On Multi-Robot Area and Boundary Coverage

by

Pooyan Fazli

M.Sc., Amirkabir University of Technology (Tehran Polytechnic)

B.Sc., Amirkabir University of Technology (Tehran Polytechnic)

A THESIS SUBMITTED IN PARTIAL FULFILLMENT
OF THE REQUIREMENTS FOR THE DEGREE OF

Doctor of Philosophy

in

THE FACULTY OF GRADUATE AND POSTDOCTORAL STUDIES
(Computer Science)

The University Of British Columbia
(Vancouver)

August 2013

Abstract

Distributed coverage aims to deploy a team of robots to move around a target area to perform sensing, monitoring, data collection, search, or distributed servicing tasks. This thesis investigates three variations of the coverage problem.

First, we address the multi-robot single coverage of a target area. The aim is to guarantee that every accessible point in the area is visited in a finite time. The proposed algorithm supports heterogeneous robots having various maximal speeds, and is robust to robot failure. It also balances the workload distribution among the robots based on their maximal speeds. The obtained results on the coverage time are scalable to workspaces of different sizes, and robots of varied visual ranges.

Second, we tackle the multi-robot repeated coverage of a target area. The objective is to visit all the accessible points of the area repeatedly over time, while optimizing some performance criteria. We introduce four repeated coverage algorithms, and evaluate them under a comprehensive set of metrics including the sum of the paths/tours generated for the robots, the frequency of visiting the points in the target area, and the degree of balance in workload distribution among the robots. We also investigate the effects of environment representation, and the robots' visual range on the performance of the proposed algorithms. The results can be used as a framework for choosing an appropriate combination of repeated coverage algorithm, environment representation, and the robots' visual range based on the particular workspace and the metric to be optimized.

Third, we focus on the multi-robot repeated coverage of the boundaries of a target area and the structures inside it. Events may occur at any position on the boundaries, and the robots are not *a priori* aware of the event distribution. The goal is to maximize the total detection reward of the events. The reward a robot

receives for detecting an event depends on how early the event is detected. To this end, we introduce an online, distributed algorithm and investigate the effects of robots' visual range, communication among the robots, and the event frequency on the performance of the algorithm.

Preface

All the work in this thesis has been conducted under the supervision of Professor Alan K. Mackworth. The list of publications resulted from each chapter of the thesis and the contributions of the author and the collaborators are as follows:

- Chapter 3 (On Multi-Robot Single Area Coverage)

  The author identified and suggested the problem, designed the algorithms, implemented part of the software, conducted the experiments, analyzed the data, and wrote the final manuscripts. Alireza Davoodi, supervised by Philippe Pasquier, also implemented part of the software. This chapter is partly based on the following papers:

  [1] **Pooyan Fazli**, Alireza Davoodi, Philippe Pasquier and Alan K. Mackworth. *Complete and Robust Cooperative Robot Area Coverage with Limited Range*. In Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2010, pp. 5577-5582, Taipei, Taiwan, 2010 [54].

  [2] **Pooyan Fazli**, Alireza Davoodi, Philippe Pasquier and Alan K. Mackworth. *Fault-Tolerant Multi-Robot Area Coverage with Limited Visibility*. In Proceedings of the ICRA 2010 Workshop on Search and Pursuit/Evasion in the Physical World: Efficiency, Scalability, and Guarantees, Anchorage, Alaska, USA, 2010 [55].

  [3] **Pooyan Fazli**, Alireza Davoodi, Philippe Pasquier and Alan K. Mackworth. *Multi-Robot Area Coverage with Limited Visibility*. In Proceedings of the International Conference on Autonomous Agents and Multiagent Systems, AAMAS 2010, pp. 1501-1502, Toronto, Canada, 2010 [53].

[4] **Pooyan Fazli**, Alireza Davoodi, Philippe Pasquier and Alan K. Mackworth. *On Multi-Robot Area Coverage*. In Proceedings of the Seventh Japan Conference on Computational Geometry and Graphs, JCCGG 2009, pp. 75-76, Kanazawa, Japan, 2009 [52].

- Chapter 4 (On Multi-Robot Repeated Area Coverage)

  The author identified and suggested the problem, designed the algorithms, implemented part of the software, conducted the experiments, analyzed the data, and wrote the final manuscripts. Alireza Davoodi also implemented part of the software. This chapter is mainly based on the following papers:

  [1] **Pooyan Fazli**, Alireza Davoodi, and Alan K. Mackworth. *Multi-Robot Repeated Area Coverage*. Autonomous Robots, 34(4):251-276, 2013 [57].

  [2] **Pooyan Fazli**, Alireza Davoodi, and Alan K. Mackworth. *Multi-Robot Repeated Area Coverage: Performance Optimization Under Various Visual Ranges*. In Proceedings of the Ninth Conference on Computer and Robot Vision, CRV 2012, pp. 298-305, Toronto, Canada, 2012 [56].

- Chapter 5 (On Multi-Robot Repeated Boundary Coverage)

  The author identified and suggested the problem, designed the algorithms, implemented the software, conducted the experiments, analyzed the data, and wrote the final manuscripts. This chapter is mainly based on the following papers:

  [1] **Pooyan Fazli**, Alan K. Mackworth. *Multi-Robot Repeated Boundary Coverage*. (Under Review).

  [2] **Pooyan Fazli**, Alan K. Mackworth. *The Effects of Communication and Visual Range on Multi-Robot Repeated Boundary Coverage*. In Proceedings of the IEEE International Symposium on Safety, Security, and Rescue Robotics, SSRR 2012, pp. 1-8, College Station, Texas, USA, 2012 [51].

  [3] **Pooyan Fazli**, Alan K. Mackworth. *Multi-Robot Repeated Boundary Coverage Under Uncertainty*. In Proceedings of the IEEE International Conference on Robotics and Biomimetics, ROBIO 2012, pp. 2167-2174, Guangzhou, China, 2012 [50].

[4] **Pooyan Fazli**, Alireza Davoodi, and Alan K. Mackworth. *Multi-Robot Repeated Boundary Coverage Under Uncertainty*. In Proceedings of the IROS 2011 Workshop on Robotics for Environment Modeling, San Francisco, California, USA, 2011 [49].

- The following papers, written solely by the author, were published in Doctoral Consortium and Graduate Symposiums:

  [1] **Pooyan Fazli**. *On Multi-Robot Area Coverage*. In Proceedings of the Twenty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2010, Atlanta, Georgia, USA, 2010 [46].

  [2] **Pooyan Fazli**. *On Multi-Robot Area Coverage*. In Proceedings of the International Conference on Autonomous Agents and Multiagent Systems, AAMAS 2010, pp. 1669-1670, Toronto, Ontario, Canada, 2010 [48].

  [3] **Pooyan Fazli**. *On Multi-Robot Area Coverage*. In Proceedings of the Canadian Conference on Artificial Intelligence, CanadianAI 2010, pp. 384-387, Ottawa, Canada, 2010 [47].

Table of Contents

List of Tables

List of Figures

Glossary

A Set of Actions

AVP Average Visiting Period

BG Boundary Guards

BWD Balance in Workload Distribution

CC Cyclic Coverage

CDT Constrained Delaunay Triangulation

CLK Chained Lin-Kernighan

Clusters Set of Clusters

Double-MST Double-Minimum Spanning Tree

DT Delaunay Triangulation

ECC Edge-based Clustering Coverage

ESR Expected Segment Reward

$G_{r-vis-cdt}$ Reduced-Vis or Reduced-CDT

$G_{vis-cdt}$ Visibility Graph or Constrained Delaunay Triangulation

IBC Informed Boundary Coverage

MD Set of Minimum Distances

NCC Node-based Clustering Coverage

$\mathcal{O}$ Obstacles

P Workspace Nodes

PG Potential Guard

PoI Points of Interest

PR Path Reward

PSE Probability of Segment Event

$\mathcal{R}$ Set of Robots

SG Static Guards

SP Set of Shortest Paths

SPG Segment Parent Guards

SRAR Segment Reward Accumulation Rate

ST State Transition

STD Population Standard Deviation

STR State Reward

TAVP Total Average Visiting Period

TLV Time of Last Visit

TNMD Trapezoid Node with the Maximum Distance

Tours Set of Tours

TPL Total Path Length

TSP Traveling Salesman Problem

TWVP Total Worst Visiting Period

UBC Uninformed Boundary Coverage

UCC Uninformed Clustering Coverage

$\mathcal{V}$ Speed of the Robots

VA Visual Area of a Guard

VF Visiting Frequency

VG Visibility Graph

VR Visual Range

$\mathcal{W}$ Workspace

WS Workspace Size

WVP Worst Visiting Period

Acknowledgments

It would not have been possible to write this thesis without the help and support of the kind people around me.

First and foremost, I offer my sincerest gratitude to my supervisor, Professor Alan Mackworth, who supported me throughout my study with his deep wisdom, knowledge and patience, whilst allowing me to explore my own ideas. I will always consider it a tremendous honor to have worked with Alan.

I am deeply grateful to my thesis committee members, Professor Will Evans, Professor James Little, and Professor David Poole for their taking the time to answer my questions and their insightful comments and suggestions during writing the thesis. I would also like to thank the members of the examining committee, Professor Maria Gini, Professor Peter Danielson, Professor Robert Woodham, and Professor Tim Salcudean.

I am fortunate to have an invaluable network of supportive, generous and loving friends and relatives in Canada and around the world. First and foremost, I would like to thank Ladan and Farid, without whom I would not have survived in Vancouver. Guity and Mojgan deserve special mention. Gorgin who cheered me up in the most difficult times of my study.

My deepest appreciation goes to Hasti for her endless support, and encouragement.

Last but not least, my parents, Reza and Parvaneh, and my sister Torkan, for giving me their unequivocal support throughout, as always, for which my mere expression of thanks does not suffice.

Chapter 1

Introduction

Given the dynamic and uncertain environments in which future robots will have to work compared with those of the familiar and relatively simple industrial robots, the integration of the advanced physical and cognitive systems required by the next generation of robots is a challenging task. It is not feasible to design a 'universal' robot capable of working within a wide range of applications. AIBO [59], Nao[a], and PR2[b] are examples of the challenges of cost, long product life cycle and limited functionality which will apply to future robots as well. Given these challenges, multi-robot systems may be suitable alternatives to single-robot systems in many real world applications.

The topic of multi-robot systems has been extensively investigated over the past decades [23, 38, 45, 62, 106]. The most common motivations for developing multi-robot system solutions in real world applications are that a single robot cannot deal with the task complexity adequately; the task is spatiotemporally distributed; building several niche, resource-bound robots is easier than building a single powerful robot; multiple robots can support parallelism; and finally, redundancy increases robustness [105, 106].

Distributed *Area/Boundary Coverage*, as a task for multi-robot systems, is a challenging problem in different scenarios such as search and rescue operations [75], planetary exploration [95], landmine detection [3], intruder detection [63, 83],

[a] http://www.aldebaran-robotics.com/en/

[b] http://www.willowgarage.com/pages/pr2/overview

environment monitoring [84], forest monitoring [25, 26, 77, 96], ocean monitoring [123], floor cleaning [70, 102, 103, 104], and so on.

In *Area Coverage*, a team of robots cooperatively visits (observes or sweeps) an entire area *once* or *repeatedly* over time. Another class of problems is *Boundary Coverage* in which the aim of the robot team is to *repeatedly* visit (observes or sweeps) the boundaries of a target area and the structures inside rather than complete coverage of the area. There are two classes of coverage problems:

- ***Single Coverage:*** The robot team aims to guarantee that every accessible point in the area is visited in a finite time.
- ***Repeated Coverage:*** The robot team visits all the accessible points in the target area, or just on the boundaries repeatedly over time, while achieving one of the tasks below:
  - maximizing the frequency of visiting the points.
  - minimizing the sum of the paths/tours generated for the robots.
  - balancing the workload distribution among the robots.
  - detecting the maximum number of events occuring in the target area or just on the boundaries.
  - minimizing the event detection time.

  Visiting the points can be accomplished with uniform or non-uniform frequency.

There is a confusion in the literature regarding the terms *Exploration* and *Coverage*. To clarify the problem definition, we note that in exploration, we have an unknown environment in which a team of robots is trying to build a map of the area together [14, 20, 72, 122, 134, 135]. In contrast, in a coverage problem, the map of the environment may be known or unknown and the team aims to cooperatively visit the whole area or just the boundary with their sensors or physical actuators. In other words, building a map of the environment is not the ultimate aim of the coverage problem.

In this thesis we address three variations of the coverage problem, which all have the following two assumptions in common:

- The workspace, $\mathcal{W} \subset \mathbb{R}^2$, is a given polygon containing polygonal obstacles/structures.
- The robots, $\mathcal{R}$, are assumed to have a 360° field of view and a predefined circular limit of visual range.

1.1 On Multi-Robot Single Area Coverage

In the first part of the thesis, we address the problem of multi-robot single coverage of a target area. We propose an algorithm which guarantees completeness, in that every accessible point in the area is visited in a finite time by at least one of the paths assigned to the robots. It supports heterogeneous robots having various maximal speeds, and supports robustness by handling individual robot failure. The algorithm also balances the workload distribution among the robots based on their maximal speeds. Finally, it is shown that the obtained results on the coverage time of the sample environments are scalable to workspaces of different sizes, and robots of varied visual ranges.

1.2 On Multi-Robot Repeated Area Coverage

In the second part of the thesis, we tackle the problem of multi-robot repeated coverage of a target area. We introduce four repeated coverage algorithms, and evaluate their performance under a comprehensive set of metrics including the sum of the paths/tours generated for the robots, the frequency of visiting the points in the target area, and the degree of balance in workload distribution among the robots. We also investigate the effects of environment representation, and the robots' visual range on the performance of the proposed algorithms.

Given that it is not possible to develop polynomial approximation algorithms, when optimizing each of the metrics mentioned above, and the fact that some of these metrics mutually conflict in the coverage problem, we conduct an extensive experimental analysis to evaluate the proposed algorithms. The results can be used

as a framework for choosing an appropriate combination of repeated coverage algorithm, environment representation, and the robots' visual range based on the particular workspace and the metric to be optimized.

1.3 On Multi-Robot Repeated Boundary Coverage

In the third part of the thesis, we focus on the problem of multi-robot repeated coverage of the boundaries of a target area and the structures inside it. Line-of-sight communication is assumed among the robots. Events may occur at any position on the boundaries, and the robots are not *a priori* aware of the event distribution. The goal is to maximize the total detection reward of the events. The reward a robot receives for detecting an event depends on how early the event is detected.

To this end, we introduce an online, distributed algorithm, in which each robot autonomously learns the event distribution on the boundaries. Based on the policy being learned, each robot then plans in a decentralized manner to select the best path in the target area to visit the most promising parts of the boundary. The performance of the learning algorithm is compared with a heuristic algorithm for the Traveling Salesman Problem, on the basis of the total reward collected by the team during a finite period of time.

We also investigate how robots' visual range, and communication among the robots affect the performance of the robot team in the coverage problem, and how event frequency affects the impact of communication on the robots' performance.

Chapter 2

Background and State of the Art

Several research communities including robotics and agents, sensor networks, operations research, and computational geometry work on variants of the coverage problem.

In sensor networks [34, 61, 112], given an area to be monitored, the aim is to deploy and locate the minimum number of sensor nodes, so that every point in the workspace is covered by at least k sensors [73, 139]. In other similar scenarios, the mobile sensors position themselves in such a way that their density is greater in areas of the workspace with more sensory interest [119, 120].

In operations research, the *Vehicle Routing Problem* has some similarities to the coverage scenarios [65, 128]. In this problem, a number of vehicles deliver goods located at a central depot to a set of geographically dispersed customers. The objective is to minimize the total distance travelled. In the *Vehicle Routing Problem with Time Windows*, the target locations have time windows within which the deliveries (or visits) must be made [17, 18, 37, 125], and in the *Capacitated Vehicle Routing Problem*, the vehicles have limited carrying capacity for the goods that must be delivered [113, 129].

In computational geometry, this problem originates from the *Art Gallery Problem* [99, 130] and its variant for mobile agents, *i.e.*, the *(Multi) Watchman Route Problem* [30, 43, 101]. In the *Art Gallery Problem*, the goal is to position a minimum number of cameras called *guards*, sufficient to see every point of the interior of a gallery (*i.e.*, a simple polygon). In other words, a set of cameras patrols a

gallery, if every point in the gallery is visible to a camera. Cameras may have a limited or unlimited visual range. On the other hand, in the *(Multi) Watchman Route Problem* the objective is to compute routes (closed curves) agents (watchmen) should take, such that any point inside the polygon is visible from at least one of the routes assigned to the agents. Most research done on the above problems in computational geometry deals with simple polygonal spaces without obstacles, unlimited range of agents' vision, single agent scenarios, and scenarios in which a common initial location is determined for all the agents. *Pursuit-Evasion* is another closely related problem studied in both the computational geometry and the robotics communities. In this task, one or more searchers move throughout a given target area in order to guarantee the detection of all the evaders, which can move arbitrarily fast [16, 63, 131]. In *Pursuit-Evasion* scenarios, the searchers do not necessarily cover the entire target area.

In the robotics community, most research in this area is carried out under the rubric of *Area/Boundary/Perimeter Patrolling*. In the rest of this chapter, we examine the literature related to the *Multi-Agent/Robot Patrolling* scenarios.

2.1 Single Area Coverage

In a taxonomy presented by Choset [32], the proposed approaches for area coverage are divided into *offline* methods, in which the map of the environment is known, and *online* methods, in which the map of the environment is unknown. Choset [32] further divides the approaches for area coverage based on the methods they employ for decomposing the area: *Exact Cellular Decomposition*, and *Approximate Cellular Decomposition*.

In *Exact Cellular Decomposition*, the area is divided into a set of non-overlapping regions/cells whose union covers the whole environment. Trapezoidal decomposition [82], Boustrophedon decomposition [31], morse decomposition [1, 2], and Voronoi Diagrams [19, 33, 94, 109] are instances of *Exact Cellular Decomposition*. Thereafter, an adjacency graph is built on the area to connect the neighboring regions/cells in the workspace, and finally the robots plan their paths on the graph to cover the workspace.

In *Approximate Cellular Decomposition* (*e.g.*, grid-based methods), the target

area is divided into cells which are all the same size and shape. However, cells that are partially occluded by obstacles or close to the boundaries are discarded, therefore the union of the cells only *approximates* the target area. Generally, the cell size is determined by the size of the robots' actuator or the sensor range.

The methods based on *Approximate Cellular Decomposition* have limitations since they do not consider the structure of the environment and as a result are unable to handle partially occluded cells or cover areas close to the boundaries in continuous spaces. In contrast, methods based on *Exact Cellular Decomposition* do not suffer those restrictions, so traversing the adjacency graph guarantees covering the whole area. However, they are not efficient in the sense that they do not have a clear policy for moving within the cells, and may have many redundant motions when moving between the cells. The main reason for redundant motions in the methods based on *Exact Cellular Decomposition* is that each cell should be covered entirely before moving to another cell. Sometimes it would be more efficient to cover a cell partially, and return later to finish covering it.

The single area coverage problem has been studied in various ways. Zelinsky *et al.* [137] addressed the offline, complete coverage of grid environments by a single robot. Spires and Goldsmith [124] suggested an offline, decentralized algorithm to cover grid and obstacle-free target areas. The proposed method generates a path, covering the entire area, through computing the *Hilbert Space-Filling Curve* [116] of the workspace. The path is then divided among the robots, guaranteeing a robust and complete coverage of the grid area.

Butler [21] presented a complete, single-robot algorithm for covering an unknown environment with rectilinear boundaries and obstacles. The environment is incrementally decomposed into cells in the shape of a rectangle, where each can be fully covered by back-and-forth motions of a square robot. The work was extended to a centralized multi-robot setting in [22]. Unrestricted communication is assumed among the robots.

Huang [74] studied the single-robot coverage problem, in which the given polygonal workspace is divided into sub-regions, and then each sub-region is swept in a particular direction, such that the number of turns the robot has to perform to cover the entire workspace is minimized. The total path travelled and the cost of moving between the sub-regions is ignored.

Gabriely and Rimon [60] suggested an offline, complete, single-robot coverage algorithm for grid environments. In this approach, the robot traverses the path that circumnavigates the spanning tree built on the grid. Hazon and Kaminka [68] extended [60] to support multiple robots, by partitioning the spanning tree among the robots in a centralized manner. Their algorithm comes in two versions. In the first version, the robots all move in the same direction along the path and no cell is visited more than once. In the second version, the robots are allowed to backtrack over their assigned path, and no cell is visited more than twice. The authors showed that the second version is more efficient and has a better worst-case bound than the first version in terms of the total time it takes to cover the area. Zheng *et al.* [138] addressed the same problem, and proposed an approach outperforming the previous algorithms. The algorithm is based on dividing the spanning tree into sub-trees of balanced weights for the robots of the team. The coverage time of their algorithm is at most eight times greater than optimal. However, the proposed method is not robust to robot failure. Agmon *et al.* [4] discussed a more efficient way of constructing the spanning trees on the grid environments, considering the initial locations of the robots. The proposed algorithm aims at minimizing the maximal distance between every two consecutive robots along the spanning tree.

Mannadiar and Rekleitis [90] addressed the complete coverage of a known workspace in a continuous space by a single robot. The proposed method uses the Boustrophedon decomposition to partition the free space in cells; then, the solution to the *Chinese Postman Problem* is used to calculate the order in which the cells are going to be covered. The cost of moving between the cells is ignored.

Rekleitis *et al.* [114] addressed the multi-robot coverage of an unknown environment in a continuous space. A Boustrophedon decomposition is incrementally built on the workspace. In this method, the robot team is divided into sub-teams of *explorer* and *coverer* robots. The explorer robots determine the boundaries of the detected cells in the free space and the coverer robots then sweep the explored area by back-and-forth motions. The robots operate under the restriction that communication is available only when they are within line of sight of each other. In a later work [115], they proposed an auction-based algorithm for the case that an unrestricted, global communication is assumed among the robots

Luo and Yang [87, 88] employed a neural network to model a given workspace.

Each neuron, corresponding to a cell in the occupancy grid, shows if the cell is occupied (by an obstacle), unclean, or clean. They demonstrated their approach for two [87] and four [88] robots in some simulated environments. While the proposed approach is complete, it is not efficient, since the robots do redundant movements to cover the entire area.

Wagner *et al.* [132] proposed decentralized, ant-based algorithms to cover an unknown grid. Ant robots are simple robots with limited sensing and computational capabilities. Communication among the ant robots is done implicitly using the traces they leave while moving in the area. The proposed algorithms differ in the amount of memory available to every robot. The robots can also adapt to the changes made in the grid. However, these algorithms do not guarantee efficiency.

Koenig and Liu [79] studied the behavior of ant robots for online single and repeated coverage of a grid-imposed target area. The robots implicitly communicate with each other through markings they leave in the area. They compared four navigation methods that are based on real-time search [80], differing in how the markings are updated.

Although there is a wide body of literature for single coverage scenarios, repeated coverage has not received the same attention. Two classes of the repeated coverage problem in the literature are: *1) Area Patrolling*, and *2) Boundary or Perimeter Patrolling (Open or Closed Polylines)*, and each is divided into:

- ***Optimization-based repeated coverage***, in which the team's goal is to optimize some criteria, for example minimizing the average or worst visiting frequency of the points of interest in the target area, minimizing the total path traversed by the robots in the environment, or balancing the workload distribution among the robots.

- ***Adversarial repeated coverage***, in which the team's goal is to maximize the probability of detecting an adversary/event or multiple adversaries/events trying to penetrate/occur in the target area. The main idea behind some of these patrolling strategies is to use non-deterministic, probabilistic algorithms in order to avoid static patrolling patterns which, in an adversarial scenario, could be exploited by the intruders.

2.2 Repeated Area Coverage

2.2.1 Optimization-based Repeated Coverage

Machado *et al.* [89] studied several architectures for repeated coverage of unweighted graphs (*i.e.*, the distance between two adjacent nodes is constant). The goal was to minimize the time interval between two visits to a node. The proposed architectures differ on various parameters such as agent type (reactive or goal-oriented), agent communication (centralized, peer-to-peer, flag-based, or no-communication), coordination scheme (centralized vs. distributed), agent perception (local vs. global), and decision-making (random selection vs. goal-oriented selection). They showed that the *Conscientious Reactive Agents* architecture outperforms the other multi-agent architectures. An agent in *Conscientious Reactive Agents* chooses a node to visit from its neighbourhood with the highest time of being unvisited relative to the agent's own visits rather than all the other agents' visits. No communication is assumed among the agents. The work is generalized to weighted graphs in [10]

Santana *et al.* [118] studied adaptive agents learning to patrol weighted graphs to minimize the time intervals between visits to the nodes. A *Markov Decision Process (MDP)* formalism was used to model the patrolling problem. The challenge was to define a state and action space for each agent individually, and to develop proper models of instantaneous rewards which could lead to satisfactory long term performance. The *Q-Learning* algorithm was used to train the agents, which proved to be computationally expensive.

A problem with some of the existing empirical studies in the field of area patrolling is the lack of a comprehensive population of environment maps in the experiments. In the works by Almeida *et al.* [10] and Machado *et al.* [89] only six maps were used to evaluate the coverage algorithms, two of which have almost 75% similarity. In two other maps called 'circular' and 'corridor', only one representation of the environment (*i.e.*, a chain) is possible due to the structure of the environments. Santana *et al.* [118] also used two very similar maps to evaluate the proposed patrolling algorithm. Moreover, in none of the above patrolling tasks, did the authors provide details on how the graph is built to represent the environment.

They typically assume the existence of a graph which is not a complete model of the environment, but just a rough approximation of it. The proposed architectures also consider the agents as points with no extent or limit on visual range, so the problem dealt with is reduced to a graph exploration/coverage task rather than an area coverage scenario.

Elmaliach *et al.* [42] proposed a centralized algorithm, guaranteeing maximal uniform frequency, in a non-uniform, grid environment. As mentioned above, grid-based representations have limitations in handling partially occluded cells or cover areas close to the boundaries. Also, one of the limitations of the proposed approach is the requirement for a corridor's size in the workspace to be at least twice the size of the robot in order to be covered.

2.2.2 Adversarial Repeated Coverage

Sak *et al.* [117] considered the adversarial multi-agent patrol problem in general graphs. The authors assumed three types of intruders in the environment: a random intruder, an intruder that waits until the patrolling agent leaves a node to penetrate the area through that node, and an intruder that collects statistics on the period between the visits to a random node and predicts the timing of the next safe intrtusion to that node. Some patrolling algorithms were experimentally evaluated by simulation. The results showed that no patrol strategy was optimal for all the possible adversaries.

Ahmadi *et al.* [9] addressed the multi-robot repeated coverage of a known grid target area, in order to detect a set of events of interest. The frequency of the events occurrence in different parts of the environment could possibly be non-uniform. Thus, the robots should visit the points of the workspace with non-uniform frequency. The main contribution of the paper is an online algorithm to partition the area among the robots through a negotiation mechanism. The algorithm is adaptive to non-uniform frequency of events occurrence in the target area.

Guo *et al.* [66] studied a centralized multi-robot system for patrolling continuous environments. The area is partitioned into sub-regions using a *Voronoi Diagram*. Robots are then distributed from their initial positions to their assigned sub-regions to monitor them against the possible intrusions of the adversary agents.

Game theoretic approaches have attracted increasing attention in adversarial patrolling. Much of this work has used a Stackelberg leader-follower game framework to model the interactions between the patrolling agents (leaders) and the intruders (followers). The patrolling agents act based on a randomized strategy, and the intruders choose where to attack after surveillance of this randomized strategy [81]. Two major applications of the Stackelberg game framework are two security schedulers called ARMOR [110], used at the Los Angeles International Airport to randomize allocation of the checkpoints and the canine patrols on the roads towards the airport, and GUARDS [111], used to assist in resource allocation to security activities for airport protection.

Amigoni *et al.* [13] used a leader-follower strategy to determine the optimal policy for a single robot in patrolling a grid environment against an intruder. The cells of the grid have different payoffs/values for the patrolling robot and the intruder. Paruchuri *et al.* [107, 108] addressed the patrolling problem of a set of locations of interest in an adversarial scenario. The patrolling agents use policy randomization to maximize their rewards. The adversary has full knowledge of the agents, and action randomization can deteriorate the adversary's capability to predict and exploit the patrolling agents' actions.

2.3 Repeated Boundary Coverage

2.3.1 Optimization-based Repeated Coverage

Easton and Burdick [39] modeled the coverage problem of a given boundary as a *k-Rural Postman Problem* [40], and used its solution to plan the robots' inspection paths. In the *k-Rural Postman Problem*, the aim is to compute k tours within an undirected connected weighted graph, such that each edge in a required subset of edges is traversed in at least one of the tours, at minimum cost. All the tours start and finish at the same initial node, and all the the k robots involved in the monitoring task start from this common depot location at the beginning of the mission. These paths facilitate complete coverage of the boundary and also balance the workload among the robots of the team. Williams and Burdick [133] extended the previous work for the case that a revision of the original plan may be required

due to the changes in the size of the robot team or the shape of the environment. The proposed algorithm does not guarantee an upper bound on the coverage time of the boundary.

Elmaliach *et al.* [41] addressed the frequency-based patrolling of open polylines (*e.g.*, as in open-ended fences), where the two endpoints of the polylines are not connected. Jensen *et al.* [76] extended Elmaliach *et al.*'s work on patrolling open polylines, with a focus on maintaining the patrol over the long-term. They accomplish this task by replacing the robots having power level below a threshold with some reserve robots. Patrolling open perimeters is challenging because robots must revisit the just visited areas when they reach an endpoint and turn back.

Boardman *et al.* [15] presented a distributed boundary tracking controller for a team of robots. Within this system, the boundary is partitioned into sub-segments, each allocated to a robot, such that the workload is balanced among the robots. They also aimed at minimizing the phase difference between the robots, to limit the size of the gap created between the robots.

Marino *et al.* [91] proposed a decentralized multi-robot approach to patrol both open and closed polylines. A Finite State Automaton was adopted to implement the action selection mechanism.

2.3.2 Adversarial Repeated Coverage

Agmon *et al.* [5] studied patrolling a cyclic boundary, in which the robots' goal is to maximize their rewards by detecting an adversary agent, attempting to penetrate through a point on the boundary unknown to the robots. In their scenario, the full-knowledge adversary knows the location of the robots and the patrol strategy. The adversary also needs a predefined time interval to accomplish the intrusion. The patrolling robots are initially placed uniformly on the perimeter and then move together clockwise or counter-clockwise randomly in order to reduce the probability of penetration. The patrol strategy is determined by a probability value, according to which the robots switch their direction, such that the minimal probability of penetration detection is maximized. They also examined the case of a zero-knowledge adversary [6] and a partial-knowledge adversary [8] in perimeter patrolling. The uncertainty in the robots' perception was investigated in [7], in which the ability to

detect the intruders decreased as the distance grew. Czyzowicz *et al.* [35] addressed the same problem using a team of variable-speed robots.

Girard *et al.* [64] studied a centralized system composed of multiple unmanned air vehicles patrolling a border area. The border is represented as a continuous two-dimensional region divided in sub-regions. Each sub-region is assigned to an air vehicle that repeatedly patrols it with a spiral trajectory to detect the possible intrusions.

2.4 Conclusion

In this chapter, we examined the literature related to the three coverage problems we address in this thesis, namely, *Single Area Coverage*, *Repeated Area Coverage*, and *Repeated Boundary Coverage*.

Chapter 3

On Multi-Robot Single Area Coverage

3.1 Problem Definition and Preliminaries

In this chapter, we address the *Multi-Robot Single Area Coverage* problem with the following specifications:

Assumption 3.1. *The workspace, $\mathcal{W} \subset \mathbb{R}^2$, is a given polygon containing polygonal obstacles.*

Assumption 3.2. *The robots, $\mathcal{R}$, are assumed to have a 360° field of view and a predefined circular limit of visual range.*

Assumption 3.3. *The $|\mathcal{R}|$ robots of the team have various maximal speeds of $v_1, v_2, \ldots, v_{|\mathcal{R}|}$. Without loss of generality, it is assumed that the robots are numbered such that:*

$$0 \leq v_1 \leq v_2 \leq \cdots \leq v_{|\mathcal{R}|}. \tag{3.1}$$

The goal is to compute paths for the robots, such that any accessible point of the workspace is visited in a finite time by at least one of the paths assigned to the robots (Completeness Condition). Our proposed approach improves upon the coverage algorithms based on *Approximate Cellular Decomposition*, which ignore

partially occluded cells or areas close to the boundaries. It is also an improvement on the algorithms based on *Exact Cellular Decomposition*, which do not have a clear policy for moving within the cells, and may have many redundant motions while moving between the cells in the workspace. The proposed decentralized single coverage algorithm supports heterogeneous robots having various maximal speeds, and supports robustness by handling individual robot failure. It also balances the workload distribution among the robots based on their maximal speeds. Finally, the coverage algorithm is designed in a way to overcome the restrictive constraint imposed by the robots' limited visual range.

In the single area coverage problem, not only is the complete coverage of the workspace important but also computing efficient paths for the robots is crucial. Using multiple robots may reduce the *Coverage Time* by dividing the task among the robots. The *Coverage Time* of the workspace is determined as follows:

$$Coverage\ Time = \max_{r \in \mathcal{R}} \frac{\|Path(r)\|}{v_r}, \tag{3.2}$$

where $\|Path(r)\|$ is the length of the path assigned to robot r, and v_r is the speed of robot r in the workspace. We show that the obtained results on the *Coverage Time* of the sample environments are scalable to workspaces of different sizes, and robots of varied visual ranges.

Overview: Our single coverage method is composed of three main steps:

1. A set of *guards*, required to observe a given workspace, is located, considering the limited visual range constraint of the robots (Section 3.2).

2. A graph is built on the *guards* and the nodes of the workspace based on the *Constrained Delaunay Triangulation* (Section 3.3).

3. *Cyclic Coverage Algorithm:* Each robot computes the shortest tour on the *Constrained Delaunay Triangulation* graph, passing through all the *guards* of the workspace, and computes the segment assigned to it on the tour (Section 3.4).

3.2 Locating Guards with Limited Visual Range

In our problem definition, the robots are equipped with panoramic cameras with a 360° field of view. However, the cameras' visual range is limited. The proposed approach initially locates a set of *guards (SG)* within the workspace, according to the robots' limited visual range. These static guards are control points from which the whole workspace can be jointly observed [44]. In other words, if there are as many robots as there are guards, and each robot were stationed on a guard, the entire area would be visible to the robots.

Definition 3.1. Guards, $SG = \{g_1, g_2, \ldots, g_m\}$: A set of points located within the workspace, such that if a robot is positioned on each guard, the entire workspace would be visible to the robots.

To locate the guards, the algorithm decomposes the initial target area, a possibly non-convex polygon with obstacles inside, into a collection of convex polygons using a *Trapezoidal Decomposition* method, and then applies a post-processing approach to eliminate as many trapezoids as possible [136]. In the next step, a divide-and-conquer method [78] is used to subdivide successively each of the resulting convex polygons (trapezoids) into smaller convex sub-polygons until each of them can be covered visually by one guard. Figures 3.4b, 3.5, and 3.6 show the trapezoidation of a sample workspace and the computed guards, assuming varying visual ranges.

3.2.1 Trapezoidal Decomposition of the Workspace

Decomposing complex geometric environments into simpler components is crucial in many applications. For decomposing the initial workspace, $\mathcal{W}$, containing obstacles to a set of convex polygons, a *Trapezoidal Decomposition* method is used. A trapezoid is a convex quadrilateral in which two of the edges are parallel.

Trapezoidation was first addressed by Chazelle and Incerpi [28] and Fournier and Montuno [58]. A randomized algorithm for trapezoidation was later proposed by Seidel [121] as a basis for triangulation of the environments. None of the work mentioned considered general polygonal environments containing obstacles. For our purpose, we use the method suggested by Zalik and Clapworthy [136] to achieve the trazpezoidal decomposition of a workspace with obstacles.

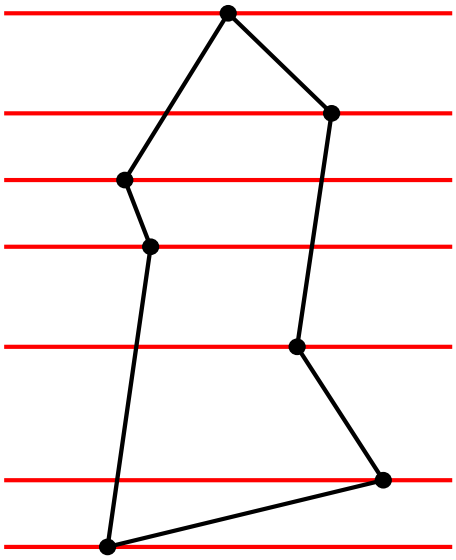


Figure 3.1: *Trapezoidation*

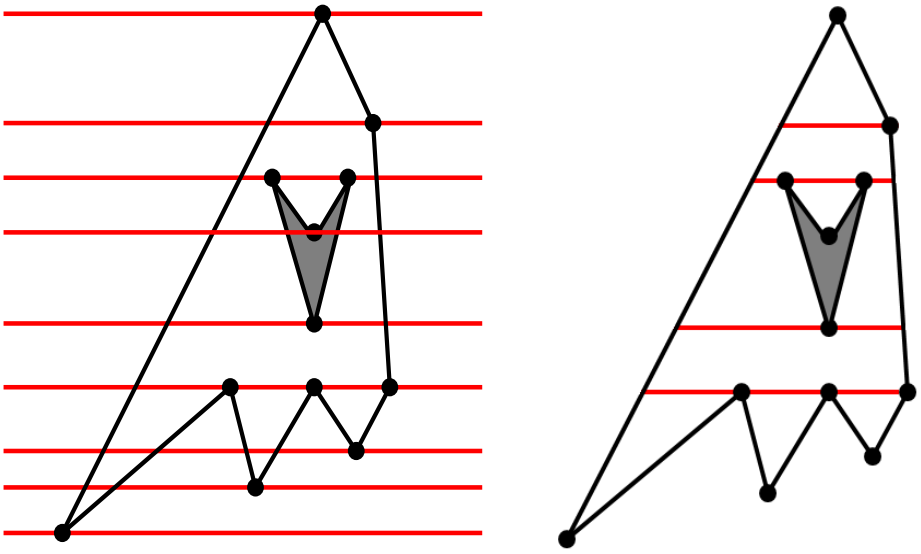
A horizontal trapezoidation of a polygon involves drawing horizontal lines or *slices* through every node of the polygon. Some triangles might also be generated by trapezoidation of a workspace, considering that a triangle is a degenerate trapezoid, in which one of the parallel sides has zero length (Figure 3.1).

