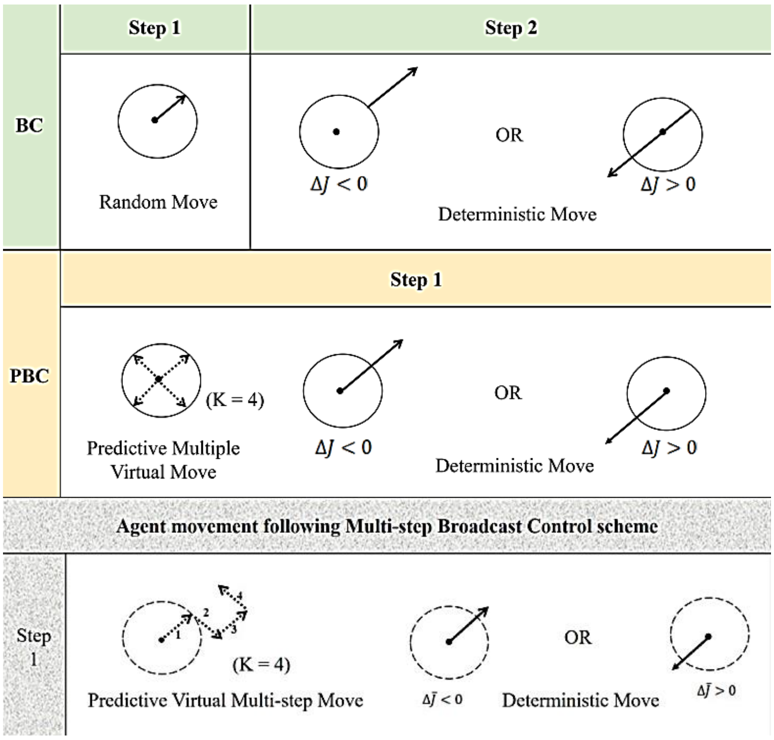

Figure 3.1: Movement Comparison for BC/PBC/MBC Scheme (Darmaraju et al., 2022).

BC will have 2 steps which the agents will move in random positions in the first step and followed by the deterministic move in the second step as shown in Figure 3.1. The agents' movement sequence is as follow; random move will be performed by agents at even time steps followed by deterministic move according to the value of objective function in odd time steps. However, this is troublesome as it takes up the convergence time. PBC which is the variant of BC and it combine the 2 steps into 1 step. Moreover, random movement in the BC scheme is replace by a predictive multiple virtual moves and followed by the deterministic move as shown in Figure 3.1. This results in the convergence time shorten by half as it initially required 2 steps and in PBC the predictive move and deterministic move is done in 1 step.

MBC is later proposed by (Darmaraju et al., 2022) and this is the modified version based on the PBC scheme. The working principle of MBC can be visualized from Figure 3.1 which the movement of agents are the same as PBC just the predictive move is given certain gain so that there will be multiple predictive steps before the deterministic move. In the proposed work from (Darmaraju et al., 2022), the MBC scheme is inspired by the varying importance of the environmental as well as the model predictive control theory. Since the global task are not known for the agents in BC scheme, the convergence time will be longer when dealing with coverage task that have different density functions. To overcome the issue arise, MBC scheme proposed involves making predictions for future variables in an environment by considering multiple steps into the future. A greater weight is assigned to predict the immediate step and the weight assigned to each step will decrease gradually when the number of steps increase. To ensure fair and balanced service coverage in an environment with varying population densities, a greater number of agents are allocated to areas with high population density than to those with lower population density. With the use of multi-step forward views, the MBC agents are made aware of the dense sections earlier than conventional BC schemes, thereby improving the timeliness of their response. MBC outperform BC schemes and its variant in both coverage achievement and deployment efficiency as the stochastic accuracy gradient accuracy is greatly increase with its noteworthy feature. If the density function for the environment is equally distributed, the MBC scheme will perform similarly as PBC scheme.

Objective function for the coverage control (Cortés et al., 2004):

$$J_{obj}(x) = \int_Q \min_{i\in 1,2,\ldots N} f(\|q - x_i\|)\phi(q)(dq) \tag{3.1}$$

where

f = coverage performance function

q = uniformly distributed points of the environment.

ϕ = weightage function is used to manage the relative significance of the points within the uppercase letter, Q.

x_i = location of the agents indexed as 'i' in domain ranges up to Nth position.

The objective function for coverage control is consistent and remains the same across all coverage control algorithms. Additionally, 'N' represents the total count of agents within MAS.

The objective function with respect to time, $J_{obj}(x(t))$ indicates the overall coverage performance and the target is to minimize the objective function with equation 3.2.

$$J\big(x(t)\big) = \min_{x\epsilon\mathbb{R}^{n\mathcal{N}}} J(x) \tag{3.2}$$

The N agents are strategically positioned within the space when the lowest value of objective function, $J_{obj}(x(t))$ is obtained.

To visualize the performance of coverage achievement, Voronoi tessellation method (Fortune, 1992) is used to partition the entire area of interest into polygonal cells. Voronoi tessellation method is created by points *(p_1, p_2, …p_n)* while the ideal division of set Q will align with the Voronoi partition denoted as V(P), which consists of subsets V1, V2, and so on up to Vn. This will result in the equation depicted in equation 3.3.

$$V_i = \{q\epsilon\mathcal{Q}\,\big|\,\|q - x_i\| \leq \|q - x_j\|, \forall j \neq i\} \tag{3.3}$$

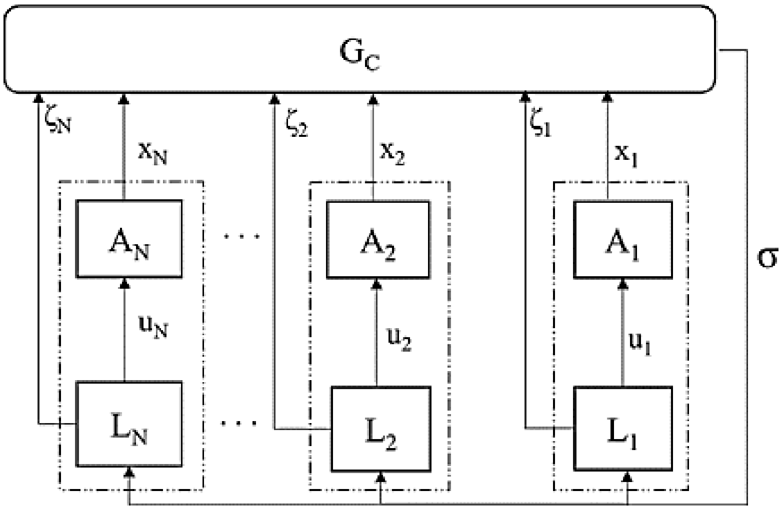

Figure 3.2: Framework for BC/PBC/MBC (Darmaraju et al., 2022).

In Figure 3.2, the illustration displays the framework encompassing BC, PBC, and MBC. Within this Multi Agent System (MAS), there exists a singular global controller, denoted as Gc. In this MAS, there are N agents, each represented as Ai where i ranges from 1 to N. Gc's role involves assessing the objective function by considering the overall system performance at every step. BC will take a random move followed by the deterministic move which the objective function will be evaluated while a PBC and MBC will perform the predictive movement and deterministic movement in a single step which the objective function will be evaluated in every iteration. The Gc will calculate the objective function by calculating the difference between the predictive and deterministic move and the evaluated results will be broadcasted to the agents represented as σ and the local controller, L_i will calculate based on the broadcasted control signal to determines the control action, $u_i(t)$. The local controller, denoted as Li for each individual agent Ai, produces a control signal by utilizing the information disseminated by Gc, leading to the formation of the following equation:

$$L_i : \begin{cases} \zeta_i(t+1) = \alpha(\zeta_i(t), \sigma_B(t), t) \\ u_i(t) \quad\quad = \beta(\zeta_i(t), \sigma_B(t), t) \end{cases} \tag{3.4}$$

where,

$\zeta_i(t)$ = state of local controller with initial value of 0.

$u_i(t)$ = results generated by local controller.

The controller functions for α and β can be described as:

$$\alpha(\zeta_i(t), \sigma_B(t), t) \coloneqq [\Delta_i(t)^t, \sigma_B(t)]^T \tag{3.5}$$

$$\beta(\zeta_i(t), \sigma_B(t), t) = \begin{cases} c(t)\Delta_i(t), & x < 0 \\ -c(t)\zeta_{i1}(t) - \alpha(t)\left(\dfrac{\sigma_B(t) - \zeta_{i2}(t)}{c(t)}\right)\zeta_{i1}^{[-1]}(t), & x \geq 0 \end{cases} \tag{3.6}$$

The agent's state equation is expressed in the following manner:

$$A_i : x_i(t+1) = x_i(t) + u_i(t), \qquad i = 1, 2, \dots N \tag{3.7}$$

where,

$x_i(t)$ = position in n-dimensional space while

$u_i(t)$ = control input.

The broadcast signal originating from Gc is characterized as:

$$G_c : \sigma_B(t) = J\big(x(t)\big) \epsilon \mathbb{R} \tag{3.8}$$

<u>MBC Scheme</u>

Within MBC scheme, K is designated as the upper limit for the maximum number of predictive virtual steps taken within the planning horizon. At every iteration, agents transmit their current state, denoted as $x_i(t)$, to the global controller. The state-space equation for agent, *i* within the MBC framework is presented as follows:

$$\hat{x}_i^{(k+1)}(t) := \hat{x}_i^{(k)}(t) + \hat{u}_i^{(k)}(t) \quad \text{for } k = 0, 1, 2, \dots, K-1 \tag{3.9}$$

$$\hat{u}_i^{(k)}(t) := c(t)\Delta_i^{(k)}(t) \quad \text{for } k = 0, 1, 2, \dots, K-1 \tag{3.10}$$

where,

$\hat{u}_i^{(k)}$ = virtual input

$\hat{x}_i^{(k+1)}$ = ccccc

$\hat{x}_i^{(k)}$ = future virtual states

The global controller calculates the broadcast signal, denoted as σ_M thorough a composite process outlined as follows:

$$\sigma_M(t) := \begin{bmatrix} J\left(\hat{x}^{(1)}(t)\right) - J\left(\hat{x}^{(0)}(t)\right) \\ J\left(\hat{x}^{(2)}(t)\right) - J\left(\hat{x}^{(1)}(t)\right) \\ \cdot \\ \cdot \\ \cdot \\ J\left(\hat{x}^{(K)}(t)\right) - J\left(\hat{x}^{(K-1)}(t)\right) \end{bmatrix} \in \mathbb{R}^K \tag{3.11}$$

where,

$\sigma_M{}^{(k)}(t) := J\left(\hat{x}^{(k+1)}(t)\right) - J\left(\hat{x}^{(k)}(t)\right)$

$\sigma_M(t) \quad = \left[\sigma_M{}^{(0)}(t), \dots, \sigma_M{}^{(K-1)}(t)\right]^T$.

In MBC scheme, the parameters in the local controller, L_i has been modified to:

$$\zeta_i(t) := \left[\Delta_i{}^{(0)}(t)^T, \dots, \Delta_i{}^{(K-1)}(t)^T\right]^T \in \{-1, 1\}^{nK} \tag{3.12}$$

$$u_i(t) := -\alpha(t)\frac{1}{\sum_{k=0}^{K-1}\lambda^{(k)}}\sum_{k=0}^{K-1}\frac{\sigma_M{}^{(k)}(t)\lambda^{(k)}}{c(t)}\Delta_i{}^{(k)[-1]}(t), \tag{3.13}$$

3.2.2 Pseudocode and Flowchart of Proposed Algorithm

Pseudocode for Multi-step Coverage Control Algorithm

- Initialize parameters and data structures.

- Create a map matrix representing the environment.
- for Iterate from 1 to 500:

  Update the map.

  Update agent trajectories and plot them if needed.

  Perform agent movements and optimization using PBC4MPC.

  Update agent positions.
- Record data.
- Plot the recorded data.
- Record the algorithm run time.
- Create a Voronoi diagram based on agent positions.
- Calculate mean distance travelled by agents.

<u>Coverage Control Flowchart</u>

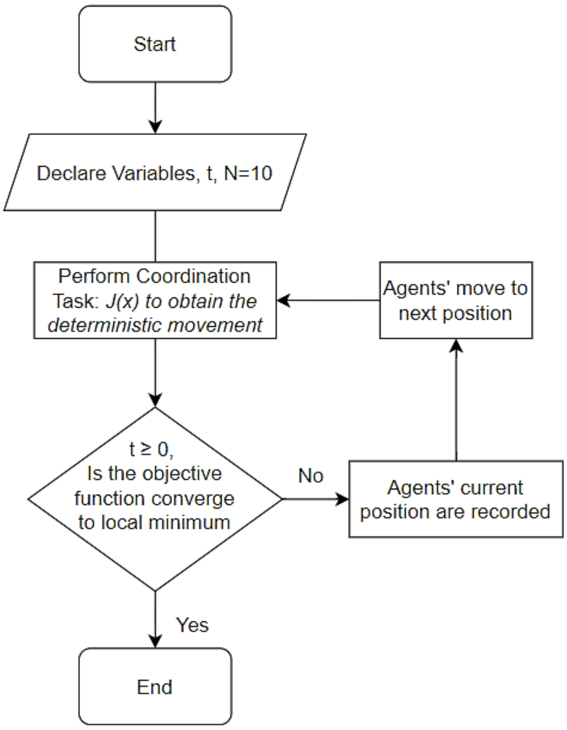

Figure 3.3: Flowchart for Coverage Task

The pseudocode and flowchart have briefly shown the overview of how the coverage task are going to conducted.

