MULTI-UAV COVERAGE PATH PLANNING FOR RECONSTRUCTION OF 3D STRUCTURES

by

Shyam Sundar Kannan

A Thesis

Submitted to the Faculty of Purdue University

In Partial Fulfillment of the Requirements for the Degree of

Master of Science

Department of Computer and Information Technology

West Lafayette, Indiana

May 2019

THE PURDUE UNIVERSITY GRADUATE SCHOOL
STATEMENT OF COMMITTEE APPROVAL

Dr. Byung-Cheol Min, Chair
    Department of Computer and Information Technology
Dr. Thomas J. Hacker
    Department of Computer and Information Technology
Dr. Xiumin Diao
    School of Engineering Technology

Approved by:
    Dr. Eric T. Matson
        Head of the Graduate Program

I dedicate my thesis work to my parents and grandparents. A special gratitude to my mom Rajalakshmi for her love, support, inspiration and making me into who I am today.

ACKNOWLEDGMENTS

I would like to express my deepest gratitude to my mentor and guide Dr. Byung-Cheol Min. I owe him a life long debt and without him it would not have been possible for me to complete my masters thesis. He provided me with freedom, support and encouragement in making my ideas a reality. More than an academic advisor, Dr. Min was a life guide on how to be a good human who treats everyone with love and respect. My sincere thanks to Dr. Thomas Hacker and Dr. Xiumin Diao for accepting to be in my committee and helping me shape my research by providing feedback. My special thanks to Timothy Ropp for being a part of the project and supporting me in every phase of the project.

I thank Purdue University and Department of Computer and Information Technology for providing me with this opportunity to learn and conduct my research. I sincerely thank Prof. Guity Ravai for her continuous support.

I am deeply grateful to Dr. Ramanathan Muthuganapathy and Dr. Subhasree Methirumangalath for introducing me to research and helping me develop my skills to perform independent research. Purdue would have still been a dream for me without you.

My sincere thanks to my lab mates: Wonse Jo, Shaocheng Luo, Dr. Ramviyas Parasuraman, Arabinda Samantaray, Manoj Penmetcha, Tamzidul Mina, Dr. Yogang Singh, Jun Han Bae, Sangjun Lee, Jee Hwan Park, Yu-Hsi Wang, Yuta Hoashi and Walter Kruger. It was fun working with everyone and I have learnt a lot from everyone over the last two years.

I thank all my beloved friends at Purdue: Mythra Varun, Vikranth Reddy, Praveen Abbaraju, Naveen Madapana, Ram Kishore, Shashaank Yogesh, Swarnawa Mukerjee, Vishnu Radhakrisha, Khushal Bhatija, Shanmukh Sarode, Suyash Agrawal, Rohan Tibrewala, Vineet Nagaraj, Arjun Shakdher, Kyle Spisak, Nidhi Sakhala, Pooja Patil and Vaishnavi Radkar . Without you people my life at Purdue would have never been this great.

Last but not the least, my cordial gratitude to my father Kannan, mother Rajalakshmi and sister Priya for their affection and care. My special thanks to my beloved grandfather Narayanaswamy for your inspiration and motivation. If not for you all my graduate school dreams would have never come true.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

UAV	Unmanned Aerial Vehicle
CPP	Coverage Path Planning
TSP	Travelling Salesman Problem
CSP	Coverage Salesman Problem
GPU	Graphical Processing Unit
DOF	Degree of Freedom
SITL	Software in the Loop
ROS	Robot Operating System
SLAM	Simultaneous Localization and Mapping
DSM	Digital Surface Model
LiDAR	Light Detection and Ranging
SfM	Structure from Motion
SVO	Semi-direct Visual Odometry
RC	Rectangular Cuboids
MAV	Mirco Aerial Vehicle
VI	Visual Intertial
STL	Standard Triangle Language

ABSTRACT

Author: Kannan, Shyam Sundar. M.S.
Institution: Purdue University
Degree Received: May 2019
Title: Multi-UAV Coverage Path Planning for Reconstruction of 3D Structures
Major Professor: Byung-Cheol Min

Path planning is the generation of paths for the robots to navigate based on some constraints. Coverage path planning is where the robots needs to cover an entire workspace for various applications like sensing, inspection and so on. Though there are numerous works on 2D coverage and also coverage using a single robot, the works on 3D coverage and multi-agents are very limited. This thesis makes several contributions to multi-agent path planning for 3D structures.

Motivated by the inspection of 3D structures, especially airplanes, we present a 3D coverage path planning algorithm for a multi-UAV system. We propose a unified method, where the viewpoints selection and path generation are done simultaneously for multiple UAVs. The approach is scalable in terms of number of UAVs and is also robust to models with variations in geometry. The proposed method also distributes the task uniformly amongst the multiple UAVs involved and hence making the best use of the robotics team. The uniform task distribution is an integral part of the path planner. Various performance measures of the paths generated in terms of coverage, path length and time also has been presented.

CHAPTER 1. INTRODUCTION

In this chapter, we elaborate the motivation for performing this research and the real world problem behind the motivation: the inspection of airplane and other structures using an Unmanned Aerial vehicle (UAV). This problem has motivated us to look into the development of a novel coverage path planning technique that is capable of covering the entire structure. Motivation in Section 1.1 is followed by the significance and objectives of this thesis. To lay a foundation for the technical argument in this thesis, a review of relevant prior work is given in Chapter 2.

1.1 Background and Motivation

In the past decade, the easy availability of UAVs with inexpensive and lightweight sensors have paved way for the application of aerial vehicles in various domains. UAVs improve the efficiency and economy of the tasks, but their contributions are significant when it comes to the speed of the task. There are numerous tasks that fall under this category, but one of the most practical one is the inspection of airplanes, 3D structures and building for search and rescue [1].

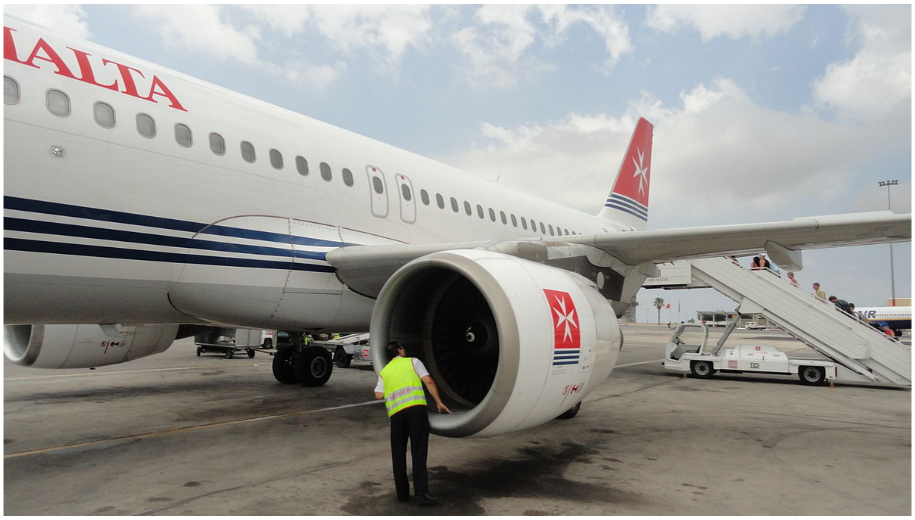


Figure 1.1. Human performing inspection of the airplane [2] [©2018 Kristoferb].

When the airplane inspection is performed by the humans as illustrated in Figure 1.1, not only the humans are at risk of serious injury, but there is a possibility that hidden ordinance may go unnoticed if any part of the airplane is missed in the inspection. The aim of using autonomous UAV is to perform the task safely and to obtain complete coverage of every exposed surface.

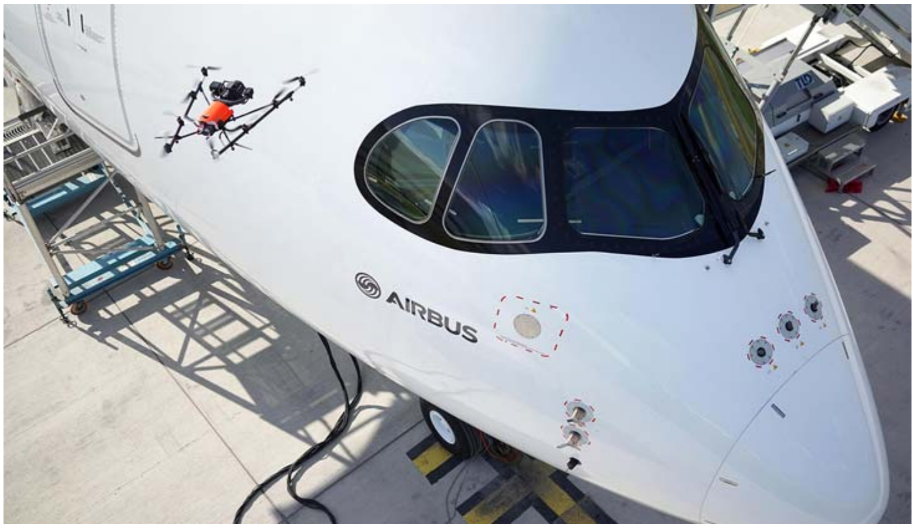


Figure 1.2. Ascending Technologies UAV performing inspection [3] [©2018 Airbus].

Efforts to automate the inspection of the airplane include works from Airbus and Ascending Technologies [3], which uses one UAV and 42 mega-pixel high resolution camera to identify the defects. This system was able to reduce the inspection time from two hours to less than 15 minutes, establishing that the automation of inspection reduces the inspection time substantially. Figure 1.2 shows an Ascending Technologies UAV performing inspection of an Airbus 330 aircraft.

A major concern in the usage of the UAVs for the inspection tasks is the limited battery capacity of the UAVs. From previous research [4], it evident that the UAV consumes more energy with greater changes in the altitude of the UAV. In order to ensure that the UAV completes the inspection task without having issues of battery getting drained, it is necessary that the paths generated are energy efficient too.

The goal of this thesis is to generate paths for multiple UAVs for inspection such that we get complete coverage of the structure being inspected and also at the same time minimize the variations in the pitch in order to conserve energy. We assume that a prior model of the airplane or the 3D structure to be inspected is available for planning the geometric inspection route. The model helps not only in planning the route but also in localizing the defects on the surface. Path planning to achieve sensor coverage of a large, complex structure, using a sensor with a limited field of view, is the central focus of this thesis. Motivated by autonomous airplane inspection problem, we contribute new algorithm for the solution of coverage path planning over complex 3D structures.

1.2 Significance

Inspection and maintenance of airplane and other structures is a mandatory and important task in order to preserve the quality of the structures and also maintain the safety. So it is necessary that these structures are inspected at regular intervals. Due to the tall and gigantic nature of these structures, performing manual inspection is a tedious task. This is time consuming and also dangerous. For instance, when a windmill is inspected manually, the humans need to tie ropes to the structures at great heights. This process takes couple of hours and involves many potential dangers. Also, the windmill cannot be operational, when the inspection process is going on and this also crates an economic loss considering the fact the windmill is non-operational for a long span of time.

This research automates the inspection process using UAVs. UAVs are agile and can cover huge area in a short span of time. Also the usage of UAVs does not need any pre-setup like setting up the harness etc. as in the case of manual inspection and hence making the process a lot quicker. This not only makes the inspection process quick but also gives economic benefits. For instance, automating the inspection of airplane can contribute a lot to the airline industry in avoiding flight delays and so on. Hence, automating the inspection process using UAVs gives significant advantages in terms of time and cost to the overall process.

1.3 Objectives

Based on the motivation, we focus on achieving the following objectives:

- To generate paths for the UAVs such that all the paths have almost equal coverage and also attain complete coverage of the structure.
- To optimize the paths such that there is no huge variation in the pitch from one point to another.
- To perform experiments (simulation) to measure the coverage achieved, when the UAVs trace the path.

1.4 Research Question

The research plans to answer the following research question:
Can automating the inspection process using multiple UAVs yield complete coverage of the 3D structure?

1.5 Assumptions

The assumptions of the thesis include:

- We assume that the 3D model of the structures are readily available for the structures we wish to inspect.
- It is assumed the odometry data obtained from the localization system is error free.

1.6 Limitations

This research includes the following limitations:

- The system does not optimize the number of UAVs needed. The system generates *n* trajectories, if *n* UAVs are available. Sometimes *n* can be more than the optimal number of UAVs *m* needed to carry out inspection is the shortest span of time. When *n* is more than the *m*, the system tends to over do the task which might lead to sub-optimal performance.

1.7 Delimitation's

This research includes the following delimitation's:

- The research does not focus on generating trajectories with collision avoidance. The trajectories generated by this method can have overlapping trajectories which can cause a collision.

1.8 Thesis Layout

This thesis is organized in five chapters. The thesis starts with the motivation, goals and objectives which marks the first chapter. Chapter 2 focuses on reviewing the current literature on coverage path planning and state of the art methods in inspection.

Chapter 3 gives an overview of the end to end framework of the entire system, along with the various software. The Chapter 4 explains the algorithm with the various steps involved in it. The Chapter 5 evaluates the proposed algorithm in terms of various parameters. Finally, the work is summarized and concluded in Chapter 5.

CHAPTER 2. REVIEW OF LITERATURE

2.1 Coverage Path Planning

Coverage Path Planning (CPP) is a motion planning problem, which focuses on generating paths for getting complete coverage of the workspace. In CPP, the complete coverage path for a robot is determined in its workspace. The robot must cross every point in the workspace which is very similar to covering salesman problem (CSP) which is a variant of the travelling salesman problem (TSP), where every neighborhood must be crossed instead of every city like in TSP.

CPP algorithm is primarily used in service robots like the ones used in agricultural and harvesting, cleaning and housekeeping, lawn moving and inspection of structures. The final goal of CPP is to achieve complete coverage of the area and also at the same time reducing the time needed to complete the task. The authors in [5] defines the criteria for the region filling operation as follows:

- Robot should cover the entire workspace.
- The paths generated should not overlap.
- There should not be any repetition of the paths.
- Robot needs to avoid the obstacles in the workspace.
- It is better to use simple motion trajectories.
- The path generated should be optimal.

The problem to generate coverage paths to coverage the entire workspace is a well researched topic. The following sections review the existing works on coverage path planning in 3D.

2.1.1 3D Cellular Decomposition

Earlier methods in coverage path planning considered decomposing the region around the structure into regions and then developing trajectories passing through all the cell. Atkar et al. proposed an online method complete coverage method for 3-dimensional orientable surfaces in [6] by extending Morese decomposition to 3D. This work mainly focuses on trajectory generation for spray-painting robots. In this method, the coverage is done for a virtual surface which is at a fixed distance from the actual surface. For spray painting the end effector needs to be at a fixed offset distance from the surface. In this method, the coverage path is generated by using slicing place a fixed interval on the virtual surface. The intersection of the planes form a closed loop around the structure and this is used for generating the trajectory. This loop is used by the robot in moving from one plane to another and this process is repeated iteratively. This method acheives complete coverage if the surface is completely convex, but if there are some concavities in the surface, this method may not give complete coverage. Even in this approach, Reeb graph encodes the topology of the surface similar to Morse decomposition. Now, the edges of the graph are connected to get the path for complete coverage. Simulation was used to validate the performance of the algorithm. Obstacles on the target surface and the dynamics of the robot are not considered in this work.

Atkar et al. in their later works [7], [8], [9] focused on not only achieving complete coverage, but also that the paint deposition should have certain uniformity requirements. These methods were off line opposed to their previous work on an online method focused on spray painting for automotive parts. In order to get uniformity in the painting task, the method segments the surface of the input CAD model into topologically simple cells with similar curvature. Now, coverage path are generated for each individual cell. The methods were validated both in simulations as well as in real.

2.1.2 3D Structure Coverage using Geometric methods

Coverage path planning is predominantly used in inspection of 3D structures. A method for complete coverage of 3D urban structure was proposed by Cheng et al. [10]. In this method offline planning on a given model of 3D structures is done to generate time-optimal trajectories for UAVs. The structures to be coverage like buildings are simplified in to hemisphere and cylinders depending on the geometry of the structures. Now, it is easy to generate trajectories for this simpler shapes compared to the complex urban setting. This method was validated using a fixed-wing aircraft and this method had been illustrated in Figure 2.1.

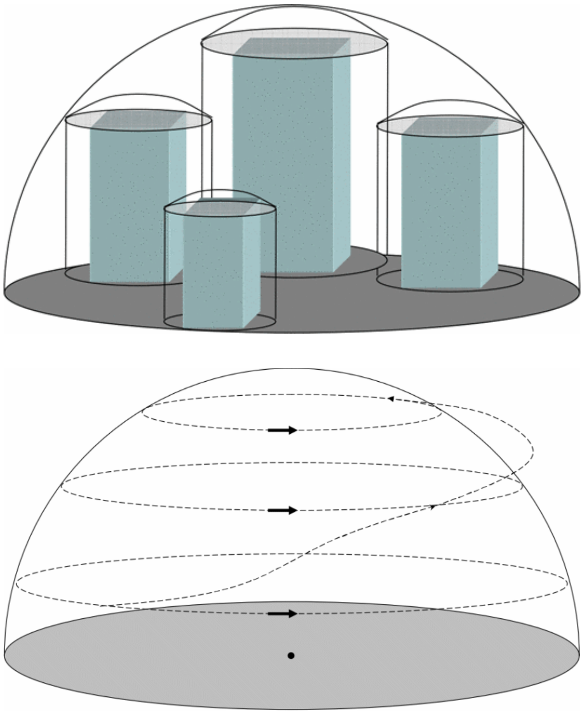


Figure 2.1. 3D model of an urban setting and the coverage path generated in [10] [©2008 IEEE].

An adaptive search space coverage method was proposed in [11]. In this work methods for generating view points for complete coverage and also trajectory passing through those points was proposed. Exhaustive simulation based method was used for generating the viewpoints which requires a graphical processing unit (GPU). The coverage was quantified by computing the ratio of the volume captured to the actual volume of the 3D structure. A multi-layer coverage path planning method has been presented in [1]. In this method, the given 3D model is segmented into k layers and coverage path planning is performed locally at each layer using Lin-Kernighan heuristic (LKH) solver [12]. Once the paths are generated for covering each layer, the layer are connected and hence giving a path to cover the entire model. A given model being segmented into layer has been shown in Figure 2.2.

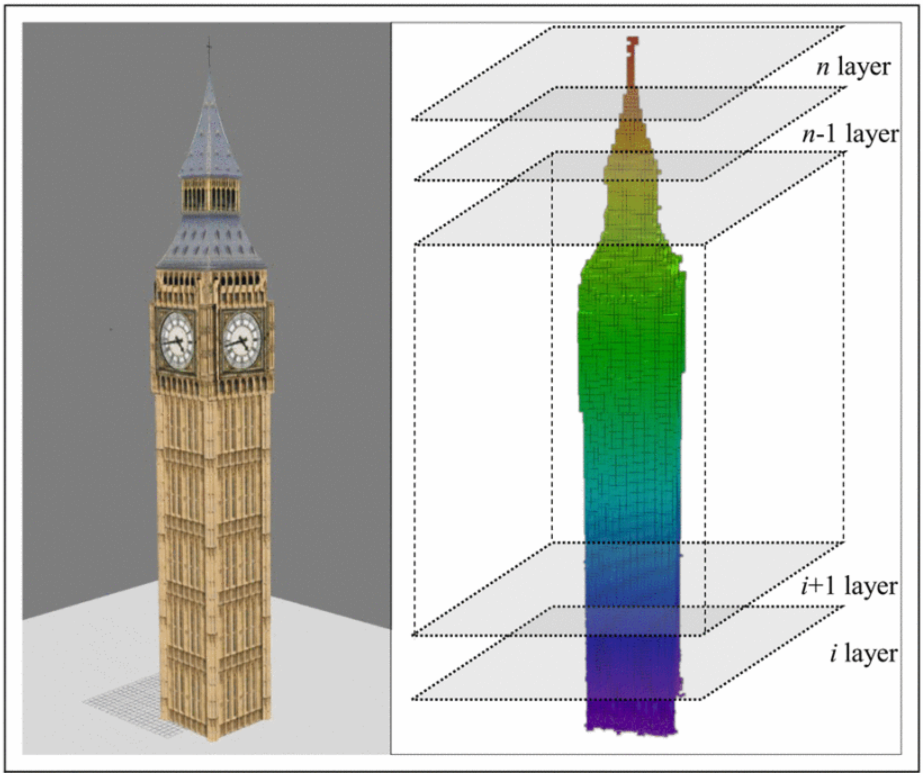


Figure 2.2. Illustration of multi-layer coverage path planning method as briefed in [1] [©2018 IEEE].