Post-Processing: Simplifying the Trapezoidation

The initial trapezoidation algorithm generates more trapezoids than necessary for our application. This is because the algorithm does not look for simplifications to merge the resulting trapezoids. The post-processing step aims at reducing the number of trapezoids generated by the algorithm. This can be achieved by removing the slices lying inside the obstacles or outside the boundaries of the workspace or those not crossing any nodes of the workspace.

Post-processing is more effective in cluttered areas, and since the number of guards located by the algorithm is directly correlated to the number of trapezoids, fewer trapezoids will result in fewer guards.

Figure 3.2b shows the post-processing performed after trapezoidation of the sample polygonal workspace of Figure 3.2a.



(a) *Trapezoidation* **(b)** *Simplifying the Trapezoidation*

Figure 3.2: *Trapezoidation and the Post-Processing Step*

3.2.2 Locating Guards within the Trapezoids

Having decomposed the area into a collection of convex polygons (*i.e.*, trapezoids), a *potential guard (PG)* is computed within each trapezoid. Then, the trapezoid node with the maximum distance (*TNMD*) from the selected potential guard is determined. The potential guard is considered a guard, if it can cover the entire trapezoid, that is, if and only if the distance between *TNMD* and *PG* is smaller than the robots' visual range, otherwise the trapezoid is divided into smaller sub-trapezoids.

Computing a Potential Guard

The following point is selected as a *potential guard* within a trapezoid:

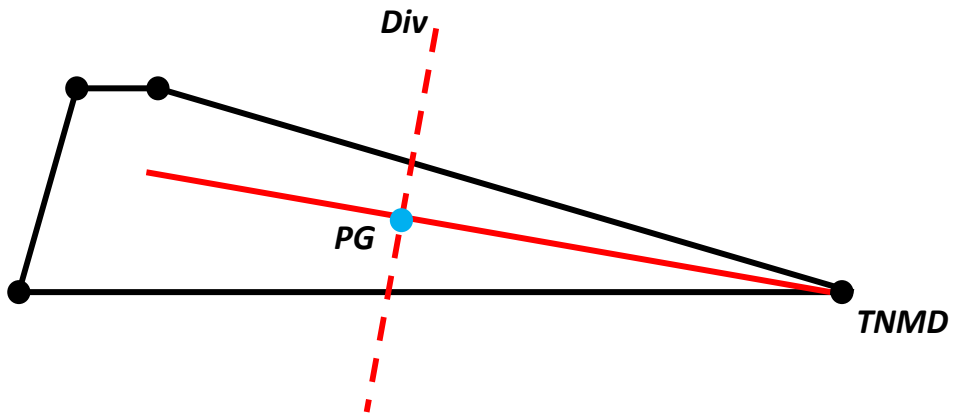


Figure 3.3: *Computing the Potential Guard and the Trapezoid Division*

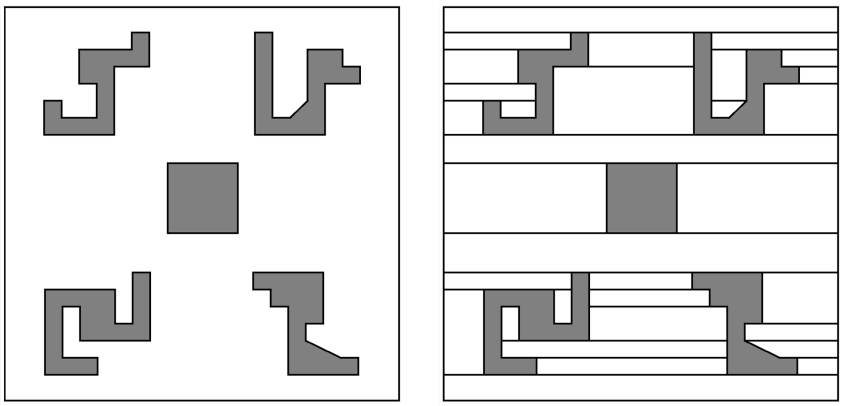
$$PG(x,y) = \frac{\sum_{i=1}^{|E|} \|E_i\| \times M_i(x,y)}{\sum_{i=1}^{|E|} \|E_i\|}, \tag{3.3}$$

where $M_i(x,y)$ is the coordinates of the midpoint of the i-th edge E_i of the trapezoid, and $\|E_i\|$ is the size of the edge E_i . This point is the weighted average of the midpoints of the trapezoid's edges. The idea behind this selection is to bias the *potential guard* towards the longer edges of the trapezoid (Figure 3.3).

Trapezoid Division

The aim of this stage is to generate trapezoids that can be covered with only one guard. If a *potential guard* can not cover the entire trapezoid, the trapezoid is successively divided into smaller pieces, until each can be covered by one *guard*.

At each division step, the goal is to reduce the distance between *PG* and *TNMD* as much as possible. To this end, the line *Div*, that is perpendicular to the line segment connecting *PG* and *TNMD* and passes through *PG* is computed (Figure 3.3). *Div* divides the trapezoid into two sub-trapezoids, and the same procedure is recursively performed on each of the derived sub-trapezoids.



(a) *Original Map* **(b)** *Trapezoidation*

Figure 3.4: *Trapezoidation of a Sample Workspace*

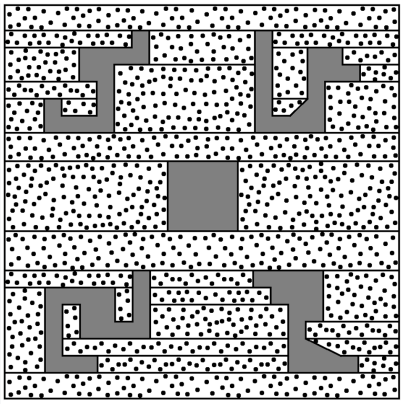
Figure 3.4b shows the trapezoidation of the sample workspace of Figure 3.4a. The size of the workspace is $10m \times 10m$. Figure 3.5 and Figure 3.6 also show the computed guards on the workspace, assuming varied visual ranges of $0.25m$, $0.5m$, $0.75m$, $1m$, and $1.5m$.

3.3 Building the Graph

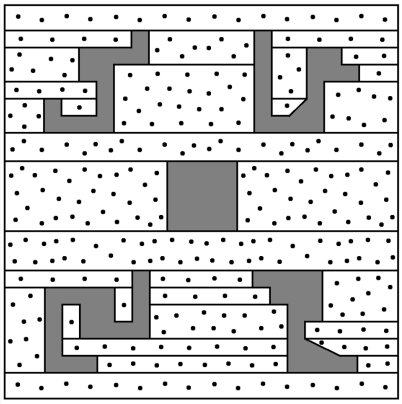
In this section, we investigate a graph structure for environment representation based on the *Constrained Delaunay Triangulation (CDT)*. Having located the static guards in the previous step, the *Constrained Delaunay Triangulation* is then built on the computed guards and the nodes of the workspace.

3.3.1 Delaunay Triangulation

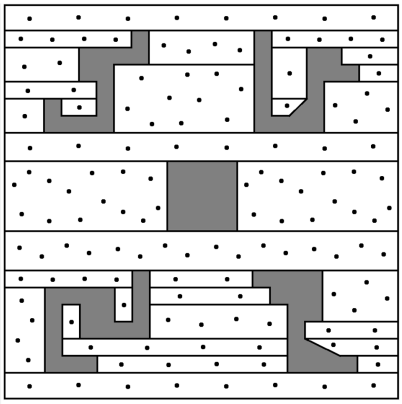
The *Delaunay Triangulation (DT)* of a set of nodes, N, in the Euclidean plane is a triangulation, $DT(N)$, such that the circumcircle of any triangle in the triangulation $DT(N)$ does not contain any nodes other than the three that define it *(Delaunay Condition)* [85]. For instance, in Figure 3.7, the triangles A and B satisfy the *Delaunay Condition*, but in Figure 3.8, the triangles do not satisfy the condition. The *Delaunay Triangulation* corresponds to the dual graph of the *Voronoi Tessellation* [98].



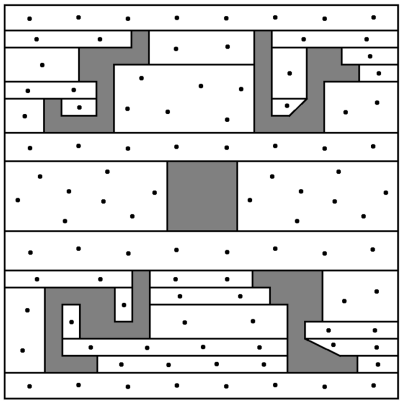
(a) *Visual Range = 0.25m*



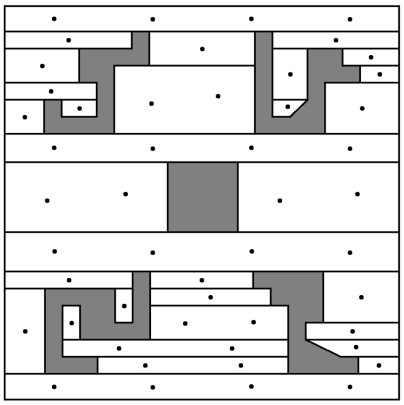
(b) *Visual Range = 0.5m*



(c) *Visual Range = 0.75m*

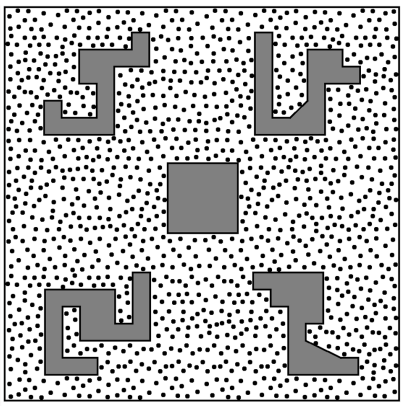


(d) *Visual Range = 1m*

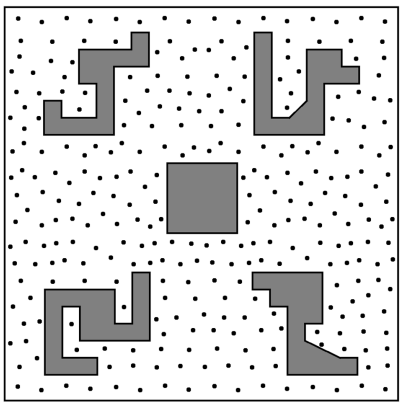


(e) *Visual Range = 1.5m*

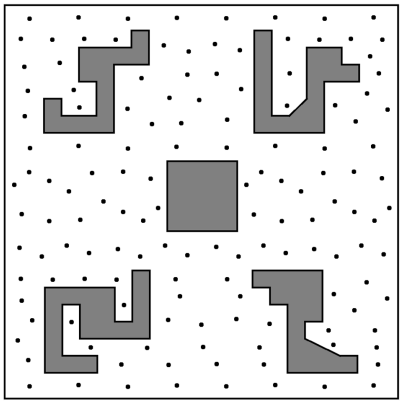
Figure 3.5: *Trapezoidation + Computed Guards*



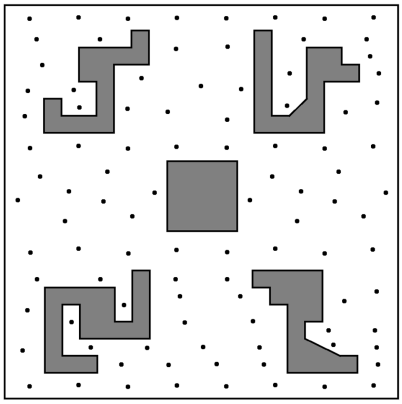
(a) *Visual Range = 0.25m*



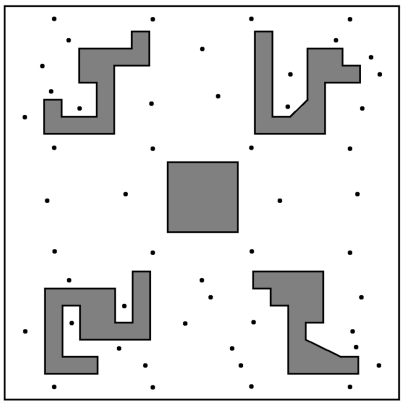
(b) *Visual Range = 0.5m*



(c) *Visual Range = 0.75m*



(d) *Visual Range = 1m*



(e) *Visual Range = 1.5m*

Figure 3.6: *Computed Guards*

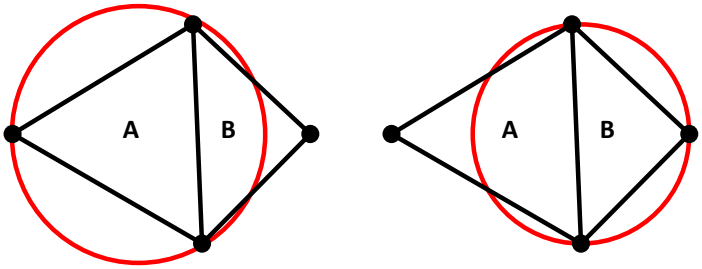


Figure 3.7: *Triangles A and B Satisfy the Delaunay Condition*

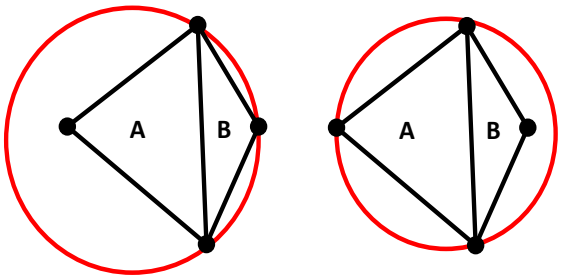


Figure 3.8: *Triangles A and B Do Not Satisfy the Delaunay Condition*

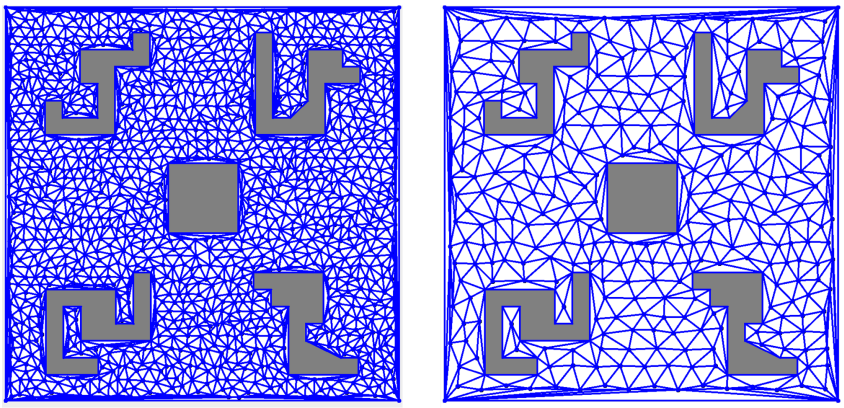
3.3.2 Constrained Delaunay Triangulation

The *Constrained Delaunay Triangulation (CDT)* is a variant of the standard *Delaunay Triangulation* in which a set of pre-specified edges (in our case, the edges of the workspace) must lie in the triangulation [29]. Often, a *Constrained Delaunay Triangulation* contains triangles that do not satisfy the *Delaunay Condition*, *i.e.*, the circumcircle associated with each triangle may contain nodes in its interior, other than the three nodes of the triangle.

Figure 3.9 illustrates the *Constrained Delaunay Triangulation* built on the sample workspace and the computed guards based on varying visual range of the robots.

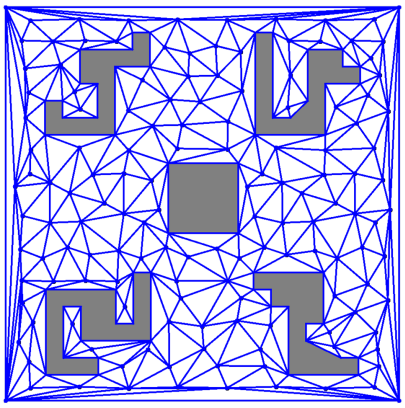
3.4 Cyclic Coverage

Cyclic Coverage (Algorithm 1*)* computes the shortest tour passing through all the computed guards in the workspace using the *Chained Lin-Kernighan* algorithm

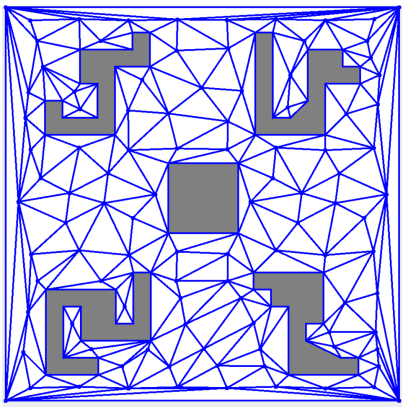


(a) *Visual Range = 0.25m*

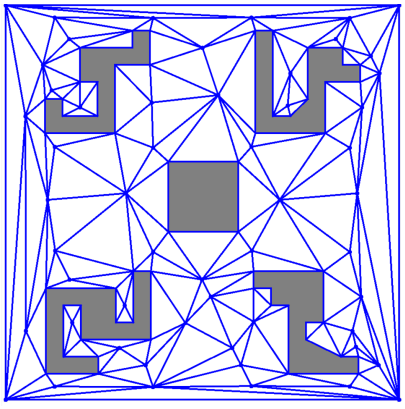
(b) *Visual Range = 0.5m*



(c) *Visual Range = 0.75m*

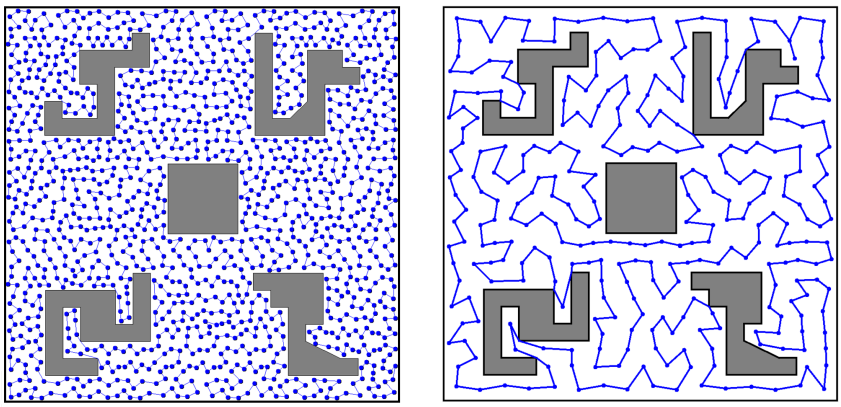


(d) *Visual Range = 1m*



(e) *Visual Range = 1.5m*

Figure 3.9: *Constrained Delaunay Triangulation*



(a) *Visual Range = 0.25m*

(b) *Visual Range = 0.5m*

(c) *Visual Range = 0.75m*

(d) *Visual Range = 1m*

(e) *Visual Range = 1.5m*

Figure 3.10: *Tours Built on the Sample Workspace*

Algorithm 3.1: Cyclic Coverage

Input:
$G_{cdt}(V_{cdt}, E_{cdt})$, where $V_{cdt} = SG \bigcup P$ `/* CDT */`
$SG = \{g_1, g_2, \ldots, g_m\}$ `/* Static Guards */`
$P = \{p_1, p_2, \ldots, p_n\}$ `/* Workspace Nodes */`
$|\mathcal{R}|$ `/* Number of Robots */`
$\mathcal{V} = \{v_1, v_2, \ldots, v_{|R|}\}$ `/* Speed of the Robots */`

Output:
A tour, *dTour*, passing through all the guards of the *CDT* graph, and distributed among the robots.

```
1 begin
2 |  tour ⟵ BuildTour(G_cdt)
3 |  dTour ⟵ DistributeRobots(tour, |R|, V)
4 |  return dTour
5 end
```

(Section 3.4.1) *(line 2)*. The input of the *Chained Lin-Kernighan* algorithm is the distance matrix of the guards in the *CDT* graph. The matrix represents the shortest path distances between all pairs of guards of the *CDT* graph, and these paths can include any nodes of the workspace as well. The input to the *Chained Lin-Kernighan* algorithm should be a complete graph and the distance matrix is indicative of a complete graph, even though the *CDT* graph may not be complete.

Figure 3.10 illustrates the tours built on the sample workspace under varying visual ranges.

If there are $|\mathcal{R}|$ robots of various maximal speeds of $v_1, v_2, \ldots, v_{|\mathcal{R}|}$, the tour built on the workspace is then divided in proportion to the speeds of the robots *(line 3)*. Without loss of generality, we assume that the robots are numbered such that:

$$0 \leq v_1 \leq v_2 \leq \cdots \leq v_{|\mathcal{R}|}. \tag{3.4}$$

Then, we can partition a tour of size $\|dTour\|$ into $|\mathcal{R}|$ segments, such that the length of the i-th segment $dTour_i$ equals:

$$\|dTour_i\| = \frac{v_i}{v_1 + v_2 + \cdots + v_{|\mathcal{R}|}} \times \|dTour\| . \tag{3.5}$$

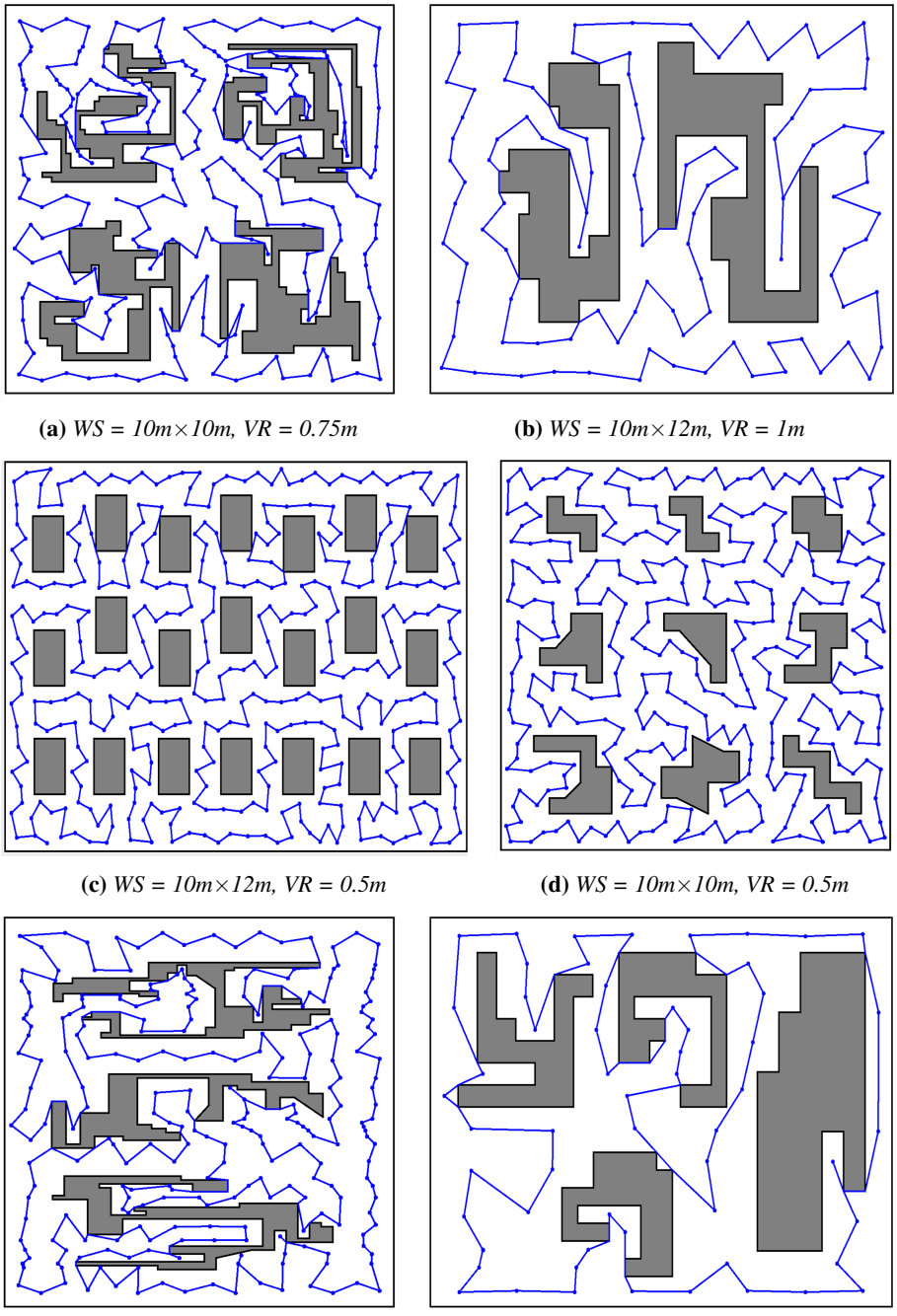
Cyclic Coverage balances the workload distribution among the robots based on their maximal speeds. According to equation 3.5, if the robots have equal speeds, they position themselves equidistantly around the tour and traverse the sub-path assigned to them.

Since the map of the workspace, the size of the robot team, and their maximal speeds are known to the robots, each robot can compute the shortest tour in the workspace and the segment assigned to it on the tour in a decentralized manner, and no communication is needed among the robots.

3.4.1 Chained Lin-Kernighan

Chained Lin-Kernighan (CLK), a modification of the *Lin-Kernighan* algorithm [86], is generally considered to be one of the best heuristic methods for generating *optimal* or *near-optimal* solutions for the *Euclidean Traveling Salesman Problem* [11]. Given the distance between each pair of a finite number of nodes in a *complete* graph, the *Traveling Salesman Problem (TSP)* is to find the shortest tour passing through all the nodes exactly once and returning to the starting node [12].

Lin-Kernighan is a *local search* algorithm [71] and a generalization of the *k-opt* algorithm [27]. A *k-opt* algorithm explores all the *TSP* tours which can be obtained by removing k edges from the original tour and adding k different edges such that the resulting tour is feasible. In order to improve efficiency, Lin and Kernighan introduce a variable *k-opt* algorithm, which adaptively decides at each iteration what value of k to use [86]. Given the computation time limit, the process is repeated by generating new initial tours and applying the *Lin-Kernighan* algorithm to possibly find a tour shorter than the best one thus far. Martin *et al.* [92, 93] suggest that instead of repeatedly starting from new tours, which is inefficient, the alternative is to perturb the *Lin-Kernighan* tour, and then reapply the algorithm. If this leads to a shorter tour, then discard the old tour, and start with the new one. Otherwise, continue with the old tour and perturb it again. The implementation of the *Chained Lin-Kernighan* method which we use in our study is based on the *Concorde TSP* library [11].



(a) *WS = 10m×10m, VR = 0.75m*

(b) *WS = 10m×12m, VR = 1m*

(c) *WS = 10m×12m, VR = 0.5m*

(d) *WS = 10m×10m, VR = 0.5m*

(e) *WS = 10m×10m, VR = 0.75m*

(f) *WS = 10m×12m, VR = 1.5m*

Figure 3.11: *Computed Tours, WS: Workspace Size, VR: Visual Range*

3.4.2 Example Solutions

Figure 3.11 shows the shortest tours computed on different environments assuming different visual ranges of the robots. As shown in the examples, the amount of redundant movement to cover the workspace is minimized, except within the narrow corridors and closed spaces where the robots have no other option but cover the area twice by moving back and forth inside the space.

3.4.3 Completeness

The proposed approach covers every accessible point in the workspace in a finite time. As mentioned earlier, the computed static guards are points from which the whole target area can be jointly observed, considering the limited visual range constraint of the robots. Hence, traversing the shortest tour created on the workspace by the robots leads to visiting all the static guards and therefore the full coverage of the workspace.

3.4.4 Robustness

Robot failure during execution can jeopardize the completion of the area coverage task. By failure, we mean that the robot is not capable of moving anymore, and by fault tolerance, we refer to the ability of the team to respond to individual robot failure that may occur at any time during the coverage mission.

Our approach guarantees robustness through a simple mechanism. Since the robots are moving on a cycle, each robot has $|\mathcal{R}| - 1$ robots behind it, so when a robot fails, the first operating robot behind it can take over covering the segment associated with the failed robot. There is no need to reconfigure and reposition the robots around the cycle after a robot fails. The algorithm guarantees covering the entire workspace in a finite time even if $|\mathcal{R}| - 1$ robots fail. This assumes the failed robots do not block the path of the other robots, and it is capable of communicating its failure with the other robots.

Stages of the Algorithm		Time Complexity
Locating Guards		$O(n^2 \log_2 n)$
Building Graph	Constrained Delaunay Triangulation	$O((n+m)\log(n+m))$
Cyclic Coverage	Chained Lin-Kernighan	$O(m^{2.2})$[a]

Table 3.1: *Time Complexity of Different Stages of the Single Coverage Algorithm. n: Number of Workspace Nodes, m: Number of Guards*

3.5 Complexity Analyses of the Algorithm

The basic version of the coverage problem with just one robot with unlimited visual range operating in a simple polygon without obstacles has an exact polynomial time solution [24, 126]. But, extending the problem to include obstacles in the workspace, to multiple robots, or to limited visual range, makes the corresponding decision problems *NP-hard* [97]. Furthermore, it is not even possible to develop polynomial approximation algorithms, when minimizing the *Coverage Time* [100].

The time complexity of the stages of the proposed single coverage algorithm is shown in Table 3.1.

3.6 Evaluation and Experimental Simulations

We have developed a simulator to test the algorithm in different scenarios. The simulator can support different numbers of robots in the target area, different visual ranges for the robots, and varying degrees of clutter in the environment. A random map generator was also developed as a part of the simulator which extends a library [127] to build polygonal environments with free form polygonal obstacles within the space.

We consider three types of environments in the experiments: sparse ($0-25\%$ cluttered), semi-cluttered ($25-50\%$ cluttered), cluttered ($50-75\%$ cluttered). Ten different maps are used in the experiments for each of the three environment types (30 in total). The clutter percentage of an environment is the ratio of the area of the obstacles to the whole target area: $\frac{Obstacles}{Obstacles+Free\ Space}$. Figure 3.12 rep-

[a]This entry is based on experimental results [11, 86]. The worst case time complexity of the *Chained Lin-Kernighan* algorithm is apparently not available in the literature [69].

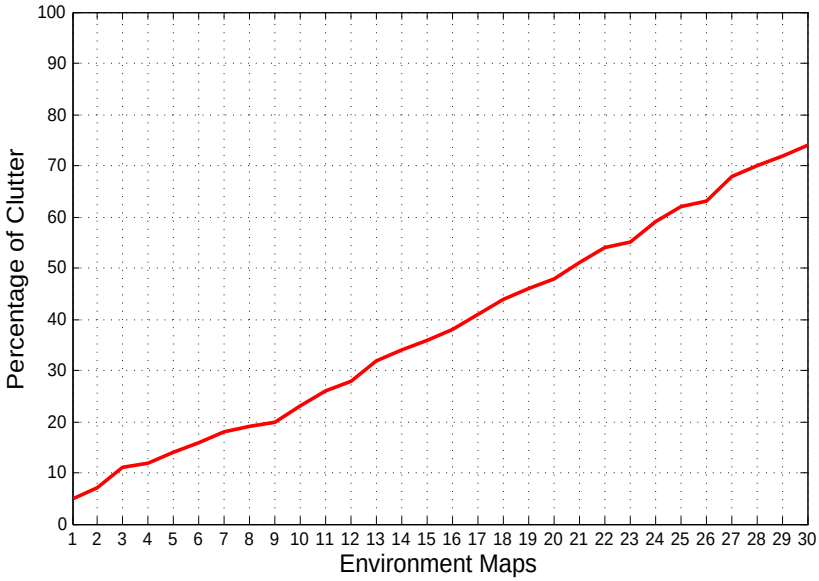


Figure 3.12: *Percentage of Clutter in Each Map*

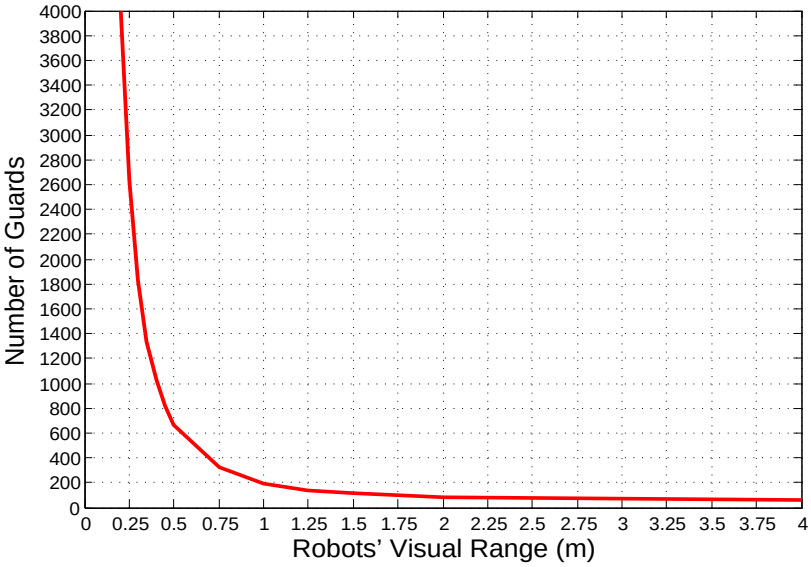


Figure 3.13: *Average Number of Guards Computed on the Selected Maps as a Function of the Robots' Visual Range*

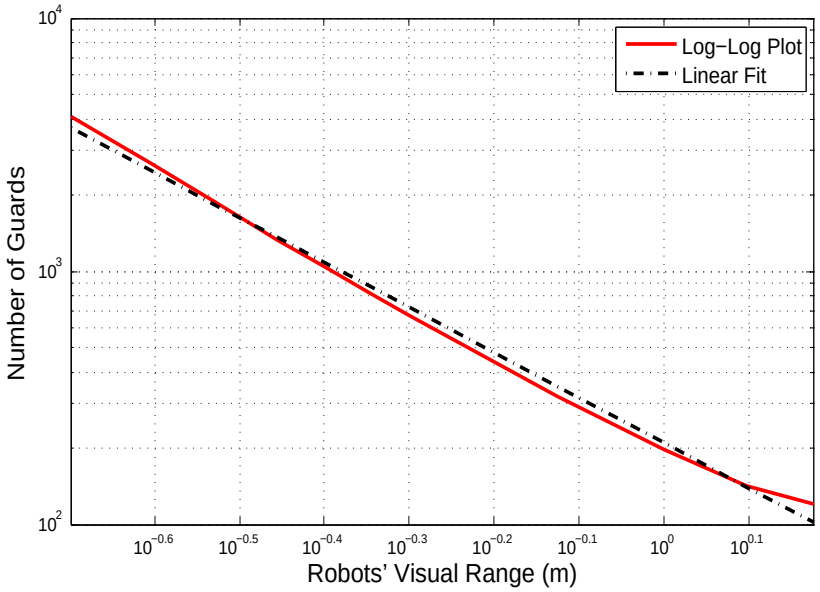


Figure 3.14: *The Log-Log Plot of Figure 3.13 with a Linear Fit*

resents the percentage of clutter in each of the maps used in the experiments (sorted from the least cluttered to the most cluttered). The size of the environments is $15m \times 15m$. The maps can be accessed at: http://www.cs.ubc.ca/~pooyanf/Research/CoverageMaps.zip.

Figure 3.13 shows the average number of guards computed on the 30 maps used in the experiments under visual ranges up to $4m$. As shown in the figure, the number of computed guards decreases with the increase in the visual range of the robots, and beyond a visual range of $1.5m$, the number of guards becomes essentially fixed. Figure 3.14 shows the corresponding log-log plot and the best linear fit ($y = 209.47 \times x^{-1.784}$) for visual ranges up to $1.5m$.

Based on this distribution and some initial experiments to find the proper interval between the visual ranges, so that the difference of the results between two subsequent visual ranges is shown more clearly, we chose visual ranges $0.25m$, $0.5m$, $0.75m$, $1m$ and $1.5m$ to evaluate the coverage algorithm.