3.3 Three-Dimensional Multi-step Broadcast Control

The proposed project aims to develop a coverage control algorithm that suits to be implemented in a Three-dimensional (3D) environment. To do so, the algorithm must be converted into a 3D coverage control algorithm. The conversion of Two-dimensional (2D) to 3D can be form by adding additional z-axis to the original algorithm. The z-axis must be added into the generation of map to form up a 3D space. Next, the objective function of the coverage algorithm has to be added as well so that the function can be calculated to perform decision making. Besides, the output of the local controller will also have to ensure that the working dimension is in 3-axis to provide a coverage instruction to the agents. Last but not least, the positions of the agents have to be in 3 dimensional which is x, y, and z axis. To realize this, the additional of z-axis must be added into the state of agents.

To evaluate the performance of the proposed approach, the results are simulated using MATLAB for 2D space and 3D space. The parameters that will determine the performance of the conversion of algorithm is the distance of agents travelled and the convergence time taken by the agents. The agents that perform coverage task in 3D space should travelled according to Euclidean distance and the convergence time should be shorter due to the degree of freedom for the agents are not limited. To make the comparison fair, the agents that perform coverage task in 2D space will elevate the altitude to the specific level before the coverage start. Hence, the distance and time for the agents to elevated to the specific level will be considered for comparison.

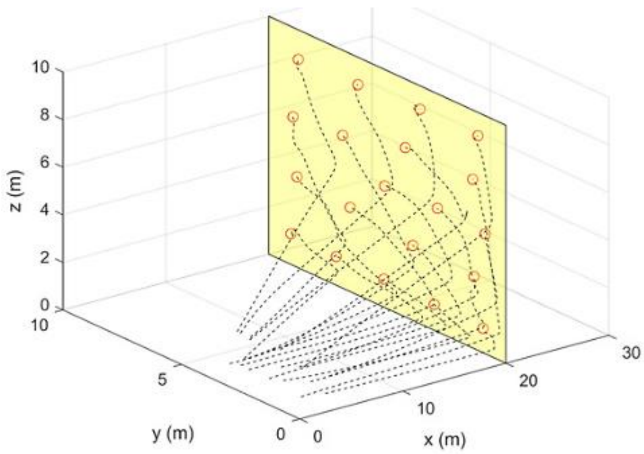

Figure 3.4: Static Coverage for MUAV in 3D (Elmokadem and Savkin, 2021).

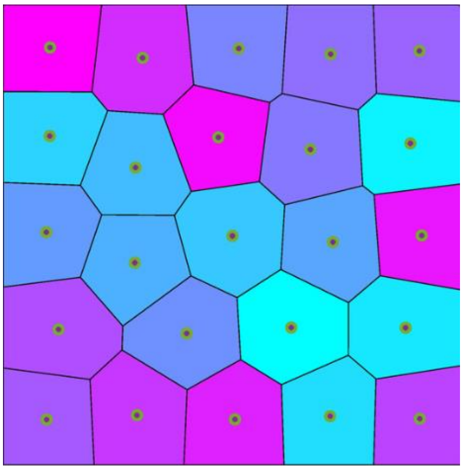

Figure 3.5: Static Coverage for MUAV in 2D (Elmokadem and Savkin, 2021).

The expected outcome for the coverage control algorithm should be able to visualize as in Figure 3.4, instead of the results shown in Figure 3.5.

3.4 Project Planning

Effective project planning is a cornerstone of project success. A proper project planning is important as this topic has not been gone through in the syllabus, and it required more time and effort to complete the project.

Table 3.1: Gantt Chart for Part 1 of Final Year Project

Part-1 Final Year Report															
No.	**Project Activities**	**W1**	**W2**	**W3**	**W4**	**W5**	**W6**	**W7**	**W8**	**W9**	**W10**	**W11**	**W12**	**W13**	**W14**
M1	Problem Formulation & Project Planning														
M2	Literature Review														
M3	Algorithm Research & Methodology														
M4	Preliminary Testing / Investigation														
M5	Report Writing & Presentation														

Table 3.2: Gantt Chart for Part 2 of Final Year Project

Part-2 Final Year Report															
No.	**Project Activities**	**W1**	**W2**	**W3**	**W4**	**W5**	**W6**	**W7**	**W8**	**W9**	**W10**	**W11**	**W12**	**W13**	**W14**
M1	3D Multi-step Broadcast Control Coverage Algorithm Development														
M2	Result & Discussion														
M3	Final Report Preparation														
M4	Poster Preparation														

3.5 Summary

This chapter provide a detailed description of Multi-step Broadcast Control (MBC) Scheme which is a variant of Broadcast Control (BC). Unlike the traditional BC, MBC Scheme provides greater stochastic accuracy and better coverage achievement as it predicts multi steps ahead before the deterministic move. This greatly reduced the convergence time.

In this project, the MBC scheme will be converted to a coverage control algorithm that are able to function in a Three-dimensioanl (3D) space without affecting the coverage performance of the original MBC scheme in a Two-dimensional (2D) space. It will be great if the results outperform the original approach.

The performance of the MBC scheme will be evaluated in both 2D and 3D environments. The performance of the MBC scheme in 2D and 3D space will be evaluated based on the distance travelled by the agents and the convergence time. MATLAB is the simulation tools used with same starting and goal positions. Moreover, project planning and time management are strictly enforced to ensure the project completed the milestones on time to complete the project.

CHAPTER 4

RESULTS AND DISCUSSION

4.1 Introduction

This chapter mainly focus on presenting and discussing the simulation results of Three-dimensional (3D) Multi-step Broadcast Control (MBC) algorithm in coverage tasks using MATLAB. The results of the developed 3D MBC algorithm will be presented in section 4.2 together with the comparison results between the Two-dimensional (2D) MBC algorithm and the 3D MBC algorithm. In section 4.3, the further development of the 3D MBC algorithm in its real-life application will be presented and discussed.

4.2 Comparison Between 2D and 3D MBC Algorithms

In this section, the results of Three-dimensional (3D) Multi-step Broadcast Control (MBC) algorithm will be presented, and comparison will be made to evaluate the performance of both Two-dimensional (2D) and 3D algorithms. The performance of the algorithms will be evaluated by using the evaluation parameters stated in the objective of this research, which is the convergence analysis, path length, coverage quality, computational load, and scalability. The comparison results will be summarized and tabulated at the end of this section.

The 3D MBC algorithms, using stochastic optimization method, are developed based on the 2D MBC, which is also one of the variants of the Broadcast Control (BC) scheme. Therefore, both algorithms will undergo the process of minimizing the objective functions to perform coverage control. The 2D MBC scheme is inspired by the varying importance of the environment as well as the model predictive control theory. Hence, it involves making predictions for future variables in an environment by considering multiple steps into the future. The weight of the steps to predict the future is represented by the variable ‘k’. A higher value that are assigned to the ‘k’ variable will results in more steps predicted as ‘k’ is designated as the upper limit for the maximum predictive virtual steps taken within the planning horizon. This function is crucial, as the MBC scheme will have a higher convergence time when dealing with coverage tasks that have different density functions. Thus, predicting steps will allow agents to be aware of dense sections earlier than conventional BC schemes. This function is important, as the 3D MBC algorithms can be applied to more applications compared to 2D MBC algorithms. A 2D MBC algorithm will allow a greater number of agents to move to areas with high population density compared to those with low population density. This function is greatly inherited in the 3D MBC algorithm, as it can perform more than the 2D MBC algorithm by assigning different heights to agents with different areas of interest.

Moreover, the objective function is the same as the 2D MBC algorithm which is the coverage control objective function proposed by (Cortés et al., 2004), as mentioned above in section 3.2.1, equation 3.1. The equation of 3.2 mentioned in section 3.2.1 is the same for both 2D and 3D MBC algorithms, where the target is to minimize the objective function across time. A convergence graph can be illustrated with the calculations from equation 3.2. Some modifications were made with equations mentioned in section 3.2.1, as the calculations for 2D are slightly different compared to 3D, with the addition of one axis in n-dimensional space. The addition of one axis will have to be incorporated into the equations for local controller equations, agents' state equation, as well as the global controller equation.

The Voronoi Tessellation method (Fortune, 1992) used in 3D MBC algorithm remains unchanged from the 2D MBC algorithm. The Voronoi Tessellation method is remained unchanged as the Voronoi tessellation is used to partition the entire environment space into polygonal cells to visualize the performance of the coverage achievement. Hence, there is no need to adjust the settings for Voronoi, as the coverage achievement for the 3D MBC algorithm can be visualized in the X-Y axes similar to the 2D MBC algorithm.

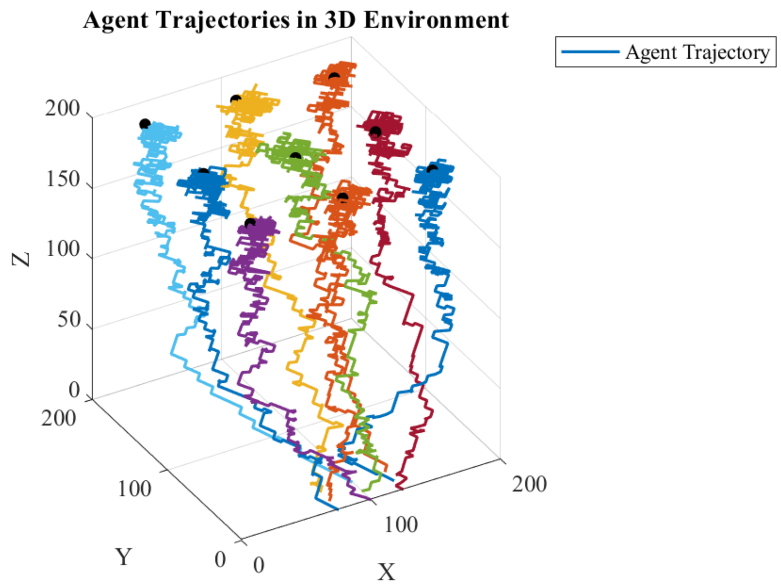

Figure 4.1: Agents' Trajectories in Performing Coverage Task using 3D MBC Algorithms for 9 agents.

The developed 3D MBC algorithm successfully perform coverage task in 3D environment as shown in Figure 4.1. The environment is set to be a 3D space of 200m x 200m x 200m. The starting position of the agents is set in within the range of 0 - 25 in Y-axis and 75 - 125 in X-axis. The starting positions of the agents will lie within this range regardless of any scenarios and the number of agents.

The motion of the agents is based on randomness, as mentioned earlier, where they are given an initial value of 1 or -1 for direction in the random move stage, and the deterministic move stage determines the direction of the agents based on the objective function calculated in each iteration. This characteristic eventually leads to the results obtained from every attempt being unstable. This greatly affects the accuracy of the results obtained, and the repeatability of the results will be disputable. To counter the issue, the random values generated from the first attempt will be stored in separate empty matrix array files for the X, Y, and Z axes. When the sequence of random values from the first attempt is stored into the empty matrix array files, the sequence of random values stored can be extracted for the following attempts to ensure the precision of results. Implementation of this feature minimizes tolerances due to the randomly generated values. The precision of the results and repeatability of the simulation can be guaranteed since the random values assigned to every iteration for the random movement are the same for every attempt. The simulation results for every evaluation parameter will be the same no matter how many attempts are given.

To evaluate the performance of the 3D MBC algorithm, several evaluation parameters have been determined. There are 5 evaluation parameters: convergence analysis, path length, coverage quality, computational load, and scalability. The comparison between 2D and 3D MBC algorithms allows for a clear identification of superiority. By employing evaluation parameters, we can discern the performance of both algorithms.

4.2.1 Convergence Analysis

Convergence analysis focuses on the convergence rate, where the higher the convergence rate, the shorter the convergence time. Convergence time can be described as the time required for the multi-robots to form cooperative formations, and the smaller the convergence time, the better the algorithm (Zhuang et al., 2022). This has a direct correlation with the effectiveness and efficiency of the algorithm, making convergence time a crucial metric in Multi-Agent Systems (MAS). The ability of a group of Unmanned Aerial Vehicle (UAV) to converge to a certain formation or complete a task affects operational efficiency. A shorter convergence time indicates less time spent waiting for agents to coordinate, leading to faster task execution and improved overall efficiency in every application.

Besides, shorter convergence time will help in resource utilization. The resources such as computational power and energy will be utilized in a more efficient way. For example, if the agents can form cooperative formations quickly, lesser time spending will be required in idling or redundantly performing actions, thereby extending the operational lifespan by conserving valuable resources.

Moreover, a shorter convergence time is equivalent to a shorter response time, which is crucial in MAS applied in dynamic environments or time-sensitive scenarios such as search and rescue or surveillance. In such applications, rapid convergence to environmental changes is paramount. A MAS with a shorter convergence time can respond more promptly to dynamic environments, thereby improving its effectiveness in real-life situations.