A fast algorithm for the inspection of 3D structures by proposed by Bircher et al. in [13]. The method is scalable for both fixed wing UAV as well as vertical take off UAV, and hence giving this method an edge over other methods. The method can be used for both high rise structures and 3D planar surfaces too.

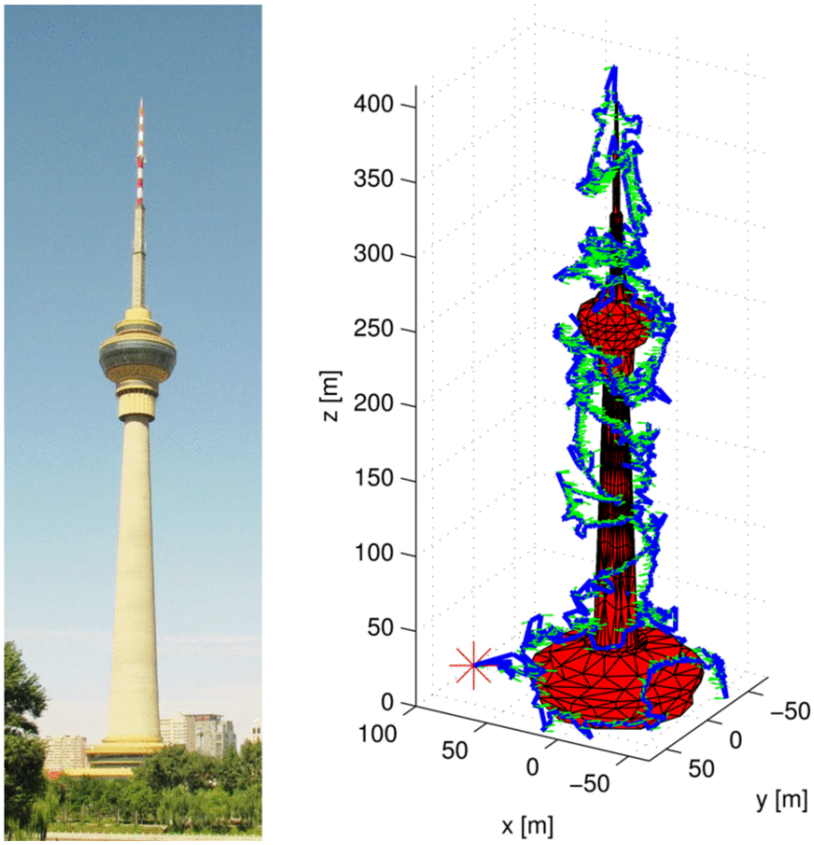


Figure 2.3. The path generated for a tower for the method in [13] [©2015 IEEE].

2.1.3 Sampling-based coverage of complex 3D structures

Sampling based methods were primarily developed to handle scenarios where there are occlusions in the 3D structures and certain regions are visible only from few points. In [14] sampling based global path planning strategies is used to compute collision free paths through confined areas.

Another sampling based method has been presented in [15] where coverage paths are generated for 3D structures especially ship hulls. In hull inspection the robot covers the part of the hull under the water surface. Sonar is primarily used in hull inspection for mapping. In this method 6 degree-of-freedom trajectories are generated for the AUV equipped with bathymetry sonar for mapping and inspection. Similar the previous works, this method also need a 3D model of the structure as input. In this approach, initially a path is built using randomly sampling the space around the structure until a set of nodes giving complete coverage is attained. Using these points generated from random sampling, shortest path covering all the nodes is computed. Then, this trajectory is smoothened while maintaining the coverage. In this method, random sampling to reduce the computations involved finding the nodes. The method was validated using sonar on real hulls. Figure 2.4 illustrates the method where the of paths for inspecting the ship hull before and after smoothing has been shown.

In [15], Englot and Hover proposed a method similar to the previous one where they first generate a set of view points that completely covers the structure. Then a path passing through all these view points is computed using Travelling Salesman Poblem, TSP. This might sometimes give paths which cannot achieved by the dynamics of the robot. Papadopoulos et al. in [16] proposed a random sampling method in which they initially generate view points randomly the surface for complete coverage. These random view points are joined based on the feasibility of the robot to achieve the next configuration. The view locations which are reachable are only considered for the generation of the feasible paths.

2.1.4 Bathymetric surfaces coverage

Coverage planning has been used to map the ocean floor too. Bathymetric surface maps are maps consisting of the elevation details of the ocean bed.Topological and elevation information obtained from these maps are utilized for the coverage planning on the ocean floor using autonomous underwater vehicles. A survey of the bathymetic surface is done by the entire coverage of the surface and by capturing the elevation profile of the terrain and also capturing pictures of the surface.

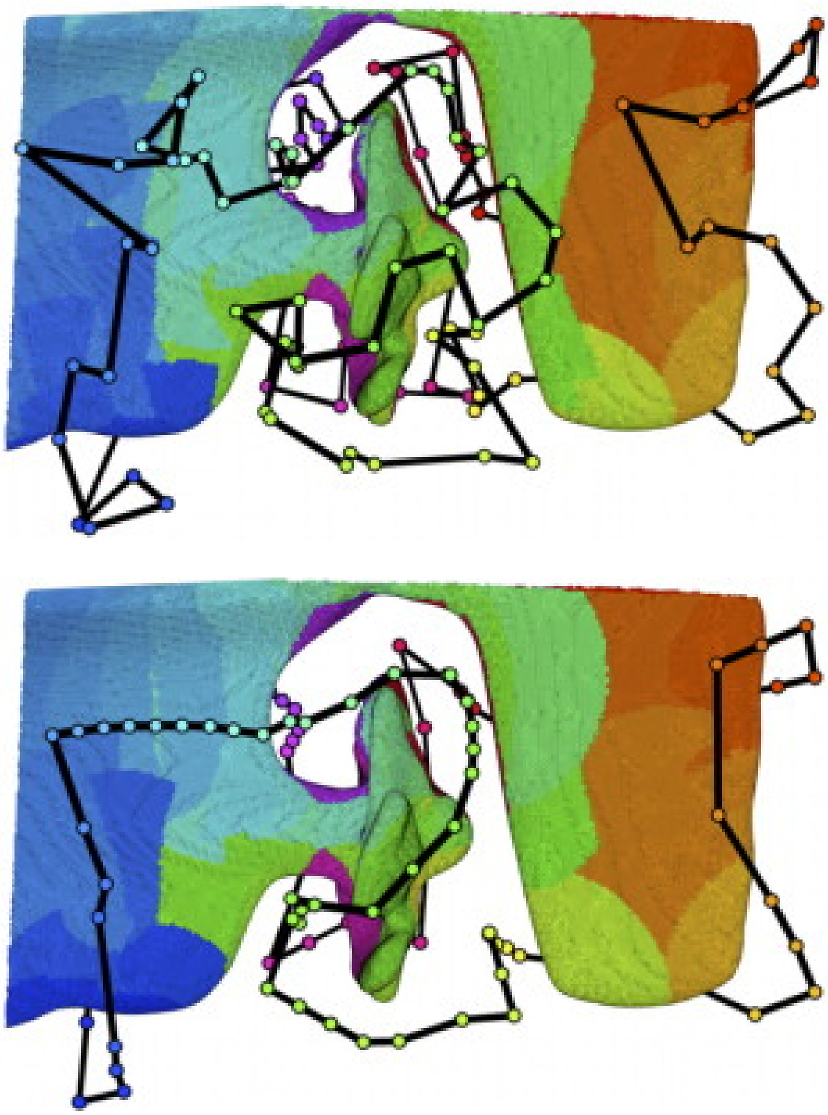


Figure 2.4. Coverage path given by [15] - Top Image - Path for complete coverage before smoothening which is 176 m in length and contains 121 nodes. Bottom Image - Path for complete coverage after smoothening which is 102 m in length and contains 97 nodes [©2019 AAI].

However, the rugged nature of the terrain prevents the easy and complete coverage by moving along a 2D plane. An algorithm to generate path in 3D to cover the variations in the terrain is needed. The authors in [17] presented a method which initially finds the regions covered by traditional surveying methods based on the gradient of the terrain. Then, for each of the identified region, it does 3D coverage locally. Over the remaining areas it does traditional 2D coverage. This method is able to completely map the bathymetric terrain and was verified in real time over a volcanic terrain.

2.1.5 Arable farming coverage

Most of the works on agricultural coverage consider the farm as a 2D plane. However, in [18] a coverage method for arable farms where the field is represented as elevation maps was presented. In this methods, a speed curve is generated by incrementally offsetting on both the sides. Factors such as number of turns taken by the UAV, soil erosion cost and the remaining area to cover are considered in the generation of the speed curve. This method was validated on a real farm.

2.1.6 Multi-UAV 3D Coverage

A number of works have focused on using multi-robot system for coverage path planning. These methods usually assume that UAVs can fly without much constraints on the height.

An integer programming based coverage path planning algorithm for multiple UAV was proposed by Ahmadzadeh et al. in [17]. Integer programming (IP) provides an easy and a convenient way to represent the constraints on the limited maneuverability of the UAV. Using IP a solution for the generating path for the UAV which follows all the constraints is obtained. For this method both simulation and field experiments were performed.

A partitioning based methods for multi-UAV system was presented by Maza and Ollero in [19]. Initially the terrain to be covered is partitioned into polygons. The range of the UAV and the dynamics are considered in the creation of the polygons. The number of polygons created depends on the number of UAVs available. Now, zigzag patterns are generated for the UAV to cover the polygons. The number of turns are minimized in generating the zigzag pattern. The proposed method was validated in simulation.

In [20], a task scheduler based method was proposed. The task scheduler tries to different regions in the terrain for the multiple UAVs to cover with a goal of trying to maximize the area covered by each UAV unlike previous method were geometry was used in partitioning. The scheduling under goes a negotiation process in allocating the regions. Finally, the UAVs allocated non overlapping regions and they plan locally for each region to attain complete coverage.

2.2 3D Reconstruction

In this section, we discuss on the existing methods used for the 3D Reconstruction of the models. These techniques are predominantly used in inspection, surveillance, mapping and navigation. Reconstruction is usually done with a 3D imaging sensor which combines the 3D obtained from each from into a single map. The accuracy of the map created greatly depends on the localization used which usually done using simultaneous localization and mapping (SLAM) where the location of the robot estimated while the map is being built. Reconstruction techniques are usually used for mapping a region or structures and the following subsection focuses on the existing techniques for these.

2.2.1 3D Environment Mapping

3D Environment mapping are mainly used for exploration and surveillance scenarios like search and rescue etc. CPP based mapping methods are used to get complete map of the region.

Semi-direct visual odometry (SVO) based method was proposed in [21] that follows a trajectory also at the same time maps the environment it covers. It estimates the camera location in the map frame and uses this information to extend the map. Regularized monocular depth algorithm is used for the reconstruction using the sensor and location data. Another , real-time mapping system using moncular camera has been presented in [22]. It uses a visual SLAM approach for the mapping and localization.

The octree-based representation of the environment is a map in which encapsulates the shape and the reflectance distribution of the objects in the environment. The two major steps involved in the registration of the octree map are surface association i.e. finding loop closures in the data collected and estimating the pose. Octomap and RMAP, 3D mapping frameworks were proposed in [23] and [24]. OctoMap framework is used to build octree of the 3D space which clearly represents the occupied, free and unknown space in terms of probability. RMAP framework uses Rtree data structure made of rectangular cuboids. Here the environment is represented in terms of probabilities using these rectangular cuboids and occupancy grid. Figure 2.5 the result of Octomap.

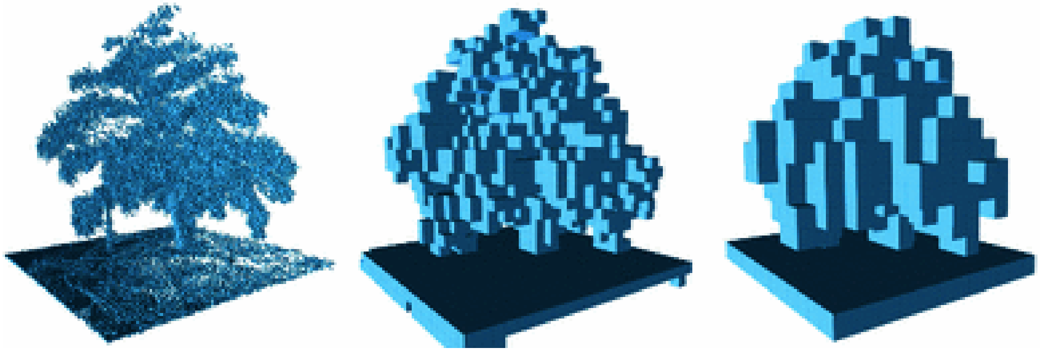


Figure 2.5. The octree generated using Octomap [23] [©2019 Elsevier B.V].

Furthermore, octree-based maps were generated in [25], [26] and [27] which are an extension of OctoMap [23]. Visual Inertial (VI) and vision sensor data were fused in [25], where this information was used in the construction of the map and also for localization. In [26] octree method was proposed. Here LiDARs were used for pose tracking and the map is constructed with this reference.

OctoSLAM algorithm was proposed in [27] which is also an extension of OctoMap where Hector SLAM [28] algorithm was used for mapping. The Data from other inertial sensors and altimeter were also fused in map generation and localization. A novel mapping technique which uses spatial approach was presented in [29] where occupancy grid is represented as normal distribution function. This method was able to update the map at a higher frequency and can also create maps at various resolution. The higher refresh rate enables the mapping of dynamic objects in the environment with higher accuracy.

2.2.2 3D Structures Reconstruction

For 3D structure reconstruction, a model of the structure is usually given as input. In [30], Hover et al. proposed a method for reconstructing the ships external hull in order to detect mines. Sonar data was used in this process and was processed into a point cloud. The point cloud generated is then sub-sampled to estimate the normal and orientation which are later used to detect the mines.

Image-based 3D reconstruction of structures, where the images captured by the UAV from multiple UAVs are combined using the geospatial information was proposed by Maureret al. in [31]. The images are combined to create semi dense 3D models which is considered to give an approximate representation of digital surface model (DSM). Roof Reconstruction using UAV has been presented in [32] . Here the elevation data from the Light Detection and Ranging (LiDAR) is used for the Reconstruction. This approach does reconstruction using parametric shapes, segmentation and DSM simplification.

In [33] an image-based modelling in which they parametrize the surface was proposed. This approach uses structure from motion (SfM) coupled with bundle adjustments for the reconstruction. Here, the point cloud is estimated using bundle adjustments and geometric refinements. Finally the surface is estimated using Poisson surface reconstruction algorithm [34]. Another camera based reconstruction was presented in [35]. The reconstruction of the surface was done using traditional image mapping techniques from computer vision such as SIFT or SURF. The approach has shown reconstructed models of the Chillon Castle and Matterhorn Mountains in the Switzerland.

2.3 Summary

In this section, we presented a comprehensive review of the relevant existing literature in coverage path planning and 3D reconstruction. It can be observed that the existing methods especially for UAV are affected by some form of limitation either in terms of coverage or being extensible to multiple agents or for the coverage of 3D structures. Also, though there are numerous good algorithms for reconstruction, they have not been well integrated with the coverage path planning system.

Hence, it can be clearly seen that the there is enough research on coverage path planning in 3D, but still an optimal solution has not been developed yet. So, there remains a need for further research.

CHAPTER 3. FRAMEWORK

This chapter describes the methodology used in generating paths for the inspection of 3D structures. This includes the end to end framework, the software setup and the algorithm for generating the paths. The end to end framework describes the entire flow from processing the input model till the inspection and reconstruction of the structure. The software setup briefs the various software's and packages used in the implementation. Finally, the algorithm for generating the paths for the UAVs has been presented.

3.1 End to End (E2E) framework

The proposed framework consists of two main parts i) offline planning; ii) inspection and reconstruction of the structure. The input 3D model of the structure is processed offline and paths for inspecting it are generated. Then, based on the path generated the UAVs reconstruct a model of the structure which is used for the inspection. The overall structure of the E2E architecture has been depicted in Figure 3.1.

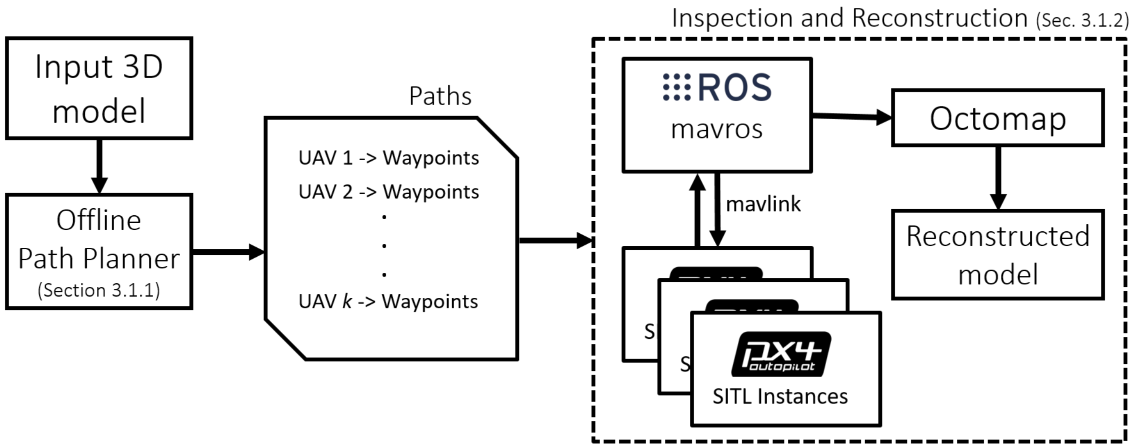


Figure 3.1. Schematic of the E2E framework.

3.1.1 Offline Planning

The offline planner takes the 3D model of the structure and the number of UAVs available for the inspection process as input. The input model is processed in two steps: i) viewpoint generation and ii) path generation; in order to get the waypoints for complete coverage.

3.1.1.1 Viewpoint Generation

The aim of this step is to generate a set of viewpoints around the 3D structure which guides the path generation. Viewpoints are set of 3D coordinates and the yaw angle from which the sensors can have a complete coverage over the structure.

The generation of viewpoints depends on the parameters of the sensors. During the inspection, the UAV are mounted with 3D Cameras (e.g. XBox Kinect, Intel Realsense) and they are used for mapping the surface of the structure. The parameters of the camera such as field of view and the distance of view (maximum distance the 3D camera works) are used for generating the viewpoints. In addition to these information, the 3D model of the structure is also needed. Now, the region of view of the 3D model from an external point can be computed based on the camera parameters. The objective of viewpoint generation is to generate a set of view points around the structure from which the complete structure can be covered. The viewpoints are later used to generate paths.

3.1.1.2 Path Generation

Now, we intend to generate paths using the viewpoints generated. A 4 degree of freedom (DOF) (position + yaw) path comprising of a set of waypoints will be generated.