Figure 3.15 shows the number of guards computed on each of the selected maps under the chosen visual ranges for the robots. As shown in the figure, for each of

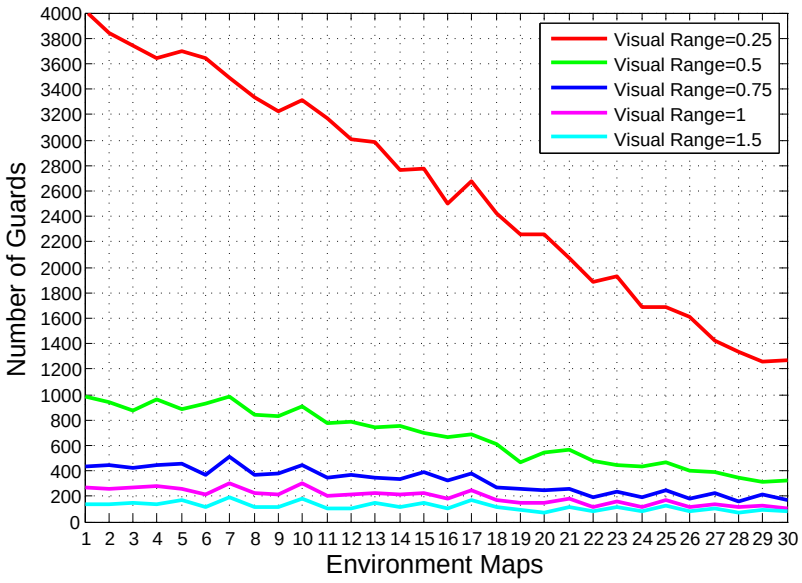


Figure 3.15: *Number of Guards Computed on Each Selected Map as a Function of the Chosen Visual Ranges*

the visual ranges, there is a negative correlation between the number of computed guards and the percentage of clutter in the sample workspaces, that is, as the clutter of the environment increases, the number of guards declines in general:

$$Number\ of\ Guards \propto -Percentage\ of\ Clutter. \tag{3.6}$$

In other words, there is a positive correlation between the free space of the workspace and the number of computed guards:

$$Number\ of\ Guards \propto Free\ Space. \tag{3.7}$$

3.6.1 Coverage Time

Figure 3.16 shows the *Coverage Time* of the environments by a team of 4 robots under visual ranges of $0.25m$, $0.5m$, $0.75m$, $1m$ and $1.5m$. The robots are assumed to have a unit speed of $1m/sec$. The figure shows that as the robots' visual range and

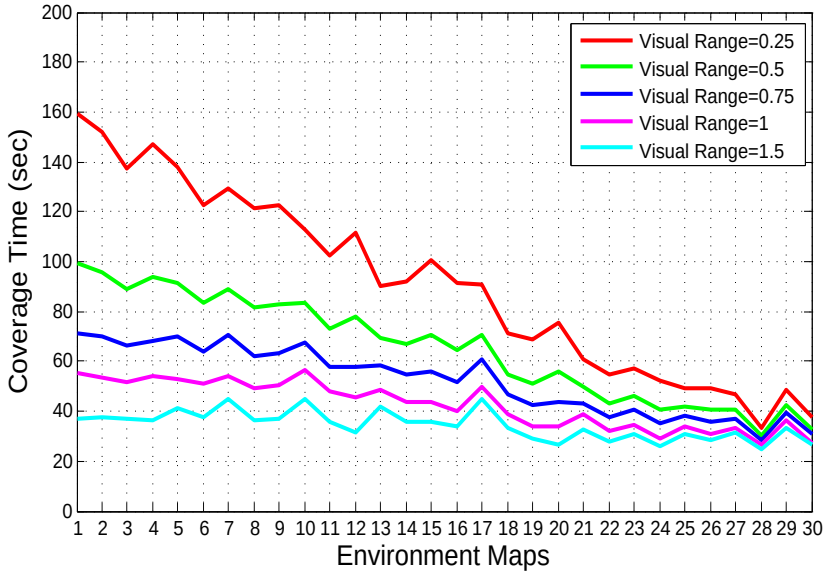


Figure 3.16: *Visual Range vs. Coverage Time*

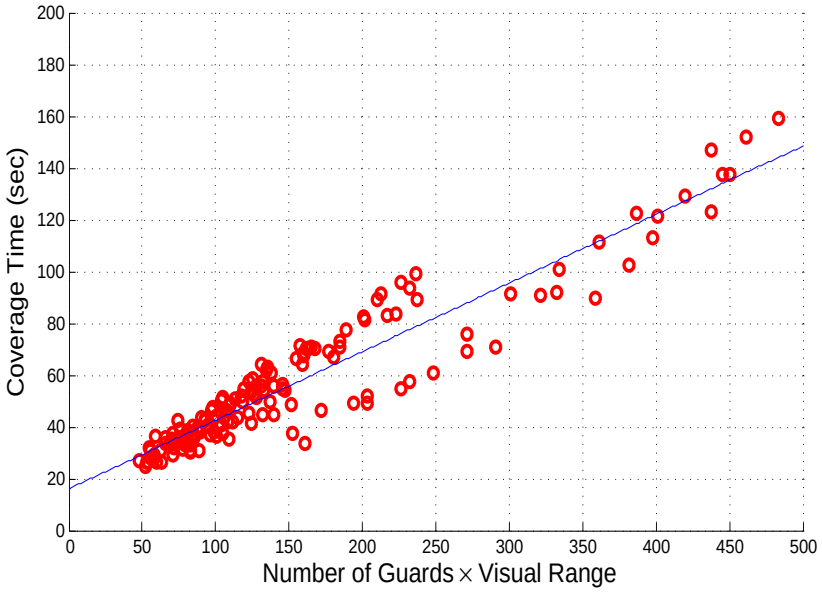


Figure 3.17: *Coverage Time* $\propto$ *Number of Guards* $\times$ *Visual Range*

the percentage of clutter in the environments increase, the *Coverage Time* declines.

Moreover, Figure 3.17 shows that the *Coverage Time* is a function of the number of guards and the robots' visual range, and there is a positive correlation ($R^2 = 0.9$) between the *Coverage Time* and the number of guards times the robots' visual range:

$$Coverage\ Time \propto Number\ of\ Guards \times Visual\ Range, \tag{3.8}$$

and considering Equation 3.7, we can conclude that the obtained results on the *Coverage Time* of the sample environments are scalable to workspaces of different sizes (*i.e.*, different amounts of free space), and robots' of varying visual ranges.

3.7 Conclusions

We addressed the problem of multi-robot single coverage of polygonal workspaces containing polygonal obstacles. The robots have a 360° field of view and a predefined circular limit of visual range. The proposed *Cyclic Coverage* algorithm is complete, meaning that any accessible point of the workspace is visited in a finite time by at least one of the paths assigned to the robots.

Cyclic Coverage improves upon the coverage algorithms based on *Approximate Cellular Decomposition*, which ignore partially occluded cells or areas close to the boundaries. It is also an improvement on the algorithms based on *Exact Cellular Decomposition*, which do not have a clear policy for moving within the cells, and may have many redundant motions while moving between the cells in the workspace.

The proposed algorithm supports heterogeneous robots having various maximal speeds, and supports robustness by handling individual robot failure. It also balances the workload distribution among the robots based on their maximal speeds. Finally, it is shown that the obtained results on the *Coverage Time* of the sample environments are scalable to workspaces of different sizes (*i.e.*, different amounts of free space), and robots of varied visual ranges.

Chapter 4

On Multi-Robot Repeated Area Coverage

4.1 Problem Definition and Preliminaries

In this chapter, we address the *Multi-Robot Repeated Area Coverage* problem with the following specifications:

Assumption 4.1. *The workspace, $\mathcal{W} \subset \mathbb{R}^2$, is a given polygon containing polygonal obstacles.*

Assumption 4.2. *The robots, $\mathcal{R}$, are assumed to have a 360° field of view and a predefined circular limit of visual range.*

Assumption 4.3. *The $|\mathcal{R}|$ robots of the team are homogeneous, having the same maximal speed.*

The goal is is to visit all the accessible points of the target area repeatedly over time, using a given number of robots. In order to evaluate the performance, some metric criteria need to be determined, but before that we introduce some basic definitions:

- ***Full Single Coverage:*** all the robots traverse the paths assigned to them just once.

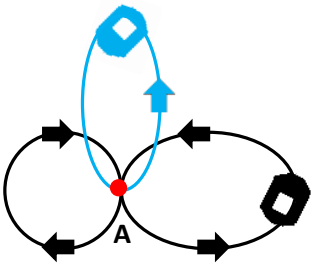


Figure 4.1: *Visiting Period and Visiting Frequency*

- ***Visiting Period (VP):*** the time interval between two visits to a point of interest in the target area. A point of interest can have more than one *Visiting Period*, due to the possibility that the point may be visited more than once in different time intervals by one or more than one robot in a *Full Single Coverage*. For example, in Figure 4.1, point *A* (shown by the red dot) has 3 *Visiting Periods*, 2 are determined by the black robot/tour and 1 is determined by the blue robot/tour.

- ***Average Visiting Period (AVP):*** the average of the *Visiting Periods* of a point of interest.

- ***Worst Visiting Period (WVP):*** the maximum period of time it takes a point of interest to be re-visited in the target area.

- ***Visiting Frequency (VF):*** the number of visits to a point of interest by a single robot in a *Full Single Coverage*. If a point of interest is visited by more than one robot in a *Full Single Coverage*, the point will have more than one *Visiting Frequency*, each associated with a different robot. For example, in Figure 4.1, point *A* has 2 *Visiting Frequencies*, one is determined by the black robot/tour and the other is determined by the blue robot/tour. The *Visiting Frequency* of point *A* on the black tour is 2 and on the blue tour is 1.

The repeated coverage algorithms will be evaluated based on the following metrics:

- ***Total Path Length (TPL):*** the sum of the lengths of the paths assigned to the robots in order to have a *Full Single Coverage*.

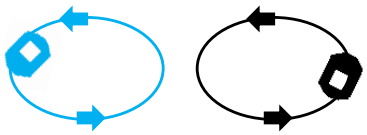
- ***Total Average Visiting Period (TAVP):*** the average of the *Average Visiting Periods* of all the points of interest in the target area.
- ***Total Worst Visiting Period (TWVP):*** the maximum *Worst Visiting Period* of all the points of interests in the target area.
- ***Balance in Workload Distribution (BWD):*** the degree of balance in the workload distribution among a team of robots $\in [0, 100]$.

In this study, the aim is to minimize *TPL*, *TAVP*, and *TWVP* and to maximize *BWD* in the repeated coverage scenario. Interestingly, it is not possible even to develop polynomial approximation algorithms, when optimizing any one of the metrics mentioned above, unless $P = NP$ [100]. Furthermore, optimizing all these metrics simultaneously is another challenge, because some are mutually conflicting in the coverage problem. These considerations require us to conduct an extensive experimental analysis to evaluate the performance of the algorithms.

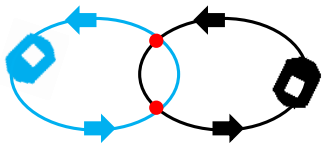
Overview: In this chapter:

1. Three *Cluster-based* algorithms are introduced for the distributed repeated coverage problem, differing as to how they partition the workspace among the robots, namely, the *Uninformed Clustering Coverage*, the *Edge-based Clustering Coverage*, and the *Node-based Clustering Coverage* algorithms.
2. The repeated version of *Cyclic Coverage* is used as a benchmark to compare the performance of the repeated coverage algorithms.
3. The effects of environment representation, and the robots visual range on the performance of the repeated coverage algorithms are investigated.

The results can be used as a framework for choosing an appropriate combination of repeated coverage algorithm, environment representation, and the robots' visual range based on the particular workspace and the metric to be optimized.



(a) *Non-overlapped Paths*



(b) *Overlapped Paths*

Figure 4.2: *Non-overlapped vs. Overlapped Paths for Two Robots*

4.1.1 Computing the Evaluation Metrics

The paths of the robots in the target area may not overlap, that is, there is no common *Point of Interest* among the robots' paths (as shown in Figure 4.2a), or may overlap, that is, there are some common *Points of Interest* among the robots' paths (*e.g.* red dots shown in Figure 4.2b). Considering this, the evaluation metrics are defined as below:

Total Path Length (TPL)

$$TPL = \sum_{i=1}^{|\mathcal{R}|} \|Path(r_i)\|. \tag{4.1}$$

$|\mathcal{R}|$ is the number of robots, $Path(r_i)$ is the path built for robot r_i, and $\|Path(r_i)\|$ is the length of the path.

Total Average Visiting Period (TAVP)

$$TAVP = \frac{\sum_{node \in PoI} AVP(node)}{|PoI|}, PoI = Points\ of\ Interest, \tag{4.2}$$

where

$AVP(node) = \frac{\|Path(r_i)\|}{VF_i(node)}$, $i \in \{1,2,...,|\mathcal{R}|\}$, $node \in Path(r_i)$, if $node$ is not a common

Point of Interest among the robots' paths, or
$AVP(node) = \frac{1}{\sum_i \frac{VF_i(node)}{\|Path(r_i)\|}}$, $i \in \{1,2,...,|\mathcal{R}|\}$, $node \in Path(r_i)$, if *node* is a common *Point of Interest* among the robots' overlapped paths.

$VF_i(node)$ is the *node Visiting Frequency* in $Path(r_i)$.

Total Worst Visiting Period (TWVP)

$$TWVP = max_{node \in PoI}(WVP(node)), \tag{4.3}$$

where

$WVP(node) = max\{VP_i(node)\}$, $node \in Path(r_i)$, if *node* is not a common *Point of Interest* among the robots' paths, or
$WVP(node) = min_{i=1,2,...,|\mathcal{R}|}\{max\{VP_i(node)\}\}$, $node \in Path(r_i)$, if *node* is a common *Point of Interest* among the robots' overlapped paths.

$\{VP_i(node)\}$ is the set of all the *Visiting Periods* of the *node* in $Path(r_i)$.

In computing the *WVP* of a *Point of Interest* which is common among the robots' paths, we first calculate the maximum *Visiting Period* of the point in each robot's path, and then choose the minimum of the maximum values. Recall that each *Point of Interest* can belong to the path of more than one robot, and can also have more than one *Visiting Period* in each robot's path.

Balance in Workload Distribution (BWD)

$$\begin{aligned} &BWD(Paths) = \\ &(1 - \frac{STD(\{\|Path(r_i)\| \, | i = 1,2,...,|\mathcal{R}|\})}{STD(\{TPL, \alpha_1, \alpha_2, \ldots, \alpha_{|\mathcal{R}|-1} | \alpha_i = 0\})}) \times 100, \end{aligned} \tag{4.4}$$

where $STD(.)$ is the population standard deviation, and $Paths = \{Path(r_1), Path(r_2), \ldots, Path(r_{|\mathcal{R}|})\}$ is the $|\mathcal{R}|$ paths created for the $|\mathcal{R}|$ robots. For the case of one robot, we assume that $BWD(Paths) = 100$. In BWD's computation, $STD(\{\|Path(r_i)\| \, | i = 1,2,...,|\mathcal{R}|\})$ is the population standard deviation of the set of paths created for the robots, and $STD(\{TPL, \alpha_1, \alpha_2, \ldots, \alpha_{|\mathcal{R}|-1} | \alpha_i = 0\})$ is the worst case scenario, in which one robot is in charge of the whole task, *i.e.* TPL, and the other robots are idle with zero path length $(\alpha_1, \alpha_2, \ldots, \alpha_{|\mathcal{R}|-1})$. A

workload distribution is 100% *balanced* if the standard deviation of the lengths of the constructed paths for the robots is zero, *i.e.*, the paths assigned to the robots all have equal lengths.

4.1.2 Stages of the Repeated Coverage Algorithms

The stages of the proposed repeated coverage algorithms are as follows:

1. A set of static guards (*SG*), *Points of Interest*, required to observe a given workspace, is located, considering the limited visual range constraint of the robots (Section 4.2).

2. A graph is built on the guards and the nodes of the workspace based on either the *Visibility Graph (VG)* or the *Constrained Delaunay Triangulation (CDT)* (Section 4.3).

3. The graph is reduced to either the *Reduced-Vis* or the *Reduced-CDT* representation (Section 4.4).

4. Coverage Algorithms:

   (a) *Cluster-based Coverage Algorithms:* The *Reduced Graph* is partitioned into as many clusters as the number of robots. To this end, three different clustering algorithms are introduced, namely: *Uninformed Clustering*, *Edge-based Clustering*, and *Node-based Clustering*. Finally, a tour is built for each robot on the clustered *Reduced Graph*. For this purpose, two tour building algorithms are proposed, namely: *Double-Minimum Spanning Tree*, and the *Chained Lin-Kernighan* algorithms (Section 4.5).

   (b) *Cyclic Coverage Algorithm: Cyclic Coverage* finds the shortest tour on the whole *VG* or *CDT* graph, passing through all the static guards, and then distributes the robots equidistantly around it (Section 4.6).

In the following sections, we will explain the different stages of the proposed algorithms for repeated coverage of a target area in detail.

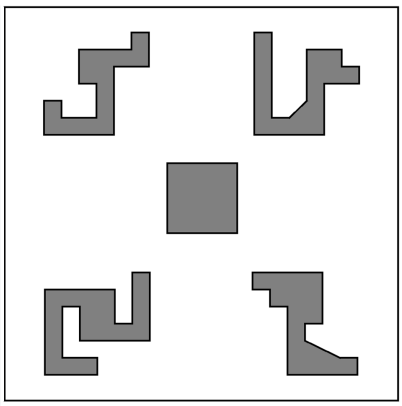


Figure 4.3: *Original Map*

4.2 Locating Guards with Limited Visual Range

In our problem definition, we assume the robots are equipped with panoramic cameras with a 360° field of view. However, the cameras' visual range is limited. The proposed approach initially locates a set of guards *(Points of Interests)* within the workspace, according to the robots' limited visual range. As mentioned in Chapter 3, these static guards are control points from which the whole workspace can be jointly observed. In other words, if there are as many robots as there are guards, and each robot were stationed on a guard, the entire area would be visible to the robots.

We use the same method presented in Section 3.2 of Chapter 3 to locate the guards in the workspace.

4.3 Building the Graphs

Having located the static guards in the previous step, the *Visibility Graph* and the *Constrained Delaunay Triangulation* are then built on the guards and the nodes of the workspace[a].

[a]For brevity, and given the similarity of the results between the *Visibility Graph* and the *Constrained Delaunay Triangulation* under different optimization metrics which will be discussed in the Evaluation and Experimental Simulations section (Section 4.8), we just show the graphical results of the coverage algorithms on the sample workspaces modeled by the *Visibility Graph*.

4.3.1 Visibility Graph

The *Visibility Graph (VG)* is a graph structure used in computational geometry and robot motion planning [82]. The *Visibility Graph (VG)* of a set of nodes, N, in the Euclidean plane is a graph, $VG(N)$, in which if two nodes are mutually visible, they are connected by an edge. Two nodes are mutually visible if the line segment joining them does not intersect any obstacle [36]. Figures 4.4a, 4.4c, and 4.4e illustrate the *Visibility Graphs* built on the sample workspace of Figure 4.3 under visual ranges of $0.5m$, $1m$, and $1.5m$.

4.3.2 Constrained Delaunay Triangulation

The *Delaunay Triangulation (DT)* of a set of nodes, N, in the Euclidean plane is a triangulation, $DT(N)$, such that the circumcircle of any triangle in the triangulation $DT(N)$ does not contain any nodes other than the three that define it *(Delaunay Condition)* [85].

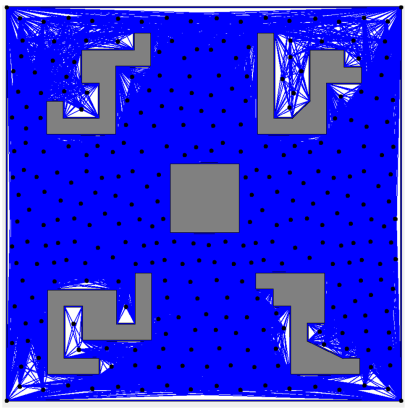
The *Constrained Delaunay Triangulation (CDT)* is a variant of the standard *Delaunay Triangulation* in which a set of pre-specified edges (in our case, the edges of the workspace) must lie in the triangulation [29].

4.4 Graph Reduction

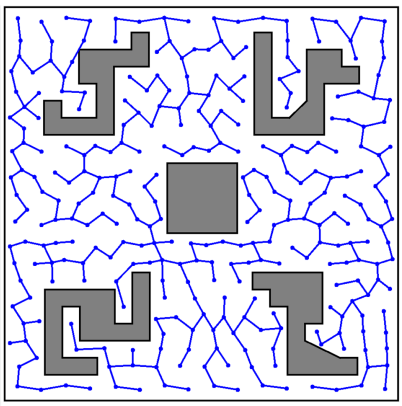
The aim of the graph reduction method is to improve efficiency by minimizing the time taken for the robots to traverse the graph. *Algorithm 4.1* describes the steps of the construction of a *Reduced Graph* (*Reduced-Vis* or *Reduced-CDT*) on a given environment. The input to the algorithm is the *VG* or the *CDT* discussed in Section 4.3.

The method starts by using the *Floyd-Warshall* algorithm to find the set $MD = \{md(g_i, g_j) | g_i, g_j \in SG\}$ of minimum distances, and the set $SP = \{sp(g_i, g_j) | g_i, g_j \in SG\}$ of shortest paths between any pair of guards g_i and g_j of the input graph *(line 2)*.

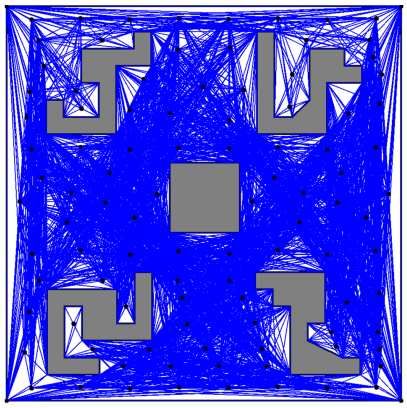
The guards associated with the minimum value of all the minimum distances in *MD*, namely the closest pair of guards in the workspace, are then selected *(line 3)*, and the corresponding shortest path in *SP*, including all its nodes and edges, forms the initial component of the *Reduced Graph (line 4)*.



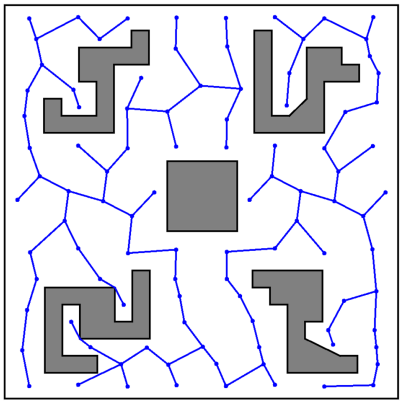
(a) *Visibility Graph, VR=0.5m*



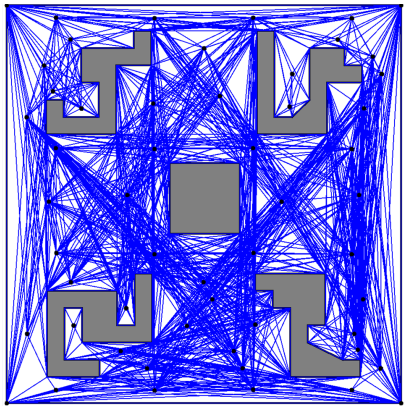
(b) *Reduced-Vis, VR=0.5m*



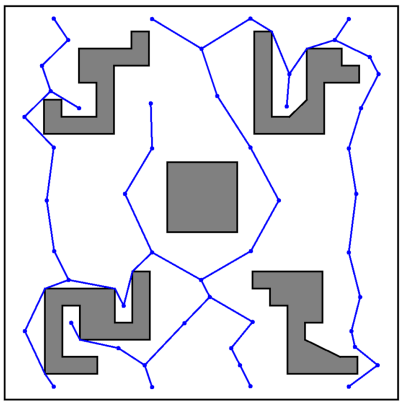
(c) *Visibility Graph, VR=1m*



(d) *Reduced-Vis, VR=1m*



(e) *Visibility Graph, VR=1.5m*



(f) *Reduced-Vis, VR=1.5m*

Figure 4.4: *VG and Reduced-Vis*

Algorithm 4.1: Graph Reduction

Input:
Graph $G_{vis-cdt}(V_{vis-cdt}, E_{vis-cdt})$, where $V_{vis-cdt} = SG \bigcup P$ `/*VG or CDT*/`
$SG = \{g_1, g_2, ..., g_m\}$ `/* Static Guards */`
$P = \{p_1, p_2, ..., p_n\}$ `/* Workspace Nodes */`

Output:
$G_{r-vis-cdt}(V_{r-vis-cdt}, E_{r-vis-cdt})$ where $V_{r-vis-cdt} = SG \bigcup \widetilde{P}$, $\widetilde{P} \subset P$
`/* Reduced Graph */`

```
1  begin
2      (MD, SP) ⟵ FloydWarshall(G_vis-cdt)
3      (g_i, g_j) ⟵ arg min_(g_i,g_j) MD
4      G_r-vis-cdt ⟵ InitialReducedGraph(sp(g_i, g_j))
5      while ¬ all the guards added do
6          g ⟵ FindClosestGuardTo(G_r-vis-cdt)
7          Expand(G_r-vis-cdt, g)
8      end
9      return G_r-vis-cdt
10 end
```

Next, among all the guards that have not yet been added to the graph, the algorithm finds the closest guard to the current component *(line 6)*, merging the corresponding shortest path with it *(line 7)*. Following the same process, the algorithm keeps expanding the component until there are no more guards to be added to the graph *(lines 5-8)*. The resultant graph is the final *Reduced Graph (line 9)*. The nodes of the graph includes all the guards, as the *Points of Interest* in the target area and the subset of the workspace nodes ($\widetilde{P} \subset P$). Traversing the *Reduced Graph* guarantees complete coverage of the target area given the limited visual range of the robots.

Figures 4.4b, 4.4d, and 4.4f illustrate the *Reduced-Vis* graphs computed on the *Visibility Graphs* of Figures 4.4a, 4.4c, and 4.4e.

Algorithm 4.2: Uninformed Clustering Coverage

Input:
$G_{vis-cdt}(V_{vis-cdt}, E_{vis-cdt})$, where $V_{vis-cdt} = SG \bigcup P$ `/* VG or CDT */`
$SG = \{g_1, g_2, ..., g_m\}$ `/* Static Guards */`
$P = \{p_1, p_2, ..., p_n\}$ `/* Workspace Nodes */`
$G_{r-vis-cdt}$: the *Reduced-Vis* or the *Reduced-CDT* Graph
$|\mathcal{R}|$: Number of Robots

Output:
A set of $|\mathcal{R}|$ tours, $Tours = \{T_1, T_2, \ldots, T_{|\mathcal{R}|}\}$ where $\bigcup_{i=1}^{|\mathcal{R}|} SG_{T_i} = SG$, SG is the set of all guards and SG_{T_i} is the set of guards of the tour T_i

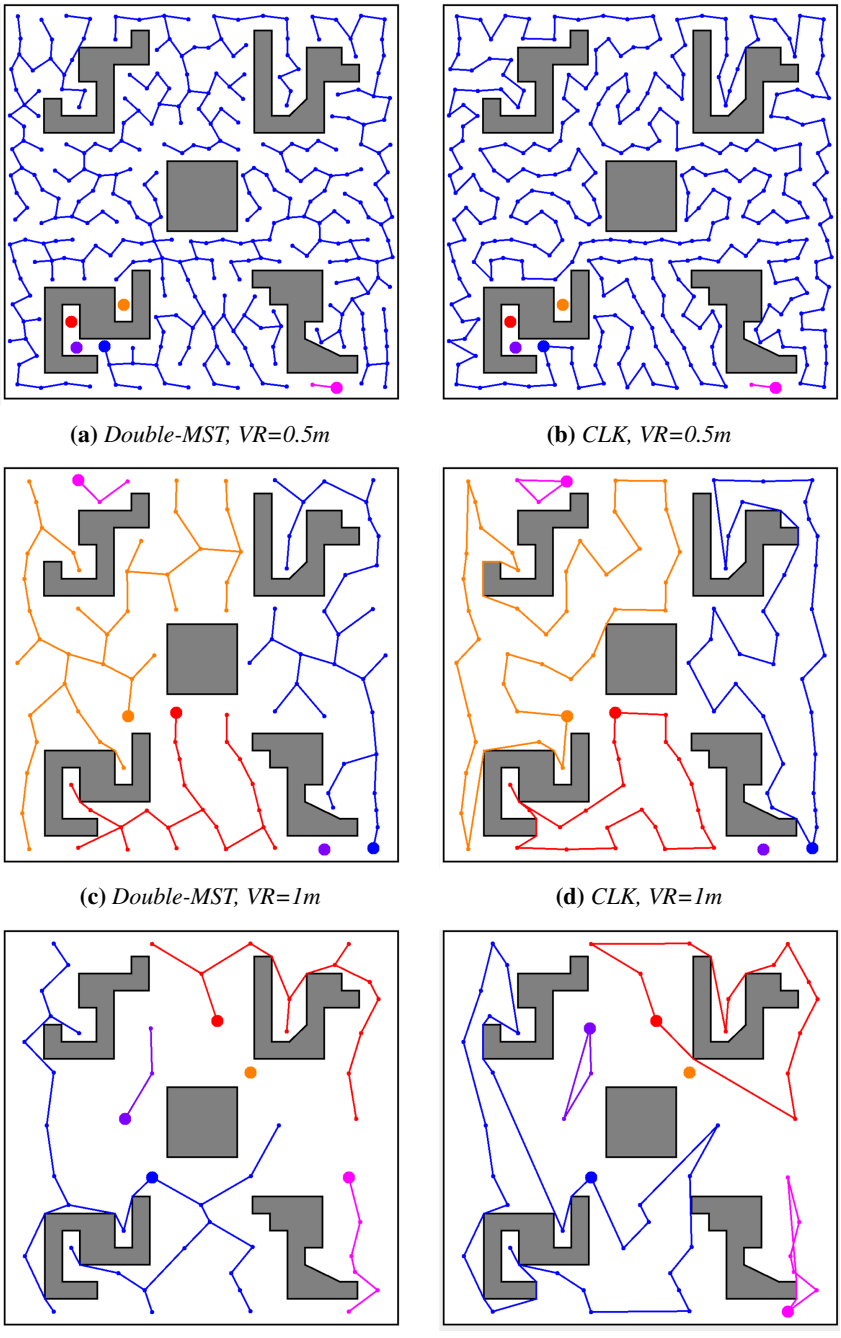
```
1 begin
2     /*Remove |R| − 1 longest edges of G_{r−vis−cdt}*/
      Clusters ⟵ RemoveLongestEdges(G_{r−vis−cdt}, |R| − 1)
3     foreach C_i ∈ Clusters do
4         T_i ⟵ BuildTour(C_i)
5     end
6     return Tours
7 end
```

4.5 Cluster-based Coverage Algorithms

Cluster-based coverage algorithms decompose the *Reduced Graph* into $|\mathcal{R}|$ (number of robots) clusters, $Clusters = \{C_1, C_2, \ldots, C_{|\mathcal{R}|}\}$, such that $\bigcup_{i=1}^{|\mathcal{R}|} SG_{C_i} = SG$; SG is the set of all guards and SG_{C_i} is the set of guards of the cluster C_i. Below, three different *cluster-based* coverage algorithms are presented. The input of the proposed algorithms can be either of the *Reduced Graphs*. Having built the clusters on the *Reduced Graphs*, a tour is built on the generated cluster for each robot, $Tours = \{T_1, T_2, \ldots, T_{|\mathcal{R}|}\}$. The tour building algorithms are discussed in Section 4.5.4.

4.5.1 Uninformed Clustering Coverage

Uninformed Clustering Coverage (Algorithm 4.2) partitions the *Reduced Graph* into $|\mathcal{R}|$ clusters by removing the $|\mathcal{R}| - 1$ longest edges of the graph *(line 2)*. There-



(a) *Double-MST, VR=0.5m*

(b) *CLK, VR=0.5m*

(c) *Double-MST, VR=1m*

(d) *CLK, VR=1m*

(e) *Double-MST, VR=1.5m*

(f) *CLK, VR=1.5m*

Figure 4.5: *Tours Built for Five Robots by Uninformed Clustering Coverage*

after, a tour is built on each cluster generated for the robots *(lines 3-5)*. There is no overlap (existence of common guards) among the tours built by *Uninformed Clustering Coverage*.

Figure 4.5 illustrates the tours built for five robots on the sample workspace by the *Uninformed Clustering Coverage* algorithm, under visual ranges of $0.5m$, $1m$, and 1.5.

4.5.2 Edge-based Clustering Coverage

In *Edge-based Clustering Coverage (Algorithm 4.3)*, the clusters are initiated as follows: the endpoint guards of the longest path in the original graph (*VG* or *CDT*) are selected as the starting points of the first two clusters. For the next cluster, a guard is selected such that it maximizes the sum of the distances from the starting points of the first two clusters. Similarly, for the next cluster, a guard is selected that maximizes the sum of the distances from the starting points of the first three clusters. This continues until $|\mathcal{R}|$ initial guards are found for the $|\mathcal{R}|$ clusters of the robots *(line 2)*.

The reason to initiate the clusters in the original *VG* or *CDT* graph is to distribute the clusters spatially as much as possible far away from each other in the target area. Distance between the guards in the original *VG* or *CDT* graph, in contrast to the Euclidean distance, takes into account the obstacles in the area, and is more accurate than the distance in the *Reduced Graph*, because there are many edges between the nodes in the original graph which were removed in the *Reduced Graph*.

Starting from the initial guards, clusters are expanded in the *Reduced Graph* sequentially by choosing one guard at a time, until all the guards of the *Reduced Graph* have been selected at least once *(lines 3-11)*. Each cluster selects guards in a way that satisfies the following constraints:

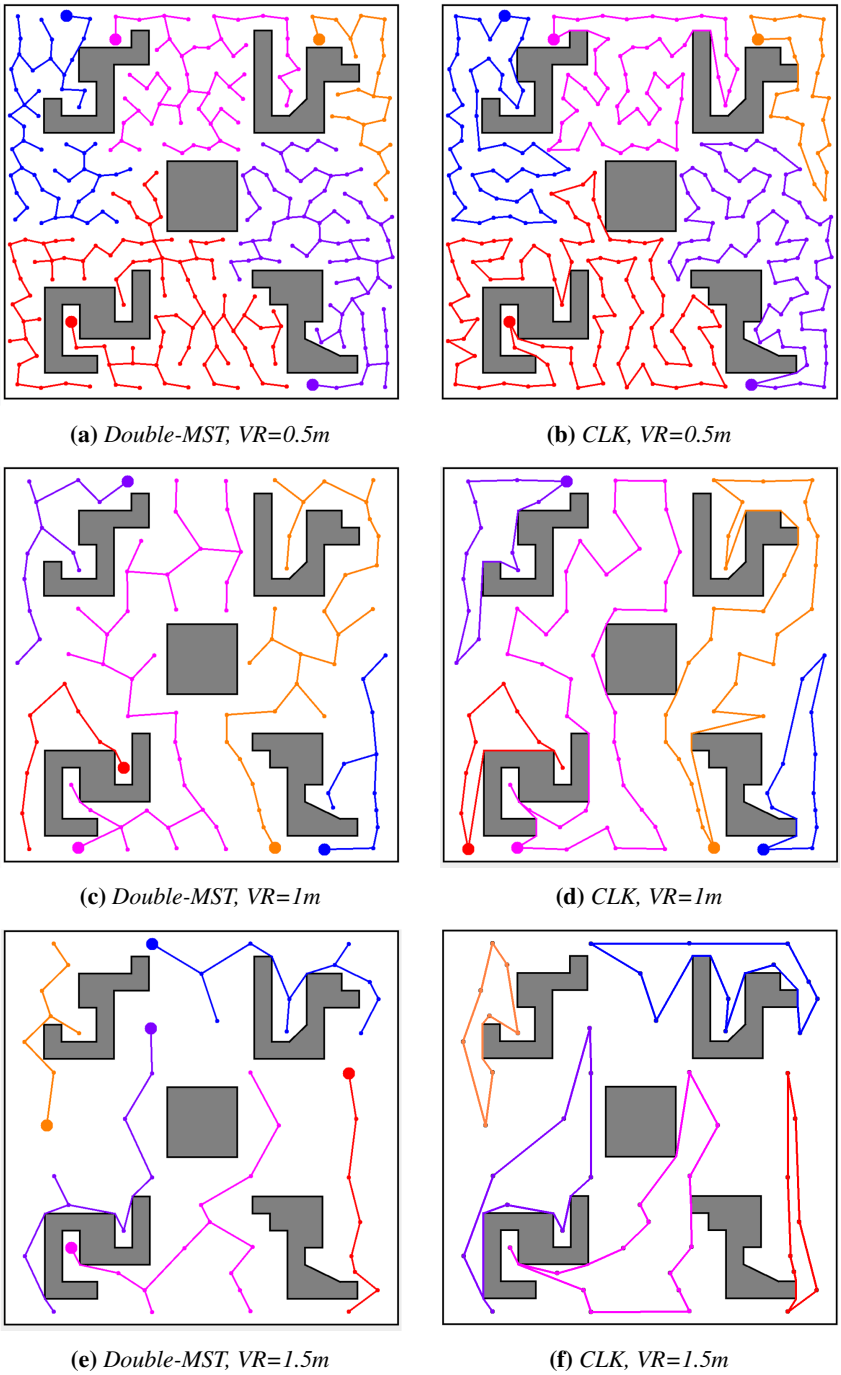
- Find the nearest unselected guard in the *Reduced Graph*, add it and the corresponding edge/path to the cluster *(line 5)*.
- Do not add a guard which has already been chosen by another cluster, unless there is no other unselected immediate guard *(lines 6-8)*.

