

# Grid-Based Coverage Path Planning with Multiple UAVs for Search and Rescue Applications

by

Welri Botes



*Thesis presented in partial fulfilment of the requirements for  
the degree of Master of Engineering (Electrical and  
Electronic) in the Faculty of Engineering at Stellenbosch  
University*

Supervisor: Dr JAA Engelbrecht

Co-supervisor: Mr JC Schoeman

September 2021

# Declaration

By submitting this thesis electronically, I declare that the entirety of the work contained therein is my own, original work, that I am the sole author thereof (save to the extent explicitly otherwise stated), that reproduction and publication thereof by Stellenbosch University will not infringe any third party rights and that I have not previously in its entirety or in part submitted it for obtaining any qualification.

Date: ..... 2021/08/30 .....

Copyright © 2021 Stellenbosch University  
All rights reserved.

# Abstract

## Grid-Based Coverage Path Planning with Multiple UAVs for Search and Rescue Applications

W. Botes

*Department of Electrical and Electronic Engineering,  
University of Stellenbosch,  
Private Bag X1, Matieland 7602, South Africa.*

Thesis: MEng (EE)

September 2021

Vibrating a tillage tool is an effective way of reducing the draft force required to pull it through the soil. The degree of draft force reduction is dependent on the combination of operating parameters and soil conditions. It is thus necessary to optimize the vibratory implement for different conditions. Numerical modelling is more flexible than experimental testing and analytical models, and less costly than experimental testing. The Discrete Element Method (DEM) was specifically developed for granular materials such as soils and can be used to model a vibrating tillage tool for its design and optimization. The goal was thus to evaluate the ability of DEM to model a vibratory subsoiler and to investigate the cause of the draft force reduction. The DEM model was evaluated against data ...

# Uittreksel

**TODO: Insert Afrikaans Title Here**

*(“Grid-Based Coverage Path Planning with Multiple UAVs for Search and Rescue Applications”)*

W. Botes

*Departement Elektries en Elektroniese Ingenieurswese,  
Universiteit van Stellenbosch,  
Privaatsak X1, Matieland 7602, Suid Afrika.*

Tesis: MIng (EE)

September 2021

Om ‘n tand implement te vibreer is ‘n effektiewe manier om die trekkrag, wat benodig word om dit deur die grond te trek, te verminder. Die graad van krag vermindering is afhanklik van die kombinasie van werks parameters en die grond toestand. Dus is dit nodig om die vibrerende implement te optimeer vir verskillende omstandighede.

Numeriese modulering is meer buigsaam en goedkoper as eksperimentele opstellings en analitiese modelle. Die Diskrete Element Metode (DEM) was spesifiek vir korrelrige materiaal, soos grond, ontwikkel en kan gebruik word vir die modellering van ‘n vibrerende implement vir die ontwerp en optimering daarvan. Die doel was dus om die vermoë van DEM om ‘n vibrerende skeurploeg te modelleer, te evalueer, en om die oorsaak van die krag vermindering te ondersoek.

Die DEM model was geëvalueer teen data ...

# Acknowledgements

I would like to express my sincere gratitude to both my supervisors, Dr Japie Engelbrecht and Mr JC Schoeman. Without their guidance and expertise, this project would not have been possible. A further thanks to all those at the Electronic Systems Laboratory who offered advice and support throughout the past two years.

# Contents

<b>Declaration</b>	<b>i</b>
<b>Abstract</b>	<b>ii</b>
<b>Uittreksel</b>	<b>iii</b>
<b>Acknowledgements</b>	<b>iv</b>
<b>Contents</b>	<b>v</b>
<b>List of Figures</b>	<b>vii</b>
<b>List of Tables</b>	<b>viii</b>
<b>Nomenclature</b>	<b>ix</b>
<b>Acronyms</b>	<b>x</b>
<b>1 Introduction</b>	<b>1</b>
1.1 Background . . . . .	1
1.2 Research Aim . . . . .	3
1.3 Research Objectives . . . . .	3
1.4 Scope and Limitations . . . . .	4
1.5 System Overview . . . . .	4
1.6 Methodology . . . . .	4
1.7 Project Outline . . . . .	4
<b>2 Literature Review</b>	<b>5</b>
2.1 Motion Planning . . . . .	5
2.2 Coverage Path Planning . . . . .	7
2.3 Single Robot Coverage Path Planning . . . . .	7
2.4 Distributed Offline MCPP . . . . .	7
2.5 Non-Distributed Offline MCPP . . . . .	12
2.6 Online MCPP . . . . .	13
2.7 UAVs and Search and Rescue . . . . .	13

<b>3</b>	<b>Environment Representation</b>	<b>14</b>
3.1	Background . . . . .	14
3.2	Environment Discretisation . . . . .	16
<b>4</b>	<b>Divide Areas Algorithm</b>	<b>19</b>
4.1	Background . . . . .	19
4.2	Implementation . . . . .	20
<b>5</b>	<b>Subregion Coverage Technique</b>	<b>22</b>
5.1	Background . . . . .	22
<b>6</b>	<b>Dynamic Constraints of a UAV</b>	<b>23</b>
6.1	Background . . . . .	23
<b>7</b>	<b>Refuelling Protocol</b>	<b>24</b>
7.1	Background . . . . .	24
<b>8</b>	<b>Results</b>	<b>25</b>
<b>9</b>	<b>Conclusions and Future Work</b>	<b>26</b>
	<b>Appendices</b>	<b>27</b>
<b>A</b>	<b>Discrete Element Method Theory</b>	<b>28</b>
A.1	Ball elements . . . . .	28
	<b>List of References</b>	<b>29</b>

# List of Figures

1.1	DroneSAR Mobile Application Showing Coverage Plan[1] . . . . .	2
2.1	Simulation showing coverage of hexagonal partitions with back and forth motions with three robots. [2] . . . . .	8
2.2	Figures showing results for the Voronoi partitioning scheme for two different distance measures. [3] . . . . .	10
2.3	Figure showing the resulting area partition using the negotiation protocol. This example is for two robots and includes a no fly zone. [4] . . . . .	11
2.4	MSTC algorithm showing the paths for three robots on an environment grid. [5] . . . . .	12
3.1	Diagram showing relevant variables concerned with calculating the Field of View for a camera. . . . .	16
3.2	Figure showing the rectangular approximation for a human viewed from above for calculation of the GSD . . . . .	17
3.3	Diagrams showing cross track overlap for camera Field of View for different discretisation techniques. . . . .	18
4.1	. . . . .	21
A.1	Ball Element Parameters . . . . .	28



# List of Tables

# Nomenclature

## Constants

$$g = 9.81 \text{ m/s}^2$$

## Variables

$\phi$	Camera sensor to lens angle . . . . .	[deg]
$H$	Height from camera lens to ground . . . . .	[m]
$FOV$	Field of view (on ground) . . . . .	[m]
$AOV$	Camera angle of view . . . . .	[deg]
$f$	Camera focal length . . . . .	[mm]
$h_{len}$	Camera sensor height . . . . .	[mm]
$w_{len}$	Camera sensor width . . . . .	[mm]
$GSD$	Ground sampling distance . . . . .	[cm/px]

## Subscripts

$x$	The x dimension
$y$	The y dimension

# Acronyms

**AOV** Angle of View

**CPP** Coverage Path Planning

**DARP** Divide Areas Algorithm for Optimal Multi-Robot Coverage Path Planning

**FOV** Field of View

**GM-VPC** Geodesic-Manhattan Voronoi-Partition-Based Coverage

**GSD** Ground Sampling Distance

**MCP** Multi-Robot Coverage Path Planning

**MFC** Multi-Robot Forest Coverage

**MSTC** Multi-Robot Spanning Tree Coverage

**SAR** Search and Rescue

**STC** Spanning Tree Coverage

**UAV** Unmanned Aerial Vehicle

**UGV** Unmanned Ground Vehicle

# Chapter 1

## Introduction

### 1.1 Background

Unmanned Aerial Vehicles (UAVs) are a technology that have gained popularity in various applications [6]. Originally, UAVs required a ground pilot to manoeuvre them, but are becoming an increasingly automated technology. Applications where UAV automation has been used include, but are not limited to, structure inspections[7], smart farming[8], disaster management[9], power line inspections[10], surveillance[11] and wildfire tracking[12].

