# Multi-Robot Path Planning for Persistent Coverage Tasks with Performance Guarantees

Maria Charitidou

Master of Science Thesis

TU Delft Delft University of Technology

Delft Center for Systems and Control

# Multi-Robot Path Planning for Persistent Coverage Tasks with Performance Guarantees

Master of Science Thesis

For the degree of Master of Science in Systems and Control at Delft University of Technology

Maria Charitidou

June 11, 2019

Faculty of Mechanical, Maritime and Materials Engineering (3mE) · Delft University of Technology


# Abstract

Recent advances on the design of autonomous mobile agents have motivated the use of the latter in persistent surveillance tasks with applications in agriculture, information gathering and search and rescue missions. In these tasks the goal is to design agents' paths so as every point in the area of interest is covered more than once.

In literature the persistent surveillance problem has been addressed under two different frameworks: 1) the visitation frequency of a finite set of points (discrete spaces) and 2) the amount of coverage offered by the agents over the area of interest. In the first category agents aim at visiting a finite set of points in the area so as the time elapsed since a point was last visited is minimized over the set. Here, the environment is considered static and no information on the coverage condition of the points is available. In the second category the environment is characterized by a property that changes over time, called *coverage level* and the goal is to design agents' paths so as the coverage level is maintained at a desired level over the area. In this approach the error between the actual and desired coverage level of the area is bounded. However, locally no information is available on the frequency at which each point gets covered.

The novelty of this thesis lies in the integration of the aforementioned solution approaches so as complete coverage information is provided for a finite set of points. Here, the goal is to design the agents' paths so as the coverage level of each point is bounded from below by a desired level. The paths are planned over a finite optimization horizon and found as a solution to a Mixed Integer Linear Program (MILP). Two formulations of the problem are proposed and their efficiency is validated in simulation. The recursive feasibility of the centralized problem is also guaranteed when a set of time-varying terminal constraints is introduced to the problem. These constraints force agents to move along a pre-defined set of closed paths designed in a way that the coverage level boundness constraint is always satisfied. A two-step method is proposed for their design that relies on the solution of a Linear Program (LP) when a set of closed paths violating the objective is given, found as a solution to a modified version of the proposed MILP at step 1. Finally, two distributed formulations of the problem are introduced in which agents solve in parallel a local path planning problem. The difference on these methods lies in the accuracy of the information available to the agents for planning. Despite this difference simulation tests show a difference of only 1.5% in the coverage level performance when various sized teams of agents are employed for the task.

# Contents

# List of Figures

# List of Tables

# Acknowledgements

First, I would like to express my deepest gratitude to my supervisor Dr.ir. T. Keviczky for giving me the chance to make my dream on conducting research on the field of multi-robot control come true. Without his invaluable feedback and guidance this thesis would not have been the same.

I would also like to thank my mother for the long hours she spent on patiently listening to me and my father for always supporting and believing in me. Last but not least, I would like to thank my sister, Fani for always being on my side and encouraging me to move forward.

Delft, University of Technology
June 11, 2019

Maria Charitidou

‘‘**Το μυαλό δεν είναι ένα δοχείο που πρέπει να γεμίσει αλλά μια φωτιά που πρέπει ν΄ ανάψει.**’’

”The mind is not a vessel to be filled but a fire to be kindled.”

— *Πλούταρχος* (Plutarch)

# Chapter 1

# Introduction

Over the last decades technological evolution and hardware enhancement has led to the design of robotic agents able to perform a wide range of tasks spanning from component assembly to package delivery. These tasks have been assigned to robots as they are found to be either time-consuming or cost-prohibitive or in some cases dangerous for humans. To meet the task objectives, agents are often equipped with different kinds of sensors and actuators that provide them the ability to perceive, exploit and learn their environment either from a static position or while moving around the area. In fact, one of the most interesting and promising capabilities of the agents is mobility due to the plethora of possibilities it offers for a variety of tasks in domestic and outdoor environments. The applications of mobile agents are numerous. Among others we distinct surveillance [2], cleaning [3], forest fire monitoring [4], crop inspection [5] and information gathering e.g in inland waterways [6] or in underwater missions [7].

In the aforementioned applications agents need to cover every point in a known environment multiple times to ensure the currency of their observations (e.g in surveillance, crop inspection, etc) or the collection of a rich data set in information gathering tasks or a high coverage performance (e.g in cleaning). This problem, known in literature as the *persistent coverage control problem* is the main focus of this thesis.

So far, the persistent coverage tasks were performed either manually by humans or depending on the application using satellite imaging, commercial vehicles and Wireless Sensor Networks (WSN). Satellite imaging has been extensively used in agriculture for water stress-detection [8] as also in environmental monitoring tasks for oil-spill and fire detection [9], [10]. However, as mentioned in [5] this method suffers from "the lack of of imagery with optimum spatial and spectral resolutions and an unfavorable revisit time" for most applications. Addressing these problems other methods were proposed using commercial vehicles such as aircrafts or ships that allow humans to manually take pictures or measurements from the area of interest. Nonetheless, the operational costs of these methods are often too high.

Another popular approach presented in literature for collecting information is the use of WSNs. A WSN is defined as a collection of sensor nodes that are able to periodically collect

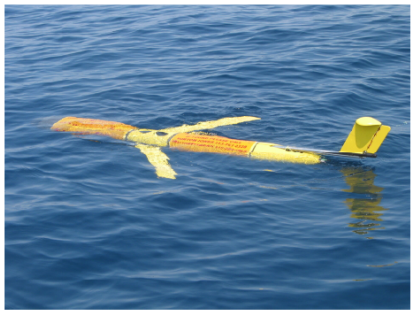
data from an area of predefined size around them. Sensor Networks have been successfully used in Precision Agriculture for crop inspection in warehouses [11], in rainforests for measuring $CO_2$ flux [12] as also in surveillance tasks [13]. In a WSN each sensor is able to communicate with the others and often a central station that is responsible for collecting and processing the data obtained by the sensors. Although this approach, contrary to the others provides autonomy in data collection in practice exhibits severe limitations. Two of the most important ones are: 1) the constrained mobility of the sensors and 2) the increased number of sensors required when large areas are considered.

On the other hand, mobile agents like those shown in Figure 1-1 are able to reach every part of the area and provide coverage even at difficult accessible areas. In many situations, however, a single agent may not always be able to successfully complete the task on its own. In other cases multi-robot teams may provide a better coverage result or complete the task at a compatible amount of time. Nonetheless, their control requires more sophisticated techniques and poses new challenges to the research community [14].


**(a)** Ground Vehicle for Warehouse Applications [15]

**(b)** UAV for weed detection [16]


**(c)** Networked Robotic Boats [6]



**(d)** Autonomous Underwater Vehicle (Glider) [17]

**Figure 1-1:** Different Platforms used for Surveillance Tasks

In literature multi-robot control problems have been extensively studied over these years. Depending on the amount of information available to agents, two main solution approaches are proposed, namely the *centralized* and *distributed* approach. In the first method a single agent is responsible for collecting the information from the others, plan the actions of team and assign a task to each agent. In the second method agents design their action based on their own information and the information received from their neighbors. In general, centralized methods provide better solutions due to the global information available. However, they usually not scale well with the number of agents in the team. Distributed approaches on the other often sacrifice performance since their solutions rely on local information. Nonetheless,

a great advantage of the these methods is the inherent robustness to single-agent hardware failures. Acknowledging the advantages and limitations of each method in this thesis both a centralized and distributed approach to the persistent coverage task is presented. Before that we review the current State of the Art in the field.

## 1-1 State of the Art

Coverage Control addresses the problem of designing agents' moves so as every point in the area is covered either once (*dynamic coverage control problems*) or multiple times (*persistent coverage control problems*) or until a desired level of coverage is locally reached (*dynamic coverage control problems*). This thesis focuses on the persistent coverage control problem. However, as several methods of dynamic coverage control could be applied in persistent coverage tasks after slight modifications a summary on the most important work in the dynamic coverage control field is presented.

In the following Subsections several methods are presented addressing the problem of dynamic coverage and persistent coverage control applied in 2D areas. In these methods the agents are considered equipped with sensors of finite range. Moreover, a point in the environment is generally considered *covered* at time step $k$ if it is included in the sensing area of an agent at $k$ while a point is *visited* at $k$ if an agent's position is identical to the coordinates of the point at time step $k$.

### 1-1-1 Dynamic Coverage Control Problems

Dynamic coverage control considers the problem of deciding agents' moves so as every point in the area is covered only once or until a pre-defined level of coverage is reached. In the first kind of tasks the area is decomposed into a number of cells and agents cover the cell following a lawn-mowing pattern shown in Figure 1-2. The task terminates when every cell is covered.

In [18] a review on the recent work for complete coverage is presented and categorized based on the type of decomposition applied to the area to: *heuristic*, *approximate, exact and semi-approximate* decomposition methods. In these methods the number, size and boundaries of the cells may be known a-priori by the agents or found on-line. In [19] a polygonal area is considered and offline decomposed into polygonal sub-regions based on the agents' sensing abilities. In [20] the agents define the start and end of each cell online. A team-coverage method is considered in which two agents search for the end point of the cell while the others sweep the area. When the end point is found 2 new cells open and the team is split to cover them. In the same article an auction-based, centralized method is also proposed. Here the area is decomposed in stripes and agents explore them so as to identify the start and end point of the cells. When part of the cell is not accessible the responsible agent calls an auction assigning the exploration task of this part to an other, task-free agent.

In the second category of coverage methods the sensor model of the agents is considered known. Then, every point in the area is associated to a level of coverage equal to the sum of the levels of coverage provided by the agents over a finite horizon. Considering a pre-defined, desired level of coverage the goal is to plan agents' moves so as the error between the actual and desired level of coverage is minimized over the area. To achieve this in [21] a local control

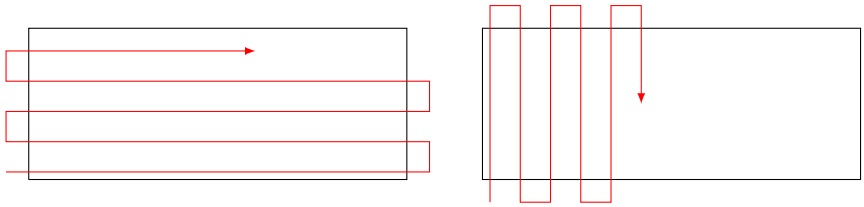


**Figure 1-2:** Lawn-Mowing Patterns

law is introduced based on the gradient of the error forcing agents move towards points with non-zero error. A feedback control law is also introduced for agents to escape local minima and move towards less covered areas. Authors guarantee error convergence to zero when the agents have full or partial coverage/position information from their neighbors. In [22] a similar approach is followed when agents are equipped with anisotropic sensors. In [23] a distributed method is presented based on the gradient control law of [21] and combined with a global strategy defining the order with which points in the area should be covered.

### 1-1-2 Persistent Coverage Control Problems

Persistent coverage control addresses the problem of planning the agents' actions so as every point in the area of interest is covered more than once. Over these years, existing work in the field has focused on the design of agents' paths/trajectories that maximize either the frequency of visits to a finite set of points or the amount of coverage offered by the agents over the area. In this Section several methods from both categories are presented with an emphasis given on the second approach due to its close relation to the subject of this thesis. The Section concludes with a summary of selected methods that differentiate from the aforementioned solution approaches.

**Visitation Frequency Approach** In this approach a finite set of points is considered and the goal is to design agents' paths so as the maximum time elapsed since the last visit at a point is minimized over the set. Here, a graph $G(V, E)$ is introduced with $V$ the set of points to be visited and $E$ the set of edges, with each edge expressing whether two points are accessible from one another. In [24] a Hamiltonian path is designed over the graph $G$ satisfying the frequency of visitation objective. In the same work a partition-based strategy is proposed in which each agent is responsible for visiting the subset of points in $V$ it was assigned to by following a Hamiltonian path. In [25] a more detailed study on coverage methods in graph based environments is presented and polynomial time methods are proposed for the design of agents' paths in chain, tree and cyclic graphs. A greedy policy is introduced in [26]. Here agents decide their next move based on the distance to the point and the time since it was last visited. For the first time in this work the non-holonomic constraints of the agents are considered and their effect on the choice of the next point is studied. Finally, in [27] a method guaranteeing a constant factor approximation of the optimal paths is proposed such that each point of $V$ is visited with a pre-defined priority.

**Coverage Level Approach** Here the goal is to design agents' paths/trajectories so as a time dependent quantity characterizing the environment is maintained at a desired level over the

area. This quantity, called *coverage level* could be the amount of dust in a cleaning task, the temperature or the accuracy of information in a heating or an information gathering task respectively. As time passes the coverage level at every point changes forcing agents to constantly move around the area to keep its coverage level close to the objective. This unique characteristic distinguishes persistent coverage from dynamic coverage tasks as the first never terminate.

In literature the available work on the subject could be further classified into two categories. In the first category agents follow either a single or multiple pre-defined, closed paths and speed controllers are designed so as the coverage level of every point in a finite set is asymptotically bounded. In the second category the area is decomposed into sub-regions and each agent is assigned to a region. Agents exchange coverage level information with their neighbors and update their own. Then, based on the updated information each agent designs its path such that it passes through less-covered points in its region.

In [28] a finite set of points is considered and each point is assigned to an uncertainty value that grows when the point is not covered and decreases when included in the sensing area of an agent. The agents are supposed to move along pre-defined, closed paths. The agent's velocity is parametrized by a set of basis functions. Then, the parameters of these functions are found as the solution of a linear program that takes into consideration agents' speed limits while guaranteeing the uncertainty decrease. In [29] agents move along a single path and speed controllers are defined as in [28] guaranteeing the coverage level of every point of a finite set to be asymptotically maximized and every other point in the area to be periodically covered. In [30] a similar approach to [29] is followed with a non-linear coverage level model.

In all these methods the closed paths are designed offline to satisfy a variety of objectives. Examples of methods for designing closed paths are found in [31], [32]. In [31] a given, closed path is considered and an adaptive controller is designed changing the shape of the path and forcing it to concentrate on dynamic areas where points change at a high rate. In [32] a set of polygonal paths are designed guaranteeing periodic coverage of every point in the area. These paths are suitable for obstacle-free areas or areas with known, static obstacles. However, their implementation is limited or even prohibitive in dynamic environments (environments with moving obstacles). In case of a persistent coverage task if an obstacle blocks the way of an agent along its path, path re-planning in addition to speed controller re-design is required. The same holds in case of a sudden failure of an agent.

The latter problem is implicitly addressed in the second category [33], [34] in which distributed methods are designed for planning paths passing through less covered points in the area when the network is always connected. These methods are described by three major steps. In the first step agents compute the coverage level of every point in the area and exchange the information with their neighbors. In the second step an update of the information is made and a second round of information exchange is proposed so as areas simultaneously covered by many agents are identified. In that way agents ensure that the coverage level of every point is as accurate as if global coverage information was available. In the last step a path planning solution is proposed. In [34] a feedback control law is introduced that comprises of a term forcing agents to move towards points that are most profitable in terms of coverage and another term guaranteeing the transition to these points is made through less covered areas. In [33] an optimal path planning method is proposed over the set of candidate goal points improving the coverage level. Here, contrary to [34] the goals are chosen based

on how profitable the complete paths towards the goals are. A potential field function is introduced that takes into consideration the coverage profit at every point and the position of any obstacles in the area. Then, the path is computed by following the gradient descent of the function and the path with the highest coverage profit is chosen.

**Other Approaches** Other methods considering the coverage level dynamics of the area are found in [35] in which Infinitesimal Perturbation Analysis is used for obtaining optimal ellipsoidal trajectories with respect to the total coverage level of the area. In [36], [37] the dynamic coverage methods presented in Section 1-1-1 are considered and modified to accommodate the dynamics of the coverage level. In these papers, paths are planned to follow the gradient of the coverage error towards reaching less-covered points in the area. Finally, in [38] a team of heterogeneous agents is considered and a distributed method is designed. Authors follow the first two steps proposed in [34] for providing agents the most accurate coverage level information. However, they differentiate themselves in the 3$^{\text{rd}}$ step by letting agents' paths follow the gradient of the coverage level error while guaranteeing collision avoidance.

## 1-2 Problem Statement

The coverage level methods presented above may be further categorized based on the coverage frequency information they offer in two main categories. In the first category we consider all methods: 1) guaranteeing the desired level of coverage while 2) providing information on how often each point in the area is covered. These objectives are generally achieved when predefined paths are considered and found to be limited to static environments [29], [30], [39]. On the other hand, the second category includes all methods that aim at maintaining the coverage level over the area at a desired level by planning agents' actions online. Although these methods are easily implementable in complex environments they do not provide specific information on the coverage frequency of the points in the area [33], [34], [36], [37], [38].

In this Thesis an online path planning method is designed that offers complete coverage information for the area of interest in terms of coverage frequency and coverage level. This work is motivated by tasks requiring agents to repeatedly visit parts of the area within a maximum time interval for guaranteeing a desired level of coverage until their next visit. Examples of such tasks could be sterilizing an examination room in a hospital or provide measurements of at least a desired level of accuracy within a maximum time interval.

In general providing a desired constant level of coverage at every point of a continuous area would require agents' sensors/actuators to provide the same amount of coverage at every point in the area. However, in practice this is not possible. For this reason a predefined, finite set of points is considered. Here the coverage level of every point in the set decreases by a point-dependent, time-independent decay rate as time passes and resets to a constant value when visited by an agent (details can be found in Chapter 2). The goal of the task is to plan agents' moves so as every point in the set is visited before its coverage level drops below $\underline{Z}$. In Figure 1-3 the evolution of the coverage level of a point in the set is shown. For every time $t \geq t_i$ the coverage level of the point is less than $\underline{Z}$. Therefore, in order for the task to be successful an agent should visit the point before $t_i$.

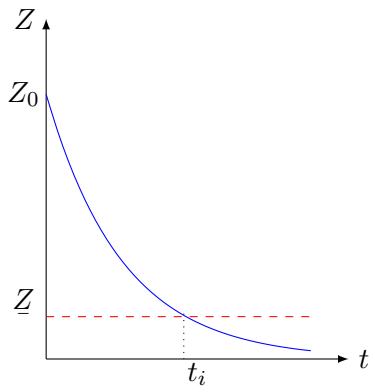



**Figure 1-3:** Relation of Coverage Time and Desired Level of Coverage

Given the finite set of points and the the coverage level dynamics of the points in the set, the goal of this thesis could be expressed as follows:

*Design agents' paths so as the coverage level of every point of the finite set is bounded from below by a desired level $\underline{Z}$.*

## 1-3 Thesis Outline

In Figure 1-4 a schematic representation of the structure of this thesis is presented.

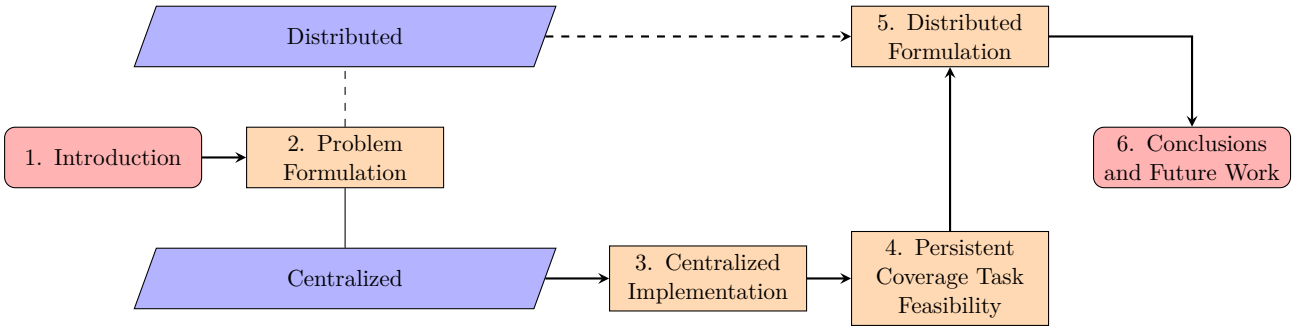



**Figure 1-4:** Schematic Outline of the Thesis

In Chapter 2 the area of interest, the coverage level dynamics and the admissible actions of the agents are introduced. A Mixed Integer Linear Program (MILP) is designed the solution of which provides the agents' paths over a finite planning horizon. Here, two different formulations of the problem are presented and their efficiency is validated in simulation.

In Chapter 3 further tests are conducted using the second formulation of the problem. The performance of the first feasible and optimal solution is evaluated in terms of coverage quality and computational effort of the solver. Finally, the scalability of the problem is studied with respect to the number of agents employed for the task, the planning horizon and the number of cells introduced due to the approximate decomposition implemented on the area of interest.

In Chapter 4 the feasibility of the persistent coverage task is studied. A set of constraints is added to the problem so as the agents' poses and the coverage level of the cells at the end of the horizon are limited to values belonging to a time-varying terminal constraint set. This set is considered part of a finite sequence of terminal sets the union of which defines a set of closed paths for the agents that satisfy the objective. The recursive feasibility of the problem is, then, proved and a two-step method is presented for constructing the closed paths of the sequence.

In Chapter 5 two distributed formulations of the persistent coverage task are presented. In these problems the agents solve in parallel a local path planning problem considering also their neighbors' actions and perform the first step of the path they computed for themselves. The difference on these methods lies on the communication model and the amount of information the agents have when planning their moves. The two problems are formally introduced and their performance is evaluated for different case scenarios and team sizes.

In Chapter 6 the basic results of this thesis are summarized and suggestions are made for future work.

# Chapter 2

# Problem Formulation

In Chapter 1 an overview of the recent work on the subject of persistent coverage control was presented. There, a vast amount of work proposed methods for designing agents' paths so as a desired level of coverage is maintained over the area. In this thesis a slightly different definition of the persistent coverage task is considered. Here the goal is to plan agents' moves so as the coverage level of every point of a finite set is bounded from below by a desired level $\underline{Z}$. The persistency of the task stems from the fact that the coverage level of every point decays over time forcing agents to constantly move to keep it above the desired level. The order with which points are visited depends on the choice of agents' paths. The latter are found as a solution to a *Mixed Integer Linear Program* (MILP) that aims at maximizing the coverage level of every point in the finite set over a finite horizon while minimizing subsequent visits at a point by different agents.

In this chapter the mathematical formulation of the problem is presented. Initially, the details on the area and agents employed are presented and the coverage level dynamics are introduced. A straightforward MILP formulation of the problem is presented. However, it is found to be too computational intensive even for small-sized problems. To increase computational performance, a second problem formulation is proposed that differs from the first one in the definition of the binary variables related to agents' position and heading. The chapter concludes with a discussion on the relative advantages and limitations of each method.

## 2-1 Problem Preliminaries

In this thesis a plane area is considered and a finite set of points over the area is introduced. A team of $n_r$ agents is employed for the task. Here, the agents' position and orientation is considered. At each time step the agents are able to move at a point of the finite set. However, the agents' moves are constrained in the sense that every point is not accessible by any other point in the finite set. In the following Subsections the chosen set of points is introduced and the agents' admissible moves in conjunction with the coverage level dynamics are presented.

### 2-1-1 The Area of Interest

A known, compact area $Q \subset \mathbb{R}^2$ is considered. The area is decomposed into a grid of $n_w$ square cells with $n_w = C \times L$ and $C$, $L$ the number of columns and rows of the grid respectively. A Cartesian Coordinate system is associated to the grid as shown in Figure 2-1. Each cell in the grid is expressed by an index $w \in I$ where $I = \{1, \ldots, n_w\}$. Let $c_w = (x_{i_1}, y_{i_2})$ express the coordinates of the center of cell $w$. Then, the index $w$ is found based on the following equation:

$$w = i_1 + (i_2 - 1) \cdot C$$

where $i_1 \in \{1, \ldots, C\}$, $i_2 \in \{1, \ldots, L\}$.

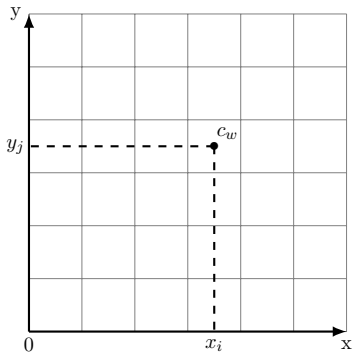



**Figure 2-1:** Representation of the Grid

When a grid decomposition of the area is implemented a crucial decision for the success of the task and the computational performance of the algorithm is the choice of the grid resolution. In [40] authors mention the problem of incomplete coverage when the size of the square cells is identical to the coverage area of an agent. This problem is usually related to the choice of agents' actions. For example if an agent reaches the center of a cell and makes a sharp turn at place (e.g by 90°) there might be points in the cell (e.g near the corners) that are left uncovered. To avoid this kind of problems, here the square cells are chosen smaller than the size of the coverage area of the agents. More specifically, here we consider agents with identical coverage abilities (*homogeneous* agents), able to cover an area of finite size. Considering the finite coverage area of an agent the following assumptions are made:

- Each square cell is encompassed by the the sensing area of an agent.
- When an agent is placed at $c_w$, $w \in I$ the area defined as the intersection of the agent's sensing area and a neighboring cell of $w$ is negligible.

The first assumption guarantees that every point in the cell is sensed when an agent is placed at its center while the second implies that each agent is able to cover a single cell per time step.

Based on the above we may introduce the following definition:

**Definition 2-1.1.** *A cell $w \in I$ is considered covered by the time an agent reaches its center $c_w$.*

Consequently, the area is considered *covered* if every center $c_w,\ w \in I$ is visited at least once. In this work except from guaranteeing a lower bound on the coverage level of the points in the set we expect agents to satisfy the fundamental purpose of coverage control which is covering every point in the area. To achieve this we consider as the finite set of points the set of the centers of the cells. In that way, if an agent visits the center of a cell $c_w,\ w \in I$ the following hold: 1) the coverage level of $c_w$ is lower bounded by $\underline{Z}$ and 2) every point in $w$ gets covered.

### 2-1-2 Introducing the Robotic Agents

A team of $n_r$ agents is employed for the task. Here agents are considered equipped with sensors/actuators able to cover all points of a cell. Moreover the following assumptions are made:

- Agents have full knowledge of their real position with respect to a world coordinate frame (e.g from GPS).
- Agents have full knowledge of the cell boundaries and coordinates of the centers of the cells.