Algorithm 4.3: Edge-based Clustering Coverage

Input:
$G_{vis-cdt}(V_{vis-cdt}, E_{vis-cdt})$, where $V_{vis-cdt} = SG \bigcup P$ `/* VG or CDT */`
$SG = \{g_1, g_2, ..., g_m\}$ `/* Static Guards */`
$P = \{p_1, p_2, ..., p_n\}$ `/* Workspace Nodes */`
$G_{r-vis-cdt}$: the *Reduced-Vis* or the *Reduced-CDT* Graph
$|R|$: Number of Robots

Output:
A set of $|\mathcal{R}|$ tours, $Tours = \{T_1, T_2, \ldots, T_{|\mathcal{R}|}\}$ where $\bigcup_{i=1}^{|\mathcal{R}|} SG_{T_i} = SG$, SG is the set of all guards and SG_{T_i} is the set of guards of the tour T_i

```
1  begin
2      Clusters ⟵ InitiateClusters(G_vis-cdt)
3      while ¬ all the guards of the graph G_r-vis-cdt visited do
4          foreach C_i ∈ Clusters do
5              find g ∈ SG which is the nearest immediate guard to C_i in
               G_r-vis-cdt and ¬ visited
6              if there is no such a guard g then
7                  find g ∈ SG which is the nearest immediate guard to C_i in
                   G_r-vis-cdt
8              end
9              C_i.add(g)
10         end
11     end
12     foreach C_i ∈ Clusters do
13         if there are common guards between C_i and another cluster then
14             if the guards have been selected earlier by the other cluster then
15                 if ¬ removing the common guards from C_i disconnects either
                   cluster then
16                     RemoveCommonGuardsFrom(C_i)
17                 end
18             end
19         end
20         T_i ⟵ BuildTour(C_i)
21     end
22     return Tours
23 end
```



(a) *Double-MST, VR=0.5m*

(b) *CLK, VR=0.5m*

(c) *Double-MST, VR=1m*

(d) *CLK, VR=1m*

(e) *Double-MST, VR=1.5m*

(f) *CLK, VR=1.5m*

Figure 4.6: *Tours Built for Five Robots by Edge-based Clustering Coverage*

Remove Common Guards: When all the guards of the graph have been selected by at least one cluster, remove as many as possible of the guards shared by the clusters of the robots. To this end, for each cluster, discard the guards and their corresponding edges/paths from the cluster if they have been selected earlier by any other cluster *(lines 13-19)*. There are cases where the overlap among the clusters cannot be resolved. If removing the overlap among the clusters disconnects any of them, then the overlap is left in all the clusters.

Finally, a tour is built on the generated cluster for each robot *(line 20)*.

Figure 4.6 illustrates the tours built for five robots on the sample workspace by the *Edge-based Clustering Coverage* algorithm, under visual ranges of $0.5m$, $1m$, and $1.5m$.

4.5.3 Node-based Clustering Coverage

Node-based Clustering Coverage (Algorithm 4.4) initially uses the *k-Means* clustering algorithm [67] to divide the guards into $|\mathcal{R}|$ disjoint clusters, $Clusters = \{C_1, C_2, \ldots, C_{|\mathcal{R}|}\}$, in which each guard belongs to the cluster with the nearest mean/centroid *(line 2)*. In other words, given the set of guards $SG = \{g_1, g_2, ..., g_m\}$, *k-Means* minimizes the within-cluster sum of squares:

$$Clusters = \underset{Clusters}{\arg\min} \sum_{i=1}^{|\mathcal{R}|} \sum_{g_j \in C_i} \left\| g_j - \mu_i \right\|^2, \tag{4.5}$$

where $Clusters = \{C_1, C_2, \ldots, C_{|\mathcal{R}|}\}$ is the final clusters of the guards, μ_i is the mean of the guards in cluster C_i, matched to the closest guard in the workspace. $\left\| g_j - \mu_i \right\|^2$ is the distance between a guard and the mean of the cluster in the original *VG* or *CDT* graph.

In the first iteration of *k-Means*, the initial means are found in the same way as finding the starting points of the clusters discussed in *Edge-based Clustering Coverage*. This aims to distribute the clusters spatially as much as possible far away from each other in the target area. Given this initial set of $|\mathcal{R}|$ means, the algorithm proceeds by alternating between two steps: *1) Assignment Step:* Assign each guard to the cluster with the closest mean, *2) Update Step:* Calculate the new means to be the centroid of the guards in the cluster. These two steps are alternated

Algorithm 4.4: Node-based Clustering Coverage

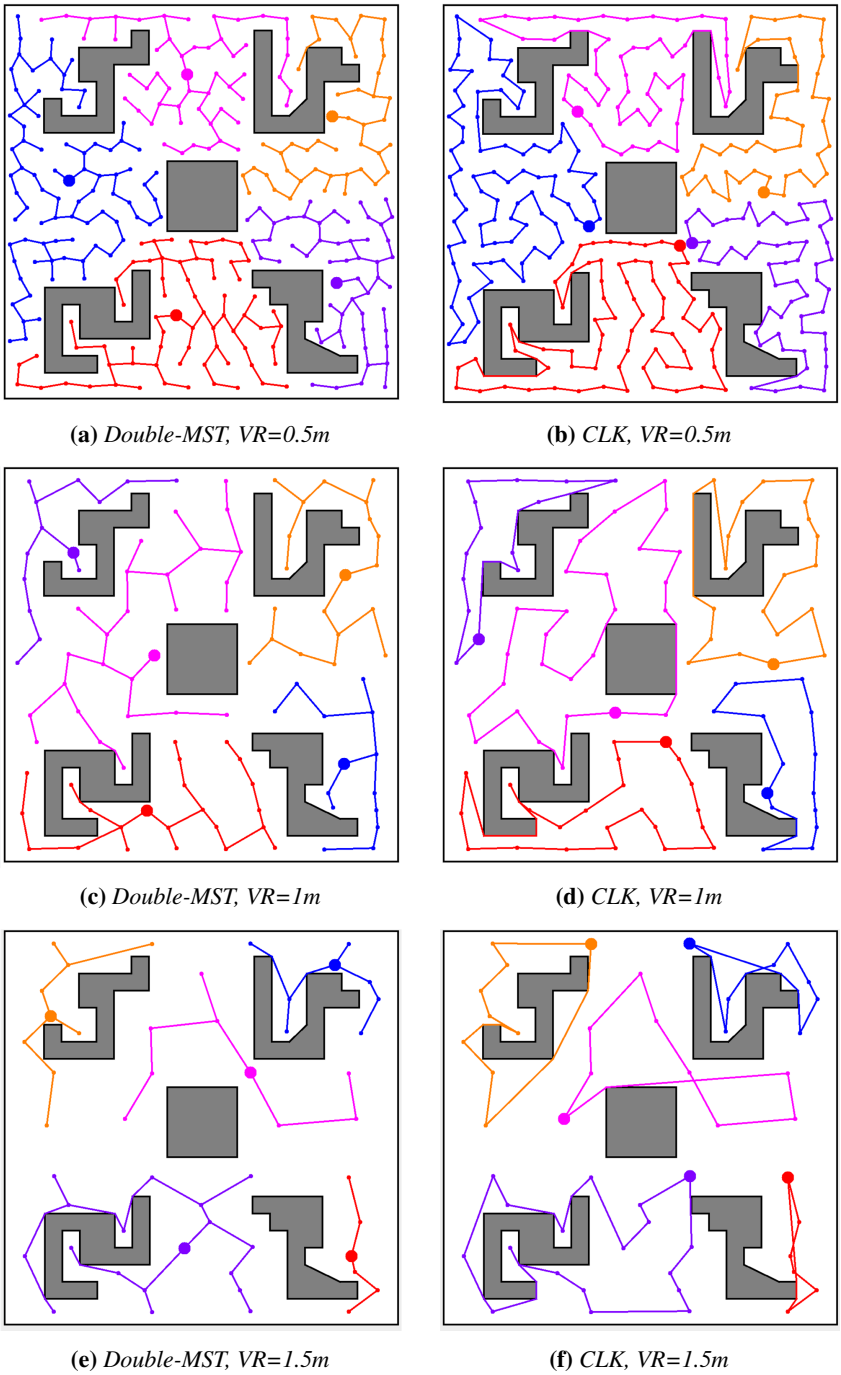
Input:
$G_{vis-cdt}(V_{vis-cdt}, E_{vis-cdt})$, where $V_{vis-cdt} = SG \bigcup P$ `/* VG or CDT */`
$SG = \{g_1, g_2, ..., g_m\}$ `/* Static Guards */`
$P = \{p_1, p_2, ..., p_n\}$ `/* Workspace Nodes */`
$G_{r-vis-cdt}$: the *Reduced-Vis* or the *Reduced-CDT* Graph
$|R|$: Number of Robots

Output:
A set of $|\mathcal{R}|$ tours, $Tours = \{T_1, T_2, \ldots, T_{|\mathcal{R}|}\}$ where $\bigcup_{i=1}^{|\mathcal{R}|} SG_{T_i} = SG$, SG is the set of all guards and SG_{T_i} is the set of guards of the tour T_i

```
1  begin
2      InitialCentroids ⟵ FindInitialCentroids(G_vis-cdt, |R|)
3      Tours ⟵ kMeans(G_vis-cdt, |R|, InitialCentroids)
4      foreach C_i ∈ Clusters do
5          C_i ⟵ ConnectGuards(C_i, G_r-vis-cdt)
6          DCComponents ⟵ FindDCComponents(C_i)
7          C_i ⟵ BuildMST(G_vis-cdt, DCComponents)
8          T_i ⟵ BuildTour(C_i)
9      end
10     return Tours
11 end
```

until there is no further change in the assignment of the guards. Since the computed means may not lie on the guards of the graph, they are matched to the closest guard in the workspace.

Having built the $|\mathcal{R}|$ clusters on the guards *(line 3)*, we connect each pair of guards in each cluster if they have a corresponding path (including the intermediate nodes of the workspace) in the *Reduced Graph (line 5)*. Thereafter, we do a connectivity test on all the clusters, meaning that every pair of guards in each cluster should be connected through a path. To this end, we first find the disconnected components within the cluster *(line 6)* and then compute the *Minimum Spanning Tree* on them by getting help from the edges of the original *VG* or *CDT* graph, and the nodes of the workspace *(line 7)*. The *Minimum Spanning Tree* will not add any



(a) *Double-MST, VR=0.5m* **(b)** *CLK, VR=0.5m*

(c) *Double-MST, VR=1m* **(d)** *CLK, VR=1m*

(e) *Double-MST, VR=1.5m* **(f)** *CLK, VR=1.5m*

Figure 4.7: *Tours Built for Five Robots by Node-based Clustering Coverage*

other new guard to the set of guards existing in the cluster. This prevents the overlap among the clusters; however, the tree can add any nodes of the workspace to the cluster. We add the *Minimum Spanning Tree*'s corresponding edges and nodes to the cluster *(line 7)*, and finally a tour is built on the cluster *(line 8)*. The tour is then assigned to a robot, and the robot repeatedly traverses the tour. In *Node-based Clustering Coverage*, there is no overlap among the tours generated by the algorithm,

Figure 4.7 illustrates the tours built for five robots on the sample workspace by the *Node-based Clustering Coverage* algorithm, under visual ranges of $0.5m$, $1m$, and $1.5m$.

4.5.4 Building the Tour

Having built the clusters on the *Reduced Graph*, we use two algorithms to build the tours on the clusters:

Double-Minimum Spanning Tree (Double-MST)

Takes a cluster as an input and returns a cycle whose length is twice the length of the cluster. In this algorithm, every edge of the cluster is visited twice.

Chained Lin-Kernighan (CLK)

A modification of the *Lin-Kernighan* algorithm [86], is generally considered to be one of the best heuristic methods for generating *optimal* or *near-optimal* solutions for the *Traveling Salesman Problem (TSP)* [11]. Given the distance between each pair of a finite number of nodes in a *complete* graph, the *Traveling Salesman Problem* is to find the shortest tour passing through all the nodes exactly once and returning to the starting node [12].

Having built the clusters on the *Reduced Graph*, the *Chained Lin-Kernighan* algorithm takes the distance matrix of the guards of each cluster in the initial *VG* or *CDT* graph as an input, and finds the shortest tour passing through all the guards of the cluster. The matrix represents the shortest path distances between all pairs of guards of the cluster in the initial *VG* or *CDT* graph, without taking into account any additional guards other than the ones existing in the cluster. However, the shortest

path between the guards in the cluster can include any nodes of the workspace. This guarantees that if there is not an overlap among the guards of the generated clusters, then there will not be any overlap among the guards of the tours either after applying the *Chained Lin-Kernighan* algorithm on the clusters. The input to the *Chained Lin-Kernighan* algorithm should be a complete graph and the distance matrix is indicative of a complete graph, even though the clusters themselves are not complete.

Clearly, the length of each cluster built on the *Reduced Graph* is strictly less than the length of the tour built on the cluster by the *Chained Lin-Kernighan* algorithm.

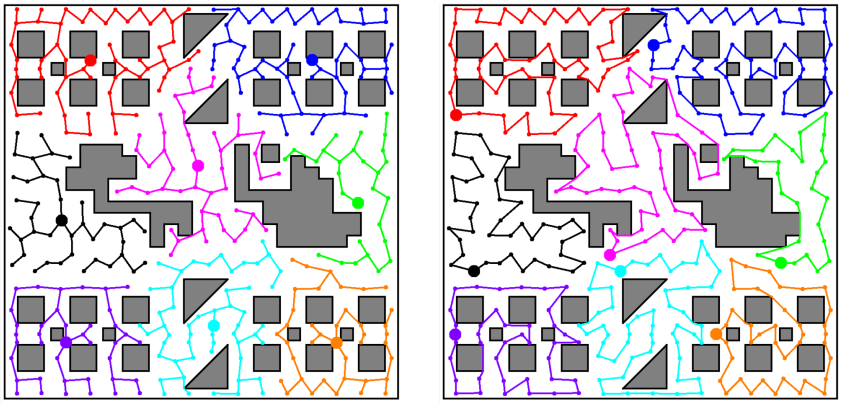
4.5.5 Overlap Among the Tours

Overlap (existence of common guards) among the tours generated by the *Cluster-based* algorithms affects the performance of the algorithms in some cases. The affected cases will be discussed in the Evaluation and Experimental Simulations section (Section 4.8).

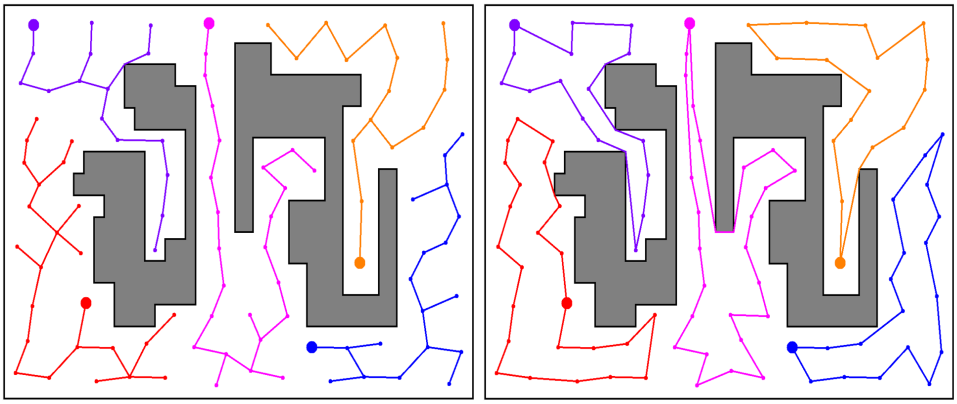
In summary, there is no overlap among the tours built by the *Uninformed Clustering Coverage* and the *Node-based Clustering Coverage* algorithms, using either *Double-MST* or *CLK*. The reason is that the original clusters (before building the tours on them) generated by these two coverage algorithms have no overlap on the guards. However, there may be some unresolved overlaps among the clusters and as a result among the tours generated by *Edge-based Clustering Coverage*, using either *Double-MST* or *CLK*. In the event of an overlap among the tours, the robots assigned to each of the overlapped tours, all have to traverse the common parts among themselves.

4.5.6 Example Solutions

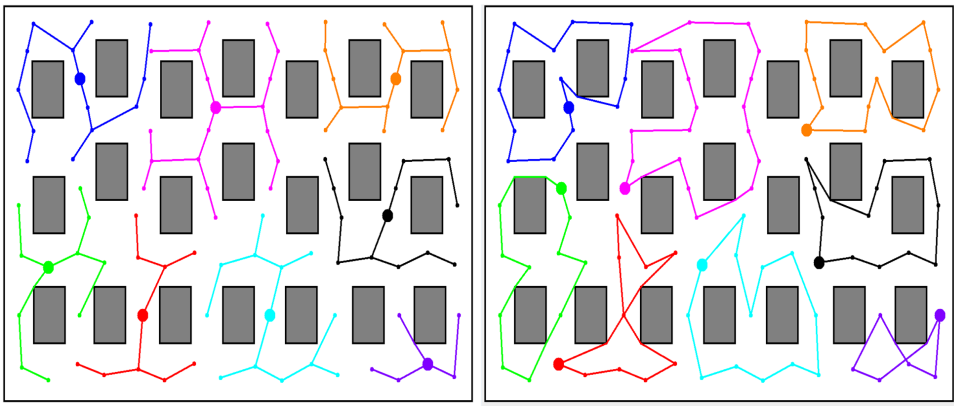
Figures 4.8, 4.9, 4.10, and 4.11 show the tours built on different workspaces under various visual ranges of the robots, using the three *cluster-based* algorithms: *Uninformed Clustering Coverage (UCC)*, *Edge-based Clustering Coverage (ECC)*, and *Node-based Clustering Coverage (NCC)*.



(a) *NCC, WS=10m×10m,VR=0.5m* **(b)** *NCC, WS=10m×10m,VR=0.5m*

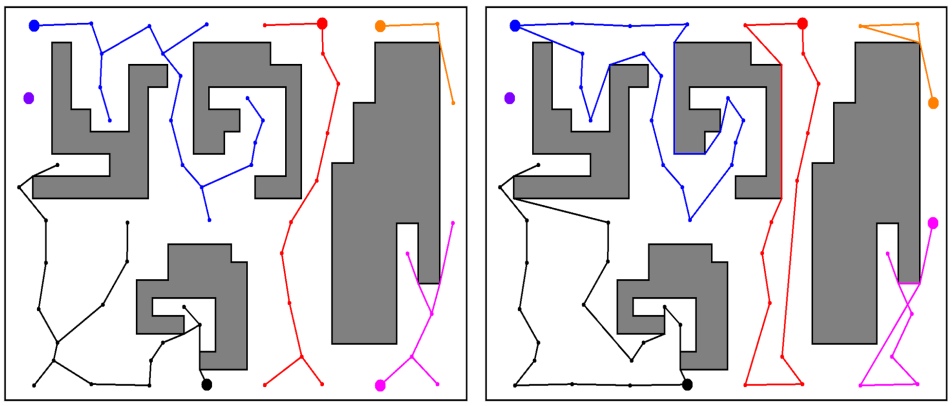


(c) *ECC, WS=10m×12m, VR=1m* **(d)** *ECC, WS=10m×12m, VR=1m*



(e) *NCC, WS=10m×12m, VR=1m* **(f)** *NCC, WS=10m×12m, VR=1m*

Figure 4.8: *Computed Tours by Different Coverage Algorithms*



(a) *UCC, WS=10m×12m, VR=1.5m*

(b) *UCC, WS=10m×12m, VR=1.5m*

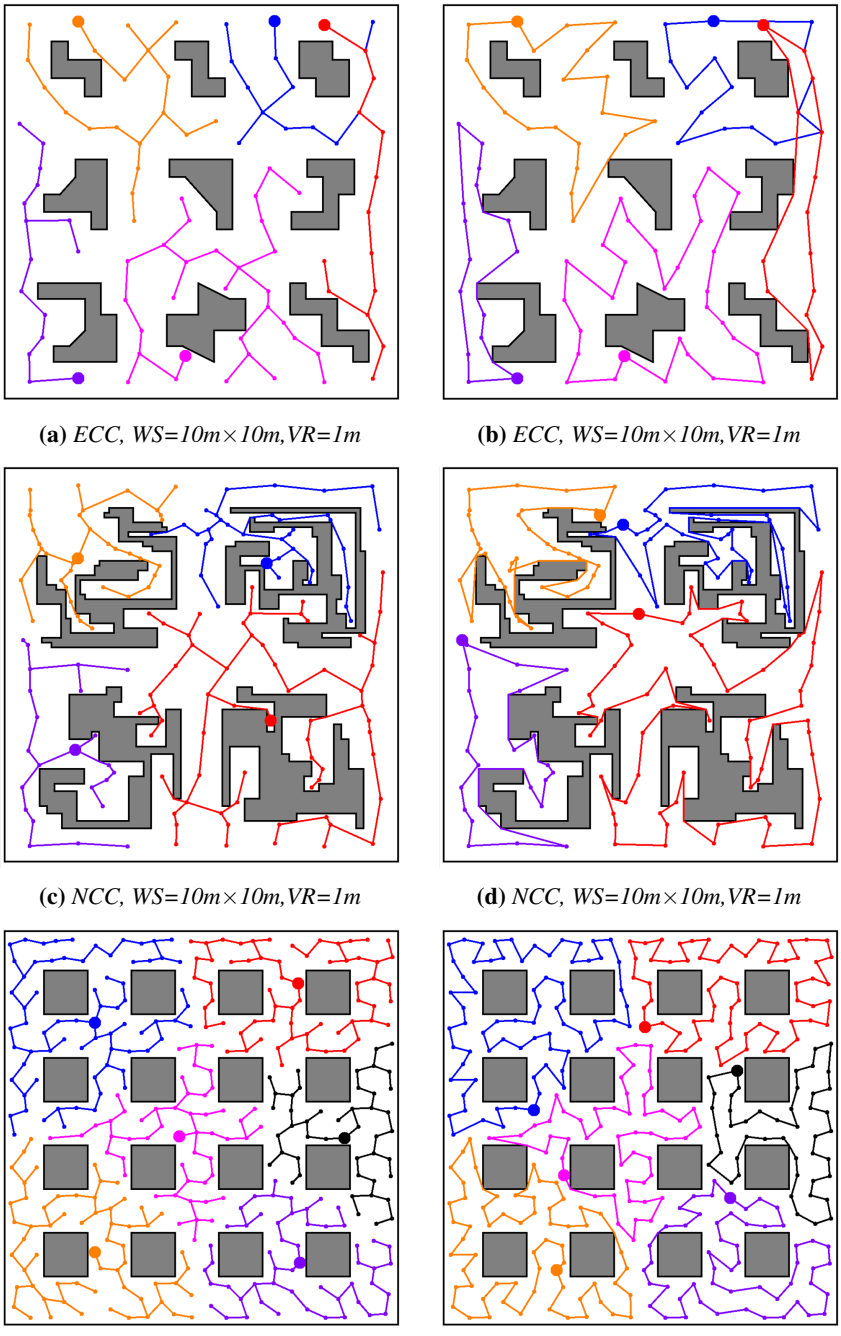
(c) *NCC, WS=10m×10m,VR=1m*

(d) *NCC, WS=10m×10m,VR=1m*

(e) *NCC,WS=10m×10m,VR=0.75m*

(f) *NCC,WS=10m×10m,VR=0.75m*

Figure 4.9: *Computed Tours by Different Coverage Algorithms*

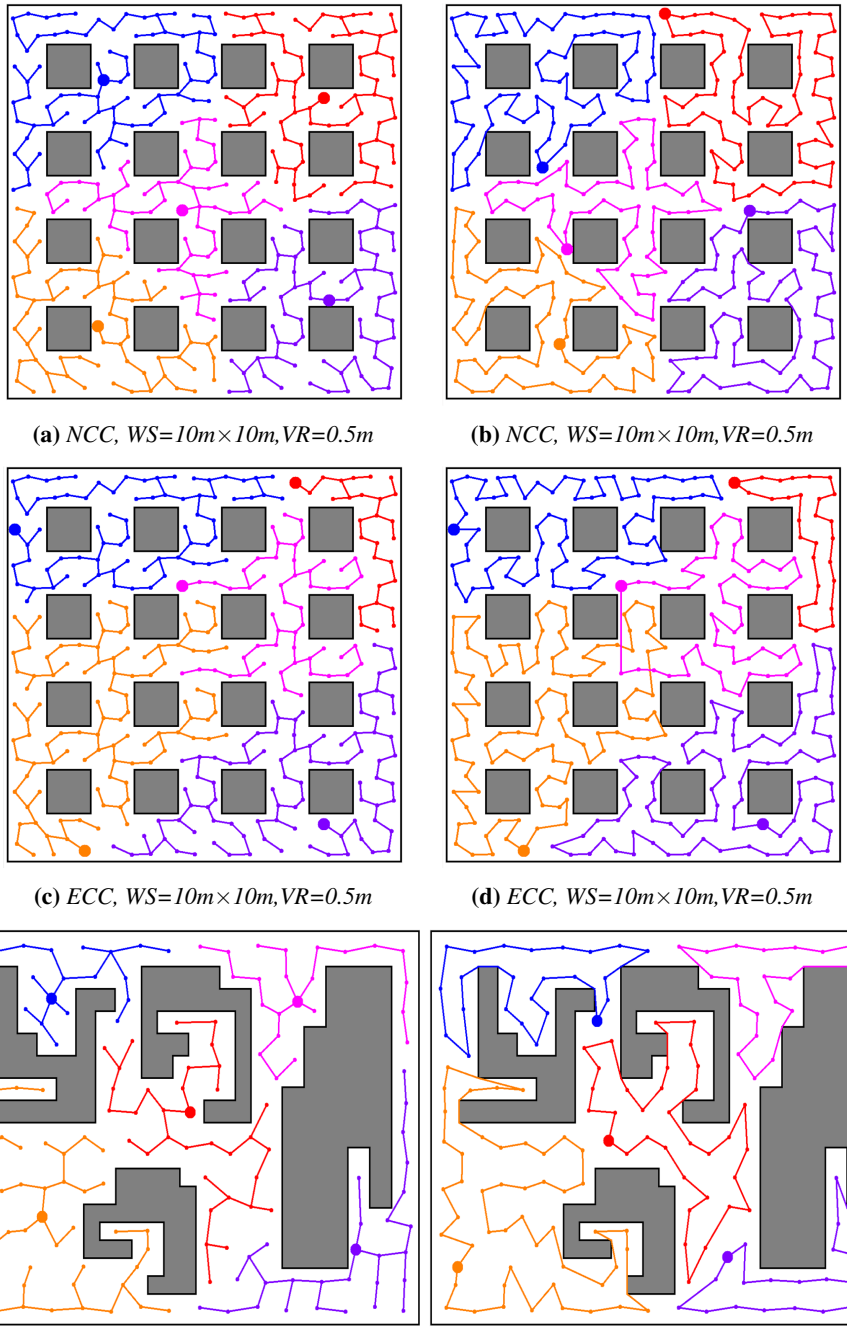


(a) *ECC, WS=10m×10m,VR=1m* **(b)** *ECC, WS=10m×10m,VR=1m*

(c) *NCC, WS=10m×10m,VR=1m* **(d)** *NCC, WS=10m×10m,VR=1m*

(e) *NCC, WS=10m×10m,VR=0.5m* **(f)** *NCC, WS=10m×10m,VR=0.5m*

Figure 4.10: *Computed Tours by Different Coverage Algorithms*



(a) *NCC, WS=10m×10m,VR=0.5m* **(b)** *NCC, WS=10m×10m,VR=0.5m*

(c) *ECC, WS=10m×10m,VR=0.5m* **(d)** *ECC, WS=10m×10m,VR=0.5m*

(e) *NCC, WS=10m×12m, VR=0.75m* **(f)** *NCC, WS=10m×12m, VR=0.75m*

Figure 4.11: *Computed Tours by Different Coverage Algorithms*

4.6 Cyclic Coverage

We use the repeated version of the *Cyclic Coverage* algorithm *(Algorithm* 4.5*)* from Chapter 3 as a benchmark to compare the coverage algorithms. Similar to the *Cluster-based* coverage algorithms, *Cyclic Coverage* locates the guards, and builds the graph (*VG* or *CDT*). However, rather than reducing and partitioning the graph among the robots, it creates a tour passing through all the guards of the graph, using the *Chained Lin-Kernighan* algorithm *(line 2)*. The input of the *Chained Lin-Kernighan* algorithm is the distance matrix of the guards in the *VG* or *CDT* graph. The robots are then distributed equidistantly around the tour and traverse the whole tour repeatedly over time *(line 3)*.

Given the set of guards in the original *VG* or *CDT* graph, *Cyclic Coverage* produces *optimal* or *near-optimal* solutions for the *single-robot* case under *Total Path Length* and the *Total Worst Visiting Period*. The notion of *Balance in Workload Distribution* is not defined in this approach, since all the robots traverse the whole tour built on the original graph.

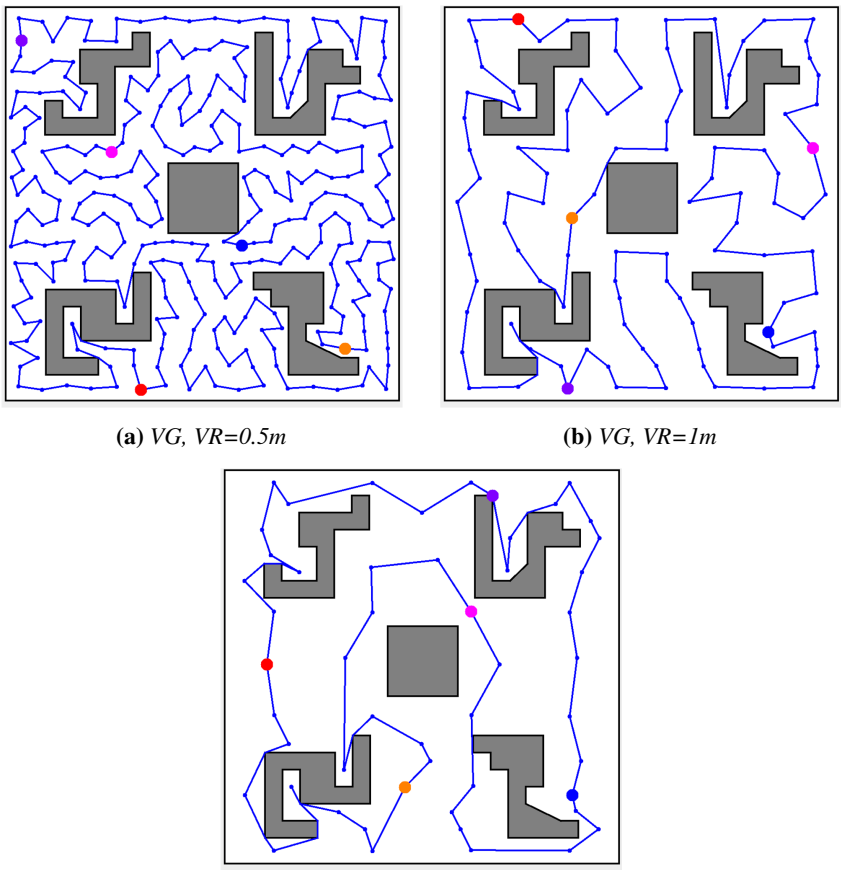
Figure 4.12 illustrates the tour built for five robots on the sample workspace by the *Cyclic Coverage* algorithm, under visual ranges of $0.5m$, $1m$, and $1.5m$.

Algorithm 4.5: Cyclic Coverage

Input:
$G_{vis-cdt}(V_{vis-cdt}, E_{vis-cdt})$, where $V_{vis-cdt} = SG \bigcup P$ `/* VG or CDT */`
$SG = \{g_1, g_2, ..., g_m\}$ `/* Static Guards */`
$P = \{p_1, p_2, ..., p_n\}$ `/* Workspace Nodes */`
$|\mathcal{R}|$: Number of Robots

Output:
A tour, *dTour*, distributed among the robots, passing through all the guards of the *VG* or *CDT* graph.

```
1 begin
2     tour ⟵ BuildTour(G_vis-cdt)
3     dTour ⟵ DistributeRobots(tour, |R|)
4     return dTour
5 end
```



(a) *VG, VR=0.5m* **(b)** *VG, VR=1m*

(c) *VG, VR=1.5m*

Figure 4.12: *Tours Built by Five Robots by Cyclic Coverage*

4.7 Complexity Analyses of the Algorithms

Similar to the single coverage problem, the basic version of the repeated coverage problem with just one robot with unlimited visual range operating in a simple polygon without obstacles has an exact polynomial time solution [24, 126]. But, extending the problem to include obstacles in the workspace, to multiple robots, or to limited visual range, makes the corresponding decision problems *NP-hard* [97].

Interestingly, it is not even possible to develop polynomial approximation algorithms, when optimizing each of the metrics (*TPL, TAVP, TWVP, and BWD*) defined for the repeated coverage problem, unless $P = NP$ [100]. Furthermore, optimiz-

Stages of the Algorithm		Time Complexity
Locating Guards		$O(n^2 \log_2 n)$
Building Graph	Visibility Graph	$O((n+m)^3)$
	Constrained Delaunay Triangulation	$O((n+m)log(n+m))$
Graph Reduction		$O(m^3)$
Clustering the Reduced Graph	Uninformed Clustering	$O(\|\mathcal{R}\|(n'+m))$
	Edge-based Clustering	$O(\|\mathcal{R}\|mlog(m))$
	Node-based Clustering	$O(\|\mathcal{R}\|Im)+O(m^2)$
Cyclic Coverage	Chained Lin-Kernighan	$O(m^{2.2})^a$
Building Tours	Double-MST	$O(n'+m)$
	Chained Lin-Kernighan	$O(m^{2.2})^a$

Table 4.1: *Time Complexity of Different Stages of the Repeated Coverage Algorithms. n: Number of Workspace Nodes, n': Number of Workspace Nodes in the Reduced Graph, m: Number of Guards, I: Number of Iterations of the Algorithm, $|\mathcal{R}|$: Number of Robots*

ing all the metrics simultaneously is another challenge, because some are mutually conflicting in the coverage problem.

The time complexity of the stages of the proposed coverage algorithms are shown in Table 4.1.

4.8 Evaluation and Experimental Simulations

The goal of the experiments is to evaluate the performance of the four repeated coverage algorithms:

- *Uninformed Clustering Coverage (UCC)*
- *Edge-based Clustering Coverage (ECC)*
- *Node-based Clustering Coverage (NCC)*
- *Cyclic Coverage (CC)*

[a]This entry is based on experimental results [11, 86]. The worst case time complexity of the *Chained Lin-Kernighan* algorithm is apparently not available in the literature [69].

under the effect of the following independent variables:

- Robots' visual range.
- Environment representation (*i.e.* the combination of the graph representation and the tour building algorithms).

For the environment representation, we have four combinations of the graph representation and the tour building algorithms:

- *Visibility Graph (VG)* and *Double-MST*
- *Visibility Graph (VG)* and *Chained Lin-Kernighan (CLK)*
- *Constrained Delaunay Triangulation (CDT)* and *Double-MST*
- *Constrained Delaunay Triangulation (CDT)* and *Chained Lin-Kernighan (CLK)*

The performance of the coverage algorithms is evaluated based on these criteria:

- *Total Path Length (TPL)*
- *Total Average Visiting Period (TAVP)*
- *Total Worst Visiting Period (TWVP)*
- *Balance in Workload Distribution (BWD)*

We perform the experiments on the 30 maps used in Chapter 3. The size of the environments is $15m \times 15m$. Robots are assumed to move with the speed of $1m/s$ in the target area. In order to eliminate the dependency of the results on specific maps, we use the results of *Cyclic Coverage* with *Visibility Graph* under *TPL, TAVP*, and *TWVP* as the reference and the results of the coverage algorithms on each map under these metrics, are normalized as ratios to the reference solution. For optimization metrics *TPL, TAVP*, and *TWVP*, the average values of ratios over all the maps are shown respectively in Figures 4.13, 4.14, and 4.15 for different

numbers of robots $(1, 2, \ldots, 15)$, under the selected visual ranges. Note that the figures do not demonstrate the average of the actual values of the coverage algorithms but the average normalized ratios to the reference solution over all the maps. That is why the values of *Cyclic Coverage* with *Visibility Graph*, shown by the red line in plots 4.13, 4.14, and 4.15, is fixed (equal to 1) for different number of robots. For *BWD*, since the values are bounded between 0 and 100, we show the average of the actual values over all the maps in Figure 4.16.

In sum, we have collected data from $9000 = 4\ Coverage\ Algorithms \times 15\ \#Robots \times 5\ Visual\ Ranges \times 30\ Maps$ runs of the simulator. The results can be used as a framework for choosing an appropriate combination of repeated coverage algorithm, environment representation, and the robots' visual range based on the particular workspace and the metric to be optimized.

4.8.1 Running Time of the Algorithms

Table 4.2 shows the average running time of all the stages of building a *Reduced Graph* over all the maps, under the selected visual ranges of the robots. The stages include: computing the guards, building the graph (*VG* or *CDT*), and reducing the graph.

Tables 4.3, 4.4, 4.5, and 4.6 respectively show the average running time of the *Uninformed Clustering Coverage (UCC)*, *Edge-based Clustering Coverage (ECC)*, *Node-based Clustering Coverage (NCC)*, and the *Cyclic Coverage (CC)* algorithms over all the maps and all number of robots, under the selected visual ranges of the robots and the two tour building algorithms.

The developed software is in Java and C++, and the simulations were run on a single-core Pentium 4 (3.2Ghz) desktop computer, with 3GB of memory.

4.8.2 Results for *Total Path Length*

Figure 4.13 shows the performance of the coverage algorithms under *Total Path Length* on the basis of the pre-determined visual ranges of the robots, and the choice of environment representation.