Most of the research mentioned was done on the premise of using multi-rotor UAVs, quad-rotor vehicles in particular. It is important to note that the term UAVs also encompasses other aircraft types, like single rotor and fixed wing UAVs. Hybrids also exist that contain both rotary-wing and fixed-wing components[6].

Using UAVs poses a considerable advantage in applications like the ones mentioned when compared to Unmanned Ground Vehicles (UGVs). Their capacity to fly over landscapes and around three dimensional structures makes their potential applications increase substantially. Relatively high altitude flying is a key reason why they are well suited to the application suggested in this paper, which is automated coverage path planning for Search and Rescue (SAR) missions.

According to [13], the purpose of motion planning algorithms are determined by the field of research. In control theory, it refers to algorithms that are designed to find trajectories for agents within a non-linear system. This contrasts with the usual focus of control theory on feedback and optimization because the trajectories are usually computed using open-loop methods. Motion planning takes on a subtly different meaning in the world of robotics or artificial intelligence, but the control theory definition is the one that will be used throughout this text.

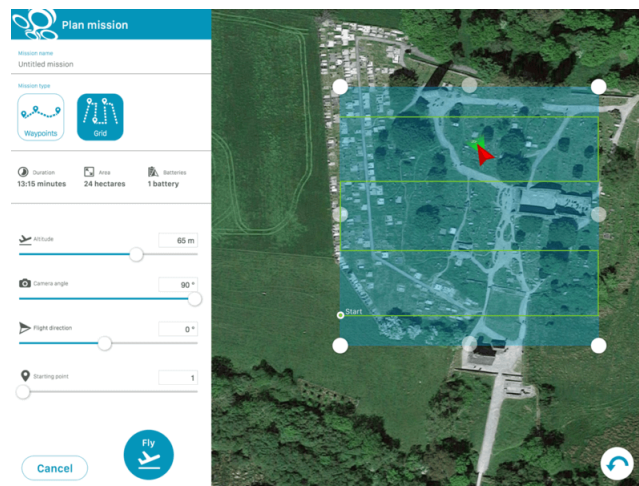
Coverage Path Planning (CPP) is a variant of the general motion planning problem. Originally, motion planning algorithms were predominantly used to

find solutions for start-goal problems[14]. This could mean getting an agent, a UAV for example, from some starting position to some goal position in an environment[15]. Coverage path planning is different from start-goal path planning in that it tries to determine a path for an agent to pass over all points in an environment[14]. It can be used with ground vehicles, for example, to automate field machines for smart farming[16]. Further examples include vacuum cleaning robots, spray painting robots[17], window cleaning robots[18] and automated lawn mowers[19]. For underwater vehicles it can be used for the inspection of difficult to reach underwater structures[20].

Furthermore, there have been developments in the use of UAVs to perform automated search and rescue operations using coverage path planning. Perhaps the most notable example is a project by DroneSAR where they use DJI drones to perform search and rescue tasks. Their implementation includes a mobile application that allows the user to designate a search area manually[1]. Search and rescue operations often span large areas and UAVs fly above most ground obstacles. Therefore, it is realistic that they assume the search environment can be mapped prior to the search operation[21].

DroneSAR uses one drone per search and rescue operation. Once the environment has been designated, the drone performs a back and forth manoeuvre across the area to achieve coverage. The search operation can be halted if the imaging system detects a possible target in the area. The drone can be switched to manual flight mode for closer inspection and the co-ordinates of the target, for example a person in turmoil, can be sent to the search and rescue team. Their system also allows for the manual assignment of way-points to a flight path to bypass the back and forth manoeuvre.[22]

In Figure 1.1, a screenshot of their mobile application is shown. It illustrates the back and forth manoeuvre used to achieve coverage of the designated area.



**Figure 1.1:** DroneSAR Mobile Application Showing Coverage Plan[1]

This paper also looks at coverage path planning for search and rescue, but suggests a multiple UAV approach to the problem. According to [1], when looking for a victim in one square kilometre on land, it takes a five-person rescue team two hours on average to find the victim. DroneSAR found that their drone could do the same job in under 20 minutes. Adding multiple UAVs to cover an area could reduce this time even more, since it would mean more area is covered per unit time. This is important because in a search and rescue operation, time is always of the essence.

This paper also focuses on a grid-based approach to the coverage problem. According to the taxonomy represented in [14], this is referred to as an approximate method. Although one can achieve complete coverage of the grid, the grid itself is not an exact representation of the environment. It does, however, greatly simplify the process of allocating areas to different UAVs, which is a key process for multiple UAV coverage. Physical implementation of this method will not be addressed as part of the scope for this paper.

## 1.2 Research Aim

The main goal of this research is to develop a coverage path planning algorithm for multiple unmanned aerial vehicles (UAVs) to search an area. The research is intended to be applicable in search and rescue operations using unmanned aerial vehicles to assist.

## 1.3 Research Objectives

Based on the main aim of this project set out in Section 1.2, a set of research objectives were formulated. These are intended to give a clearer picture of the main research goals and scope of the project. Scope and limitations are further discussed in Section 1.4 and the methodology used to achieve these objectives are detailed in Section 1.6. The research objectives are as follows:

1. Develop a coverage path planning algorithm for an environment that is known a priori and contains static obstacles.
2. Ensure that the final algorithm is an approximately complete solution.
3. Incorporate into the algorithm's functionality an ability to have a changing number of starting UAVs that have random initial positions.
4. Evaluate the algorithm's performance in both randomly generated and mapped, real world environments to ascertain whether or not it is suitable for search and rescue operations.

## 1.4 Scope and Limitations

## 1.5 System Overview

## 1.6 Methodology

## 1.7 Project Outline

# Chapter 2

## Literature Review

As mentioned in Section 1.1, Coverage Path Planning is a subset of the general motion planning problem. This chapter will therefore begin with a brief overview of motion planning before discussing CPP as a whole. Literature pertaining to CPP is then discussed in several sections. Firstly, it is addressed in the context of the single robot CPP problem. Several techniques used to achieve coverage using only one robot are summarized. Following this are three sections dedicated to Multi-Robot Coverage Path Planning (MCP) problem. The first two cover distributed and non-distributed offline MCP respectively. This is followed by a section presenting some online MCP implementations. Many of the implementations were done with some application in mind, but the last section covers UAVs and how they have been applied to SAR operations in particular.