Each agent is assigned to an index $r \in K$ with $K = \{1, \dots, n_r\}$ with its position denoted by $p_r^k$ and its heading by $\theta_r^k$. At each time step $k$ an agent is placed at the center of a cell $w \in I$ with its heading $\theta_r^k$ taking values from the set $\{0, \pi, \frac{\pi}{2}, \frac{3\pi}{2}\}$. Based on their current position and heading agents are able to perform one of the following actions:

- Stay at place (position and heading stays the same)
- Turn at place by 90°
- Move to an adjacent cell with respect to their heading

A directed graph $G'(V', E')$ is introduced with each node representing a possible position and heading $(x_r^k, y_r^k, \theta_r^k)$ of an agent $r$ and every edge an admissible move. More specifically, we can define the following nodes:

- $(x_{i_1}, y_{i_2}, \theta_1),\ i_1 \in \{1, \dots, C\},\ i_2 \in \{1, \dots, L\},\ \theta_1 = 0$
- $(x_{i_1}, y_{i_2}, \theta_2),\ i_1 \in \{1, \dots, C\},\ i_2 \in \{1, \dots, L\},\ \theta_2 = \frac{\pi}{2}$
- $(x_{i_1}, y_{i_2}, \theta_3),\ i_1 \in \{1, \dots, C\},\ i_2 \in \{1, \dots, L\},\ \theta_3 = \pi$
- $(x_{i_1}, y_{i_2}, \theta_4),\ i_1 \in \{1, \dots, C\},\ i_2 \in \{1, \dots, L\},\ \theta_4 = \frac{3\pi}{2}$

Then, for arbitrary center coordinates $(x_{i_1}, y_{i_2})$ depending on the heading $\theta_{i_3},\ i_3 \in \{1, 2, 3, 4\}$ the admissible moves are defined as in Figure 2-2. For simplicity we omit $x, y, \theta$ and use only the indexes $i_1, i_2, i_3$. An example of the graph $G'$ for a $2 \times 2$ grid is presented in Figure 2-3. The graph consists of 16 nodes with each node being expressed by the triple of indexes $(i_1, i_2, i_3)$. For each node the admissible moves are defined as shown in Figure 2-2. As discussed before, nodes corresponding to different cells are connected only if the centers of the corresponding cells share the same x or y coordinate.

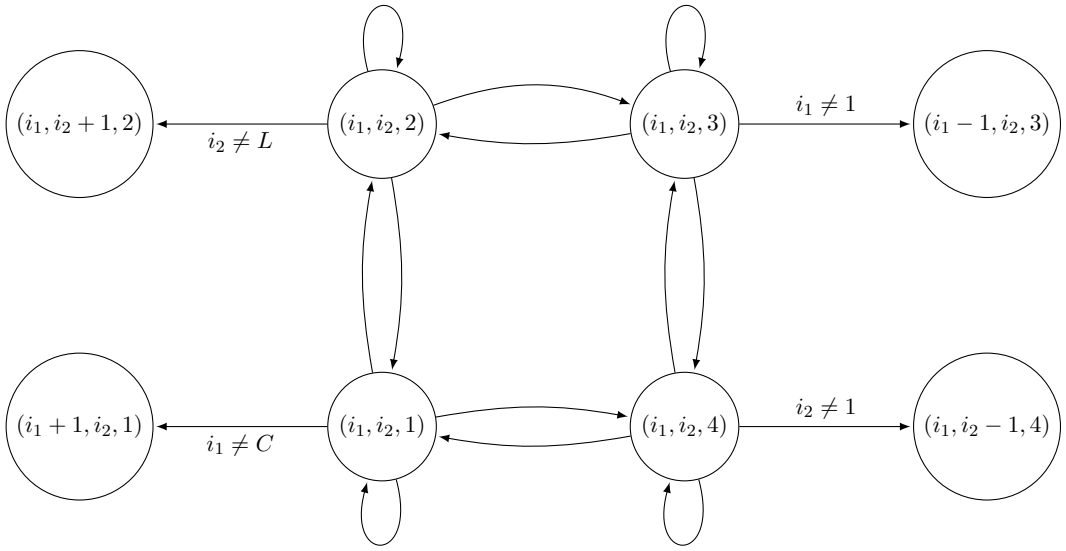



**Figure 2-2:** Representation of the admissible moves for an arbitrary node $(x_{i_1}, y_{i_2}, \theta_{i_3})$, $i_3 \in \{1, 2, 3, 4\}$.

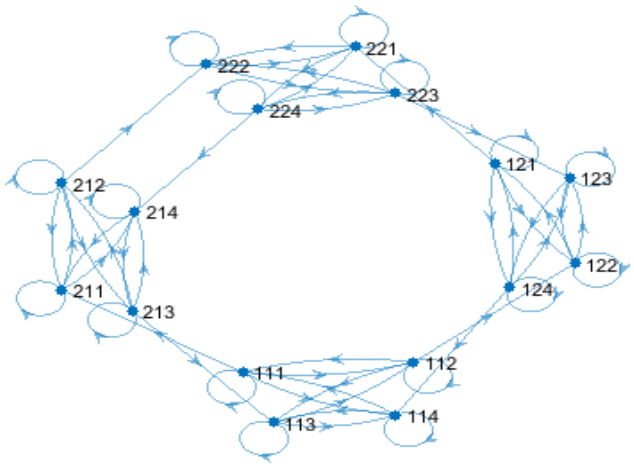



**Figure 2-3:** The Graph defining agents' moves for a $2 \times 2$ Grid

### 2-1-3 Coverage Level Dynamics

A positive value called *coverage level* is assigned to every cell center $c_w$, $w \in I$. This value, represented by $Z(c_w, k\tau)$ may express the amount of dust at $c_w$ for a cleaning task, the accuracy of information in an information gathering task or the temperature in a heating task. Here $\tau$ is the period at which the the coverage level dynamics are discretized. Considering agents moving at a constant speed, the period is equal to the time required for an agent to move from the center of a cell to an adjacent plus the time required for an agent to cover the cell center. The latter amount is considered negligible in information gathering and surveillance tasks or known and same for every cell(e.g in tasks like heating or watering). To

simplify the notation the coverage level of a cell center $c_w,\ w \in I$ at time instant $k\tau$ will be denoted by $Z(w,k)$ where $k$ is the time step and the *coverage level of $c_w$ at time step $k$* will be expressed for simplicity as the *coverage level of cell $w$ at time step $k$.*

In this work the coverage level of $w$ decreases over time by a constant value $d_w \in (0,1)$ called *coverage decay factor*. This factor expresses "how important" is to cover $w$. A low value of $d_w$ results in a sharp decrease of $Z(w,k)$ forcing agents to visit $w$ more frequently while a higher value of $d_w$ is assigned to cells requiring less attention. When $w$ is covered by an agent its coverage level resets to a positive, constant value $\bar{Z}$ such that $\bar{Z} > \underline{Z}$. Based on the above the evolution of the coverage level of every cell $w$ over time is defined as follows:

$$Z(w,k) = d_w(1-\sigma_w^k)Z(w,k-1) + \sigma_w^k\ \bar{Z}, \quad w \in I \tag{2-1}$$

where $\sigma_w^k$ is a binary variable defined as:

$$\sigma_w^k = \begin{cases} 1, & \exists\ r \in K : p_r^k = c_w \\ 0, & \text{otherwise} \end{cases}$$

Initially, cells are fully covered with $Z(w,0) = \bar{Z},\ w \in I$. As time passes the coverage level of every cell $w$ decreases. In order to maintain a low coverage bound $\underline{Z}$ at every cell agents are forced to visit each cell $w$ at most after $t_w$ time steps with $t_w$ found as:

$$t_w = \left\lfloor \frac{\ln \underline{Z} - \ln \bar{Z}}{\ln d_w} \right\rfloor, \quad w \in I \tag{2-2}$$

Here, $t_w$ could be considered as the worst case length of the time interval between two subsequent visits at cell $w$. We call this value the *age* of cell $w$. At each time step of this interval the coverage level of $w$ decreases monotonically. When $w$ gets covered its coverage level resets to $\bar{Z}$. Thus, $\bar{Z}$ is its maximum value. Throughout this thesis we aim at planning paths maintaining the coverage level of cells at least equal to $\underline{Z}$. For this reason, we require $Z(w,k)$ to be bounded as:

$$Z(w,k) \in [\underline{Z}, \bar{Z}], \quad w \in I \tag{2-3}$$

An example of the coverage level evolution over time is presented in Figure 2-4. Here we have: $\bar{Z} = 100, \underline{Z} = 20$ and $d_w = 0.8$. As expected the coverage level decreases over time until $t_w + 1$ when an agent covers the cell for the first time resetting its coverage level to $\bar{Z}$. After the agent leaves $w$ the coverage level starts dropping again. However, this time its value resets to $\bar{Z}$ before it reaches close to $\underline{Z}$. This is achieved by having an agent visiting the cell exactly at $t_w - 1$.

Considering now the total number of cells and the corresponding coverage level values the problem of defining when to visit each cell becomes more complex. Therefore, a more sophisticated method is needed for assigning agents to cells. In the following section an optimization problem is designed that plans agents' paths over a finite planning horizon $N$ so as the coverage level of the cells over $N$ is maximized.

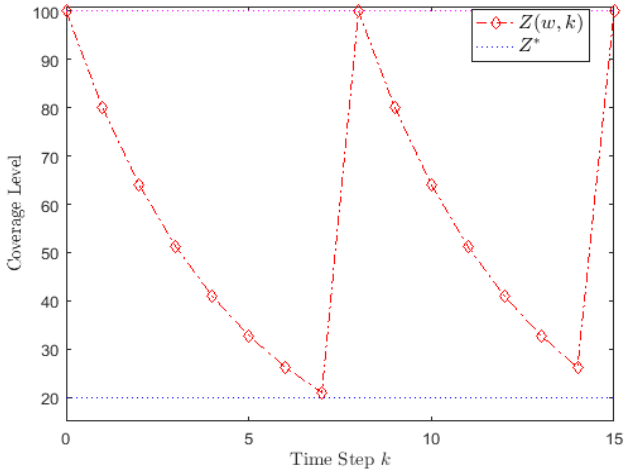



**Figure 2-4:** Evolution of the Coverage Level value over Time

## 2-2 Problem Formulation I

In this section a first step is made towards designing an optimization problem the solution of which defines the agents' paths that guarantee a lower bound on $Z(w,k), w \in I$ over a finite horizon of $N$ time steps. Here a set of binary variables is introduced expressing whether $(x_{i_1}, y_{i_2}, \theta_{i_3})$, $(i_1, i_2, i_3) \in V'$ is the state of agent $r$ at time step $k$ for every $r \in K$, $k \in \{1, \dots, N\}$. Then, a Mixed Integer Linear program (MILP) is designed that aims at maximizing the coverage level of every cell over the horizon while penalizing coverage of the same cell by different agents on consecutive time steps.

To simplify the notation of the problem we will introduce a single index for every node $(i_1, i_2, i_3) \in V'$ as follows:

$$q = 4\,(i_1 - 1) + 4\,(i_2 - 1)\,C + i_3, \tag{2-4}$$

$i_1 \in \{1, \dots, C\}, i_2 \in \{1, \dots, L\}, i_3 \in \{1, 2, 3, 4\}$. The new graph with nodes the 1-index poses of the agents (position and heading) will be denoted by $G(V, E)$. Moreover, we will define the adjacency matrix $A = (a_{qq'})$, $q, q' \in V$ corresponding to graph $G$. Here $a_{qq'} = 1$ if there exists an admissible transition from pose $q$ to $q'$ in $E$ (represented as $(q, q') \in E$). Otherwise, $a_{qq'} = 0$.

For each cell center there exist 4 nodes in $V$, one for every possible heading. Given the agent's state $q \in V$ the index of cell $w$ is found as:

$$w = \left\lceil \frac{q}{4} \right\rceil, \quad q \in V \tag{2-5}$$

A set of binary variables $\delta_q^r(k)$ is introduced for every $r \in K$, $k \in \{1, \dots, N\}$ as follows:

$$\delta_q^r(k) = \begin{cases} 1, & \text{if } (p_r^k, \theta_r^k) = (x_{i_1}, y_{i_2}, \theta_{i_3}) \\ 0, & \text{otherwise} \end{cases}$$

with $p_r^k$ the position and $\theta_r^k$ the heading of agent $r$ at time $k$ and $q \in V$ the node corresponding to state $(x_{i_1}, y_{i_2}, \theta_{i_3}) \in V'$.

At each time step a cell can host at most one agent. Therefore, if an agent $r$ is about to visit cell $w$ at time step $k$, this cell is excluded from the cell choices of the other agents. This can be formed mathematically as follows:

$$\sum_{q=4\ (w-1)+1}^{4w} \sum_{r \in K} \delta_q^r(k) \leq 1, \quad w \in I,\ k \in \{1, \dots, N\} \tag{2-6}$$

An agent's pose (position and heading) is expressed by a node of graph $G$ and is unique per agent and time step. This is equivalent to the following constraint:

$$\sum_{q \in V} \delta_q^r(k) = 1, \quad k \in \{1, \dots, N\},\ r \in K$$

Each agent decides its next pose among the set of admissible poses related to its current position and heading. The admissible transitions between 2 nodes are defined using the adjacency matrix $A$. For any $q \in V$ the pose $q' \in V$ is an admissible pose if the element $a_{qq'}$ of matrix $A$ is equal to 1. Therefore, if the position and heading of agent $r$ is expressed by node $q$ at time step $k$ its future pose at $k+1$ will be one of the nodes $q' \in V$ such that $a_{qq'} = 1$. This is encapsulated in the following constraint:

$$\delta_q^r(k) - \sum_{q' \in V} a_{qq'}\ \delta_{q'}^r(k+1) \leq 0,\ k \in \{1, \dots, N-1\},\ q' \in V,\ r \in K$$

The choice of agents' future move is based on the coverage level of every cell in the grid. As time passes the coverage level of the cells decreases forcing agents to visit them and reset their coverage level to $\bar{Z}$ at most after $t_w$ time steps. The switching between the "coverage level decreasing mode" and the "reset mode" is controlled by a binary variable $\sigma_w^k \in \{0, 1\}$ related to agents' current state as follows:

$$\sigma_w^k - \sum_{q=4\ (w-1)+1}^{4w} \sum_{r \in K} \delta_q^r(k) = 0,\ w \in I, k \in \{1, \dots, N\} \tag{2-7}$$

The above constraint guarantees that $\sigma_w^k = 1$ if there exists a single agent placed at $c_w$ irrespective of its heading. Moreover, due to constraint (2-6) if it exists, this agent will be unique. Substituting (2-7) to (2-1), we have the coverage level dynamics explicitly dependent on the binary variables expressing the agents' poses. Therefore, we consider variables $\sigma_w^k$ redundant and will not include them as extra variables in the optimization problem.

In our task we require the coverage level to be always above a desired level $\underline{Z}$. Considering the coverage level dynamics introduced in (2-1), the constraint $Z(w,k) \geq \underline{Z}$ becomes quadratic due to the term $\sigma_w^k \cdot Z(w, k-1)$ since $\sigma_w^k,\ Z(w, k-1)$ are dependent on the agents' poses that are the variables of the problem. To resolve this issue and keep the constraints linear we introduce a set of mixed variables $\alpha_w(k),\ w \in I,\ k \in \{1, \dots, N\}$ such that $\alpha_w(k) = \sigma_w^k\ Z(w, k-1)$.

Considering (2-7) the variables $\alpha_w(k)$ take the value $\sigma_w^k\ Z(w,k-1)$ if and only if the following inequalities are satisfied:

$$\begin{aligned}
&\alpha_w(k) \leq \bar{Z} \sum_{q=4\ (w-1)+1}^{4w} \sum_{r\in K} \delta_q^r(k)\\
&\alpha_w(k) \geq 0\\
&\alpha_w(k) \leq Z_w(k-1)\\
&\alpha_w(k) \geq Z_w(k-1) - \bar{Z}\left(1 - \sum_{q=4\ (w-1)+1}^{4w} \sum_{r\in K} \delta_q^r(k)\right)
\end{aligned}$$

Substituting the newly introduced variables to (2-1) and considering the coverage level response at time step $k$ with respect to the initial condition $Z(w,0)=\bar{Z}$, we are in position to define the following constraints:

$$\begin{aligned}
&Z(w,k) = d_w^k\ \bar{Z} + \sum_{t=1}^{k} d_w^{k-t}\left(-d_w\ \alpha_w(t) + \bar{Z} \sum_{q=4\ (w-1)+1}^{4w} \sum_{r\in K} \delta_q^r(t)\right)\\
&Z(w,k) \geq \underline{Z}
\end{aligned}$$

$w \in I,\ k \in \{1,\ldots,N\}$.

Here, we also aim at penalizing agents' intention to cover the same cell at consecutive time steps. To achieve this, we introduce a set of binary variables $\mu_w^r(k)$ as follows:

$$\mu_w^r(k) = \sum_{q=4(w-1)+1}^{4w} \sum_{q'=4(w-1)+1}^{4w} \sum_{\substack{r'\in K\\ r\neq r'}} \delta_q^r(k)\ \delta_{q'}^{r'}(k+1)$$

$w \in I,\ k \in \{1,\ldots,N-1\}, r \in K$. Then, $\mu_w^r(k)$ is equal to 1 if there exists an agent $r'$ covering cell $w$ at time $k+1$ after $r$ covered the same cell at time step $k$. The above definition is equivalent to having the following constraints satisfied:

$$\begin{aligned}
\mu_w^r(k) - \sum_{q=4(w-1)+1}^{4w} \delta_q^r(k) &\leq 0\\
\mu_w^r(k) - \sum_{\substack{r'\in K\\ r\neq r'}} \sum_{q'=4(w-1)+1}^{4w} \delta_{q'}^{r'}(k+1) &\leq 0\\
\sum_{q=4(w-1)+1}^{4w} \delta_q^r(k) + \sum_{\substack{r'\in K\\ r\neq r'}} \sum_{q'=4(w-1)+1}^{4w} \delta_{q'}^{r'}(k+1) - \mu_w^r(k) &\leq 1
\end{aligned}$$

Having introduced the set of variables and constraints of this problem we are now in position to introduce the objective function $J(x)$. Here $x$ is the variable vector with $x = \begin{bmatrix}\delta^T & \alpha^T & \mu^T\end{bmatrix}^T$ and $\delta^T$, $\alpha^T$, $\mu^T$ defined as:

$$\delta^T = \begin{bmatrix}\delta_1^1(1) & \ldots & \delta_{4n_w}^1(N) & \delta_1^2(1) & \ldots & \delta_{4n_w}^2(N) & \ldots & \delta_{4n_w}^{n_r}(N)\end{bmatrix}^T$$

$$\alpha^T = \begin{bmatrix} \alpha_1(1) & \dots & \alpha_{n_w}(1) & \alpha_1(2) & \dots & \alpha_{n_w}(2) & \dots & \alpha_{n_w}(N) \end{bmatrix}^T$$
$$\mu^T = \begin{bmatrix} \mu_1^1(1) & \dots & \mu_{n_w}^1(1) & \mu_1^1(2) & \dots & \mu_{n_w}^1(N-1) & \mu_1^2(1) & \dots & \mu_{n_w}^{n_r}(N-1) \end{bmatrix}^T$$

In this problem the goal is to maximize the total coverage level of the area over the optimization horizon while penalizing unnecessary coverage of cells covered at subsequent time instants. Based on that, the objective function is defined as:

$$J(x) = \sum_{k=1}^{N} \sum_{w \in I} Z(x, w, k) - \beta \sum_{k=1}^{N-1} \sum_{w \in I} \sum_{r \in K} \mu_w^r(k) \tag{2-8}$$

where $\beta$ is a positive weight expressing how important is to penalize agents' intention to cover cells already covered by other members of the team at the previous time step.

Considering the initial states of the agents and the initial coverage level of the area such that the following hold:

$$\delta_{q_r}^r(0) = 1, \quad r \in K$$
$$Z(w, 0) = \bar{Z}, \quad w \in I$$

we may introduce the complete MILP formulation of the problem as follows:

$$\max_x J(x) \tag{2-9}$$

subject to:

$$\sum_{q \in V} \delta_q^r(k) = 1$$
$$\forall k \in \{1, \dots, N\},\ \forall r \in K \tag{2-9a}$$

$$\delta_q^r(k) - \sum_{q' \in V} a_{qq'}\, \delta_{q'}^r(k+1) \leq 0,$$
$$\forall k \in \{1, \dots, N-1\},\ \forall q' \in V,\ \forall r \in K \tag{2-9b}$$

$$\delta_{q_r}^r(0) - \sum_{q \in V} a_{q_r q}\, \delta_q^r(1) = 0$$
$$\forall r \in K \tag{2-9c}$$

$$\sum_{q=4\,(w-1)+1}^{4w} \sum_{r \in K} \delta_q^r(k) \leq 1$$
$$\forall w \in I,\ \forall k \in \{1, \dots, N\} \tag{2-9d}$$

$$\alpha_w(k) - \bar{Z} \sum_{q=4\,(w-1)+1}^{4w} \sum_{r\in K} \delta_q^r(k) \leq 0$$
$$\forall w \in I,\ \forall k \in \{1,\dots,N\} \tag{2-9e}$$

$$\alpha_w(k) \geq 0$$
$$\forall w \in I,\ \forall k \in \{1,\dots,N\} \tag{2-9f}$$

$$\alpha_w(k) - Z(w,k-1) \leq 0$$
$$\forall w \in I,\ \forall k \in \{1,\dots,N\} \tag{2-9g}$$

$$\alpha_w(k) - Z_w(k-1) + \bar{Z}\left(1 - \sum_{q=4\,(w-1)+1}^{4w} \sum_{r\in K} \delta_q^r(k)\right) \geq 0$$
$$\forall w \in I,\ \forall k \in \{1,\dots,N\} \tag{2-9h}$$

$$d_w^k\ \bar{Z} + \sum_{t=1}^{k} d_w^{k-t}\Big( - d_w\ \alpha_w(t) + \bar{Z} \sum_{q=4\,(w-1)+1}^{4w} \sum_{r\in K} \delta_q^r(t)\Big) = Z(w,k)$$
$$\forall w \in I,\ \forall k \in \{1,\dots,N\} \tag{2-9i}$$

$$Z(w,k) \geq \underline{Z}$$
$$\forall w \in I,\ \forall k \in \{1,\dots,N\} \tag{2-9j}$$

$$\mu_w^r(k) - \sum_{q=4(w-1)+1}^{4w} \delta_q^r(k) \leq 0$$
$$\forall r \in K,\ \forall w \in I,\ \forall k \in \{1,\dots,N-1\} \tag{2-9k}$$

$$\mu_w^r(k) - \sum_{\substack{r'\in K\\ r\neq r'}} \sum_{q'=4(w-1)+1}^{4w} \delta_{q'}^{r'}(k+1) \leq 0$$
$$\forall r \in K,\ \forall w \in I,\ \forall k \in \{1,\dots,N-1\} \tag{2-9l}$$

$$\sum_{q=4(w-1)+1}^{4w} \delta_q^r(k) + \sum_{\substack{r'\in K\\ r\neq r'}} \sum_{q'=4(w-1)+1}^{4w} \delta_{q'}^{r'}(k+1) - \mu_w^r(k) \leq 1$$
$$\forall r \in K,\ \forall w \in I,\ \forall k \in \{1,\dots,N-1\} \tag{2-9m}$$

$$\delta_{q_r}^r(0) = 1$$
$$r \in K \tag{2-9n}$$

$$Z(w,0) = \bar{Z}$$
$$w \in I \tag{2-9o}$$

$$\delta_q^r(k) \in \{0,1\}$$
$$\forall q \in V,\ \forall r \in K,\ \forall k \in \{1,\dots,N\} \tag{2-9p}$$

$$\alpha_w(k) \in [0,\bar{Z}]$$
$$\forall w \in I,\ \forall k \in \{1,\dots,N\} \tag{2-9q}$$

$$\mu_w^r(k) \in \{0,1\}$$
$$\forall r \in K,\ \forall w \in I,\ \forall k \in \{1,\dots,N-1\} \tag{2-9r}$$

## 2-3 Motivation for a New Problem Formulation

The optimization problem described above could be considered as a straightforward formulation for defining agents' paths, when the allowable actions and the coverage level evolution with respect to the initial conditions are defined. Although (2-9) is easily interpretable, in practice exhibits severe limitations that make it prohibitive for online planning. More specifically, even when small-sized problems are considered the computational time required for the problem to converge to the optimal solution is significantly high.

This problem is typical in mixed integer programming and strongly related to the efficiency of the available solution methods for such problems. As no method for immediately finding the optimal integer solution is available, existing methods rely on exhaustively solving a set of linear problems first [41]. One such method is the *Branch and Bound* method in which a search tree is defined with root node the original MILP problem and child nodes MILP problems that include the constraints of the original problem plus some constraints further restricting the value of the integer variables.

Initially, the original MILP problem is relaxed to a linear program (LP) for which methods like Simplex or Interior Point provide a solution in negligible time irrespective of the problem's size. If the linear relaxation of the problem is infeasible the same holds for the mixed integer problem. When the linear problem is feasible we can distinguish the following two cases:

- Every integer variable of the original problem assumes an integer value. In that case the solution of the linear relaxation and the MILP problem are identical and the Branch and Bound algorithm is terminated.
- There exists an integer-restricted variable of the original problem that assumes a fractional value. Then, a variable $x_i$ is chosen among the integer variables with the fractional value and the search tree branches on. Two new nodes are introduced with each node defined as an MILP problem containing all the constraints of the original problem plus an extra constraint that restricts the value of the chosen variable $x_i$. This constraint is

either defined as $x_i \leq \lfloor f \rfloor$ or $x_i \geq \lceil f \rceil$ where $f$ is the fractional value. Each subproblem is again relaxed to a linear problem and its solution is computed. If the objective value of the LP relaxation is no better than the best LP solution obtained so far then no further branching is implemented. Otherwise an integer variable with a fractional value is chosen and the procedure is repeated until each integer variable assumes an integer value.

As the size of the search tree grows exponentially over time the performance of the algorithm lies on its ability to early identify nodes that will not yield any improvement on the value of the objective function of the LP formulation. To achieve this, commercial solvers like IBM ILOG CPLEX, Gurobi etc use a variant of the Branch and Bound method called *Branch and Cut*. This method differs from Branch and Bound as the LP relaxation of each node of the search tree could be solved more than once after adding an extra set of inequality constraints called *cuts*. These cuts are satisfied by any solution with integer variables taking integer values but might be violated by solutions with the corresponding variables assuming fractional values. Therefore, by adding these constraints we aim at obtaining a different solution of the LP relaxation with less fractional integer variables.

An example of the Branch and Cut method is shown in Figure 2-5 for a simple maximization integer program with two variables. The grey area represents the *convex hull* of the set of the integer solutions. This set is the smallest set including all possible solutions of the integer problem. Initially, the problem is relaxed to a LP problem and the optimal solution is found to be the upper right corner of the polygon area. After that the LP relaxation is solved once more after adding 2 extra constraints (cuts) shown in green. These constraints restrict further the convex hull of the problem resulting in a new polygon area with an integer, feasible point on its boundaries. This point is the closest among all candidates to the LP optimum point thus the optimum of the integer program.