Effect of Robots' Visual Range: In all the tested visual ranges for the robots, at

Visual Range	0.25m	0.5m	0.75m	1m	1.5m
Reduced Graph	13352	1494	599	496	492

Table 4.2: *Building Reduced Graph Running Time (msec)*

Visual Range	0.25m	0.5m	0.75m	1m	1.5m
UCC-DMST	109	80	71	71	71
UCC-CLK	48628	3060	383	238	232

Table 4.3: *UCC Running Time (msec)*

Visual Range	0.25m	0.5m	0.75m	1m	1.5m
ECC-DMST	43725	3306	483	180	169
ECC-CLK	95369	9229	847	336	320

Table 4.4: *ECC Running Time (msec)*

Visual Range	0.25m	0.5m	0.75m	1m	1.5m
NCC-DMST	2048	295	102	81	79
NCC-CLK	49598	5434	481	260	240

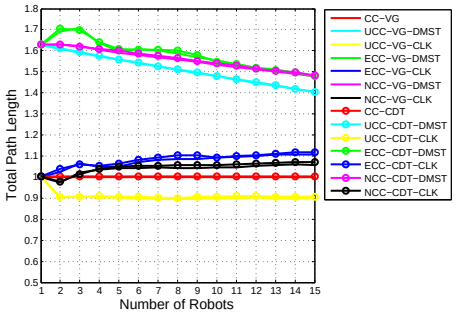
Table 4.5: *NCC Running Time (msec)*

Visual Range	0.25m	0.5m	0.75m	1m	1.5m
CC	711	210	129	121	114

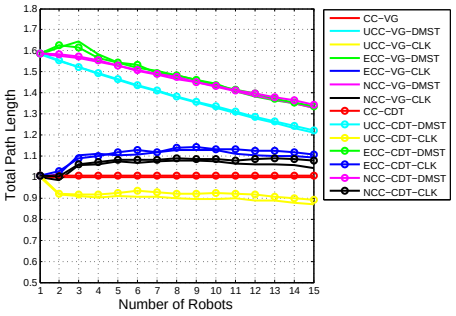
Table 4.6: *CC Running Time (msec)*

least one of the *Cluster-based* algorithms (*i.e.*, *Uninformed Clustering Coverage*) outperforms *Cyclic Coverage* ($p < 0.01$), and interestingly, as the visual range of the robots increases, there are more *Cluster-based* algorithms dominating *Cyclic Coverage*, especially in the scenarios in which more robots are involved.

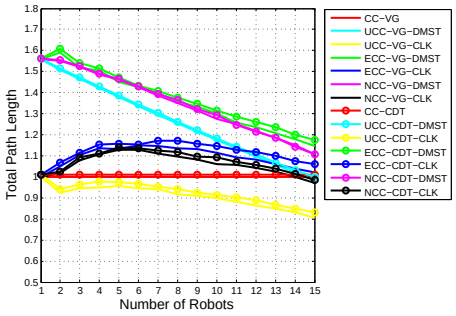
Increasing the robots' visual range leads to increase in the distance between the guards computed in the environments, and as a result increase in the the length



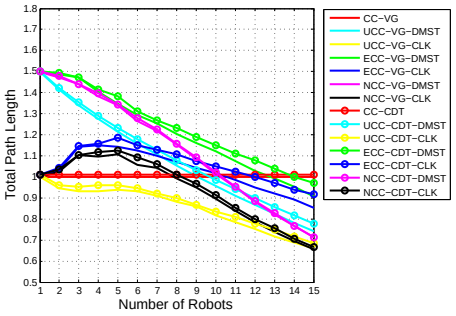
(a) *Robots' Visual Range = 0.25m*



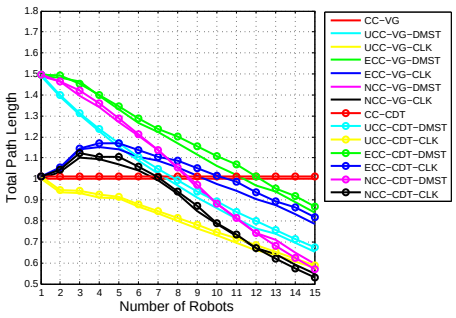
(b) *Robots' Visual Range = 0.5m*



(c) *Robots' Visual Range = 0.75m*



(d) *Robots' Visual Range = 1m*



(e) *Robots' Visual Range = 1.5m*

Figure 4.13: *Total Path Length. CC:Cyclic Coverage, UCC:Uninformed Clustering Coverage, ECC: Edge-based Clustering Coverage, NCC: Node-based Clustering Coverage, DMST:Double-MST, CLK:Chained Lin-Kernighan, VG:Visibility Graph, CDT:Constrained Delaunay Triangulation*

of the edges of the graph built on the environment. Subsequently, *Cluster-based* algorithms remove the edges in between the clusters, unless the resultant clusters overlap one another when using *Edge-based Clustering Coverage*. Increasing the visual range and the number of robots leads to, respectively, longer and more edges being removed from the *Reduced Graph*, and consequently improving the *Total Path Length* by the *Cluster-based* algorithms.

Effect of Environment Representation: As for the impact of the *VG* and the *CDT* on the performance of the algorithms, we did not find a significant difference between the two under *TPL*. However, as for the impact of the tour building algorithms, all the coverage methods perform significantly better under *CLK* than Double-MST $(p < 0.01)$, but this superiority declines with the increase of the robots' visual range and the number of robots. Increasing the robots' visual range leads to fewer guards being computed in the environment, and as a result decrease in the size of the graph and the *Reduced Graph* (in terms of the number of edges) built on the environment. Increasing the number of robots also leads to smaller clusters being built out of the *Reduced Graph*, and as the size of the clusters declines, the difference between *Double-MST* and *CLK* built on the clusters declines as well.

Conclusion: *Uninformed Clustering Coverage*, with no overlap among the tours it builds and removing the longest edges of the *Reduced Graph*, outperforms the other algorithms including *Cyclic Coverage* under *Total Path Length*. *Node-based Clustering Coverage* also dominates *Edge-based Clustering Coverage* $(p < 0.01)$. Overlaps among the tours generated by *Edge-based Clustering Coverage* dilutes the algorithm's performance under *Total Path Length*.

Overall, the results imply that although, given the set of guards in the original *VG* or *CDT* graph, *Cyclic Coverage* produces *optimal* or *near-optimal* solutions for *single-robot* cases, it is not the best solution when extending the problem to multi-robot scenarios.

Under *TPL*, the coverage algorithms show similar performance under the two graph representation algorithms (*i.e.*, *VG* and *CDT*), but the choice of tour building algorithm significantly affects the coverage approaches, in that using *CLK* leads to

shorter paths for the robots compared with *Double-MST*.

4.8.3 Results for *Total Average Visiting Period*

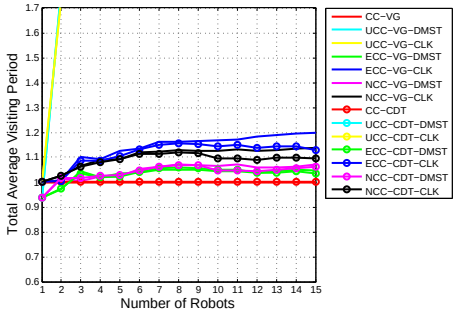
Figure 4.14 shows the performance of the coverage algorithms under *Total Average Visiting Period* on the basis of the pre-determined visual ranges of the robots, and the choice of environment representation.

Effect of Robots' Visual Range: As the visual range of the robots increases, the *Cluster-based* algorithms show better performance. For visual ranges of $0.75m$, $1m$, and $1.5m$, both the *Node-based Clustering Coverage* $(p < 0.01)$ and the *Edge-based Clustering Coverage* $(p < 0.01)$ algorithms dominate *Cyclic Coverage*, especially in the scenarios in which more robots are involved. However, for small visual ranges (*i.e.*, $0.25m$), *Cyclic Coverage* is the best choice of the coverage mission $(p < 0.01)$. For visual range of $0.5m$, *Node-based Clustering Coverage*, *Edge-based Clustering Coverage* and *Cyclic Coverage* are all in balance.

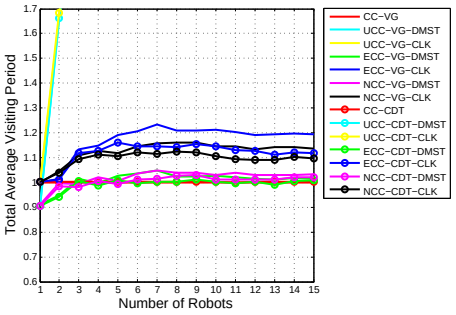
Similar to the *Total Path Length*, increasing the visual range and the number of robots leads to respectively longer and more edges being removed from the *Reduced Graph* by the *Cluster-based* algorithms, enabling them to outperform *Cyclic Coverage* under *Total Average Visiting Period*. *Node-based Clustering Coverage* and *Edge-based Clustering Coverage* also outperform *Uninformed Cyclic Coverage* $(p < 0.01)$ by building more balanced clusters (see Section 4.8.5), helping improve the *Total Average Visiting Period*.

Effect of Environment Representation: As for the impact of the *VG* and the *CDT* on the performance of the algorithms, we did not find a significant difference between the two under *TAVP*. However, as for the impact of the tour building algorithms, all the *Cluster-based* algorithms perform better under *Double-MST* than *CLK* $(p < 0.01)$. *Double-MST* leverages overlapped paths for the robots, and so improves the performance under *Total Average Visiting Period* compared to *CLK* which discourages overlapped paths.

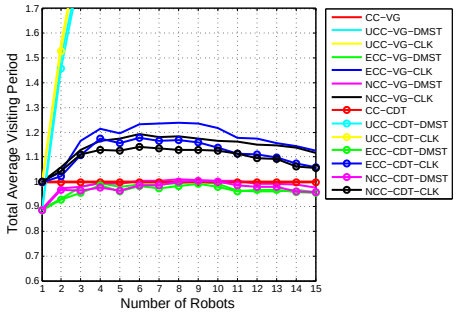
Conclusion: *Node-based Clustering Coverage* and *Edge-based Clustering Cover-*



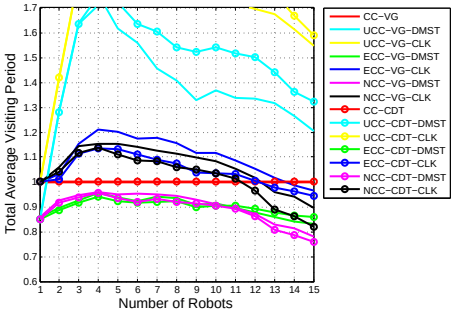
(a) *Robots' Visual Range = 0.25m*



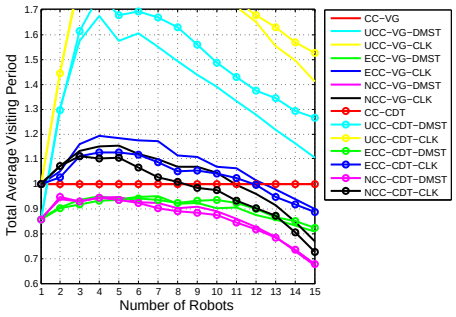
(b) *Robots' Visual Range = 0.5m*



(c) *Robots' Visual Range = 0.75m*



(d) *Robots' Visual Range = 1m*



(e) *Robots' Visual Range = 1.5m*

Figure 4.14: *Total Average Visiting Period. CC:Cyclic Coverage, UCC:Uninformed Clustering Coverage, ECC: Edge-based Clustering Coverage, NCC: Node-based Clustering Coverage, DMST:Double-MST, CLK:Chained Lin-Kernighan, VG:Visibility Graph, CDT:Constrained Delaunay Triangulation*

age are the best options when working with robots having medium and large visual ranges, and *Cyclic Coverage* is the choice of the coverage mission for robots with small visual ranges.

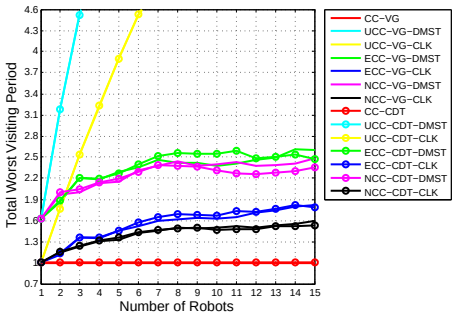
Under *TAVP*, the coverage algorithms show similar performance under the two graph representation algorithms (*i.e.*, *VG* and *CDT*), but the choice of the tour building algorithm significantly affects the coverage approaches, such that using *Double-MST* outperforms *CLK* in minimizing the *Total Average Visiting Period* of the points in the target area.

4.8.4 Results for *Total Worst Visiting Period*

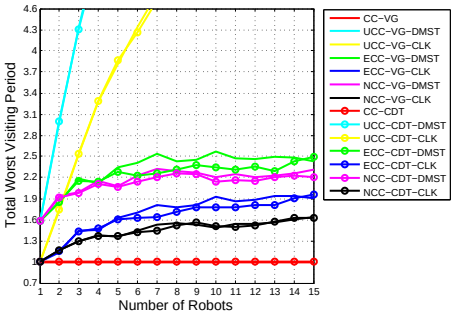
Figure 4.15 shows the performance of the coverage algorithms under *Total Worst Visiting Period* on the basis of the pre-determined visual ranges of the robots, and the choice of environment representation.

Effect of Robots' Visual Range: *Cyclic Coverage* dominates the *Cluster-based* algorithms under *Total Worst Visiting Period* in all the tested visual ranges for the robots $(p < 0.01)$. *Node-based Clustering Coverage* also outperforms the other *Cluster-based* algorithms $(p < 0.01)$. *Node-based Clustering Coverage* builds more balanced clusters compared to *Uninformed Clustering Coverage* (see Section 4.8.5), and builds clusters with no overlap compared to *Edge-based Clustering Coverage*. More balanced and less overlapped clusters helps improve the *Total Worst Visiting Period*.

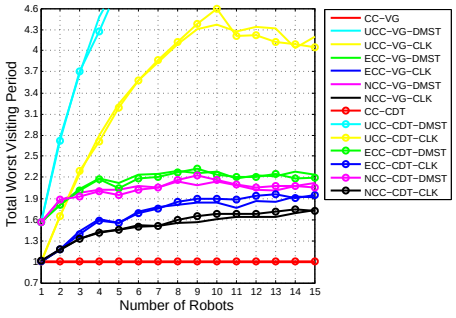
Effect of Environment Representation: As for the impact of the *VG* and the *CDT* on the performance of the algorithms, we did not find significant difference between the two under *TWVP*. However, as for the impact of the tour building algorithms, all the coverage mechanisms perform significantly better under *CLK* than *Double-MST* $(p < 0.01)$, but this superiority declines with the increase of the robots' visual range and the number of robots. Increasing the robots' visual range leads to fewer guards being computed in the environment, and as a result decrease in the size of the graph and the *Reduced Graph* (in terms of the number of edges) built on the environment. Increasing the number of robots also leads



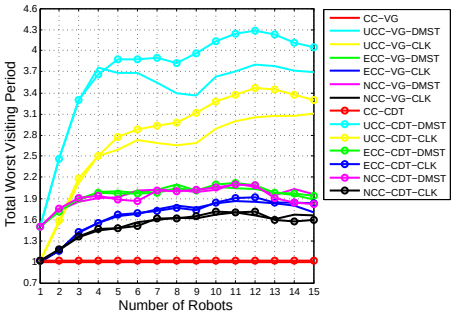
(a) *Robots' Visual Range = 0.25m*



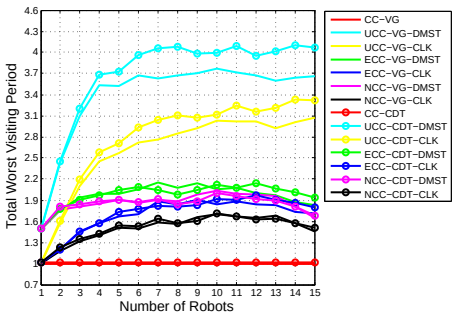
(b) *Robots' Visual Range = 0.5m*



(c) *Robots' Visual Range = 0.75m*



(d) *Robots' Visual Range = 1m*



(e) *Robots' Visual Range = 1.5m*

Figure 4.15: *Total Worst Visiting Period. CC:Cyclic Coverage, UCC:Uninformed Clustering Coverage, ECC: Edge-based Clustering Coverage, NCC: Node-based Clustering Coverage, DMST:Double-MST, CLK:Chained Lin-Kernighan, VG:Visibility Graph, CDT:Constrained Delaunay Triangulation*

to smaller clusters being built out of the *Reduced Graph*, and as the size of the clusters declines, the difference between *Double-MST* and *CLK* built on the clusters declines as well.

Conclusion: *Cyclic Coverage* is the best choice for minimizing the *Total Worst Visiting Period*, regardless of the visual range of the robots.

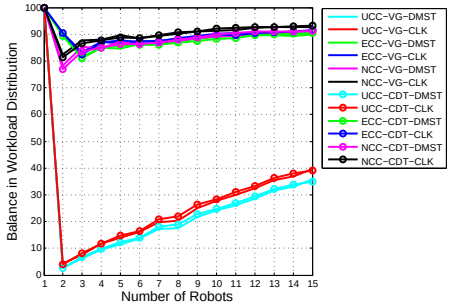
Under *TWVP*, the coverage algorithms show similar performance under the two graph representation algorithms (*i.e.*, *VG* and *CDT*), but the choice of tour building algorithm affects the coverage approaches especially in small visual ranges, such that using *CLK* outperforms *Double-MST* in minimizing the *Total Worst Visiting Period* of the points in the target area.

4.8.5 Results for *Balance in Workload Distribution*

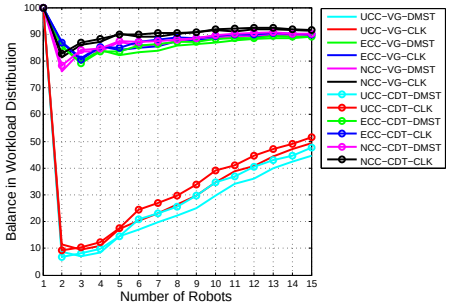
Figure 4.16 shows the performance of the coverage algorithms under *Balance in Workload Distribution* on the basis of the pre-determined visual ranges of the robots, and the choice of environment representation.

Effect of Robots' Visual Range: For small visual ranges (*i.e.*, $0.25m$, $0.50m$), we noticed slight improvements in the *Balance in Workload Distribution* with the increase in the number of robots in the environment. However, for visual range of $0.75m$, this improvement disappears for the *Node-based Clustering Coverage* and the *Edge-based Clustering Coverage* algorithms, and for visual ranges of $1m$ and $1.5m$, the *Balance in Workload Distribution* declines in both the algorithms, especially in *Node-based Clustering Coverage* and in scenarios in which more robots are involved.

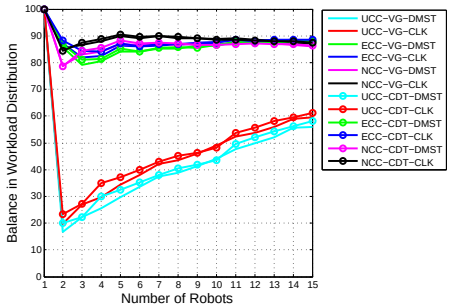
As discussed before, increasing the robots' visual range leads to fewer guards being computed in the environment, and as a result decrease in the size of the graph and the *Reduced Graph* (in terms of the number of edges) built on the environment. Conseuqntly, increasing the number of robots (*i.e.*, increasing the number of clusters) in the environment makes building balanced clusters out of the *Reduced Graph* more difficult. *Edge-based Clustering Coverage* is less affected by this issue, as it maintains some overlaps among the clusters.



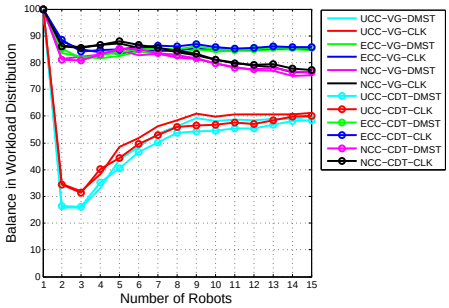
(a) *Robots' Visual Range = 0.25m*



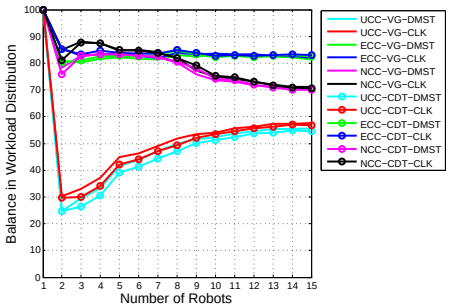
(b) *Robots' Visual Range = 0.5m*



(c) *Robots' Visual Range = 0.75m*



(d) *Robots' Visual Range = 1m*



(e) *Robots' Visual Range = 1.5m*

Figure 4.16: *Balance in Workload Distribution. UCC:Uninformed Clustering Coverage, ECC: Edge-based Clustering Coverage, NCC: Node-based Clustering Coverage, DMST:Double-MST, CLK:Chained Lin-Kernighan, VG:Visibility Graph, CDT:Constrained Delaunay Triangulation*

Effect of Environment Representation: As for the impact of the *VG* and the *CDT* on the performance of the algorithms, we did not find any significant difference between the two under *BWD*. However, as for the impact of the tour building algorithms, all the *Cluster-based* algorithms perform slightly better under *CLK* than *Double-MST* ($p < 0.01$).

Conclusion: For small visual ranges, *Node-based Clustering Coverage* is the best choice for balancing the workload distribution among the robots in the coverage mission; however, with the increase of the visual range, *Edge-based Clustering Coverage* dominates the *Node-based Clustering Coverage* algorithm for maximizing the *Balance in Workload Distribution*.

Under *BWD*, the coverage algorithms show similar performance under the two graph representation algorithms (*i.e.*, *VG* and *CDT*), but the choice of tour building algorithm affects the coverage approaches, such that using *CLK* slightly outperforms the *Double-MST* in maximizing the *Balance in Workload Distribution* among the robots.

4.9 Summary and Conclusions

We have addressed the problem of repeated coverage of a target area, of any polygonal shape, by a team of robots each having a limited circular visual range. Four optimization criteria were defined to evaluate the performance of the robot team in the target areas. These metrics include *Total Path Length (TPL)*, *Total Average Visiting Period (TAVP)*, *Total Worst Visiting Period (TWVP)*, and the *Balance in Workload Distribution (BWD)*.

Three distributed *Cluster-based* algorithms, namely: *Uninformed Clustering Coverage*, *Edge-based Clustering Coverage*, *Node-based Clustering Coverage* were introduced for the problem. *Cyclic Coverage*, used as a benchmark to compare the algorithms, produces *optimal* or *near-optimal* solutions for the *single-robot* case, in the *VG* or *CDT* graph built on the workspace, and under *TPL* and TWVP.

We conducted an extensive experimental analysis to evaluate the performance

of the algorithms. In summary:

- Under all the optimization criteria, the coverage algorithms show similar performance under the two graph representation algorithms, *VG* and *CDT*.
- Under *TPL*, *TWVP*, and *BWD*, all the *Cluster-based* algorithms perform better under *CLK* than *Double-MST*.
- Under *TAVP*, all the *Cluster-based* algorithms perform better under *Double-MST* than *CLK*.
- Under *TPL*, in all the tested visual ranges for the robots, at least one of the *Cluster-based* algorithms (*i.e.*, *Uninformed Clustering Coverage*) outperforms *Cyclic Coverage*, and as the visual range of the robots increases, there are more *Cluster-based* algorithms dominating *Cyclic Coverage*, especially in the scenarios in which more robots are involved.
- Under *TAVP*, *Node-based Clustering Coverage* and *Edge-based Clustering Coverage* are the best options when working with robots having medium and large visual ranges, and *Cyclic Coverage* is the choice of the coverage mission for robots with small visual ranges.
- Under *TWVP*, *Cyclic Coverage* dominates the *Cluster-based* algorithms in all the tested visual ranges for the robots.
- Under *BWD*, for small visual ranges, *Node-based Clustering Coverage* is the best choice for balancing the workload distribution among the robots; however, with the increase of the visual range, *Edge-based Clustering Coverage* dominates the *Node-based Clustering Coverage* algorithm for maximizing the *BWD*.

The results can be used as a framework for choosing an appropriate combination of repeated coverage algorithm, environment representation, and the robots' visual range based on the particular workspace and the metric to be optimized.

Chapter 5

On Multi-Robot Repeated Boundary Coverage

5.1 Problem Definition and Preliminaries

In this chapter, we address the *Multi-Robot Repeated Boundary Coverage* problem with the following specifications:

5.1.1 Workspace

Assumption 5.1. *The workspace, $\mathcal{W} \subset \mathbb{R}^2$, is a given polygon containing polygonal structures.*

5.1.2 Robots

Assumption 5.2. *The robots, $\mathcal{R}$, are assumed to have a 360° field of view and a predefined circular limit of visual range.*

Assumption 5.3. *Line-of-sight communication is assumed among the robots*[a]*. This type of communication can only occur when the transmitting and the receiving robots are in direct view of each other, with no obstacle between them.*

[a]We previously investigated the robots having limited circular communication range in [50] and [51].

5.1.3 Events

Assumption 5.4. *The events may occur at any position on the boundaries.*

Assumption 5.5. *The events may have different types. Each event type has its own importance weight.*

Definition 5.1. *Event Type: m* types of events may occur on the boundaries. The set of all event types is $E = \{E_1, E_2, ..., E_m\}$. Similarly, an event of type E_i is denoted as e_i.

Definition 5.2. *Event Importance:* The importance of an event of type E is given by $weight(E)$. It is assumed that $weight(E) \in (0,1]$ such that 1 is the highest degree of importance. The importance can also be referred to as the *priority*, in that an event of higher importance should have higher priority to be detected.

Assumption 5.6. *The boundaries are heterogeneous, in that events of one type may occur with varied probabilities on different parts of the boundaries, and this probability may change over time.*

Assumption 5.7. *A robot can detect an event, if the event is within the visual range of the robot.*

Assumption 5.8. *Once a robot detects an event, the event is discarded from the boundary. In other words, an event is detected only once.*

Assumption 5.9. *A robot is aware of the type of an event and its importance weight once it detects the event.*

Assumption 5.10. *The robots are not a priori aware of the probability distribution of the events on the boundaries.*

Assumption 5.11. *The reward a robot receives for detecting an event depends on how early the event is detected. At each time step after the event occurrence, the detection reward of the event is decreased by a multiplicative discount factor.*

The goal of the robot team is to maximize the total detection reward of the events.

As far as we are aware, there is no work using the boundary coverage framework studied in this chapter. In our work, instead of patrolling a single open or closed polyline, the robots patrol the inner boundaries of a full environment and the structures inside it, and it is assumed that different parts of the boundaries may have different priorities depending on the probability distribution of the events. Also, our robots can detect multiple events/intruders simultaneously, as opposed to single intruder scenarios studied in previous work.

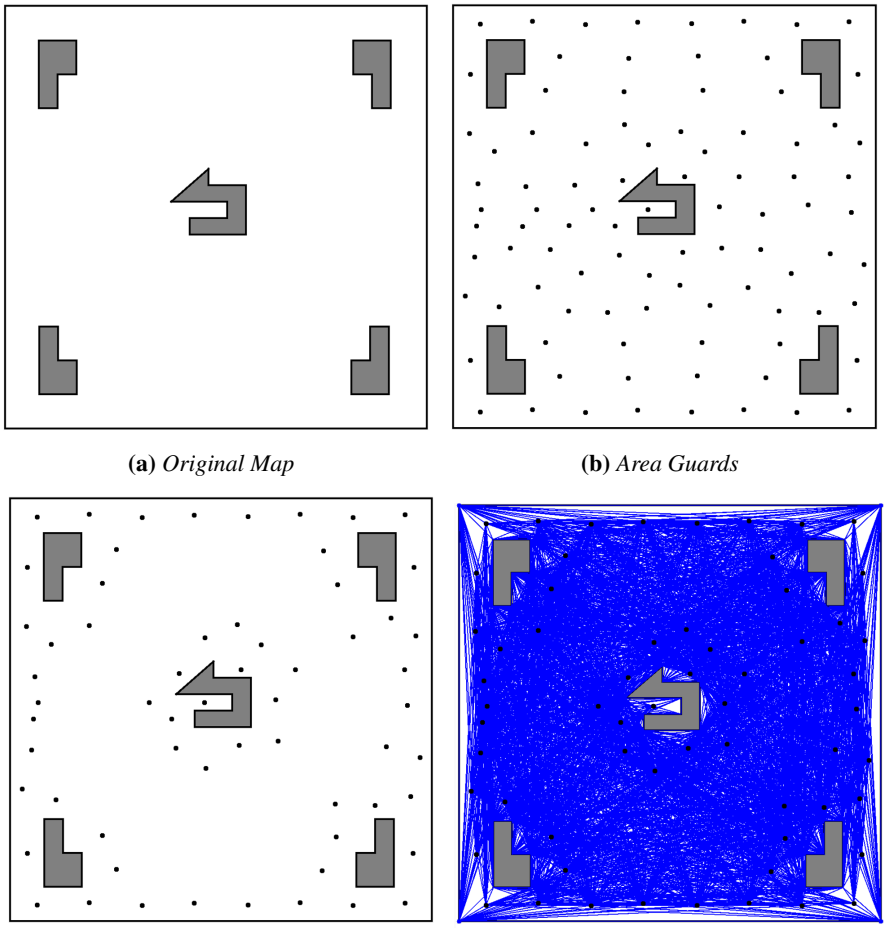
To address the problem, two classes of algorithms are proposed: (1) *Uninformed Boundary Coverage* and (2) *Informed Boundary Coverage*.

Uninformed Boundary Coverage uses a heuristic algorithm for the Traveling Salesman Problem to patrol the boundaries. On the other hand, *Informed Boundary Coverage* is primarily based on an online algorithm in which each robot autonomously learns the event distribution on the boundaries. Based on the policy being learned, each robot then plans in a distributed manner to select the best possible path to visit the most promising parts of the boundary.

The performance of the proposed algorithms is evaluated on the basis of the total reward received by the team during a finite period of time. We also investigate how robots' visual range, and communication among the robots affect the performance of the robot team in the coverage problem, and how event frequency affects the impact of communication on the robots' performance.

5.2 Environment Modeling

Uninformed Boundary Coverage and *Informed Boundary Coverage* both require that a roadmap is built within the workspace, capturing the connectivity of the free space close to the boundaries. To this end, a graph-based representation called the *Boundary Graph* is constructed on the workspace. The *Boundary Graph* enables the robots to move throughout the workspace to monitor the boundaries of the area and the structures inside. Since the workspace is known to the robots, each robot can independently build the *Boundary Graph* in the target area. In order to construct the *Boundary Graph*, a sufficient number of control points, called the *boundary guards*, are placed within the workspace, considering the limited visual range of the robots.



(a) *Original Map* **(b)** *Area Guards*

(c) *Boundary Guards* **(d)** *Boundary Graph*

Figure 5.1: *Sequential Stages of Building the Boundary Graph*

5.2.1 Locating Boundary Guards with Limited Visual Range

In our problem definition, we presume the robots are equipped with panoramic cameras with a 360° field of view. However, the cameras' visual range is limited. The proposed approach initially locates a set of guards within the workspace, according to the robots' limited visual range. As mentioned in Chapter 3, these static guards are control points from which the whole workspace can be jointly observed. In other words, if there are as many robots as there are guards, and each robot were

stationed on a guard, the entire area would be visible to the robots.

Since, in the current problem, we are interested in monitoring only the boundaries, not all the computed area guards are necessary. So, each guard such that the visual area of a robot does not intersect the boundaries when it is located on that guard is removed from the set of area guards. Figure 5.1c illustrates the *boundary guards (BG)* computed on the sample workspace of Figure 5.1a.

$$BG = \{g_1, g_2, ..., g_k\}. \tag{5.1}$$

5.2.2 Boundary Graph

Once the boundary guards are located in the target area, a *Visibility Graph (VG)* is constructed on the guards and the nodes of the workspace (Figure 5.1d). In order to build the *Visibility Graph*, any pair of nodes (*i.e.*, boundary guard or workspace node) which are mutually visible are connected by an edge. Two nodes are mutually visible if the edge connecting them does not intersect any structure within the workspace.

Visibility Graph is used to build the roadmap, because it provides the robots with more paths and more freedom of movement to traverse the target area, compared to other representations such as *Constrained Delaunay Triangulation* or *Voronoi Diagram*.

5.2.3 Boundary Segmentation

The boundaries of the area and the structures are divided into identical length *segments*, each of which is small enough such that if an event occurs in a *segment*, the event is visible from any part of that *segment*. In other words, if a robot's visual range covers just part of a *segment* on the boundary, the robot is still capable of detecting all the events occurring in any part of the *segment*. This fact is based on the assumption that events have some extension on the boundary.

$$Segments = \{seg_1, seg_2, ..., seg_n\}. \tag{5.2}$$

Definition 5.3. *Visual Area of a Guard (VA):* The *visual area* of a guard, $VA(g)$, is the set of all the *segments* visible to the robot when it is located on the guard g.

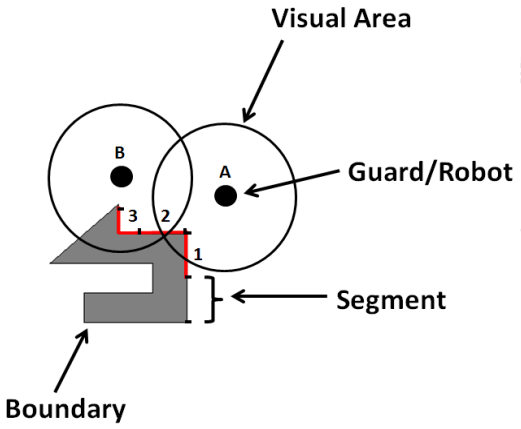


Figure 5.2: *Boundary Segmentation*

$$VA(g) = \{seg_i, seg_j, ..., seg_s\}. \tag{5.3}$$

Definition 5.4. *Shared Segment:* A *shared segment* is common to the *visual area* of two or more guards.

Definition 5.5. *Segment Parent Guards (SPG): Parent guards* of a *shared segment* are guards whose *visual area* contain that *segment*.

$$SPG(seg) = \{g_i, g_j, ..., g_p\}. \tag{5.4}$$

The notion of *segment parent guards* implies that an event occurred in a *segment* can be detected when and only when a robot is located at one of the *parent guards* of the *segment*. An event is detected only once.

Assumption 5.12. *Events are only detected when a robot is located on a guard.*

In Figure 5.2, the *visual area* of guard *A* covers *segments* 1 and 2, and the *visual area* of guard *B* covers *segments* 2 and 3. *Segment* 2 is a *shared segment* between guards *A* and *B*, and subsequently, guards *A* and *B* are the *parent guards* of *segment* 2. A robot located on guard *A* can detect the events that occurred in any part of *segments* 1 and 2, and a robot located on guard *B* can detect the events occurred in any part of *segments* 2 and 3.

Algorithm 5.1: Uninformed Boundary Coverage

Input:
$G_{Boundary}(V_{Boundary}, E_{Boundary})$, where $V_{Boundary} = BG \bigcup P$
$BG = \{g_1, g_2, ..., g_k\}$ `/* Boundary Guards */`
$P = \{p_1, p_2, ..., p_n\}$ `/* Workspace Nodes */`
$|R|$: Number of Robots

Output:
A tour, *dTour*, distributed among the robots, passing through all the guards of the *Boundary Graph*

```
1 begin
2 |   tour ⟵ BuildTour(G_Boundary)
3 |   dTour ⟵ DistributeRobots(tour, |ℛ|)
4 |   return dTour
5 end
```

5.3 Uninformed Boundary Coverage

In *Uninformed Boundary Coverage* (*Algorithm* 5.1), a tour is constructed on the *Boundary Graph* using the *Chained Lin-Kernighan* algorithm.

The input of the *Chained Lin-Kernighan* algorithm is the distance matrix of the *Boundary Graph*. The matrix consists of the shortest path distances between all pairs of guards in the *Boundary Graph*, and is consequently indicative of a complete graph, even though the *Boundary Graph* itself is not complete. Having built the shortest tour passing through all the guards of the *Boundary graph*, the robots position themselves equidistantly around the tour and move repeatedly around it in the same direction.

5.4 Informed Boundary Coverage

In *Informed Boundary Coverage*, the robots try to maximize the total detection reward of the events. To this end, each robot independently learns the event distribution on the boundaries and estimates the expected reward of visiting a state in the target area at each time step. Each robot then plans in a decentralized manner to select the best possible path to visit the most promising states in the target area.

The initial locations of the robots are chosen randomly in the target area.

The *Multi-Robot Repeated Boundary Coverage* problem is formulated as a tuple (R, BG, A, ST, STR) where:

- R is the set of robots involved in the coverage mission.
- BG is the set of states or boundary guards, representing the position of the robots in the target area.
- A is the set of actions available for a robot in each state. An action is defined as moving from one guard to any other guard in the *Boundary Graph*. At the beginning, each robot calculates the shortest path between each pair of guards in the *Boundary Graph* using the *Floyd-Warshall* algorithm. Hence, the robots will not need to repeatedly compute the shortest paths in the graph during the planning stage of the coverage mission.
- ST is the state transition function which is deterministic, in that it guarantees reaching the target state chosen by the robots from the current state, when the action is performed.
- STR is the state reward at a particular time, which is the sum of the discounted importance of the events at the state (*i.e.,* boundary guard):

$$STR(g,t) = \sum_{seg \in VA(g)} \sum_{E_i \in E} \sum_{e_i \in E_i} weight(E_i) \times \gamma^{t-st(e_i)}, \tag{5.5}$$

  where $t - st(e_i)$ is the time interval between starting event e_i and the time t. Once a robot arrives at a guard g, it can detect all the events that occurred within the $VA(g)$, the *visual area* of the guard g. It is assumed that the reward a robot receives for an event depends on how early the event is detected. At each time step after the event occurrence, the detection reward of the event is multiplied by a discount factor of $\gamma = 0.95$. The discount factor, γ, was chosen by a trial and error procedure.