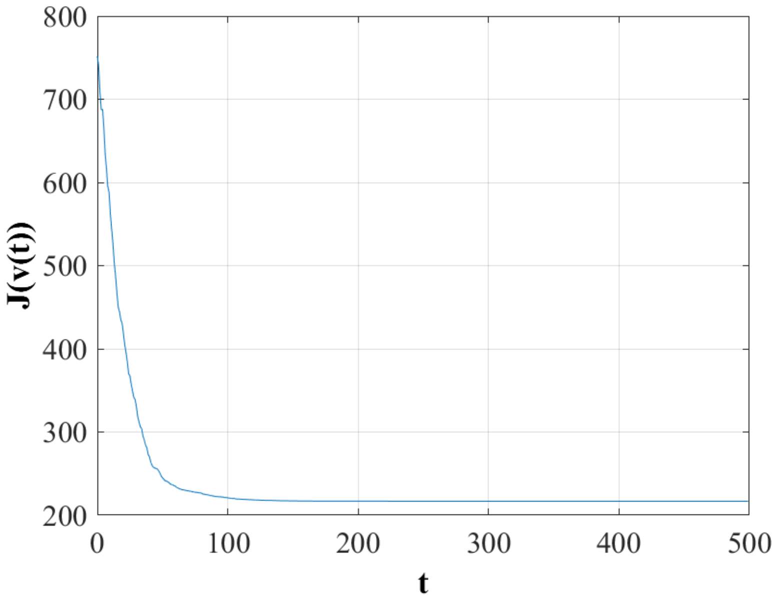

Figure 4.2: 2D MBC Algorithm's Convergence Graph for 9 Agents

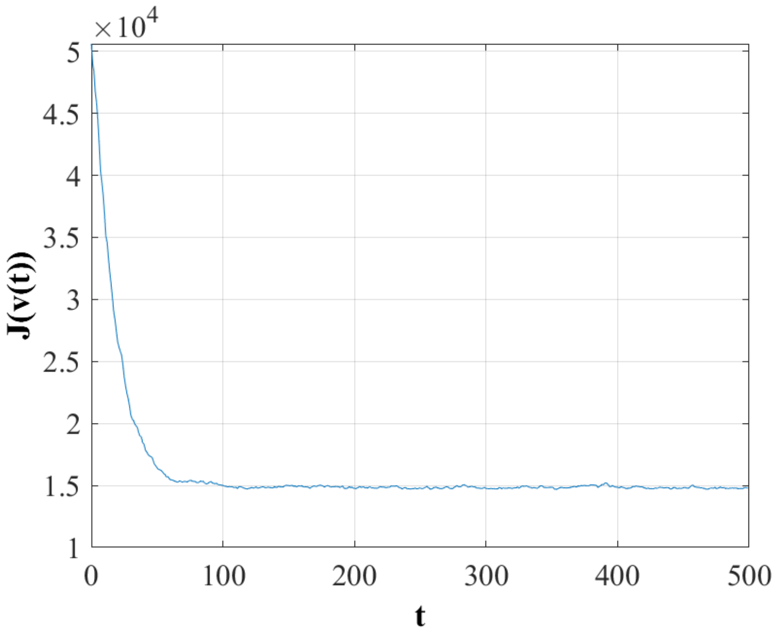

Figure 4.3: 3D MBC Algorithm's Convergence Graph for 9 Agents

Figure 4.2 and Figure 4.3 show the Two-dimensional (2D) and Three-dimensional (3D) Multi-step Broadcast Control (MBC) Convergence graphs of the objective function against time for 9 agents. The 't' on the graph represents the iteration. For both Figure 4.2 and Figure 4.3, the total number of iterations run is 500. The purpose of showing these graphs is to identify at which iteration the MAS achieves the desired formation. The x-axis is represented in 't' because it also represents the convergence time, as the number of iterations can be multiplied by the time required for each iteration to obtain the convergence time for the system. However, time is not focused on in this section, as the time per iteration can be affected by the computational load required to run the algorithm. In this section, the convergence rate will be highlighted by observing the shortest iteration required for the system to achieve stability and complete the given task.

From Figure 4.2, it can be observed that the calculation results for the objective function start to be consistent at approximately 100 iterations. After 100 iterations, the graph shows a consistent trend, with no changes in the value for the Y-axis (calculated objective function). This indicates that the 2D MBC algorithm will complete the operation formation for performing the coverage task at approximately 100 iterations.

Figure 4.3 indicates that the results of the objective function become stable at approximately 60 iterations. Beyond this point, there is a consistent trend in the graph, showing no further changes in the calculated objective function value per iteration on the Y-axis. This suggests that the 3D MBC algorithm achieves completion of the operation formation for the coverage task within approximately 60 iterations.

In short, the convergence analysis of Figure 4.2 and Figure 4.3 provides valuable insights into the convergence behavior of the 2D MBC algorithm and the 3D MBC algorithm. Both the 2D and 3D MBC demonstrate efficient convergence behavior, with the 3D variant exhibiting better convergence behavior. This indicates that, with the condition of other variables remaining constant, the 3D MBC algorithm will have a faster convergence time and response time compared to the 2D MBC algorithm.

4.2.2 Computational Load

Computational load can also be known as computational complexity, which is a measure of the amount of computing resources in term of time and space needed for a particular algorithm to run (Anon, n.d.). To be exact, computational complexity can be divided into two aspects, which is time complexity and space complexity.

Time complexity is the total amount of time the system requires to run the algorithm until the operation is complete with varying input size. In layman terms, it means the number of operation perform, or the number of iteration done to complete the task by an algotithm. Different algorithms will have different time complexities, and the performance of the algorithm on inputs of different sizes will be impacted.

Space complexity commonly refers to the amount of memory space an algorithm requires with varying size of the input. Space complexity varies depending on factors such as data structures, recursion depth, and temporary storage used. Hence, simplifying the code will eventually reduce memory consumption. Unnecessary code should be eliminated and simplified to prevent quality degradation, and efficient memory consumption is critical and should be emphasized.

In this section, the computational load of the algorithm can be observed by using the time consumed by the algorithm to run the required iterations. Computational complexity can be identified by calculating the time consumed by the algorithm starting from the first iteration to the final iteration. More time required to complete the operation indicates that the algorithm consumes more memory space. The time complexity will be focused in this section as the memory used by the algorithm is not being measured and tested. Time complexity describes the amount of time taken for the algorithm to finish its operation, with size of input as a variable function, while the results were obtained with a consistent input size of 9 agents. Hence, this indicates that there is differing time complexity between the Two-dimensional (2D) Multi-step Broadcast Control (MBC) algorithm and the Three-dimensional (3D) algorithm as the observed results were different.

```
>> MainMBC

co =

    0.0100

Elapsed time is 3.862232 seconds.
```

Figure 4.4: Computational Time Required for 2D MBC Algorithm with 9 Agents

```
>> MainMBC
Elapsed time is 2046.036723 seconds.
```

Figure 4.5: Computational Time Required for 3D MBC Algorithm with 9 Agents

Figure 4.4 indicates that it took 3.8622 seconds for the 2D MBC algorithm to complete its execution fully, whereas Figure 4.5 demonstrates that the 3D MBC algorithm required 2046.0367 seconds to achieve full execution completion. Both Figure 4.4 and Figure 4.5 show the time complexity for their respective algorithms, and the comparison results show that the 3D MBC algorithm has a larger computational load than the 2D MBC algorithm. It is undeniable that running a 3D algorithm in 3D space demands a significant amount of time for execution due to the increased calculations required. This is because there's an additional axis compared to the 2D scenario, leading to a greater computational load.

4.2.3 Coverage Quality

In Multi Agent System (MAS), coverage quality can be defined as the effectiveness and completeness with which a group of agents explore and monitor the given space or environment. Coverage quality assesses how effectively and thoroughly a MAS explore and covers the target area by considering the aspects of percentage of area covered and uniformity of coverage distribution. Besides, minimal coverage gaps are crucial as gaps can lead to missed observations or incomplete task fulfillment.

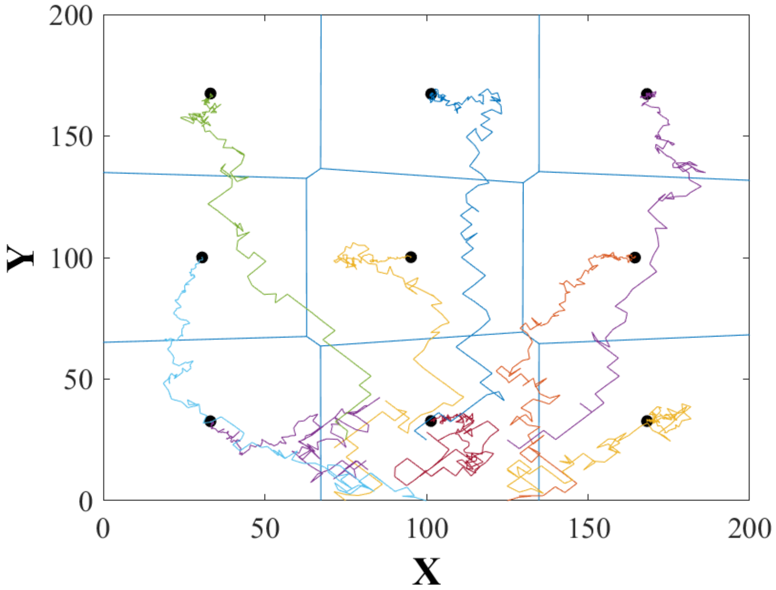

Figure 4.6: Coverage Quality for 2D MBC Algorithm with 9 Agents

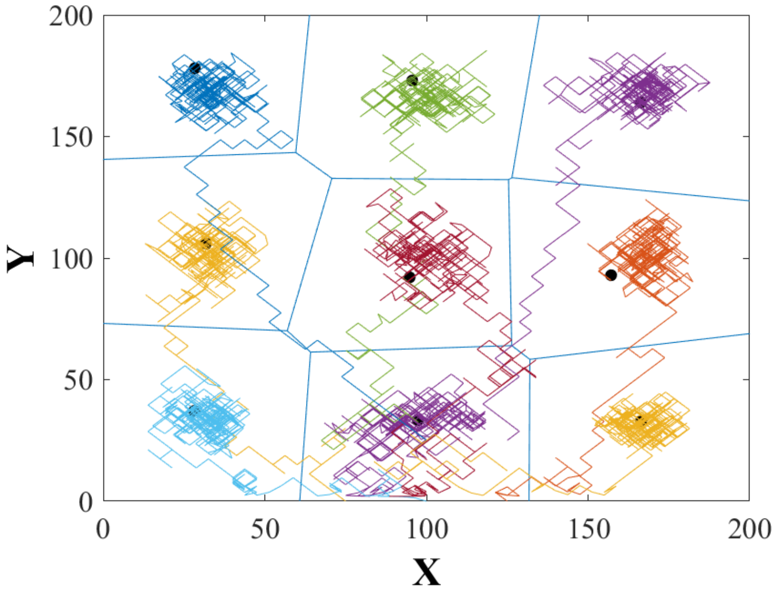

Figure 4.7: Coverage Quality for 3D MBC Algorithm with 9 Agents

Figure 4.6 and Figure 4.7 show the coverage quality for 9 agents with the Two-dimensional (2D) Multi-step Broadcast Control (MBC) algorithm and Three-dimensional (3D) MBC algorithm respectively. The coverage quality can be observed through Voronoi partitioning. The evenly distributed partition proves the quality of the coverage status, and the characteristics of Voronoi partition ensure that the percentage of cover area is maximized.

In Figure 4.6, it is a complete coverage task done by the 2D MBC algorithm, with the agents evenly distributed with 3 agents in a column and 3 agents in a row aligned with each other to form a 9-agent coverage network.

Similarly, Figure 4.7 showcases a complete coverage network done by the 3D MBC algorithm, with the agents evenly distributed with 9 agents arranged in a similar grid pattern as the 2D MBC algorithm. From Figure 4.7, it can be observed that the agents' path is more complex compared to the agent's path in Figure 4.6, where the agents' paths are shorter. This scenario happens due to the results taken in Figure 4.7 being a slice of the 2D view of the X and Y axes. The agents in Figure 4.7 seem to have longer paths because the agents are rising in altitude, and the captured 2D view is from the top of the 3D space, which might result in a misconception of extended agents' paths. Assessing coverage quality in a 3D space can be challenging, necessitating a top view for accurate evaluation. Hence, a slice of 2D view is captured along with Voronoi partition to visualize the coverage quality achieved by the 3D MBC algorithm.

Both Figure 4.6 and Figure 4.7 exhibit comparable coverage quality, as both agents are evenly distributed with each other and aligned to form a 3x3 static coverage network. Both achieve full percentage of area cover with the help of Voronoi partition, ensuring the completeness of static coverage.

4.2.4 Path Length

In Multi Agent System (MAS), path length usually refers to the total distance traveled by the agents while executing a task or covering a certain area. Path length is a crucial metric that will directly impact the overall performance of MAS. Path length can be separated into individual agent paths and cumulative path lengths.

The individual agent path is the distance traveled by each individual agent from its starting point to its destination, including all intermediate waypoints or extra paths needed due to the obstacles encountered along the way. Cumulative path length will be the sum of all individual agent paths. In this section, the method used to calculate the path length is by using the cumulative path length divided by the number of agents involved. The mean distance traveled by each individual agent will be calculated. Implementing individual agent paths in this study proves challenging due to variations in the distances between starting locations and ending points for each agent. To obtain an accurate path length result, the mean distance traveled by the agents is calculated with the aid of MATLAB.

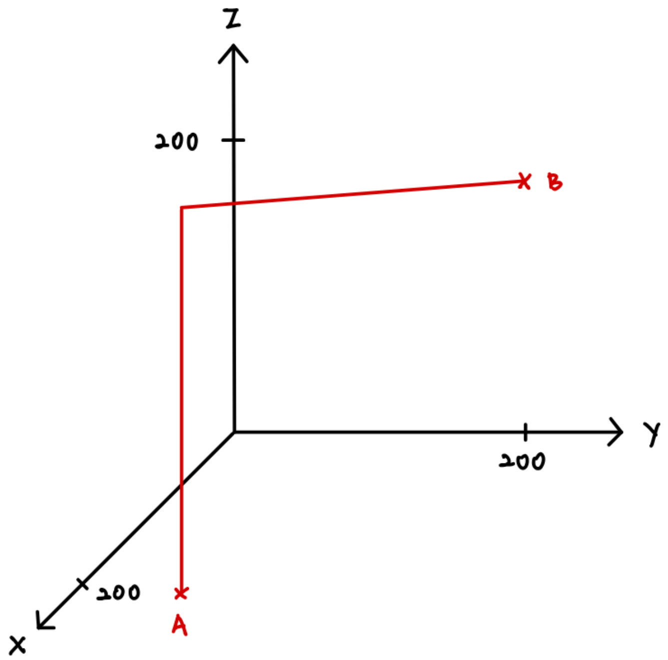

Figure 4.8: Travelling Route for Agents with 2D MBC algorithm.