Paths are generated in an optimal manner taking into account the dynamics of the UAV (e.g, avoid sharp turns) and considering maximum coverage. An exploration based approach is used to generate the path. The number of path generated depends on the number of UAVs available. These paths are later followed by the UAV to inspect the structure. The path generation algorithm has been explained in detail in the following chapter.

3.1.2 Inspection and Reconstruction

Once the offline planner generates the paths, the UAVs need to traverse the paths and reconstruct a new model of the structure using the data collected from the LiDAR or 3D camera attached to the UAV.

The followed software packages are used in the inspection and reconstruction process.

3.1.2.1 Robot Operating System Robot Operating System (ROS) [36] is middleware used in robotics and also an interface between various robots. The implementation primarily uses ROS and its topics (messaging formats) for communication. This will be enable the easy and centralized control between the various UAVs and the centralized controller. The packages available with ROS helps in the UAV control, mapping the structure and also in the localization.

3.1.2.2 MAVLink MAVLink or Micro Air Vehicle Link [37] is a commonly used protocol for communicating with micro aerial vehicle. This protocol helps in communication between the UAV and the ground stations. This protocol helps in both controlling the UAV as well as reading the various sensor data from the UAV.

3.1.2.3 MAVROS MAVROS [38] is an intermediator which acts an interface between the MAVLink and ROS. This helps us to make use of the functions of MAVLink from ROS. Using MAVROS the UAV can be made autonomous using the functionalities of ROS and the other packages available in ROS helps us add more functionality to the UAV.

3.1.2.4 Octomap Octomap [39] is package used to map the environment as an octree. This is used to create a map of the area inspected. This helps understanding the outcome of the inspection both qualitatively and also quantitatively. The view inspection of the model obtained helps in understanding the result of the scan qualitatively.

3.1.2.5 PX4 Autopilot PX4 autopilot [40] is an open-source autopilot system used for controlling autonomous aircraft's. The PX4 controller includes features like stabilization, waypoint follow and so on. It is a low level controller and MAVLink protocol is used for interacting with it from a high level. PX4 can be used for controlling aircraft's in both real time and in simulation.In the experiments PX4 is preferred because of its robustness and compatibility with ROS.

3.1.2.6 Rotors Simulator Rotors Simulator [41] is UAV simulator in ROS. It has the capability to control the UAV in a simulated environment using high level commands like velocity or waypoints. The simulator used Gazebo for visualization. The simulator has number of UAV models like AscTec Hummingbird, Firefly, Pelican and so on which can be used readily. The simulator also allows extension of the features by integrating new sensors with the UAV.

3.2 Summary

In this chapter, an overview of the end to end framework of the entire system has been presented along with the purpose of each component. The various software's used along with a brief description about them also have been presented.

CHAPTER 4. COVERAGE PATH PLANNING

In this chapter, the methodology for generating paths for the inspection of the 3D structure has been elaborated. The various steps in processing the input model, generating the viewpoints and connecting the viewpoints to generate the paths have been discussed along with the pseudo-code.

4.1 Inputs for the Path Planner

4.1.1 3D Model of Structure

The 3D mesh of the structure which needs to be inspected is given as input. A mesh is composed of vertices, edges and faces that defines the shape of the object in 3D space. The geometry of the structure given by the mesh model is used to generate the viewpoints from the entire model can be inspected. The model is also used in the trajectory generation to avoid collisions on to the structure. Figure 4.1 shows the mesh model for a cooling tower. The cooling tower model has been used in the later steps for the purpose of illustration.

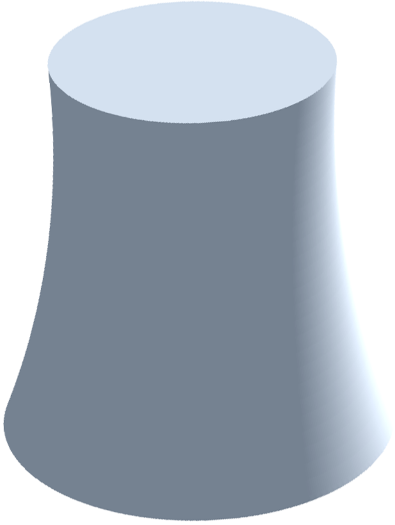


Figure 4.1. Mesh model for a cooling tower.

4.1.2 Number of UAVs

In addition to the 3D model of the structure to be inspected, the number of UAVs n available for the inspection is also given as input. The n UAVs used for the inspection process has been shown in Figure 4.2.

Figure 4.2. n UAVs used for the inspection.

4.1.2.1 UAV with 3D Camera

Each of the UAV used in the inspection process is mounted with a 3D camera as shown in Figure 4.3.The data collected by these 3D camera helps in the reconstruction of the model which aids the inspection process. The 3D cameras have a limited field of view being of the optics of the sensors used. The blue cone in Figure 4.4 shows the field of view of camera with θ as the viewing angle (here the camera has uniform viewing angle along the horizontal and vertical directions) and with the object at a distance d.

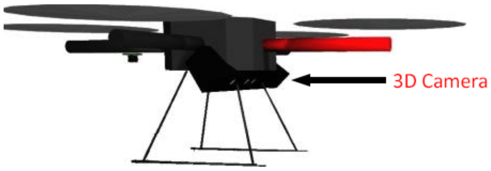


Figure 4.3. UAV mounted with 3D camera.

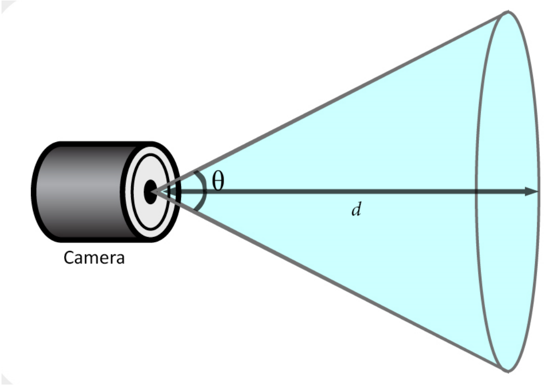


Figure 4.4. Field of view (blue cone) of the camera.

4.2 Viewpoints Generation

Viewpoints generation is the first step in the path planning process where a set of viewpoints are generated around the structure. In the later steps, a subset of these viewpoints are connected in generating the path to get complete coverage of the structures. The input 3D mesh model of the structure is used as a reference to generate the viewpoints.

The input mesh model is discretized as an octree for the purpose of each computation. Conversion of a sample 3D model (Stanford Bunny) into an octree has been depicted in Figure 4.5 and for the cooling tower model in Figure 4.6.

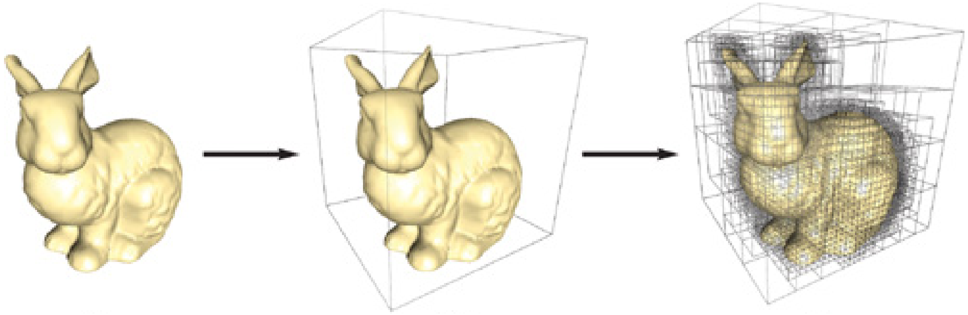


Figure 4.5. Step by step construction of the Octree from mesh model for Stanford bunny [42]

.

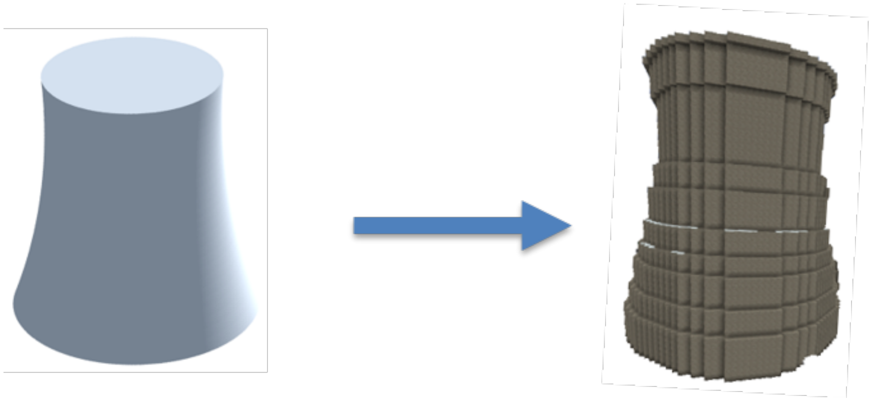


Figure 4.6. Conversion of the cooling tower model to octree.

.

Now, for every occupied cell in the octree, a candidate viewpoint is generated. These viewpoints are placed at a fixed distance d from the surface of the structure along the direction of the normal at that points. For safe flight of the UAV during inspection, it is important that the UAV maintains a safe distance from the structure so that there is not any accidental collisions with the structure during the flight. This distance which the UAV needs to maintain from the surface of the structure is termed as *safe flight distance, d*. The value of d is not set too high considering the spatial resolution of the camera. Farther the UAV, lower the spatial resolution of the images captures from the UAV, which can hinder the process of finding detects on the structure. So, the *safe flight distance* is decided such that the UAV maintains a safe distance from the structure, also at the same time it is not too far away from the structure.

Consider a voxel in the octree centered at the point $\vec{p}$ (red voxel in Figure 4.7) and direction of the normal at that point is $\vec{n}$ (orange arrow in Figure 4.7), then the viewpoint $\vec{vp}$ for the point $\vec{p}$ (green dot in Figure 4.7)is generated as (which all the coordinates are in the world frame):

$$\vec{vp} = \vec{p} + d * \vec{n}. \tag{4.1}$$

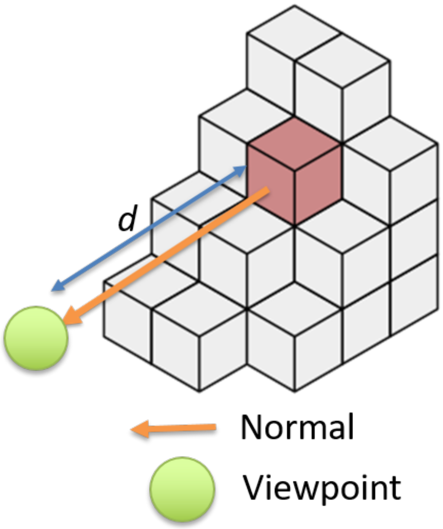


Figure 4.7. The viewpoint generated for the voxel highlighted in red.

.

In the same way, the viewpoints are computed for all the voxels in the octree and this constitutes the viewpoint cloud, vps. However, this point cloud of view points generated can be too dense and can make the later computations expensive. In order to speed up the path generation process this point cloud of viewpoints generated is sub-sampled using Voxel Grid filter and this produces a sub-sampled point cloud of lesser density.

For the viewpoints generated the view directions are also computed. View directions are direction of sight for the UAV at that viewpoint. In the view angle computation only the yaw angle is computed since the UAV is always parallel to the $xy-$plane (assuming that the model is placed on the $xy-$plane). So the roll and pitch angles are always fixed. The direction of the view $\vec{v}$ is computed by projecting the normal direction (n_x, n_y, n_z) of the octree voxel from which the viewpoint was generated on to the $xy-$plane and then reversing the direction of the vector to point it towards the surface of the structure as

$$\vec{v} = (-n_x, -n_y, 0) = -1 * (n_x, n_y, n_z) - (n_x, n_y, 0). \tag{4.2}$$

There can some voxels in the octree at which the direction of the normal is perpendicular to the $xy-$plane (i.e the UAV is parallel to the $xy-$plane) and it is not possible for the UAV to look in the direction of the normal. For this case, the view directions are not computed in prior and the direction of heading of the UAV at that instant is used as the viewing direction. In Figure 4.8, the UAV is parallel to the $xy-$plane and hence the viewing direction is decided based on the direction of heading of the UAV at that instant.

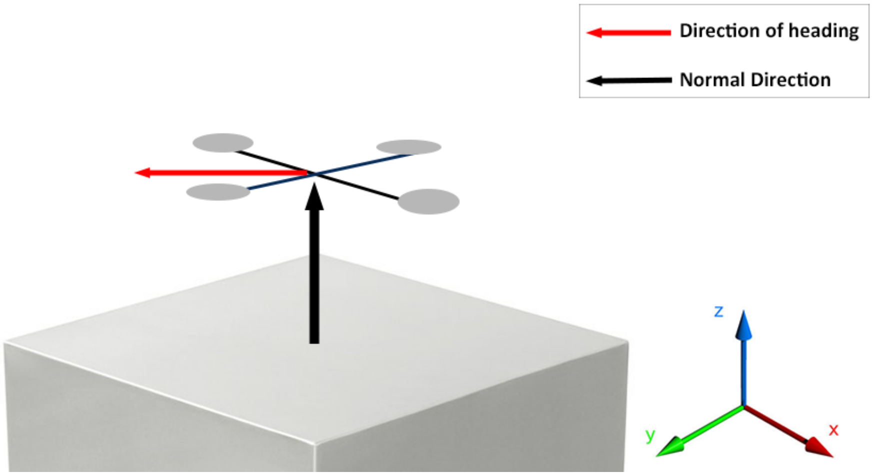


Figure 4.8. The UAV is parallel to the $xy-$plane and hence the heading direction is used as the viewing direction (red arrow in the left figure), not the normal direction (black arrow).

Depending on the geometry of the model, there might be some viewpoints which are inside the structure. Typically, these happen in concave regions of the model. These viewpoints are identified using ray tracing and are eliminated. A ray from the voxels in the octree corresponding to the points in the viewpoint point cloud is traced along its normal direction. The rays intersection with the model is computed and the distance at which the ray intersects the model d_i is computed. If this distance plus the dimension of bounding box of the UAV is less than the *safe flight distance, d*, then that viewpoint is not safe for the UAV for fly and that viewpoint is eliminated. The final viewpoint point cloud obtained after all these process has been shown in Figure 4.9 and the pseudo-code for generating the viewpoints have been elaborated in Algorithm 1.

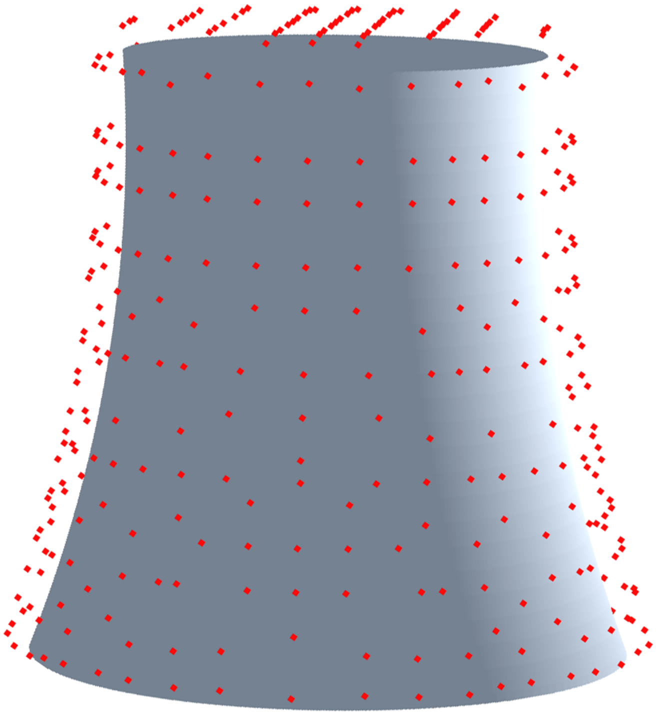


Figure 4.9. Cooling tower model with the view points generated (in red).

4.3 Path Generation

Once, the viewpoints are generated, paths are generated by connecting the viewpoints. The paths are generated by selecting one viewpoint after and hence forming a series of viewpoints which creates the path. The next viewpoint selection is done based on heuristic function. The heuristics depends on the distance to next viewpoints, new area covered (unseen area covered by the sensor), and the turn angle. Given *n* UAVs, the algorithm generates *n* paths, one for each UAV.

Algorithm 1 Pseudo-code for generating the viewpoint point cloud.

```
Input: Input Model $M$, Safe flight distance $d$
Output: Viewpoints $vps$
Initialize $vps \leftarrow empty\ list$
voxels $\leftarrow$ convert the input model $M$ to octree
for each $v \in$ voxels do
    $\vec{n} \leftarrow$ normal at $v$
    $vp = p + d * \vec{n}$
    vps.insert(vp)
end for
$vps \leftarrow Voxel\_Grid\_Filter(vps)$
for each $vp \in$ vps do
    $(n_x, n_y, n_z) \leftarrow$ normal at $vp$
    if $n_x \neq 0$ & $n_y \neq 0$ then
        View Direction, $\vec{v} \leftarrow (-n_x, -n_y, 0) = -1 * (n_x, n_y, n_z) - (n_x, n_y, 0)$
    else
        View Direction, $\vec{v} \leftarrow null$
    end if
    $d_i \leftarrow$ intersection distance of $\vec{v}$ with $M$
    if $d_i < d + sizeofUAV$ then
        vps.remove(vp)
    end if
end for
return $vps$
```

4.3.1 Viewing Cone Radius

Before, beginning the path generation process, the radius and the region of the viewing cone based on the field of view of the camera is computed. The camera is assumed to have equal field of view along the horizontal and vertical directions. Let the field of view be θ. The camera is at a distance d (safe flight distance) from the surface. Now the radius of the viewing cone R_C is computed as

$$R_C = d * atan(\theta/2). \tag{4.3}$$

The radius of the viewing cone computed initially since the later steps depend on this value. The viewing cone radius R_C has been illustrated in Figure 4.10.

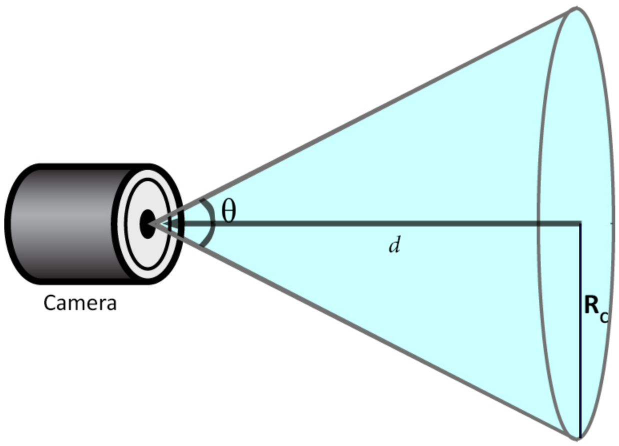


Figure 4.10. The viewing cone of the camera with radius R_C, field of view θ and safe flight distance d.

4.3.2 Start Point Selection

The path generation process needs some initial viewpoints to which more viewpoints are added to generate the path. The initial start points are selected such that the start points are as close possible to the ground plane ($xy-$plane), so that it is easy for the UAV to start the inspection process as soon as it takes off from the ground. In order to do this the viewpoints generated are sorted in the increasing order of their $z-value$. If the system has only one UAV, the viewpoint with the least $z-value$ is chosen as the start point. The start point chosen for one UAV scenario is shown in cyan in Figure 4.11. In the case of more than UAV, we need to multiple start points. It is ideal if the start points of the UAVs are not too close so that when the path generation is initiated the paths tend to cover different regions of the model. To space out the start points to different regions of the model depending on the geometry of the model, a parameter *start point separation, s* is used. The second start point should be at least at a distance of $s.R_C$ away from the first view point and also have the least $z-$value. This process is iteratively repeated for getting n start points when we have n UAVs in the system. This ensures that the start points are not close to one another. The pseudo code for selecting the viewpoints has been briefed in Algorithm 2.