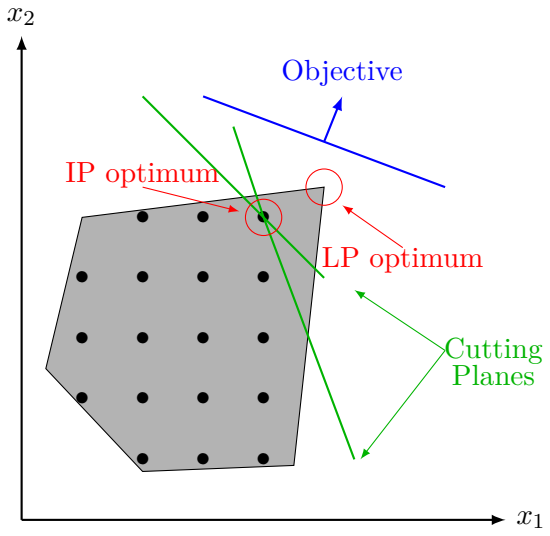



**Figure 2-5:** Example of Branch and Cut Method on an integer maximization problem with two variables [1]

Acknowledging the performance improvement the addition of cutting planes to Branch and

Bound has led to ([42], [43]), the operational research community has paid much attention on the design of different sets of inequality constraints (cuts). For a detailed study on the available cuts the reader may resort in [44].

Despite the increased performance of the Branch and Cut method, the formulation of the MILP problem still plays a critical role on the computational time required for finding the optimal solution. More specifically, the fast convergence of the algorithm depends on how close is the feasible set of the LP relaxation to the convex hull of the solution space of the MILP problem. In case these sets are identical the solution of the MILP and the LP problem is identical. A MILP formulation with this property is called a *tight* formulation. However, defining the convex hull of a MILP problem in practice requires a great number of cuts and computational effort making it more appealing to solve the original MILP problem. Therefore, experts aim at designing a formulation that closely approximates the convex hull while keeping the computational time required for the solution of the problem at a reasonable level.

An example of a less-tight formulation is problem (2-9). Even when a small grid of $3 \times 3$ cells is considered with 2 agents employed for the task the time required for the problem to converge is more than 1000 *sec.* Although the solution with the optimal objective value is often found at a relatively short amount of time the solver requires much time to exhaustively check every possible solution before discarding them for not improving the objective. To resolve this problem authors of [45] suggest experts to include a set of cuts in the problem before solving it. These cuts as aim at reducing the solution area of the LP relaxation of each node in the search tree so as to better approximate the convex hull of the solution space of the original problem. Furthermore, they can simultaneously activate more built-in the solver cuts for further strengthening the LP relaxations and providing better LP objective values.

Although in many problems introducing appropriate cuts may improve the computational time of the problem, in our case defining such cuts is not trivial. This is generally a result of the freedom agents have on deciding the order of the cells to be visited over the planning horizon. More specifically, in (2-9) agents are able to either stay at a cell (e.g for turning) or move to an adjacent so as the coverage level of each cell is always lower bounded by $\underline{Z}$. Contrary to other MILP methods agents do not need to wait a number of steps to visit a cell (e.g as in vehicle routing problems with time windows [46]) nor need to visit or return at the end of the horizon at a desired place (e.g at their depot [47]). This provides them the freedom to visit cells at any admissible order as long as the motion constraints are satisfied without paying attention on their final destination. As the goal is to maximize the total coverage level of the area the agents strive to keep a balance between visiting cells with high decay rates the earliest possible and keeping the coverage level of the other cells higher than $\underline{Z}$. These objectives could be satisfied by having agents following several different combinations of candidate paths. Therefore, the solver needs to exhaustively check every possible combination of paths before finding the optimal.

Aiming at maintaining a lower bound on the coverage of each cell, an expert could consider adding the following cuts:

$$\sum_{k=1}^{t_w+1} \sum_{r \in K} \sum_{q=4(w-1)+1}^{4w} \delta_q^r(k) \geq 1, \quad w \in I \tag{2-10}$$

$$\sum_{r\in K}\sum_{q=4(w-1)+1}^{4w}\delta_q^r(t)-\sum_{k=t+1}^{t+t_w+1}\sum_{r\in K}\sum_{q=4(w-1)+1}^{4w}\delta_q^r(k)\leq 0,\quad w\in I,\ t\in T_w \tag{2-11}$$

where $T_w=\{t\in[1,N]\cap\mathbb{Z}\mid t+t_w+1\leq N\}$

Here (2-10) guarantees that cell $w$ will be visited at most after $t_w$ steps while the second constraint makes sure that $w$ will be visited at most after $t_w$ steps from the time it was covered. Although these cuts might improve the computational time of the problem, they are only valid when $t_w\leq N-1,\ w\in I$. Then, for given decay factors $d_w$ and desired coverage level $\underline{Z}$ increasing the reset value $\bar{Z}$ forces every $t_w$ to exceed $N-1$. This removes the necessity of revisiting the cells in the area over the finite planning horizon providing agents more freedom on choosing their paths. Based on that finding an intuitive constraint set approximating the convex hull of the problem is found to be relatively complex and strongly dependent on the choice of the parameters of the problem. For this reason instead of improving the formulation (2-9) we propose a different problem formulation presented in the following section.

## 2-4 Problem Formulation II

Aiming at reducing the computational time of the problem a new formulation of the persistent coverage task is introduced. The novelty of this method lies on the following two factors:

- the definition of the binary variables related to the agents' poses
- the definition of the coverage level of the cells using a set of inequality constraints, known in operational research community as *big-M* constraints

Here a set of binary variables $x_{qq'}^k$ is introduced with each variable defining whether a transition from pose $q\in V$ to $q'\in V$ at time step $k$ is active or not as follows:

$$x_{qq'}^k=\begin{cases}1, & \text{if at time step } k\ \exists\ r\in K \text{ performing a transition from pose } q \text{ to } q'\\ 0, & \text{otherwise}\end{cases}$$

where $(q,q')\in E,\ k\in\{1,\dots,N\}$.

Contrary to the previous formulation here the binary variables are not explicitly defined for every agent but a single set of variables may describe the state transition of all agents in the team. This is possible for two reasons: 1) agents start from different cells and 2) each cell can host at most one agent per time step. Due to the latter a transition to a pose $q'\in V$ at time step $k$ is allowed only for a single agent. Therefore, when the complete pose transition history over the horizon is known starting from the final pose of an agent and propagating back in time we are in position to find its initial pose and thus identify its index.

Considering the binary variables $x_{qq'}^k$ we are in position to define a new set of binary variables $\mu_w^k$ as follows:

$$\mu_w^k=\sum_{(q,q')\in V_w}\sum_{(s,s')\in V_w\setminus V_w'}x_{qq'}^k\ x_{ss'}^{k+1} \tag{2-12}$$

where $V_w=\{(q,q')\in E\ :\ q\in V,\ q'=4(w-1)+1,\dots,4w\}$ and $V_w'=\{(q,q')\in E\ :\ q=4(w-1)+1,\dots,4w,\ \ q'=4(w-1)+1,\dots,4w\}$. These variables provide information

on whether a cell $w$ is visited by different agents at consecutive time steps $k, k+1$ or not. Definition (2-12) is quadratic on the variables $x^k_{qq'}$. For this reason a set of linear inequalities is introduced so as when satisfied the variables $\mu^k_w$ take the same values as in (2-12). These inequalities are defined as follows:

$$\mu^k_w - \sum_{(q,q')\in V_w} x^k_{qq'} \leq 0$$
$$\mu^k_w - \sum_{(s,s')\in V_w \setminus V'_w} x^{k+1}_{ss'} \leq 0$$
$$\sum_{(q,q')\in V_w} x^k_{qq'} + \sum_{(s,s')\in V_w \setminus V'_w} x^{k+1}_{ss'} - \mu^k_w \leq 1$$

$\forall w \in I,\ k \in \{1,\dots,N\}$.

In addition to the binary variables $x^k_{qq'}$ a different method for defining the coverage level of each cell is introduced. Here the coverage levels $Z(w,k)$ are included as variables of the problem. A set of linear inequalities is, then, introduced that aims at defining $Z(w,k)$ with respect to $Z(w,k-1)$ and the current poses of the agents rather than the agents' pose history (as in (2-9)). To achieve this a popular type of inequalities is used called *big-M* constraints. These constraints are usually introduced to limit the value of some continuous variables based on the value of a binary variable. These constraints have generally the following form:

$$\sum_i x_i - My \leq 0$$

where $x_i$ are continuous variables, $y$ is binary and $M$ is a positive, large constant chosen in a way that $\sum_i x_i \leq M$ is always satisfied for any value of $x_i$ when the binary variable $y = 1$.

Using these constraints we will try to define the value of $Z(w,k)$ based on whether an agent visited $w$ at time step $k$ or not. This is achieved by introducing the following constraints:

$$-Z(w,k) + \bar{Z} \sum_{(q,q')\in V_w} x^k_{qq'} \leq 0 \tag{2-13}$$
$$Z(w,k) - d_w\ Z(w,k-1) - \bar{Z} \sum_{(q,q')\in V_w} x^k_{qq'} \leq 0 \tag{2-14}$$
$$Z(w,k) - d_w\ Z(w,k-1) - (1-d_{\max})\ \bar{Z} \sum_{(q,q')\in V_w} x^k_{qq'} \geq 0 \tag{2-15}$$
$$Z(w,k) \leq \bar{Z} \tag{2-16}$$

$w \in I, k \in \{1,\dots,N\}$ and $d_{\max} = \max_{w\in I} d_w$.

If $\sum_{(q,q')\in V_w} x^k_{qq'} = 1$ then due to (2-13), (2-16) the coverage level $Z(w,k)$ will be equal to $\bar{Z}$ otherwise constraints (2-14), (2-15) guarantee that $Z(w,k) = d_w\ Z(w,k-1)$.

Considering the new variable vector $\bar{x}^T = \begin{bmatrix} b^T & z^T & \mu^T \end{bmatrix}^T$ with:

$$b^T = \begin{bmatrix} x^1_{11} & x^1_{12} & \dots & x^1_{4n_w\ 4n_w} & x^2_{11} & \dots & x^2_{4n_w\ 4n_w} & \dots & x^N_{4n_w\ 4n_w} \end{bmatrix}^T$$
$$z^T = \begin{bmatrix} Z(1,1) & Z(2,1) & \dots & Z(n_w,1) & Z(1,2) & \dots & Z(n_w,N) \end{bmatrix}^T$$

$$\mu^T = \begin{bmatrix} \mu_1^1 & \mu_2^1 & \dots & \mu_{n_w}^1 & \mu_1^2 & \dots & \mu_{n_w}^2 & \dots & \mu_{n_w}^{N-1} \end{bmatrix}^T$$

and the new objective function $J'(\bar{x})$ defined as :

$$J'(\bar{x}) = \sum_{k=1}^{N} \sum_{w \in I} Z(\bar{x}, w, k) - \beta \sum_{k=1}^{N-1} \sum_{w \in I} \mu_w^k$$

where $\beta$ is a positive weight expressing the importance of penalizing cell coverage at consecutive time steps we may introduce the new problem formulation as follows:

$$\max_{\bar{x}} J'(\bar{x}) \tag{2-17}$$

subject to:

$$\sum_{q \in V} \sum_{q' \in V} x_{qq'}^k = n_r \quad \forall k \in \{1, \dots, N\} \tag{2-17a}$$

$$\sum_{(q,q') \in V_w} x_{qq'}^k \leq 1 \quad \forall w \in I,\ \forall k \in \{1, \dots, N\} \tag{2-17b}$$

$$x_{qq'}^k - \sum_{(q',q'') \in E} x_{q'q''}^{k+1} \leq 0 \quad \{q, q' \in V :\ (q,q') \in E\},\ \forall k \in \{1, \dots, N-1\} \tag{2-17c}$$

$$-Z(w,k) + \bar{Z} \sum_{(q,q') \in V_w} x_{qq'}^k \leq 0 \quad \forall w \in I,\ \forall k \in \{1, \dots, N\} \tag{2-17d}$$

$$Z(w,k) - d_w\, Z(w,k-1) - \bar{Z} \sum_{(q,q') \in V_w} x_{qq'}^k \leq 0 \quad \forall w \in I,\ \forall k \in \{1, \dots, N\} \tag{2-17e}$$

$$Z(w,k) - d_w\, Z(w,k-1) - (1 - d_{\max})\, \bar{Z} \sum_{(q,q') \in V_w} x_{qq'}^k \geq 0 \quad \forall w \in I,\ \forall k \in \{1, \dots, N\} \tag{2-17f}$$

$$Z(w,k) \leq \bar{Z} \quad \forall w \in I,\ \forall k \in \{1, \dots, N\} \tag{2-17g}$$

$$Z(w,k) \geq \underline{Z}$$

$$\forall w \in I,\ \forall k \in \{1,\dots,N\} \tag{2-17h}$$

$$\mu_w^k - \sum_{(q,q')\in V_w} x_{qq'}^k \leq 0$$
$$\forall w \in I,\ k \in \{1,\dots,N-1\} \tag{2-17i}$$

$$\mu_w^k - \sum_{(s,s')\in V_w \setminus V_w'} x_{ss'}^{k+1} \leq 0$$
$$\forall w \in I,\ k \in \{1,\dots,N-1\} \tag{2-17j}$$

$$\sum_{(q,q')\in V_w} x_{qq'}^k + \sum_{(s,s')\in V_w \setminus V_w'} x_{ss'}^{k+1} - \mu_w^k \leq 1$$
$$\forall w \in I,\ k \in \{1,\dots,N-1\} \tag{2-17k}$$

$$\sum_{(q_r,q)\in E} x_{q_r q}^1 = 1$$
$$\forall r \in K \tag{2-17l}$$

$$Z(w,0) = \bar{Z}$$
$$\forall w \in I \tag{2-17m}$$

$$x_{qq'}^k \in \{0,1\}$$
$$\forall (q,q') \in E,\ \forall k \in \{1,\dots,N\} \tag{2-17n}$$

$$\mu_w^k \in \{0,1\}$$
$$\forall w \in I,\ \forall k \in \{1,\dots,N-1\} \tag{2-17o}$$

Constraint (2-17a) guarantees that only one pose transition per agent is allowed at each time step while (2-17b) makes sure that each cell hosts at most one agent per time step. Equation (2-17c) limits the choice of the pose for the next time step so as the transition from the current poses to be admissible. Constraints (2-17d)-(2-17g) define the coverage level values for each cell while 2-17h and (2-17i)-(2-17k) guarantee the lower bound of the coverage level and define the values of the binary variables $\mu_w^k$ respectively. Constraint (2-17l) defines the admissible pose transition for the first time step when the initial pose of each agent $r$ is expressed by node $q_r \in V$ of graph $G$. Finally, constraint (2-17m) defines the initial coverage level of each cell while (2-17n)-(2-17o) guarantee that $x_{qq'}^k,\ \mu_w^k$ are binary.

In Table 2-1 a comparison of the proposed formulations is made in terms of the number of binary and continuous variables introduced in each problem. While the number of continuous variables associated to the coverage level of the cells is the same for both formulations the number of binary variables significantly differs. This difference lies on the fact that the variables $x_{qq'}^k$ of formulation II associated to the agents' moves are independent of the number

of agents. The same holds for the binary variables $\mu_w^k$ required for penalizing subsequent coverage of the same cell by different agents.

On first sight the number of binary variables(BV) of formulation II is higher than the BV of formulation I. However, in practice this is true only when a small team of agents is considered ($n_r \leq 3$). As the computational time of MILP problems is generally dependent on the number of binary variables of the problem we expect formulation II to be more efficient when a large team of agents is employed for the task.

| Formulation | # BV for states | # BV for penalty | # CV | # constraints |
|---|---|---|---|---|
| I | $4n_r n_w N$ | $n_r n_w (N-1)$ | $n_w N$ | $< n_r(N+2) + (8n_r+5)n_w N$ |
| II | $\|E\|\ N$ | $n_w(N-1)$ | $n_w N$ | $N + n_r - 3n_w + (10n_w + \|E\|)N$ |

**Table 2-1:** Number(#) of Binary Variables(BV), Continuous Variables(CV) and constraints introduced in the proposed formulations. $|E|$ is the cardinality of the set of edges $E$ of graph $G$ and is equal to $16n_w - 2(C+L)$.

## 2-5 Experimental Efficiency Evaluation

In this section the performance of each optimization problem is evaluated in terms of the computational time required for each problem to reach its optimum. A grid of $6 \times 6$ cells is considered and 4 agents are employed for the task. Then, given different combinations of cell ages $t_w$ the computational times required for each case to converge to the optimal value are compared when formulations (2-9), (2-17) are considered. From now on we will refer to the persistent coverage problem discussed so far as the *Dense Persistent Coverage Problem*(DPCP).

To further examine how efficient the proposed formulations are a new persistent coverage problem is introduced, called *Sparse Persistent Coverage Problem*(SPCP). In this problem the goal is 1) to maximize the coverage level of a subset of cells called *Cells of Interest*(COI) and 2) minimize coverage of cells at subsequent time steps by different agents while 3) maintaining the coverage level of every COI above $\underline{Z}$. Here the *Cells of Interest* are defined as the cells whose decay factors are lower than 0.85, thus $d_w \leq 0.85$. Let the set of COIs be expressed by $I_c$. Then, the SPCP problem can be formulated in a similar way to (2-9), (2-17). However, in this problem the coverage level constraints and the part of the objective function related to the coverage levels are defined only for the cells of $I_c$.

### 2-5-1 Preliminaries

Here a known, square area $Q$ is considered with area size $24 \times 24$ m$^2$. A team of 4 agents is employed for its coverage with each agent equipped with sensors of finite, known sensing range. The area is decomposed into a grid of $n_w$ square cells. As mentioned in Section 2-1-1 the number of cells $n_w$ and the area size of the cells should be chosen such that: 1) each cell is encompassed by the sensing area of the agents and 2) every agent senses all points of at most 1 cell per time step. Here a grid of $6 \times 6$ cells is considered with cells of size 16 m$^2$.

By problem definition every cell can host only one agent per time step. For this reason, agents are initially placed at different cells (depots) on the right side of the area with their headings chosen to point towards agent-free cells in the grid. The exact initial poses of the agents are shown in Figure 2-7.

Let $\mathbf{t} \in \mathbb{R}^m_{>0}$ be a vector with elements the cell ages $t_w, w \in I$. Given a sample $(\mathbf{t}_1, \ldots, \mathbf{t}_{n_s})$ of size $n_s = 10$ we will evaluate the amount of time required for the optimization problem I and II to converge to the optimal solution. Since the number and initial pose of the agents is known $\mathbf{t}_l$ should be chosen with elements $t^l_w$ large enough for an agent to be able to reach $w$ from its initial position at most after $t^l_w + 1$ steps. Otherwise, the coverage level of $w$ will drop below $\underline{Z}$ and the problem will become infeasible. In that case it is advised to re-consider the values of $\mathbf{t}_l$ and ensure that a path to any cell $w$ exists with traversal time equal at most to $t^l_w + 1$ for any agent $r \in K$.

Admittedly, when $t^l_w$ are known there exist several combinations of $\bar{Z}$, $\underline{Z}$, $d_w$ so as (2-2) is satisfied. Here we consider $\bar{Z}$, $\underline{Z}$ known and equal to 300 and 20 respectively. Then, the goal is to assign to each $\mathbf{t}_l$, $l \in \{1, \ldots, n_s\}$ a vector $\mathbf{d}_l \in (0,1)^{n_w}$ with elements the coverage decay factors $d_w, w \in I$ such that:

$$t^l_w \geq \left\lfloor \frac{\ln \underline{Z} - \ln \bar{Z}}{\ln d^l_w} \right\rfloor, \quad w \in I$$

where $t^l_w$, $d^l_w$ are the age and decay factor of cell $w$ corresponding to the $\mathrm{l}^{\mathrm{th}}$ sample.

In Section 2-1-3 the choice of $d_w$ was discussed. There it was mentioned that a low value of $d_w$ forces agents to visit $w$ more frequently. Therefore, $d_w$ should be chosen so as the following always holds:

$$t_{w_1} \leq t_{w_2} \Leftrightarrow d_{w_1} \leq d_{w_2}$$

for any $w_1, w_2 \in I$. In addition to this constraint here $d^l_w$ are chosen so as $\bar{\mathbf{d}}_\mathbf{l} \geq 0.85$ where $\bar{\mathbf{d}}_\mathbf{l}$ is the mean value of vector $\mathbf{d}_l$, $l \in \{1, \ldots, n_s\}$.

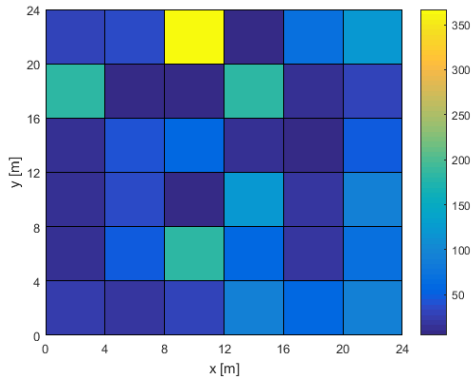



**Figure 2-6:** Example of Worst Case Coverage Time Steps $t_w$

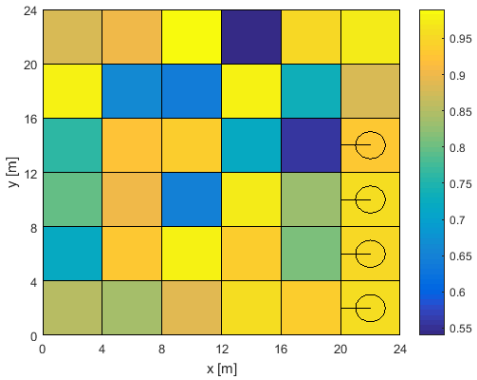



**Figure 2-7:** Coverage Decay Factors corresponding to $t_w$

An example of the maximum number of time steps $t_w$ spent before a cell $w$ is to be covered is shown in Figure 2-6. Here, most of the cells need to be visited at most after 5 to 120 time steps (cells in dark blue) and only a small number requires coverage after 180 steps. A clearer indication on the cells that need more frequent coverage is given in Figure 2-7. Here the cells

with low $t_w$ values are represented in blue and light green and are assigned to decay factors ranging from 0.55 to 0.8. In this example the lowest decay factor value 0.55 is assigned to cells with center coordinates $(18, 14)$, $(14, 22)$ as the time step before which they should be visited is equal to 5 steps.

### 2-5-2 Computational Results

In this section the computational results are presented for the DPCP and SPCP problem when expressed by the proposed formulations of Sections 2-2 and 2-4. The tests are run at Intel Xeon W-2145 3.70GHz CPU, 31GB RAM, Linux 4-4-0 64-bit operating system, MathWorks MATLAB 2018b 64-bit and the MILP programs are solved using the commercial solver GUROBI 8.0.1 64-bit for MATLAB.

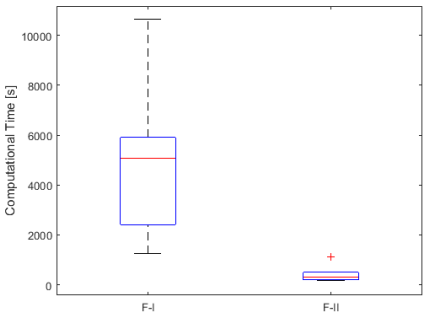
In general MILP problems are known for their long convergence times. To avoid excessive computational effort per case a time-limit is set on GUROBI equal to 3 hours. Therefore, the optimization problem will terminate either when the optimal value is found or when the time-limit is reached. In the latter case the solver will return the best solution obtained until the time-limit is reached.

In Figure 2-8a the computational time required for the DPCP to converge to the optimal solution is presented when DPCP is expressed by formulation I and II. Over the total number of cases only the 60% converge to the optimal solution when both formulations are considered. For these cases formulation II outperforms I as only 300 sec are required on average for the solver to find the optimum compared to 5000 sec of formulation I. For the rest of the cases and when problem (2-9) is considered the optimal solution is not found within the time limit. However, when formulation II is considered for the same cases the optimal paths are found within 440 sec on average as shown in Figure 2-8b.

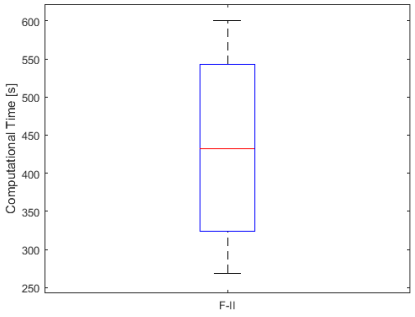
The superiority of formulation II is also evident from the computational results related to the SPCP problem. In this problem the optimal solution is found only for the 50% of the cases with both formulations. Here the difference on the computational time is even bigger with formulation II requiring only 60 sec to be solved compared to 5500 sec of formulation I (Figure 2-8c). For the rest 50% of the cases the solver needs more than 10800 sec to find the optimal solution (formulation I) while for formulation II the time is limited to 180 sec. Based on the above, we can conclude that formulation II performs significantly better in a variety of cases. Therefore, from now on and unless else stated this formulation will be used for our computations.

## 2-6 Conclusions

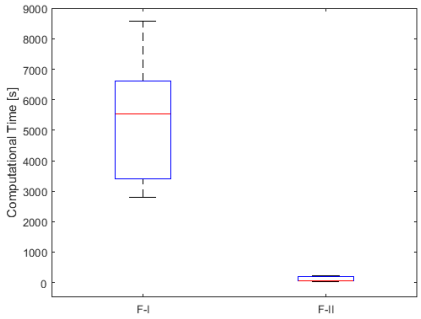
In this chapter the basic elements of the problem are introduced. A grid decomposition of the area is considered and a team of agents is employed for the task. At each time step agents may perform one of the following actions: 1) stay at place with the current heading, 2) turn at place by 90° or 3) move to an adjacent cell on the direction of their heading. Each cell of the grid is assigned to a value, called *coverage level* that decreases over time and resets to a constant value $\bar{Z}$ only when the cell is covered by an agent. Then, the goal of the task is to coordinate agents' actions so as each cell is completely covered with its coverage



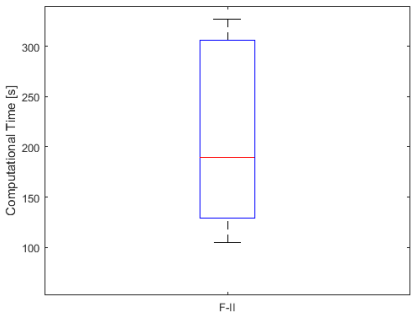

**(a)** DPCP:Cases converged to the Optimum




**(b)** DPCP:Performance of F-II when F-I reaches the Time Limit




**(c)** SPCP:Cases converged to the Optimum




**(d)** SPCP:Performance of F-II when F-I reaches the Time Limit

**Figure 2-8:** Evaluating MILP Efficiency in terms of Computational Time. Here *F-I* stands for Formulation I and *F-II* for Formulation II

level maintained above a desired level $\underline{Z}$. To achieve this a *Mixed Integer Linear* problem is introduced aiming at: 1) maximizing the total coverage level of the area over a finite planning horizon and 2) penalizing agents' desire to visit a cell covered at the previous time step by a peer.