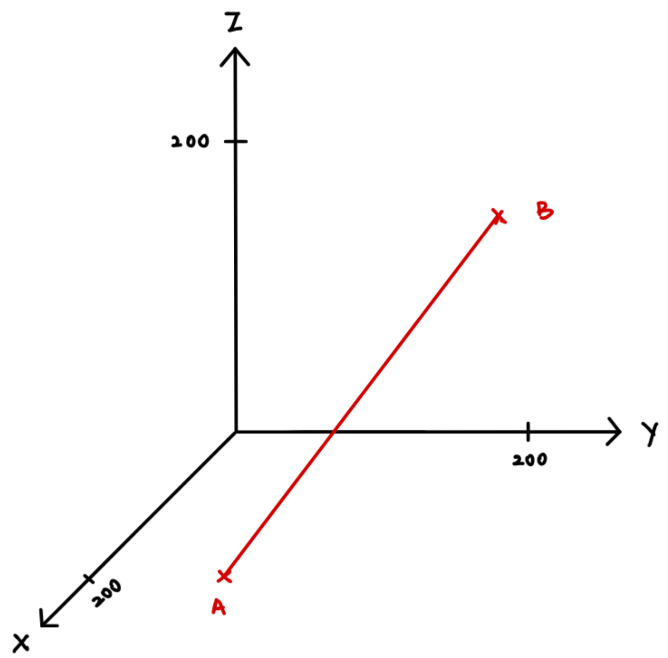

Figure 4.9: Travelling Route for Agents with 3D MBC algorithm.

In this study, the path length traveled by the agents in the Two-dimensional (2D) Multi-step Broadcast Control (MBC) algorithm and the Three-dimensional (3D) MBC algorithm will be compared. To meet the comparison requirement, an additional 200m must be added to the distance traveled by the 2D MBC algorithm. This adjustment compensates for the fact that the mean distance calculated for the 2D MBC algorithm operates within a single layer of 2D, where all agents are at the same altitude. This scenario can be simulated by augmenting the mean distance traveled by 200m, effectively simulating the scenario of the agents for the 2D MBC algorithm ascending to 200m, which is the desired altitude before commencing the coverage operation. Figure 4.8 highlights the necessity of adding the extra 200 meters for altitude, as the original path length generated in 2D cannot be directly compared to the path length generated in 3D. Thus, the 2D view can be converted into a 3D view with the agents rising to 200m in altitude and starting their coverage task, during which the agents will be moving in Manhattan distance. Whereas Figure 4.9 illustrates the path traveled by agents in the 3D MBC algorithm, where the agents are moving in Euclidean distance, which is a direct straight linear line from the start point to the end point. Figure 4.8 and Figure 4.9 depict the same starting point at point A and ending point at point B, representing the destination. However, they employ different metrics for measuring the traveling distance.

```
>> MainMBC

co =

    0.0100

Elapsed time is 3.862232 seconds.

ans =

         0  216.8190

The mean distance traveled by the 9 agents is 87.0964
>>
```

Figure 4.10: Path Length for 9 Agents using 2D MBC Algorithm Without Altitude Length

```
>> MainMBC
Elapsed time is 2046.036723 seconds.
The mean distance traveled by the 9 agents is 284.0383
>>
```

Figure 4.11: Path Length for 9 Agents using 3D MBC Algorithm

Figure 4.10 illustrates the mean distance traveled by the 9 agents using the 2D MBC algorithm in a 2D space. Additionally, the mean distance calculated must be augmented by an additional 200m, as mentioned earlier, to allow for comparison in a 3D space. Figure 4.11 depicts the mean distance traveled by the agents using the 3D MBC algorithm in a 3D space. To enable a valid comparison, the mean distance traveled using the 2D MBC algorithm must be increased by 200m, allowing for comparison with the mean distance traveled using the 3D MBC algorithm in the form of Manhattan distance, as shown in Figure 4.8.

Table 4.1: Summarized Table for Mean Distance Travelled by 9 Agents

Algorithm	**Mean Distance Travelled by 9 Agents**	**Distance Metrics**
2D MBC	87.0964 m + 200 m = 287.0964 m	Manhattan
3D MBC	284.0383 m	Euclidean

Technically, the Euclidean distance will be shorter than the Manhattan distance, as the Euclidean distance forms a straight line between the initial point and the destination, while the Manhattan distance sums up all the real distances between the initial point and the destination. Table 4.1 presents a summarized table for the mean distance traveled by agents using both algorithms. From the table, it can be observed that the path length using the 3D MBC algorithm is slightly shorter than the path length using the 2D MBC algorithm.

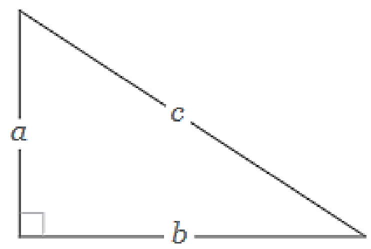

Figure 4.12: Right Angle Triangle (Represent Distances for 3 Points)

The Euclidean distance, represented as 'c' in Figure 4.12, is calculated using the Pythagorean theorem, while the Manhattan distance simply sums up the distances of 'a' and 'b' from Figure 4.12. The formula for the Pythagorean theorem to calculate the length of 'c' is:

$$c = \sqrt{a^2 + b^2}, \tag{4.1}$$

Hence, if 'a' and 'b' both have a value of 2, the Manhattan distance will total 4, while the Euclidean distance will be 2.83, calculated using equation 4.1. The results shown in Table 4.1 do not seem ideal, as there is only a small difference between the Euclidean distance and the Manhattan distance. This phenomenon occurs because the traveling path for the agents is sufficiently smooth to form a straight line, rather than the agents moving in a zigzag style, which would increase the path length. The distance traveled by agents using the 3D MBC algorithm can be further shortened in the future through enhancements in path-smoothing techniques. In short, the path length using the 3D MBC algorithm is shorter than the path length using the 2D MBC algorithm, justifying the need for the 3D MBC algorithm.

4.2.5 Scalability

Scalability describes the ability of a system, technology, or network to handle an increasing workload or resource demands while maintaining or improving efficiency, performance, and reliability. In Multi Agent System (MAS), scalability measures the capacity of the system to effectively accommodate changes or growth in usage without significant modifications or negative impacts on its functionality and performance. Scalability is important as it serves as an indicator of the system's ability to scale up and down in response to changes in demand, workload, or operational factors while continuing to meet desired performance and specifications. In this study, varying scales of operation may include changes in the number of agents, dimensions of the environment, and the complexity of tasks.

A scalable MAS can accommodate an increasing number of robots without degradation in performance, such as coordination efficiency, communication, and task allocation mechanisms. A scalable system must have the capability to handle both small and large agent systems effectively.

The scalability of a system extends beyond just the input size and can also encompass factors such as the dimensions of the environment when assessing the system's ability to adapt and perform efficiently. A scalable MAS should be able to operate effectively in environments of different sizes, ranging from indoor confined spaces to vast outdoor areas. The system should be able to adapt to changes in the size of the environment and specific characteristics to ensure efficient task execution and comprehensive coverage.

Moreover, MAS should be able to handle tasks of varying complexity, from simple coverage to specific formations, and even cooperate in performing multiple objectives simultaneously. The computational, sensing, and decision-making capabilities should meet the demands of increasingly challenging tasks without losing performance.

In this section, the scalability of the system will be justified by arranging 4 agents and 16 agents to act as scale-down and scale-up, respectively. The key focus of this section is to compare the scalability of the Two-dimensional (2D) Multi-step Broadcast Control (MBC) algorithm and the Three-dimensional (3D) MBC algorithm.

3D Simulation Model for Coverage Task using 3D MBC Algorithm.

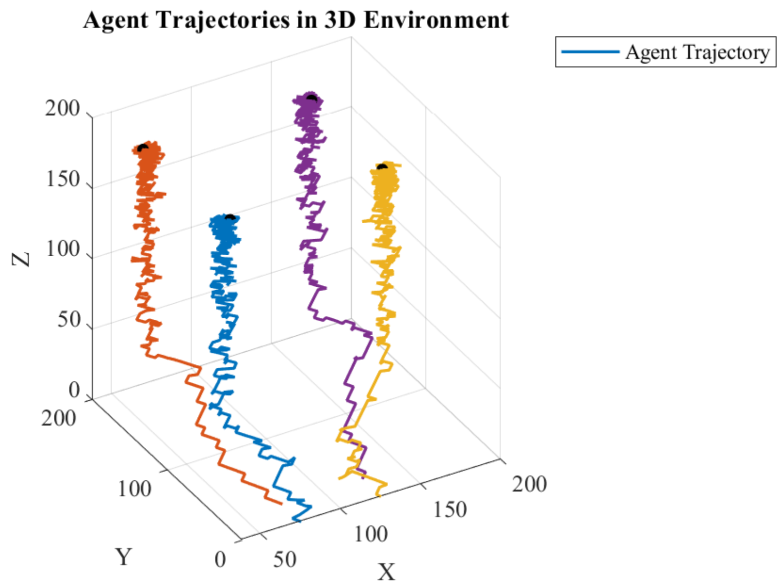

Figure 4.13: Agents' Trajectories in Performing Coverage Task using 3D MBC Algorithms for 4 agents.

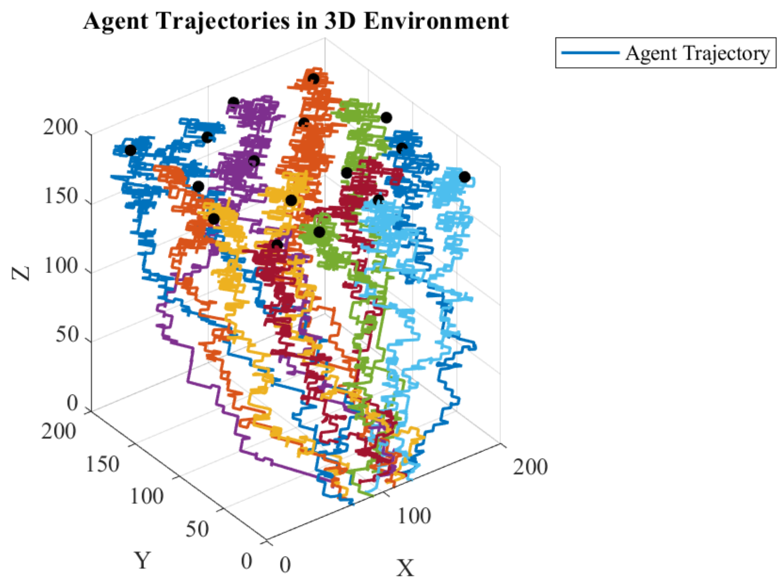

Figure 4.14: Agents' Trajectories in Performing Coverage Task using 3D MBC Algorithms for 16 agents.

Figure 4.13 and Figure 4.14 show that the development of the 3D MBC algorithm successfully simulated the coverage operation for 4 agents and 16 agents, respectively. Both figures demonstrate that the 3D MBC algorithm can scale down and scale up to accommodate 16 agents. Further research can be done to investigate the maximum scale-up limit of the 3D MBC algorithm.

Coverage Quality for 4 Agents and 16 Agents using Both Algorithm.

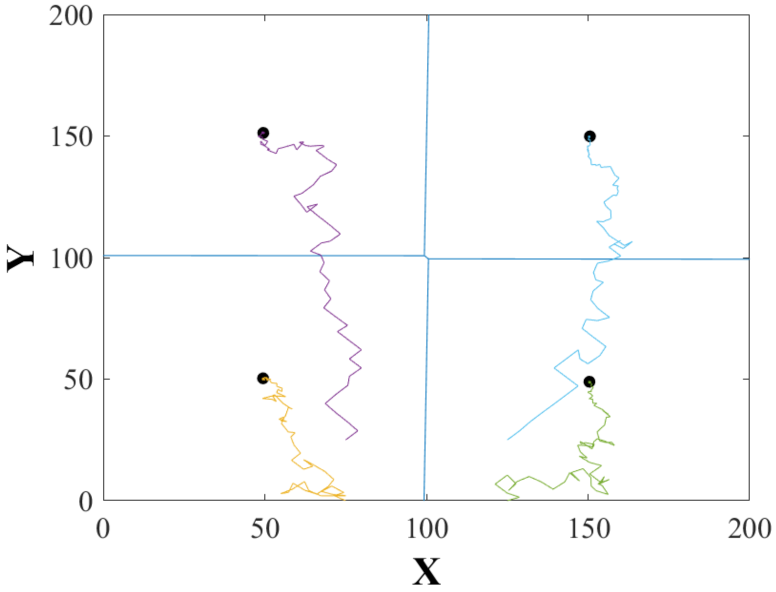

Figure 4.15: Coverage Quality for 2D MBC Algorithm with 4 Agents

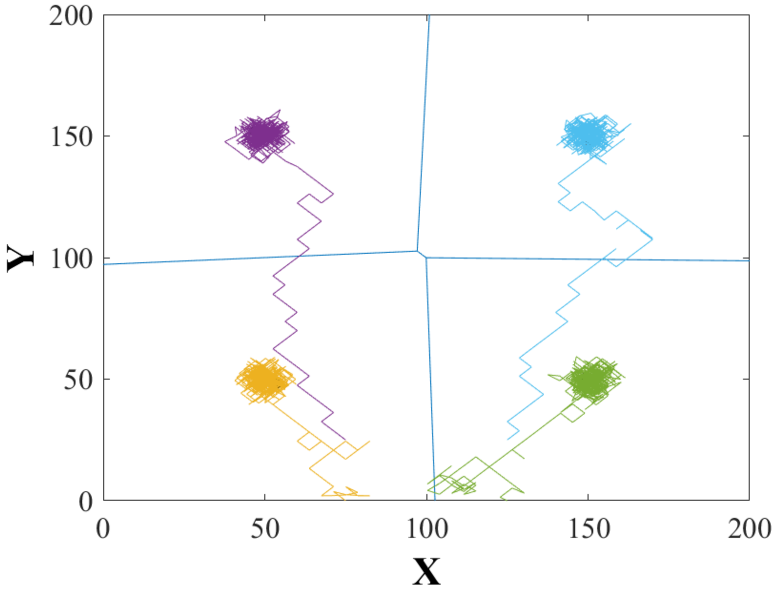

Figure 4.16: Coverage Quality for 3D MBC Algorithm with 4 Agents in X-Y Axes View

Figure 4.15 and Figure 4.16 depict scaled-down versions for both the 2D MBC algorithm and the 3D MBC algorithm. It is evident that the coverage quality generated by both algorithms is similar, with agents aligned to form a 2x2 formation consisting of two agents in a row and two in a column. This demonstrates that the coverage quality remains consistent even when transitioning from the 2D MBC algorithm to the 3D MBC algorithm.