Algorithm 2 get_start_positions - Function to get start positions

```
Input: viewpoints $vps$, viewing_radius $R_C$, start point separation $s$, number of UAV $n$
Output: Start_points $s\_pts$
Initialize $s\_pts \leftarrow empty\ list$
Initialize counter ← 1
$vps \leftarrow$ sort $vps$ in the order of $z$-coordinate
$s\_pts$.insert(vps(0))
for counter = 2 to n do
    for each $vp \in$ vps do
        if distance $(vp, spts(counter-1)) < s * R_C$ then
            $s\_pts$.insert(vp)
        end if
    end for
end for
return $s\_pts$
```

4.3.3 Path Expansion

Now that we have the start points, these points needs to be expanded into a path which the UAVs can traverse to perform the inspection. The path expansion is done by finding the next best viewpoint from the viewpoint point cloud and repeating this process iteratively until the entire structure is covered. This process of next best viewpoint selection is done using three heuristics:

- The distance between two viewpoints (d_v)
- The angle between the two viewpoints (γ)
- The coverage (C)

4.3.3.1 Distance between the viewpoints d_v When a new viewpoint is selected, it is seen to it that the new viewpoint is close to the current viewpoint. It is ideal if the viewpoints are closer since when the two consecutive viewpoints are closer the probability of missing out some regions of the structure is less. In other, if the UAVs moves farther away from one step to another, it is highly possible that the UAVs missed to cover some region in between. Hence, in the process of selecting the next best viewpoint it is seen to it that the viewpoints are closer.

4.3.3.2 Angle between the viewpoints γ It is well known fact that, it is easy for the UAV to trace the path exactly if the path is less complex and does not have many turns. It is better if the paths are more or less streamlined. In order to achieve this, it seen to it that the new angle of heading of the UAV is close to the current angle of heading. This ensures that the UAV keeps moving in its current direction of propagation. So, while selecting the next best viewpoint, we minimize the angle between the line joining the previous viewpoint with the current viewpoints and the angle between the current viewpoint and the next viewpoint. When the UAV starts from the start point, the angle is computed with respect to the $xy-$plane, since no prior information about the heading angle is available.

4.3.3.3 Coverage C Coverage is the numerical measure of the surface area seen by the camera in the UAV from a particular viewpoint. Coverage is measured by counting the number of octree voxels that is visible from a viewpoint. Octree voxels seen from some other viewpoints previously are not counted, i.e only the unseen octree voxels are accounted. While selecting the next best viewpoint, the next viewpoint should have a good coverage, so that the UAV moves towards the regions which are not covered. The coverage from a viewpoint is measured using Occlusion Culling.

Now, from the starting point of the UAVs, we need to find the next best viewpoint for each UAV based on the heuristics proposed above. The search for the next best viewpoint is made only within a limited space. In order to define the search space around the current viewpoint, we use another parameter named *viewpoint search number, v*. This number defines the search radius within which the next viewpoint is looked for. Only the viewpoints within a distance of $v * R_C$ are considered. Defining the search radius in terms of the radius of the viewing cone helps in the easily visualization while deciding the parameter. This parameter is decided based on the geometry of the model and the sampling density of the viewpoints point cloud. In Figure 4.11, the search sphere is marked with a green circle with the UAV currently being located a the viewpoint marked in green.

The three heuristics are computed for all the viewpoints within the sphere. Firstly, all the viewpoints with zero coverage are eliminated. This eliminates the chance of the UAV moving to point already visited by another UAV. Now, for the subset of viewpoints a heuristic value is computed based on the heuristic function H defined as

$$H(d_v, \gamma, C) = C - (2\gamma) - 7d_v. \tag{4.4}$$

A linear heuristic function has been chosen since it is easy to tune to the weight for each of the parameter and arrive at the optimal value sooner than having a non-linear function. The coverage is added since the coverage needs to be maximized and the angle and the distance are subtracted since they need to be minimized. The heuristic value is computed for all the viewpoints in the vicinity and the one with maximum value is chosen as the next viewpoint. In Figure 4.11, the possible next viewpoints are marked in yellow, with the selected viewpoint based on the heuristics is marked connected with an orange line.

4.3.3.4 Tuning the heuristic function The coefficients values for the heuristic function were generated using a sample set of connected points generated using a model of a regular shapes. Models of cylinder, cone and cuboid were used for this purpose. These geometric primitives were chosen based on the fact that any 3D model can be constructed just by the union and the intersection of these geometric primitives [43]. Since the geometry is well defined a series of connected viewpoints can be generated based on the shape properties. Now, these points were selected by iteratively validating the function with a series of values in iteration and the best values which maximizes the difference of the heuristic value for the next viewpoint selection. This set of values is averaged with the current set of values and this process is repeated until the path is completed. This process is again repeated for all the primitive models. The coefficient values resulting in the end is chosen as the actual coefficients. Finally the coefficients were reduced to their least integers to simplify the function.

The line connecting the current viewpoint with the selected viewpoint, is checked for collision with the structure. If there is an overlap, that viewpoint is not considered and the next best is picked. This process continues for each and every UAV starting from its start position and iteratively building its path one step at a time.

This process of path expansion is continued until there is at least one single view point with coverage greater than 0, in the viewpoint search vicinity. For each of the path of the UAV, this criteria is used to stop the path growing process. The path generation process completely stops when paths of all these UAVs meet this criteria. Finally, after the exploration process, we get *n* paths for *n* UAVs to follow.

Figure 4.11 shows a close up view of the cooling tower model in the path generation process. The green viewpoint is the current viewpoint and the next viewpoint should be with in the range of the green circle. All the next possible viewpoints are marked in yellow. It can be clearly observed that there only two viewpoints with lesser angle and out of those 2, the closest one is selected. The path selected is shown is orange.

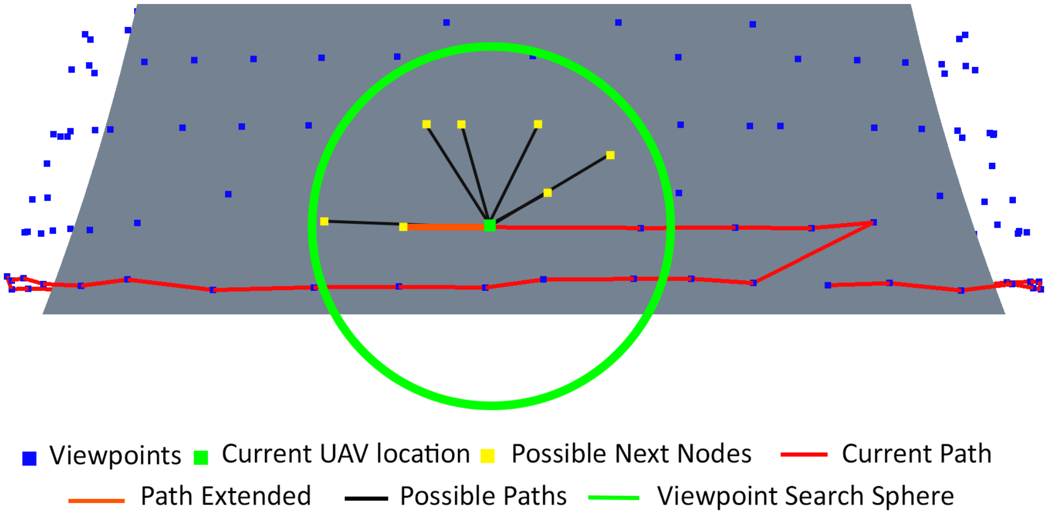


Figure 4.11. Close up view of the cooling tower; The current location of the UAV is marked in green and the path generated so far is marked in red. The UAV can move to any of the viewpoints marked in yellow and the path chosen based on the heuristics is marked in orange.

4.4 Non-Convexity of the Heuristic Function

In this section the non-convexity of the heuristic function generated is analyzed. The convexity of the heuristic function generated is tested in order to find out whether the path generated by the heuristic function is the optimal path or not. It is well known that if that function used is non-convex, it is not feasible to attain the global mimima and hence achieving the optimal solution may not be an option. In order to check the convexity, we compute the Hessian Matrix of the function.

The Hessian Matrix for any function can be computed as

$$Hess\ f(x,y,z) = \begin{bmatrix} \frac{\partial^2 f}{\partial x^2} & \frac{\partial^2 f}{\partial x \partial y} & \frac{\partial^2 f}{\partial x \partial z} \\ \frac{\partial^2 f}{\partial y \partial x} & \frac{\partial^2 f}{\partial y^2} & \frac{\partial^2 f}{\partial y \partial z} \\ \frac{\partial^2 f}{\partial z \partial x} & \frac{\partial^2 f}{\partial z \partial y} & \frac{\partial^2 f}{\partial z^2} \end{bmatrix} \tag{4.5}$$

The Hessian matrix for the generated heuristic function is

$$Hess\ H(d_v, \gamma, C) = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \tag{4.6}$$

In this case the Hessian matrix is clearly not a positive (semi-)definite and the heuristic function used in not convex. Since the function generated is convex it is unlikely, that the path generated using this function is optimal. It is estimated that the path generated are just good and neat.

4.5 Summary

This section narrated the path planning algorithm along with the pseudo code. The procedure followed for tuning the heuristic function was also explained. Finally, convexity analysis on the function proved that the function used in non-convex and it not feasible to achieve optimal paths for this problem.

CHAPTER 5. EVALUATION AND RESULTS

The results of the research are reported in this chapter. The paths generated for the UAVs to follow given the 3D model of the structure are presented along with detailed analysis on them. Discussions on the path lengths, coverage, scalability and the robustness of the proposed algorithm on various complex structures are also presented. Further analysis on distribution of the coverage amongst the UAVs shows that the paths generated distribute the coverage equally amongst the UAVs. Analysis on how the parameters affect the result and importance of tuning the right parameters has been explained. Finally, the reconstruction and inspection of the structure has been carried out in a simulated environment using the paths generated and the reconstructed models are also shown.

5.1 Results

In this section the paths generated for various structures has been presented. Paths have been generated for models of cooling tower, Boeing 747 airplane and Big Ben. The models are chosen such that they have a real world significance for inspection and also the models shows variations in the geometry. Paths were generated for scenarios with 1, 3 and 5 and the paths have presented visually. In the scenarios where multiple paths are generated, the path are highlighted in different colors.

The parameter used for generating the paths are also presented. For all the scenarios, the sensor parameters has been chosen in accordance with Intel RealSense R200 camera. The camera has field of view of 77^o and it works well in the range of 2 to 3 m. So, the *safe flight distance, d* is also chosen in that range, so that UAV is always at a distance away from the structure at which the camera performs the best in identifying the defects.

5.1.1 Paths for Cooling Tower Model

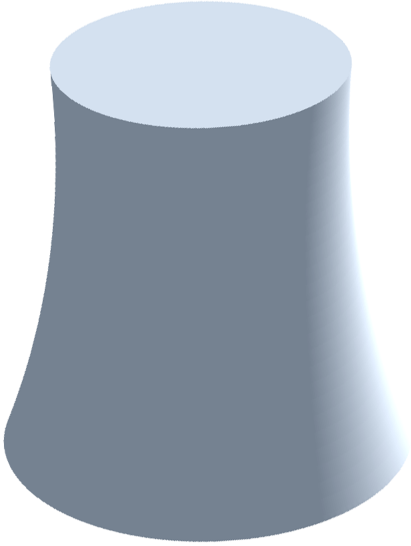


Figure 5.1. Input model of the cooling tower used in the experiment.

5.1.1.1 Cooling Tower - 1 UAV

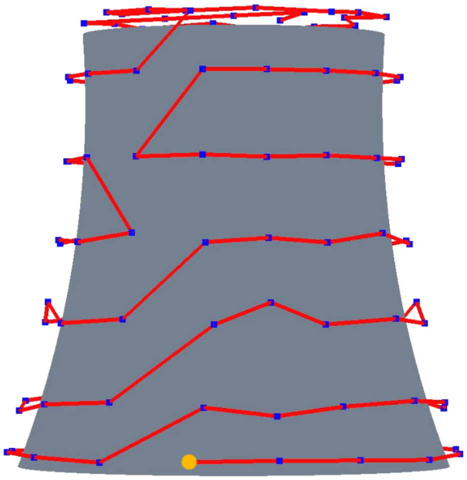


Figure 5.2. Path generated (in red) for 1 UAV for the cooling tower model (side view).

The orange dot marks the starting location.

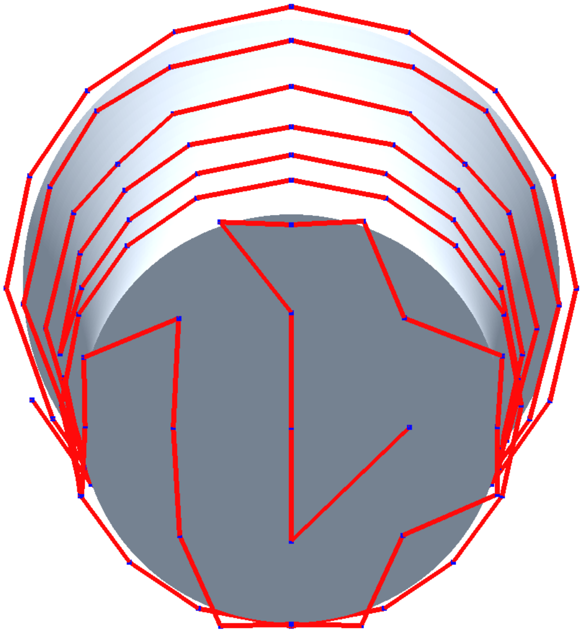


Figure 5.3. Path generated (in red) for 1 UAV for the cooling tower model. (top view)

From the Figures 5.2 and 5.3, it can be observed that the path generated (in red) covers the entire volume of the structure in a spiral way. The path generation starts at the bottom (orange dot) of the tower and the path slowly ascends in a spiral way, until it reaches the top of the tower. The path initially covers the sides of the tower and then moves to the top of the tower, hence giving a complete coverage of the structure. Since, there is only one UAV involved start point separation parameter is not needed. The path generated was able to cover 98.6% of the volume of the tower.

Table 5.1. Parameters used for cooling tower model with 1 UAV.

Parameter	Value	Parameter	Value
Number of UAVs, n	1	Safe flight distance, d (m)	3
Sensor range, θ	77^o	Start point separation, s	-
Viewpoint search number, v	1.5	Angle threshold, α	15^o

5.1.1.2 Cooling Tower - 3 UAVs

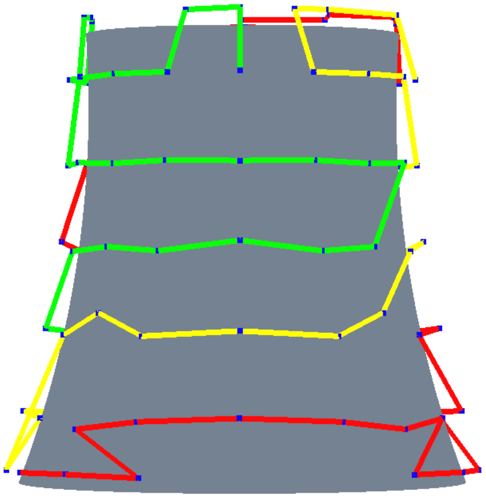


Figure 5.4. Path generated (in red, green and purple) for 3 UAVs for the cooling tower model with the paths in different colors.

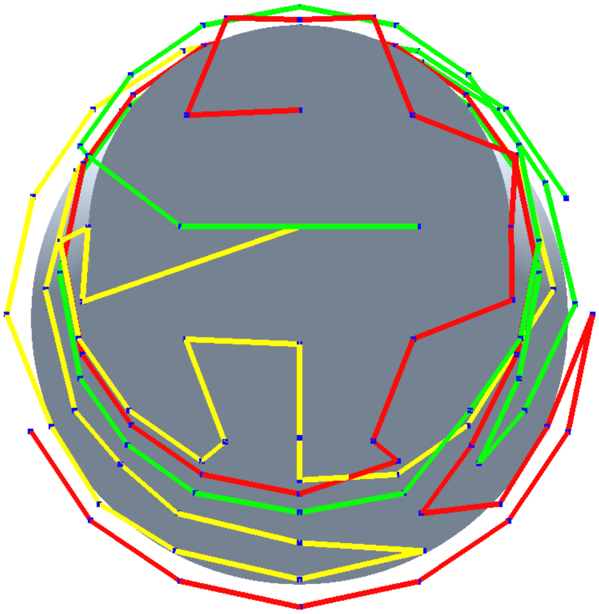


Figure 5.5. Path generated (in red, green and purple) for 3 UAVs for the cooling tower model (top view).

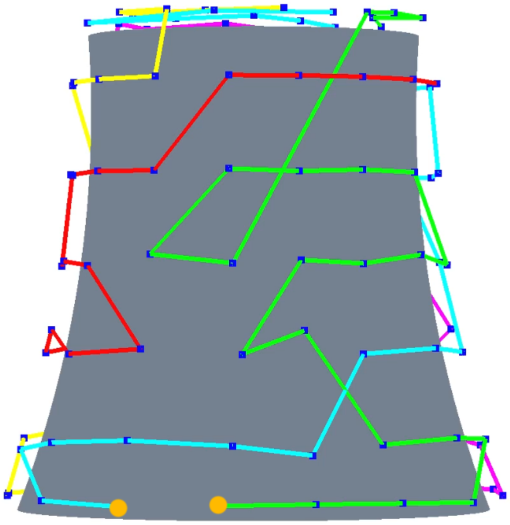
The paths generated (in red, green and purple) for the cooling tower model for 3 UAV is shown in Figures 5.4 and 5.5. In this case, start point separation is used since more than one UAV is involved. The parameter is set such that the points are close the ground plane (marked in orange). Though there are some void spaces between the paths, those areas are still covered by the UAVs due to the wide coverage of the camera attached to the UAV. The path generated was able to cover 98.2% of the volume of the tower. Each of the UAVs had a coverage of 982, 1027 and 998. The coverage's are almost close signifying the equal distribution of the task.

Table 5.2. Path generated for 5 UAVs (in red, green, yellow, cyan and purple) for the cooling tower model with the paths represented in different colors.

Parameter	Value	Parameter	Value
Number of UAVs, n	3	Safe flight distance, d (m)	3
Sensor range, θ	77^o	Start point separation, s	4.3
Viewpoint search number, v	1.5	Angle threshold, α	15^o

Table 5.3. Parameter used for cooling tower model with 3 UAV.

5.1.1.3 Cooling Tower - 5 UAVs



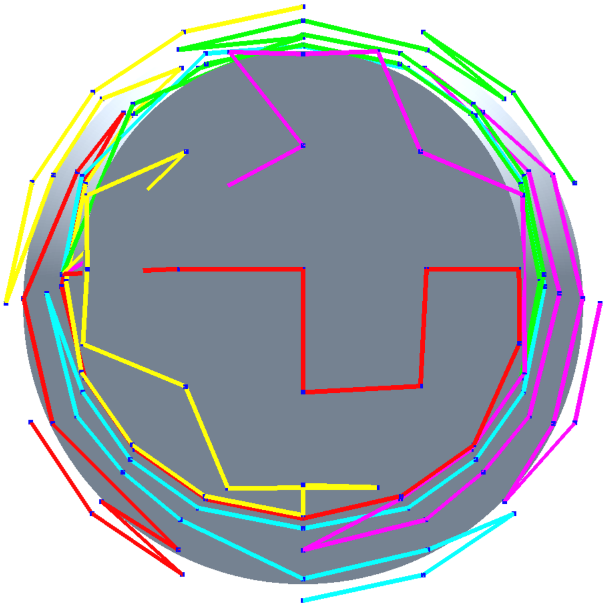


Figure 5.6. Path generated (in red, green, yellow, cyan and purple) for 5 UAVs for the cooling tower model (top view).