Two problem formulations are introduced and their advantages and limitations are discussed. While formulation I can be characterized as a straightforward formulation of the problem in practice is found to be too computational intensive even for small-sized problems. To resolve this problem, we considered adding a set of inequality constraints, called *cuts* so as the solution space of the new problem could better approximate the convex hull of the solution space of the original problem. This, however, is found to be non-trivial as agents do not have any other restriction on the choice of the cells they have to visit than maintaining the coverage level of the cells above $\underline{Z}$. For this reason a new formulation of the problem is proposed. In this formulation the number of binary variables defining the agents' poses is independent on the team size. Therefore, as the number of agents increases the number of binary variables remains the same. By contrast, in formulation I this number is proportional to the number of agents. Therefore, we expect the first formulation to be less efficient.

This is verified by the computational results presented in Section 2-5-2. Two persistent coverage tasks are defined, namely the Dense and the Sparse that differ on the number of

cells whose coverage level must be lower bounded. In both tasks the results show superiority of formulation II over I. Therefore, from now on and unless else stated this formulation will be considered for our computations.

# Chapter 3

# Centralized Implementation

In this Chapter the performance of the DPCP problem is evaluated in terms of the coverage level of the area and the computational time for convergence. Here, the problem is solved in a *centralized* manner. A single agent is aware of the global position and coverage level information and responsible for planning the actions of the team. Under this framework three main experiments are conducted. In the first experiment the coverage performance is tested when different combinations of cell ages $t_w$ are considered. Each combination of $t_w, w \in I$ constitutes a *case*. For each case the number of cells, the planning horizon, the team size and the initial poses of the agents are known and pre-defined. As the computational time required for finding the optimal value of the optimization problem corresponding to each case is long, in the second experiment the coverage level of the area is evaluated when agents follow paths found as the first feasible solution of the problem. The chapter concludes with the third experiment in which given a finite number of cases we examine how the computational time to the optimal solution scales with respect to the parameters of the problem. Here we consider changes of one of the following parameters: 1) the team size 2) the optimization horizon and 3) the grid resolution while the other parameters of the problem stay the same.

## 3-1 Optimal Paths: Coverage Performance Evaluation

In this section the performance of the persistent coverage task is tested in terms of the coverage level provided in the area. Here the area $Q$ introduced in Chapter 2 is considered and decomposed into a $6 \times 6$ grid. A team of 4 agents is employed for the task with agents equipped with sensors/actuators able to cover all points inside a cell. Their initial poses (positions and headings) are shown in Figure 3-1 .

In this experiment the cell agents $t_w$ are given specifying the longest time interval between 2 subsequent visits at a cell. Let $\mathbf{t} \in \mathbb{R}^{n_w}$ be a vector with elements the ages $t_w \in I$ with $n_w$ the cardinality of $I$. Here a sample of vectors $\mathbf{t}_l$, $l \in \{1, \ldots, 20\}$ is considered with each $\mathbf{t}_l$ a vector with elements $t_w^l$, $w \in I$. As in Chapter 2 for each vector $\mathbf{t}_l$ a vector $\mathbf{d}_l \in (0,1)^{n_w}$ is

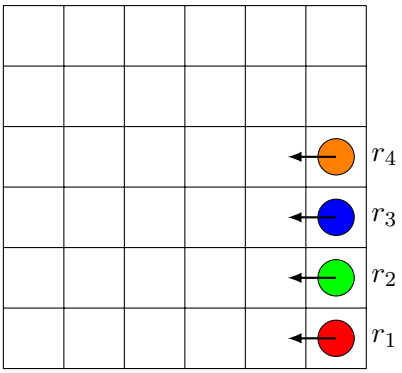



**Figure 3-1:** Initial poses of the agents

considered with elements $d_w^l \in (0,1)$ such that:

$$t_{w_1}^l \leq t_{w_2}^l \Leftrightarrow d_{w_1}^l \leq d_{w_2}^l, \quad w_1, w_2 \in I,\ l \in \{1,\ldots,20\}$$

and $\bar{\mathbf{d}}_\mathbf{l} \geq 0.85$ where $\bar{\mathbf{d}}_\mathbf{l}$ is the mean value of vector $\mathbf{d}_l$, $l \in \{1,\ldots,20\}$.

The desired level $\underline{Z}$ is arbitrarily chosen and set equal to 20 ($\underline{Z} = 20$). Moreover, the coverage level reset value $\bar{Z}$ is set equal to 300. This value has been found experimentally to be the minimum value guaranteeing feasibility of the optimization problem associated to each $\mathbf{t}_l$ while the following is always true:

$$t_w^l \geq \left\lfloor \frac{\ln \underline{Z} - \ln \bar{Z}}{\ln d_w^l} \right\rfloor, \quad w \in I, i \in \{1,\ldots,20\}$$

Here formulation II is used as proved to be the most efficient among the proposed. Then, paths are designed over an optimization horizon of $N = 10$ time steps.

In Figure 3-2 an example of the decay factors considered in a case scenario is presented. Cells with high decay rates are shown in light green and blue. Here the cells requiring more frequent coverage lie on the left side of the area and are assigned to decay factors ranging from 0.7 to 0.85. Therefore, we expect agents to move first towards these cells so as to maintain the desired lower level of coverage.

As shown in Figure 3-3 our intuition on the agents' moves matches the actual paths designed by the planner. Initially, agents $r_3, r_4$ that are the closest agents to cells with decay factors 0.7-0.8 start moving towards them while agent $r_2$ moves towards the upper right corner where two cells exist with decay rate of 0.85. By time step $k = 4$ cells with the highest decay rate ($d_w = 0.7$) get covered and agents $r_3, r_4$ continue moving linearly with respect to the x-axis so as to cover the rest "blue" cells. The upper, right corner gets covered a little later (by $k = 7$) and $r_2$ starts moving to the right side of the grid where most of the COIs are located. Meanwhile, agent $r_1$ having covered "less-important" cells on its way to the upper part of the area starts covering cells of high decay rate previously covered by $r_4$. The coverage level of these cells (with center coordinates (2,14) to (14,14) along x-axis) drops significantly fast. Therefore, they need to be visited at the earliest possible in order for the agents to ensure not only task feasibility but also a high amount of coverage in the area.

By the end of the planning horizon agents cover all COIs (cells whose decay rate is at most 0.85) at least once. Although the coverage level of the majority of these cells is sufficient at the end of the horizon there are some cells whose coverage level is nearly equal to $\underline{Z}$. These cells have been visited only once at the beginning of the horizon and not again since then. Due to their high decay rate their coverage level decreased significantly fast. However, as shown in Figure 3-3k all agents are far away with the closest agent requiring at least 5 time steps to reach them. For this reason, if it was to continue the persistent coverage task with initial coverage level values and agent poses as the ones shown in Figure 3-3k the corresponding optimization problem would become infeasible. This problem, the *feasibility problem of the persistent coverage task* is discussed in the next Chapter and infeasibility is avoided by introducing a terminal constraint set in the optimization problem.

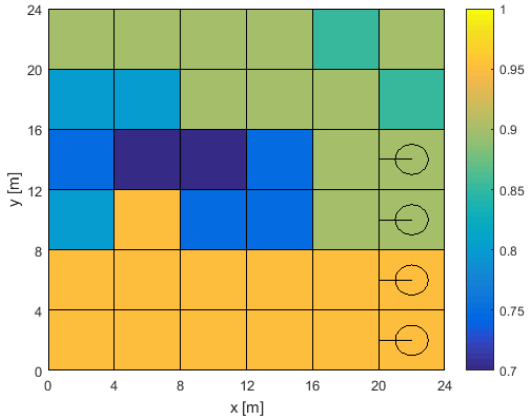



**Figure 3-2:** Coverage Decay Factors of a Case Scenario in a $6 \times 6$ Grid

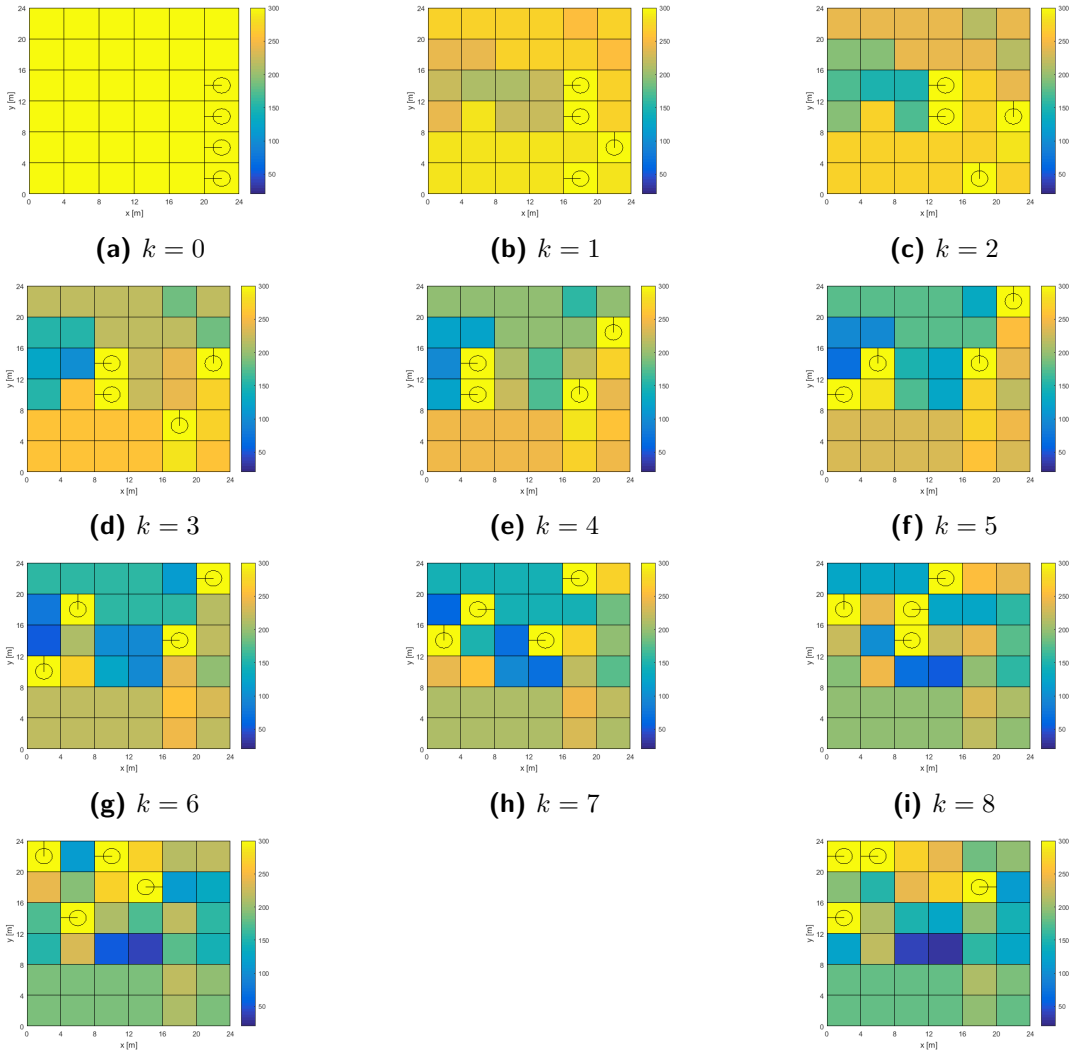
In Figure 3-4 the Normalized Mean Coverage Level (NMCL) per cell and time step is shown for the 20 cases considered at the beginning of the section. Here, the average Coverage Level of each cell is found to be between 64 and 71% of its maximum value $\bar{Z}$, thus significantly higher than $\underline{Z}$ (6.6% of $\bar{Z}$). Although the planning horizon is relatively short we observe a significant decrease on the Coverage Level compared to the initial value $\bar{Z}$. This is generally related to the number of COIs of each case and the corresponding decay factors. When the number of COIs is large agents need to move along the area aiming at covering them all. Depending on the horizon length, however, it is highly likely that some of these cells are visited only once or not at all (e.g when $d_w$ is close to 0.85). Even if they get covered, by the time agents leave them their coverage level starts dropping again sharply decreasing the total coverage level value. The decrease is even steeper when high decay rates are considered.

These conclusions are verified by the statistical data of Table 3-1. Here the Normalized Mean Coverage Level values are included in 4 bins, the same shown in Figure 3-4. Then, 2 basic elements are presented: 1) the number of COIs averaged over the number of cases associated to each bin (rounded on the first decimal) and 2) the corresponding coverage decay factors of the COIs averaged over the number of COIs of a case and the number of cases associated to each bin. Here the lowest NMCL values are related to the largest number of COIs (19 cells on average). In the majority of cases a large number of COIs (17 cells) is considered in conjunction with a high average decay rate (equal to 0.7762) resulting in a relatively low NMCL value on the interval [0.66,0.68]. The highest NMCL values as expected are found

when a small number of COIs is considered.

| NMCL | [0.64,0.66] | [0.66,0.68] | [0.68,0.7] | [0.7,0.72] |
|---|---|---|---|---|
| Average Number of COIs | 19 | 17 | 16 | 13 |
| Average decay factors of COIs | 0.7819 | 0.7762 | 0.7763 | 0.7769 |

**Table 3-1:** Relation of NMCL to the Number of COIs and the corresponding Decay Factors



**(a)** $k = 0$ **(b)** $k = 1$ **(c)** $k = 2$

**(d)** $k = 3$ **(e)** $k = 4$ **(f)** $k = 5$

**(g)** $k = 6$ **(h)** $k = 7$ **(i)** $k = 8$

**(j)** $k = 9$ **(k)** $k = 10$

**Figure 3-3:** Snapshots of the Coverage Level Map when a team of 4 agents is employed for an Optimization Horizon $N = 10$ steps

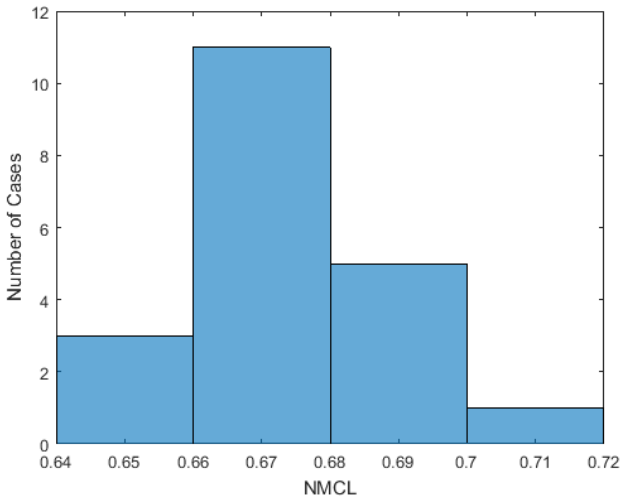



**Figure 3-4:** Normalized Mean Coverage Level over the Horizon and Number of Cells

## 3-2 Optimal and First Feasible Paths: A Comparison

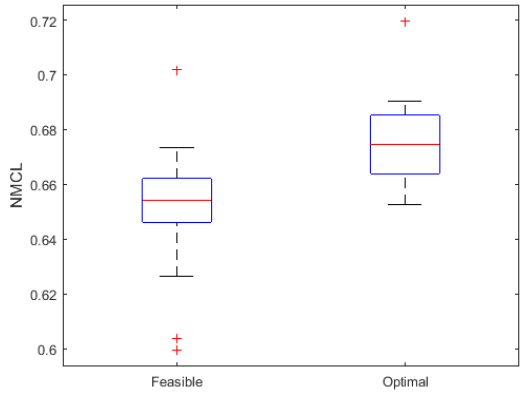
So far the performance of the persistent coverage task was evaluated in terms of the coverage level of the cells over a number of case scenarios. Agents' paths were found as a solution to the optimization problem (2-17). This formulation of the problem was chosen as found to be more efficient in terms of computational time when compared to formulation I. Despite the computational time improvement the time required for the problem to converge to the optimal solution is still found to be significantly high, thus prohibitive for online planning.

Addressing this problem, in this section we propose using the first feasible solution of the optimization problem (2-17). This choice is motivated by the need of minimizing the computational time of the problem for the agents to be able to perform the computations on-board at a minimum cost. In Figure 3-5 the quality of the first feasible and optimal paths is evaluated in terms of: 1) the normalized coverage level (NMCL) averaged over the number of cells and planning steps and 2) the computational time. The different combinations of the coverage decay factors considered here were introduced in Section 3-1. The desired level of coverage $\underline{Z}$ is set equal to 20 and the problem is solved over an optimization horizon of 10 time steps. The initial poses of the agents are shown in Figure 3-1.

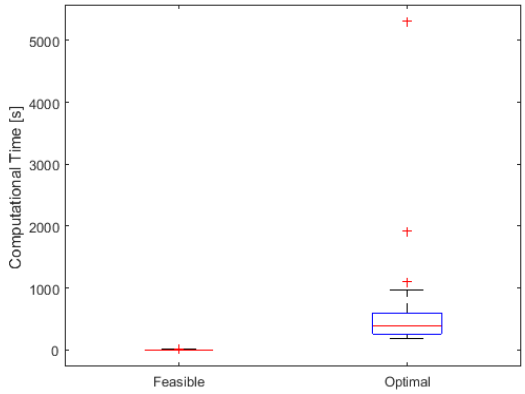
In this Section two different reset values are considered equal to 300 and 800. These values are chosen so as the age at the COI with the highest decay rate is always less than the optimization horizon length ($\min_{w\in I} t_w \leq N$, where $t_w$ is defined by (2-2)). When $\bar{Z} = 300$ this value is equal to 7 time steps. Therefore, agents need to visit the corresponding cell at least once over the horizon so as to maintain its coverage level above $\underline{Z}$. When $\bar{Z} = 800$ the age increases to 9 time steps providing agents more flexibility on the time at which they may visit the cell. In addition, the constraints forcing agents to maintain the coverage level of each cell above $\underline{Z}$ are redundant for the majority of the planning steps. This increases the computational time of the problem as the solver needs to check more combinations of paths in order to identify the one maximizing the total coverage level of the area. This is verified by Figures 3-5b and 3-5d. The time required for finding the optimal solution when $\bar{Z} = 800$

is on average equal to 2000 sec contrary to 500 sec when $\bar{Z} = 300$.

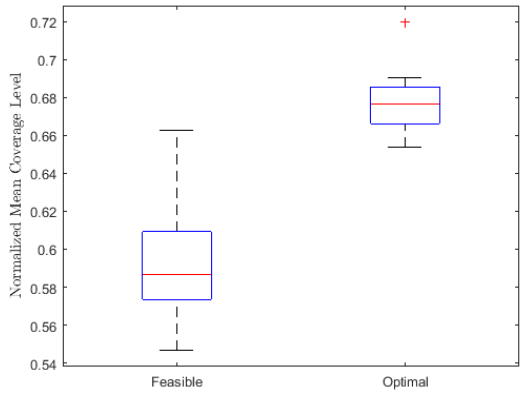
While the time to the optimal solution increases as the reset value increases, when the time for finding the first feasible solution is considered a different relation to the value of $\bar{Z}$ is observed. For $\bar{Z} = 300$ the solver needs 3 sec to find the first feasible solution contrary to 20 msec when $\bar{Z} = 800$. This time difference is related to the number of active constraints of the problem. When a high reset value is considered the constraints forcing a lower bound on the coverage level of every cell are almost always satisfied for the paths found as a solution to the problem without them. Hence, the solution spaces of the problems with and without the lower bound constraints are almost identical. However, as the reset value decreases the solution space of the problem with the lower bound constraints becomes smaller since some solutions of the problem without these constraints are excluded for not satisfying them. Therefore, the solver needs more time for checking candidate solutions before finding a feasible one.



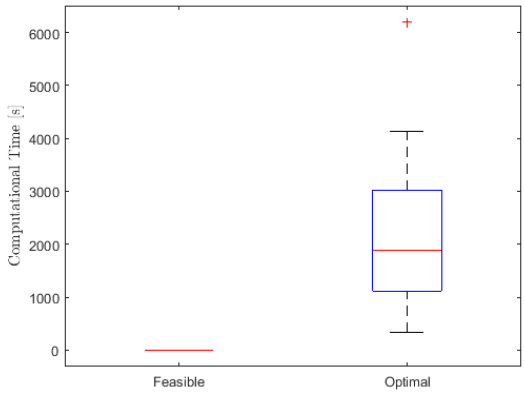

**(a)** NMCL per cell and time step for $\bar{Z} = 300$




**(b)** Computational Time for $\bar{Z} = 300$




**(c)** NMCL per cell and time step for $\bar{Z} = 800$




**(d)** Computational Time for $\bar{Z} = 800$

**Figure 3-5:** First Feasible versus Optimal Paths for Different Reset Values

Despite the increased computational time when $\bar{Z} = 300$ the quality of the first feasible paths in terms of coverage level is significantly higher compared to the case of $\bar{Z} = 800$. As shown in Figure 3-5a when $\bar{Z} = 300$ the average NMCL value associated to the first feasible solution is almost 2% less than the corresponding value of the optimal solution. This difference rises to

10% (Figure 3-5c) when the reset value is increased. Based on the above we can conclude that the profitability of using the first feasible paths becomes lower as the minimum $t_w$ approaches the optimization horizon length. Therefore, as will be discussed in Chapter 5 another solution of the problem may be considered that maintains a balance between the coverage level loss and computational time so as the persistent coverage task is always feasible over the simulation horizon.

## 3-3 Problem Scalability

In previous Sections the path planning problem was evaluated for different combinations of cell ages $t_w,\ w \in I$. There the number of agents and cells in addition to the planning horizon length were randomly chosen. In this Section we consider the same coverage decay factors introduced in Section 3-1 for each combination of $t_w,\ w \in I$ and study how the problem scales with the size of team, the length of the optimization horizon and the grid resolution.

### 3-3-1 Number of Agents

Initially, the scalability of the problem is examined with respect to the number of agents employed for the task. Here, we consider a team of 3,4 and 5 agents with initial poses shown in Figure 3-6. The $6 \times 6$ grid introduced in Section 3-1 is considered and the goal of the task is to design agents' paths so as the coverage level of each cell is maintained at least equal to $\underline{Z} = 20$. The paths are designed over a planning horizon of 10 steps with the reset value $\bar{Z}$ equal to 300.

Here 20 different combinations of coverage decay factors are considered. As the MILP problems often suffer from long computational times throughout this Section a time limit is set equal to 5 hours. Then, the optimization problem will terminate either when the optimal solution is found or when the time limit is reached. In the latter case, the best solution found is returned.

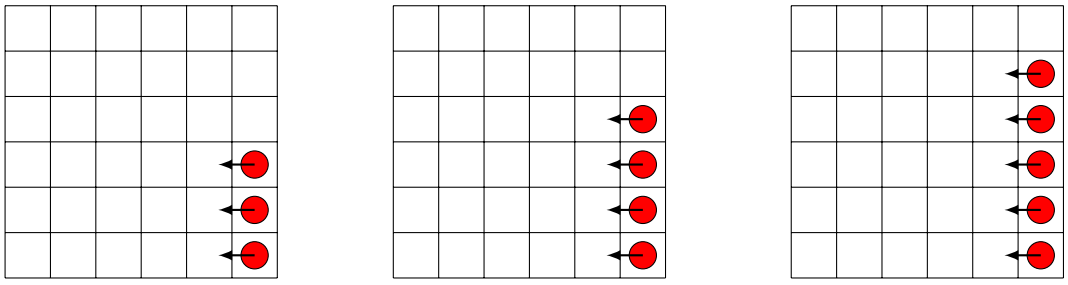


**Figure 3-6:** Initial Poses of the agents when a team of 3, 4 and 5 agents is considered.

In Figure 3-7 the computational time required for finding the optimal solution is presented for different team sizes. Here the optimal solution is found within the pre-defined time interval only for the 80% of the cases. In the rest of the cases the time limit is reached before the optimal paths are found when a team of 5 agents is employed for the task. As shown in the Figure the time to the optimal solution increases more than linear with the number of agents. When a small team of 3 agents is considered the optimal paths are found on average within 8

sec. As the team becomes bigger more cells are covered simultaneously per time step. Agents are able to visit more cells will low decay rates or may reach cells with high decay rate faster. Therefore, the number of feasible paths and their combinations increases resulting in higher computational time (300 and 4100 sec when $n_r = 4$ and $n_r = 5$ respectively).

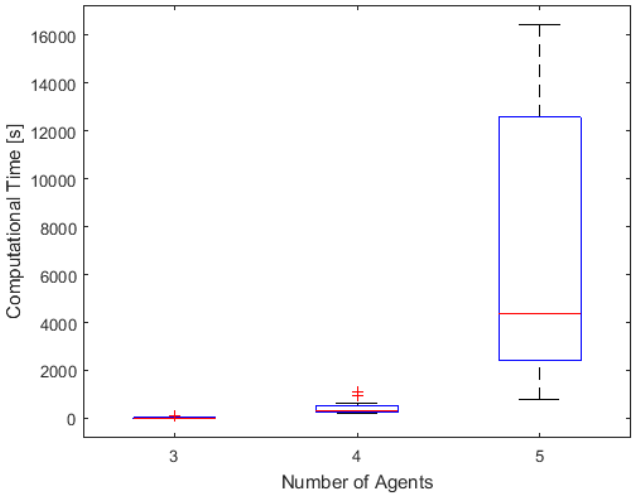



**Figure 3-7:** Computational Time to Convergence for Different Team Sizes

### 3-3-2 Optimization Horizon Length

In the second experiment we examine how the computational time increases with respect to the planning horizon $N$. Here a team of 4 agents is employed for the task with initial positions shown in Figure 3-1. The area is decomposed into a grid of $6 \times 6$ cells. As before the different decay factors introduced in Section 3-1 are considered and the reset value $\bar{Z}$ is set equal to 300. Moreover, the desired coverage level $\underline{Z}$ is set equal to 20.

The optimization problem (2-17) is solved for different planning horizon lengths equal to 7, 10 and 12 time steps. In this problem the number of binary variables defining the agents' poses is proportional to the planning horizon length. Therefore, we expect an increasing complexity when longer planning horizons are considered.