### 2.1 Motion Planning

Perhaps one of the most noteworthy items of literature presented on motion planning is [13]. In this book, a differentiation is made between motion planning and trajectory planning. Motion planning, by their definition, refers to a series of translations and/or rotations required to get an agent from one point to another within some environment. Trajectory planning would then take this plan and find a strategy to execute it within the dynamic constraints of the agent. *Agent* is a term from the field of artificial intelligence and is interchangeable with *robot* or *decision maker*.

The book refers to planning algorithms as a strategy that is used by one or more decision makers to move from some starting state to some goal state within some context. State is a very general term and can refer to a number of different implementations. An example would be a robotic arm, the decision maker, that needs to move from some starting orientation, or state, to a goal orientation. Another common scenario involves simple start-goal path planning. An example would be a car that needs to travel from one point in a



city to another. Here, the car is the decision maker (or agent) and a strategy needs to be found to move the car along some valid path to get from one point to another within the context of the city. Generally, it is favorable to try and find the shortest viable path in this kind of scenario.

In most applications, the environment wherein the agent exists is important to determine a valid strategy to reach the goal state. For the robotic arm scenario, it is important that the location of any obstructions within reach of the arm are known so they can be avoided. This kind of collision avoidance is important for the car as well. It must know where valid paths lie within the city to avoid collisions and stay along the road. The environment is generally either represented as discrete or continuous, depending on the planning algorithm.

Information about the environment can be available a priori, in which case it is referred to as offline planning. This means that all information about the environment and its dynamics are known before execution of the planning algorithm. For the car, this would mean that before executing the path, all the obstacles in the environment are known. Dynamic obstacles e.g. other moving cars, would have their future positions and velocities mapped out beforehand as well. The planner can thus take these into account during the planning phase, prior to execution of the plan. Online planners also exist, where the environment is an unknown, or is only partially known a priori. This usually means the agent must have onboard sensors to map out its environment as it moves through it. This is often applicable in scenarios where actions by the agent change the environment or when there is a moving target with an unknown trajectory.

Motion planning can be extended to multiple agents in a single environment. Optimizing paths in these scenarios can be quite challenging, because agents must now not only avoid collisions within the environment, but also with one-another, while trying to achieve a certain goal. In the context of path planning with UAVs, the nature of the goal makes the problem fall into different categories, according to the authors of [23].

In a start-goal problem, if the goal location is the same for all UAVs, it is referred to as a rendezvous task. If they all have different goal locations, it is an allocation task. And lastly, if the goal is not to move from a starting position to a goal, but instead to cover every point in an environment, it is called a coverage task. The initial positions of the UAVs can also be classified into aggregated or dispersed. Respectively, these refer to all the UAVs starting at the same position or starting at different locations within the environment. Once an algorithm is designed to apply to a specific scenario, its effectiveness can be measured in a number of ways. Firstly, completeness refers to an algorithm that will always find the solution for a scenario, if one is possible[13]. Another important measure is efficiency, which can be improved by making an algorithm run faster. Lastly is robustness, which indicates how resistant an algorithm is to failure [5].

## 2.2 Coverage Path Planning

Coverage path planning algorithms can usually be classified into online or offline. These classifications refer to the amount of information available about the environment that needs to be covered [6]. Online planning is a dynamic approach wherein information about the environment is gathered during deployment and is not known a priori. Offline planning generally means that the environment is mapped out in full a priori. Robot trajectories for area coverage are planned prior to execution. Hybrid applications exist, wherein some information is known beforehand, but changes in the environment, for example moving obstacles or targets, can be collected and accounted for dynamically during execution [24].

## 2.3 Single Robot Coverage Path Planning

This section is divided into subsections based on....

### 2.3.1 Exact Methods

An early form of coverage path planning can be found in [14]. This is known as exact cellular decomposition. This was originally used for single robot applications where an area is divided into polygons, called cells, which are then searched sequentially by a single robot using simple back and forth motions. This is known as an exact solution because the area is still treated as continuous space and is not discretised as with approximate methods. Therefore, it is possible to get complete coverage of an area. The boustrophedon and trapezoidal decomposition are well known exact cellular techniques.

### 2.3.2 A\* Path Coverage Path Planning

### 2.3.3 Spanning Tree Coverage Path Planning

### 2.3.4 Wavefront Planning

### 2.3.5 Artificial Intelligence Based Planning

## 2.4 Distributed Offline MCPP

A well established offline coverage path planning approach involves the divide areas technique. This partitions an area into regions for individual robots to cover. Each robot should then be able to cover its area using one of the individual area coverage techniques mentioned in Section 2.3.

### 2.4.1 Hexagonal Segmentation

A notable distributed approach uses regular hexagons to segment the area of interest [2]. This implementation is reminiscent of the exact methods often used for single robot coverage path planning mentioned in Section 2.3.1. Exact methods generally divide an area into arbitrarily sized polygons called cells. The robot moves between these cells and covers them using simple motions. In the multi-robot scenario, the area is still divided into cells, but they need to be distributed between the robots evenly for searching. The ideal situation is to assign equal sized areas to each robot so that their path lengths are similar and they can complete their paths at roughly the same time.

Hexagonal cells make it easier to assign cells to robots, seen as they are all of the same size. Hexagons are clustered using the K-means algorithm to ensure a similar number of cells are assigned to each robot. The seeds are synonymous with the robots, therefore once the seed locations are finalized for even cell distribution, the robot initial positions are established. The robots cannot start from any random location, which can be undesirable.

The hexagons that are assigned to a given robot are contiguous and form a sub-region that is then covered using simple manoeuvres. Back and forth motions are generally quite popular. Static obstacles are considered in this implementation, but the smallest obstacle resolution is the size of a hexagon which may not be very representative of the environment. Figure 2.1, taken directly from their paper, illustrates the back and forth manoeuvres used to cover the hexagonal partitions. Black hexagons represent no-fly zones and/or static obstacles. The dark red, green and blue regions represent the sub-regions as they are assigned to the respective robots for coverage.



**Figure 2.1:** Simulation showing coverage of hexagonal partitions with back and forth motions with three robots. [2]

## 2.4.2 Voronoi Partitioning

In the mathematics field, there are methods of area division to divide a polygon into a number of equal area polygons [25]. Another relevant method that also stems from the field of mathematics, is the Voronoi partition. This assigns regions within an area to seeds based on distance. The idea is that a region assigned to a seed represents all the points where the distance to that seed is shorter than to any other seed.

If the Voronoi partition is applied to the MCPP problem, the seeds become synonymous with robots. This partition works for any number of robots at any starting positions, but unless they are evenly spaced, the areas will not have equal sizes. Distances in these scenarios are usually Euclidean and the boundaries between areas represent the position where the distances from two seeds are equal.