In the Figures **??** and 5.6, paths for 5 UAVs are shown (in red, yellow, green, cyan, purple). The UAVs start the traversal at different regions in the model and covers different regions in the structure. It can be seen that an UAV approached a region covered by another UAV it tends to take a turn to move towards an uncovered region. Also, the UAV keeps moving until it has uncovered regions at its reach. The large line in the green path is because that there was not any uncovered regions close by, so it make a big jump to a reachable uncovered region. This ensures that the least area is missed. The path generated was able to cover 98.9% of the volume of the tower.

Table 5.4. Parameter used for cooling tower model with 5 UAV.

Parameter	Value	Parameter	Value
Number of UAVs, n	5	Safe flight distance, d (m)	3
Sensor range, θ	77^o	Start point separation, s	4.3
Viewpoint search number, v	1.5	Angle threshold, α	15^o

5.2 Paths for Boeing 747 Airplane Model

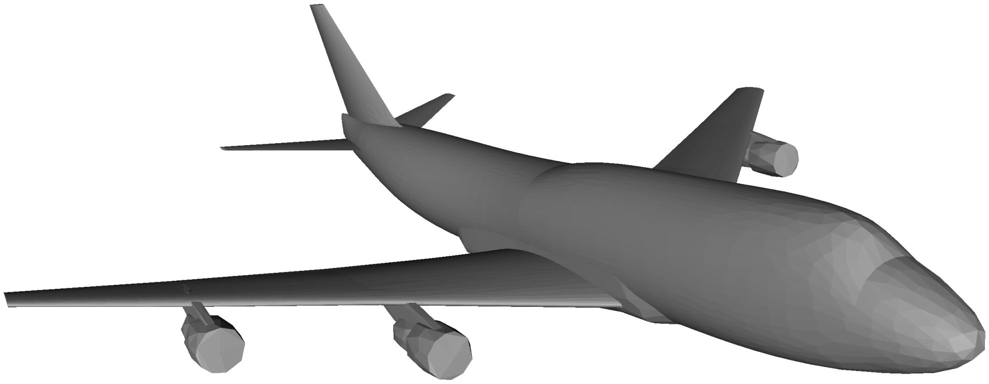


Figure 5.7. Input model of the Boeing 747 airplane used in the experiment.

5.2.0.1 B747 Airplane - 1 UAV

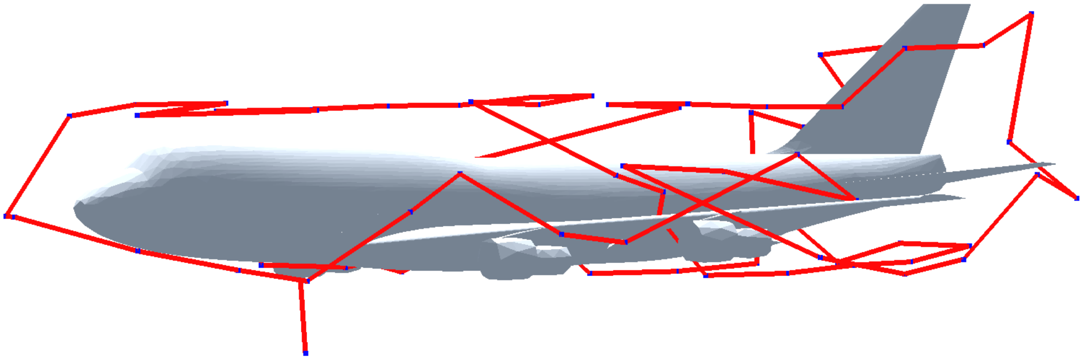


Figure 5.8. Path generated (in red) for 1 UAV for the B747 model (side view).

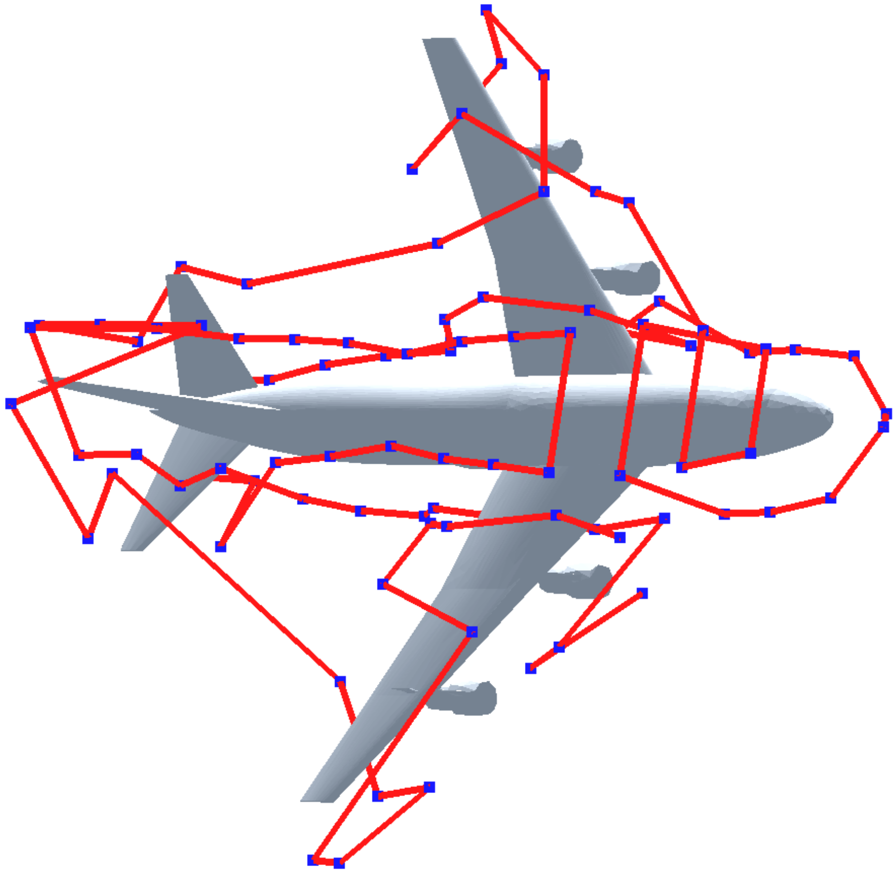


Figure 5.9. Path generated (in red) for 1 UAV for the B747 model (top view).

The path generated (in red) for inspecting the UAV using 1 UAV, has been shown in Figures 5.8 and 5.9. The path starts close to the bottom of the airplane near one of the engines and it grows subsequently. The path goes around the wings, the engines, the fuselage and the rudder, and hence covering the entire airplane structure. Though the path generation started close to the engine in the right wing, the heuristics defined was able to take the path around the entire airplane even around the left wing and hence giving a complete view of the airplane. The path generated was able to cover 97.8% of the volume of the airplane.

Table 5.5. Parameter used for B747 airplane model with 1 UAV.

Parameter	Value	Parameter	Value
Number of UAVs, n	1	Safe flight distance, d (m)	3.5
Sensor range, θ	77^o	Start point separation, s	-
Viewpoint search number, v	1.1	Angle threshold, α	15^o

5.2.0.2 B747 Airplane - 3 UAVs

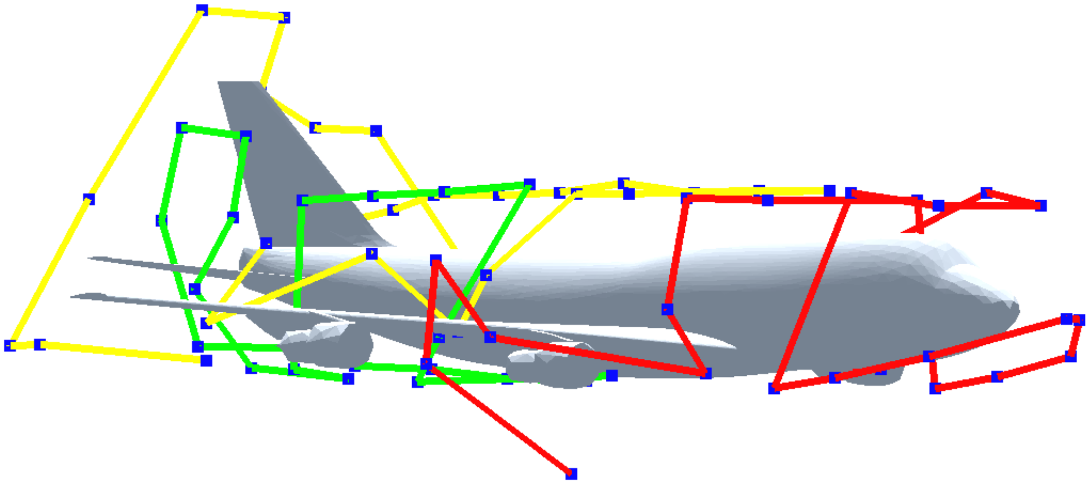


Figure 5.10. Path generated (in red, yellow and green) for 3 UAVs for the B747 airplane model (side view).

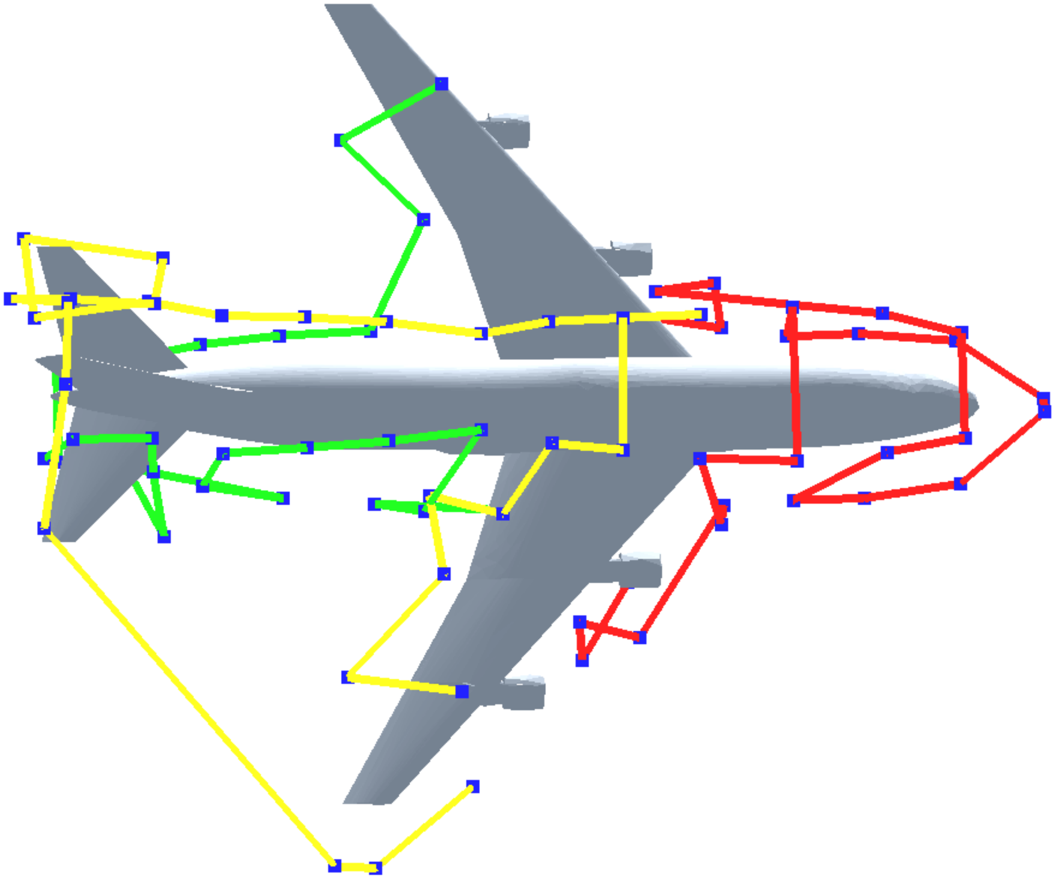


Figure 5.11. Path generated (in red, yellow and green) for 3 UAVs for the B747 airplane model (top view).

In this case with 3 UAVs to inspect the B747 airplane, the start point separation has been chosen such that the start points are close to different parts of the airplane. The 3 paths generated for this model has been shown in Figures 5.10 and 5.11. In this case, the start points are placed close to the nose tip, the engine and the elevators of the airplane. Also, in this case it can be clearly seen how the UAVs cover different regions of the airplane. One UAV covers the region close to the nose of the plane, another close to the middle and the last one predominantly near the rudder. The path generated was able to cover 96.3% of the volume of the airplane, some regions on the wings of the airplane were missed.

Table 5.6. Parameter used for B747 airplane model with 3 UAVs.

Parameter	Value	Parameter	Value
Number of UAVs, n	3	Safe flight distance, d (m)	3.5
Sensor range, θ	77^o	Start point separation, s	3
Viewpoint search number, v	1.1	Angle threshold, α	15^o

5.2.0.3 B747 Airplane - 5 UAVs

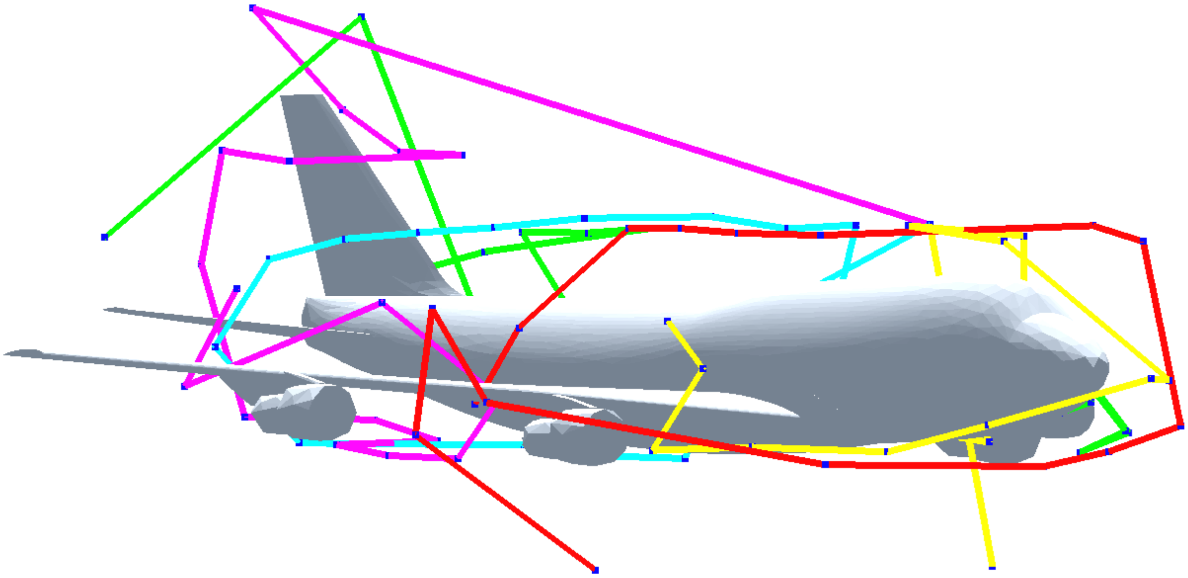


Figure 5.12. Path generated (in red, green, yellow, cyan and purple) for 5 UAVs for the B747 model (side View).

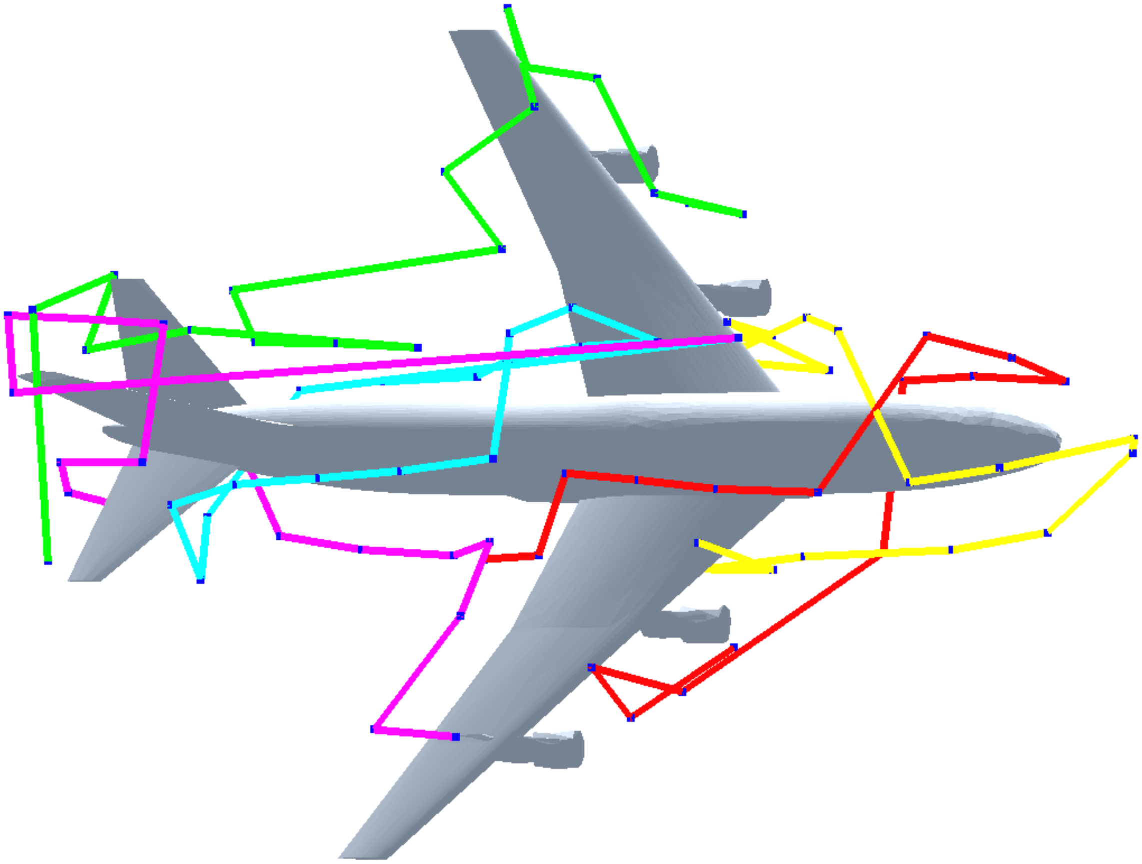


Figure 5.13. Path generated (in red, green, yellow, cyan and purple) for 5 UAVs for the B747 model (top View).

The paths generated (in red, yellow, green, cyan, purple) for 5 UAVs for inspecting the B747 airplane model gives a total coverage of 97.4%. The start point separation parameter has been selected with a high value of 3, since it was ideal to have the start points for the different UAVs close to the various parts of the airplane. In the Figure 5.12 and 5.13 , it can be seen that one UAV starts close to the engine of the airplane, another close to the wing of the plane and so on. Also, it can be seen that the many of the UAVs in this case tends to make sharp turns since having 5 UAVs leads to the region around an UAV being covered by other UAVs and hence the UAVs are forced to make sharp turn approaching towards a region which was not covered earlier by other UAVs.

Table 5.7. Parameter used for B747 airplane model with 5 UAVs.

Parameter	Value	Parameter	Value
Number of UAVs, n	5	Safe flight distance, d (m)	3.5
Sensor Range, θ	77^o	Start Point separation, s	3
Viewpoint search number, v	1.1	Angle Threshold, α	15^o

5.3 More Results

In this section, paths generated for some more models (Big Ben and Windmill) are shown along with the parameters used for generating them.