This is verified in Figure 3-8. Here, the time to the optimal solution is shown for the cases terminating within the time limit for any value of $N$ (50% of the total). When $N = 7$ the computational time is significantly low and equal to 7 sec. Within this time agents move towards the reachable COIs to minimize coverage loss. As the horizon length increases more COIs are becoming reachable by the agents. Therefore, the number of feasible paths and their combinations increases steeply forcing the solver to spend more time to identify the best possible solution (281 sec for $N = 10$ and 6100 sec for $N = 12$).

### 3-3-3 Grid Resolution

In the final experiment the computational complexity of the problem is examined with respect to the number of cells in the grid. Here, the area of interest is decomposed into a grid of

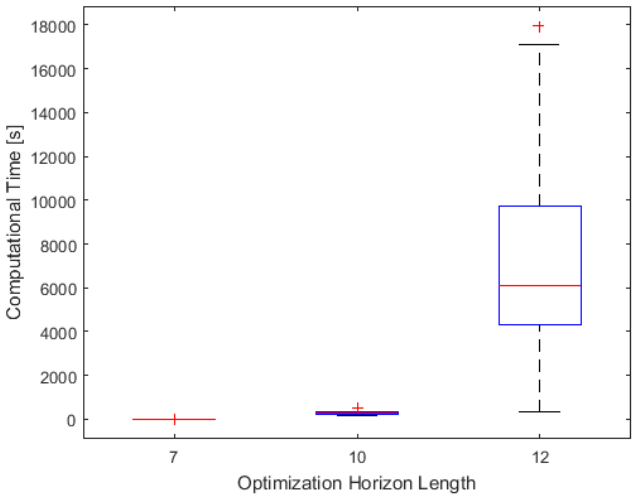



**Figure 3-8:** Computational Time to Convergence when Different Horizon Lengths are considered

$4 \times 4$, $6 \times 6$ and $8 \times 8$ cells. A team of 4 agents is employed for the task and their paths are designed over a planning horizon of 10 steps. For every cell in each newly introduced grid we consider every cell in the $6 \times 6$ grid with which their intersection is non-empty. Then, the decay factor of this cell is defined as the minimum value among the decay factors of the cells of the $6 \times 6$ grid found. An example is shown in Figure 3-9. Here the first cell of the $4 \times 4$ grid is considered and in light grey its intersection is shown with the cells of the $6 \times 6$ grid. Then, its decay factor is defined as the minimum decay factor of the cells in the $6 \times 6$ grid with index $w \in \{1, 2, 7, 8\}$ and $w$ defined as:

$$w = i_1 + 6\,(i_2 - 1), \quad i_1, i_2 \in \{1, \ldots, 6\}$$

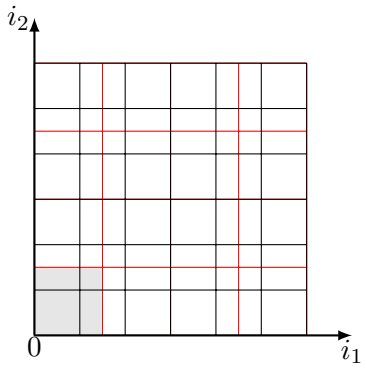



**Figure 3-9:** Example of how the decay factor $d_w$ is defined for a cell w in a $4 \times 4$ grid based on the decay factors of the $6 \times 6$ grid

In this experiment the agents are initially placed on the right side of the area along the last column of the grid. The desired level of coverage is set equal to $\underline{Z} = 20$. Furthermore, the reset value is set equal to $\bar{Z} = 800$. This value is chosen as the minimum value for which

every optimization problem is solved within the time limit irrespective the choice of the grid resolution.

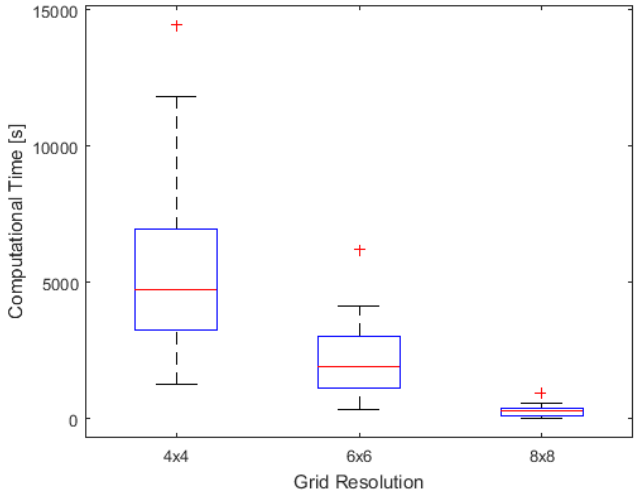



**Figure 3-10:** Computational time spent until the Optimal Solution is found when different grid resolutions are considered

In Figure 3-10 the computational time to the optimal solution is shown when a different number of cells is considered. Here, the problem converges to the optimal solution faster when the number of cells is larger(290 sec on average when $n_w = 64$ contrary to 4800 when $n_w = 16$). When a high resolution grid is considered and due to the definition of the decay factors the number of COIs increases. The coverage level of these cells decreases faster than the others'. Therefore, agents' priority is to visit these cells as soon as possible. When a coarse resolution ($4 \times 4$) is considered agents are able to visit all cells within a small amount of time steps (e.g 3 steps if agents move linearly towards the opposite side of the area). This allows the agents to cover the area multiple times within the horizon and the question is how agents should move across the area so as the total coverage level is maximized. Since no other restriction related to the coverage level is considered rather than the lower bound constraint, the solver is not able to early discard non-profitable solutions. Based on that we can conclude that a coarser resolution may result in more frequent visits at the cells in the grid, however, at the expense of high computational time. Therefore, a grid resolution should be chosen so as a balance between the number of cells visited and the computational effort is generally maintained.

## 3-4 Conclusions

In this Chapter the optimization problem (2-17) is considered and evaluated in terms of coverage performance and computational effort. A $6 \times 6$ grid is considered and the efficiency of the problem is examined for a sample of 20 different combinations of decay factors. Initially, 4 agents are employed for the task and the coverage performance is evaluated when agents follow the optimal paths. As shown agents maintain the average coverage level of the cells over 64% of the maximum value, thus significantly higher to $\underline{Z}$ (6.6% of $\bar{Z}$). However, this is

achieved at the cost of computational effort. Aiming at finding a solution that maintains a balance between coverage performance and computational time, we consider the first feasible solution of the problem and evaluate its coverage quality for different combinations of cell ages $t_w, w \in I$. As the minimum $t_w$ increases the first feasible paths are found at negligible time but their quality in terms of coverage level is considerably less than the corresponding optimal(10% on average). Finally, the scalability of the problem is studied with respect to the number of agents, the planning horizon and the number of cells. As expected the computational complexity of the problem increases as the number of agents and planning steps rises. However, when a high-resolution grid is considered the time to the optimal solution drops significantly since agents move towards covering the increased (compared to the $6 \times 6$ grid) number of COIs in the area.

# Chapter 4

# Persistent Coverage Task Feasibility

So far we studied the persistent coverage problem over a finite optimization horizon $N$. In this problem the resulting paths were found as a solution to an optimization problem that maximizes coverage for a finite number of steps without, however, taking into consideration agents' performance after $N$. Therefore, if we were to solve the optimization problem with initial conditions the coverage level values and agents' poses at time step $N$ the new problem could be easily found to be infeasible. In this Chapter the feasibility of the persistent coverage task is studied and the *recursive feasibility* property of the optimization problem designed in Chapter 2 is proven when a terminal constraint is introduced to the optimization problem.

Recursive feasibility has been extensively studied in literature for the design of Model Predictive controllers (MPC) [48],[49]. There, given the system dynamics an optimization problem is designed that takes into consideration input and state constraints of the system. The solution of that problem is a sequence of control inputs. From those only the first control input is applied to the system and the optimization problem is solved again with initial condition the new state of the system. The feasibility of the problem is guaranteed by constraining the terminal state of the system to be inside a pre-defined terminal set. This set is designed to be *Control Invariant*. Thus, given a terminal state there always exists an admissible input steering the system to another terminal state. Then, given the optimal control sequence of the problem solved at absolute time step $i$ there always exists a control sequence for $i+1$ satisfying the constraints of the problem. This control sequence is the shifted by one step optimal sequence of step $i$ augmented in the end by the input steering the terminal state at $i$ to another state in the terminal set. Since $i$ is arbitrary this holds for any time step. Therefore, the optimization problem is considered *recursively feasible*.

Drawing inspiration from MPC problems and especially from the work in [50], in this Chapter we prove recursive feasibility of the problem when a time-variant terminal constraint set is considered. Each set includes a proposed pose for each agent and the coverage level of the cells as a result of the agents' actions. Based on this set a small number of constraints is added to the MILP problem so as the agents' poses are identical to the poses of the set and the coverage level of each cell is lower bounded by the corresponding value of the terminal set. A finite terminal constraint set sequence is designed such that the union of the sets defines a

set of closed paths that when repeatedly followed by the agents the coverage level of the area is bounded from below by $\underline{Z}$. Therefore, given the terminal constraint set $X$ and the optimal solution of the problem at absolute time step $i$, there always exists a path for each agent such that the problem at time step $i+1$ with terminal set the one following $X$ in the sequence (or the first set if the terminal set at $i$ is the last in the sequence) is always feasible.

A detailed proof of the problem's feasibility is presented in Section 4-1. Then, in Section 4-2 a two-step method is proposed for designing the set sequence. The Chapter ends with Section 4-3 summarizing the results of the chapter.

## 4-1 Recursive Feasibility

In this Section the terminal constraint set sequence is formally introduced and the recursive feasibility property of the optimization problem of Chapter 2 is proved when the final poses of the agents and coverage level of the cells become constrained.

To simplify notation we will first introduce a general representation of the optimization problem. Let $\mathbf{z}_i(k) \in [\underline{Z}, \bar{Z}]^{n_w}$ be a vector including the coverage level of every cell $w \in I = \{1, \dots, n_w\}$ at time step $k$. Here $i$ is the absolute time index while $k$ takes values in $\{1, \dots, N\}$ with $N$ the planning horizon of the problem. As discussed in Section 2-1-3 each element of $\mathbf{z}_i(k)$ is monotonically decreasing when the corresponding cell is not covered and resets to $\bar{Z}$ when an agent visits it. Thus, $\bar{Z}$ is its maximum value. Moreover, due to the task objective the coverage level of each cell should be lower bounded by $\underline{Z}$. Therefore, the coverage level of each cell takes values in $[\underline{Z}, \bar{Z}]$.

The evolution of $\mathbf{z}_i(k)$ over time is expressed by a function $F$: $[\underline{Z}, \bar{Z}]^{n_w} \times U \to [\underline{Z}, \bar{Z}]^{n_w}$ as:

$$\mathbf{z}_i(k) = F\big(\mathbf{z}_i(k-1), \mathbf{u}_i(k)\big)$$

where $[\underline{Z}, \bar{Z}]^{n_w}$ stands for the product $[\underline{Z}, \bar{Z}] \times \ldots \times [\underline{Z}, \bar{Z}]$ and $\mathbf{u}_i(k) \in U$ is a vector containing the pose of each agent at time step $k$ expressed as a node in the graph $G$. Here $U$ is a subset of $V^{n_r}$ with $n_r$ the number of agents containing all vectors $\mathbf{u}_i(k) \in V^{n_r}$ such that:

1. Each agent has a unique pose in $V$
2. Each cell hosts at most one agent per $k$

The function $F : [\underline{Z}, \bar{Z}]^{n_w} \times U \to [\underline{Z}, \bar{Z}]^{n_w}$ is defined as:

$$F\big(\mathbf{z}_i(k-1), \mathbf{u}_i(k)\big) = \begin{bmatrix} d_1(1-\sigma_1(\mathbf{u}_i(k)) & & \mathbf{0} \\ & \ddots & \\ \mathbf{0} & & d_{n_w}(1-\sigma_{n_w}(\mathbf{u}_i(k)) \end{bmatrix} \mathbf{z}_i(k-1) + \bar{Z} \begin{bmatrix} \sigma_1(\mathbf{u}_i(k)) \\ \vdots \\ \sigma_{n_w}(\mathbf{u}_i(k)) \end{bmatrix}$$

where $d_w \in (0,1)$ is the coverage decay factor of cell $w \in I$, $\bar{Z}$ the reset value of the coverage level and $\sigma_w : U \to \{0,1\}$ a function defined for every cell $w$ as follows:

$$\sigma_w(\mathbf{u}_i(k)) = \begin{cases} 1, & \exists\, r \in K : \left\lceil \frac{\mathbf{u}_i^r(k)}{4} \right\rceil = w \\ 0, & \text{otherwise} \end{cases}$$

where $\mathbf{u}_i^r(k)$ is the r$^{\text{th}}$ element of vector $\mathbf{u}_i(k)$, thus the pose of agent $r$ at time step $k$ and $\left\lceil \frac{\mathbf{u}_i^r(k)}{4} \right\rceil$ is the index of the cell agent $r$ is placed at (as defined in (2-5)).

Let $H\big(\mathbf{u}_i(k-1), \mathbf{u}_i(k)\big)$ be a function counting the number of agents covering cells at time step $k$ that were covered by other peers at time step $k-1$. Then, the problem can be written in a compact way as follows:

$$\max_{\mathbf{u}_i \in U^N} \mathbf{1}^T \begin{bmatrix} \mathbf{z}_i(1) \\ \vdots \\ \mathbf{z}_i(N) \end{bmatrix} - \beta\, \mathbf{1}^T \begin{bmatrix} h_i(1) \\ \vdots \\ h_i(N) \end{bmatrix} \tag{4-1}$$

subject to:

$$F\big(\mathbf{z}_i(k-1), \mathbf{u}_i(k)\big) = \mathbf{z}_i(k)$$
$$k \in \{1, \dots, N\} \tag{4-1a}$$
$$H\big(\mathbf{u}_i(k-1), \mathbf{u}_i(k)\big) = h_i(k)$$
$$k \in \{1, \dots, N\} \tag{4-1b}$$
$$\Big(\mathbf{u}_i(k-1), \mathbf{u}_i(k)\Big) \in E^{n_r}$$
$$k \in \{1, \dots, N\} \tag{4-1c}$$
$$\begin{bmatrix} \mathbf{z}_i(N) \\ \mathbf{u}_i(N) \end{bmatrix} \in X_i^f \tag{4-1d}$$
$$\mathbf{z}_i(0) = \mathbf{z}_i^0 \tag{4-1e}$$
$$\mathbf{u}_i(0) = \mathbf{u}_i^0 \tag{4-1f}$$
$$\mathbf{z}_i(k) \in [\underline{Z}, \bar{Z}]^{n_w}$$
$$k \in \{1, \dots, N\} \tag{4-1g}$$
$$\mathbf{u}_i(k) \in U$$
$$k \in \{1, \dots, N_p\} \tag{4-1h}$$

Here $\mathbf{u}_i = \begin{bmatrix} \mathbf{u}_i(1) & \dots & \mathbf{u}_i(N) \end{bmatrix}^T$ is the variable vector containing the agents' poses over the horizon and $\beta > 0$ is a positive weight expressing the importance of avoiding coverage at subsequent time steps. Equation (4-1a) defines the coverage level dynamics and (4-1b) the number of cells being covered at consecutive time steps by different agents. Equation (4-1c) guarantees that for every agent the transition from the current to the future pose is admissible over the graph $G$. Equation (2-17d) forces the final coverage level values and poses of the agents to be inside a terminal constraint set $X_i^f \subset [\underline{Z}, \bar{Z}]^{n_w} \times U$ while (2-17e)-(2-17f) define the initial conditions of the problem. Finally, (2-17g)-(2-17h) guarantee the coverage level values and poses of the agents to be inside the allowable sets.

For $M \in \mathbb{N},\ M > 1$ consider a sequence of sets $\{S_0, \dots, S_{M-1}\}$. Here $M$ could be considered as the number of steps required for the agents to cover every cell in the area and maintain the coverage level of each cell above $\underline{Z}$ before they return back to their initial pose. Each set $S_u$ could be defined as:

$$S_v = \left\{ \begin{bmatrix} \mathbf{z} \\ \mathbf{u} \end{bmatrix} \in [\underline{Z}, \bar{Z}]^{n_w} \times U :\ \mathbf{z} \succeq \mathbf{z}_v,\ \mathbf{u} = \mathbf{u}_v \right\}, \qquad v = 0, \dots M-1 \tag{4-2}$$

with $\mathbf{z}_u$ a vector including the coverage level of the cells at time step $v$ when agents' poses are defined as elements of the vector $\mathbf{u}_v$. Vectors $\mathbf{z}_v, \mathbf{u}_v$ are chosen such that the following always hold:

$$\begin{cases} (\mathbf{u}_v, \mathbf{u}_{v+1}) \in E^{n_r} \\ \mathbf{z}_{u+1} = F(\mathbf{z}_v, \mathbf{u}_{u+1}) \end{cases} \qquad \forall v \in \{0, \dots, M-2\} \tag{4-3}$$

$$\begin{cases} (\mathbf{u}_{M-1}, \mathbf{u}_0) \in E^{n_r} \\ \mathbf{z}_0 = F(\mathbf{z}_{M-1}, \mathbf{u}_0) \end{cases} \tag{4-4}$$

These constraints guarantee that the union of the sets in the sequence $\{S_v\}_{v=0}^{M}$ describes a set of closed paths and the evolution of the coverage level of the cells when agents follow these paths. In (4-3) the first constraint guarantees that for every agent the transition from the current to the future pose is admissible over the graph $G$ and the second that the coverage level of each cell takes values as defined in (2-1). Constraints of (4-3) are similar to the ones introduced in (4-1) for the optimization problem we discussed so far. The difference on the terminal paths lies in the extra constraints of (4-4). These constraints guarantee that the paths are closed and that the coverage level of each cell satisfies (2-1). Therefore, if agents follow these paths repeatedly the area would be completely covered and the coverage level will always take values in the interval $[\underline{Z}, \bar{Z}]$. Based on that we can prove the following:

**Lemma 1.** *Let the vectors* $\begin{bmatrix} \mathbf{z}_a \\ \mathbf{u}_a \end{bmatrix} \in S_v,\ v \in \{0, \dots, M-2\},\ \begin{bmatrix} \mathbf{z}_b \\ \mathbf{u}_b \end{bmatrix} \in S_{M-1}$ *and* $\begin{bmatrix} \mathbf{z}_c \\ \mathbf{u}_c \end{bmatrix},\ \begin{bmatrix} \mathbf{z}_d \\ \mathbf{u}_d \end{bmatrix} \in [\underline{Z}, \bar{Z}]^{n_w} \times U$ *be defined such that :* $\mathbf{u}_c = \mathbf{u}_{v+1},\ \mathbf{u}_d = \mathbf{u}_0$. *If the following are true:*

$$\mathbf{z}_c = F(\mathbf{z}_a, \mathbf{u}_c) \tag{4-5}$$

$$\mathbf{z}_d = F(\mathbf{z}_b, \mathbf{u}_d) \tag{4-6}$$

then $\begin{bmatrix} \mathbf{z}_c \\ \mathbf{u}_c \end{bmatrix} \in S_{v+1}$ and $\begin{bmatrix} \mathbf{z}_d \\ \mathbf{u}_d \end{bmatrix} \in S_0$.

*Proof.* From the definition of the terminal sets of equation (4-2) $\begin{bmatrix} \mathbf{z}_c \\ \mathbf{u}_c \end{bmatrix} \in S_{v+1}$ and $\begin{bmatrix} \mathbf{z}_d \\ \mathbf{u}_d \end{bmatrix} \in S_0$ are true if and only if: $\mathbf{z}_c \succeq \mathbf{z}_{v+1}$ and $\mathbf{z}_d \succeq \mathbf{z}_0$. Let us first prove that $\mathbf{z}_c \succeq \mathbf{z}_{v+1}$.

When agents' poses are defined by the vector $\mathbf{u}_{v+1}$ $n_r$ cells are covered. The index of these cells can be found using (2-5) by replacing $q \in V$ with the elements of $\mathbf{u}_{v+1}$. Since these cells are covered their coverage level resets to $\bar{Z}$ due to (4-5). Therefore, due to the equality

$u_c = u_{v+1}$ the elements of the vectors $\mathbf{z}_c,\ \mathbf{z}_{v+1}$ corresponding to the coverage level values of these cells are identical and equal to $\bar{Z}$.

Let $\mathbf{z}'_c,\ \mathbf{z}'_a,\ \mathbf{z}'_v\ \mathbf{z}'_{v+1} \in [\underline{Z}, \bar{Z}]^{n_w - n_r}$ be the vectors with elements the coverage level values of the cells that are not covered by the agents when the poses of the latter are found in $\mathbf{u}_{v+1}$. In these vectors the coverage level values of the cells are placed in ascending order with respect to the cell index. Then, from (4-5) we have that:

$$\mathbf{z}'_c = D\ \mathbf{z}'_a \tag{4-7}$$

where $D$ is a positive, diagonal matrix containing the coverage decay factors of the uncovered cells placed in the matrix with the same order as the coverage level values of $\mathbf{z}'_a$.

Since $\begin{bmatrix}\mathbf{z}_a \\ \mathbf{u}_a\end{bmatrix} \in S_v$ we have that: $\mathbf{z}_a \succeq \mathbf{z}_v$ and also $\mathbf{z}'_a \succeq \mathbf{z}'_v$. Multiplying both sides with $D$ and based on (4-7) we have that:

$$\mathbf{z}'_c = D\ \mathbf{z}'_a \succeq D\ \mathbf{z}'_v \tag{4-8}$$

From (4-3) we have that: $\mathbf{z}'_{v+1} = D\ \mathbf{z}'_v$ thus concluding that: $\mathbf{z}'_c \succeq \mathbf{z}'_{v+1}$. Moreover, as discussed above the elements of the vectors $\mathbf{z}_c,\ \mathbf{z}_{v+1}$ corresponding to the coverage level of the cells being covered by the agents are identical. Hence, we can conclude that $\mathbf{z}_c \succeq \mathbf{z}_{v+1}$.

In a similar manner we can prove that $\mathbf{z}_d \succeq \mathbf{z}_0$. □

Initially, the terminal constraint $X_0^f$ could be chosen as one of the sets of the sequence as long as the optimization problem is feasible for the initial agents' poses and coverage level of the cells. Suppose $X_0^f = S_v$. It follows that $X_1^f = S_{v+1},\ X_2^f = S_{v+2},\ \ldots,\ X_{M-1-v}^f = S_{M-1},\ X_{M-v}^f = S_0, \ldots,\ X_{M-1}^f = S_v$. Then, for any $i \in \mathbb{N}$ we can obtain the following rule [50]:

$$X_0^f = S_v \Longrightarrow X_i^f := S_{(v+i) \bmod M} \tag{4-9}$$

Based on the above we are now in position to prove the recursive feasibility of the problem.

**Theorem 1.** *Suppose problem (4-1) is feasible at time instant $i$ with initial coverage level values $\mathbf{z}_i(0)$, initial agents' poses $\mathbf{u}_i(0)$ and terminal set $X_i^f$ as defined in (4-9). Suppose $\mathbf{u}_i^* \in U^N$ is the optimal input sequence of (4-1) at time instant $i$. Then, the problem will be feasible at time instant $i+1$ with initial agents' poses defined by the vector $\mathbf{u}_i^*(1)$ and initial coverage level values $\mathbf{z}_{i+1}(0) = F(\mathbf{z}_i(0), \mathbf{u}_i^*(1))$.*

*Proof.* For any $i \in \mathbb{N}$ there exists an index $p \in \{0, \ldots, M-1\}$ such that: $\begin{bmatrix}\mathbf{z}_i^*(N) \\ \mathbf{u}_i^*(N)\end{bmatrix} \in X_i^f = S_{(v+i) \bmod M} = S_p$ where $\mathbf{u}_i^*(N),\ \mathbf{z}_i^*(N)$ are the vectors including the optimal poses of the agents and the optimal coverage levels of the cells at the end of the horizon respectively. Then, we will have that:

$$X_{i+1}^f = \begin{cases} S_{p+1}, & p < M-1 \\ S_0, & p = M-1 \end{cases} \tag{4-10}$$

In order for the problem to be feasible at $i+1$ an input sequence $\mathbf{u}_{i+1} \in U^N$ needs to be introduced so as $\begin{bmatrix}\mathbf{z}_{i+1}(N) \\ \mathbf{u}_{i+1}(N)\end{bmatrix} \in X_{i+1}^f$.

Consider the input sequence $\mathbf{u}_{i+1} = \begin{bmatrix}\mathbf{u}_i^*(2) & \dots & \mathbf{u}_i^*(N) & \mathbf{u}\end{bmatrix}^T$ with:

$$\mathbf{u} = \begin{cases}\mathbf{u}_{p+1}, & p < M-1\\ \mathbf{u}_0, & p = M-1\end{cases}$$

Due to (4-1a) we have that: $\mathbf{z}_{i+1}(N) = F(\mathbf{z}_i^*(N), \mathbf{u})$. Then, from Lemma 1 it holds that $\begin{bmatrix}\mathbf{z}_{i+1}(N)\\ \mathbf{u}\end{bmatrix} \in X_{i+1}^f$ which completes the proof. □

## 4-2 Constructing the Terminal Constraint Set Sequence

Having proved the recursive feasibility of the problem when a terminal set of the form (4-2) is added to the problem, in this Section we propose a 2-step method for constructing the terminal set sequence. In the first step a set of closed paths is designed aiming at maximizing the coverage level of the cells over the horizon while minimizing the number of cells covered by different agents at consecutive time steps. In this optimization problem the goal is 1) to have every cell in the grid covered and 2) to maintain the coverage level of the cells above $\underline{Z}$.

As the paths designed are closed, at the end of the horizon agents return back to their initial cells and heading. Nevertheless, at that point the coverage level of the cells might not be enough for the persistent coverage task to be feasible when agents start following the designed paths for a second time. Thus, there might be a time step for which the coverage level of a cell drops below $\underline{Z}$ since no agent was close enough for a visit. To resolve this problem a second step is considered. In this step a linear program (LP) is designed for finding the minimum reset value $\bar{Z}$ for which the coverage level of the cells is always bounded from below by $\underline{Z}$ as agents follow the closed paths of step 1 repeatedly.

The aforementioned paths are found as a solution to an optimization problem. This problem is defined in a similar manner as problems (2-9), (2-17) with an extra set of constraints. These constraints guarantee that each cell in the area gets covered at least once and the agents' final poses are identical to the agents' initial ones. In this Section we will build upon formulation I described in (2-9) as it is straightforward to impose the final step constraints. These constraints are formulated as:

$$\delta_{q_r}^r(N) = 1, \quad r \in K$$

where $q_r \in V$ is the initial pose of agent $r$ expressed as a node of graph $G$.