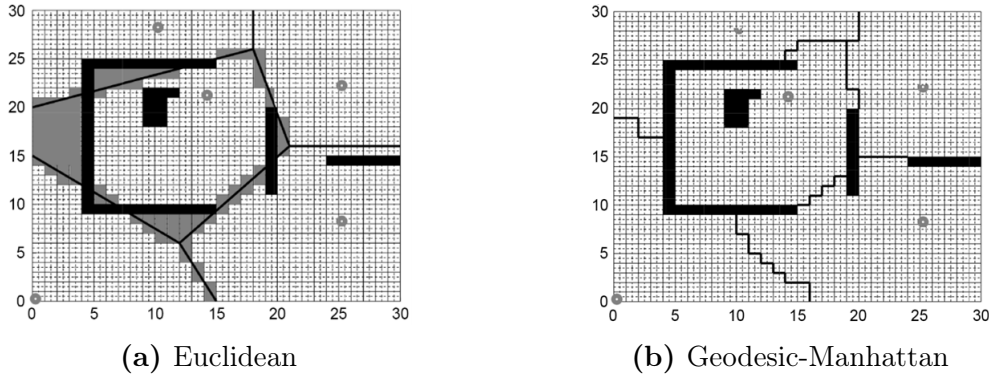
The authors of [3] implement MCPP using Voronoi partitions in discrete space with static obstacles. They use square discretisation of the area and compare several different methods. They investigate geodesic-Manhattan-, Manhattan-, geodesic- and Euclidean-distance-based Voronoi partitions.

The Euclidean-based technique results in what the authors term "non-contiguous sub-regions". This means that cells that are part of a sub-region are not accessible by the robot assigned to them, due to obstacles within that sub-region. They solve this problem by using geodesic distance. This uses Euclidean measurements, but instead of a straight line distance between two cells, it calculates the distance using a collision free path between the two cells.

Another problem arises, due to their use of discrete space. This is that when using Euclidean distances, some cells were partially in two sub-regions instead of fully in one or the other. Their solution is to use Manhattan distances. Ultimately they claim to have solved these problems by using geodesic-Manhattan-based distances to generate the partition. And thus they coined the term Geodesic-Manhattan Voronoi-Partition-Based Coverage (GM-VPC)).

Figure 2.2 shows figures from the paper that show the results of an area division using different distance measures with a Voronoi partition. In both figures the black blocks represent obstacles, the round dots are the robot starting positions and the black lines over the grid represent the partition boundaries.

Figure 2.2a shows the results using Euclidean distances. Here, the grey blocks are areas that would not be covered. This is clearly remedied using the GM-VPC technique in Figure 2.2b. They implemented two different versions of GM-VPC, which utilize respectively an exact and an approximate individual area search technique. They implemented a boustrophedon coverage plan for the exact solution and a spanning tree for the approximate version. Both of these performed better when using geodesic-Manhattan distances.



**Figure 2.2:** Figures showing results for the Voronoi partitioning scheme for two different distance measures. [3]

### 2.4.3 Negotiation Protocol

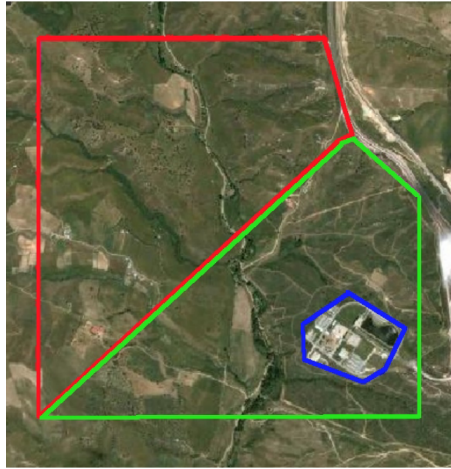
A negotiation or bargaining protocol refers to a process involving task partitioning. In the context of area division for CPP, the task represents the area to be divided [4]. The authors of [4] present a negotiation model based on Rubinstein’s alternate-offers protocol, for the purpose of area division. The focus of their implementation was to develop a distributed algorithm capable of considering robot capabilities. This means that the robots wouldn’t have to be homogeneous and can have different flight-time capabilities, manoeuvrability, on-board equipment and so forth [26].

They implemented their algorithm and found that it can achieve near optimum results. It tries to maximise the size of each robot’s subdivision of the area (based on its capabilities), while also minimising sub-area overlap. The algorithm also works to avoid static obstacles or no fly zones that are present in the area. Moreover, they proved that it could be applied in a situation where re-planning may be necessary.

A more complete implementation of the algorithm including an individual area search technique was also developed and tested [26]. In this implementation they use a wavefront planner for the individual area coverage path generation. This requires discretisation of the area into cells. In their case, they used rectangles whose size was determined by on-board camera Field of View (FOV). In order for the polygons generated by the negotiation protocol to work effectively, they use a method called Bresenham’s line algorithm to approximate the lines that divide the areas in discrete space, so that they pass through the centroids of cells.

The area division achieved sometimes produces non-convex shapes, which the wavefront planner can handle effectively. Their implementation also minimizes energy consumption by minimizing the number of turns and not allowing backtracking. They also have the ability to specify the initial take-off positions of the robots. Distance from the specified take-off point to the starting point for sub-area coverage are considered in the sub-task negotiations. The authors

also mention being able to specify robot landing positions pre-emptively. One visible drawback in their implementation is that their coverage appears incomplete. The boundaries between areas pass through waypoints (cell centroids), that effectively get excluded from the coverage algorithm and are not covered. Using an exact method to search the individual areas could produce better results. Changing the boundaries to lie on the edges of cells rather than passing through their centroids could also make a difference.



**Figure 2.3:** Figure showing the resulting area partition using the negotiation protocol. This example is for two robots and includes a no fly zone. [4]

#### 2.4.4 MSTC

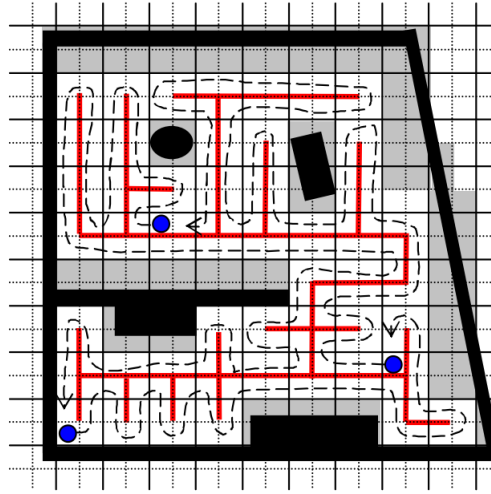
Multi-Robot Spanning Tree Coverage (MSTC) is a variant of single robot Spanning Tree Coverage (STC) as presented in Section 2.3.3. The authors of [5] designed the first variants of MSTC. The two variations they suggest are one that allows for backtracking and one that does not. Both variations still utilize a single spanning tree, but simply circumnavigate the tree with multiple robots instead of a single one.

They place a lot of emphasis on robustness and efficiency, in addition to completeness. They demonstrate an algorithm that segments the path around a spanning tree to evenly distribute it among robots. This distribution of robots is however, unrealistic. Their method becomes incredibly inefficient when robots are clustered closely together. This is because a robot simply navigates the path until it reaches the initial position of the next robot on the path.

Figure 2.4 shows the paths that are generated when the robots are evenly distributed along the path that circumnavigates the tree. Blue dots represent the robot initial positions and the spanning tree is shown in red. The second method they suggest remedies this somewhat. It allows for backtracking and

improves the efficiency.

The ideal situation is that all the robots have near equal path lengths, provided they are homogeneous robots. This is not guaranteed with this algorithm when the robots have random starting positions, but allowing for backtracking can improve the results and allow the coverage to be completed in a shorter amount of time.