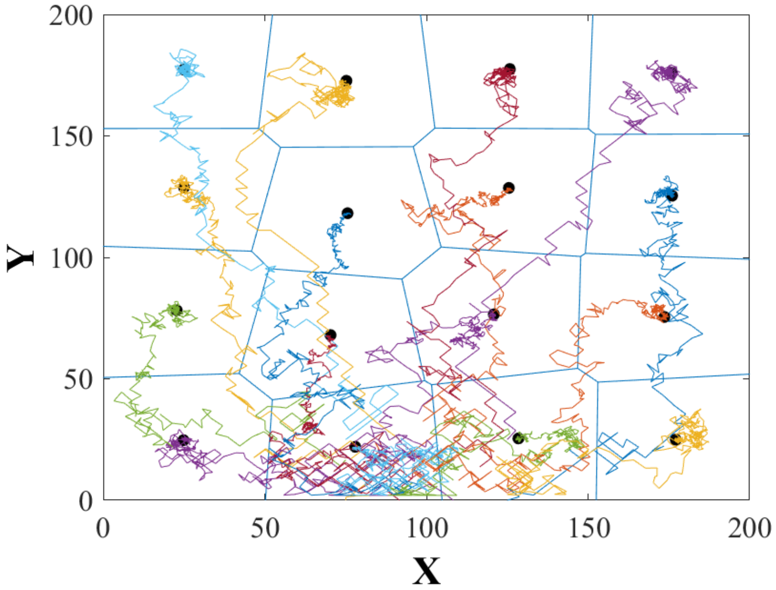

Figure 4.17: Coverage Quality for 2D MBC Algorithm with 16 Agents

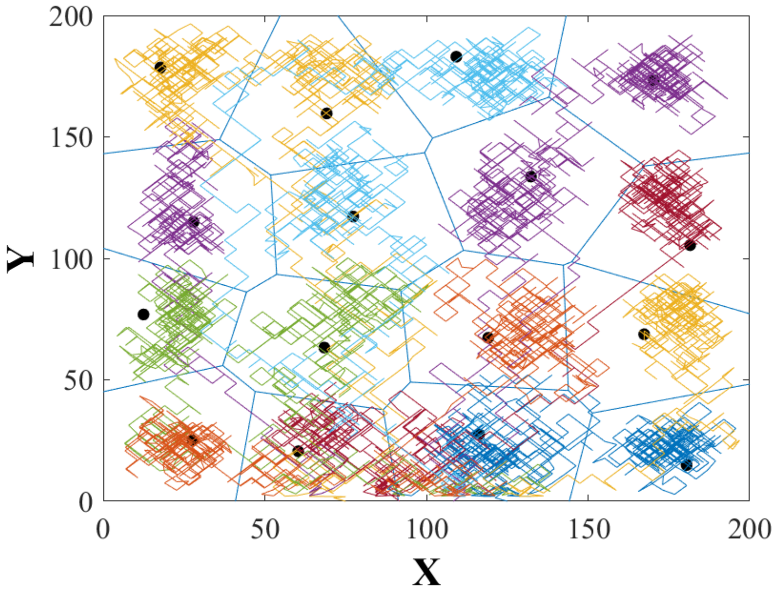

Figure 4.18: Coverage Quality for 3D MBC Algorithm with 16 Agents in X-Y Axes View

Similarly, both Figure 4.17 and Figure 4.18 show the coverage quality generated by the 2D MBC algorithm and the 3D MBC algorithm, respectively. In terms of coverage quality, both algorithms achieve good coverage, with the agents evenly distributed and covering approximately the same area. However, there is a slight degradation in Figure 4.18 due to the misalignment of agents in the row and column. This misalignment can be further resolved by implementing fine-tuning optimization to adjust the gains, as there are a few gains that need to be tuned when scaling is performed. In short, the coverage quality remains consistent when scaling up the system, whether using the 2D MBC algorithm or the 3D MBC algorithm.

Coverage Analysis for 4 Agents and 16 Agents using Both Algorithm.

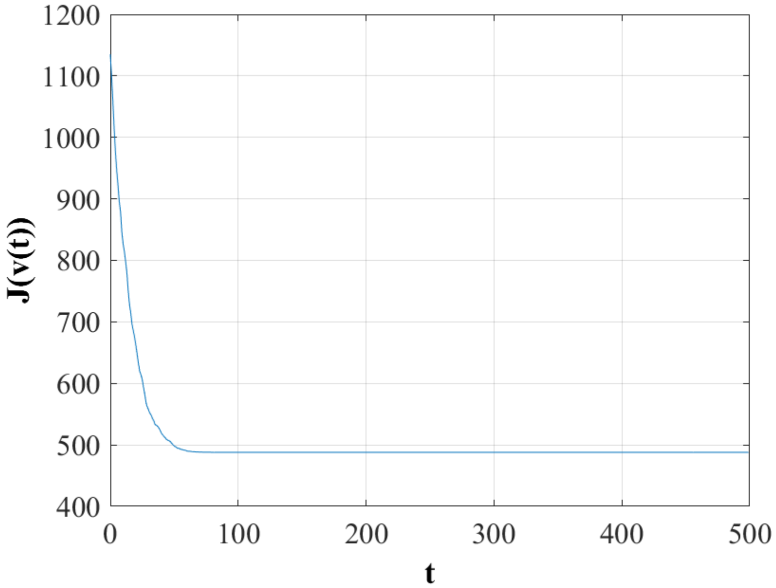

Figure 4.19: 2D MBC Algorithm’s Convergence Graph for 4 Agents

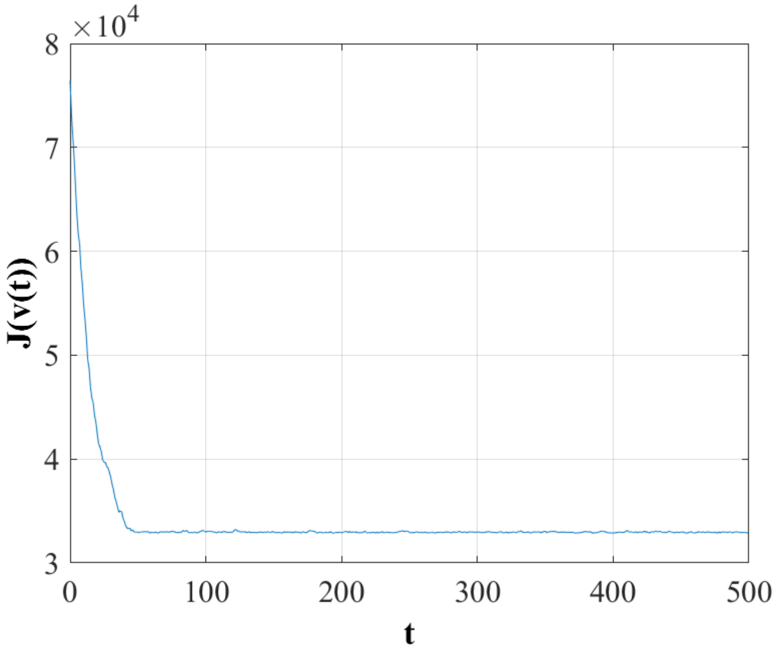

Figure 4.20: 3D MBC Algorithm's Convergence Graph for 4 Agents

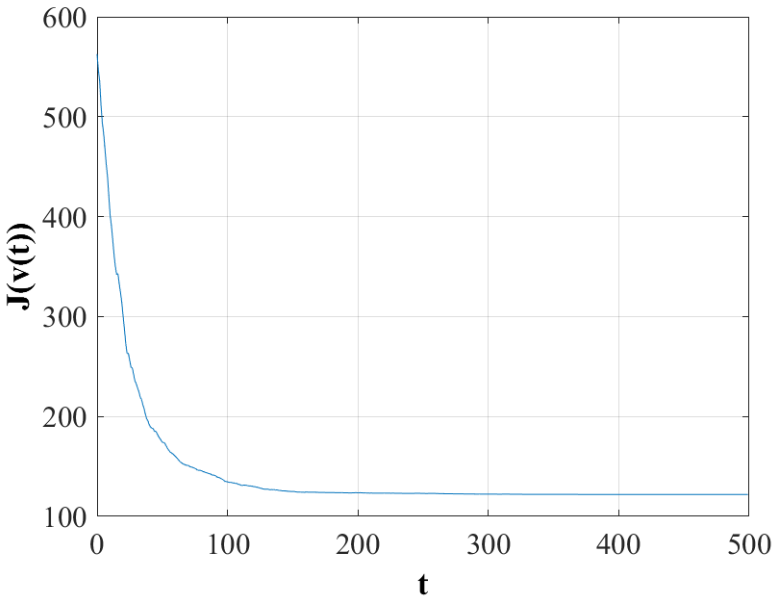

Figure 4.21: 2D MBC Algorithm's Convergence Graph for 16 Agents

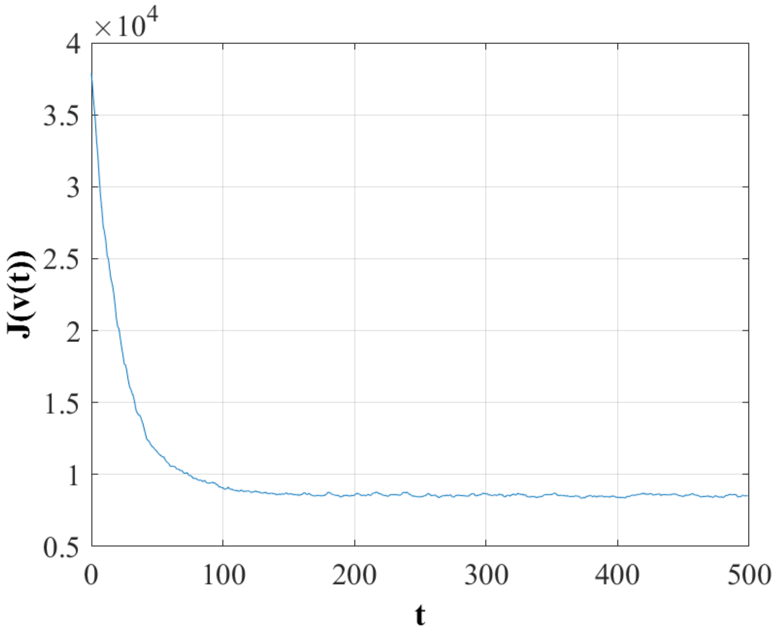

Figure 4.22: 3D MBC Algorithm's Convergence Graph for 16 Agents

Figure 4.19 and Figure 4.20 depict the 2D MBC system and 3D MBC system scaled down to 4 agents, while Figure 4.21 and Figure 4.22 show the 2D MBC system and 3D MBC system scaled up to 16 agents.

In Figure 4.20, there is a shorter convergence time of approximately 45 compared to the convergence time in Figure 4.19, which is approximately 55. Figure 4.20 illustrates a significant decrease in the calculated value for the objective function, making it evident when the system converges and stabilizes.

Figure 4.21 and Figure 4.22 take longer to converge and stabilize. Figure 4.21 demonstrates that the convergence graph from the 2D MBC algorithm reaches convergence and stability around 130 iterations, whereas Figure 4.22 illustrates that the convergence graph from the 3D MBC algorithm achieves convergence and stability at approximately 110 iterations. The system takes longer to converge and stabilize compared to the system with 4 agents and 9 agents.

Although the convergence time for the systems with 4 agents, 9 agents, and 16 agents varies, the scalability of the systems remains as the focus of this section is to compare the scalability of the 2D MBC algorithm and the 3D MBC algorithm. Despite the change in scale, the characteristics of the systems remain consistent, indicating that the performance of the systems in 2D and in 3D are the same. This indicates that the performance for both algorithms is robust and consistent across different agent quantities, demonstrating their scalability in handling varying system input sizes.

<u>Path Length & Computational Time Required for 4 Agents and 16 Agents using Both Algorithm.</u>

```
>> MainMBC

co =

    0.1000

Elapsed time is 1.937151 seconds.

ans =

         0  487.8393

The mean distance traveled by the 4 agents is 87.5983
fx >>
```

Figure 4.23: Path Length and Computational Time Required for 4 Agents Using 2D MBC Algorithm.

```
>> MainMBC
Elapsed time is 1088.038116 seconds.
The mean distance traveled by the 4 agents is 286.6854
fx >>
```

Figure 4.24: Path Length and Computational Time Required for 4 Agents Using 3D MBC Algorithm.

```
>> MainMBC

co =

    0.0100

Elapsed time is 14.420975 seconds.

ans =

         0  121.9169

The mean distance traveled by the 16 agents is 87.482
fx >>
```

Figure 4.25: Path Length and Computational Time Required for 16 Agents Using 2D MBC Algorithm.

```
>> MainMBC
Elapsed time is 3762.609770 seconds.
The mean distance traveled by the 16 agents is 272.3628
fx >>
```

Figure 4.26: Path Length and Computational Time Required for 16 Agents Using 3D MBC Algorithm.