5.3.1 Paths for Big Ben model

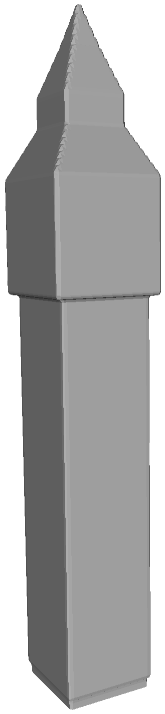


Figure 5.14. Input model of the Big Ben used in the experiment.

5.3.1.1 Big Ben - 1 UAV

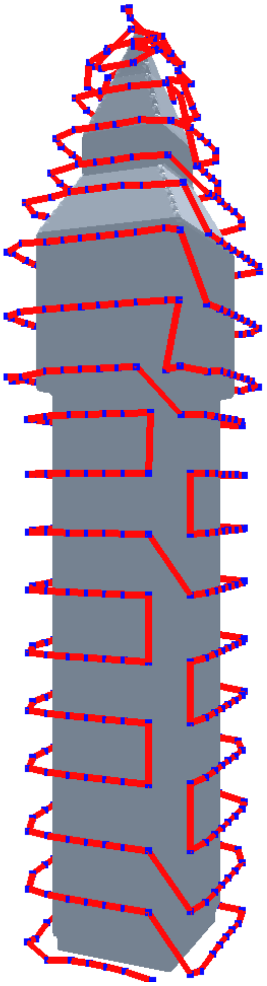


Figure 5.15. Path generated (in red) for 1 UAV for the Big Ben model. Coverage: 99.7%

Table 5.8. Parameter used for Big Ben model with 1 UAV.

Parameter	Value	Parameter	Value
Number of UAVs, n	1	Safe flight distance, d (m)	3
Sensor Range, θ	77^o	Start Point separation, s	-
Viewpoint search number, v	1	Angle Threshold, α	15^o

5.3.1.2 Big Ben - 3 UAVs

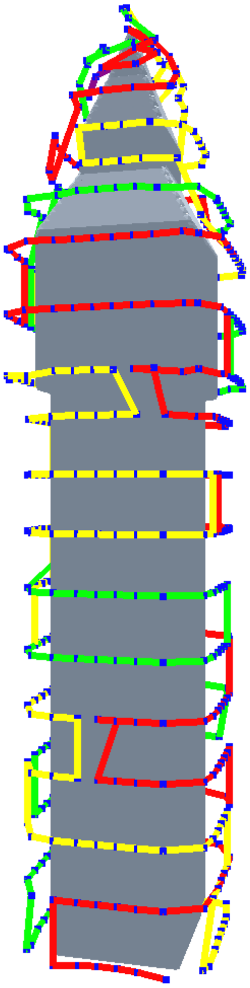


Figure 5.16. Path generated (in red, yellow and green) for 3 UAVs for the Big Ben model.

Coverage: 99.7%

Table 5.9. Parameter used for Big Ben model with 3 UAVs.

Parameter	Value	Parameter	Value
Number of UAVs, n	3	Safe flight distance, d (m)	3
Sensor Range, θ	77^o	Start Point separation, s	2
Viewpoint search number, v	1	Angle Threshold, α	15^o

5.3.1.3 Big Ben - 5 UAVs

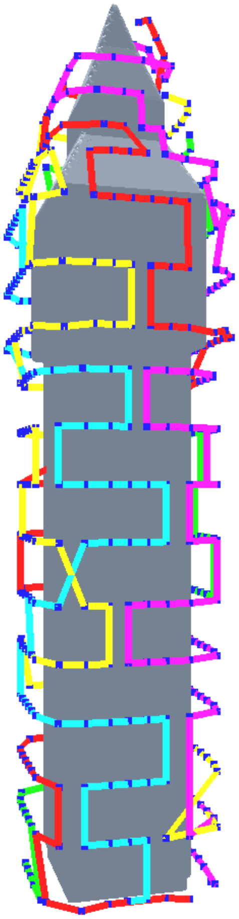


Figure 5.17. Path generated (in red, yellow, green, cyan and purple) for 5 UAVs for the Big Ben model. Coverage: 99.7%

Table 5.10. Parameter used for Big Ben model with 5 UAVs.

Parameter	Value	Parameter	Value
Number of UAVs, n	5	Safe flight distance, d (m)	3
Sensor Range, θ	77^o	Start Point separation, s	2
Viewpoint search number, v	1	Angle Threshold, α	15^o

5.3.2 Paths for Windmill model

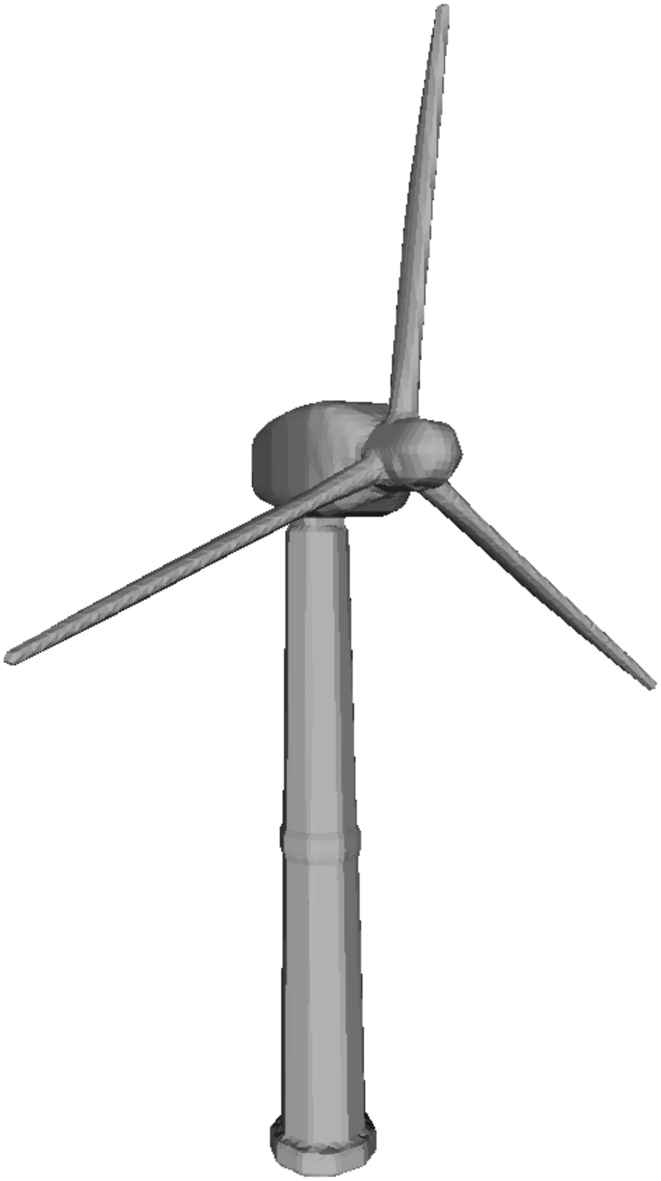


Figure 5.18. Input model of the Windmill used in the experiment.

5.3.2.1 Windmill - 1 UAV

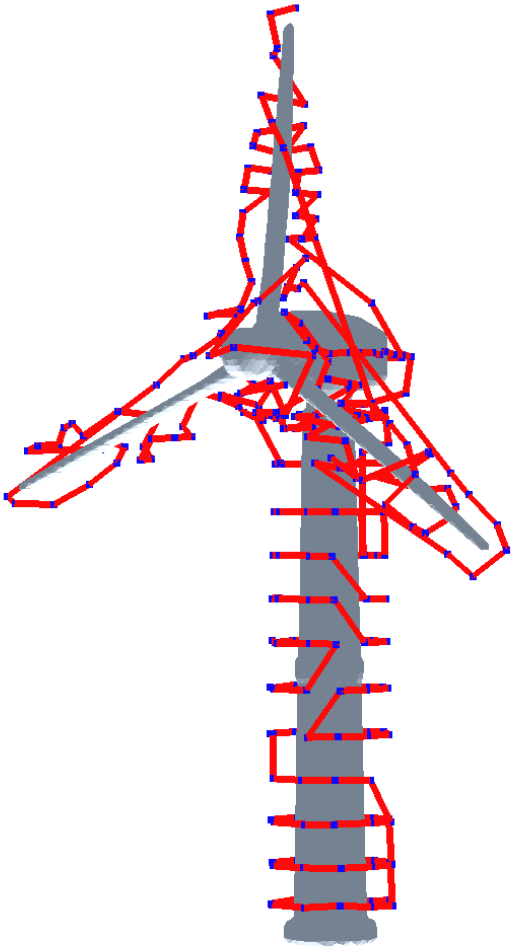


Figure 5.19. Path generated (in red) for 1 UAV for the Windmill model. Coverage: 98.7%.

Table 5.11. Parameter used for Windmill model with 1 UAV.

Parameter	Value	Parameter	Value
Number of UAVs, n	1	Safe flight distance, d (m)	3
Sensor Range, θ	77^o	Start Point separation, s	-
Viewpoint search number, v	1	Angle Threshold, α	15^o

5.3.2.2 Windmill - 3 UAVs

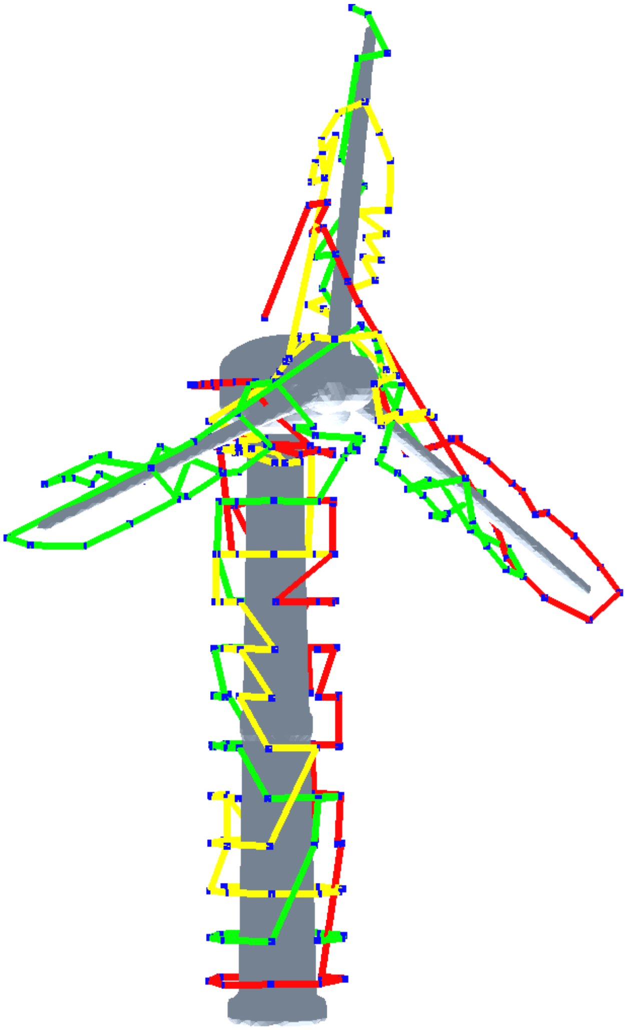


Figure 5.20. Path generated (in red, yellow and green) for 3 UAVs for the Windmill model. Coverage: 98.7%.

Table 5.12. Parameter used for Windmill model with 3 UAVs.

Parameter	Value	Parameter	Value
Number of UAVs, n	3	Safe flight distance, d (m)	3
Sensor Range, θ	77^o	Start Point separation, s	1
Viewpoint search number, v	1	Angle Threshold, α	15^o

5.3.2.3 Windmill - 5 UAVs

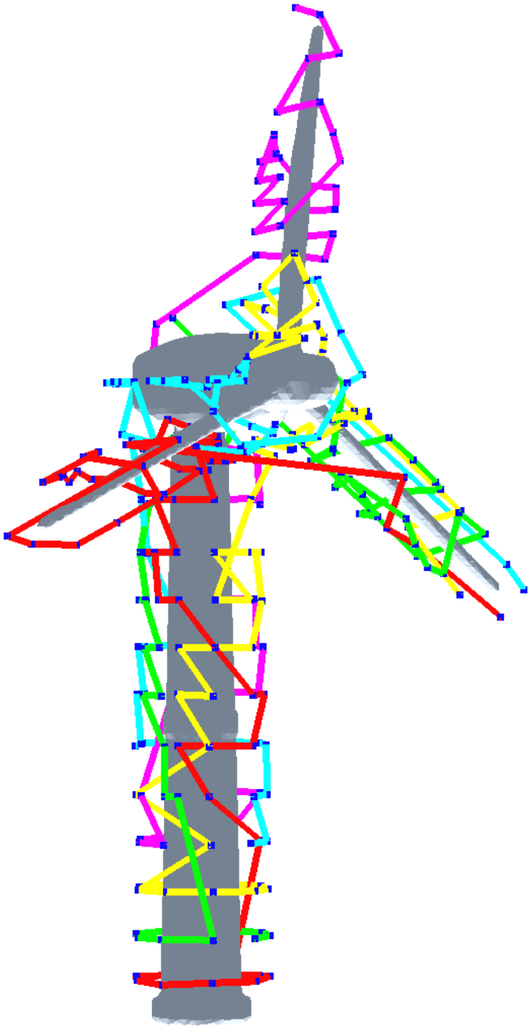


Figure 5.21. Path generated (in red, yellow, green, cyan and purple) for 5 UAVs for the Windmill model. Coverage: 99.1%.

Table 5.13. Parameter used for Windmill model with 5 UAVs.

Parameter	Value	Parameter	Value
Number of UAVs, n	5	Safe flight distance, d (m)	3
Sensor Range, θ	77^o	Start Point separation, s	1
Viewpoint search number, v	1	Angle Threshold, α	15^o

5.4 Performance with multiple-UAVs

This section elaborates the performance of the algorithm for different number of UAVs. A key requirement of a multi-agent system is that the task is distributed uniformly amongst the agents. The proposed algorithm accommodates this requirement and UAVs have almost equal coverage and path lengths. The following plots are based on the outputs of the cooling tower model with 1, 3 and 5 UAVs as shown in Figures 5.2, 5.4, **??**.

5.4.1 Coverage

A key feature of the algorithm is that the coverage task is distributed equally amongst the UAVs. The coverage of each of the path generated is almost the same.

In the case of the cooling tower structure, there were 3062 voxels in the structure and 3007 of the voxels were covered when 3 UAV were used. If the total coverage of 3062 has be divided equally to three parts, we get 1020. The three path generated had coverage of 982, 1027, and 998. This illustrates the algorithms capability to equally divide the task. Also, even in the case of 5 UAVs the coverage of each path should ideally be 612. The paths generated coverage of 607, 586, 598, 637, 600. All the values are close to 612. This equal distribution of coverage is summarized as a plot in Figure 5.22.

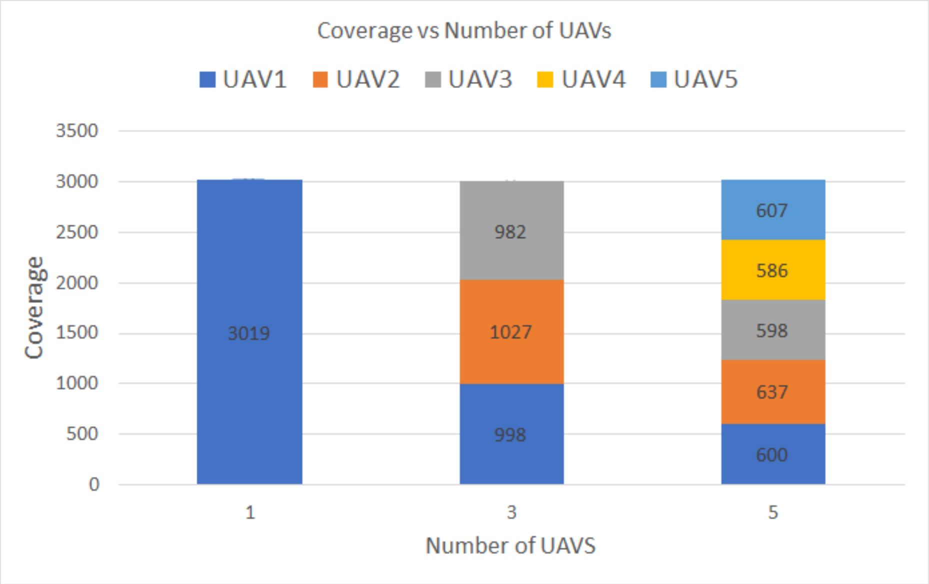


Figure 5.22. Plot showing volume coverage by each UAV in a multi-UAV scenario for the cooling tower structure.

5.4.2 Path Lengths

The algorithm not only distributes the task equally in terms of coverage but also in terms of the path length also. Similar to the coverage, it was seen that the path lengths of the UAVs were pretty close. For the cooling tower model with 3 UAVs the path lengths were 47, 62, 52. Also, a trend of decrease in the total path lengths with the increase in the number of UAVs were observed, due to the disjoint nature of the paths compared to an unified path in single UAV case. The following plots in Figure 5.23 summarizes that the path lengths are almost same for all the UAVs.

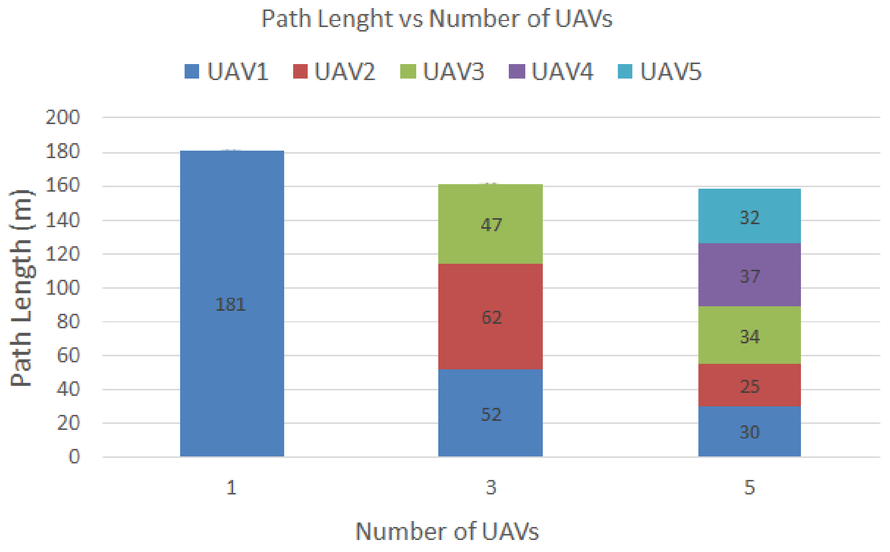


Figure 5.23. Plot showing path length for each UAV in a multi-UAV scenario for the cooling tower structure.

5.5 Effect of the Parameters

In this section, the effect of the parameters: angle threshold and start point separation on the results has been discussed elaborately. The paths generated for various parameters has been depicted along with a brief explanation for the results.

5.5.1 Angle Threshold, α

The angle threshold is a key parameter in generated smooth paths for the UAVs. It helps in reducing the variations in the pitch and smoothens the paths. When the UAVs pitch various beyond this angle threshold, the path planner corrects the direction of heading to bring it back to the horizontal state. Experimentally it was found that the angle threshold value of 15^o yield the best result and in most of the results shown above the same angle threshold was used.

The cooling tower model was experimented with various angle threshold such as 15^o, 25^o, 35^o, 45^o, 55^o and 65^o. The paths generated for these variations in the angle threshold are shown in Figures 5.24 and 5.25.It can be observed that the paths generated for angles 15^o and 25^o are almost the same and smooth. Small threshold angle produce neat and clean paths. When the angle was increased to 35^o, it can be seen that there are few places were the path tends to peak up. From Figure 5.25 it can be seen that, with further increase in the threshold to 45^o, 55^o and 65^o, the number of jumps in the paths also increases and the paths are not smooth any more. Hence, choosing a small angle threshold is essential for getting smooth paths.