Definition 5.6. *Time of Last Visit (TLV):* Each robot separately keeps track of the times of the last visit to the guards. If $BG = \{g_1, g_2, ..., g_k\}$ is the set of boundary

guards, then for each guard $g \in BG$, $TLV_r(g)$ represents the last time the guard g was visited by robot $r \in R$, or by any other robot who managed to communicate its visit to the guard to robot r. Therefore, the times of the last visit to the guards are not globally shared by the robot team, rather each robot, at each time step, may have different beliefs from the rest of the robot team of the times of the last visit to the guards.

Subsequently, robot r can calculate the time of the last visit to each *segment* of the boundary:

$$TLV_r(seg) = max\{TLV_r(g) | g \in SPG(seg)\}. \tag{5.6}$$

Intuitively, the time of the last visit to a *segment* is the most recent visit of robot r, or any other robot who communicates to robot r its visit to one of the *segment*'s *parent guards*.

Definition 5.7. *Policy:* A policy $\pi(r) : BG \rightarrow A$ at each state determines which action should be performed next by robot r.

Note that the learning procedure described below is performed by each robot independently of the rest of the team.

5.4.1 Learning

If robot r had complete knowledge of the starting time of the events in each state, it would be able to calculate the *STR* to find a policy, maximizing the total reward of the boundary coverage mission, but since this information is not available to the robot, it estimates the *STR* as the sum of the *Expected Segment Reward (ESR)* of the *segments* comprising a state:

$$STR_r(g,t) \simeq \sum_{seg \in VA(g)} (ESR_r(seg,t)). \tag{5.7}$$

Expected Segment Reward (ESR) is defined to represent the expected reward of a *segment*, *seg*, at the time t:

$$ESR_r(seg,t) = \sum_{E_i \in E} \sum_{e_i \in E_i} (1 + \gamma^1 + \gamma^2 + ... + \gamma^{t-TLV_r(seg)}) \times PSE_r(E_i, seg) \times weight(E_i), \tag{5.8}$$

where γ is the reward discount factor. We assume that for every time step after an event occurs without being detected, the event detection reward is discounted by γ. Furthermore, the *Probability of Segment Event (PSE)* is defined for each event type $E_i \in E$ and each *segment*, *seg*, to indicate the probability of events of type E_i occur within the *segment* at each time step.

In equation (8), $\sum_{E_i \in E} \sum_{e_i \in E_i} PSE_r(E_i, seg) \times weight(E_i)$ is the *Segment Reward Accumulation Rate* of the events in the *segment*, *seg*, and is represented by $SRAR_r(seg)$. If robot r knows the *SRAR* of the events in a *segment*, it can calculate the *segment*'s *ESR* for any arbitrary time t.

Estimating the SRAR of the Segments

In the initialization step, the robot assumes the *SRAR* of all the *segments* is 1. A learning procedure for estimating the *SRAR* of the *segments* gradually updates their initial value. When the robot arrives at a guard g, it can detect whether or not an event has occurred at the *segments* belonging to $VA(g)$. The *SRAR* of the guard's *segments* is then updated using the following equation:

$$\forall seg \in VA(g),\ SRAR_r(seg) = (1-\alpha) \times SRAR_r(seg) + \alpha \times \frac{\sum_{E_i \in E} \sum_{e_i \in E_i} (weight(E_i))}{t_r(g) - TLV_r(seg)}, \tag{5.9}$$

where α is the learning rate set to 0.9 and $t_r(g)$ is the time of the visit to the guard g by robot r. The learning rate, α, was chosen by a trial and error procedure. This equation gives more weight to the new information than the old information. The robot performs the updating process for all the event types and all the *segments* of the guards.

In summary, the *SRAR* of a *segment* is updated once the robot visits one of its *parent guards*. As already mentioned, the *SRAR* represents the reward accumulation rate in a *segment*. Now we can use the *SRAR* to estimate the *ESR* of the *segments* using the following procedure.

Estimating the ESR of the Segments

At the beginning of the mission, robot r initializes the *ESR* of all the *segments* to zero. Then, at each time step, if the robot has yet to arrive at a guard, the *ESR* of all the *segments* of the boundary is updated using the following equation:

$$\begin{aligned} &\forall seg \in Segments, \\ &ESR_r(seg,t) = \gamma \times ESR_r(seg,t-1) + SRAR_r(seg). \end{aligned} \tag{5.10}$$

If robot r arrives at a guard g, it detects all the events that have occurred in its *segments* and communicates the guard ID to the robots located in its line-of-sight. Since all the events occurred in the *segments* of g have been detected, the expected reward of the *segments* at the time of the visit to the guard, $t(g)$, becomes zero. Consequently, for the robot and all the communicated robots ($cR \subset R$):

$$\forall cr \in cR \; \forall seg \in VA(g), \; ESR_{cr}(seg, t_{cr}(g)) = 0. \tag{5.11}$$

$$\forall cr \in cR \; \forall seg \in VA(g), \; TLV_{cr}(seg) = Current \; Time. \tag{5.12}$$

Note that each robot has its own knowledge of the *ESR* of the *segments*, so the other robots (except the ones who received the communication) may still assume that there are some undetected events in the *segments* of g at the time, and subsequently their *ESR*s of the *segments* of g are not zero.

In the Experiments and Results section (Section 5.5), we will show how communication among the robots influences the performance of the algorithm and compare it with the case that robots do not communicate or share any knowledge of the target area during the operation.

The updating process of the *SRAR* and the *ESR* of the *segments* continues during the coverage operation.

5.4.2 Planning

Once a robot arrives at a guard and detects all the events which may have occurred in the *segments* of the guard, the robot selects the next action to perform. As already mentioned, an action is defined as moving from one guard to another guard

in the *Boundary Graph*. At each state, the robot considers all the precomputed shortest paths to all the other guards it can move to. For each path, $path(g_c, g_d)$, where g_c is the current guard and g_d is the destination guard, *Path Reward (PR)* is defined as the reward the robot receives when moving from the guard g_c to the guard g_d. The path from g_c to g_d includes zero or more intermediate guards and can be represented as:

$$path(g_c, g_d) = [g_c, g_i, g_j, ..., g_r, g_d]. \tag{5.13}$$

Given the speed of the robot, the arrival time at each of the guards on the path can be estimated. Hence, robot r can have an estimate of the $ESR_r(seg, t_r(g))$ for each *segment* of the guard $g \in path(g_c, g_d)$, in which $t_r(g)$ is the arrival time of the robot to the guard g. For such a path, robot r calculates the *Path Reward (PR)* as below:

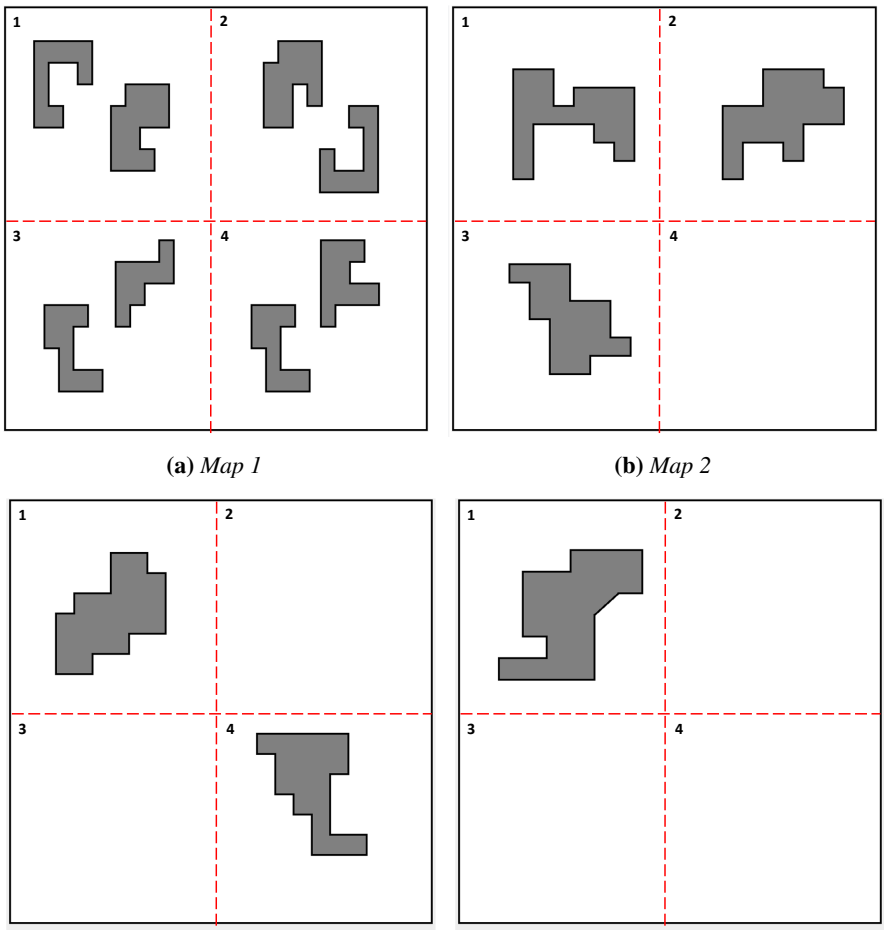
$$PR_r(path(g_c, g_d)) = \sum_{g \in path(g_c, g_d)} \sum_{seg \in VA(g)} ESR_r(seg, t_r(g)). \tag{5.14}$$

When calculating the *PR*, the robot should take into account the *segments* shared by some *parent guards* as well, namely the robot in its calculations initializes the *shared segment*'s *ESR* to zero when it is going to visit one of its *parent guards* along the path.

Next, for each path, the *Average Path Reward (APR)* is calculated using the following equation:

$$APR_r(path(g_c, g_d)) = \frac{PR_r(path(g_c, g_d))}{t_r(g_d) - TLV_r(g_c)}, \tag{5.15}$$

where g_c is the current guard, $TLV_r(g_c)$ is the current time (*i.e.*, the time of the last visit to the guard g_c by robot r), g_d is the destination guard, and $t_r(g_d)$ is the arrival time of robot r to the guard g_d. The robot will select a path with the maximum *Average Path Reward* to traverse next.



(a) *Map 1* **(b)** *Map 2*

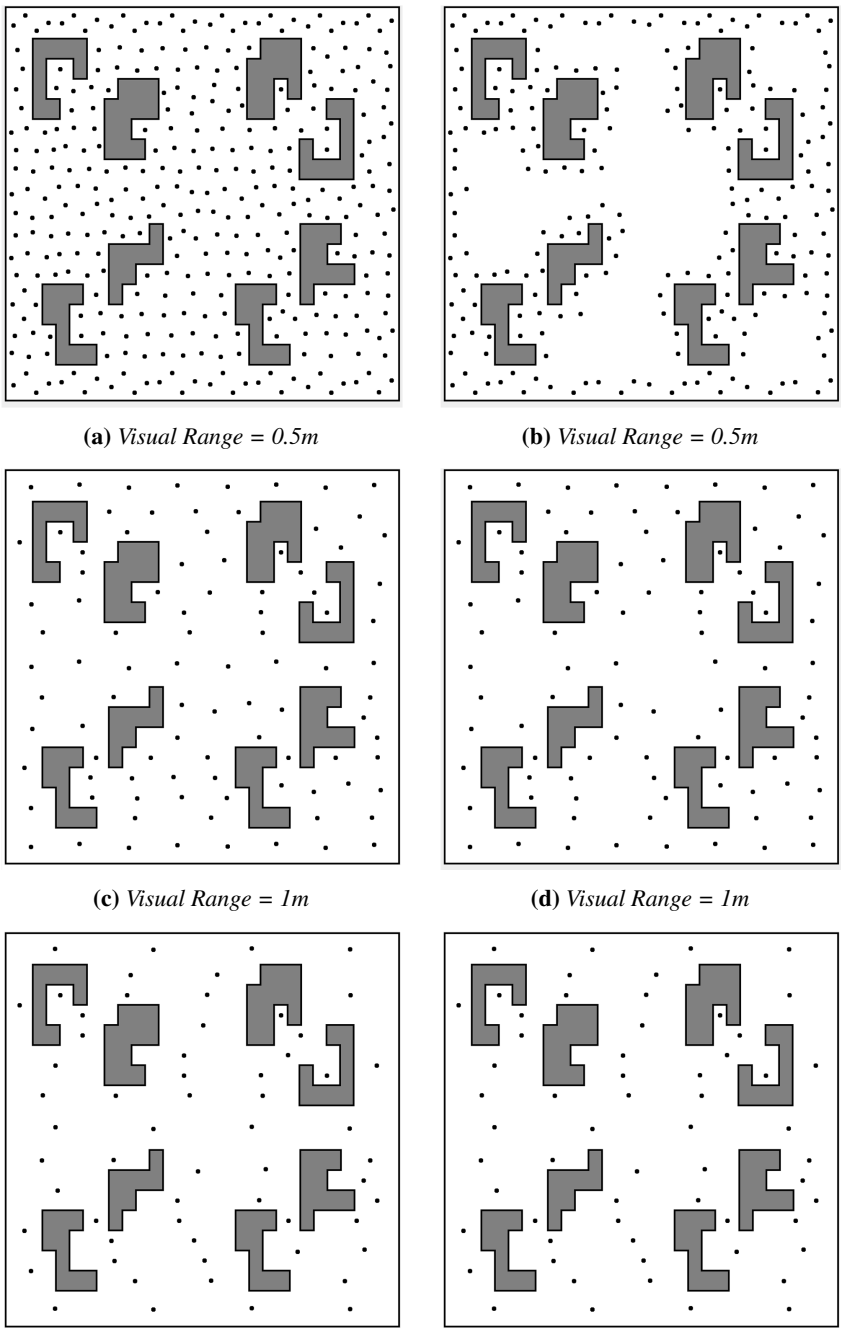
(c) *Map 3* **(d)** *Map 4*

Figure 5.3: *Maps Used in the Experiments*

5.5 Evaluation and Experimental Simulations

We wish to compare *Uninformed Boundary Coverage (UBC)* with the two variants of *Informed Boundary Coverage (IBC)* (i.e. non-communicating and communicating robot teams) in terms of the total reward being received by the team for detecting the events in a finite simulation time.

The experiments are conducted using 10 robots on the sample workspaces of Figure 5.3. The size of the workspaces is $10m \times 10m$, and the robots move with a



(a) *Visual Range = 0.5m*

(b) *Visual Range = 0.5m*

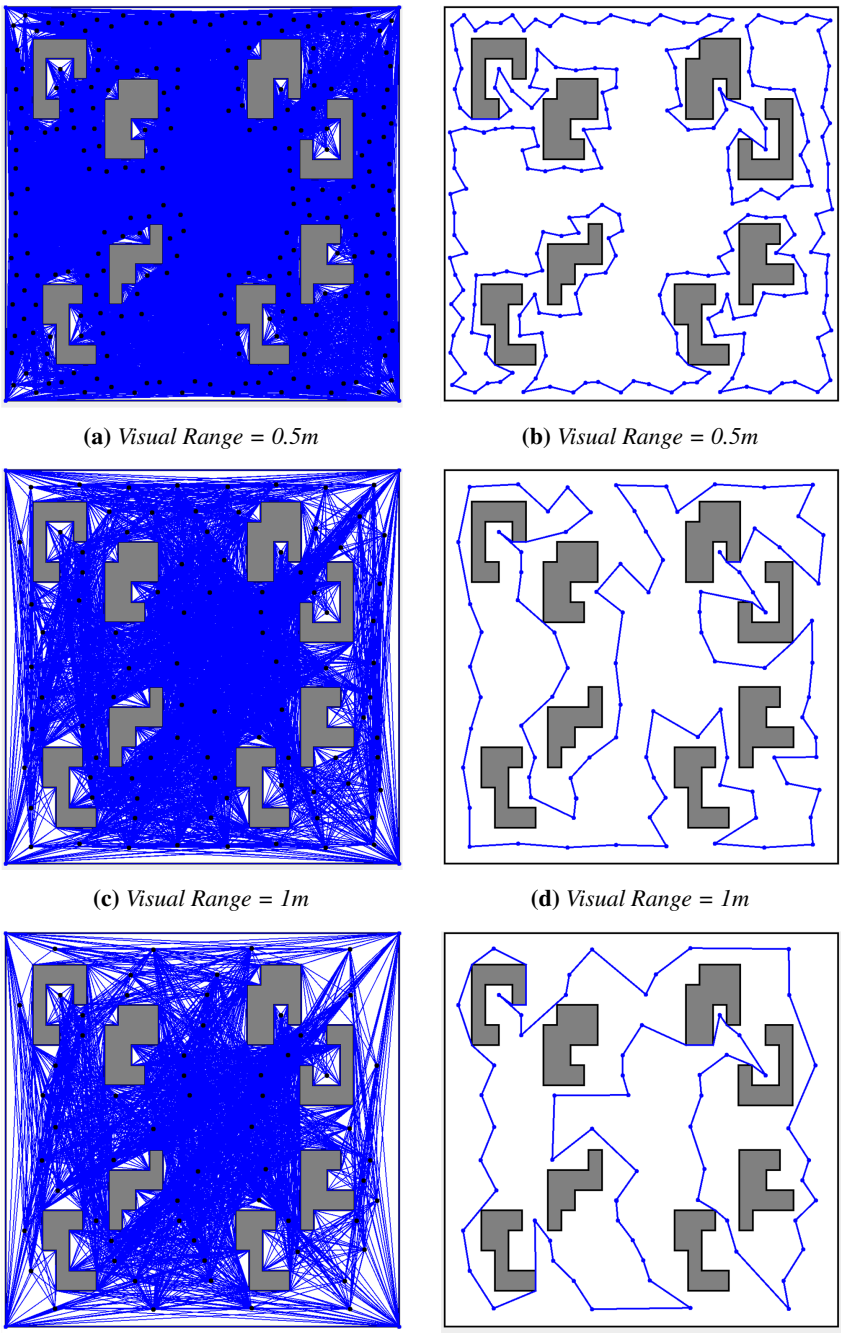
(c) *Visual Range = 1m*

(d) *Visual Range = 1m*

(e) *Visual Range = 1.5m*

(f) *Visual Range = 1.5m*

Figure 5.4: *Map 1: Area Guards + Boundary Guards*



(a) *Visual Range = 0.5m*

(b) *Visual Range = 0.5m*

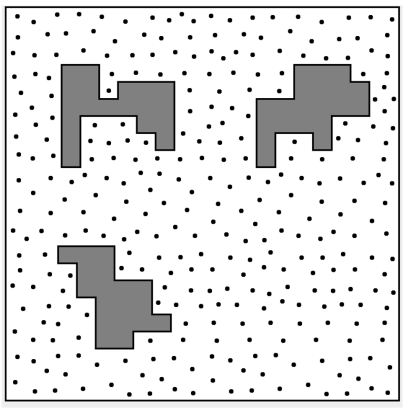
(c) *Visual Range = 1m*

(d) *Visual Range = 1m*

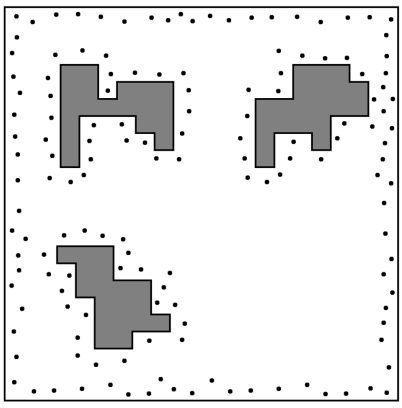
(e) *Visual Range = 1.5m*

(f) *Visual Range = 1.5m*

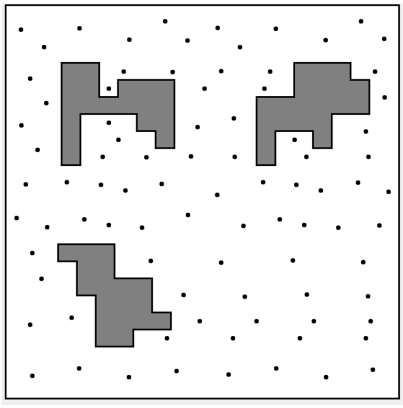
Figure 5.5: *Map 1: Boundary Graph + Tour Computed by UBC*



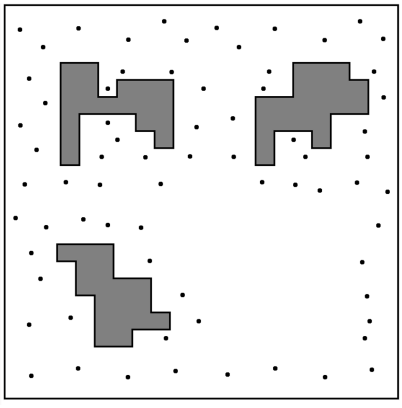
(a) *Visual Range = 0.5m*



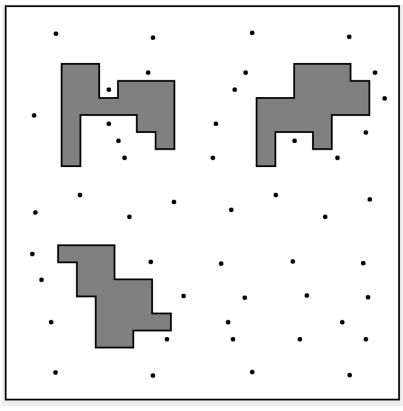
(b) *Visual Range = 0.5m*



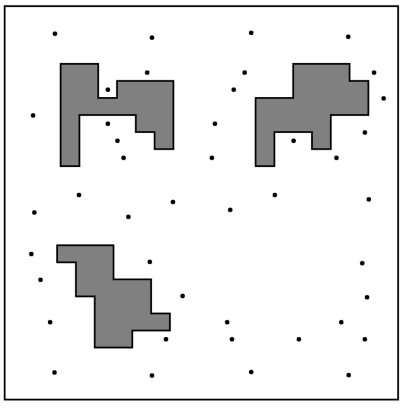
(c) *Visual Range = 1m*



(d) *Visual Range = 1m*

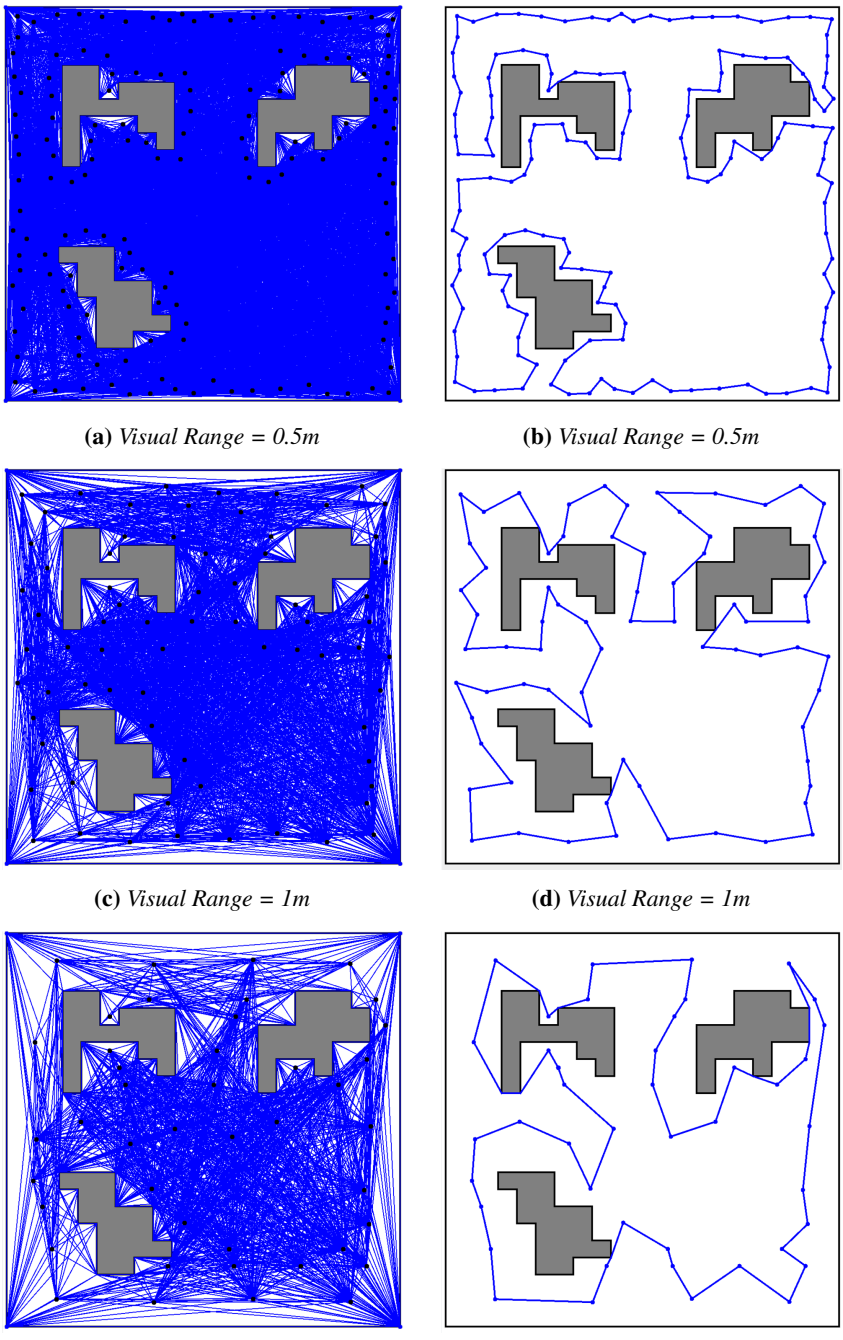


(e) *Visual Range = 1.5m*



(f) *Visual Range = 1.5m*

Figure 5.6: *Map 2: Area Guards + Boundary Guards*



(a) *Visual Range = 0.5m*

(b) *Visual Range = 0.5m*

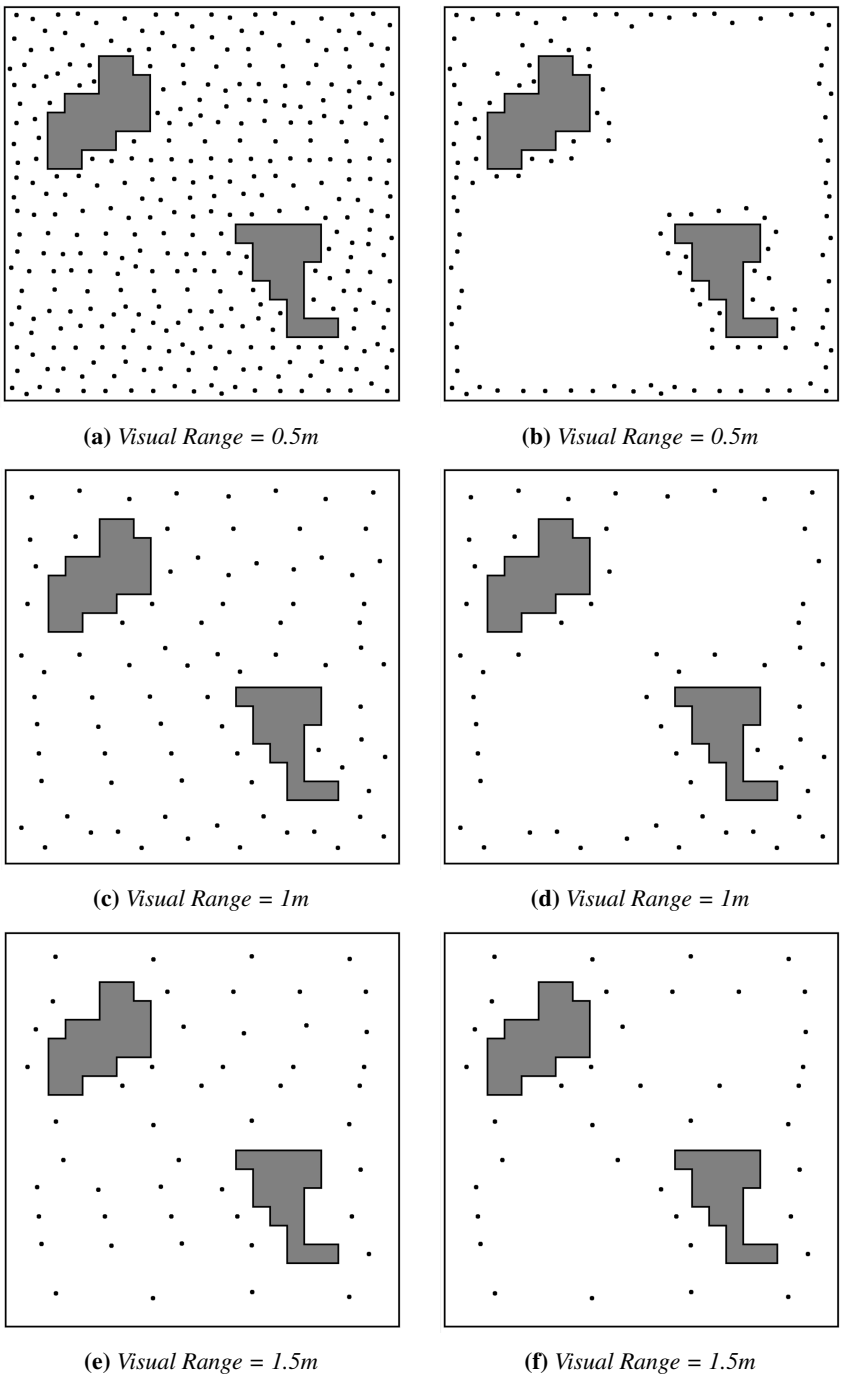
(c) *Visual Range = 1m*

(d) *Visual Range = 1m*

(e) *Visual Range = 1.5m*

(f) *Visual Range = 1.5m*

Figure 5.7: *Map 2: Boundary Graph + Tour Computed by UBC*



(a) *Visual Range = 0.5m*

(b) *Visual Range = 0.5m*

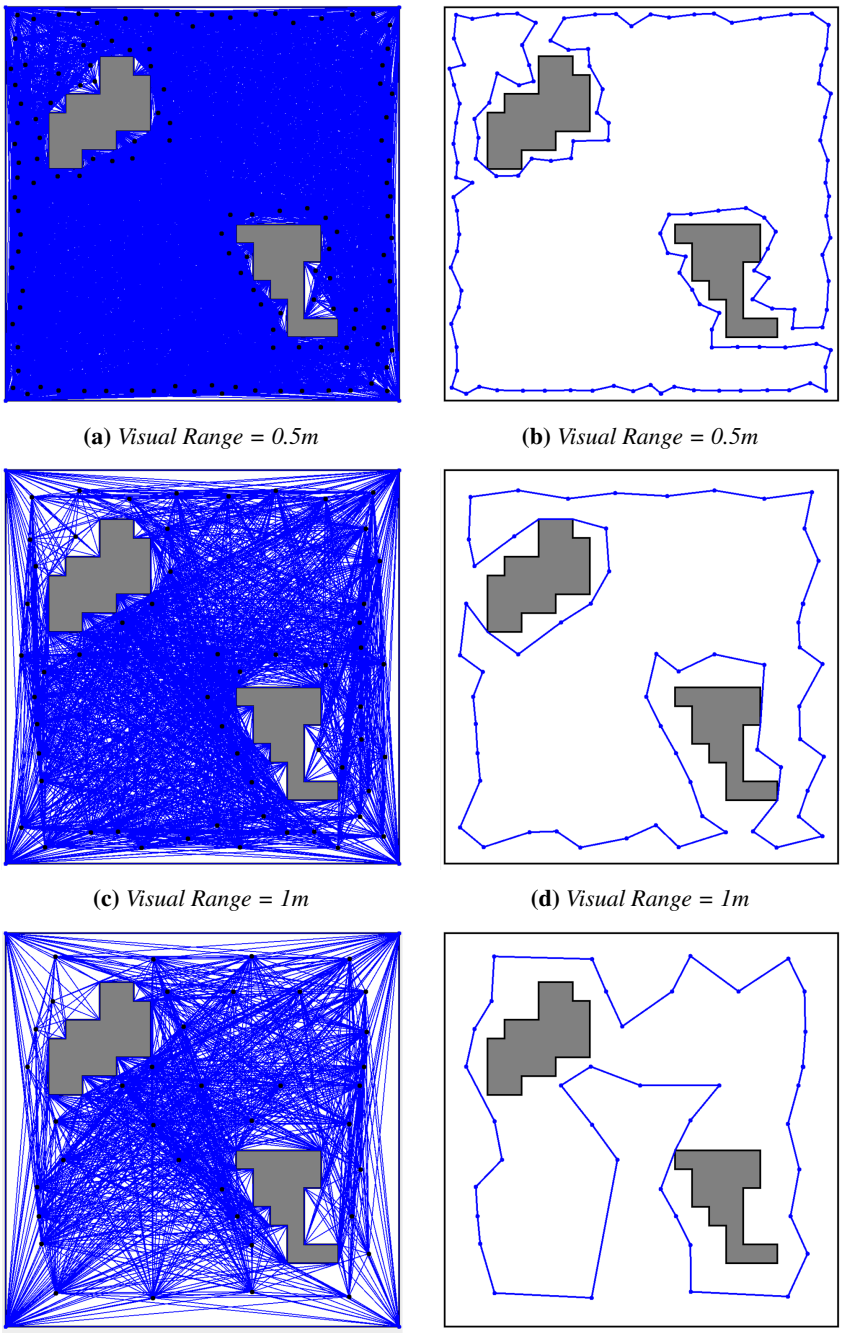
(c) *Visual Range = 1m*

(d) *Visual Range = 1m*

(e) *Visual Range = 1.5m*

(f) *Visual Range = 1.5m*

Figure 5.8: *Map 3: Area Guards + Boundary Guards*



(a) *Visual Range = 0.5m*

(b) *Visual Range = 0.5m*

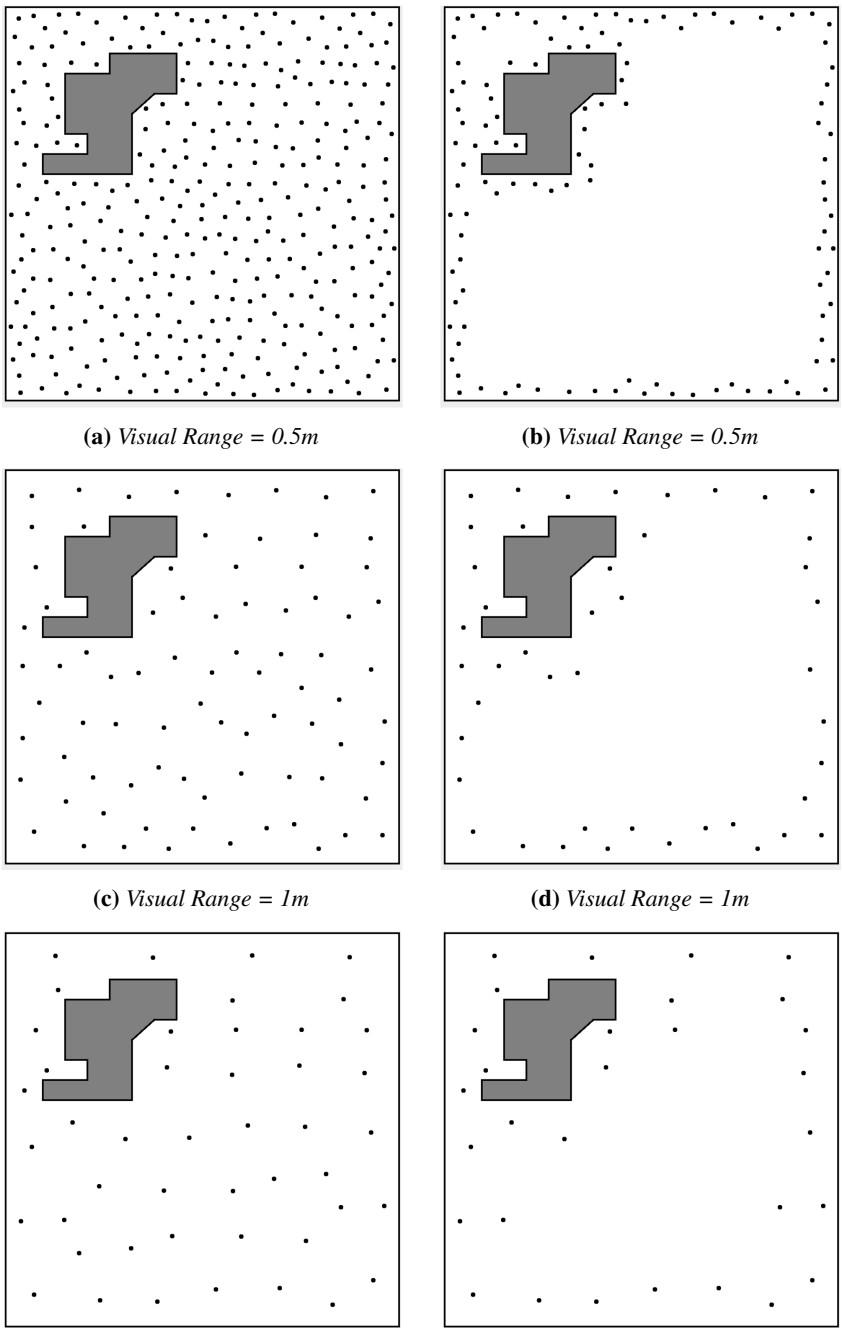
(c) *Visual Range = 1m*

(d) *Visual Range = 1m*

(e) *Visual Range = 1.5m*

(f) *Visual Range = 1.5m*

Figure 5.9: *Map 3: Boundary Graph + Tour Computed by UBC*



(a) *Visual Range = 0.5m*

(b) *Visual Range = 0.5m*

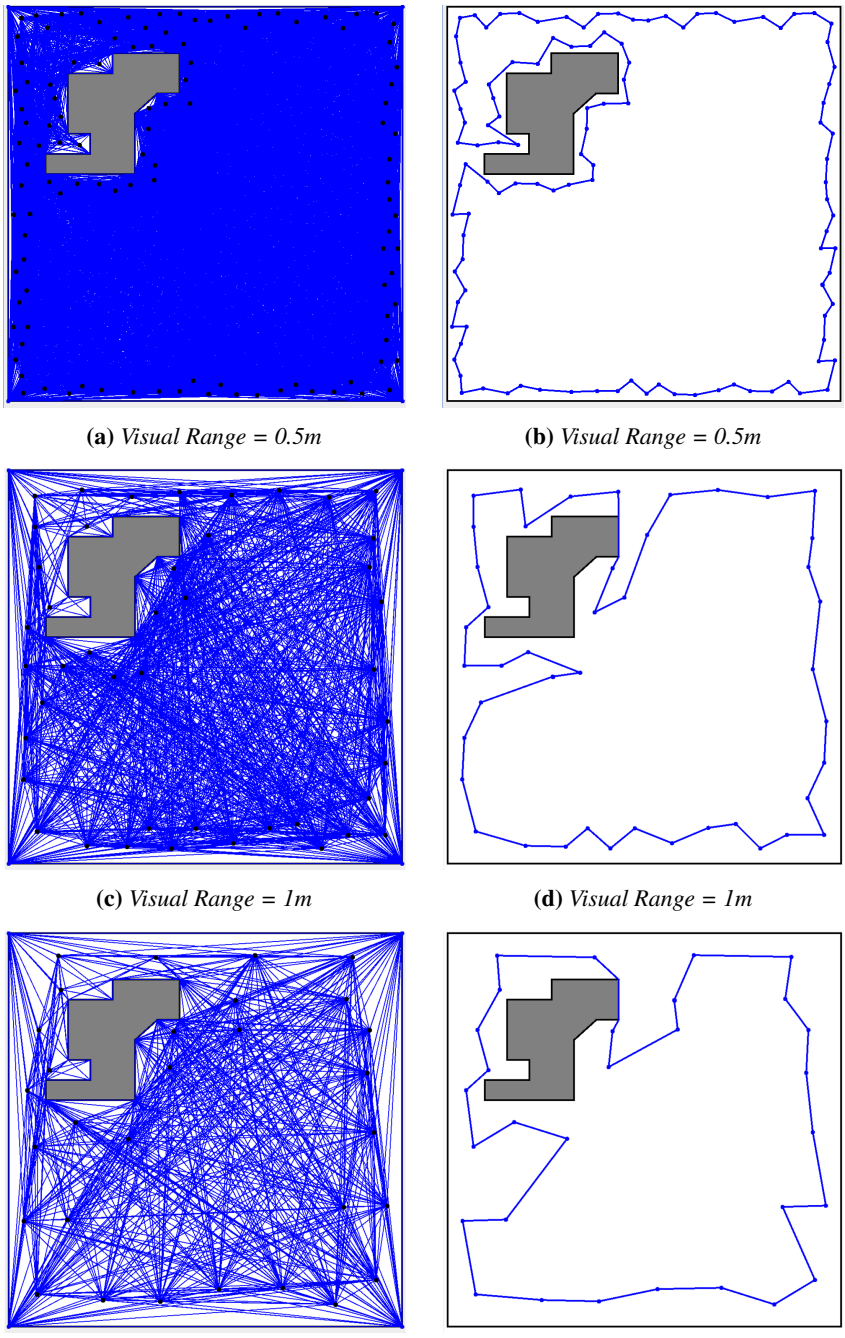
(c) *Visual Range = 1m*

(d) *Visual Range = 1m*

(e) *Visual Range = 1.5m*

(f) *Visual Range = 1.5m*

Figure 5.10: *Map 4: Area Guards + Boundary Guards*



(a) *Visual Range = 0.5m*

(b) *Visual Range = 0.5m*

(c) *Visual Range = 1m*

(d) *Visual Range = 1m*

(e) *Visual Range = 1.5m*

(f) *Visual Range = 1.5m*

Figure 5.11: *Map 4: Boundary Graph + Tour Computed by UBC*

speed of $0.1m/cycle$. Three different visual ranges are considered for the robots: $0.5m$ (low), $1m$ (medium), $1.5m$ (high). Figures 5.4, 5.5, 5.6, 5.7, 5.8, 5.9, 5.10, and 5.11 show the area guards, boundary guards, *Visibility Graphs*, and the tours computed by *Uninformed Boundary Coverage* on the four maps, under various visual ranges of the robots.