Table 4.2: Summarized Table for System Scalability in Path Length and Computational load

Algorithm	Number of Agents	Computational Load, s	Mean Path Length, m
2D MBC	4	1.9372	87.5983 + 200 = 287.5983
	9	3.8622	87.0964 + 200 = 287.0964
	16	14.4200	87.4820 + 200 = 287.4820
3D MBC	4	1088.0381	286.6854
	9	2046.0367	284.0383
	16	3762.6098	272.3628

The results obtained from Figure 4.23, Figure 4.24, Figure 4.25, and Figure 4.26 were summarized and tabulated in Table 4.2. From Table 4.2, the scalability of the system for 2D MBC and 3D MBC exhibit similar performance when scaling up to 16 agents and scaling down to 4 agents. It can be observed that the mean distance traveled by the agents is approximately the same. This phenomenon occurs because the coverage quality is approximately the same, and the distance traveled by the agents is calculated as the mean, which is the sum of all the real distances traveled by each agent divided by the number of agents. In an ideal case, the mean distance traveled by the system shall have the same value to ensure the coverage performance. This fact is valid only when the environment is obstacle-free, and the dimensions of the environment are the same when scaling up and down.

When the number of agents increases in both 2D MBC algorithm and 3D MBC algorithm, the computational load also increases in an untraceable sequence. Hence, enhancements are required for the scalability of both the 2D MBC algorithm and the 3D MBC algorithm.

Although the scalability for both algorithms need improvement, the primary focus of this section is to compare the scalability between the 2D MBC algorithm and the 3D MBC algorithm. With this aim in mind, the scalability of the 3D MBC algorithm remains consistent even after the transition from the 2D MBC algorithm since both algorithms have the same scalability characteristic.

4.3 Application of 3D Multi-step Broadcast Control

As mentioned earlier, implementing algorithms in a Three-dimensional (3D) space can indeed increase computational complexity and resource requirements. However, transitioning from Two-dimensional (2D) to 3D is still required due to several reasons that support the research extension from 2D to 3D, which can be valuable.

The reason supporting the need for 3D algorithms is realism and applicability. While a Multi Agent System (MAS) with a 2D algorithm may be sufficient for some applications, if the agents involved in the MAS are either Unmanned Aerial Vehicles (UAVs) or Unmanned Underwater Vehicles (UUVs), a 2D algorithm may not be enough to handle applications involving UAVs and UUVs. Real-world scenarios exist in 3D environments, and the development of 3D algorithms will ensure that the study is more aligned with practical applications, increasing its potential in real-life deployments. Unlike in 2D environments where altitude is assumed to be constant, Multi

Unmanned Aerial Vehicle (MUAV) in 3D environments must navigate through varying altitude levels as they encounter obstacles at different altitudes. A 2D algorithm may not adequately account for obstacles above or below the MUAV's flight path, leading to inefficient navigation, path planning, and potential collisions. Moreover, UAVs in the system can achieve optimal path planning as it involves not only horizontal movements but also vertical maneuvers. The real world is a dynamic environment where varying weather conditions, target locations, and terrain exist. A 3D algorithm can control the MUAV to adapt to evolving conditions by adjusting speed, altitude, and direction. A 2D algorithm may lack the flexibility to respond to the dynamic environment effectively.

The algorithms developed for 3D environments are more robust and versatile as they must include additional factors such as the elevation of the agents, altitude of the system, changes in the environment, and obstacles in 3D space. Algorithms for 3D space can offer improved performance and adaptability in challenging environments compared to 2D algorithms, even though the algorithms are more computationally intensive.

Exploring and developing 3D coverage control for MAS opens new avenues for research and innovation. Opportunities and possibilities exist to develop novel algorithms, optimization techniques, and coordination strategies that may not be realizable or necessary in a 2D setting when dealing with the complexities of 3D space. Besides, advancing research in the field of static coverage for MUAV in 3D contributes to the academic community's understanding of complex systems and algorithms.

This study opens a framework of 3D for MUAV in static coverage, and more elements can be added to this framework for greater efficiency and realism that can be deployed in real-life applications. In this research, several functions were implemented to prove the effectiveness of 3D algorithms.

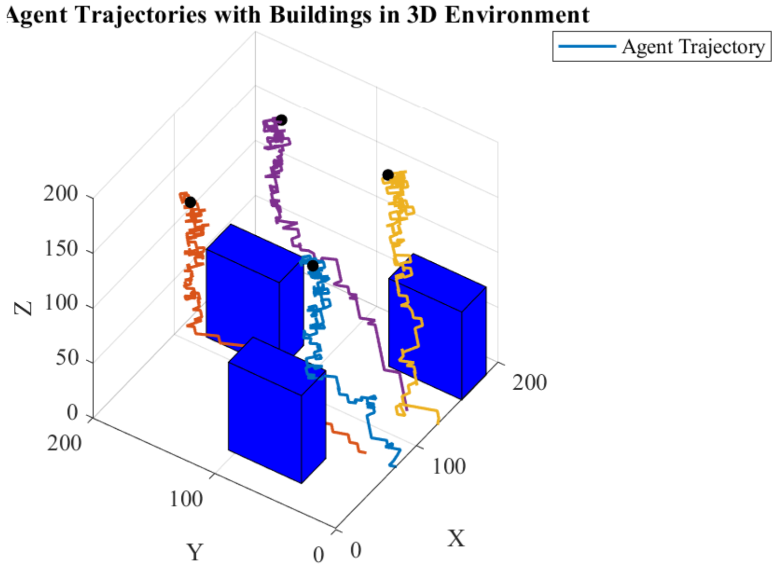

Figure 4.27: 3D MBC Algorithm with Collision Avoidance for 4 Agents

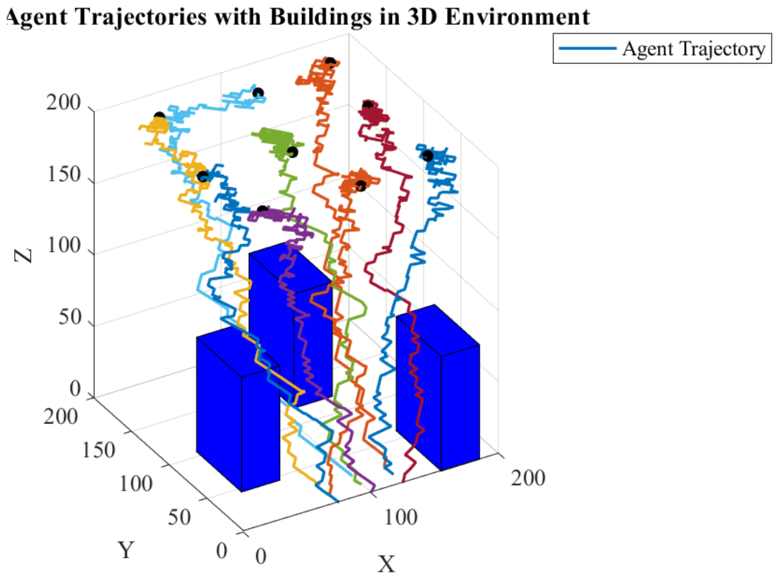

Figure 4.28: 3D MBC Algorithm with Collision Avoidance for 9 Agents

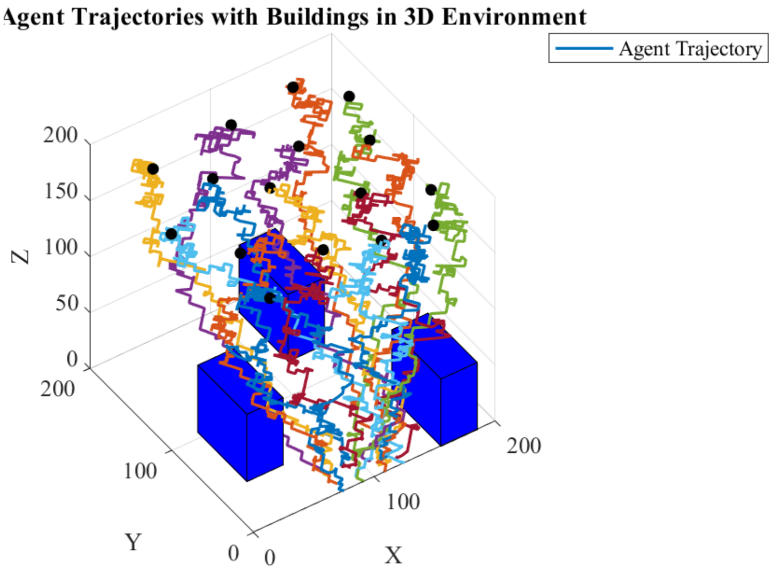

Figure 4.29: 3D MBC Algorithm with Collision Avoidance for 16 Agents.

Figure 4.27, Figure 4.28, and Figure 4.29 depict the agents performing coverage tasks with collision avoidance using the 3D Multi-step Broadcast Control (MBC) Algorithm. The collision avoidance implemented in the 3D MBC is against static obstacles. The rectangular prisms shown in the figures represent buildings and high-rises. The environment map is grid-based, with the map sliced into smaller cubic pieces. Initially, the weightage for each grid of the map is the same. However, a negative value was assigned to the grid covered by the blue rectangular prism to indicate that the area is prohibited, and the agents will not move to that area. A higher positive value assigned to a grid will result in attracting more agents to that area. Hence, the area of interest mentioned in the study of the 2D algorithm often adjusts the density function for the grid to attract more agents to that area. The successful implementation of obstacle avoidance for static obstacles is illustrated in Figures 4.27, 4.28, and 4.29. These figures depict the agents navigating through the environment without colliding with the buildings, which are represented by rectangular prisms.

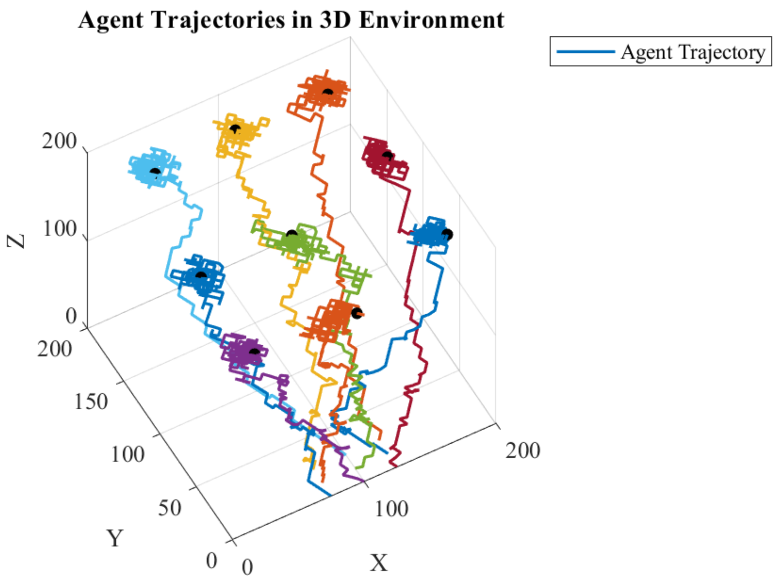

Figure 4.30: 3D MBC Algorithm with Different Altitude Based on Area of Interest.

Figure 4.30 shows the agents completing the coverage task with different heights as results using the 3D MBC algorithm. There are two heights for the agents: 200m and 100m. The agents within the area of interest will have a maximum height of 100m, as the height for the area of interest was preset. If the agents lie within the area of interest, they will have a height of 100m, while others will have a normal height of 200m. For Figure 4.30, the area of interest was set to be in the range of 0-135 for both the X-axis and Y-axis.

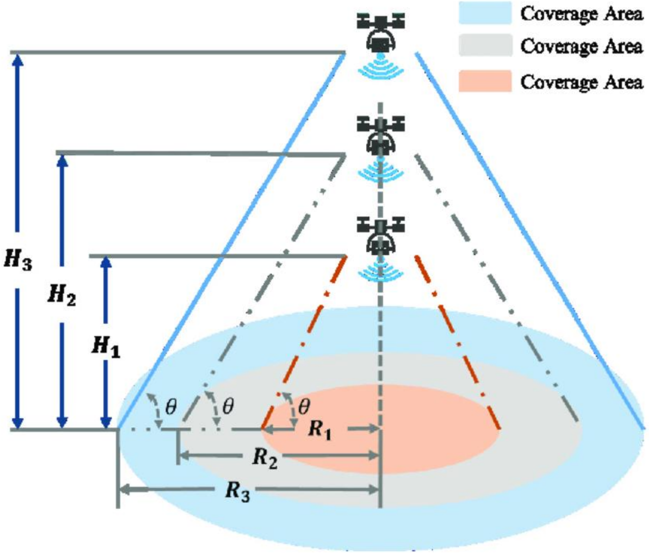

Figure 4.31: Concept of the Coverage Area of UAV (Peng et al., 2022)

In 2D, the concept of the area of interest in the context of multi-agent systems usually focuses on dividing the area and assigning higher density weights to certain regions to attract more agents. However, this approach may not be sufficient for real-life applications, as simply stacking agents in the area of interest does not necessarily lead to significant improvements in the obtained data. While having more agents in the area of interest eliminates coverage gaps, it may not contribute to achieving high-resolution accuracy in the data collected.