In addition to this constraint we force agents to cover every cell in the area at least once. This can be expressed as:

$$\sum_{k=1}^{N} \sum_{q=4(w-1)+1}^{4w} \sum_{r\in K} \delta_q^r(k) \geq 1, \quad w \in I$$

Based on the above the optimization problem of step 1 could be formulated as follows:

$$\max_x J(x) \tag{4-11}$$

subject to:

$$\sum_{q\in V} \delta_q^r(k) = 1$$
$$\forall k \in \{1,\dots,N\},\ \forall r \in K \tag{4-11a}$$

$$\delta_q^r(k) - \sum_{q'\in V} a_{qq'}\ \delta_q^r(k+1) \le 0,$$
$$\forall k \in \{1,\dots,N-1\},\ \forall q' \in V,\ \forall r \in K \tag{4-11b}$$

$$\delta_{q_r}^r(0) - \sum_{q\in V} a_{q_r q}\ \delta_q^r(1) = 0$$
$$\forall r \in K \tag{4-11c}$$

$$\sum_{q=4\,(w-1)+1}^{4w} \sum_{r\in K} \delta_q^r(k) \le 1$$
$$\forall w \in I,\ \forall k \in \{1,\dots,N\} \tag{4-11d}$$

$$\alpha_w(k) - \bar{Z} \sum_{q=4\,(w-1)+1}^{4w} \sum_{r\in K} \delta_q^r(k) \le 0$$
$$\forall w \in I,\ \forall k \in \{1,\dots,N\} \tag{4-11e}$$

$$\alpha_w(k) \ge 0$$
$$\forall w \in I,\ \forall k \in \{1,\dots,N\} \tag{4-11f}$$

$$\alpha_w(k) - Z(w,k-1) \le 0$$
$$\forall w \in I,\ \forall k \in \{1,\dots,N\} \tag{4-11g}$$

$$\alpha_w(k) - Z_w(k-1) + \bar{Z}\left(1 - \sum_{q=4\,(w-1)+1}^{4w} \sum_{r\in K} \delta_q^r(k)\right) \ge 0$$
$$\forall w \in I,\ \forall k \in \{1,\dots,N\} \tag{4-11h}$$

$$d_w^k\ \bar{Z} + \sum_{t=1}^{k} d_w^{k-t}\left(-d_w\ \alpha_w(t) + \bar{Z} \sum_{q=4\,(w-1)+1}^{4w} \sum_{r\in K} \delta_q^r(t)\right) = Z(w,k)$$
$$\forall w \in I,\ \forall k \in \{1,\dots,N\} \tag{4-11i}$$

$$Z(w,k) \ge \underline{Z}$$
$$\forall w \in I,\ \forall k \in \{1,\dots,N\} \tag{4-11j}$$

$$\mu_w^r(k) - \sum_{q=4(w-1)+1}^{4w} \delta_q^r(k) \leq 0$$
$$\forall r \in K,\ \forall w \in I,\ \forall k \in \{1, \dots, N-1\} \quad \text{(4-11k)}$$

$$\mu_w^r(k) - \sum_{\substack{r' \in K \\ r \neq r'}} \sum_{q'=4(w-1)+1}^{4w} \delta_{q'}^{r'}(k+1) \leq 0$$
$$\forall r \in K,\ \forall w \in I,\ \forall k \in \{1, \dots, N-1\} \quad \text{(4-11l)}$$

$$\sum_{q=4(w-1)+1}^{4w} \delta_q^r(k) + \sum_{\substack{r' \in K \\ r \neq r'}} \sum_{q'=4(w-1)+1}^{4w} \delta_{q'}^{r'}(k+1) - \mu_w^r(k) \leq 1$$
$$\forall r \in K,\ \forall w \in I,\ \forall k \in \{1, \dots, N-1\} \quad \text{(4-11m)}$$

$$\sum_{k=1}^{N} \sum_{q=4(w-1)+1}^{4w} \sum_{r \in K} \delta_q^r(k) \geq 1$$
$$w \in I \quad \text{(4-11n)}$$

$$\delta_{q_r}^r(0) = 1$$
$$r \in K \quad \text{(4-11o)}$$

$$Z(w,0) = \bar{Z}$$
$$w \in I \quad \text{(4-11p)}$$

$$\delta_{q_r}^r(N) = 1$$
$$r \in K \quad \text{(4-11q)}$$

$$\delta_q^r(k) \in \{0,1\}$$
$$\forall q \in V,\ \forall r \in K,\ \forall k \in \{1, \dots, N\} \quad \text{(4-11r)}$$

$$\alpha_w(k) \in [0, \bar{Z}]$$
$$\forall w \in I,\ \forall k \in \{1, \dots, N\} \quad \text{(4-11s)}$$

$$\mu_w^r(k) \in \{0,1\}$$
$$\forall r \in K,\ \forall w \in I,\ \forall k \in \{1, \dots, N-1\} \quad \text{(4-11t)}$$

where $J(x)$ is given by (2-8).

The above problem guarantees $Z(w,k) \geq \bar{Z}$ for all cells over the horizon. However, the constraint is not satisfied when agents are back at their initial state and are about to repeat the same path. To resolve this issue a linear program is formulated aiming at finding the minimum reset value for which the coverage level of each cell is always bounded from below by $\underline{Z}$ when the same paths are repeatedly followed by the agents. Let this value be denoted by $\bar{Z}^*$.

Moreover, let $a_w^{\min},\ a_w^{\max} \in [1,N]$ denote the first and last time respectively at which $w$ is visited when agents traverse the paths designed in step 1 for the first time. Assume that agents have followed the path once and after $N$ they get back their initial poses. Then, the coverage level of every cell $w$ is equal to $Z(w,N) = d_w^{(N-a_w^{\max})}\ \bar{Z}$. Following the same path for a second time, an agent will visit the cell $w$ after $a_w^{\min}$ time instants. Thus, in order for the coverage level of every cell to be at least $\underline{Z}$ it should hold that: $N - a_w^{\max} + a_w^{\min} - 1 \leq t_w$, with $t_w$ defined in (2-2). In addition to that, the coverage level of $w$ should also be lower bounded by $\underline{Z}$ at the interval $[a_w^{\min}, N],\ w \in I$.

Let $\mathbf{z}_1(N)$ be the vector including the coverage level values $Z(w,N),\ w \in I$ when agents complete a full circle of the paths for the first time and $\mathbf{z}_2(0)$ the initial coverage level of the cells when agents are about to start following the paths for a second time. Then, we have that: $\mathbf{z}_1(N) = \mathbf{z}_2(0)$ and we require $\mathbf{z}_2(0) \in [\underline{Z}, \bar{Z}]^{n_w}$. The lower bound on the coverage level should also hold for any $k$ over the horizon. Hence, we have that $\mathbf{z}_2(k) \in [\underline{Z}, \bar{Z}]^{n_w}$ with $\mathbf{z}_2(k)$ a vector with elements the coverage level of the cells at time step $k$. As in problem (4-1) the evolution of $\mathbf{z}_2(k)$ is defined by (4-1a). Here, the agents' poses are known from step 1. We denote with $\mathbf{u}^s(k)$ the agents' poses found as a solution to (4-11). Then, the lower bound of the coverage levels in maintained over the horizon when the reset value is found as a solution to the following linear program(LP):

$$\min \bar{Z} \tag{4-12}$$

subject to:

$$\begin{aligned} &\mathbf{z}_2(k) = F(\mathbf{z}_2(k-1), \mathbf{u}_2(k)) \\ &\quad k \in \{1, \dots, N\} \end{aligned} \tag{4-12a}$$

$$\begin{aligned} &\mathbf{u}_2(k) = \mathbf{u}^s(k) \\ &\quad k \in \{1, \dots, N\} \end{aligned} \tag{4-12b}$$

$$\mathbf{z}_2(0) = \mathbf{z}_1(N) \tag{4-12c}$$

$$\mathbf{u}_2(0) = \mathbf{u}^s(N) \tag{4-12d}$$

$$\begin{aligned} &\mathbf{z}_2(k) \in [\underline{Z}, \bar{Z}]^{n_w} \\ &\quad k \in \{0, \dots, N\} \end{aligned} \tag{4-12e}$$

$$\bar{Z} > 0 \tag{4-12f}$$

Here (4-12a) expresses the dynamics of the coverage level of the cells when the agents' poses are known from step 1 (4-12b). Constraints (4-12c), (4-12d) describe the initial conditions of the problem. As discussed above the initial coverage levels are identical to the ones of the final step when agents complete for the first time a full circle of their paths and return to their initial poses. Finally, (4-12e), (4-12f) constrain the coverage level of the cells and reset value to be among the admissible values.

The reset value of (4-12) guarantees the coverage level to be always lower bounded by $\underline{Z}$. Therefore, the persistent coverage task will always be feasible when agents follow the closed paths repeatedly.

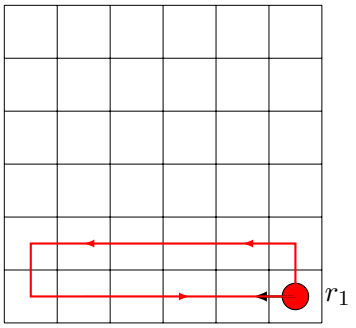
In Figure 4-1 an example of the closed paths is presented for the case scenario of Section 3-1. In this scenario a $6 \times 6$ grid is considered and a team of 4 agents is employed for the task. The initial poses of the agents and the decay factors of the cells are shown in Figure 3-2. There, the goal was to maintain the coverage level of every cell at least equal to $\underline{Z} = 20$. In Section 3-1 an optimization horizon of 10 steps was considered. However, when designing the closed paths this horizon is too short for the agents to cover every cell and return to their initial poses. Therefore, a longer horizon needs to be established.

In general, the choice of the horizon length is affected by choice of the initial reset value $\bar{Z}$. Here we randomly choose the initial reset value to be equal to 500. Then, the horizon is chosen as the shortest one for which the optimization problem (4-11) is feasible. Here $N$ is set equal to 18. As expected the coverage level of the cells at the end of the horizon is not enough for the agents to follow these paths again. Therefore, the second step of the method is implemented and the optimal reset value (denoted by $\bar{Z}^*$) is found to be equal to 1995.5.

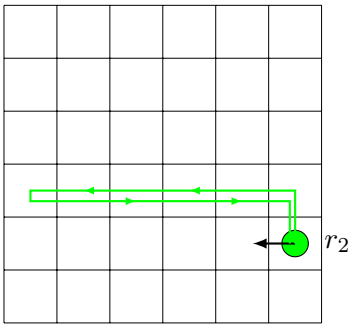
Having designed the closed paths of the terminal sequence and defined the appropriate reset value, we run the optimization problem (4-1) for a simulation horizon of 100 time steps. The initial terminal constraint set $X_0^f$ is chosen to be $S_0$. In Figure 4-2 the coverage level map of the area is shown for the first 10 steps. As before agents start moving towards the left part of the area where the cells with the lowest decay factors lie. After 10 steps there is only one cell with low coverage level (equal to 112), shown in dark blue. However, contrary to the situation shown in Section 3-1 here there are two agents nearby with the closest requiring only 5 steps to reach the cell. Based on the decay factor of the cell ($d_{15} = 0.75$) its coverage level does not drop below $\underline{Z}$ in less than 6 steps. Therefore, due to the terminal constraint the task will be feasible. Finally, it should be noted that agents aim at maximizing the coverage level of the area. Therefore, their actual paths shown in Figure 4-2 are totally different to the proposed closed paths of Figure 4-1.

## 4-3 Conclusions

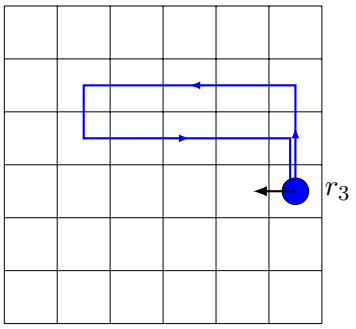
In this Chapter the feasibility of the persistent coverage task is studied. By definition, the task is feasible if agents are able to infinitely move around the area and maintain the coverage level of the cells above $\underline{Z}$. To achieve this, a finite sequence of terminal constraint sets is considered. This sequence is designed such that the union of its sets describes a group of closed paths as also the evolution of the coverage level of the cells when the aforementioned paths are followed. The idea behind this sequence is that if its closed paths are properly



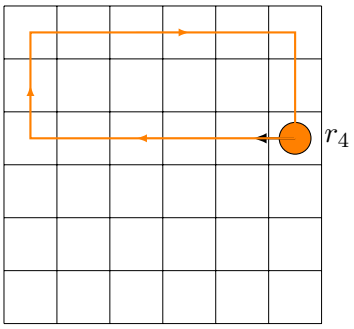

**(a)** Closed path designed for agent $r_1$




**(b)** Closed path designed for agent $r_2$




**(c)** Closed path designed for agent $r_3$



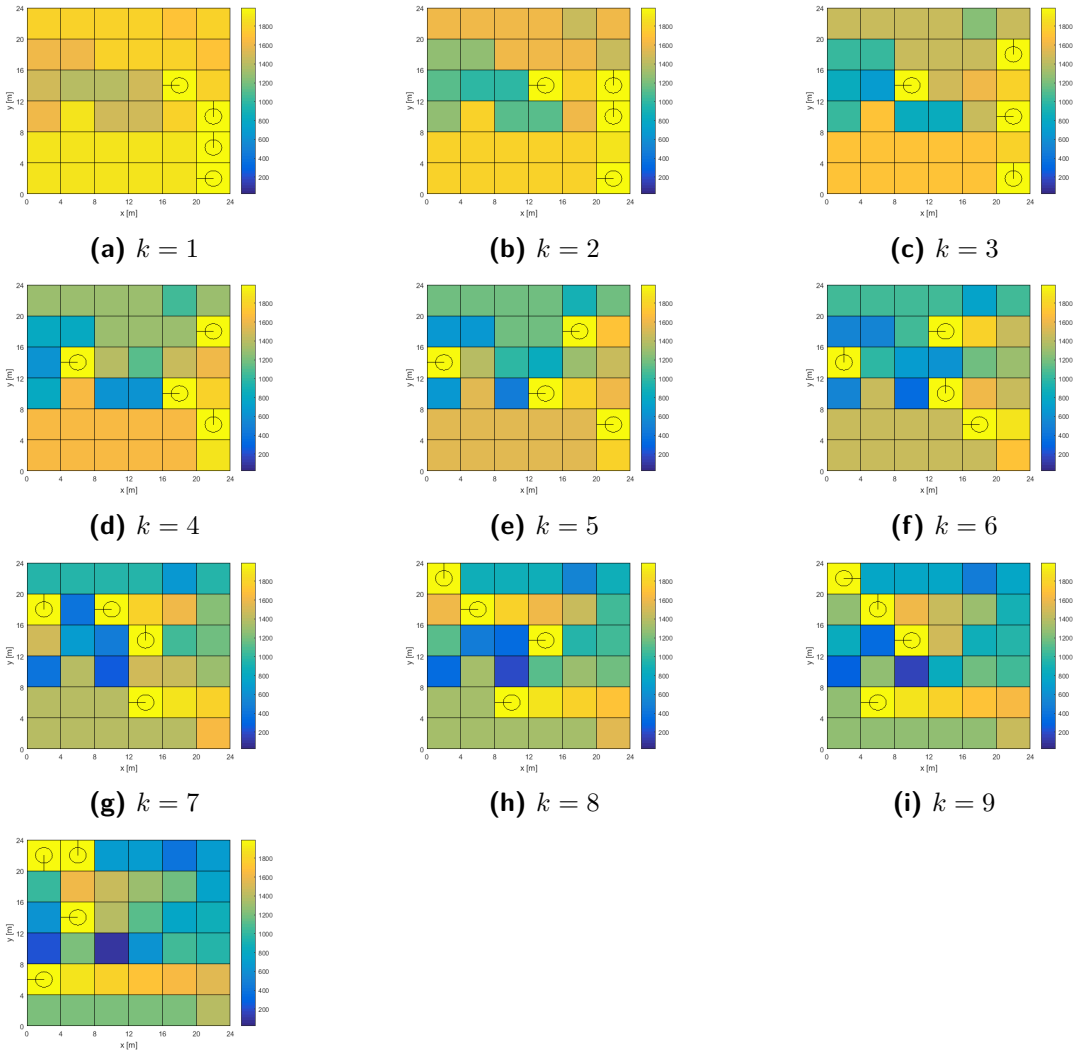

**(d)** Closed path designed for agent $r_4$

**Figure 4-1:** Example of the Closed Paths defined by the union of the Time-Variant Terminal Constraint Sets when a $6 \times 6$ grid is considered and 4 agents are employed for the task. The decay factors are shown in Figure 3-2. Here $\bar{Z}^* = 1995.5$ and $N = 18$.

constructed and agents follow them repeatedly, the coverage level of the cells will always remain above $\underline{Z}$.

Let each set include the proposed agents' poses and the coverage level of the cells resulting from these actions. Considering the finite horizon optimization problem of Chapter 2, we force the final poses of the agents and coverage level of the cells to be identical and at least equal respectively to the corresponding values of the set. Then, it is proved that when the new problem is solved recursively, it will always be feasible as there always exists a set of poses, belonging to the next set of the sequence (from the current terminal set) that maintains the coverage level of the cells above the desired $\underline{Z}$.

In addition to the feasibility proof a two-step method is proposed for designing the set sequence. In the first step a set of closed paths is designed such that all cells in the area get covered at least once. However, the resulting paths guarantee the lower bound of the coverage levels when traversed only once. Therefore, a second step is considered in which the minimum reset value is asked such that the lower coverage level bound is guaranteed as agents follow these paths repeatedly. Finally, the efficiency of the 2-step method and the feasibility of the problem are verified in simulation for the case scenario presented in Section 3-1.



**(a)** $k = 1$ **(b)** $k = 2$ **(c)** $k = 3$

**(d)** $k = 4$ **(e)** $k = 5$ **(f)** $k = 6$

**(g)** $k = 7$ **(h)** $k = 8$ **(i)** $k = 9$

**(j)** $k = 10$

**Figure 4-2:** Snapshots of the Coverage Level Map when a team of 4 agents is employed. The resulting paths are found as a solution to the optimization problem of (4-1) which includes the terminal constraint.

# Chapter 5

# Distributed Formulation

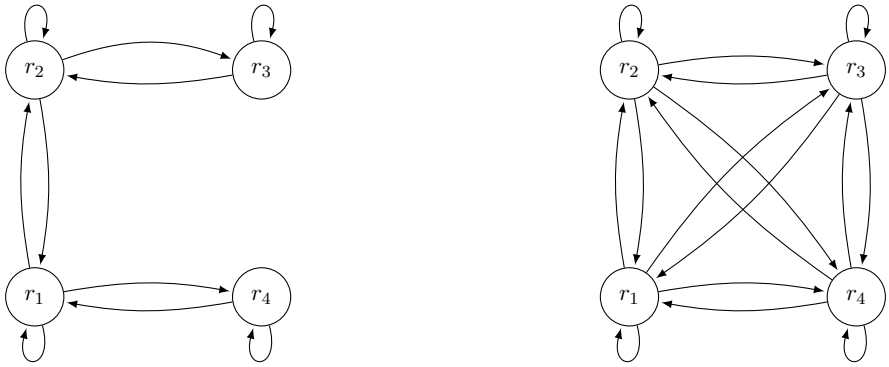
In the previous Chapters we considered the persistent coverage problem and solved it in a centralized fashion. There a single, central agent was responsible for receiving the position and coverage level information from its peers and planning their paths. In that method the resulting paths, designed based on the global information available achieve a high coverage level in the area and are collision-free. However, assigning the planning task to a single agent often exhibits severe limitations too. Due to the nature of the problem when large teams are considered solving the persistent coverage problem becomes computationally prohibitive for online planning (Section 3-3-1). Moreover, we can not dismiss the possibility of task failure due to sudden hardware failure of the agent in charge.

To avoid such problems in this Chapter we propose a distributed formulation of the problem. Here each agent solves a local path planning problem for itself and a subset of the team, called from now on *neighbors*. In this way the main problem is split into smaller sub-problems with less variables, thus becomes more computationally efficient. In this work we assume agents are collaborative. Therefore, prior to the planning step they communicate with their peers sending and usually receiving information. In that way they aim at maintaining up-to-date information of other agents' actions and the coverage level of the cells.

In Section 5-1 an initial attempt to formulate the distributed persistent coverage control problem is presented considering agents formulating a connected network. This method provides agents the most accurate information of the agents' positions and the coverage condition of the area. Nevertheless, the sub-problems often end up being identical to the centralized problem (the communication graph is complete). Therefore, in Section 5-2-1 a new method is proposed. Here agents consider as neighbors the two closest agents and are connected to a centralized base. The paths are designed based on information obtained by the neighbors while collisions with other, non-neighboring agents are avoided using the position information obtained from the centralized base. In Section 5-2-2 the results of the latter method are presented and the chapter ends with Section 5-3 where the main results are summarized.

## 5-1 Connected Network: A Motivating Example

In this Section a first approach to mathematically formulate the distributed persistent coverage task is presented. Let $\mathcal{G}(K, \mathcal{E})$ be the directed communication graph associated to the team of agents. Every node in the graph corresponds to the index of an agent and every edge $(r_1, r_2) \in \mathcal{E}$ denotes the ability of agent $r_1$ to communicate with $r_2$. Drawing inspiration from the work in [34], we assume that the network is *connected*. Therefore, there always exists a path of one or more edges over the graph connecting any pair of agents in both directions. Moreover, we define the *complete* graph as the graph in which each pair of nodes is connected with an edge. An example of a connected and a complete communication graph with 4 agents is shown in Figure 5-1a, 5-1b respectively.




**(a)** Connected Graph **(b)** Complete Graph

**Figure 5-1:** Examples of Connected and Complete Graphs when 4 agents are considered.

Let $\mathcal{N}_j \subseteq K$ denote the set of neighbors of agent $r_j$. In this Section as neighbors we consider the agents that are at most at a pre-defined distance from agent $r_j$. At each time step agents communicate with their neighbors and exchange coverage level information. Let $Z^{\text{com}}_{r_j}(w, k)$ denote the coverage level of cell $w$ agent $r_j$ sends to its neighbors at time step $k$ and $\bar{Z}_{r_j}(w, k)$ the coverage level of $w$ agent $r_j$ computes after making its move at $k$. Then, $Z^{\text{com}}_{r_j}(w, k)$ is computed based on the following equation:

$$Z^{\text{com}}_{r_j}(w, k) = \begin{cases} d_w Z_{r_j}(w, k-1), & p^k_{r_j} \neq c_w \\ \bar{Z}, & p^k_{r_j} = c_w \end{cases} \tag{5-1}$$

Agent $r_j$ sends this value to its neighbors and receive theirs. Based on the information received and its own it updates its information by keeping the most accurate/up-to-date value per cell. By definition, the coverage level value of each cell is monotonically decreasing between subsequent visits at the cell. Therefore, the higher the coverage level of a cell the more recently the cell was visited. Thus, it is natural to consider as the most up-to-date coverage level the maximum value among the available. After the update agent $r_j$ sends this information to its neighbors and the procedure is repeated for a finite number of steps until agents reach consensus on the coverage level of the area.

Based on this information each agent $r_j$ solves a local path planning problem of the form (2-17) the solution of which defines the paths of $r_j$ and its neighbors. However, each agent implements only the first step of the path it computed for its own.

The effectiveness of the algorithm is shown over a horizon of 50 steps when the case scenario introduced in Section 3-1 is considered. In this scenario the area is decomposed to a $6 \times 6$ grid and 4 agents are employed for the task. The agents are initially placed on the right side of the area as shown in Figure 3-1. The decay factors of the cells are shown in Figure 3-2.

The feasibility of the task over the simulation horizon is generally dependent on the reset value $\bar{Z}$ and the communication radius. Nonetheless, the choice of the one highly affects the other. For this reason, here we choose randomly the reset value to be $\bar{Z} = 3000$. Then, the communication radius is chosen as the minimum value for which the network remains connected over the simulation horizon. Here the communication radius is set equal to 22 m.

Initially, the distributed problem is solved by letting each local path planning problem converge to the optimal solution (*D-O method*). However, the average time required for finding this solution is found to be high enough for online planning(253 sec on average). An alternative proposed in Section 3-2 is considering the first feasible solution of each local problem. This solution is found considerably fast. Nevertheless, when agents follow the fist feasible paths the total coverage level of the area is significantly low. In order to keep a balance between the computational time and coverage performance we consider the best solution found within 30 sec for each local problem (*D-TL method*).

In Figure 5-2 the normalized coverage level is presented over the simulation horizon averaged by the number of the cells for both methods. Here the actual coverage level of each cell is considered and computed as the maximum value among the corresponding coverage level information agents possess after they implement their actions, thus $\max_{r_j \in K} Z_{r_j}(w, k)$. Although optimality is sacrificed in D-TL method the coverage level of each cell per time step is almost identical to the one of the D-O method. However, when a time limit is introduced the number of times the communication graph is complete increases (from 48 to 60%) as shown in Figure 5-3. Thus, the local path planning problems become identical to the centralized ones.

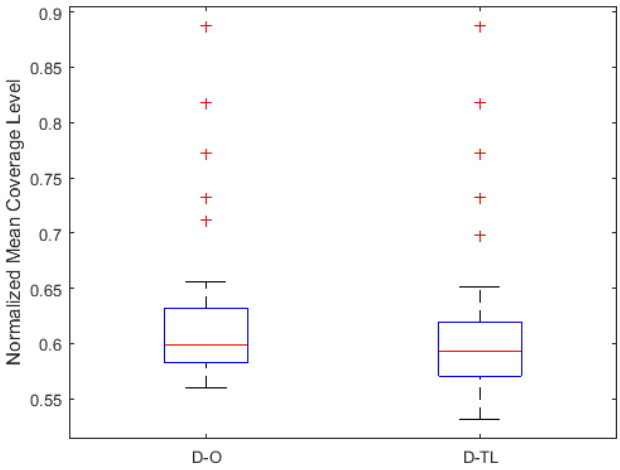
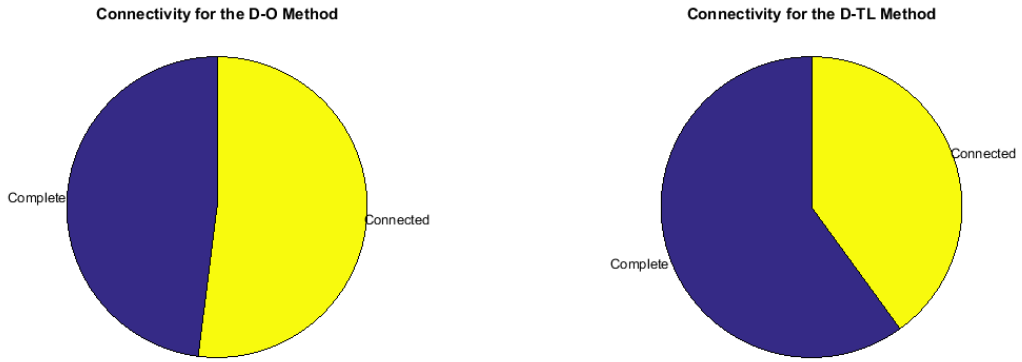



**Figure 5-2:** Normalized Coverage Level over the Horizon averaged by the Number of Cells when the Network is connected. Here *D-O* stands for the distributed problem considering the optimal solution and *D-TL* for the one considering the best solution within the Time-Limit.