**Figure 2.4:** MSTC algorithm showing the paths for three robots on an environment grid. [5]

### 2.4.5 DARP

## 2.5 Non-Distributed Offline MCPP

### 2.5.1 MFC

Multi-Robot Forest Coverage (MFC) is a method that was developed by the authors in [27] to improve upon the MSTC method mentioned in Section 2.4.4. Their intent was to construct a tree with the consideration that it will be divided afterwards, unlike what MSTC does. It allows for robot path overlap, which means there is redundant coverage and collision avoidance would need to be considered. However it can handle unique scenarios, where backtracking is unavoidable, quite well.

Their implementation is based on an algorithm presented in [28]. This is an approximation algorithm and they specifically looked at the rooted tree cover scenario. The roots represent the robot initial positions and then a tree is generated for each robot, using the objective to minimize the weight of the maximum weight tree. These trees are each circumnavigated by their robot

(root) to cover the area.

Based on the simulations they ran with MSTC and MFC, they found that MFC generated closer to optimum results and generally achieved coverage in a shorter amount of time. Because there is path overlap in certain scenarios, MFC is not a truly distributed method.

### **2.5.2 Sampling-Based**

### **2.5.3 Artificial-Intelligence-Based**

## **2.6 Online MCPP**

## **2.7 UAVs and Search and Rescue**



# Chapter 3

## Environment Representation

This section covers the environment representation.

### 3.1 Background

The goal of this paper is to develop a coverage path planning algorithm applicable in a Search and Rescue situation. It should be implementable using UAVs, which can execute three dimensional movements. In this paper however, it is assumed that altitude changes are not necessary to search an area. Assuming that there is some form of camera on-board the UAV, a constant altitude will be an advantage. It means that the Ground Sampling Distance (GSD) will remain constant, which is a widely accepted measure of camera accuracy. Assuming a camera is on a UAV pointing down towards the ground, GSD is the distance on the ground as represented by the width of one pixel [29].

Without altitude changes, UAV motions can be represented in two-dimensional space, which greatly simplifies the problem. For coverage path planning, it is important to have some demarcated two-dimensional region that requires coverage. The identified region, or environment, needs to be represented in either a discrete or continuous manner.

In the context of SAR, complete coverage is very important. It will ensure that every possible point in the environment map is covered. In this scenario, it implies that the camera will have viewed all points on the map. Achieving completeness is by no means trivial, but is more achievable in complex environments when using a discrete approach.

Discretising the environment can be done in a few different ways. Assuming a generic UAV with some thermal or visual camera on-board, one has a few options. One can discretise the area based on the UAV size, but a more common practice is to base it off the tool size. In this case, that would be the Field of View (FOV) of the on-board camera. This also makes the process of complete coverage easier, because if the camera is guaranteed to see the entirety of each

discrete cell, it is a complete algorithm so long as each cell is visited.

Therefore, to discretise the environment, the Field of View needs to be calculated. The camera specifications and UAV altitude will be the determining factors to calculate the FOV. The type of camera and the flight altitude are design decisions and will depend on the GSD necessary to realistically be able to locate the target in a Search and Rescue operation.

The diagram in Figure 3.1 shows all the relevant variables needed to calculate the Field of View along one dimension ( $FOV_x$ ). A similar diagram can be used to calculate the Field of View along the other dimension ( $FOV_y$ ). The only difference would be the sensor size variable, which changes from the sensor width ( $w_{len}$ ) to the sensor height ( $h_{len}$ ). The other variables include the focal length of the camera ( $f$ ), the height of the lens above ground ( $H$ ) and the camera's angle of view ( $AOV$ ). Lastly, there is the variable  $\phi$  which is an angle created due to the sensor being slightly smaller than the diameter of the cone of light projected onto it.

The resulting Field of View will be a rectangle of the same aspect ratio of the camera sensor, provided the camera is pointing directly down and the ground is level. For this application, it is assumed the camera is always pointing downwards. This can be accomplished when the UAV banks by placing the camera on a gimble. The assumption that the ground is level is not entirely reasonable, for example, in a mountainous region. This can be addressed by adding overlap between images to add some redundant coverage, which will be added in Chapter 7. Doing a topographical inspection is beyond the scope of this project and will not be addressed in more detail.

The calculations required to do a discretisation based on the camera Field of View will now be discussed. The first equation that is required is the calculation of the angle  $\phi$  which makes use of the small triangle:

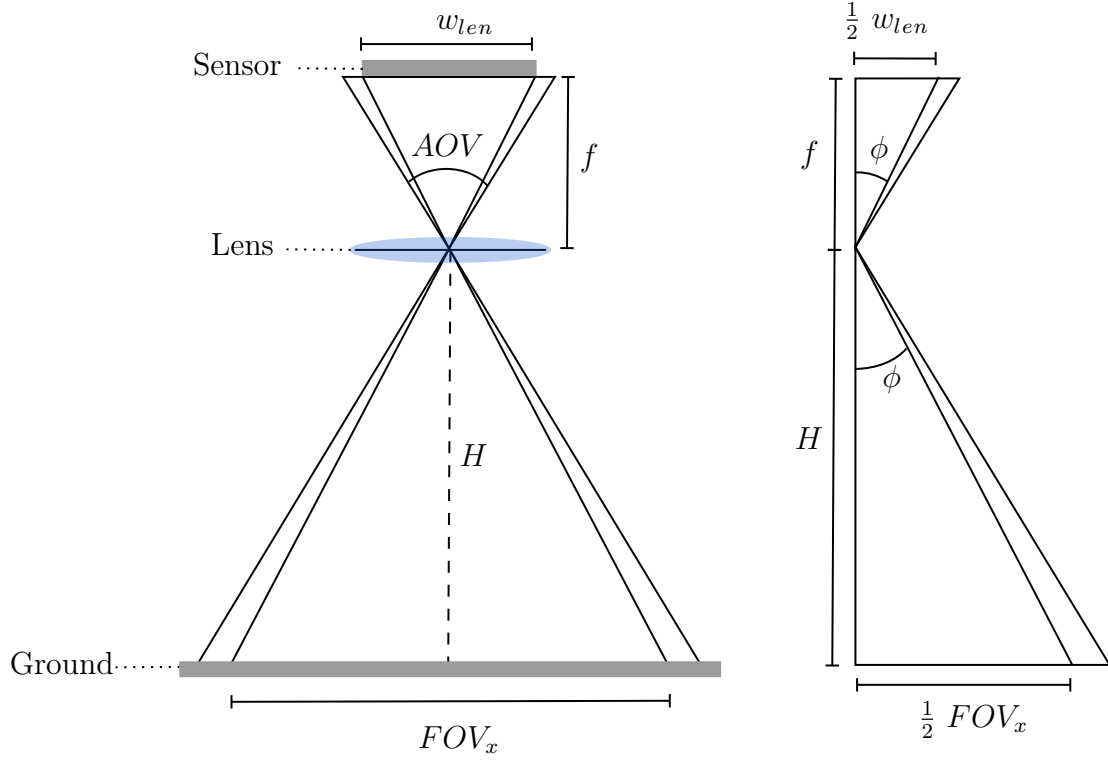
$$\begin{aligned}\tan \phi &= \frac{\frac{1}{2}w_{len}}{f} \\ \phi &= \tan^{-1} \left( \frac{w_{len}}{2f} \right)\end{aligned}\tag{3.1}$$