The workspace maps are divided into four disjoint sub-regions. In map 1, there are structures in four sub-regions of the workspace. In map 2, there are structures in three of the sub-regions. In map 3, there are structures in two of the sub-regions, and in map 4, there are structures in just one of the sub-regions.

Four experiments are designed, in each the event distribution varies in the sub-regions. For each experiment, two low- and high-frequency event models are assumed. In the high-frequency model, the rate at which the events occur is five times the rate of event occurrence in the low-frequency model. In these experiments, we investigate how robots' visual range, and communication among the robots affect the performance of the robot team in the coverage problem, and how event frequency affects the impact of communication on the robots' performance.

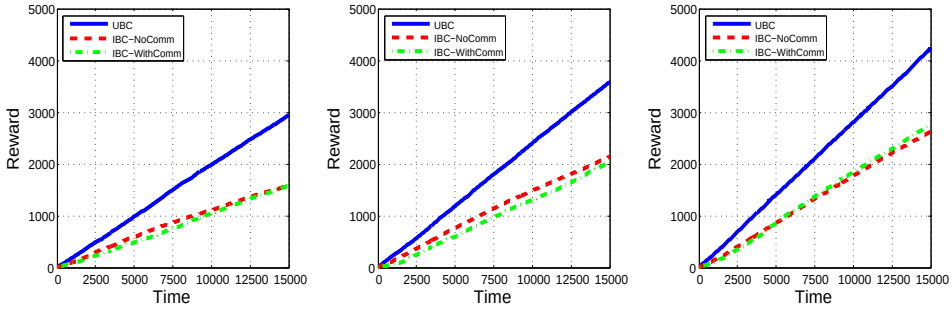
Below we examine the results obtained on map 1. In the Discussion and Conclusions section (Section 5.6), it is shown that the other maps follow a similar trend to that for map 1. The results are based on the average of 10 runs on each map.

5.5.1 Experiment 1: Uniform Event Occurrence

Low-Frequency Event Model

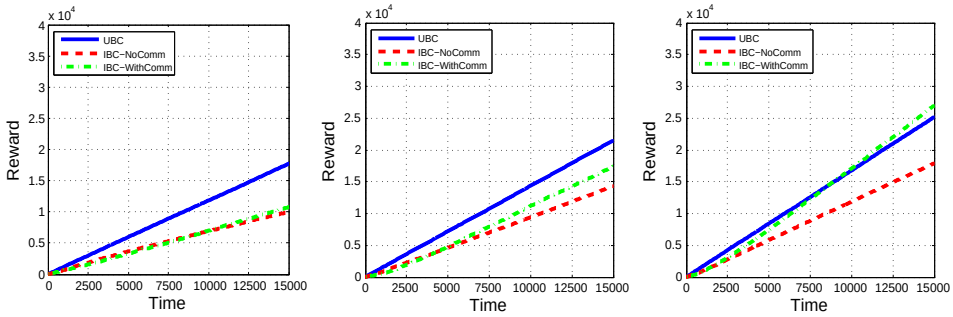
In this experiment, in each sub-region, at every cycle, an event may occur in one of the *segments* of the sub-region with a probability of 0.5. Each event has a weight of 1.

As shown in Figure 5.12, *Uninformed Boundary Coverage* collects more rewards than *Informed Boundary Coverage* in all the visual ranges. Increasing the robots' visual range also leads to all the algorithms collecting more rewards after a 15000 cycle run of the simulation. Furthermore, there is no significant difference observed between the communicating and the non-communicating robot teams in *Informed Boundary Coverage*.



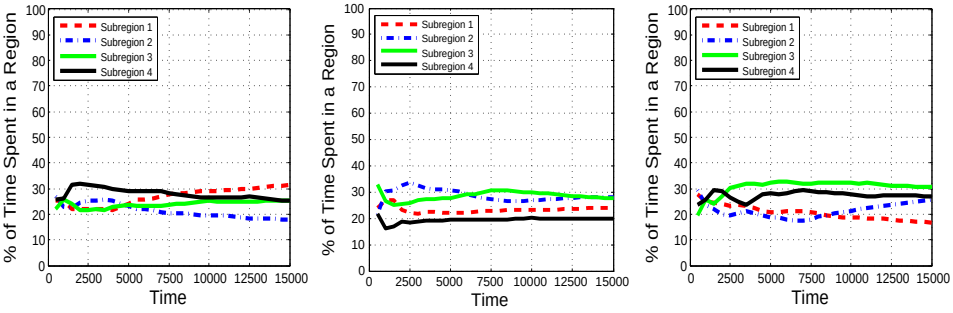
(a) *Visual Range = 0.5m* **(b)** *Visual Range = 1m* **(c)** *Visual Range = 1.5m*

Figure 5.12: *Experiment 1: Uniform Event Occurrence (Low Freq)*



(a) *Visual Range = 0.5m* **(b)** *Visual Range = 1m* **(c)** *Visual Range = 1.5m*

Figure 5.13: *Experiment 1: Uniform Event Occurrence (High Freq)*



(a) *Visual Range = 0.5m* **(b)** *Visual Range = 1m* **(c)** *Visual Range = 1.5m*

Figure 5.14: *Percentage of Time a Team of 10 Robots Using IBC-WithComm Spends in Each Sub-region on Experiment 1*

Figure 5.14 shows the communicating robots using the *Informed Boundary Coverage* algorithm spend an almost equal amount of time in each sub-region under different visual ranges.

High-Frequency Event Model

In this experiment, in each sub-region, at every cycle, five events may occur in different *segments* of the sub-region, each with a probability of 0.5. Each event has a weight of 1.

As shown in Figure 5.13, *Uninformed Boundary Coverage* collects more rewards than *Informed Boundary Coverage* in low and medium visual ranges, however as the visual range increases, the difference between the two algorithms declines, and even in the high visual range of $1.5m$, the communicating version of *Informed Boundary Coverage* outperforms *Uninformed Boundary Coverage*. Moreover, communication among the robots affects the performance of the *Informed Boundary Coverage* algorithm, specially in medium and high visual ranges, as the communicating robot team outperforms the non-communicating robot team in collecting the rewards.

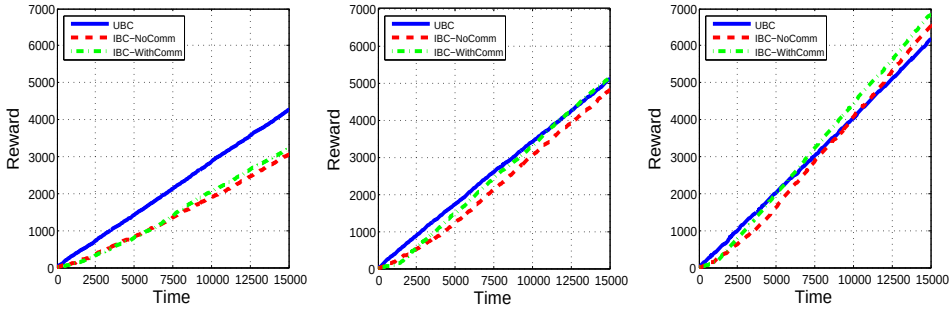
The overall pattern of time the communicating robots using *Informed Boundary Coverage* spend in each sub-region is similar to the low-frequency event model shown in Figure 5.14.

5.5.2 Experiment 2: Non-uniform Event Occurrence

Low-Frequency Event Model

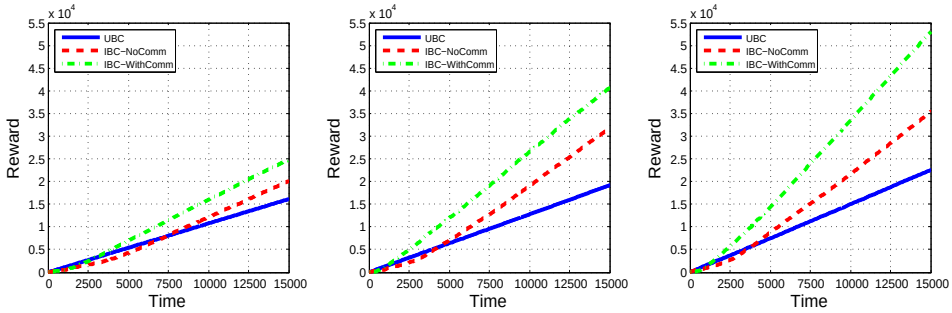
In this experiment, no events occur in sub-region 1. In sub-region 2, at every cycle, an event may occur in a *segment* with a probability of 0.3. In sub-region 3, at every cycle, an event may occur in a *segment* with a probability of 0.6, and in sub-region 4, an event may occur in a *segment* with a probability of 0.9. Each event in sub-regions 2, 3, and 4 is weighted 1.

As shown in Figure 5.15, *Uninformed Boundary Coverage* outperforms *Informed Boundary Coverage* in the low visual range of $0.5m$, but in the medium and high visual ranges of $1m$ and $1.5m$, *Informed Boundary Coverage* shows a



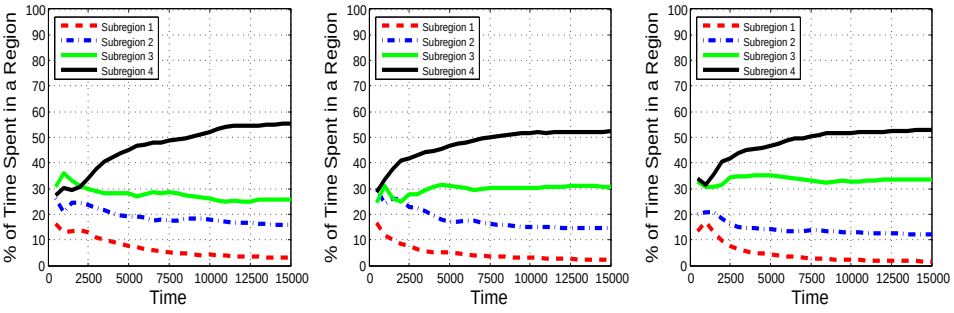
(a) *Visual Range = 0.5m* **(b)** *Visual Range = 1m* **(c)** *Visual Range = 1.5m*

Figure 5.15: *Experiment 2: Non-uniform Event Occurrence (Low Freq)*



(a) *Visual Range = 0.5m* **(b)** *Visual Range = 1m* **(c)** *Visual Range = 1.5m*

Figure 5.16: *Experiment 2: Non-uniform Event Occurrence (High Freq)*



(a) *Visual Range = 0.5m* **(b)** *Visual Range = 1m* **(c)** *Visual Range = 1.5m*

Figure 5.17: *Percentage of Time a Team of 10 Robots Using IBC-WithComm Spends in Each Sub-region on Experiment 2*

slightly better performance than *Uninformed Boundary Coverage*. Increasing the robots' visual range also leads to all the algorithms collecting more rewards after a 15000 cycle run of the simulation. Furthermore, there is no significant difference observed between the communicating and the non-communicating robot teams in *Informed Boundary Coverage*.

Figure 5.17 also shows that the communicating robots using the *Informed Boundary Coverage* algorithm learn to spend more time in sub-region 4 due to the higher number of events occurring in this area compared to the other sub-regions. The second area in which the robots spend more time is sub-region 3, and finally are sub-regions 2 and 1 subsequently.

High-Frequency Event Model

In this experiment, no events occur in sub-region 1. In sub-region 2, at every cycle, five events may occur in different *segments* of the sub-region, each with a probability of 0.3. In sub-region 3, at every cycle, five events may occur, in different *segments* of the sub-region, each with a probability of 0.6, and in sub-region 4, five events may occur in different *segments* of the sub-region, each with a probability of 0.9. Each event in sub-regions 2, 3, and 4 is weighted 1.

As shown in Figure 5.16, as opposed to the low-frequency event model, *Informed Boundary Coverage* strongly outperforms *Uninformed Boundary Coverage* in all the visual ranges. Moreover, communication among the robots affects the performance of the *Informed Boundary Coverage* algorithm, specially in medium and high visual ranges, as the communicating robot team outperforms the non-communicating robot team in collecting the rewards.

The overall pattern of time the communicating robots using *Informed Boundary Coverage* spend in each sub-region is similar to the low-frequency event model shown in Figure 5.17.

5.5.3 Experiment 3: All Events Occur in One Sub-region

Low-Frequency Event Model

In this experiment, in sub-region 1, at every cycle, an event may occur in a *segment* of the sub-region, with a probability of 0.9. No events occur in sub-regions 2, 3 and 4. Each event is weighted 1.

As shown in Figure 5.18, *Informed Boundary Coverage* outperforms *Uninformed Boundary Coverage* in all the visual ranges. Increasing the robots' visual range also leads to all the algorithms collecting more rewards after a 15000 cycle run of the simulation. Moreover, communication among the robots affects the performance of the *Informed Boundary Coverage* algorithm, specially in medium and high visual ranges, as the communicating robot team outperforms the non-communicating robot team in collecting the rewards.

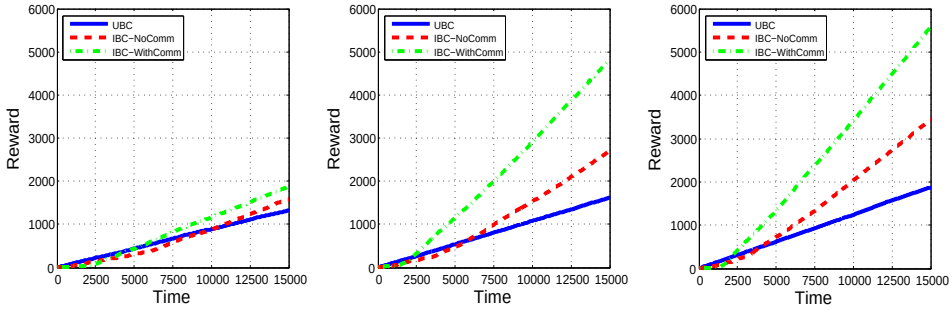
Figure 5.20 also shows that the communicating robots using the *Informed Boundary Coverage* algorithm, learn to spend more time in sub-region 1 because of the higher reward being expected for the robot team compared to the other areas. On the other hand, the robots' presence in sub-regions 2, 3 and 4 declines.

High-Frequency Event Model

In this experiment, in sub-region 1, at every cycle, five events may occur in different *segments* of the sub-region, each with a probability of 0.9. No events occur in sub-regions 2, 3 and 4. Each event is weighted 1.

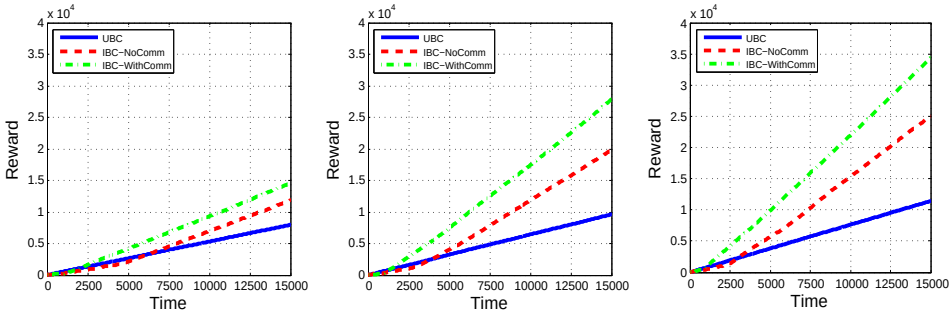
As shown in Figure 5.19, similar to the low-frequency event model, *Informed Boundary Coverage* outperforms *Uninformed Boundary Coverage* in all the visual ranges. Moreover, communication among the robots affects the performance of the *Informed Boundary Coverage* algorithm, specially in medium and high visual ranges, as the communicating robot team outperforms the non-communicating robot team in collecting the rewards.

The overall pattern of time the communicating robots using *Informed Boundary Coverage* spend in each sub-region is similar to the low-frequency event model shown in Figure 5.20.



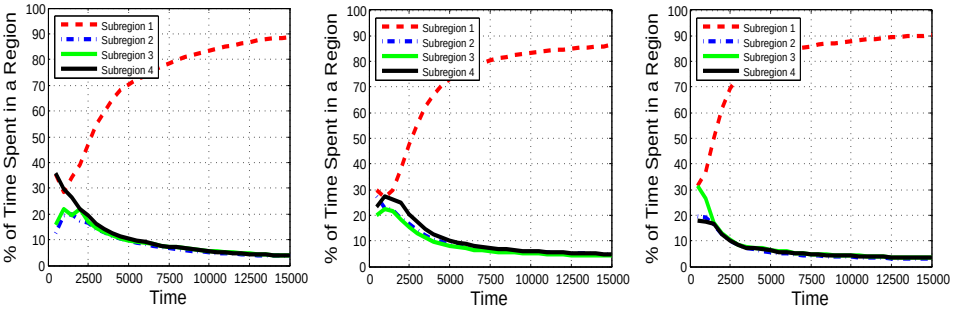
(a) *Visual Range = 0.5m* **(b)** *Visual Range = 1m* **(c)** *Visual Range = 1.5m*

Figure 5.18: *Experiment 3: All Events Occur in One Sub-region (Low Freq)*



(a) *Visual Range = 0.5m* **(b)** *Visual Range = 1m* **(c)** *Visual Range = 1.5m*

Figure 5.19: *Experiment 3: All Events Occur in One Sub-region (High Freq)*



(a) *Visual Range = 0.5m* **(b)** *Visual Range = 1m* **(c)** *Visual Range = 1.5m*

Figure 5.20: *Percentage of Time a Team of 10 Robots Using IBC-WithComm Spends in Each Sub-region on Experiment 3*

5.5.4 Experiment 4: Dynamic Event Occurrence

Low-Frequency Event Model

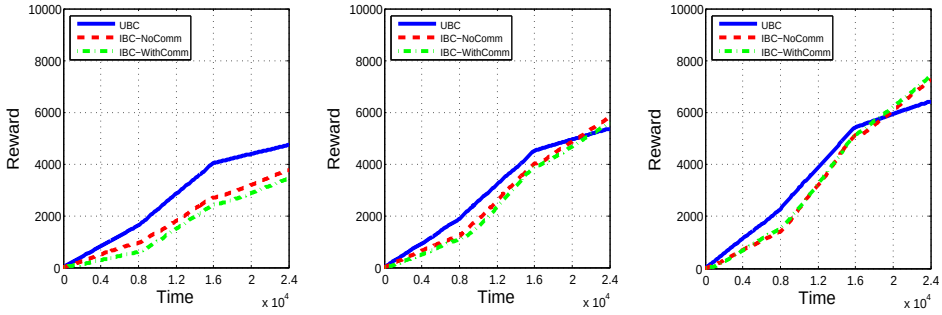
In this experiment, the event distribution changes at some times unknown to the robots during the simulation run. We assume that during the first 8000 cycles, the events occur in the workspace according to the low-frequency event model discussed in Experiment 1 (Uniform Event Occurrence), between cycles $8000 - 16000$, the events occur according to the low-frequency event model discussed in Experiment 2 (Non-uniform Event Occurrence), and between cycles $16000 - 24000$, the events occur according to the low-frequency event model mentioned in Experiment 3 (All Events Occur in One Sub-region).

As shown in Figure 5.21, in the low viusal range of $0.5m$, *Uninformed Boundary Coverage* outperforms *Informed Boundary Coverage*, however as the visual range increases, *Informed Boundary Coverage* shows a better performance than *Uninformed Boundary Coverage*. Increasing the robots' visual range also leads to all the algorithms collecting more rewards after a 24000 cycle run of the simulation. Furthermore, there is no significant difference observed between the communicating and the non-communicating robot teams in *Informed Boundary Coverage*. In this experiment, the robots adapted themselves to the changes in the event distribution on the boundaries, and updated their policies based on these changes.

Figure 5.23 also shows that in the first 8000 cycles of the simulation, the communicating robots using *Informed Boundary Coverage* spend an almost equal amount of time in each sub-region. In the second 8000 cycles, the robots presence in sub-region 4 increases and in the third 8000 cycles of the simulation, the robots spend more time in sub-region 1 and less in sub-region 4.

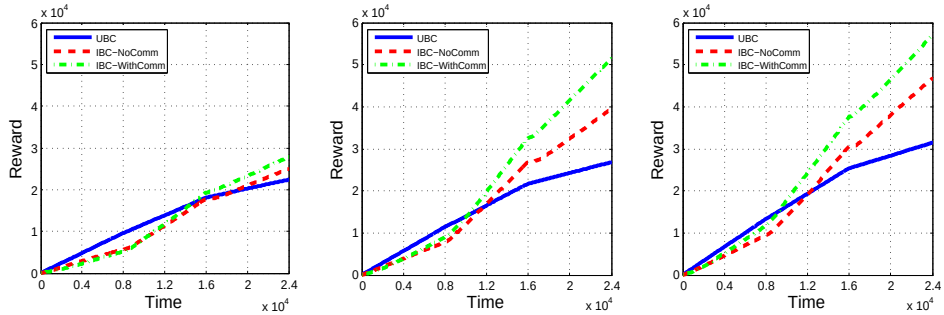
High-Frequency Event Model

Similar to the low-frequency event model, in this experiment, the event distribution changes at some times unknown to the robots during the simulation run. During the first 8000 cycles, the events occur in the workspace according to the high-frequency event model discussed in Experiment 1 (Uniform Event Occurrence), between cycles $8000 - 16000$, the events occur according to the high-frequency



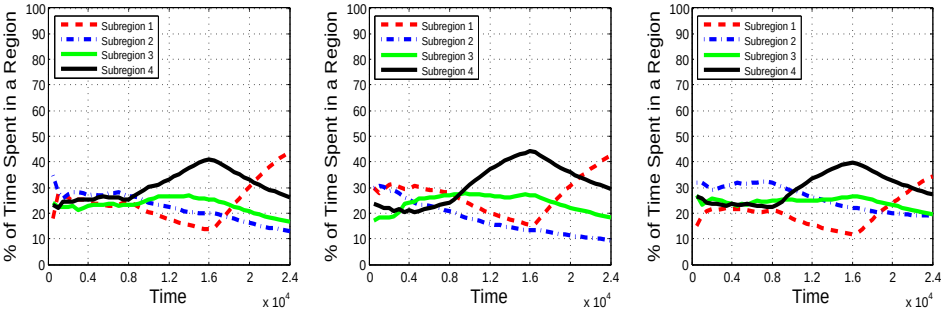
(a) *Visual Range = 0.5m* **(b)** *Visual Range = 1m* **(c)** *Visual Range = 1.5m*

Figure 5.21: *Experiment 4: Dynamic Event Occurrence (Low Freq)*



(a) *Visual Range = 0.5m* **(b)** *Visual Range = 1m* **(c)** *Visual Range = 1.5m*

Figure 5.22: *Experiment 4: Dynamic Event Occurrence (High Freq)*



(a) *Visual Range = 0.5m* **(b)** *Visual Range = 1m* **(c)** *Visual Range = 1.5m*

Figure 5.23: *Percentage of Time a Team of 10 Robots Using IBC-WithComm Spends in Each Sub-region on Experiment 4*

	# of Robots	Experiment 1			Experiment 2			Experiment 3			Experiment 4		
		UBC	IBC-NoComm	IBC-WithComm	UBC	IBC-NoComm	IBC-WithComm	UBC	IBC-NoComm	IBC-WithComm	UBC	IBC-NoComm	IBC-WithComm
Map #1	10	2940	1579	1586	4255	3048	3191	1317	1579	1871	4727	3764	3434
Map #2	10	4064	2331	2434	5999	5638	5855	1721	1920	2053	6326	4041	4401
Map #3	10	4862	2247	2354	7006	4297	4121	2220	2275	2331	7610	4313	4611
Map #4	10	5696	2448	2342	8486	7101	7388	2574	3120	3249	7978	5936	6303

(a) *Visual Range = 0.5m*

	# of Robots	Experiment 1			Experiment 2			Experiment 3			Experiment 4		
		UBC	IBC-NoComm	IBC-WithComm	UBC	IBC-NoComm	IBC-WithComm	UBC	IBC-NoComm	IBC-WithComm	UBC	IBC-NoComm	IBC-WithComm
Map #1	10	3578	2137	2020	5098	4795	5148	1616	2691	4812	5358	5798	5580
Map #2	10	4621	2650	2731	6608	7651	8087	2082	2819	4104	7012	7969	7819
Map #3	10	5042	2738	2879	7576	5523	5396	2365	3816	4602	8038	8322	8475
Map #4	10	6403	3470	3586	9189	8775	8389	2868	3715	4880	8466	9660	9930

(b) *Visual Range = 1m*

	# of Robots	Experiment 1			Experiment 2			Experiment 3			Experiment 4		
		UBC	IBC-NoComm	IBC-WithComm	UBC	IBC-NoComm	IBC-WithComm	UBC	IBC-NoComm	IBC-WithComm	UBC	IBC-NoComm	IBC-WithComm
Map #1	10	4222	2617	2752	6167	6533	6836	1871	3431	5566	6426	7245	7398
Map #2	10	4895	3417	3555	7359	8717	9232	2269	4006	5877	7726	8533	8696
Map #3	10	6128	3791	3961	9066	10528	11314	2782	4444	5026	9383	9591	9952
Map #4	10	6600	3895	4098	9582	10981	11160	2941	4490	5485	10244	11638	12294

(c) *Visual Range = 1.5m*

Table 5.1: *Low-Frequency Event Model - Total Reward Collected by the Team on Different Maps Based on Different Experiments and Various Visual Ranges*

event model discussed in Experiment 2 (Non-uniform Event Occurrence), and between cycles $16000 - 24000$, the events occur according to the high-frequency event model mentioned in Experiment 3 (All Events Occur in One Sub-region).

As shown in Figure 5.22, *Informed Boundary Coverage* outperforms *Uninformed Boundary Coverage* in all the visual ranges. Moreover, communication among the robots affects the performance of the *Informed Boundary Coverage* algorithm, specially in medium and high visual ranges, as the communicating robot team outperforms the non-communicating robot team in collecting the rewards. In this experiment, the robots also adapted themselves to the changes in the event distribution on the boundaries, and updated their policies based on these changes.

	# of Robots	Experiment 1			Experiment 2			Experiment 3			Experiment 4		
		UBC	IBC-NoComm	IBC-WithComm	UBC	IBC-NoComm	IBC-WithComm	UBC	IBC-NoComm	IBC-WithComm	UBC	IBC-NoComm	IBC-WithComm
Map #1	10	17679	9975	10751	16092	20103	24787	7961	11833	14562	22374	24950	27796
Map #2	10	24470	16504	17626	21905	27070	34783	10957	14282	18020	30542	31198	36529
Map #3	10	29285	17307	20003	26426	46179	54730	13197	15885	17959	36828	44537	51467
Map #4	10	34480	18713	19255	31101	46219	56131	15471	15777	17499	43261	42023	45528

(a) *Visual Range = 0.5m*

	# of Robots	Experiment 1			Experiment 2			Experiment 3			Experiment 4		
		UBC	IBC-NoComm	IBC-WithComm	UBC	IBC-NoComm	IBC-WithComm	UBC	IBC-NoComm	IBC-WithComm	UBC	IBC-NoComm	IBC-WithComm
Map #1	10	21459	14320	17366	19128	31646	40602	9610	19776	27769	26802	39476	51348
Map #2	10	27492	18386	22751	24608	39121	52679	12467	19023	29135	34290	43204	53342
Map #3	10	30680	20392	27447	27895	54052	64458	13936	25173	31524	38785	56809	66871
Map #4	10	35835	23329	31802	34526	48146	57834	17299	25208	35116	47864	54254	63302

(b) *Visual Range = 1m*

	# of Robots	Experiment 1			Experiment 2			Experiment 3			Experiment 4		
		UBC	IBC-NoComm	IBC-WithComm	UBC	IBC-NoComm	IBC-WithComm	UBC	IBC-NoComm	IBC-WithComm	UBC	IBC-NoComm	IBC-WithComm
Map #1	10	25142	17867	26953	22513	35345	52936	11392	25047	34296	31405	46641	56882
Map #2	10	29784	21243	30632	26801	43488	53457	13288	28368	35505	37443	54532	65120
Map #3	10	36875	26127	37395	33318	58377	68616	16661	28516	36267	46058	62927	71361
Map #4	10	38030	26444	39629	35750	52315	63190	17874	31222	39130	49810	61256	70069

(c) *Visual Range = 1.5m*

Table 5.2: *High-Frequency Event Model - Total Reward Collected by the Team on Different Maps Based on Different Experiments and Various Visual Ranges*

The overall pattern of time the communicating robots using *Informed Boundary Coverage* spend in each sub-region is similar to the low-frequency event model shown in Figure 5.23.

5.6 Discussion and Conclusions

Considering the low and high-frequency event models of the experiments, Tables 5.1 and 5.2 show the total reward received by the team using *Uninformed Boundary Coverage (UBC)* and the two variants (*i.e.*, communicating and non-communicating robots) of *Informed Boundary Coverage (IBC)* on the 4 maps of Figure 5.3 under visual ranges of $0.5m$, $1m$, and $1.5m$. The results on maps 2, 3

and 4 are consistent with the results discussed for map 1.

Low-frequency event models and low visual ranges result in the better performance of *Uninformed Boundary Coverage* against *Informed Boundary Coverage*, and also result in the absence of a significant difference between the communicating and the non-communicating robots in *Informed Boundary Coverage*.

As the robots' visual range decreases, the number of computed states, *i.e.* guards, in the target area increases. Accordingly, it becomes harder for *Informed Boundary Coverage* to distinguish the more critical states in the target area which can potentially provide more rewards. The low-frequency event model makes the situation worse, because now there are more states not conveying any valuable information regarding the events, and as a result, the communicating robots are unable to transmit helpful information about the states. On the other hand, as the event frequency and the robots' visual range increase, learning the event distribution on the boundaries by *Informed Boundary Coverage* and communication among the robots becomes more crucial.

In experiment 3 (All Events Occur in One Sub-region), as the robots converge to sub-region 1 after a while, the number of states the robots have to deal with declines and is limited to the ones in sub-region 1. Hence, even in the low-frequency event model, *Informed Boundary Coverage* outperforms *Uninformed Boundary Coverage* in all the visual ranges, and the communicating robots outperform the non-communicating robots in collecting the rewards.

Chapter 6

Conclusions and Future Directions

Distributed coverage is a challenging problem in different scenarios for multi-robot systems. The aim is to deploy a team of robots that will spread out and move around a target area to perform sensing, monitoring, data collection, search, or distributed servicing tasks. *Single Area Coverage*, *Repeated Area Coverage*, and *Repeated Boundary Coverage* are three variations of the coverage problem studied in this thesis.

6.1 Thesis Contributions

In summary, the contributions of this thesis are as follows.

6.1.1 On Multi-Robot Single Area Coverage

In the first part of the thesis, we addressed the problem of multi-robot single coverage of a target area. The robots have a limited circular visual range. We demonstrated how a workspace is modeled through locating the static guards, and building a *Constrained Delaunay Triangulation* graph on the guards and the nodes of the workspace. The proposed *Cyclic Coverage* algorithm guarantees completeness, in that every accessible point in the target area is visited in a finite time by at least one of the paths assigned to the robots.

Cyclic Coverage improves upon the coverage algorithms based on *Approximate Cellular Decomposition*, which ignore partially occluded cells or areas close to the boundaries. It is also an improvement on the algorithms based on *Exact Cellular Decomposition*, which do not have a clear policy for moving within the cells, and may have many redundant motions while moving between the cells in the workspace.

Cyclic Coverage supports heterogeneous robots having various maximal speeds, and supports robustness by handling individual robot failure. It also balances the workload distribution among the robots based on their maximal speeds. Finally, it was shown that the obtained results on the *Coverage Time* of the sample environments are scalable to workspaces of different sizes (*i.e.*, different amounts of free space), and robots of varied visual ranges.

6.1.2 On Multi-Robot Repeated Area Coverage

In the second part of the thesis, we tackled the problem of multi-robot repeated coverage of a target area. The robots have a limited circular visual range. The *Reduced-Vis* and the *Reduced-CDT* representations of the workspace were developed based on the *Visibility Graph* and the *Constrained Delaunay Triangulation* built on the workspace. Three distributed *Cluster-based* algorithms were introduced for the problem, namely, the *Uninformed Clustering Coverage*, the *Edge-based Clustering Coverage*, and the *Node-based Clustering Coverage* algorithms, differing as to how they partition the *Reduced Graph* among the robots. In order to build a tour on each partition of the clustered *Reduced Graph*, two tour building algorithms were proposed, namely, the *Double-Minimum Spanning Tree*, and the *Chained Lin-Kernighan* algorithms.