The limitations of the area of interest in 2D MAS can be addressed by transitioning from a 2D to a 3D algorithm and incorporating the area of interest concept from 2D, along with the feature shown in Figure 4.30. In a 3D MAS, when agents within the area of interest lower their altitude, they reduce their coverage radius, as illustrated in Figure 4.31. Therefore, implementing the area of interest feature from 2D into a 3D algorithm provides an optimal solution for performing coverage tasks with an area of interest. Lowering the altitude of agents within the area of interest allows for obtaining high-resolution data, while having more agents gather in the area of interest ensures full coverage without gaps.

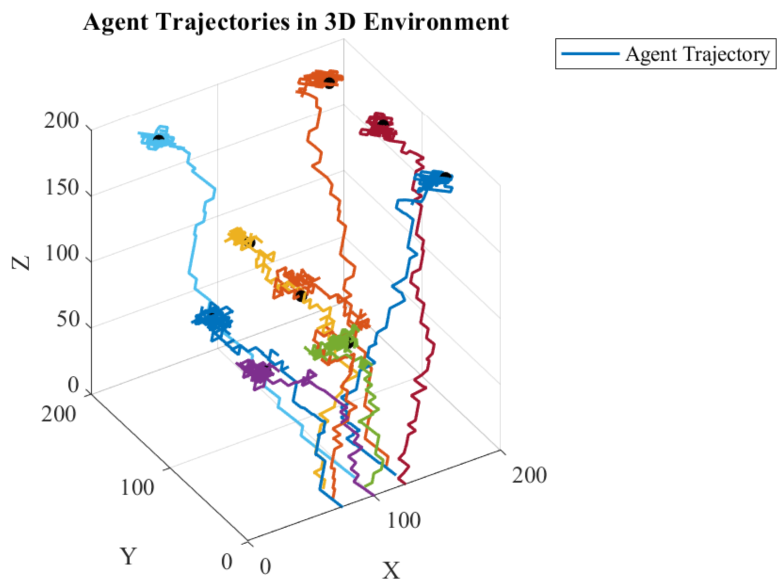

Figure 4.32: 3D MBC Algorithm with Different Altitude Based on Agents' ID.

Similarly to the previously introduced feature, Figure 4.32 illustrates agents with different altitudes based on their ID, using the 3D MBC algorithm. Unlike Figure 4.30, where altitude changes were dependent on the area of interest, in Figure 4.32, altitude changes are based on the agents' ID. Agents with IDs 1, 2, 3, 4, and 5 will have a maximum height of 100m. This feature allows users to directly control the altitude of agents based on their ID to obtain higher-resolution data.

With the successful development of the 3D MBC algorithm, several potential applications can benefit from its deployment to complete various tasks. In search and rescue operations, MUAV are often deployed to locate and assist individuals in remote or hazardous locations. Surveillance capabilities can be enhanced by adjusting the height based on areas with potential signs of life or distress signals. Additionally, dynamically adjusting the height of agents can optimize surveillance capabilities, providing emergency responders with critical information for disaster management. Other applications of the 3D MBC algorithm include border security and surveillance, environmental monitoring, infrastructure inspections, and event monitoring.

Altitude adjustment ensures that MUAV can continue operating effectively despite environmental factors. For example, if a UAV in a MAS experiences misalignment or displacement from its designated working station due to factors like wind flow, systems equipped with altitude adjustment capabilities can execute repositioning maneuvers. This involves altering altitude or horizontal direction to compensate for the displacement. The 3D MBC algorithm is well-suited to these applications, as static coverage objectives can be achieved with altitude adjustments, optimizing surveillance, improving data collection quality, and supporting decision-making in challenging environments.

4.4 Summary

The performance for both Two-dimenstional (2D) Multi-step Broadcast Control (MBC) algorithm and Three-dimensional (3D) MBC algorithm are evaluated in terms of computational load, convergence analysis, path length, coverage quality, and scalability.

Table 4.3: Summarized Table for Performance of Algorithm

Evaluation Parameter	Algorithm	
	2D MBC	3D MBC
Computational Load	☑ (3.8622 s)	☐ (2046.0367 s)
Convergence Analysis	☐ (100 iteration)	☑ (60 iteration)
Path Length	☐ (287.0964 m)	☑ (284.0383 m)
Coverage Quality	☑	☑
Scalability	☑	☑

Table 4.3 summarizes the evaluation performance for both the 2D MBC algorithm and the 3D algorithm. The primary evaluation focuses on comparing the performance of both algorithms to verify that the transitioned 3D MBC algorithm maintains or exceeds its performance level and does not deteriorate in effectiveness. From Table 4.3, there are no changes in performance in terms of coverage quality and scalability for both the 2D and 3D algorithms. It indicates that the algorithm is well developed without degradation. The 2D MBC algorithm demonstrates superior performance compared to the 3D MBC algorithm in terms of computational load. This is attributed to the inherent complexity of calculations in three-dimensional space, where the system must account for an additional axis compared to two-dimensional calculations.

However, the 3D MBC algorithm outperforms the 2D MBC algorithm in convergence analysis and path length. The Multi-Agent Systems (MAS) operating with the 3D MBC algorithm exhibit quicker convergence times, as evidenced by the convergence graph indicating that 3D requires fewer iterations compared to 2D. The convergence analysis attributes the superiority of 3D over 2D to the comparison of iterations needed to accomplish the coverage task, without factoring in computational load. If computational load were included in the analysis of convergence time, the convergence time for 3D would likely be higher than that of 2D.

According to the findings in Table 4.3, the shorter path length observed in the 3D scenario compared to the 2D scenario can be attributed to the fact that the ideal path route in 3D follows Euclidean distance, whereas in 2D it adheres to Manhattan distance. However, despite the shorter path length in 3D, the improvement over 2D is marginal, as the 3D path appears to be zigzag rather than a straight line. This indicates a lack of smooth path planning in the 3D MBC algorithm, which affects the efficiency of the path compared to 2D.

Besides comparing the 2D and 3D algorithms, the application for the 3D MBC algorithm was discussed. In summary, the application scope of the 3D MBC algorithm surpasses that of the 2D MBC algorithm due to the limitations inherent in 2D environments. The ability of the 3D MBC algorithm to adjust altitude expands its applicability to a wider range of real-life scenarios. While the capabilities of the 2D MBC algorithm can generally be replicated by the 3D version, the reverse is not true. In other words, tasks achievable with the 2D MBC algorithm can typically be performed by the 3D MBC algorithm as well, but there are certain functionalities unique to the 3D version that the 2D algorithm cannot replicate.

CHAPTER 5

CONCLUSIONS AND RECOMMENDATIONS

5.1 Conclusions

This project aimed to develop a control algorithm for a Multi-Agent System (MAS) performing coverage tasks in a Three-dimensional (3D) environment. To achieve this, an extensive literature search on existing Multi Unmanned Aerial Vehicle (MUAV) coverage control methods was conducted. Among these methods, the Multi-step Broadcast Control (MBC) algorithm was selected due to its advantages of combining characteristics of both centralized and decentralized coverage. The MBC algorithm minimizes issues associated with centralized and decentralized approaches, with relatively low computational load due to its one-to-all communication nature, eliminating the need for communication between individual agents.

To adapt the MBC algorithm for use in a 3D space, it was necessary to convert it from its original Two-dimensional (2D) design. Thus, the 3D MBC algorithm was developed using MATLAB. In the coverage control task, nine agents were deployed to cover a 200 x 200 x 200 unit area.

The newly developed 3D MBC algorithm was then utilized to assess its performance in comparison to the original 2D MBC algorithm across various metrics such as computational load, convergence analysis, path length, coverage quality, and scalability. The evaluation conducted using these metrics, yielded results indicating that the 3D MBC algorithm surpasses the 2D MBC algorithm in convergence analysis and path length. However, due to the inherent complexities of 3D calculations, the computational load is notably higher. Yet, both the coverage quality and scalability performances remain consistent between the 2D and 3D MBC algorithms. Overall, the transitioned 3D MBC algorithm not only maintains but also exceeds the performance benchmark established by the 2D MBC algorithm.

In summary, the objectives of the project have been achieved, as existing coverage methods were reviewed, and a 3D coverage control algorithm for static coverage tasks was developed. The performance of the developed coverage algorithm was then evaluated in terms of computational load, convergence analysis, path length, coverage quality, and scalability.

5.2 Recommendations for Future Work

There are numerous avenues for enhancing this project, as it establishes a framework for Multi-step Broadcast Control (MBC) algorithms in Three-dimensional (3D). Additionally, there is ample opportunity to incorporate additional elements to enrich the coverage control algorithm. It was noted that there was only a marginal reduction in the mean distance traveled during the evaluation phase, indicating that the achieved results were less than optimal. This issue arises because the path generated by the algorithm lacks smoothness. Incorporating path smoothing techniques into the coverage algorithm can significantly decrease the path length by ensuring more direct travel in Euclidean distance.

Furthermore, the performance of the algorithm has not been tested in a physical environment due to time constraints. The performance indices have only been evaluated in simulations. To evaluate the algorithm's performance in physical environments, the algorithm must be further developed with the Robotics Operating System (ROS). The algorithm's success can be validated through testing in real-world physical environments.

Therefore, future work could involve dynamic obstacles in the environment. Further research on 3D dynamic obstacle avoidance must be done to implement the method into the algorithm. Integrating dynamic obstacle avoidance into the algorithm in 3D can enhance its suitability for real-life scenarios, mitigating the risk of operational failures.

Moreover, this study can be further evolved into blanket coverage. Blanket coverage is the same as static coverage but with the difference that the altitude for every agent might not be the same. Blanket coverage adapts to the terrain by encompassing areas of varying heights, reflecting the typical diversity found in terrain features.

The future work for this project can also involve improving the scalability of the algorithm. Big O notation can be used to analyze the scalability of the 3D MBC algorithm by providing an upper bound on its time or space complexity as the input size grows. The inefficiencies in the algorithm can be addressed and identified, allowing optimization and improvement in scalability.

Furthermore, since computational load is the biggest drawback for the 3D MBC algorithm, the code and calculations for the algorithm can be reviewed, simplified, and unnecessary calculations minimized to streamline processes without sacrificing performance quality. Researching alternative data structures to identify efficient options that can be customized to meet the specific needs of the 3D MBC algorithm is also important. Implementing these enhancements will minimize memory consumption. Additionally, developing an optimized algorithm to integrate into the 3D MBC algorithm to ascertain the optimal value for the gains, ensuring the highest quality outcomes are achieved.

REFERENCES

Anon, *Computational complexity | Definition & Facts | Britannica* [Online]. Available at: https://www.britannica.com/topic/computational-complexity [Accessed: 21 April 2024].

Atınç, G.M., Stipanović, D.M. and Voulgaris, P.G., 2020. A swarm-based approach to dynamic coverage control of multi-agent systems. *Automatica*, 112, p.108637.

Azuma, S.I., Yoshimura, R. and Sugie, T., 2013. Broadcast control of multi-agent systems. *Automatica*, 49(8), pp.2307–2316.

Barrientos, A. et al., 2011. Aerial remote sensing in agriculture: A practical approach to area coverage and path planning for fleets of mini aerial robots. *Journal of Field Robotics*, 28(5), pp.667–689.

Bhattacharya, S., Ghrist, R. and Kumar, V., 2013. Multi-robot coverage and exploration in non-euclidean metric spaces. *Springer Tracts in Advanced Robotics*, 86, pp.245–262.

Bhattacharya, S., Ghrist, R. and Kumar, V., 2014. Multi-robot coverage and exploration on Riemannian manifolds with boundaries. *International Journal of Robotics Research*, 33(1), pp.113–137.

Cabreira, T.M., Brisolara, L.B. and Ferreira Paulo, R., 2019. Survey on Coverage Path Planning with Unmanned Aerial Vehicles. *Drones 2019, Vol. 3, Page 4*, 3(1), p.4.

Cao, Y., Yu, W., Ren, W. and Chen, G., 2013. An overview of recent progress in the study of distributed multi-agent coordination. *IEEE Transactions on Industrial Informatics*, 9(1), pp.427–438.

Chen, S., Li, C. and Zhuo, S., 2017. A distributed coverage algorithm for multi-UAV with average Voronoi partition. *International Conference on Control, Automation and Systems*, 2017-October, pp.1083–1086.

Cho, S.W. et al., 2021. Coverage path planning for multiple unmanned aerial vehicles in maritime search and rescue operations. *Computers & Industrial Engineering*, 161, p.107612.

Choset, H., 2001. Coverage for robotics - A survey of recent results. *Annals of Mathematics and Artificial Intelligence*, 31(1–4), pp.113–126.

Cortés, J., Martínez, S., Karataş, T. and Bullo, F., 2004. Coverage control for mobile sensing networks. *IEEE Transactions on Robotics and Automation*, 20(2), pp.243–255.

Cosma, G. et al., 2017. A survey on computational intelligence approaches for predictive modeling in prostate cancer. *Expert Systems with Applications*, 70, pp.1–19.

Darmaraju, S., Kamal, M.A.S., Shanmugavel, M. and Tan, C.P., 2022. Coverage control of mobile agents using multi-step broadcast control. *Robotica*, 40(9), pp.3290–3305.

Darmaraju, S., Shanmugavel, M., Kamal, M.A.S. and Tan, C.P., 2019. Area Coverage by a Group of UAVs using the Broadcast Control Framework. *IFAC-PapersOnLine*, 52(12), pp.370–375.

Du, Q., Emelianenko, M. and Ju, L., 2006. Convergence of the Lloyd Algorithm for Computing Centroidal Voronoi Tessellations, 44(1), pp.102–119.

Durfee, E.H., 2004. Challenges to Scaling-Up Agent Coordination Strategies. *An Application Science for Multi-Agent Systems*, pp.113–132.

Dürr, H.B., Stanković, M.S. and Johansson, K.H., 2011. Distributed positioning of autonomous mobile sensors with application to coverage control. *Proceedings of the American Control Conference*, pp.4822–4827.