Now that  $\phi$  is known, the bigger triangle is used to calculate  $FOV_x$ :

$$\begin{aligned}\frac{FOV_x}{2} &= H \times \tan \phi \\ FOV_x &= 2H \times \tan \left( \tan^{-1} \left( \frac{w_{len}}{2f} \right) \right) \\ FOV_x &= H \times \frac{w_{len}}{f}\end{aligned}\tag{3.2}$$

Similarly,  $FOV_y$  can be calculated using the other sensor size dimension  $h_{len}$ :

$$FOV_y = H \times \frac{h_{len}}{f}\tag{3.3}$$



**Figure 3.1:** Diagram showing relevant variables concerned with calculating the Field of View for a camera.

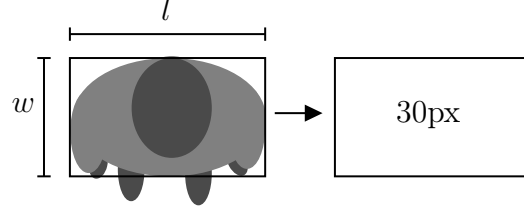
Both  $FOV_x$  and  $FOV_y$  will have the same units as  $H$ , which is metres. The resolution of the camera is the number of pixels along the image width ( $px_w$ ) multiplied with the number of pixels along its height ( $px_h$ ). These can be used to calculate the  $GSD$  by dividing either  $FOV$  by the pixel value associated with that dimension:

$$\begin{aligned}
 GSD &= \frac{100FOV_x}{px_w} \\
 GSD &= \frac{100H \times w_{len}}{f \times px_w}
 \end{aligned} \tag{3.4}$$

## 3.2 Environment Discretisation

If the size of the environment discretisation is set equal to the rectangular camera field of view, and the type of camera used is known, then a desired  $GSD$  can be used to decide on an appropriate flying height. Choosing a  $GSD$  would depend on the application, seen as it represents the level of detail that can potentially be detected in an image.

Taking the conservative approach, the assumption is that one is looking for a human being standing upright, viewed from above. To get a good estimate of the space occupied by a human in this orientation, one needs anthropometric



**Figure 3.2:** Figure showing the rectangular approximation for a human viewed from above for calculation of the GSD

data. A survey was done in Europe for people between the ages of 18 and 60 [30]. Among other measurements, they measured chest depth ( $w$ ) and elbow-to-elbow length ( $l$ ). These dimensions represent those of an upright human from above and can be seen in Figure 3.2.

To calculate GSD, a minimum number of pixels needed to make a human visible must be chosen. There is an article wherein they developed an image processing algorithm for human detection in a Search and Rescue scenario [31]. In this article they made the decision to put a 30 pixel requirement on human detection. Figure 3.2 shows the rectangular approximation for a human that the 30 pixels should represent.

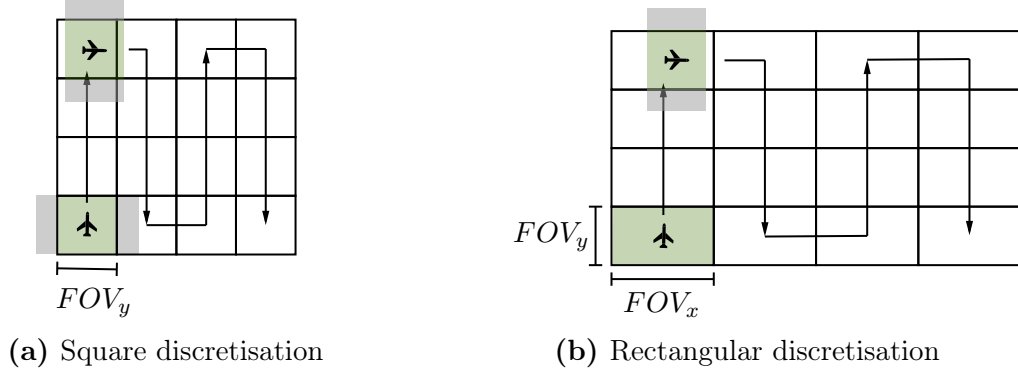
Using the lower percentile measurements of 170mm chest depth and 390mm elbow-to-elbow length along with the 30 pixel requirement, one gets a GSD of roughly 4.7cm/px. For a child, these values could be even smaller. Therefore, 4cm/px will be used to calculate an appropriate flying height, which is a reasonable value considering that most aerial surveys operate at a GSD of less than 5cm/px [29].

To show how the GSD requirement can be used, a series of cameras are chosen to demonstrate how the discretisation could be determined using the camera specifications and the maximum allowable GSD. Before showing the calculations applied to a specific scenario, they are shown in a more general sense. Firstly, one calculates the maximum allowable height:

$$H_{max} = \frac{GSD_{max} \times f \times px_w}{100w_{len}} \quad (3.5)$$

It is desirable to fly as high as possible, because this decreases the coverage time by increasing the camera Field of View. Therefore, the assumption is that the height chosen would be the maximum, provided this is within the capabilities of the UAV. Using Equations 3.2 and 3.3, one can then calculate the camera FOV at the height chosen.

One can now discretise the environment based on this field of view. One option is to do a square discretisation. One would make the squares have sides equal to the smaller FOV dimension, which is  $FOV_y$ . An example of this discretisation can be seen in Figure 3.3a. This technique will always have cross-track overlap. This means that there will be a percentage of redundant coverage. If a camera has a square FOV then the overlap goes away, but this is uncommon. They



**Figure 3.3:** Diagrams showing cross track overlap for camera Field of View for different discretisation techniques.

tend to have a 4:3 or 3:2 aspect ratio. There is the option of making the squares have side lengths equal to  $FOV_x$ , but then complete coverage would not be achievable by simply following the square centroids. This adds a layer of complexity unnecessarily. Setting the square side length equal to  $l$ , one can calculate the cross-track overlap using the following equation:

$$\begin{aligned} \text{Provided,} \quad & l \leq FOV_y \\ \text{Then,} \quad & \%Overlap = \frac{l(FOV_x - l)}{l^2} \times 100 \end{aligned} \quad (3.6)$$

Figure 3.3b shows an alternative technique where the environment is divided into rectangles. With this there is no overlap when moving in the  $y$  direction, but there is when moving in the  $x$  direction. An environment would be covered faster with this technique provided the  $y$  direction is favoured during flight, but no overlap in the  $y$  direction is risky. In this scenario, the cross-track overlap over the span of one rectangle when moving in the  $y$  direction is zero and for the  $x$  direction it can be calculated as follows:

$$\%Overlap = \frac{FOV_x(FOV_x - FOV_y)}{FOV_x - FOV_y} \times 100 \quad (3.7)$$

Both techniques assume sharp 90 degree turns are possible. Some UAVs can make sharp turns, such as multi-rotors, but this often requires deceleration to a hover, which significantly slows coverage time by lowering the average velocity during coverage. Chapter 6 covers a proposed solution for a constant speed scenario. This is considerably more favourable in a scenario where a fixed-wing UAV is used instead of a multi-rotor. Fixed-wing craft are desirable for SAR due to their high endurance. They can generally cover larger areas before needing to refuel.