Irrespective the way the solution of each local problem is obtained maintaining the network connected over the horizon requires a large communication radius. This in addition to the constrained motion of the agents results in agents forming a complete communication graph for a large portion of the simulation horizon. In that case agents communicate with the rest of the team and solve a centralized path planning problem. In this example solving this kind of problems requires maximum 700 sec. However, when larger teams are considered the computational complexity of the problem increases significantly. This makes the D-O and D-TL methods prohibitive for online planning. Therefore, a new method is designed that prevents agents from forming a complete communication graph thus allowing them solve smaller-sized path planning problems.




**(a)** Optimal Solution **(b)** Best Solution until the Time-Limit

**Figure 5-3:** Distributed Formulation with Connected Network: The Nature of the Communication Graph when the Optimal Solution and the Best Solution obtained within the time-limit (30 sec) is considered respectively for each local problem.

## 5-2 Fixed-Sized Neighboring Sets

In this Section a new distributed formulation of the problem is presented in which agents are able to communicate with a fixed number of their peers. Consequently, the local path planning problems designed at each step are solved for a fixed number of agents. As shown in Section 3-3-1 when 4 or more agents are considered the computational time of the optimization problem increases significantly. Aiming at less computationally intensive problems, in this method each local problem is solved for a team of 3 agents (team consisting of the agent under consideration and its neighbors). In Section 5-2-1 the new problem formulation is presented and in Section 5-2-2 its performance is discussed compared to the previous method.

### 5-2-1 Introducing the Method

In this problem each agent considers as neighbors the two closest agents. If there exist multiple agents at the shortest distance from the agent under consideration, as neighbors are

considered those with the lowest index. Similarly if multiple agents are at the second shortest distance from the agent, as a second closest neighbor is chosen the one with the lowest index.

At each time step each agent $r_j$ defines its neighboring set $\mathcal{N}_j$ and the coverage level map of the area based on the available information using (5-1). Let $\mathcal{M}_j \subseteq K$ be the set of the indexes of the agents that consider agent $r_j$ as their neighbor. Since the network is not assumed connected there might be time instants at which $r_j$ becomes disconnected. Therefore, $\mathcal{M}_j$ can be also empty.

The agent $r_j$ sends the information to its neighbors and receives information from the agents with indexes in $\mathcal{M}_j$ if the set is not empty. When the information exchange is over agents update the coverage level information as follows:

$$Z_{r_j}(w,k) = \max_{r_\rho \in \mathcal{M}_j \cup \{r_j\}} Z^{\text{com}}_{r_\rho}(w,k), \quad w \in I,\ r_j \in K \tag{5-2}$$

Based on the updated information each agent $r_j$ solves a local path planning problem considering also its neighbors. The solution of the problem defines the paths of the agents in $\mathcal{N}_j \cup \{r_j\}$. However, as in Section 5-1 each agent implements only the first step of the path it computed itself for its own.

Since the neighboring set $\mathcal{N}_j$ is of fixed cardinality when agent $r_j$ solves the local path planning problem it does not take into consideration the other $n_r - 2$ agents' decisions. Although this might not affect the agents' safety when these agents are far away this may no longer be valid when agents are at close proximity. An example is shown in Figure 5-4. Agent $r_1$ considers as neighbors the agents $r_2, r_3$ as they are closer to it. Moreover, it receives information from the same agents. Thus, $\mathcal{N}_1 = \mathcal{M}_1 = \{r_2, r_3\}$. In a similar manner we can define $\mathcal{N}_4 = \mathcal{M}_4 = \{r_5, r_6\}$. Agents $r_1, r_4$ ignore the existence of one another. Therefore, it is possible for both of them to choose simultaneously to move at the cell in the middle and thus collide. Other cases of possible collisions are shown in Figure 5-5.

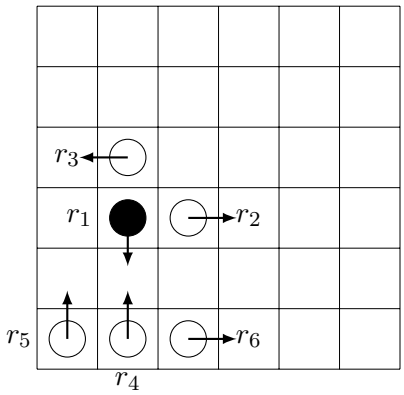



**Figure 5-4:** Example of Collision due to the Fixed Cardinality of $\mathcal{N}_1$

To resolve this problem we consider agents having access to a centralized base including position information for every agent in the team. Agents update the information in the base after completing their actions. In that way they are able to retrieve the actual position of all the agents in the team and constrain their moves so as to avoid future collisions.

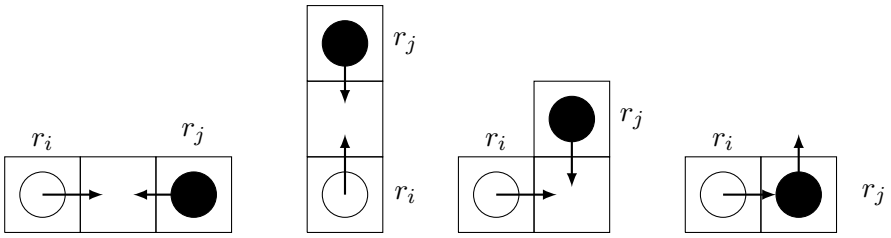



**Figure 5-5:** Examples of Possible Collision Scenarios for Agent $r_i$

At each local path planning problem collision avoidance constraints should be added not only for the agent in charge but also its neighbors. For example when agent $r_j$ solves its local path planning problem collisions should be avoided for any agent in $\mathcal{N}_j \cup \{r_j\}$. Let $\sigma_w^{r_l}(k)$ be a binary variable expressing whether agent $r_l$ is at cell $w$ at time step $k$ or not. Then, constraints of the form $\sigma_w^{r_l}(k+1) = 0,\ r_l \in \mathcal{N}_j \cup \{r_j\},\ w \in I$ should be introduced to the optimization problem in the following cases:

$$p_{r_\rho}^k = c_w, \qquad r_\rho \in K \backslash (\mathcal{N}_j \cup \{r_j\}) \tag{5-3}$$

$$(p_{r_l}^k = c_{w-1}) \wedge (\theta_{r_l}^k = 0) \wedge (p_{r_\rho}^k = c_{w+1}) \wedge (\theta_{r_\rho}^k = \pi), \qquad r_\rho \in K \backslash (\mathcal{N}_j \cup \{r_j\}) \tag{5-4}$$

$$(p_{r_l}^k = c_{w-C}) \wedge \left(\theta_{r_l}^k = \frac{\pi}{2}\right) \wedge (p_{r_\rho}^k = c_{w+C}) \wedge \left(\theta_{r_\rho}^k = \frac{3\pi}{2}\right), \qquad r_\rho \in K \backslash (\mathcal{N}_j \cup \{r_j\}) \tag{5-5}$$

$$(p_{r_l}^k = c_{w-1}) \wedge (\theta_{r_l}^k = 0) \wedge (p_{r_\rho}^k = c_{w+C}) \wedge \left(\theta_{r_\rho}^k = \frac{3\pi}{2}\right), \qquad r_\rho \in K \backslash (\mathcal{N}_j \cup \{r_j\}) \tag{5-6}$$

$$(p_{r_l}^k = c_{w-1}) \wedge (\theta_{r_l}^k = 0) \wedge (p_{r_\rho}^k = c_{w-C}) \wedge \left(\theta_{r_\rho}^k = \frac{\pi}{2}\right), \qquad r_\rho \in K \backslash (\mathcal{N}_j \cup \{r_j\}) \tag{5-7}$$

$$(p_{r_l}^k = c_{w+1}) \wedge (\theta_{r_l}^k = \pi) \wedge (p_{r_\rho}^k = c_{w+C}) \wedge \left(\theta_{r_\rho}^k = \frac{3\pi}{2}\right), \qquad r_\rho \in K \backslash (\mathcal{N}_j \cup \{r_j\}) \tag{5-8}$$

$$(p_{r_l}^k = c_{w+1}) \wedge (\theta_{r_l}^k = \pi) \wedge (p_{r_\rho}^k = c_{w-C}) \wedge \left(\theta_{r_\rho}^k = \frac{\pi}{2}\right), \qquad r_\rho \in K \backslash (\mathcal{N}_j \cup \{r_j\}) \tag{5-9}$$

Constraint (5-3) guarantees that agent $r_l$ will not visit a cell occupied by another agent at the current time step. In that way a collision could be avoided in case agent $r_\rho$ decides to stay or turn at place. Constraints (5-4)-(5-9) are introduced so as to avoid collisions when two agents are about to enter the same cell. The latter constraints cover cases similar to the first 3 examples of Figure 5-5. In general the constraints introduced here consider possible collision scenarios with agents that are not $r_j$ or any of its neighbors. Collision avoidance between agents with indexes in $\mathcal{N}_j \cup \{r_j\}$ is already taken into consideration and guaranteed by the constraints introduced in the optimization problems of Chapter 2. An overview of the steps to be followed by every agent at each time step is given in Algorithm 1 by the procedure *MakeAction*.

### 5-2-2 Simulation Results

In this Section the computational results are presented for the distributed problem introduced in Section 5-2-1. Each local path planning problem is formulated by (2-17) with the additional

**Algorithm 1** Distributed Formulation II: Steps to be followed by agent $r_j$ at time step $k$

**Input** $(p_{r_j}^{k-1}, \theta_{r_j}^{k-1}),\ Z_{r_j}(w,k-1),\ w \in I$
**Output** $(p_{r_j}^{k}, \theta_{r_j}^{k}),\ Z_{r_j}(w,k),\ w \in I$

1: **procedure** MAKEACTION
2: Define $\mathcal{N}_j$
3: Compute $Z_{r_j}^{\text{com}}(w,k), w \in I$ using (5-1)
4: Send $Z_{r_j}^{\text{com}}(w,k), w \in I$ to agents with index $r_\rho \in \mathcal{N}_j$
5: **if** $\mathcal{M}_j \neq \emptyset$ **then**
6: $Z_{r_j}(w,k) = \max_{r_\rho \in \mathcal{M}_j \cup \{r_j\}} Z_{r_\rho}^{\text{com}}(w,k), \quad w \in I$
7: **else**
8: $Z_{r_j}(w,k) = Z_{r_j}^{\text{com}}(w,k), \quad w \in I$
9: Get $(p_{r_{\rho'}}^{k-1}, \theta_{r_{\rho'}}^{k-1}),\ r_{\rho'} \in K \backslash (\mathcal{N}_j \cup \{r_j\})$ from the Centralized Base
10: Introduce any Collision Avoidance Constraints when any case (5-3)-(5-9) is true for $r_\rho \in \mathcal{N}_j \cup \{r_j\}$
11: Solve the Path planning problem (2-17) for agents $r_\rho \in \mathcal{N}_j \cup \{r_j\}$ with the extra collision avoidance constraints
12: Make the first move of the designed path for agent $r_j$
13: $Z_{r_j}(w,k) \leftarrow \begin{cases} d_w\ Z_{r_j}(w,k), & p_{r_j}^k \neq c_w \\ \bar{Z}, & p_{r_j}^k = c_w \end{cases}$
14: Update Pose Information in the Centralized Base

collision avoidance constraints. Here a time limit of 30 sec is set as finding the optimal solution for each sub-problem was found to be too computationally intensive. Throughout this Section the method introduced in Section 5-2-1 will be denoted for simplicity by *D-II*.

Initially, the method is implemented for the case scenario introduced in Section 3-1 and compared to the *D-TL* method of Section 5-1. A simulation horizon of 40 time steps is considered and the communication radius for D-TL is set equal to 22 m. Here a reset value $\bar{Z} = 10^4$ is chosen as found to be the minimum value for which each local path problem of D-II is feasible. In Figure 5-6 the normalized coverage level of each cell is presented averaged over the number of cells and simulation time steps. Here the actual coverage level of each cell is considered and computed as the maximum among the corresponding coverage level values of the agents that are defined after agents make their actions (Step 13 in Algorithm 1).

As shown in the Figure the D-TL method outperforms D-II for the majority of the time steps. At the beginning, irrespective of the method implemented agents move towards the opposite side of the grid where the cells with the highest decay rates lie. Therefore, the mean coverage level values of both methods are almost identical. However, as time passes agents tend to make different decisions depending on the method implemented. This is related to the accuracy and amount of information agents have available when solving their local problems.

In D-TL method agents are always forming a connected network and over 80% of the simulation horizon the communication graph is complete. Here agents are able to exchange information with their peers multiple times (or once in case of a complete graph) until they reach consensus. At that time they possess the most accurate coverage level information. Therefore, they are able to decide their next move so as to maximize the actual coverage

level of the area. On the other hand, in D-II agents may be disconnected at some point, thus rely on their own information. Otherwise, they exchange information with their peers only once. Therefore, even though they form a connected network over the 60% of the simulation horizon each agent maintains a different coverage level map of the area. Hence, it is possible for the agents to visit cells previously covered by others at a short notice.

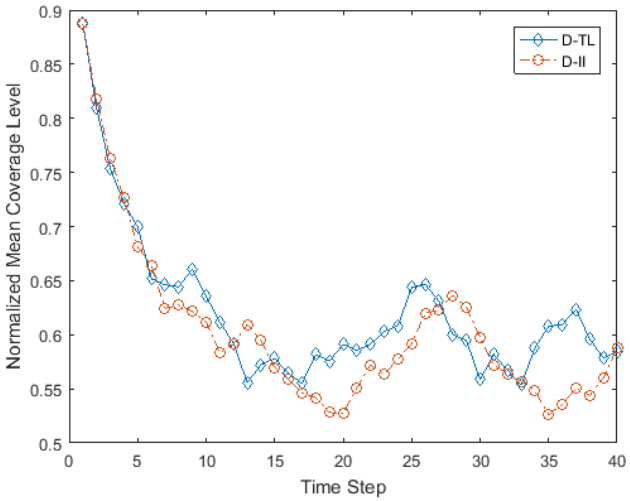



**Figure 5-6:** Normalized Coverage Level averaged by the Number of Cells over the Simulation Horizon for the case scenario presented in Section 3-1 when D-TL and D-II are implemented.

In Table 5-1 the maximum length of the time interval between two subsequent visits at a COI is presented for the case scenario of Section 3-1 when D-TL and D-II are implemented. Let this length be denoted by $v_w$ for any COI with index $w$. In this example there are 11 cells with decay rate at most equal to 0.85. When D-II is implemented the COIs with the highest decay rate are visited more frequently (every 8 and 13 time steps). By contrast, when D-TL is implemented agents sacrifice frequent visits at these cells (they visit them at most every 15 and 17 time steps) so as to spread around the area and visit the other COIs too. This is a direct consequence of the fact that agents are aware of all the other agents' actions. Having the most accurate coverage level information available they are able to make decisions that maximize the actual coverage level of the area. For this reason, they strive at visiting every COI more frequently than expected (less than $t_w$ steps).

Further experiments are conducted in which different coverage decay factors are assigned to the cells. Here we consider the 20 different combinations of $d_w$ introduced for the experiments of Section 3-1 when a grid of size $6 \times 6$ cells is considered. Initially, a team of 4 agents is employed for the task with initial poses shown in Figure 3-1. Then, the performance of the D-II method is evaluated for a larger team of 6 agents. The extra agents are placed at the free cells of the last column of the grid with their heading equal to $\pi$. Here, the reset value is experimentally chosen to be $\bar{Z} = 15 \cdot 10^4$. For this value 75% of the cases were found to be feasible when D-II is implemented irrespective the team size.

In Figure 5-7 the normalized coverage level is presented averaged over the number of cells and simulation steps for the D-TL and D-II method. In D-TL the same reset value is considered and the communication radius is set equal to 22 m. When 6 agents are considered the amount

| $c_w$ | $d_w$ | $t_w$ | $v_w$ with D-TL | $v_w$ with D-II |
|---|---|---|---|---|
| $(6, 14)$ | 0.7 | 17 | **17** | 13 |
| $(10, 14)$ | 0.7 | 17 | **15** | 8 |
| $(10, 10)$ | 0.75 | 21 | 10 | 10 |
| $(14, 10)$ | 0.75 | 21 | **17** | 14 |
| $(2, 14)$ | 0.75 | 21 | 9 | 12 |
| $(14, 14)$ | 0.75 | 21 | 13 | 17 |
| $(2, 10)$ | 0.8 | 27 | 10 | 12 |
| $(2, 18)$ | 0.8 | 27 | 11 | 15 |
| $(6, 18)$ | 0.8 | 27 | 12 | 15 |
| $(22, 18)$ | 0.85 | 38 | 11 | 18 |
| $(18, 22)$ | 0.85 | 38 | 12 | 13 |

**Table 5-1:** Maximum Time Elapsed since the Last Visit at a COI $w$ (denoted by $v_w$) for the case scenario presented in Section 3-1 when D-TL and D-II methods are implemented. In bold are shown the increased $v_w$ values of D-TL when compared to D-II.

of coverage provided increases significantly by almost 10% compared to the performance of the 4 agents.

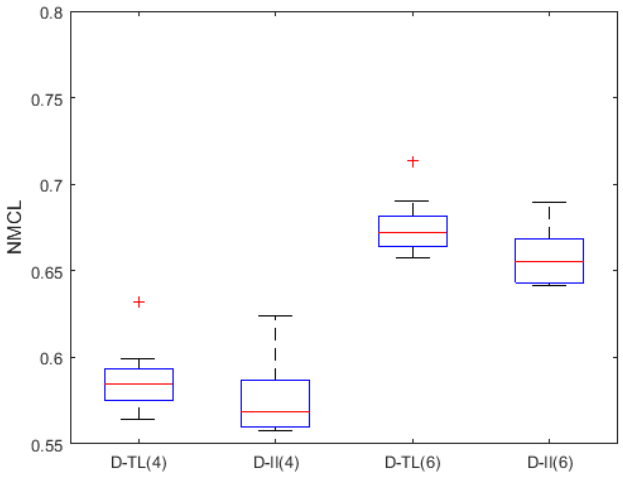
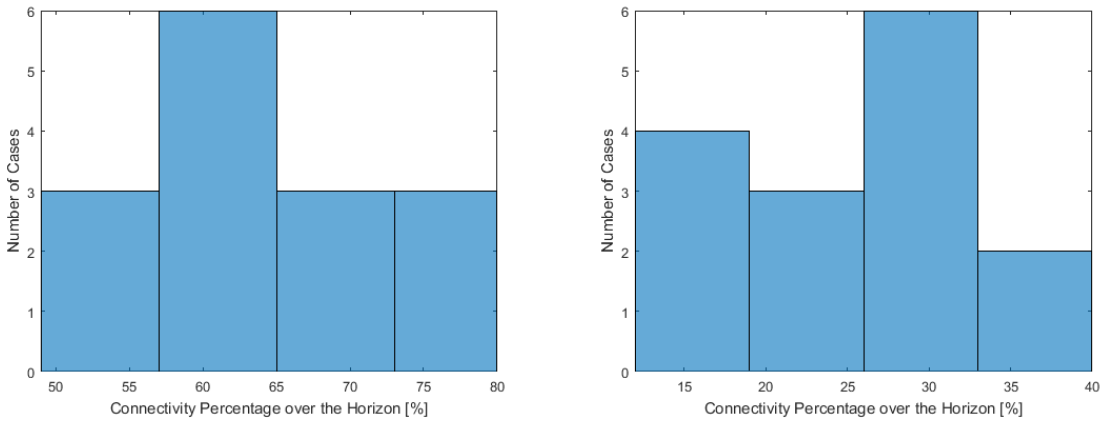



**Figure 5-7:** Normalized Mean Coverage Level per Cell and Time Step (NMCL) when a team of 4 and 6 agents is considered

Irrespective the team size the difference in the coverage performance of the proposed methods is only 1.5%. This difference is again related to the amount of information available to the agents when solving the local path planning problems. Even though in D-II agents form a connected network over 55% of the simulation horizon when $n_r = 4$ and 40% when $n_r = 6$ (Figure 5-8) they share information with their peers only once. Each agent decides its actions based on its own idea of the coverage condition of the area. Despite the lack of information it is noticeable that D-II performs equally good to D-TL without excessive information flow and the strong assumption of connectivity.




**(a)** $n_r = 4$ **(b)** $n_r = 6$

**Figure 5-8:** Percentage of Time the Network is connected when D-II is implemented

## 5-3 Conclusions

In this Chapter two different distributed formulations of the persistent coverage task are introduced. In the first method the agents are assumed always connected. Hence, they are able to exchange and update their coverage level information until a maximum consensus is reached. Based on the updated information each agent solves a local path planning problem and implements the first step of the path it computed itself. In this method agents make their decisions based on the actual coverage level of the area. Therefore, they achieve a maximum coverage performance. However, for the majority of the simulation steps the communication graph is complete. Thus, agents communicate with any other agent in the team and solve a centralized problem.

Depending on the team size this problem might become computationally intensive. To avoid such cases a second distributed formulation is proposed. In this formulation agents are able to send information to only two agents. In that way the local path planning problem solved is always of fixed team size equal to 3. Agents send coverage level information to their neighbors and if applicable receive from other agents in the team. After that they update their information and solve a local path planning problem for themselves and their neighbors. However, as before each agent implements the first step of the path it computed on its own.

In the second method the network may become disconnected at some point. Even when connected, since consensus is not achieved each agent maintains its own (local) coverage level map based on which it decides its future actions. Despite the information inaccuracy agents achieve an equally good coverage performance as in the first method with the average coverage level of a cell over the horizon to be only 1.5% less that the corresponding value of the first method.

# Chapter 6

# Conclusions and Future Work

In this Chapter the contributions of this thesis are summarized and recommendations are made for future work.

## 6-1 Conclusions

Path planning for persistent coverage control considers the problem of designing agents' paths so as every point in the area of interest is visited more than once. A vast amount of work considers agents planning their moves so as the maximum time elapsed since a point was last visited is minimized over a finite set of points in the area. In these methods the area is considered static and information on the amount of coverage provided by the agents is not available. More recent work considers the persistent coverage control problem for areas characterized by a time-varying property called *coverage level* (non-static environment). Contrary to previous work, here the goal is to maintain the coverage level of the area at a predefined, desired level. Agents design their paths so as to maximize coverage by passing through less covered parts of the area. However, it is not clear how often the points are covered and how much the maximum coverage loss between subsequent coverage times is at each point.

Aiming at overcoming the limitations of the aforementioned work, in this thesis a two dimensional area is considered and a method is proposed that provides coverage level and visitation frequency information for a finite set of points in the area. Here the goal is to design the agents' paths so as the coverage level at each point is bounded from below by a constant, pre-defined level of coverage $\underline{Z}$. When agents visit a point in the set its coverage level resets to a constant value. Then, in order for the coverage level of each point to always be at least equal to $\underline{Z}$ the point should be visited within a predefined number of time steps.

In this work the area is decomposed into a grid of square cells and the finite set of points is chosen as the set of the centers of the cells. At each time step agents are placed at the center of a cell with their heading taking values from a finite set. The size of the square cells is chosen such that every point in the cell gets covered by the time an agent is placed at its center. In that way agents provide persistent coverage to a finite set of points while guaranteeing

complete coverage of the area when every cell is visited at least once. For simplicity we refer to the coverage level of the center of cell $w$ as the coverage level of cell $w$.

In Chapter 2 a MILP problem is designed that aims at maximizing the coverage level of every cell over a finite planning horizon while minimizing cell coverage by different agents at consecutive time steps. Two problem formulations are proposed that differ on the definition of the binary variables expressing the agents' poses. In formulation I a set of binary variables, proportional to the number of agents is introduced expressing whether an agent $r$ has a pose $q \in V$ at time step $k$. The efficiency of this problem is evaluated and the problem is found to be too computational intensive requiring on average 5000 sec to converge to the optimal solution. This is related to the fact that the solution space of the problem is not close enough to its convex hull. Hence, the solver requires more time to evaluate possible solutions before finding the optimal one. To resolve this problem, adding a set of inequality constraints (*cutting planes method*) is proposed so as a set of candidate solutions with integer variables taking fractional values is excluded from further examination. In our case finding these constraints is found to be non-trivial. Therefore, a new formulation of the problem is designed in which the binary variables $x^k_{qq'}$ introduced express whether an agent performs a specific pose transition $(q, q') \in E$ at time step $k$. As shown, formulation II outperforms I requiring only 6% of the computational time of formulation I for finding the optimal solution to a number of cases introduced.

In Chapter 3 formulation II is further examined and its optimal solution is evaluated in terms of coverage performance and computational effort. As shown agents provide a high amount of coverage in the area, however, at the cost of computational effort. Although the time to the optimal solution is significantly less than of formulation I it is still found to be high enough for online planning. For this reason, the first feasible solution is proposed as alternative choice for defining agents' paths and its quality is examined in terms of coverage level and computational effort. As shown the first feasible paths are found at negligible time. However, their coverage performance depends on the minimum age of a COI and may differ from the optimal performance by 10%. In the last Section of the Chapter the scalability of the problem is examined with respect to the number of agents, the horizon length and the number of cells. As expected the computational complexity of the problem increases as the team size and planning steps increase. However, when a higher grid resolution is considered the time to the optimal solution is significantly less when compared to coarser ones.

In Chapter 4 the feasibility of the optimization problem is studied. Here a finite sequence of terminal constraint sets is constructed. Each set includes a pose for every agent in the team and the coverage level of the cells in the grid. The poses in each set are defined such that the union of the sets in the sequence introduces a closed path for each agent. Then, the coverage level values of the set are defined based on the agents' poses in the set and the coverage level values at the previous time step when agents are assumed to follow their closed paths. Considering a terminal set of the sequence, a set of constraints is introduced in the optimization problem of Chapter 2 that forces 1) the final poses of the agents to be identical to the ones of the set and 2) the final coverage level values to be at least equal to the corresponding values of the set. Then, it follows that if the problem at the current time step is feasible, the problem introduced at the next step will also be feasible as long as its terminal constraints are associated to the next terminal constraint set in the sequence (or the first set when the current choice of terminal set is the last of the sequence). In the same Chapter a two-step method is proposed for designing the closed paths. In the first step the closed paths

are found as a solution to a modified version of the proposed MILP. However, when agents follow these paths the coverage level of the cells is not always bounded from below by $\underline{Z}$. Therefore, in the next step given the closed paths of step 1, a LP problem is solved for finding the minimum reset value $\bar{Z}$ for which the coverage level boundness constraints are satisfied.