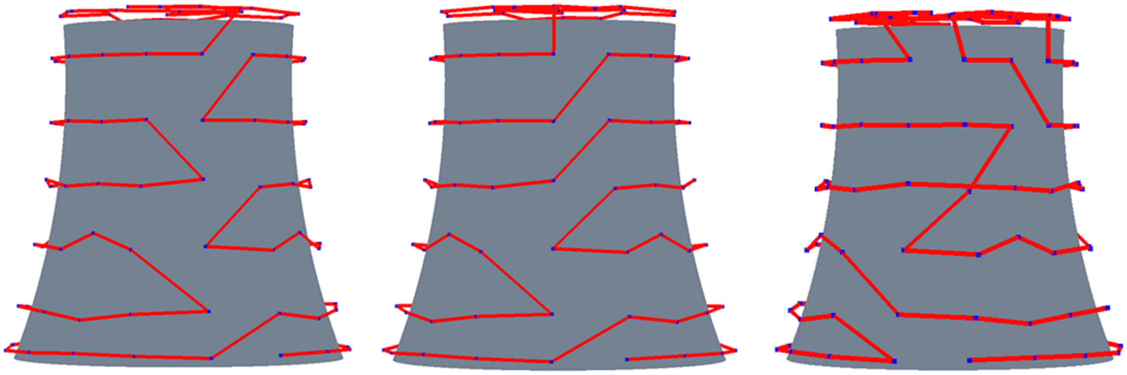


Figure 5.24. Paths generated for the cooling tower model for 1 UAV with the angle thresholds as 15^o, 25^o and 35^o

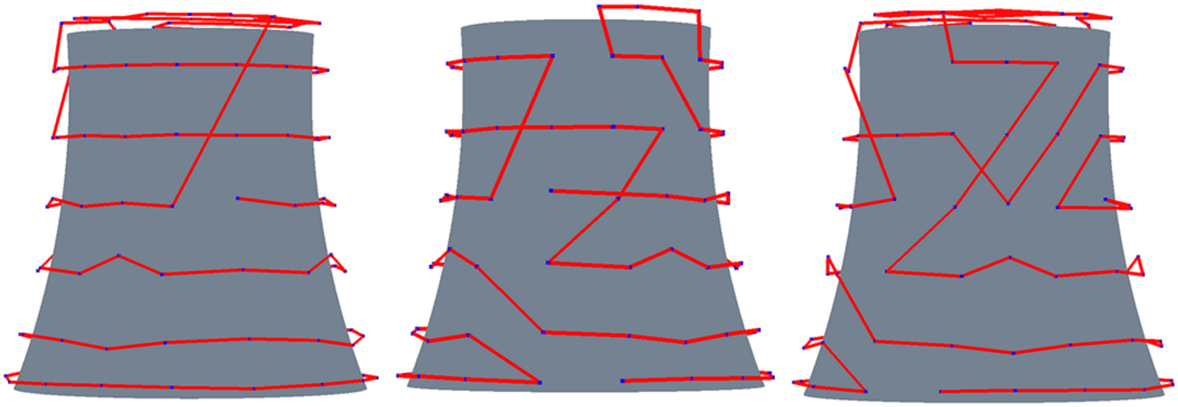


Figure 5.25. Paths generated for the cooling tower model for 1 UAV with the angle thresholds as 45^o, 55^o and 65^o.

5.5.2 Start Point separation, *s*

The Start Point separation, *s* is another important parameter in getting good results. Though the results are not much affected by this parameter, a wrong parameter can spoil the results by yielding lesser coverage. In the following experiment, various values of the start point separation was tried for the cooling tower model with 5 UAVs.

In Figure 5.26, the start Point separation, *s* was set as 2 and the paths generated were smooth and also 98.9% coverage was achieved. When the parameter was slightly increased to 3, the paths generated were completely different from the previous scenario but still the paths were smooth and 98.8% coverage was achieved. Figure 5.27 shows the result generated with the start point separation as 3. Further increase in the parameter to 4 did not yield good result and the coverage was reduced to 84.6%. This clearly shows that the right placement of the starting points is quite essential for achieving complete coverage. In Figure 5.28 the paths generated with start point separation as 4 is shown and many uncovered regions can be clearly seen in the figure. Increasing the parameter to 5 again yielded good result with 98.8% coverage and the paths generated are shown in Figure 5.29. Again, increasing in the parameter to 6, depleted the quality of the results yielding 86.5% coverage only.

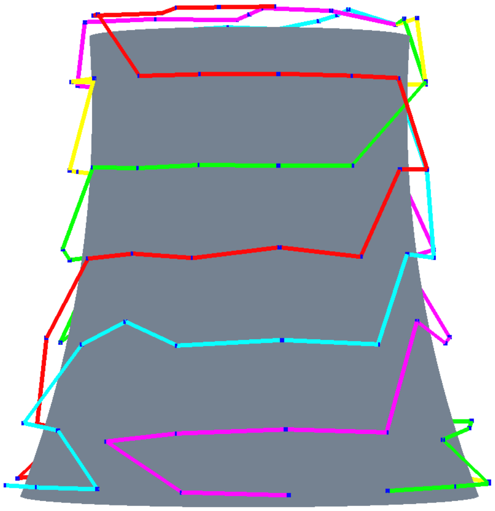


Figure 5.26. Paths generated for 5 UAVs for the cooling tower model with the Start point separation as 2.

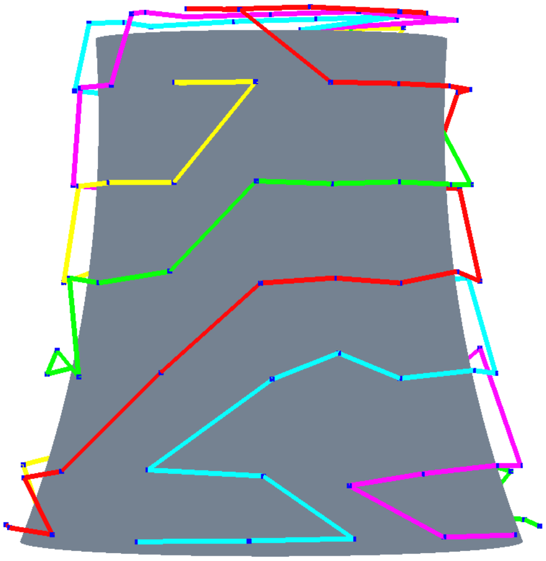


Figure 5.27. Paths generated for 5 UAVs for the cooling tower model with the Start point separation as 3.

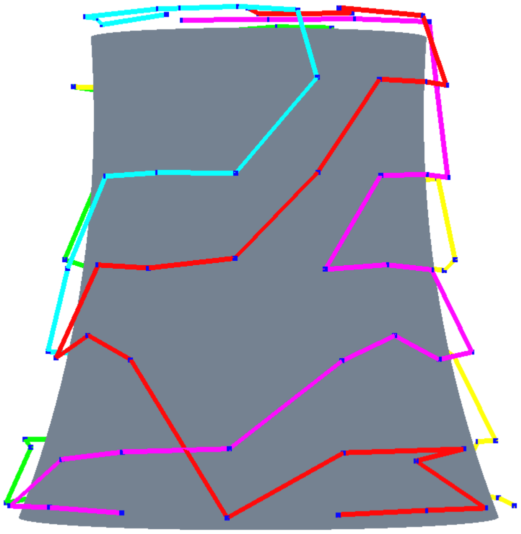


Figure 5.28. Paths generated for 5 UAVs for the cooling tower model with the Start point separation as 4.

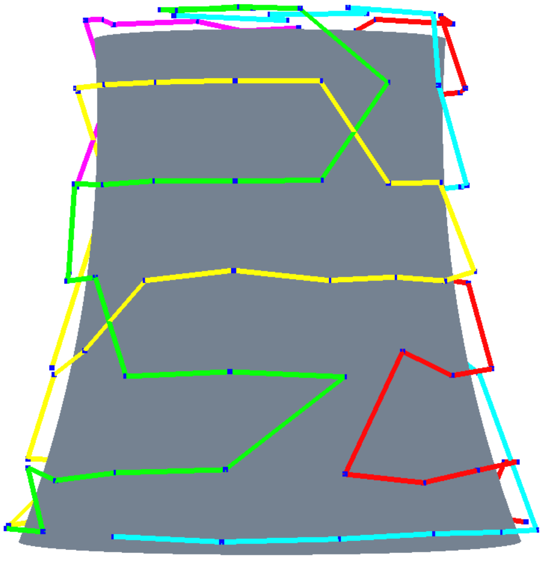


Figure 5.29. Paths generated for 5 UAVs for the cooling tower model with the Start point separation as 5.

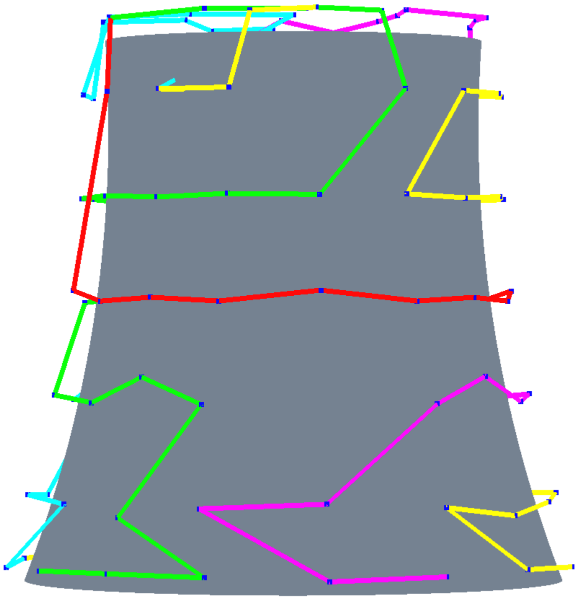


Figure 5.30. Paths generated for 5 UAVs for the cooling tower model with the Start point separation as 6.

From the Figures above with the variation in the start point separation, it can be clearly seen that the start point separation affects the paths generated and the coverage achieved. Also, its not just that one single value of start point separation gives good coverage, but multiple configuration of start points can give complete coverage. At the same time, for some configuration of start points, the paths generated are such that they do not yield coverage. Since the path planner uses the previous waypoint as reference in selecting the next waypoint, selecting the right start point using the right parameter is essential.

5.6 Saturation on number of UAVs

Though the algorithm is scalable in terms of numbers of UAVs, but the quality of the paths generated tends to deplete when more UAVs are deployed relative to the size of the model. In this section, paths are generated for 7, 9, 11 and 13 UAVs for the cooling tower model and the path generated are analyzed. In Figure 5.31, the paths for 7 UAVs are generated and it can be seen that the paths are smooth and most of the paths looks to be stream lined. But when the number of UAVs is increased to 9 as shown in Figure 5.32, the paths are not as streamlined as it was before and many paths tends to just pitch up. With the further increase in the number of UAVs to 11 as in Figure 5.33, almost all the paths pitch up and they are ideal for the UAVs to trace them. So far for all the cases, the coverage was equally distributed amongst the UAVs. But when the count was increased to 13, the paths are pitch up and also few paths have lesser coverage compared to the other paths. In Figure 5.34 (left most figure), the path in grey is looks to be shorted compared to other paths. When 13 UAVs are deployed, each UAV should ideally have around 7.6% of coverage. But the gray path had only 4% of the volume covered, which is less than the expected. From this it can inferred that though the path planner is scalable, it is not ideal to deploy more UAVs relative to the size of the structure. When more UAVs are deployed for a smaller, the planner tends to under perform and the paths pitch up and also tends to unequal distribution of coverage amongst the paths generated. Hence, the planner saturates at a point in terms of number of UAVs deployed since it is not feasible to deploy many UAVs when the model does not require as many UAVs for its inspection.

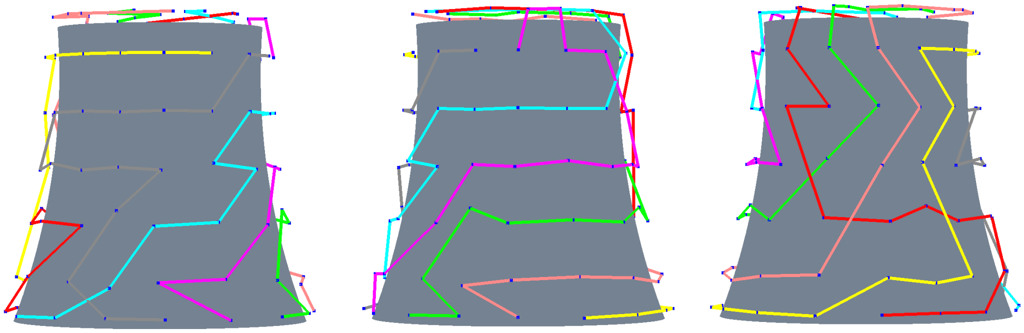


Figure 5.31. Paths generated for the cooling tower model for 7 UAVs in multiple views.

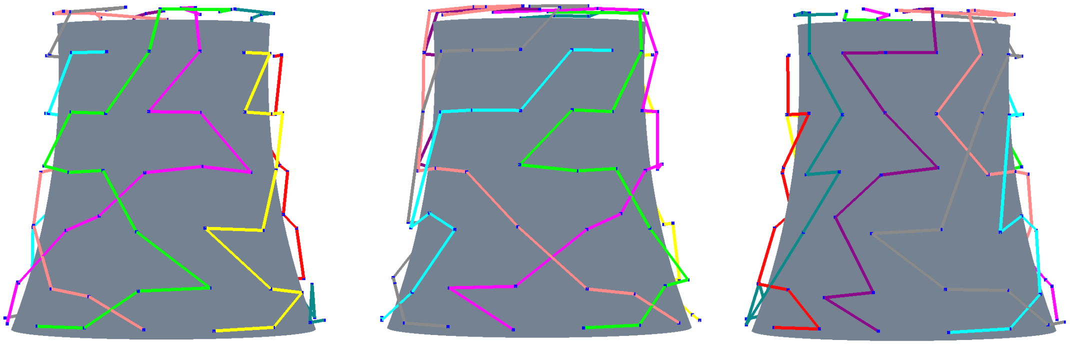


Figure 5.32. Paths generated for the cooling tower model for 9 UAVs in multiple views.

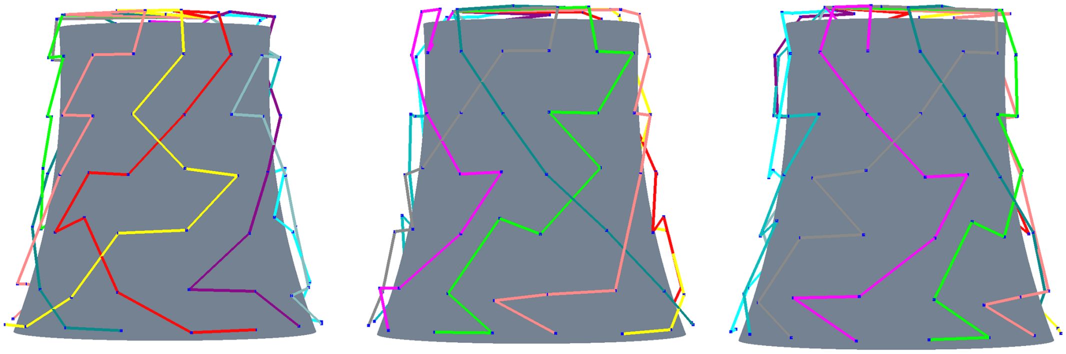


Figure 5.33. Paths generated for the cooling tower model for 11 UAVs in multiple views.

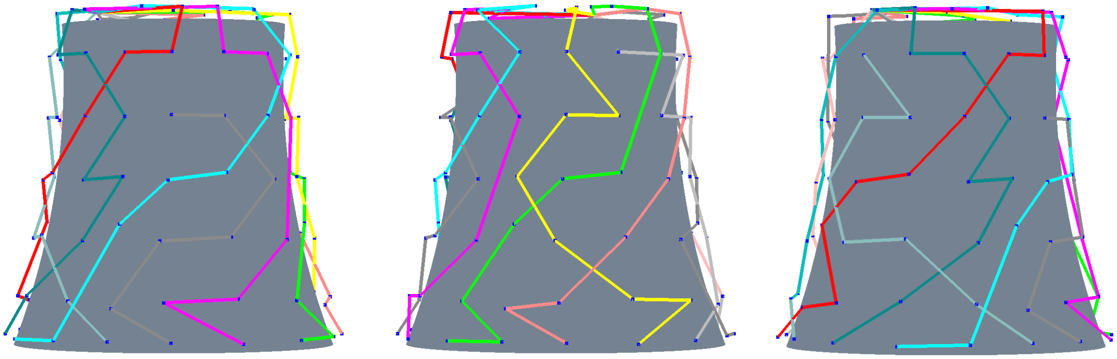


Figure 5.34. Paths generated for the cooling tower model for 13 UAVs in multiple views.

5.7 Coverage Achieved Evaluation

The quality of the paths generated by the algorithm can be quantified by measuring the coverage achieved when the paths generated are used. The estimated coverage for each model for a certain number of UAVs is compared with the coverage achieved when the UAVs trace the path generated. Covered volume is the volume of the rebuilt model when the UAV traces the paths generated and the predicted volume is the volume of the actual model estimated using the trajectories. Actual volume can also be referred as the ideal volume covered by the path. The volume is estimated using the octree grids. The ratio of covered volume and actual volume gives coverage achieved percentage as shown in Equation 5.1.

$$Coverage\ Achieved\% = \frac{CoveredVolume}{PredictedVolume} \times 100. \tag{5.1}$$

Experiments were performed in a simulated environment to measure the coverage achieved. The simulation setup utilizes the packages mentioned in the Chapter 3. The simulated environment consists of the model of the structures imported as a collada file. Gazebo simulator is used for simulating the environment. ROS acts the basic framework for controlling all the UAVs. The communication between ROS and Gazebo for controlling the UAV is achieved using Rotor simulator. The input set of waypoints which the UAV needs to trace is given as a text point with one waypoint per line. These waypoints are given as input to the Rotors simulator and it converts these waypoints to action message which controls the UAV. The Rotors simulator was extended to provide support for flying more than one UAV in the simulated environment. This extension was achieved by running multiple Rotor simulator under different namespace within one environment. MAVROS is already integrated with the Rotors simulator and it used MAVLink protocol for its final communication to the UAV using PX4 framework.

The UAV models given as part of Rotor simulator was extended to add an Intel Realsense camera. AscTec Firefly model provided with the Rotors simulator was customized with the Realsense camera and the point cloud published by it was sent to the Octomap framework for reconstructing the model. In order to have a good tracking of the path limits on the maximum velocity of the UAV was set to $2m/s$. This ensures that the UAV can trace the paths easily.

In the case where multiple UAVs are used, each UAV publishes a point cloud from its camera. Data from all the UAVs are sent to the octomap along with the transform information which are later used for building one consolidated model of the structure.

The coverage achieved and the time taken were measured and the models were reconstructed. The reconstructed models for the various inputs with different number UAVs are shown below.