A comprehensive set of performance metrics were defined including: *Total Path Length (TPL)*, *Total Average Visiting Period (TAVP)*, *Total Worst Visiting Period (TWVP)*, and the *Balance in Workload Distribution (BWD)*. *Cyclic Coverage*, used as a benchmark to compare the algorithms, produces *optimal* or *near-optimal* solutions for the *single-robot* case, in the *VG* or the *CDT* graph built on the set of guards computed on the workspace, and under *TPL* and TWVP; however, *Cyclic Coverage* is not always the best solution when extending the problem to multi-

robot scenarios.

Given that it is not possible to develop polynomial approximation algorithms, when optimizing each of the metrics mentioned above, and the fact that some of these metrics mutually conflict in the coverage problem, we conducted an extensive experimental analysis to evaluate the performance of the algorithms. In summary:

- Under all the optimization criteria, the coverage algorithms show similar performance under the two graph representation algorithms, *VG* and *CDT*.
- Under *TPL*, *TWVP*, and *BWD*, all the *Cluster-based* algorithms perform better under *CLK* than *Double-MST*.
- Under *TAVP*, all the *Cluster-based* algorithms perform better under *Double-MST* than *CLK*.
- Under *TPL*, in all the tested visual ranges for the robots, at least one of the *Cluster-based* algorithms (*i.e.*, *Uninformed Clustering Coverage*) outperforms *Cyclic Coverage*, and as the visual range of the robots increases, there are more *Cluster-based* algorithms dominating *Cyclic Coverage*, especially in the scenarios in which more robots are involved.
- Under *TAVP*, *Node-based Clustering Coverage* and *Edge-based Clustering Coverage* are the best options when working with robots having medium and large visual ranges, and *Cyclic Coverage* is the choice of the coverage mission for robots with small visual ranges.
- Under *TWVP*, *Cyclic Coverage* dominates the *Cluster-based* algorithms in all the tested visual ranges for the robots.
- Under *BWD*, for small visual ranges, *Node-based Clustering Coverage* is the best choice for balancing the workload distribution among the robots; however, with the increase of the visual range, *Edge-based Clustering Coverage* dominates the *Node-based Clustering Coverage* algorithm for maximizing the *BWD*.

The results can be used as a framework for choosing an appropriate combination of repeated coverage algorithm, environment representation, and the robots' visual range based on the particular workspace and the metric to be optimized.

6.1.3 On Multi-Robot Repeated Boundary Coverage

In the third part of the thesis, we focused on the problem of multi-robot repeated coverage of the boundaries of a target area and the structures inside it. The robots have a limited circular visual range and line-of-sight communication is assumed among the robots. Events may occur at any position on the boundaries, and the robots are not *a priori* aware of the event distribution. The goal is to maximize the total detection reward of the events. The reward a robot receives for detecting an event depends on how early the event is detected.

To this end, the *Boundary Graph* representation was developed to model the workspace, and the *Informed Boundary Coverage* algorithm was presented, in which each robot autonomously learns the event distribution on the boundaries. Based on the policy being learned, each robot then plans in a decentralized manner to select the best path in the target area to visit the most promising parts of the boundary. The performance of the learning algorithm was compared with a heuristic algorithm for the Traveling Salesman Problem (*i.e.*, *Uninformed Boundary Coverage*), on the basis of the total reward collected by the team during a finite period of time.

Four experiments were designed, in each the event distribution varies in the sub-regions of the selected maps: *1) Uniform Event Occurrence*, *2) Non-uniform Event Occurrence*, *3) All Events Occur in One Sub-region*, and *4) Dynamic Event Occurrence*. For each experiment, two low- and high-frequency event models were assumed. In the high-frequency model, the rate at which the events occur is five times the rate of event occurrence in the low-frequency model. In these experiments, we investigated how robots' visual range, and communication among the robots affect the performance of the robot team in the coverage mission, and how event frequency affects the impact of communication on the robots' performance.

In summary, low-frequency event models and low visual ranges result in the better performance of *Uninformed Boundary Coverage* against *Informed Bound-*

ary Coverage, and also result in the absence of a significant difference between the communicating and the non-communicating robots in *Informed Boundary Coverage*.

As the robots' visual range decreases, the number of computed states, *i.e.* guards, in the target area increases. Accordingly, it becomes harder for *Informed Boundary Coverage* to distinguish the more critical states in the target area which can potentially provide more rewards. The low-frequency event model makes the situation worse, because now there are more states not conveying any valuable information regarding the events, and as a result the communicating robots are unable to transmit helpful information about the states. On the other hand, as the event frequency and the robots' visual range increase, learning the event distribution on the boundaries by *Informed Boundary Coverage* and communication among the robots becomes more crucial.

In experiment 3 (All Events Occur in One Sub-region), as the robots converge to sub-region 1 after a while, the number of states the robots have to deal with declines and is limited to the ones in sub-region 1. Hence, even in the low-frequency event model, *Informed Boundary Coverage* outperforms *Uninformed Boundary Coverage* in all the visual ranges, and the communicating robots outperform the non-communicating robots in collecting the rewards.

As far as we are aware, there is no work using the boundary coverage framework studied in this thesis. In our work, instead of patrolling a single open or closed polyline, the robots patrol the inner boundaries of a full environment and the structures inside it, and it is assumed that different parts of the boundary may have different priorities depending on the probability distribution of the events. Also, our robots can detect multiple events/intruders simultaneously, as opposed to single intruder scenarios studied in previous work.

6.2 Future Directions

There are many challenging possible future research directions for the work reported here:

- **Non-uniform Environments:** The maximal speed allowed for the robots may vary in different parts of the workspace.

- **Non-polygonal Environments:** In this thesis, we addressed the coverage problem in polygonal environments containing polygonal obstacles. Future work could be to extend the work to non-polygonal or free-form workspaces and obstacles. Since our proposed coverage algorithms are based on the guards computed in the target area, therefore as long as we can compute the required guards in free-form workspaces, the coverage algorithms will be valid in those areas as well.

- **Heterogeneity:** Heterogeneity can be defined in various aspects and contexts, such as different movement capabilities, robots having different visual ranges, or different task/event handling abilities.

- **Coalition Formation:** Some tasks/events may require multiple robots to be handled.

- **Learning and Uncertainty:** Noisy robot sensors, action uncertainty, unknown obstacles, and the like, can lead to different challenging learning problems. For noisy sensors, the accuracy of the area information, e.g. events, realized by a robot could vary with the distance of that part of the area from the robot.

- **Dynamic Environments:** The robot team should have the ability to change its policy over time in response to a changing environment with dynamic obstacles, either to improve performance or to prevent unnecessary degradation in performance.

- **Coverage in 3D Spaces:** Extending the work from 2D planar coverage to coverage of non-planar workspaces in a 3D space is a challenging future direction. Distributed coverage of non-planar workspaces is vital in a broad class of real-world applications in rough terrains.

- **Deployment on Real Robots:** Performing the experiments on real robots in real-world scenarios is an obvious extension of this thesis. The physical limitations of the robots in the real world is a critical issue as well.

Bibliography

[1] E. U. Acar and H. Choset. Sensor-based coverage of unknown environments: Incremental construction of morse decompositions. *The International Journal of Robotics Research*, 21(4):345–366, 2002. → page 6

[2] E. U. Acar, H. Choset, A. A. Rizzi, P. N. Atkar, and D. Hull. Morse decompositions for coverage tasks. *The International Journal of Robotics Research*, 21(4):331–344, 2002. → page 6

[3] E. U. Acar, H. Choset, Y. Zhang, and M. Schervish. Path planning for robotic demining: Robust sensor-based coverage of unstructured environments and probabilistic methods. *The International Journal of Robotics Research*, 22(7-8):441–466, 2003. → page 1

[4] N. Agmon, N. Hazon, and G. A. Kaminka. Constructing spanning trees for efficient multi-robot coverage. In *Proceedings of the IEEE International Conference on Robotics and Automation, ICRA 2006*, pages 1698–1703, 2006. → page 8

[5] N. Agmon, S. Kraus, and G. A. Kaminka. Multi-robot perimeter patrol in adversarial settings. In *Proceedings of the IEEE International Conference on Robotics and Automation, ICRA 2008*, pages 2339–2345, 2008. → page 13

[6] N. Agmon, V. Sadov, G. A. Kaminka, and S. Kraus. The impact of adversarial knowledge on adversarial planning in perimeter patrol. In *Proceedings of the International Conference on Autonomous Agents and Multiagent Systems, AAMAS 2008*, pages 55–62, 2008. → page 13

[7] N. Agmon, S. Kraus, and G. A. Kaminka. Uncertainties in adversarial patrol. In *Proceedings of the International Conference on Autonomous Agents and Multiagent Systems, AAMAS 2009*, pages 1267–1268, 2009. → page 13

[8] N. Agmon, S. Kraus, G. A. Kaminka, and V. Sadov. Adversarial uncertainty in multi-robot patrol. In *Proceedings of the International Joint Conference on Artifical Intelligence, IJCAI 2009*, pages 1811–1817, 2009. → page 13

[9] M. Ahmadi and P. Stone. A multi-robot system for continuous area sweeping tasks. In *Proceedings of the IEEE International Conference on Robotics and Automation, ICRA 2006*, pages 1724 –1729, 2006. → page 11

[10] A. Almeida, G. Ramalho, H. Santana, P. A. Tedesco, T. Menezes, V. Corruble, and Y. Chevaleyre. Recent advances on multi-agent patrolling. In *Proceedings of the Brazilian Symposium on Artificial Intelligence, SBIA 2004*, pages 474–483, 2004. → page 10, 10

[11] D. L. Applegate, W. J. Cook, and A. Rohe. Chained Lin-Kernighan for large traveling salesman problems. *INFORMS Journal on Computing*, 15: 82–92, 2003. → pages 28, 28, 31, 55, 63

[12] D. L. Applegate, R. E. Bixby, V. Chvatal, and W. J. Cook. *The Traveling Salesman Problem: A Computational Study*. Princeton University Press, Princeton, NJ, USA, 2007. → pages 28, 55

[13] N. Basilico, N. Gatti, and F. Amigoni. Leader-follower strategies for robotic patrolling in environments with arbitrary topologies. In *Proceedings of the International Conference on Autonomous Agents and Multiagent Systems, AAMAS 2009*, pages 57–64, 2009. → page 12

[14] M. Batalin and G. S. Sukhatme. Efficient exploration without localization. In *Proceedings of the IEEE International Conference on Robotics and Automation, ICRA 2003*, pages 2714–2719, 2003. → page 2

[15] M. Boardman, J. Edmonds, K. Francis, and C. Clark. Multi-robot boundary tracking with phase and workload balancing. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2010*, pages 3321–3326, 2010. → page 13

[16] R. Borie, C. Tovey, and S. Koenig. Algorithms and complexity results for graph-based pursuit evasion. *Autonomous Robots*, 31(4):317–332, 2011. → page 6

[17] O. Bräysy and M. Gendreau. Vehicle routing problem with time windows, part I: Route construction and local search algorithms. *Transportation Science*, 39(1):104–118, 2005. → page 5

[18] O. Bräysy and M. Gendreau. Vehicle routing problem with time windows, part II: Metaheuristics. *Transportation Science*, 39(1):119–139, 2005. → page 5

[19] A. Breitenmoser, M. Schwager, J.-C. Metzger, R. Siegwart, and D. Rus. Voronoi coverage of non-convex environments with a group of networked robots. In *Proceedings of the IEEE International Conference on Robotics and Automation, ICRA 2010*, pages 4982–4989, 2010. → page 6

[20] W. Burgard, M. Moors, D. Fox, R. Simmons, and S. Thrun. Collaborative multi-robot exploration. In *Proceedings of the IEEE International Conference on Robotics and Automation, ICRA 2000*, pages 476–481, 2000. → page 2

[21] Z. Butler, A. Rizzi, and R. Hollis. Contact sensor-based coverage of rectilinear environments. In *Proceedings of the IEEE International Symposium on Intelligent Control/Intelligent Systems and Semiotics, ISIC/ISAS 1999*, pages 266–271, 1999. → page 7

[22] Z. Butler, A. Rizzi, and R. Hollis. Cooperative coverage of rectilinear environments. In *Proceedings of the IEEE International Conference on Robotics and Automation, ICRA 2000*, pages 2722–2727, 2000. → page 7

[23] Y. U. Cao, A. S. Fukunaga, and A. Kahng. Cooperative mobile robotics: Antecedents and directions. *Autonomous Robots*, 4(1):7–27, 1997. → page 1

[24] S. Carlsson, H. Jonsson, and B. J. Nilsson. Finding the shortest watchman route in a simple polygon. *Discrete and Computational Geometry*, 22: 377–402, 1999. → pages 31, 62

[25] D. W. Casbeer, R. W. Beard, T. W. McLain, S.-M. Li, and R. Mehra. Forest fire monitoring with multiple small UAVs. In *Proceedings of the American Control Conference, ACC 2005*, pages 3530–3535, 2005. → page 2

[26] D. W. Casbeer, D. B. Kingston, R. W. Beard, and T. W. Mclain. Cooperative forest fire surveillance using a team of small unmanned air vehicles. *International Journal of Systems Science*, 37:351–360, 2006. → page 2

[27] B. Chandra, H. Karloff, and C. Tovey. New results on the old k-opt algorithm for the traveling salesman problem. *SIAM Journal on Computing*, 28:1998–2029, 1999. → page 28

[28] B. Chazelle and J. Incerpi. Triangulation and shape-complexity. *ACM Transactions on Graphics*, 3(2):135–152, 1984. → page 17

[29] L. P. Chew. Constrained delaunay triangulations. In *Proceedings of the Symposium on Computational Geometry, SCG 1987*, pages 215–222, 1987. → pages 24, 44

[30] W. Chin and S. Ntafos. Optimum watchman routes. In *Proceedings of the Symposium on Computational Geometry, SCG 1986*, pages 24–33, 1986. → page 5

[31] H. Choset. Coverage of known spaces: The boustrophedon cellular decomposition. *Autonomous Robots*, 9(3):247–253, 2000. → page 6

[32] H. Choset. Coverage for robotics – a survey of recent results. *Annals of Mathematics and Artificial Intelligence*, 31(1-4):113–126, 2001. → page 6, 6

[33] J. Cortes, S. Martinez, T. Karatas, and F. Bullo. Coverage control for mobile sensing networks. *IEEE Transactions on Robotics and Automation*, 20(2):243–255, 2004. → page 6

[34] B. Cărbunar, A. Grama, J. Vitek, and O. Cărbunar. Redundancy and coverage detection in sensor networks. *ACM Transactions on Sensor Networks*, 2(1):94–128, 2006. → page 5

[35] J. Czyzowicz, L. Gasieniec, A. Kosowski, and E. Kranakis. Boundary patrolling by mobile agents with distinct maximal speeds. In *Proceedings of the European Symposium on Algorithms*, ESA 2011, pages 701–712, 2011. → page 14

[36] M. De Berg, O. Cheong, M. Van Kreveld, and M. Overmars. *Computational Geometry: Algorithms and Applications*. Springer-Verlag, New York Inc., 2008. → page 44

[37] M. Desrochers, J. Desrosiers, and M. Solomon. A new optimization algorithm for the vehicle routing problem with time windows. *Operation Research*, 40(2):342–354, 1992. → page 5

[38] G. Dudek, M. R. M. Jenkin, E. Milios, and D. Wilkes. A taxonomy for multi-agent robotics. *Autonomous Robots*, 3:375–397, 1996. → page 1

[39] K. Easton and J. Burdick. A coverage algorithm for multi-robot boundary inspection. In *Proceedings of the IEEE International Conference on Robotics and Automation, ICRA 2005*, pages 727–734, 2005. → page 12

[40] H. A. Eiselt, M. Gendreau, and G. Laporte. Arc routing problems, part II: The rural postman problem. *Operations Research*, 43(3):399–414, 1995. → page 12

[41] Y. Elmaliach, A. Shiloni, and G. A. Kaminka. A realistic model of frequency-based multi-robot polyline patrolling. In *Proceedings of the International Conference on Autonomous Agents and Multiagent Systems, AAMAS 2008*, pages 63–70, 2008. → page 13

[42] Y. Elmaliach, N. Agmon, and G. A. Kaminka. Multi-robot area patrol under frequency constraints. *Annals of Mathematics and Artificial Intelligence*, 57(3-4):293–320, 2009. → page 11

[43] J. Faigl. Approximate solution of the multiple watchman routes problem with restricted visibility range. *IEEE Transactions on Neural Networks*, 21 (10):1668–1679, 2010. → page 5

[44] J. Faigl, M. Kulich, and L. Přeučil. A sensor placement algorithm for a mobile robot inspection planning. *Journal of Intelligent and Robotic Systems*, 62:329–353, 2011. → page 17

[45] A. Farinelli, L. Iocchi, and D. Nardi. Multirobot systems: a classification focused on coordination. *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, 34(5):2015–2028, 2004. → page 1

[46] P. Fazli. On multi-robot area coverage. In *Proceedings of the Twenty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2010*, 2010. → page vi

[47] P. Fazli. On multi-robot area coverage. In *Proceedings of the Canadian Conference on Artificial Intelligence, CanadianAI 2010*, pages 384–387, 2010. → page vi

[48] P. Fazli. On multi-robot area coverage. In *Proceedings of the International Conference on Autonomous Agents and Multiagent Systems, AAMAS 2010*, pages 1669–1670, 2010. → page vi

[49] P. Fazli and A. K. Mackworth. Multi-robot repeated boundary coverage under uncertainty. In *Proceedings of the Workshop on Robotics for Environment Modeling, IROS 2011*, 2011. → page vi

[50] P. Fazli and A. K. Mackworth. Multi-robot repeated boundary coverage under uncertainty. In *Proceedings of the IEEE International Conference on*

Robotics and Biomimetics, ROBIO 2012, pages 2167–2174, 2012. → pages v, 77

[51] P. Fazli and A. K. Mackworth. The effects of communication and visual range on multi-robot repeated boundary coverage. In *Proceedings of the IEEE International Symposium on Safety, Security, and Rescue Robotics, SSRR 2012*, pages 1–8, 2012. → pages v, 77

[52] P. Fazli, A. Davoodi, P. Pasquier, and A. K. Mackworth. On multi-robot area coverage. In *Proceedings of the Seventh Japan Conference on Computational Geometry and Graphs, JCCGG 2009*, pages 75–76, 2009. → page v

[53] P. Fazli, A. Davoodi, P. Pasquier, and A. K. Mackworth. Multi-robot area coverage with limited visibility. In *Proceedings of the International Conference on Autonomous Agents and Multiagent Systems, AAMAS 2010*, pages 1501–1502, 2010. → page iv

[54] P. Fazli, A. Davoodi, P. Pasquier, and A. K. Mackworth. Complete and robust cooperative robot area coverage with limited range. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2010*, pages 5577–5582, 2010. → page iv

[55] P. Fazli, A. Davoodi, P. Pasquier, and A. K. Mackworth. Fault-tolerant multi-robot area coverage with limited visibility. In *Proceedings of the Workshop on Search and Pursuit/Evasion in the Physical World: Efficiency, Scalability, and Guarantees, ICRA 2010*, 2010. → page iv

[56] P. Fazli, A. Davoodi, and A. K. Mackworth. Multi-robot repeated area coverage: Performance optimization under various visual ranges. In *Proceedings of the Ninth Conference on Computer and Robot Vision, CRV 2012*, pages 298 –305, 2012. → page v

[57] P. Fazli, A. Davoodi, and A. K. Mackworth. Multi-robot repeated area coverage. *Autonomous Robots*, 34(4):251–276, 2013. → page v

[58] A. Fournier and D. Y. Montuno. Triangulating simple polygons and equivalent problems. *ACM Transactions on Graphics*, 3(2):153–174, 1984. → page 17

[59] M. Fujita. AIBO: Toward the era of digital creatures. *The International Journal of Robotics Research*, 20(10):781–794, 2001. → page 1

[60] Y. Gabriely and E. Rimon. Spanning-tree based coverage of continuous areas by a mobile robot. *Annals of Mathematics and Artificial Intelligence*, 31(1-4):77–98, 2001. → page 8, 8

[61] A. Gasparri, B. Krishnamachari, and G. S. Sukhatme. A framework for multi-robot node coverage in sensor networks. *Annals of Mathematics and Artificial Intelligence*, 52(2):281–305, 2008. → page 5

[62] B. P. Gerkey and M. J. Matarić. A formal analysis and taxonomy of task allocation in multi-robot systems. *The International Journal of Robotics Research*, 23(9):939–954, 2004. → page 1

[63] B. P. Gerkey, S. Thrun, and G. Gordon. Visibility-based pursuit-evasion with limited field of view. *The International Journal of Robotics Research*, 25(4):299–315, 2006. → pages 1, 6

[64] A. Girard, A. Howell, and J. Hedrick. Border patrol and surveillance missions using multiple unmanned air vehicles. In *Proceedings of the IEEE Conference on Decision and Control, CDC 2004*, pages 620 –625, 2004. → page 14

[65] B. L. Golden, S. Raghavan, and E. A. Wasil, editors. *The Vehicle Routing Problem: Latest Advances and New Challenges*. Springer, New York, USA, 2008. → page 5

[66] Y. Guo, L. E. Parker, and R. Madhavan. Collaborative robots for infrastructure security applications. In N. Nedjah, L. Coelho, and L. Mourelle, editors, *Mobile Robots: The Evolutionary Approach*, pages 185–200. Springer, Berlin, Heidelberg, 2007. → page 11

[67] J. A. Hartigan and M. A. Wong. Algorithm as 136: A k-means clustering algorithm. *Journal of the Royal Statistical Society: Series C (Applied Statistics)*, 28(1):100–108, 1979. → page 52

[68] N. Hazon and G. A. Kaminka. Redundancy, efficiency and robustness in multi-robot coverage. In *Proceedings of the IEEE International Conference on Robotics and Automation, ICRA 2005*, pages 735–741, 2005. → page 8

[69] K. Helsgaun. General k-opt submoves for the Lin-Kernighan TSP heuristic. *Mathematical Programming Computation*, 1(2):119–163, 2009. → pages 31, 63

[70] C. Hofner and G. Schmidt. Path planning and guidance techniques for an autonomous mobile cleaning robot. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 1994*, pages 610–617, 1994. → page 2

[71] H. Hoos and T. Sttzle. *Stochastic Local Search: Foundations and Applications*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2004. → page 28

[72] A. Howard. Multi-robot simultaneous localization and mapping using particle filters. *The International Journal of Robotics Research*, 25(12): 1243–1256, 2006. → page 2

[73] C.-F. Huang and Y.-C. Tseng. The coverage problem in a wireless sensor network. In *Proceedings of the ACM International Conference on Wireless Sensor Networks and Applications, WSNA 2003*, pages 115–121, 2003. → page 5

[74] W. Huang. Optimal line-sweep-based decompositions for coverage algorithms. In *Proceedings of the IEEE International Conference on Robotics and Automation, ICRA 2001*, pages 27–32, 2001. → page 7

[75] J. Jennings, G. Whelan, and W. Evans. Cooperative search and rescue with a team of mobile robots. In *Proceedings of the International Conference on Advanced Robotics, ICAR 1997*, pages 193–200, 1997. → page 1

[76] E. Jensen, M. Franklin, S. Lahr, and M. Gini. Sustainable multi-robot patrol of an open polyline. In *Proceedings of the IEEE International Conference on Robotics and Automation, ICRA 2011*, pages 4792 –4797, 2011. → page 13

[77] M. P. Johnson, F. Fang, and M. Tambe. Patrol strategies to maximize pristine forest area. In *Proceedings of the Twenty-Sixth AAAI Conference on Artificial Intelligence, AAAI 2012*, 2012. → page 2

[78] G. D. Kazazakis and A. A. Argyros. Fast positioning of limited visibility guards for inspection of 2D workspaces. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2002*, pages 2843–2848, 2002. → page 17

[79] S. Koenig and Y. Liu. Terrain coverage with ant robots: a simulation study. In *Proceedings of the International Conference on Autonomous Agents, AGENTS 2001*, pages 600–607, 2001. → page 9

[80] R. E. Korf. Real-time heuristic search. *Artificial Intelligence*, 42(2-3): 189–211, 1990. → page 9

[81] D. Korzhyk, Z. Yin, C. Kiekintveld, V. Conitzer, and M. Tambe. Stackelberg vs. nash in security games: an extended investigation of interchangeability, equivalence, and uniqueness. *Journal of Artificial Intelligence Research*, 41(2):297–327, 2011. → page 12

[82] J.-C. Latombe. *Robot Motion Planning*. Kluwer Academic Publishers, Norwell, MA, USA, 1991. → pages 6, 44

[83] S. M. LaValle and J. Hinrichsen. Visibility-based pursuit-evasion: the case of curved environments. *IEEE Transactions on Robotics and Automation*, 17(2):196–202, 2001. → page 1

[84] S. M. Lavalle, D. Lin, L. J. Guibas, J.-C. Latombe, and R. Motwani. Finding an unpredictable target in a workspace with obstacles. In *Proceedings of the IEEE International Conference on Robotics and Automation, ICRA 1997*, pages 737–742, 1997. → page 2

[85] D. T. Lee and B. J. Schachter. Two algorithms for constructing a delaunay triangulation. *International Journal of Computer Information Science*, 9 (3):219–242, 1980. → pages 21, 44

[86] S. Lin and B. Kernighan. An effective heuristic algorithm for the Traveling-Salesman Problem. *Operations Research*, 21(2):498–516, 1973. → pages 28, 28, 31, 55, 63

[87] C. Luo and S. Yang. A real-time cooperative sweeping strategy for multiple cleaning robots. In *Proceedings of the IEEE International Symposium on Intelligent Control, ISIC 2002*, pages 660–665, 2002. → pages 8, 9

[88] C. Luo, S. Yang, and D. Stacey. Real-time path planning with deadlock avoidance of multiple cleaning robots. In *Proceedings of the IEEE International Conference on Robotics and Automation, ICRA 2003*, pages 4080–4085, 2003. → pages 8, 9

[89] A. Machado, G. Ramalho, J.-D. Zucker, and A. Drogoul. Multi-agent patrolling: an empirical analysis of alternative architectures. In *Proceedings of the 3rd International Workshop on Multi-agent-based Simulation II, MABS 2002*, pages 155–170, 2002. → page 10, 10

[90] R. Mannadiar and I. M. Rekleitis. Optimal coverage of a known arbitrary environment. In *Proceedings of the IEEE International Conference on Robotics and Automation, ICRA 2010*, pages 5525–5530, 2010. → page 8

[91] A. Marino, L. E. Parker, G. Antonelli, and F. Caccavale. Behavioral control for multi-robot perimeter patrol: a finite state automata approach. In *Proceedings of the IEEE International Conference on Robotics and Automation, ICRA 2009*, pages 3350–3355, 2009. → page 13

[92] O. Martin, S. Otto, and E. Felten. Large-step markov chains for the traveling salesman problem. *Complex Systems*, 5(3):299–326, 1991. → page 28

[93] O. Martin, S. W. Otto, and E. W. Felten. Large-step markov chains for the TSP incorporating local search heuristics. *Operations Research Letters*, 11: 219–224, 1992. → page 28

[94] S. Martinez, J. Cortes, and F. Bullo. Motion coordination with distributed information. *IEEE Control Systems Magazine*, 27(4):75–88, 2007. → page 6

[95] M. J. Mataric and G. S. Sukhatme. Task-allocation and coordination of multiple robots for planetary exploration. In *Proceedings of the International Conference on Advanced Robotics, ICAR 2001*, pages 61–70, 2001. → page 1

[96] L. Merino, F. Caballero, J. Martnez-de Dios, J. Ferruz, and A. Ollero. A cooperative perception system for multiple UAVs: Application to automatic detection of forest fires. *Journal of Field Robotics*, 23(3-4):165–184, 2006. → page 2

[97] B. J. Nilsson. *Guarding Art Galleries - Methods for Mobile Guards*. Ph.D. Dissertation, Lund University, 1995. → pages 31, 62

[98] A. Okabe, B. Boots, K. Sugihara, and S. N. Chiu. *Spatial Tessellations: Concepts and Applications of Voronoi Diagrams*. John Wiley and Sons, Inc., 2nd edition, 2000. → page 21

[99] J. O'Rourke. *Art Gallery Theorems and Algorithms*. Oxford University Press, New York, NY, USA, 1987. → page 5

[100] E. Packer. *Robust Geometric Computing and Optimal Visibility Coverage*. Ph.D. Dissertation, Stony Brook University, 2008. → pages 31, 39, 62

[101] E. Packer. Computing multiple watchman routes. In *Proceedings of the Seventh International Workshop on Experimental Algorithms, WEA 2008*, pages 114–128, 2008. → page 5

[102] J. Palacin, J. A. Salse, I. Valganon, and X. Clua. Building a mobile robot for a floor-cleaning operation in domestic environments. In *Proceedings of the IEEE Instrumentation and Measurement Technology Conference, IMTC 2003*, pages 1391–1396, 2003. → page 2

[103] J. Palacin, T. Palleja, I. Valganon, R. Pernia, and J. Roca. Measuring coverage performances of a floor cleaning mobile robot using a vision system. In *Proceedings of the IEEE International Conference on Robotics and Automation, ICRA 2005*, pages 4236–4241, 2005. → page 2

[104] T. Palleja, M. Tresanchez, M. Teixido, and J. Palacin. Modeling floor-cleaning coverage performances of some domestic mobile robots in a reduced scenario. *Robotics and Autonomous Systems*, 58(1):37–45, 2010. → page 2

[105] L. E. Parker. L-ALLIANCE: Task-oriented multi-robot learning in behavior-based systems. *Advanced Robotics*, 11(4):305–322, 1996. → page 1

[106] L. E. Parker. ALLIANCE: an architecture for fault tolerant multi-robot cooperation. *IEEE Transactions on Robotics and Automation*, 14(2): 220–240, 1998. → page 1, 1

[107] P. Paruchuri, M. Tambe, F. Ordóñez, and S. Kraus. Security in multiagent systems by policy randomization. In *Proceedings of the International Conference on Autonomous Agents and Multiagent Systems, AAMAS 2006*, pages 273–280, 2006. → page 12

[108] P. Paruchuri, J. P. Pearce, M. Tambe, F. Ordonez, and S. Kraus. An efficient heuristic approach for security against multiple adversaries. In *Proceedings of the International Conference on Autonomous Agents and Multiagent Systems, AAMAS 2007*, pages 311–318, 2007. → page 12

[109] L. C. A. Pimenta, V. Kumar, R. Mesquita, and G. A. S. Pereira. Sensing and coverage for a network of heterogeneous robots. In *Proceedings of the IEEE Conference on Decision and Control, CDC 2008*, pages 3947–3952, 2008. → page 6

[110] J. Pita, M. Jain, J. Marecki, F. Ordóñez, C. Portway, M. Tambe, C. Western, P. Paruchuri, and S. Kraus. Deployed ARMOR protection: the application of a game theoretic model for security at the los angeles international airport. In *Proceedings of the International Joint Conference on Autonomous Agents and Multiagent Systems: Industrial Track, AAMAS 2008*, pages 125–132, 2008. → page 12

[111] J. Pita, M. Tambe, C. Kiekintveld, S. Cullen, and E. Steigerwald. GUARDS: game theoretic security allocation on a national scale. In *Proceedings of the International Conference on Autonomous Agents and Multiagent Systems*, AAMAS 2011, pages 37–44, 2011. → page 12

[112] S. Poduri and G. S. Sukhatme. Constrained coverage for mobile sensor networks. In *Proceedings of the IEEE International Conference on Robotics and Automation, ICRA 2004*, pages 165–171, 2004. → page 5

[113] T. Ralphs, L. Kopman, W. Pulleyblank, and L. Trotter. On the capacitated vehicle routing problem. *Mathematical Programming*, 94:343–359, 2003. → page 5

[114] I. M. Rekleitis, V. Lee-Shue, A. Peng New, and H. Choset. Limited communication, multi-robot team based coverage. In *Proceedings of the IEEE International Conference on Robotics and Automation, ICRA 2004*, pages 3462–3468, 2004. → page 8

[115] I. M. Rekleitis, A. P. New, E. S. Rankin, and H. Choset. Efficient boustrophedon multi-robot coverage: an algorithmic approach. *Annals of Mathematics and Artificial Intelligence*, 52(2-4):109–142, 2008. → page 8

[116] H. Sagan. *Space-Filling Curves*. Springer-Verlag, New York, 1994. → page 7

[117] T. Sak, J. Wainer, and S. K. Goldenstein. Probabilistic multiagent patrolling. In *Proceedings of the Brazilian Symposium on Artificial Intelligence: Advances in Artificial Intelligence*, SBIA 2008, pages 124–133, 2008. → page 11

[118] H. Santana, G. Ramalho, V. Corruble, and B. Ratitch. Multi-agent patrolling with reinforcement learning. In *Proceedings of the International Conference on Autonomous Agents and Multiagent Systems, AAMAS 2004*, pages 1122–1129, 2004. → page 10, 10

[119] M. Schwager, J.-J. E. Slotine, and D. Rus. Decentralized, adaptive control for coverage with networked robots. In *Proceedings of the IEEE International Conference on Robotics and Automation, ICRA 2007*, pages 3289–3294, 2007. → page 5

[120] M. Schwager, J.-J. Slotine, and D. Rus. Consensus learning for distributed coverage control. In *Proceedings of the IEEE International Conference on Robotics and Automation, ICRA 2008*, pages 1042–1048, 2008. → page 5

[121] R. Seidel. A simple and fast incremental randomized algorithm for computing trapezoidal decompositions and for triangulating polygons. *Computational Geometry: Theory and Applications*, 1(1):51–64, 1991. → page 17

[122] R. Simmons, D. Apfelbaum, W. Burgard, D. Fox, M. Moors, S. Thrun, and H. Younes. Coordination for multi-robot exploration and mapping. In *Proceedings of the Seventeenth AAAI Conference on Artificial Intelligence, AAAI 2000*, pages 852–858, 2000. → page 2

[123] R. N. Smith, M. Schwager, S. L. Smith, B. H. Jones, D. Rus, and G. S. Sukhatme. Persistent ocean monitoring with underwater gliders: Adapting sampling resolution. *Journal of Field Robotics*, 28(5):714–741, 2011. → page 2

[124] S. V. Spires and S. Y. Goldsmith. Exhaustive geographic search with mobile robots along space-filling curves. In *Proceedings of the International Workshop on Collective Robotics, CRW 1998*, pages 1–12, 1998. → page 7

[125] E. D. Taillard, P. Badeau, M. Gendreau, F. Guertin, and J.-Y. Potvin. A tabu search heuristic for the vehicle routing problem with soft time windows. *Transportation Science*, 31(2):170–186, 1997. → page 5

[126] X. Tan. Fast computation of shortest watchman routes in simple polygons. *Information Processing Letter*, 77(1):27–33, 2001. → pages 31, 62

[127] A. Tomás and A. Bajuelos. Quadratic-time linear-space algorithms for generating orthogonal polygons with a given number of vertices. In *Proceedings of the International Conference on Computational Science and Its Applications, ICCSA 2004*, pages 117–126, 2004. → page 31

[128] P. Toth and D. Vigo. *The Vehicle Routing Problem*. Society for Industrial Mathematics, Philadelphia, PA, 2001. → page 5

[129] P. Toth and D. Vigo. Models, relaxations and exact approaches for the capacitated vehicle routing problem. *Discrete Applied Mathematics*, 123 (1-3):487–512, 2002. → page 5

[130] J. Urrutia. Art gallery and illumination problems. In *Handbook of Computational Geometry*, pages 973–1027. North-Holland, 2000. → page 5

[131] R. Vidal, O. Shakernia, H. Kim, D. Shim, and S. Sastry. Probabilistic pursuit-evasion games: theory, implementation, and experimental evaluation. *IEEE Transactions on Robotics and Automation*, 18(5): 662–669, 2002. → page 6

[132] I. Wagner, M. Lindenbaum, and A. Bruckstein. Distributed covering by ant-robots using evaporating traces. *IEEE Transactions on Robotics and Automation*, 15(5):918–933, 1999. → page 9

[133] K. Williams and J. Burdick. Multi-robot boundary coverage with plan revision. In *Proceedings of the IEEE International Conference on Robotics and Automation, ICRA 2006*, pages 1716–1723, 2006. → page 12

[134] B. Yamauchi. A frontier-based approach for autonomous exploration. In *Proceedings of the IEEE International Symposium on Computational Intelligence in Robotics and Automation, CIRA 1997*, pages 146–151, 1997. → page 2

[135] B. Yamauchi. Frontier-based exploration using multiple robots. In *Proceedings of the International Conference on Autonomous Agents, AGENTS 1998*, pages 47–53, 1998. → page 2

[136] B. Zalik and G. J. Clapworthy. A universal trapezoidation algorithm for planar polygons. *Computers & Graphics*, 23(3):353–363, 1999. → page 17, 17

[137] A. Zelinsky, R. Jarvis, J. C. Byrne, and S. Yuta. Planning paths of complete coverage of an unstructured environment by a mobile robot. In *Proceedings of the International Conference on Advanced Robotics, ICAR 1993*, pages 533–538, 1993. → page 7

[138] X. Zheng, S. Jain, S. Koenig, and D. Kempe. Multi-robot forest coverage. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS 2005*, pages 3852–3857, 2005. → page 8

[139] Z. Zhou, S. Das, and H. Gupta. Connected k-coverage problem in sensor networks. In *Proceedings of the International Conference on Computer Communications and Networks, ICCCN 2004*, pages 373–378, 2004. → page 5