Elfes, A., 1987. Sonar-Based Real-World Mapping and Navigation. *IEEE Journal on Robotics and Automation*, 3(3), pp.249–265.

Elmokadem, T. and Savkin, A. V., 2021. Computationally-Efficient Distributed Algorithms of Navigation of Teams of Autonomous UAVs for 3D Coverage and Flocking. *Drones 2021, Vol. 5, Page 124*, 5(4), p.124.

Fernandez, R. et al., 2021. Emergent Heterogeneous Strategies from Homogeneous Capabilities in Multi-Agent Systems. , pp.491–498.

Fevgas, G., Lagkas, T., Argyriou, V. and Sarigiannidis, P., 2022. Coverage Path Planning Methods Focusing on Energy Efficient and Cooperative Strategies for Unmanned Aerial Vehicles. *Sensors 2022, Vol. 22, Page 1235*, 22(3), p.1235.

Fortune, S., 1992. Voronoi Diagrams and Delaunay Triangulations. *WorldScientific*, pp.193–233.

Gabriely, Y. and Rimon, E., 2002. Spiral-STC: An on-line coverage algorithm of grid environments by a mobile robot. *Proceedings - IEEE International Conference on Robotics and Automation*, 1, pp.954–960.

Galceran, E. and Carreras, M., 2013. A survey on coverage path planning for robotics. *Robotics and Autonomous Systems*, 61(12), pp.1258–1276.

Ghaddar, A. and Merei, A., 2020. EAOA: Energy-Aware Grid-Based 3D-Obstacle Avoidance in Coverage Path Planning for UAVs. *Future Internet 2020, Vol. 12, Page 29*, 12(2), p.29.

Gupta, M. et al., 2019. Monitoring Access to User Defined Areas with Multi-Agent Team in Urban Environments. *International Symposium on Multi-Robot and Multi-Agent Systems, MRS 2019*, pp.56–62.

Hayashi, N., Muranishi, Y. and Takai, S., 2015. Distributed event-Triggered control for Voronoi coverage. *Proceedings of 1st International Conference on Event-Based Control, Communication and Signal Processing, EBCCSP 2015*.

Howard, A., Matarić, M.J. and Sukhatme, G.S., 2002. Mobile Sensor Network Deployment using Potential Fields: A Distributed, Scalable Solution to the Area Coverage Problem. *Distributed Autonomous Robotic Systems 5*, pp.299–308.

Hu, J. et al., 2020. Convergent multiagent formation control with collision avoidance. *IEEE Transactions on Robotics*, 36(6), pp.1805–1818.

Huang, S., Teo, R.S.H. and Leong, W.L., 2018. Review of coverage control of multi unmanned aerial vehicles. *2017 Asian Control Conference, ASCC 2017*, 2018-January, pp.228–232.

Ito, Y., Kamal, M.A.S., Yoshimura, T. and Azuma, S. ichi, 2020. Pseudo-perturbation-based broadcast control of multi-agent systems. *Automatica*, 113, p.108769.

Jayanthi, S., Shaheen, H., Balashivudu, U. and Rani, M.S., 2023. Evolution and Significance of Unmanned Aerial Vehicles. *Unmanned System Technologies*, pp.287–311.

Jennings, N.R., 2000. On agent-based software engineering. *Artificial Intelligence*, 117(2), pp.277–296.

Jiménez, A.C., García-Díaz, V. and Bolaños, S., 2018. A Decentralized Framework for Multi-Agent Robotic Systems. *Sensors 2018, Vol. 18, Page 417*, 18(2), p.417.

Jing, W., Deng, D., Wu, Y. and Shimada, K., 2020. Multi-UAV coverage path planning for the inspection of large and complex structures. *IEEE International Conference on Intelligent Robots and Systems*, pp.1480–1486.

Julia, M., Reinoso, O., Gil, A. and Ballesta, M., 2011, *Behaviour Based Multi-Robot Integrated Exploration*, pp.1–20.

Kennedy, J., Chapman, A. and Dower, P.M., 2019. Generalized coverage control for time-varying density functions. *2019 18th European Control Conference, ECC 2019*, pp.71–76.

Khan, A., Noreen, I. and Habib, Z., 2017. On Complete Coverage Path Planning Algorithms for Non-holonomic Mobile Robots: Survey and Challenges. *JOURNAL OF INFORMATION SCIENCE AND ENGINEERING*, 33, pp.101–121.

Kumar, K. and Kumar, N., 2023. Region coverage-aware path planning for unmanned aerial vehicles: A systematic review. *Physical Communication*, 59, p.102073.

Lee, J.G. and Shim, H., 2022. Design of Heterogeneous Multi-agent System for Distributed Computation. *Lecture Notes in Control and Information Sciences*, 488, pp.83–108.

Li, J., Li, X. and Yu, L., 2018. Multi-UAV cooperative coverage path planning in plateau and mountain environment. *Proceedings - 2018 33rd Youth Academic Annual Conference of Chinese Association of Automation, YAC 2018*, pp.820–824.

Li, W.T. and Liu, Y.C., 2017. Dynamic coverage control for mobile robot network with limited and nonidentical sensory ranges. *Proceedings - IEEE International Conference on Robotics and Automation*, pp.775–780.

Marden, J.R. and Wierman, A., 2008. Distributed welfare games with applications to sensor coverage. *Proceedings of the IEEE Conference on Decision and Control*, pp.1708–1713.

Miao, Z. et al., 2021. Artificial Potential Field Method for Area Coverage of Multi Agricultural Robots. *Communications in Computer and Information Science*, 1469 CCIS, pp.67–76.

Modares, J., Ghanei, F., Mastronarde, N. and Dantu, K., 2017. UB-ANC planner: Energy efficient coverage path planning with multiple drones. *Proceedings - IEEE International Conference on Robotics and Automation*, pp.6182–6189.

Moravec, H.P. and Elfes, A., 1985. High resolution maps from wide angle sonar. *Proceedings - IEEE International Conference on Robotics and Automation*, pp.116–121.

muro, *Multi Agent Coverage Control* [Online]. Available at: http://muro.ucsd.edu/CoverageControl/coveragecontrol.html [Accessed: 13 September 2023].

Nor, M.H.M., Ismail, Z.H. and Ahmad, M.A., 2017. Broadcast control of multi-agent systems with instability phenomenon. *USYS 2016 - 2016 IEEE 6th International Conference on Underwater System Technology: Theory and Applications*, pp.7–12.

Oh, J.S., Choi, Y.H., Park, J.B. and Zheng, Y.F., 2004. Complete coverage navigation of cleaning robots using triangular-cell-based map. *IEEE Transactions on Industrial Electronics*, 51(3), pp.718–726.

Okabe, A., Boots, B., Sugihara, K. and Sung, N.C., *Spatial Tessellations: Concepts and Applications of Voronoi Diagrams*, pp.1–584.

Ota, J., 2006. Multi-agent robot systems as distributed autonomous systems. *Advanced Engineering Informatics*, 20(1), pp.59–70.

Patel, S. et al., 2020. Multi-Agent Coverage in Urban Environments, pp.1–9.

Pedersen, T. and Dyrkolbotn, S.K., 2013. Agents homogeneous: A procedurally anonymous semantics characterizing the homogeneous fragment of ATL. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 8291 LNAI, pp.245–259.

Peng, H., Tsai, A.H., Wang, L.C. and Han, Z., 2022. LEOPARD: Parallel Optimal Deep Echo State Network Prediction Improves Service Coverage for UAV-Assisted Outdoor Hotspots. *IEEE Transactions on Cognitive Communications and Networking*, 8(1), pp.282–295.

Perez-Imaz, H.I.A., Rezeck, P.A.F., MacHaret, D.G. and Campos, M.F.M., 2016. Multi-robot 3D coverage path planning for First Responders teams. *IEEE International Conference on Automation Science and Engineering*, 2016-November, pp.1374–1379.

Schwager, M., Rus, D. and Slotine, J.J., 2009. Decentralized, adaptive coverage control for networked robots. *International Journal of Robotics Research*, 28(3), pp.357–375.

Tan, Y. and Zheng, Z. yang, 2013. Research Advance in Swarm Robotics. *Defence Technology*, 9(1), pp.18–39.

Thrun, S., 1998. Learning metric-topological maps for indoor mobile robot navigation. *Artificial Intelligence*, 99(1), pp.21–71.

Ueda, J., Odhner, L. and Asada, H.H., 2007. Broadcast feedback of stochastic cellular actuators inspired by biological muscle control. *International Journal of Robotics Research*, 26(11–12), pp.1251–1265.

University of Bristol, *What is a Voronoi diagram?* [Online]. Available at: https://www.bristol.ac.uk/maths/fry-building/public-art-strategy/what-is-a-voronoi-diagram/ [Accessed: 13 September 2023].

Wang, H. and Guo, Y., 2008. A decentralized control for mobile sensor network effective coverage. *Proceedings of the World Congress on Intelligent Control and Automation (WCICA)*, pp.473–478.

Wang, Z.J. and Li, W., 2013. A solution to cooperative area coverage surveillance for a swarm of MAVs. *International Journal of Advanced Robotic Systems*, 10.

Wong, S.C., Lee, M. and Macdonald, B., 2002, *Performance Metrics for Robot Coverage Tasks*, pp.7–12.

Wooldridge, M., 2011. An Introduction to Multi Agent Systems.

Xie, J. and Liu, C.-C., 2017. Multi-agent systems and their applications. *Journal of International Council on Electrical Engineering*, 7(1), pp.188–197.

Yang, L. et al., 2016. Survey of Robot 3D Path Planning Algorithms. *Journal of Control Science and Engineering*, 2016.

Zelinsky, A., R.A, J., J.C, B. and Shin'ichi, Y., 2007, *Planning Paths of Complete Coverage of an Unstructured Environment by a Mobile Robot*, pp.1–6.

Zhuang, H., Dong, K., Wang, N. and Dong, L., 2022. Multi-Robot Leader Grouping Consistent Formation Control Method Research with Low Convergence Time Based on Nonholonomic Constraints. *Applied Sciences 2022, Vol. 12, Page 2300*, 12(5), p.2300.

APPENDICES

Appendix A: Main MBC Code

```
1. Initialize parameters:
   - cPJ = 0
   - FrameNo = 1, YY = 200, XX = 200, ZZ = 200, N = 3, M = 3, P = 1
   - Y = linspace(0,25,M), X = linspace(75,125,N), Z = linspace(0,0,P)
   - R = empty array for agents, Record = empty array
   - RnXY = [-1 -1; -1 +1; +1 -1; +1 +1]
   - T = zero vector of appropriate size
   - Define Xgrid, Ygrid, Zgrid using meshgrid
   - QT = 0.25 for each element
   - Tau = 2
   - V = sum of size(Xgrid)

2. Set up mode of simulation:
   - Mode = 2

3. Load random numbers from file "randnum1000"

4. Loop for 200 iterations:
   a. Update parameters a and c based on current iteration
   b. Loop through each agent:
      i. Record current position in agent's trajectory
   c. Optimize using PBC4MPC
   d. Update agent positions based on calculated velocities
   e. Record data for current iteration

5. Plot recorded data

6. Plot agent trajectories, Voronoi diagram, and histograms

7. Calculate total distances traveled by each agent:
   a. Iterate through each agent's trajectory
   b. Calculate distances between consecutive points
   c. Sum up distances to get total distance traveled by each agent
   d. Calculate mean distance traveled by all agents

8. Display mean distance traveled by the agents
```

Appendix B: PBC4MPC Code

1. Loop for each iteration from 1 to Horizon:
   a. Compute dX1 and dY1 using random numbers rx, ry, and constant c
   b. Calculate eX1 and eY1 based on dX1 and dY1
   c. Initialize dGrid matrix and other variables
   d. Loop for each agent:
      i. Compute distance between agent and each point in the grid
      ii. Update dGrid1 with minimum distances for each agent
   e. Calculate JG for each iteration
   f. If Iterate2 is 1, set JGi0 to JG(1)
   g. Define TargetAltitude and b
   h. Initialize arrays for velocities (UxT, UyT, UzT) and weights (Wt, Wt1, dJG1)
   i. Loop for each timestep:
      i. Update weights Wt and Wt1
      ii. Calculate change in JG for each agent
      iii. Compute velocities UxT, UyT, and UzT for each agent
   k. Compute average velocities RUx, RUy, and RUz for all agents
   l. Handle NaN values in velocities
   m. Apply smoothing factor f to velocities
   n. Update positions RX, RY, and RZ for all agents
   o. Bound positions within the environment limits

2. Perform additional smoothing on velocities for each agent

Appendix C: Agent Code

```
class Agent:  # Define the Agent class
    properties:
        ID: integer  # Unique identifier for the agent
        X: float  # X-coordinate of the agent's current position
        Y: float  # Y-coordinate of the agent's current position
        Z: float  # Z-coordinate of the agent's current position
        T: float  # Time at which the agent exists
        Ux: float  # X-component of the agent's velocity
        Uy: float  # Y-component of the agent's velocity
        Uz: float  # Z-component of the agent's velocity
        X2: float  # Additional property for future use
        Y2: float  # Additional property for future use
        Z2: float  # Additional property for future use
        Uxa: float  # Moving average recorded for X-component of velocity
        Uya: float  # Moving average recorded for Y-component of velocity
        Uza: float  # Moving average recorded for Z-component of velocity
        Uxx: float  # Velocity-related property for future use
        Uyy: float  # Velocity-related property for future use
        Uzz: float  # Velocity-related property for future use
        Trj: array  # Empty array intended to store the trajectory or path of the agent over time
    methods:
        function Agent(ID, X, Y, Z, T):  # Constructor method to initialize an Agent object
            Set ID, X, Y, Z, and T properties of the Agent object

# The Agent class represents individual agents with properties such as position, velocity, and trajectory.

# Instances of this class can be created with a unique identifier and initial position coordinates.
```