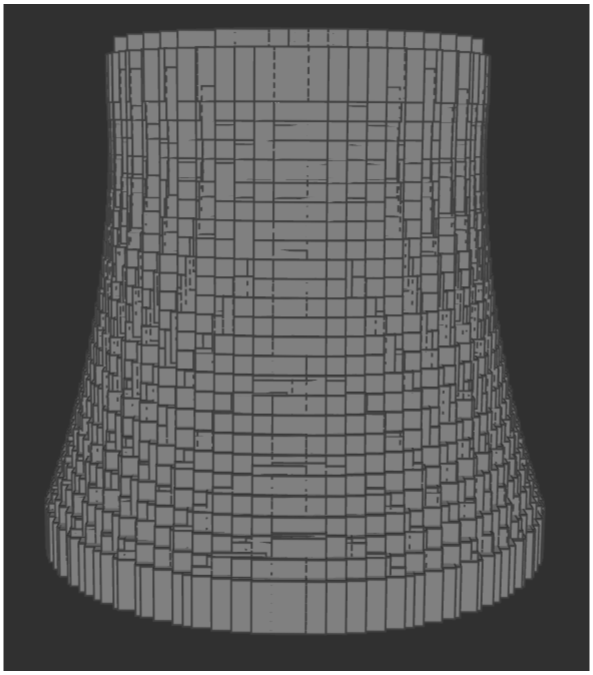


Figure 5.35. Octree reconstructed for the cooling tower model using 3 UAVs.

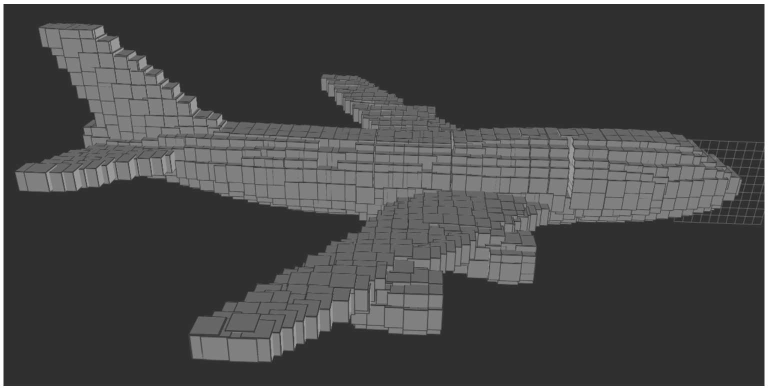


Figure 5.36. Octree reconstructed for the airplane model using 5 UAVs.

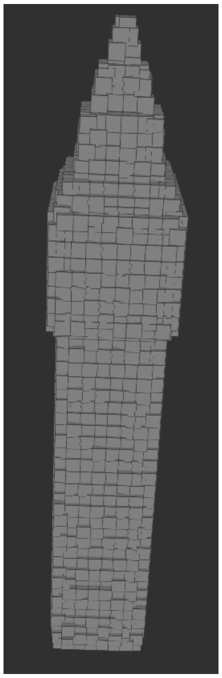


Figure 5.37. Octree reconstructed for the Big Ben model using 3 UAVs.

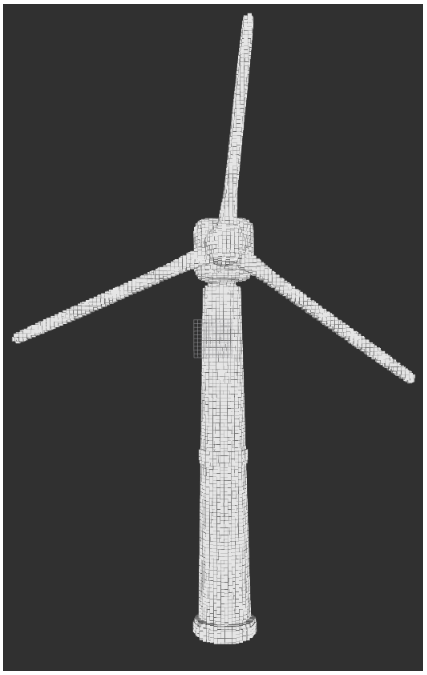


Figure 5.38. Octree reconstructed for the Windmill model using 5 UAVs.

The coverage achieved for these two models with various number of UAVs has been depicted in Table 5.14.

Table 5.14. Coverage achieved for various models with different number of UAVs.

Number of UAVs	UAV 1	UAV 3	UAV 5
Cooling Tower	99.6%	99.4%	99.4%
B747 Airplane	98.6%	99.2%	99.0%
Big Ben	99.9%	99.9%	99.8%
Windmill	99.2%	99.2%	98.9%

The time taken by the UAVs to trace the paths and reconstruct the models has been summarized in Table 5.15.

Table 5.15. Time taken by the UAVs to trace the path for various models with different number of UAVs.

Number of UAVs	UAV 1	UAV 3	UAV 5
Cooling Tower	87s	33s	20s
B747 Airplane	106s	38s	25s
Big Ben	72s	27s	22s
Windmill	82s	34s	22s

5.8 Collision Avoidance

Collision avoidance is a key factor in multi-agent path planning since there can be a collision between the robot while the robots trace the paths. The proposed path planner does not handle the collision between the UAVs directly, but it handles the UAV-UAV collision indirectly. A key factor used in the path generation the coverage of the next viewpoint. If another path is close by, that region would have been already covered by that path. When another approaches that region, the natures of the algorithm tries to repel the path in opposite direction where is more uncovered area. This repelling nature of the path planner helps in avoiding collisions in most of the cases.

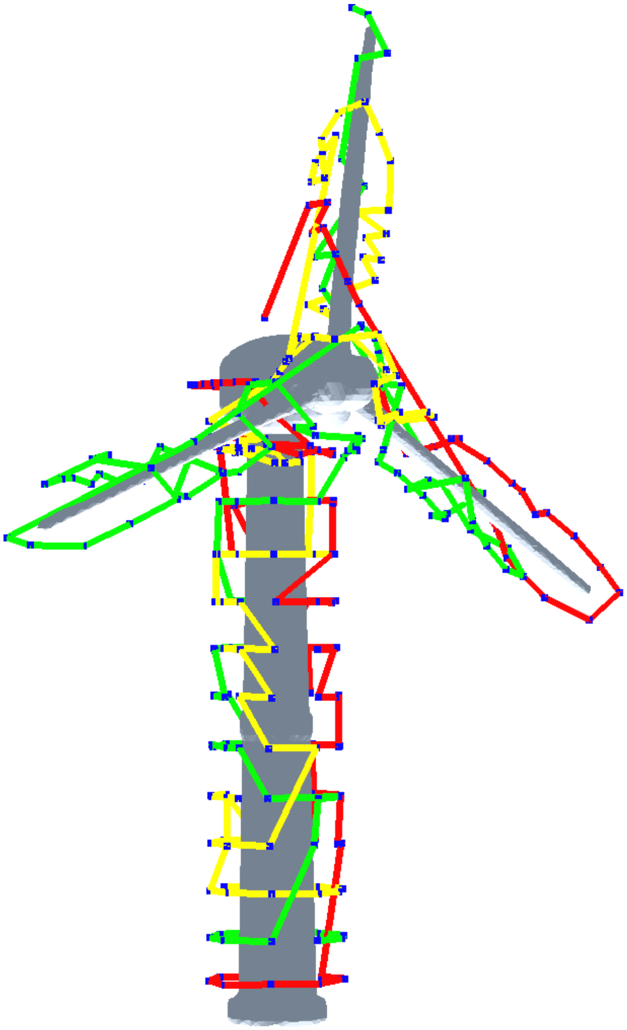


Figure 5.39. Path generated (in red, yellow and green) for 3 UAVs for the Windmill model having multiple intersections with the likelihood of collisions.

Though collisions are avoided to a great extent, the possibility of a UAV-UAV collision is not completely ruled out. From the Figure 5.39, it can be clearly seen that there many intersections in the paths generated for the UAVs to trace. If the UAVs cross the intersection point at the same time, this can lead to collision. From the simulation it was found that there were no collisions for these paths in Figure 5.39, but this does not completely rule out the possibility of having collisions in all the cases. These intersections occurs when there are many UAVs in a small region. As in this case of the windmill, the body of the windmill has a small area and 3 UAVs are deployed. This leads to the intersection. One easy way to fix this is to chose the start point separation appropriately, so that the start points are spread out and each UAV covers different regions of the structure.

The approaches explained does not completely avoid collisions. In order to avoid collisions completely, the path planner needs to be integrated with some existing collision avoidance system, which does real time tracking [44] and prevents collisions by using time scheduling [45].

5.9 Summary

From the results, it can be seen that the algorithm can generate paths for various different cases and its also scalable in terms of number of UAVs. The path planner also tires to achieve maximum and also distributed the task equally amongst the robots. The effect of parameter on the result also has been discussed. The results generated by reconstructing the model in simulation are also presented.

CHAPTER 6. CONCLUSION AND FUTURE WORK

This chapter concludes the thesis presented in this document by summarizing this research. Finally, directions for future research is also presented in this chapter.

6.1 Summary

The this presents a new scalable coverage path planning algorithm for reconstructing and inspecting 3D structures using multiple-UAVs. Firstly, the thesis presents a detailed review of the various coverage path planning methods along with their advantages and disadvantages. The review of the literature clearly shows the lack of a scalable 3D coverage path planner which can be used for any number of UAVs. A detailed analysis of various reconstruction methods used in the mapping and inspection of various environments and structures also has been presented.

The path planner generates path for the UAVs based on the following heuristics: i) Coverage; ii) Change in the angle; and iii) Distance to the next waypoint. These factors ensures that the path generated maximizes the coverage and also accounts for the dynamics of the UAV.

The proposed method is scalable in terms of number of UAVs and it is robust to different structures with variations in the geometry. From the analysis on using the algorithm with different number of UAVs on the same models shows the algorithm generates paths of almost equal lengths of the UAVs. Even the volumes covered by the various UAVs are almost the same. This clearly shows that the algorithm divides the task of coverage equally amongst the UAVs.

Finally, the evaluations on the coverage shows that the algorithm gives maximum coverage on various models which are geometrically different. The models reconstructed using the path generated are both qualitatively and quantitatively similar to the original model, implying that the paths cover the entire model.

6.2 Future Work

The algorithm currently works for environments which are well defined and does not work for unknown or partially known environments. Further research on extending this work to work for unknown environments based on proposed heuristics can be carried on.

Though the algorithm is scalable and robust, getting the ideal paths depend on various parameters like the viewpoint search radius and so on. The parameters are influenced by the geometry of the model and the sampling of the input model. As of now, the algorithm needs iterations of refinement from the user by tuning the parameters in order to get the desired path for the UAV. A relationship between the sampling of the model and the parameters can be deduced which can make the algorithm independent of user intervention.

REFERENCES

[1] S. Jung, S. Song, P. Youn, and H. Myung, "Multi-layer coverage path planner for autonomous structural inspection of high-rise structures," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Oct 2018, pp. 1–9.

[2] Wikipedia, "Aviation safety — Wikipedia, the free encyclopedia," http://en.wikipedia.org/w/index.php?title=Aviation%20safetyoldid=871605134, 2018, [Online; accessed 04-December-2018].

[3] "Intel airbus demo drone visual inspection of passenger airliner," July 2016. [Online]. Available: http://www.asctec.de/en/intel-airbus-demo-drone-visual-inspection-of-passenger-airliners/

[4] F. Morbidi, R. Cano, and D. Lara, "Minimum-energy path generation for a quadrotor uav," in *2016 IEEE International Conference on Robotics and Automation (ICRA)*, May 2016, pp. 1492–1498.

[5] Z. L. Cao, Y. Huang, and E. L. Hall, "Region filling operations with random obstacle avoidance for mobile robots," *Journal of Robotic Systems*, vol. 5, no. 2, pp. 87–102. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/rob.4620050202

[6] A. R. Prasad Atkar, Howie Choset and E. Acar, "Exact cellular decomposition of closed orientable surfaces embedded in r3," in *Proceedings of the 2001 IEEE International Conference on Robotics and Automation (ICRA '01)*, May 2001.

[7] P. Atkar, H. Choset, and A. Rizzi, "Towards optimal coverage of 2-dimensional surfaces embedded in 3: Choice of start curve," vol. 4, 11 2003, pp. 3581 – 3587 vol.3.

[8] P. N. Atkar, A. Greenfield, D. Conner, H. Choset, and A. Rizzi, "Hierarchical segmentation of surfaces embedded in r3 for auto-body painting." vol. 2005, 01 2005, pp. 572–577.

[9] P. Atkar, A. L. Greenfield, D. C. Conner, H. Choset, and A. Rizzi, "Uniform coverage of automotive surface patches," *The International Journal of Robotics Research*, vol. 24, no. 11, pp. 883 – 898, November 2005.

[10] P. Cheng, J. Keller, and V. Kumar, "Time-optimal uav trajectory planning for 3d urban structure coverage," in *2008 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Sept 2008, pp. 2750–2757.

[11] R. Almadhoun, T. Taha, D. Gan, J. Dias, Y. Zweiri, and L. Seneviratne, "Coverage Path Planning with Adaptive Viewpoint Sampling to Construct 3D Models of Complex Structures for the Purpose of Inspection."

[12] K. Helsgaun, "An effective implementation of the linkernighan traveling salesman heuristic," *European Journal of Operational Research*, vol. 126, no. 1, pp. 106 – 130, 2000. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0377221799002842

[13] A. Bircher, K. Alexis, M. Burri, P. Oettershagen, S. Omari, T. Mantel, and R. Siegwart, "Structural inspection path planning via iterative viewpoint resampling with application to aerial robotics," in *2015 IEEE International Conference on Robotics and Automation (ICRA)*, May 2015, pp. 6423–6430.

[14] T. Danner and L. E. Kavraki, "Randomized planning for short inspection paths," in *Proceedings 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings (Cat. No.00CH37065)*, vol. 2, April 2000, pp. 971–976 vol.2.

[15] B. Englot and F. S. Hover, “Sampling-based coverage path planning for inspection of complex structures,” in *ICAPS*, 2012.

[16] G. Papadopoulos, H. Kurniawati, and N. M. Patrikalakis, “Asymptotically optimal inspection planning using systems with differential constraints,” in *2013 IEEE International Conference on Robotics and Automation*, May 2013, pp. 4126–4133.

[17] E. Galceran and M. Carreras, “Planning coverage paths on bathymetric maps for in-detail inspection of the ocean floor,” in *2013 IEEE International Conference on Robotics and Automation*, May 2013, pp. 4159–4164.

[18] J. Jin and L. Tang, “Coverage path planning on three-dimensional terrain for arable farming,” *Journal of Field Robotics*, vol. 28, no. 3, pp. 424–440. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/rob.20388

[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*, R. Alami, R. Chatila, and H. Asama, Eds. Tokyo: Springer Japan, 2007, pp. 221–230.

[20] 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. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/rob.20403

[21] M. Faessler, F. Fontana, C. Forster, E. Mueggler, M. Pizzoli, and D. Scaramuzza, “Autonomous, vision-based flight and live dense 3d mapping with a quadrotor micro aerial vehicle,” *Journal of Field Robotics*, vol. 33, no. 4, pp. 431–450, 2016.

[22] S. Weiss, M. Achtelik, L. Kneip, D. Scaramuzza, and R. Siegwart, "Intuitive 3d maps for mav terrain exploration and obstacle avoidance," *Journal of Intelligent & Robotic Systems*, vol. 61, no. 1-4, pp. 473–493, 2011.

[23] A. Hornung, K. M. Wurm, M. Bennewitz, C. Stachniss, and W. Burgard, "Octomap: An efficient probabilistic 3d mapping framework based on octrees," *Autonomous Robots*, vol. 34, no. 3, pp. 189–206, 2013.

[24] S. Khan, A. Dometios, C. Verginis, C. Tzafestas, D. Wollherr, and M. Buss, "Rmap: a rectagular cuboid approximatio framework for 3d eviromet mappig," *Autonomous Robots*, vol. 37, no. 3, pp. 261–277, 2014.

[25] S. Omari, P. Gohl, M. Burri, M. Achtelik, and R. Siegwart, "Visual industrial inspection using aerial robots," in *Applied Robotics for the Power Industry (CARPI), 2014 3rd International Conference on*. IEEE, 2014, pp. 1–5.

[26] M. Schadler, J. Stückler, and S. Behnke, "Rough terrain 3d mapping and navigation using a continuously rotating 2d laser scanner," *KI-Künstliche Intelligenz*, vol. 28, no. 2, pp. 93–99, 2014.

[27] J. Fossel, D. Hennes, D. Claes, S. Alers, and K. Tuyls, "Octoslam: A 3d mapping approach to situational awareness of unmanned aerial vehicles," in *Unmanned Aircraft Systems (ICUAS), 2013 International Conference on*. IEEE, 2013, pp. 179–188.

[28] "Hector open source modules for autonomous mapping and navigation with rescue robots," in *RoboCup*, 2013.

[29] J. P. Saarinen, H. Andreasson, T. Stoyanov, and A. J. Lilienthal, "3d normal distributions transform occupancy maps: An efficient representation for mapping in dynamic environments," *The International Journal of Robotics Research*, vol. 32, no. 14, pp. 1627–1644, 2013.

[30] F. S. Hover, R. M. Eustice, A. Kim, B. Englot, H. Johannsson, M. Kaess, and J. J. Leonard, “Advanced perception, navigation and planning for autonomous in-water ship hull inspection,” *The International Journal of Robotics Research*, vol. 31, no. 12, pp. 1445–1464, 2012.

[31] M. Maurer, M. Rumpler, A. Wendel, C. Hoppe, A. Irschara, and H. Bischof, “Geo-referenced 3d reconstruction: Fusing public geographic data and aerial imagery,” in *Robotics and Automation (ICRA), 2012 IEEE International Conference on*. IEEE, 2012, pp. 3557–3558.

[32] N. Haala and M. Kada, “An update on automatic 3d building reconstruction,” *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 65, no. 6, pp. 570–580, 2010.

[33] H. M. Nguyen, B. Wunsche, P. Delmas, C. Lutteroth, and W. van der Mark, “High resolution 3d content creation using unconstrained and uncalibrated cameras,” in *Human System Interaction (HSI), 2013 The 6th International Conference on*. Citeseer, 2013, pp. 637–644.

[34] M. Kazhdan and H. Hoppe, “Screened poisson surface reconstruction,” *ACM Trans. Graph.*, vol. 32, no. 3, pp. 29:1–29:13, Jul. 2013. [Online]. Available: http://doi.acm.org/10.1145/2487228.2487237

[35] F. Remondino, L. Barazzetti, F. Nex, M. Scaioni, and D. Sarazzi, “Uav photogrammetry for mapping and 3d modeling–current status and future perspectives,” *International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, vol. 38, no. 1, p. C22, 2011.

[36] M. Quigley, K. Conley, B. P. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, and A. Y. Ng, “Ros: an open-source robot operating system,” in *ICRA Workshop on Open Source Software*, 2009.

[37] L. Meier, "Micro aerial vehicle link protocol (mavlink)," *mavlink. org*, 2015.

[38] "Mavros." [Online]. Available: http://wiki.ros.org/mavros

[39] K. M. Wurm and A. Hornung, "Octomap." [Online]. Available: https://octomap.github.io/

[40] "Open source for drones." [Online]. Available: https://px4.io/

[41] N. Koenig and A. Howard, "Design and use paradigms for gazebo, an open-source multi-robot simulator," in *2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (IEEE Cat. No.04CH37566)*, vol. 3, Sept 2004, pp. 2149–2154 vol.3.

[42] "Gpu gems." [Online]. Available: https://developer.nvidia.com/gpugems/GPUGems2/gpugems2_chapter37.html

[43] M. Attene, B. Falcidieno, and M. Spagnuolo, "Hierarchical mesh segmentation based on fitting primitives," *The Visual Computer*, vol. 22, no. 3, pp. 181–193, 2006.

[44] X. Qian, C. Peng, C. Nong, and Z. Xiang, "Dynamic obstacle avoidance path planning of uavs," in *2015 34th Chinese Control Conference (CCC)*, July 2015, pp. 8860–8865.

[45] Y. Wang, T. Kirubarajan, R. Tharmarasa, R. Jassemi-Zargani, and N. Kashyap, "Multiperiod coverage path planning and scheduling for airborne surveillance," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 54, no. 5, pp. 2257–2273, Oct 2018.