# Chapter 4

## Divide Areas Algorithm

### 4.1 Background

The research aim in section 1.2 refers to the development of a coverage path planning algorithm using multiple UAVs. Coverage path planning is often linked to applications such as surveying, mapping and searching, because it involves planning a UAVs path so as to cover all points in an environment. The idea is then that one could speed up the time taken to cover the environment mentioned by using several UAVs searching in tandem.

Before one can begin covering an area, the points within this area that must be searched require identifying. With coverage path planning, grid-based methods are quite popular because they make it easy to divide an area into searchable cells. Grid-based coverage path planning can be implemented using a number of different methods.

Naturally, achieving the most optimal solution possible would be most desirable. The authors of [32] propose a set of requirements for optimal coverage path planning using a grid-based approach. These fundamental conditions, as they call them, are listed below.

1. Every cell in the environment, that is not classified as an object, must be covered. This is known as complete coverage.
2. Each cell in the environment must only be searched once, and only by one of the robots. This is known as the non-backtracking requirement.
3. Each robot should have as close to an equal amount of cells as possible assigned to it for searching. Their sets of cells should be of roughly the same size.
4. The sets of cells assigned to each robot should be a connected sub-region. This means that when generating a path to search the cells within its set, a robot would not need to traverse that of another to search it's own sub-region.

5. The initial position of each robot should be contained within the set of cells assigned to it. This means that a robot would not need to travel to reach its sub-region for searching.

## 4.2 Implementation

### 4.2.1 Comparison of Distance Measures

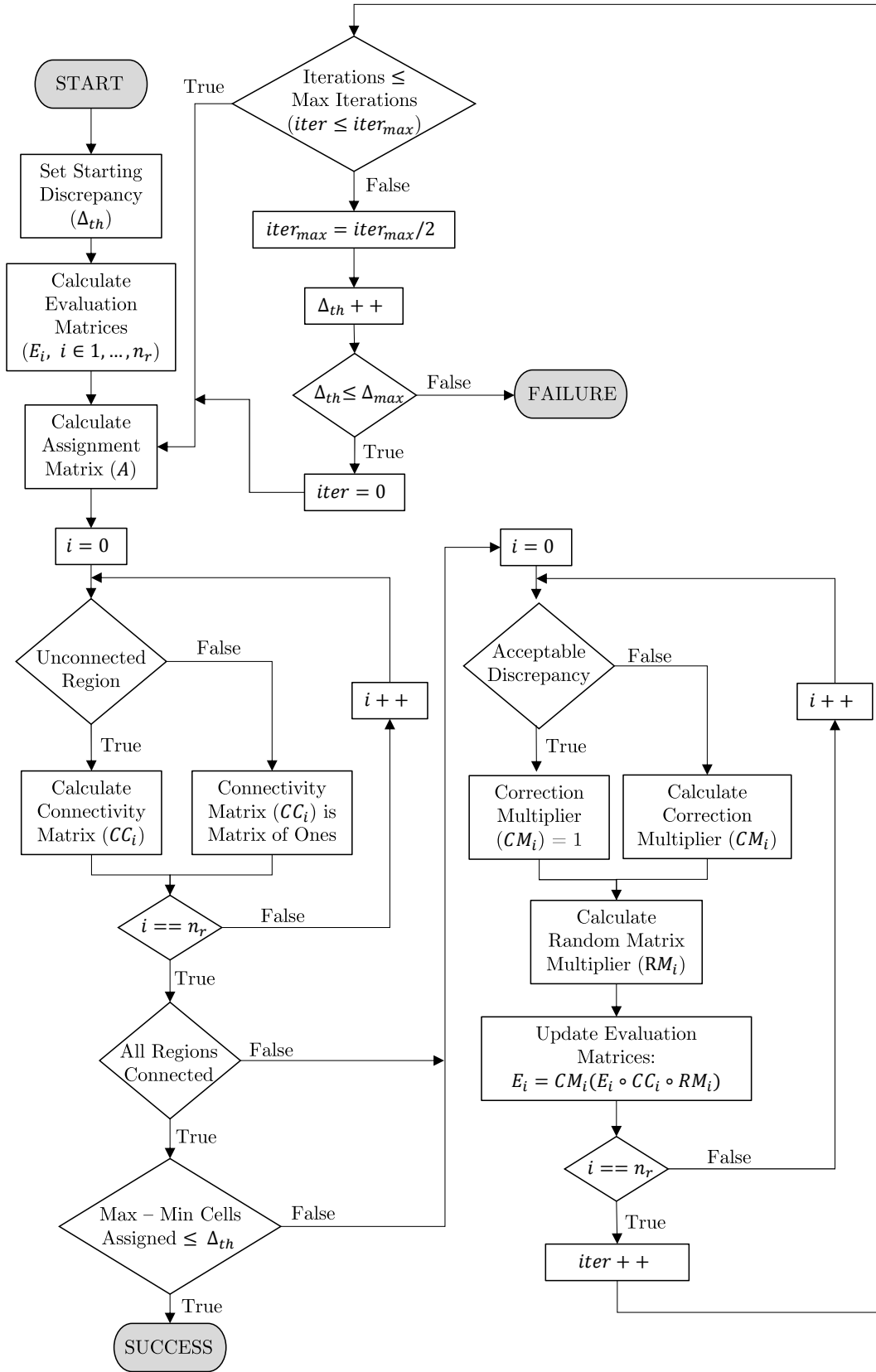


Figure 4.1



## Chapter 5

# Subregion Coverage Technique

### 5.1 Background

## Chapter 6

# Dynamic Constraints of a UAV

### 6.1 Background

# Chapter 7

## Refuelling Protocol

### 7.1 Background

## Chapter 8

## Results

## Chapter 9

# Conclusions and Future Work

# Appendices

# Appendix A

## Discrete Element Method Theory

### A.1 Ball elements

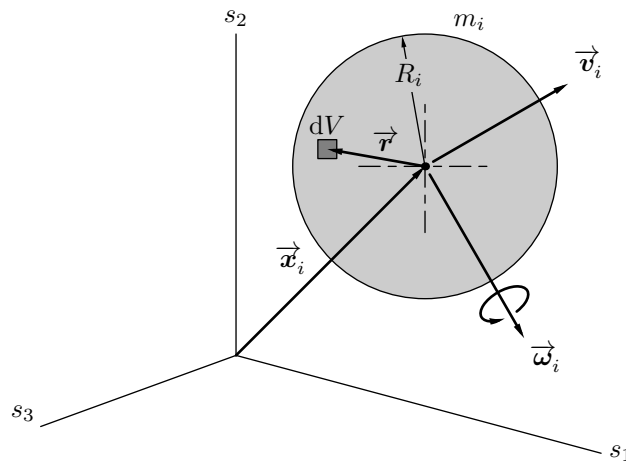
#### A.1.1 Ball mass and inertia parameters