In Chapter 5 two distributed formulations of the problem are introduced. In the first method agents are assumed always connected. At each time step agents identify their neighbors and exchange coverage level information until consensus is reached. Here as neighbors are considered the agents placed within a pre-defined distance from the agent under consideration. Based on the updated information agents solve a local path planning problem taking into consideration their neighbors and implement the first step of the path they computed for themselves. In this method the network often forms a complete communication graph. Therefore, agents solve a centralized problem the complexity of which increases with the number of agents. To avoid such cases a second formulation is proposed in which agents consider as neighbors the two closest peers. In this method agents communicate with their peers only once. Therefore, each agent maintains its own coverage level map based on which solves a local path planning problem. The performance of each method is evaluated in simulation. As shown, despite the inaccuracy on the coverage level information in the second method agents maintain the average coverage level of the cells at a high level, only 1.5% less than the corresponding value of the first method.

## 6-2 Future Work

In this thesis an attempt is made to integrate the two basic solution approaches of the persistent coverage control problem so as complete coverage information is always available for the area of interest. Towards this goal there are still several open questions and challenges to be faced. In this Section a few of them are discussed.

**Introduce a Charging Procedure for Practical Experiments** Persistent Coverage tasks may often be long lasting especially when large areas are to be covered. On the other hand, the battery life of the agents is often too short(especially when MAVs or AUVs are considered). Therefore, aiming at a practical implementation of the method, we need first to establish an automatic charging policy for the agents under consideration. A straightforward way to achieve this is by considering the cells agents are initially placed at as charging stations. Then, a set of constraints may be introduced to the problem forcing agents to visit these cells by the time their energy is below a certain level. As in practice a $n_r$ number of charging stations might not available for each task, a fuel management system should be introduced with less stations spread in the area of interest [51]. Then, a schedule should be made so as agents visit the stations before they run out of battery, without colliding with their peers and without leaving cells with inadequate coverage level.

**Consider the Time required for Cell Coverage** In this work, as discussed in Section 2-1-3 the time required for covering each cell is considered either negligible or known and constant for all cells. However, depending on the application this assumption may not be valid. For example when an induction heating task is considered for a domestic hob [52] the time required

for heating is neither negligible nor the same for different sizes of inductors. In [53] the optimal coverage times are computed for a single robot employed for covering a set of points in the area at a predefined revisitation frequency. In [52], [39] the problem of finding the optimal coverage times of a finite set of points is addressed for multiple robots when their paths are known. Nonetheless, to our best of knowledge the problem of defining the amount of time required for covering a set of points in the area while planning the agents' paths online is still an open problem.

**Cover Multiple Cells per Time Step** So far we considered agents able to cover every point in a cell at a single time step. Moreover, we assumed that the coverage level of a cell offered by agents placed at neighboring cells is negligible. This means that either the size of the cell is almost identical to the size of the agent's sensing area or the contribution of these agents to the coverage level of the neighboring cells is ignored. To reduce conservatism we may modify the coverage level dynamics by introducing a term coupling the coverage condition of the cell under consideration with that of its neighbors. In that way the resolution of the grid becomes independent of the agents' sensing abilities and a positive effect on the computational time of the problem may be expected.

**Extend the Method to Heterogeneous Agents** After the realization of the previous suggestion the method could be generalized to heterogeneous agents. Such agents, as often equipped with different sensors are able to provide better quality of coverage in the area while overcoming inherent limitations of a single platform (AGV, AUV, etc). In coverage control literature heterogeneous teams of agents were considered in locational optimization [54], [55]. In these articles the question of where to place the agents is answered so as the area of interest is completely covered. Recently, heterogeneous agents were also considered for the persistent coverage control problem in continuous space [38] and for covering a set of discrete points when their paths are given [52]. However, the effect of agents' heterogeneity on online persistent coverage control tasks in discrete spaces has not been studied yet.

**Distributed Formulation with Feasibility Guarantees** Although the distributed problems presented in Chapter 5 may work well over a finite simulation horizon, in practice their feasibility is not infinitely guaranteed. For this reason, a new distributed formulation of the problem may be proposed. In Model Predictive Control (MPC) literature the distributed control of similar problems with dynamically decoupled systems and coupled constraints has been addressed and different solution approaches were proposed. Examples are 1) the hierarchical method in [56] where a finite number of interconnected subsystems is considered and every subsystem solves a centralized MPC problem based on the plans of the previous (in index) subsystems and the predicted plans of the following ones and 2) the constraint-tightening approach of [57] in which a nested method is proposed where in the inner loop subsystems solve the MPC problem providing an approximate solution of the problem and in the outer loop bounds are set for the constraint violation. These methods may be modified and applied in our problem. However, it should be noted that since each subsystem solves a centralized MPC problem the computational time of the problem might still be high and increase with the number of agents.

# Appendix A

# Designing Closed Paths with Formulation II

In Chapter 4 a 2-step method is presented for designing the closed paths that guarantee recursive feasibility of the task. In the first step formulation I was considered as designing the closed paths was achieved by simply adding a set of $n_r + n_w$ constraints to the problem. While in that formulation forcing agents' final poses to be identical to the initial ones was achieved with (4-11q) in formulation II these constraints are not enough. This is a direct consequence of the fact that the binary variables expressing agents' poses do not have an explicit 1-1 correspondence to an agent but rather express whether an agent exists performing the corresponding pose transition. Therefore, it might happen that the final poses of some agents are identical to the initial poses of others.

In order to avoid this an extra set of variables needs to be added. These variables should specify and propagate over time the index of the agent performing a pose transition over the graph $G$. A similar work is done in [58] for the Multi-Depot Travelling Salesmen Problem. In [58] a set of integer variables is considered with each variable assigned to a node in the graph. In Travelling Salesman problem every node needs to be visited. Therefore, each variable is set equal to the index of the agent visiting the corresponding node.

Here we introduce a set of integer variables with each variable assigned to a cell and time step. Let these variables be denoted by $\kappa_w^k$. At each time step only $n_r$ cells are covered. Therefore, here we modify the definition given in [58] so as the variables to take values in the extended set $K \cup \{0\}$. If cell $w \in I$ is covered by agent $r \in K$ at time step $k$, $\kappa_w^k$ is equal to $r$ otherwise it is set equal to 0. This is equivalent to having the following constraints satisfied:

$$\kappa_w^k - n_r \sum_{(q,q') \in V_w} x_{qq'}^k \leq 0 \tag{A-1}$$

$$\kappa_w^{k-1} - \kappa_{w'}^k + n_r \, x_{qq'}^k \leq n_r, \qquad w \neq w' \tag{A-2}$$

$$\kappa_w^{k-1} - \kappa_w^k + n_r \sum_{(q,q') \in V_w'} x_{qq'}^k \leq n_r \tag{A-3}$$

$$\sum_{w=1}^{n_w} \kappa_w^k = \frac{n_r(n_r+1)}{2} \tag{A-4}$$

$$\kappa_w^k \in K \cup \{0\} \tag{A-5}$$

for any $w \in I$ and $k \in \{1, \ldots, N\}$. Constraint (A-2) is well defined for $(q, q') \in E,\ q \in \{4(w+1)+1, \ldots, 4w\},\ q' \in \{4(w'+1), \ldots, 4w'\},\ w, w' \in I,\ w \neq w'$.

Due to (A-1) and (A-5) if a cell is not covered at time step $k$ then the corresponding $\kappa_w^k$ will be set equal to 0. Otherwise, due to (A-2)-(A-3) at each time step there will be exactly $n_r$ binary variables active thus exactly $n_r$ constraints of the form:

$$\kappa_{w_1}^k \geq r_1$$
$$\vdots$$
$$\kappa_{w_{n_r}}^k \geq r_{n_r}$$

The variables $\kappa_{w_1}^k, \ldots, \kappa_{w_{n_r}}^k$ are non-zero integers belonging to the set $K$. Due to (A-4) this is possible either when the equality signs of the above constraints hold or when an agent's index is assigned to multiple variables. In case the latter scenario is true then an inequality constraint of the above is violated. Thus, the equality signs hold always and the variables are well-defined.

Constraints (A-1)-(A-5) are added to the optimization problem (2-17) in addition to the following constraints:

$$\kappa_w^0 = \begin{cases} r, & w = \lceil \frac{q_r}{4} \rceil \\ 0, & \text{otherwise} \end{cases} \tag{A-6}$$

$$\kappa_w^N = \kappa_w^0 \tag{A-7}$$

$$\sum_{(q,q_r)\in E} x_{q\,q_r}^N \geq 1, \quad r \in K \tag{A-8}$$

$$\sum_{k=1}^{N} \sum_{(q,q')\in V_w} x_{qq'}^k \geq 1, \quad w \in I \tag{A-9}$$

where $q_r \in V$ is the initial pose of agent $r$.

Here constraint (A-6) defines the initial conditions of the variables $\kappa_w^k$. Constraint (A-7) forces each agent $r$ to return at the cell it was initially placed while (A-8) guarantees that agent $r$ not only will return to the same position but will have also the same heading. Finally, (A-9) guarantees that each cell of the grid will be covered at least once.

In general, the extra integer variables added to the problem increase the computational time required for the problem to converge to the optimal solution. This is the case when the example of Section 3-1 is considered. Based on that, even though the computations are performed offline problem (2-9) is found to be a better choice for computing the closed paths of the first step.

# Bibliography


[1] Z. Gu, E. Rothberg, and R. Bixby, "Gurobi optimization." http://http://www.gurobi.com/resources/getting-started/mip-basics//, 2008.

[2] N. Nigam and I. Kroo, "Persistent surveillance using multiple unmanned air vehicles," in *IEEE Aerospace Conference*, pp. 1–14, 2008.

[3] C. H. Chen and K. T. Song, "Complete coverage motion control of a cleaning robot using infrared sensors," in *IEEE International Conference on Mechatronics*, pp. 543–548, 2005.

[4] D.W. Casbeer and D.B. Kingston and R.W. Beard and T. McLain, "Cooperative forest fire surveillance using a team of small unmanned air vehicles," *International Journal of Systems Science*, vol. 37, no. 6, pp. 351–360, 2006.

[5] J. A. J. Berni and P. J. Zarco-Tejada and L. Suarez and E. Fereres, "Thermal and Narrowband Multispectral Remote Sensing for Vegetation Monitoring From an Unmanned Aerial Vehicle," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 47, no. 3, pp. 722 – 738, 2009.

[6] M. Dunbabin, "Autonomous greenhouse gas sampling using multiple robotic boats," in *Springer Tracts in Advanced Robotics* (D. Wettergreen and T. Barfoot, eds.), vol. 113, ch. Field and Service Robotics, pp. 17–30, Springer, Cham, 2016.

[7] R. Smith, M.Schwager, S. Smith, D. Rus, and G. Sukhatme, "Persistent ocean monitoring with underwater gliders: Towards accurate reconstruction of dynamic ocean processes," in *IEEE International Conference on Robotics and Automation*, 2011.

[8] J. Van Beek and J. Farifteh and P. Janssens and H. Vandendriessche and T. Deckers and P. Coppin , "Water stress detection with high resolution satellite multispectral images in commercial pear orchards in Belgium," *Communications in Agricultural and Applied Biological Sciences* , vol. 77, no. 1, pp. 122–126, 2012.

[9] C. Brekke and A. H.S.Solberg, "Oil spill detection by satellite remote sensing," *Remote Sensing of Environment*, vol. 95, no. 1, pp. 1–13, 2005.

[10] C. Brekke and A. H.S.Solberg, "Forest fire monitoring using NOAA satellite AVHRR," *Canadian Journal of Forest Research*, vol. 16, no. 5, pp. 975–982, 1986.

[11] K. P. Ferentinos and N. Katsoulas and A. Tzounis and T. Bartzanas and C. Kittas, "Wireless sensor networks for greenhouse climate and plant condition assessment," *Biosystems Engineering*, vol. 153, pp. 70–81, 2017.

[12] A. Ruimy and P. G. Jarvis and D. D. Baldocchi and B. Saugier, "$CO_2$ Fluxes over Plant Canopies and Solar Radiation: A Review," *Advances in Ecological Research*, vol. 26, pp. 1–68, 1995.

[13] M. Bramberger and A. Doblander and A. Maier and B. Rinner and H. Schwabach, "Distributed embedded smart cameras for surveillance applications," *Computer*, vol. 39, no. 2, pp. 68–75, 2006.

[14] J. Palacios-Gasós, *Multi-Robot Persistent Coverage in Complex Environments.* PhD thesis, University of Zaragoza, 2018.

[15] J. Sánchez-Hermosilla and R. González and F. Rodríguez and J. G. Donaire , "Mechatronic Description of a Laser Autoguided Vehicle for Greenhouse Operations," *Sensors*, vol. 13, no. 1, pp. 769–784, 2013.

[16] M. Dunbabin and L. Marques, "Robots for Environmental Monitoring: Significant Advancements and Applications," *IEEE Robotics & Automation Magazine* , vol. 19, no. 1, pp. 24 – 39, 2012.

[17] R. N. Smith, J. Das, Y. Chao, D. A. Caron, B. H. Jones, and G. S. Sukhatme, "Cooperative multi-AUV tracking of phytoplankton blooms based on ocean model predictions," in *OCEANS'10 IEEE SYDNEY*, pp. 1–10, 2010.

[18] H. Choset, "Coverage for robotics-a survey of recent results," *Annals of Mathematics and Artificial Intelligence*, vol. 31, pp. 113–126, 2001.

[19] I. Maza and A. Ollero, "Multiple UAV cooperative searching operation using polygon area decomposition and efficient coverage algorithms," in *Distributed Autonomous Robotic Systems 6*, pp. 221–230, Springer, 2007.

[20] I. Rekleitis, A. Peng-New, E. Rankin, and H. Choset, "Efficient boustrophedon multi-robot coverage: an algorithmic approach," *Annals of Mathematics and Artificial Intelligence*, vol. 52, no. 2–4, pp. 109–142, 2008.

[21] I. Hussein and D. M. Stipanović, "Effective coverage control using dynamic sensor networks with flocking and guaranteed collision avoidance," in *Proceedings of the 45th IEEE Conference on Decision and Control*, pp. 3420–3425, 2006.

[22] D. Panagou, D. Stipanović, and P. Voulgaris, "Distributed dynamic coverage and avoidance control under anisotropic sensing," *IEEE Transactions on Control of Network Systems*, vol. 4, no. 4, pp. 850 – 862, 2017.

[23] C. Franco, D. Paesa, G. Lopez-Nicolas, C. Sagüés, and S. Llorente, "Hierarchical strategy for dynamic coverage," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 5341 – 5346, 2012.

[24] Y. Chevaleyre, "Theoretical analysis of the multi-agent patrolling problem," in *Proceedings IEEE/WIC/ACM International Conference on Intelligent Agent Technology*, 2004.

[25] F. Pasqualetti and A. Franchi and F. Bullo, "On Cooperative Patrolling: Optimal Trajectories, Complexity Analysis, and Approximation Algorithms," *IEEE Transactions on Robotics*, vol. 28, no. 3, pp. 592 – 606, 2012.

[26] N. Nigam and S. Bieniawski and I. Kroo and J. Vian, "Control of Multiple UAVs for Persistent Surveillance: Algorithm and Flight Test Results," *IEEE Transactions on Control Systems Technology* , vol. 20, no. 5, pp. 1236 – 1251, 2012.

[27] S. Smith and D. Rus, "Multi-robot monitoring in dynamic environments with guaranteed currency of observations," in *49th IEEE Conference on Decision and Control (CDC)*, pp. 514–521, 2010.

[28] S. Smith, M. Schwager, and D. Rus, "Persistent robotic tasks: Monitoring and sweeping in changing environments," *IEEE Transactions on Robotics*, vol. 28, no. 2, pp. 410 – 426, 2012.

[29] C. Song, L. Liu, G. Feng, Y. Wang, and Q. Gao, "Persistent awareness coverage control for mobile sensor networks," *Automatica*, vol. 49, no. 6, pp. 1867–1873, 2013.

[30] C. Song, L. Liu, G. Feng, and S. Xu, "Optimal control for multi-agent persistent monitoring," *Automatica*, vol. 50, no. 6, pp. 1663–1668, 2014.

[31] D. Soltero, M. Schwager, and D. Rus, "Generating informative paths for persistent sensing in unknown environments," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 2172–2179, 2012.

[32] P. Hokayem, D. Stipanović, and M. Spong, "On persistent coverage control," in *46th IEEE Conference on Decision and Control*, pp. 6130–6135, 2007.

[33] J. Palacios-Gasós, Z. Talebpour, E. Montijano, C. Sagüés, and A. Martinoli, "Optimal path planning and coverage control for multi-robot persistent coverage in environments with obstacles," in *IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1321–1327, 2017.

[34] J. M. Palacios-Gasós, E. Montijano, C. Sagüés, and S. Llorente, "Distributed coverage estimation and control for multirobot persistent tasks," *IEEE Transactions on Robotics*, vol. 32, no. 6, pp. 1444 – 1460, 2016.

[35] X. Lin and C. Cassandras, "An optimal control approach to the multi-agent persistent monitoring problem in two-dimensional spaces," *IEEE Transactions on Automatic Control*, vol. 60, no. 6, pp. 1659 – 1664, 2015.

[36] C. Franco, D. Stipanović, G. López-Nicolás, C. Sagüés, and S. Llorente, "Persistent coverage control for a team of agents with collision avoidance," *European Journal of Control*, vol. 22, no. 6, pp. 30–45, 2015.

[37] C. Franco, G. López-Nicolás, C. Sagüés, and S. Llorente, "Persistent coverage control with variable coverage action in multi-robot environment," in *52nd IEEE Conference on Decision and Control*, pp. 6055–6060, 2013.

[38] A. Mellone, G. Franzini, L. Pollini, and M. Innocenti, "Persistent coverage control for teams of heterogeneous agents," in *IEEE Conference on Decision and Control (CDC)*, pp. 2114– 2119, 2018.

[39] J. Palacios-Gasós, E. Montijano, C. Sagüés, and S. Llorente, "Multi-robot persistent coverage with optimal times," in *IEEE 55th Conference on Decision and Control (CDC)*, pp. 3511–3517, 2016.

[40] T. Lee, S. Baek, S. Oh, and Y. Choi, "Complete coverage algorithm based on linked smooth spiral paths for mobile robots," in *11th International Conference on Control Automation Robotics & Vision*, pp. 609–614, 2010.

[41] M. Conforti, G. Cornuéjols, and G. Zambelli, *Integer Programming*. Switzerland: Springer, 2014.

[42] B. Fleischmann, "A cutting plane procedure for the travelling salesman problem on road networks," *European Journal of Operational Research*, vol. 21, no. 3, pp. 307–317, 1985.

[43] Y. Pochet and L. Wolsey, "Solving multi-item lot-sizing problems using strong cutting planes," *Management Science*, vol. 37, no. 1, pp. 53–67, 1991.

[44] T. Achterberg, *Constraint Integer Programming*. PhD thesis, Technical University of Berlin, 2007.

[45] E. Klotz and A. Newman, "Practical guidelines for solving difficult mixed integer linear programs," *Surveys in Operations Research and Management Science*, vol. 18, no. 1-2, pp. 18–32, 2013.

[46] N. Kohl, J. Desrosiers, O. Madsen, M. Solomon, and F. Soumis, "2-path cuts for the vehicle routing problem with time windows," *Transportation Science*, vol. 33, no. 1, pp. 101–116, 1999.

[47] S. Manyam, S. Rasmussen, D. Casbeer, K. Kalyanam, and S. Manickam, "Multi-UAV routing for persistent intelligence surveillance & reconnaissance missions," in *International Conference on Unmanned Aircraft Systems (ICUAS)*, pp. 573–580, 2017.

[48] L. Grüne and J. Pannek, *Nonlinear Model Predictive Control: Theory and Algorithms*. London: Springer-Verlag, 2011.

[49] E. Kerrigan, *Robust Constraint Satisfaction: Invariant Sets and Predictive Control*. PhD thesis, Imperial College London, 2000.

[50] M. Lazar,V. Spinu, "Finite-step Terminal Ingredients for Stabilizing Model Predictive Control," *IFAC PapersOnLine*, vol. 48, no. 23, pp. 9–15, 2015.

[51] D. Mitchell, M. Corah, N. Chakraborty, K. Sycara, and N. Michael, "Multi-robot long-term persistent coverage with fuel constrained robots," in *IEEE International Conference on Robotics and Automation*, pp. 1093–1099, 2015.

[52] J.M. Palacios-Gasós and E. Montijano and C. Sagüés and S. Llorente, "Cooperative Periodic Coverage With Collision Avoidance," *IEEE Transactions on Control Systems Technology*, pp. 1–12, 2018. Early Access, DOI:10.1109/TCST.2018.2823278.

[53] J. Fargeas, B. Hyun, P. Kabamba, and A. Girard, "Persistent visitation under revisit constraints," in *International Conference on Unmanned Aircraft Systems (ICUAS)*, pp. 952–957, 2013.

[54] M. Santos and Y. Diaz-Mercado and M. Egerstedt, "Coverage Control for Multirobot Teams With Heterogeneous Sensing Capabilities," *IEEE Robotics and Automation Letters*, vol. 3, no. 2, pp. 919–925, 2018.

[55] L. Pimenta, V. Kumar, R. Mesquita, and G. Pereira, "Sensing and coverage for a network of heterogeneous robots," in *47th IEEE Conference on Decision and Control*, pp. 3947–3952, 2008.

[56] A. Richards and J. How, "A decentralized algorithm for robust constrained model predictive control," in *Proceedings of the American Control Conference*, pp. 4261–4266, 2004.

[57] M. Doan, T. Keviczky, and B. Schutter, "A hierarchical MPC approach with guaranteed feasibility for dynamically coupled linear systems," in *Intelligent Systems, Control and Automation: Science and Engineering* (J. Maestre and R. Negenborn, eds.), vol. 69, ch. Distributed Model Predictive Control Made Easy, pp. 393–406, Springer Dordrecht, 2014.

[58] M. Burger and Z. Su and B.De Schutter, "A node current-based 2-index formulation for the fixed-destination multi-depot travelling salesman problem," *European Journal of Operational Research*, vol. 265, no. 2, pp. 463–477, 2018.

# Glossary

## List of Acronyms

| | |
|---|---|
| **MILP** | Mixed Integer Linear Program |
| **LP** | Linear Program |
| **DPCP** | Dense Persistent Coverage Problem |
| **SPCP** | Sparse Persistent Coverage Problem |
| **COI** | Cell of Interest |
| **NMCL** | Normalized Mean Coverage Level over the Number of Cells and Optimization Horizon Length |

## List of Symbols

| | |
|---|---|
| $\beta$ | Weighting factor expressing the importance of penalizing subsequent coverage by different agents |
| $\delta_q^r(k)$ | Binary variable of Formulation I expressing whether agent $r$ has a pose $q$ at time step $k$ |
| $\mu_w^k$ | Binary Variable of Formulation II expressing whether $w$ is covered at time steps $k, k+1$ by different agents |
| $\mu_w^r(k)$ | Binary Variable of Formulation I expressing whether an agent $r' \neq r$ covers cell $w$ at $k+1$ after agent $r$ did at time $k$ |
| $I$ | Index Set of cells in the grid |
| $V_w'$ | Subset of $V_w$ including the pose transitions expressing the "turning at place" actions |
| $v_w$ | Maximum Time Spent between Subsequent Visits at a COI with index $w$ |
| $\alpha_w(k)$ | Mixed Variable of Formulation I equal to $Z(w, k-1)$ if $w$ is covered at $k$ or 0 otherwise |

| | |
|---|---|
| $c_w$ | Coordinates of the Center of cell $w$ |
| $Z(w,k)$ | Coverage Level of cell $w$ at time step $k$ |
| $Z_{r_j}(w,k)$ | The coverage level of cell $w$ at time step $k$ agent $r_j$ computes after making an action |
| $Z^{\text{com}}_{r_j}(w,k)$ | The idea agent $r_j$ has on the Coverage Level of cell $w$ at time step $k$ and sends to its neighbors |
| $N$ | Optimization Horizon |
| $\mathcal{E}$ | Set of edges expressing whether a pair of agents communicate |
| $\mathcal{G}$ | Directed Communication Graph associated to the team of agents (Distributed Method) |
| $n_w$ | Number of cells in the Grid |
| $\mathcal{N}_j$ | Set of indexes of the agents considered neighbors of agent $r_j$ |
| $\mathcal{M}_j$ | Set of indexes of the agents that consider agent $r_j$ as neighbor |
| $n_r$ | Number of Agents |
| $\bar{Z}$ | Reset value of $Z$ |
| $K$ | Set of agents' indexes |
| $M$ | Number of Sets in the terminal constraint set sequence |
| $S_u$ | The u$^{\text{th}}$ Terminal Constraint set of the sequence |
| $\theta^k_r$ | Heading of agent $r$ at time step $k$ |
| $\mathbf{u}_i(k)$ | Vector including the agents' poses at time step $k$ of the planning horizon with $i$ the absolute time index |
| $X^f_i$ | Terminal Constraint Set at absolute time step $i$ |
| $\underline{Z}$ | Desired Level of Coverage |
| $\mathbf{z}_i(k)$ | Vector including the Coverage level of the cells at time step $k$ of the planning horizon with $i$ the absolute time index |
| $A$ | Adjacency Matrix of graph $G$ |
| $C$ | Number of Columns in the Grid |
| $d_w$ | Coverage Decay factor of cell $w$ |
| $E$ | Set of edges of graph $G$ expressing the admissible state transitions |
| $F$ | Function expressing the dynamics of the vector $\mathbf{z}_i(k)$ |
| $G$ | The graph with nodes the agents' states and edges the admissible moves |
| $H$ | Function expressing the Number of Agents visiting cells covered at the previous time step by others |
| $k$ | Time Step |
| $L$ | Number of Rows in the Grid |
| $p^k_r$ | Position of agent $r$ at time step $k$ |
| $r$ | Agent's Index |
| $U$ | Subset of $V^{n_r}$ including the agents' poses per time step such that 1) each agent has a unique pose and 2) each cell hosts at most one agent |
| $V$ | Set of nodes of graph $G$ expressing agents' states |
| $V_w$ | Subset of $E$ including the admissible pose transitions that bring an agent $r \in K$ to the center of cell $w$ in a single time step |

| | |
|---|---|
| $w$ | Cell Index |
| $x^k_{qq'}$ | Binary Variable of Formulation II expressing whether an agent changes its pose from $q$ to $q'$ at time step $k$ |