Consider a volume element  $dV$  with respect to a static base  $S$  of an arbitrary solid body with density  $\rho$ . The mass of the body is obtained by integrating over the volume of the body,

$$m = \int_{\text{body}} \rho dV \quad (\text{A.1})$$

In figure A.1, a ball with radius  $R_i$  and uniform density  $\rho_i$  is depicted. The mass of the ball is after integration of equation (A.1)

$$m_i = \frac{4}{3}\pi\rho_i R_i^3. \quad (\text{A.2})$$



**Figure A.1:** Ball Element Parameters

# List of References

- [1] “DJI And DroneSAR Bring Search And Rescue App To First Responders,” nov 2016. [Online]. Available: <https://www.dji.com/ie/newsroom/news/dji-and-dronesar-bring-search-and-rescue-app-to-first-responders>
- [2] H. Azpúrua, G. M. Freitas, G. Douglas, and M. F. M. Campos, “Multi-robot coverage path planning using hexagonal segmentation for geophysical surveys,” *Robotica*, vol. 36, no. 2018, pp. 1144–1166, 2018.
- [3] V. G. Nair and K. R. Guruprasad, “GM-VPC: An Algorithm for Multi-robot Coverage of Known Spaces Using Generalized Voronoi Partition,” *Robotica*, vol. 38, no. 5, pp. 845–860, 2020.
- [4] C. Rossi, L. Aldama, and A. Barrientos, “Simultaneous task subdivision and allocation for teams of heterogeneous robots,” *Proceedings - IEEE International Conference on Robotics and Automation*, no. May, pp. 946–951, 2009.
- [5] N. Hazon and G. A. Kaminka, “Redundancy, efficiency and robustness in multi-robot coverage,” *Proceedings - IEEE International Conference on Robotics and Automation*, no. April, pp. 735–741, 2005.
- [6] T. M. Cabreira, L. B. Brisolara, and R. Ferreira Paulo, “Survey on coverage path planning with unmanned aerial vehicles,” *MDPI*, 2019.
- [7] J. A. Guerrero and Y. Bestaoui, “UAV path planning for structure inspection in windy environments,” *Journal of Intelligent and Robotic Systems: Theory and Applications*, 2013.
- [8] P. Lottes, R. Khanna, J. Pfeifer, R. Siegwart, and C. Stachniss, “UAV-based crop and weed classification for smart farming,” *Proceedings - IEEE International Conference on Robotics and Automation*, 2017.
- [9] I. Maza, F. Caballero, J. Capitán, J. R. Martínez-De-Dios, and A. Ollero, “Experimental results in multi-UAV coordination for disaster management and civil security applications,” *Journal of Intelligent and Robotic Systems: Theory and Applications*, 2011.
- [10] W. Chang, G. Yang, J. Yu, Z. Liang, L. Cheng, and C. Zhou, “Development of a power line inspection robot with hybrid operation modes,” *IEEE International Conference on Intelligent Robots and Systems*, 2017.



- [11] N. Basilico and S. Carpin, “Deploying teams of heterogeneous UAVs in co-operative two-level surveillance missions,” *IEEE International Conference on Intelligent Robots and Systems*, 2015.
- [12] H. X. Pham, H. M. La, D. Feil-Seifer, and M. Deans, “A distributed control framework for a team of unmanned aerial vehicles for dynamic wildfire tracking,” *IEEE International Conference on Intelligent Robots and Systems*, 2017.
- [13] S. M. Lavalle, *Planning Algorithms*. Cambridge University Press, 2006.
- [14] H. Choset, “Coverage for robotics - A survey of recent results,” *Annals of Mathematics and Artificial Intelligence*, 2001.
- [15] K. K. M. Lynch and F. C. Park, *Modern robotics : mechanics, planning, and control*. Cambridge University Press, 2017.
- [16] I. A. Hameed, “Intelligent coverage path planning for agricultural robots and autonomous machines on three-dimensional terrain,” *Journal of Intelligent and Robotic Systems: Theory and Applications*, 2013.
- [17] P. N. Atkar, A. Greenfield, D. C. Conner, H. Choset, and A. A. Rizzi, “Uniform coverage of automotive surface patches,” *International Journal of Robotics Research*, 2005.
- [18] N. Mir-Nasiri, J. Hudyjaya Siswoyo, and M. H. Ali, “Portable Autonomous Window Cleaning Robot,” *Procedia Computer Science*, 2018. [Online]. Available: <https://doi.org/10.1016/j.procs.2018.07.024>
- [19] E. M. Arkin, S. P. Fekete, and J. S. Mitchell, “Approximation algorithms for lawn mowing and milling,” *Computational Geometry: Theory and Applications*, 1999.
- [20] B. Englot and F. S. Hover, “Sampling-based coverage path planning for inspection of complex structures,” *ICAPS 2012 - Proceedings of the 22nd International Conference on Automated Planning and Scheduling*, pp. 29–37, 2012.
- [21] E. Galceran and M. Carreras, “A survey on coverage path planning for robotics,” *Robotics and Autonomous Systems*, 2013.
- [22] O. McGrath, “Are UAVs the future of search and rescue?” *HEMS/SAR*, no. 92, 2018. [Online]. Available: <https://www.airmedandrescue.com/latest/long-read/are-uavs-future-search-and-rescue>
- [23] H. Zhang, B. Xin, L. hua Dou, J. Chen, and K. Hirota, “A review of cooperative path planning of an unmanned aerial vehicle group,” *Frontiers of Information Technology and Electronic Engineering*, vol. 21, no. 12, pp. 1671–1694, 2020.
- [24] F. Kamrani and R. Ayani, “UAV Path Planning in Search Operations,” *Royal Institute of Technology (KTH) Sweden*, 2009.

- [25] R. Nandakumar and N. Ramana Rao, “Fair partitions of polygons: An elementary introduction,” *Proceedings of the Indian Academy of Sciences: Mathematical Sciences*, vol. 122, no. 3, pp. 459–467, 2012.
- [26] A. Barrientos, J. Colorado, J. D. Cerro, A. Martinez, C. Rossi, D. Sanz, and J. Valente, “Aerial remote sensing in agriculture: A practical approach to area coverage and path planning for fleets of mini aerial robots,” *Journal of Field Robotics*, vol. 28, no. 5, pp. 667–689, sep 2011.
- [27] X. Zheng, S. Jain, S. Koenig, and D. Kempe, “Multi-Robot Forest Coverage \*,” *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2005.
- [28] G. Even, N. Garg, J. Könemann, R. Ravi, and A. Sinha, “Min-max tree covers of graphs,” *International Conference on Approximation Algorithms for Combinatorial Optimization*, 2003.
- [29] Propeller Aero, “What is Ground Sample Distance ( GSD ) and How Does it Affect Your Drone Data ?” 2021. [Online]. Available: <https://www.propelleraero.com/blog/ground-sample-distance-gsd-calculate-drone-data/>
- [30] H. W. Jurgens, I. Matzdorff, and J. Windberg, “International Anthropometric Data for Work-Place and Machinery Design,” vol. 108, pp. 8–9, 1998.
- [31] P. Rudol and P. Doherty, “Human body detection and geolocalization for UAV search and rescue missions using color and thermal imagery,” *IEEE Aerospace Conference Proceedings*, pp. 1–8, 2008.
- [32] A. C. Kapoutsis, S. A. Chatzichristofis, and E. B. Kosmatopoulos, “DARP: Divide Areas Algorithm for Optimal Multi-Robot Coverage Path Planning,” *Journal of Intelligent and Robotic Systems: Theory and Applications*, vol. 86, no. 3-4, pp. 663–680, 2017. [Online]. Available: <http://dx.doi.org/10.1007/s10846-016-0461-x>