COOPERATIVE SEARCH WITH MULTIPLE QUADCOPTERS USING DOWNWARD FACING CAMERAS

Thesis

Submitted in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

by

Jeane Marina D'Souza

(ME12F04)

MECHANICAL ENGINEERING DEPARTMENT

NATIONAL INSTITUTE OF TECHNOLOGY KARNATAKA

SURATHKAL, MANGALURU -575025
December, 2020

DECLARATION

I hereby declare that the Research Thesis entitled **'Cooperative search with multiple quadcopters using downward facing cameras'** which is being submitted to the **National Institute of Technology Karnataka Surathkal**, in partial fulfillment of the requirements for the award of the degree of **Doctor of Philosophy** in the Department of **Mechanical Engineering**, is a bonafide report of work carried out by me. The material contained in this Research Thesis has not been submitted to any University or Institution for the award of any degree.

ME12F04, Jeane Marina D'Souza
Department of Mechanical Engineering

Place: NITK, Surathkal
Date:

CERTIFICATE

This is to certify that the Research Thesis entitled **'Cooperative search with multiple quadcopters using downward facing cameras'** submitted by **Jeane Marina D'Souza (Register number:ME12F04)** as the record of the research work carried out by him, is *accepted as the Research Thesis submission* in partial fulfilment of the requirements for the award of degree of **Doctor of Philosophy**.

Reasearch Guide

Chairman - DRPC
(Signature with Date and Seal)

Place: NITK, Surathkal
Date:

ACKNOWLEDGEMENT

I thank God for the showers of blessings in the form of good health and prosperity bestowed upon me every day of my life.

I would like to express my sincere gratitude to my research supervisor, **Dr. K.R Guruprasad**, for his outstanding guidance, invaluable encouragement, and consistent patience through the course of this research work. Thank you Sir for leading me into this interesting field of Robotics and control and also providing moral support and guidance, which will benefit my career development.

I want to thank **Dr. Shrikantha S. Rao** (Professor and Head), Dept. of Mechanical engineering, and **Dr. Laxminidhi S.** (Professor and Head) Department of Electronics and Communication Engineering for their helpful feedback all through my research presentations.

I am immensely grateful to **Dr. Shrikantha S. Rao**, Professor and Head, Department of Mechanical Engineering for extending the Departmental facilities, which ensured the satisfactory progress of my research work.

I thank the Director and the administration of NITK Surathkal for permitting me to pursue my research work at the Institute.

Many thanks to my colleagues Sutej P. Kulgod, Varun Venkat, Arjun Sadananda Shetty, Siddhartha Rao, Akash Prakash, Srinidhi, Vishnu Nair, Akash Mani, Niharica Bharadwaj, Akshar, Deepak Mohan, Minhaz Falaki, Akshay, Ranjitha TD, Roopa Palguna, Charitha Rao, Vasista Ayyagiri, Sanjay Wepin, Abin Ephram, Abin Pothen, Sandeep, Aamodh Suresh, Sofia Sahib, Pramod, Tanmay Gadgil, Prajwal, Shashank Rao, Laksh, and Ayush in Robotics lab for their assistance and friendship.

My heartfelt gratitude to **Claire and Ovin, our parents and family members** for their understanding, love, encouragement, and support over the years which helped me through my research days.

Jeane Marina D'Souza

ABSTRACT

Multiple robots have been extensively used in performing several tasks cooperatively, in a distributed manner. These distributed multi-robotic systems (MRS) or multi-agent systems (MAS) find application in many fields such as search and rescue, environment monitoring, surveillance, landmine detection and clearing, etc. Apart from reduced mission time owing to several agents performing the task simultaneously, distributed MRS are also robust to failure of some of the individual robots. Owing to these advantages, simpler design, and lower cost of individual robots, they are increasingly finding applications in adverse conditions such as in military applications and disasters such as natural calamities.

Searching for survivors in regions affected by a natural calamity in a civilian context, searching for mines or enemy targets in a military context, using sophisticated sensors carried on unmanned vehicles, such as UAVs or UGVs, are some of the very useful problems that need attention of the researchers from the field of multi-robotic systems. Compared to fixed-wing UAVs, owing to their maneuverability, ease of takeoff and landing, compactness, lower cost, hovering ability, etc., quadcopters are more suitable for such operations.

In this thesis, we formulate a multi-agent search strategy using quadcopter UAVs as search agents/vehicles and downward facing cameras mounted on the quadcopters as search agents. Based on practical considerations, we assumed that the search effectiveness of the camera is maximum at the center and degrades away from it, unlike in most work in the literature where it is assumed to be constant over the entire image frame. The lack of information about presence or absence or the targets of interest in the search space is modeled as an uncertainty density distribution. Here, the uncertainty is 1 when no information on the existence (or absence) of the target at a point of interest is available and 0 when it is established that the target is either present or absent at that point. Based on uncertainty density distribution and the monotonically

decreasing search effectiveness model, we address and formulate the problem of optimally deploying the quadcopters so as to maximize the uncertainty reduction (and hence information gain). Based on the observation we make on similar problem setting used in the literature, we formulate a 'deploy' and 'search' strategy using the concepts of centroidal Voronoi configuration, where the quadcopters get deployed to a centroidal Voronoi configuration, shown to be an optimal configuration maximizing the reduction in uncertainty, and then perform search resulting in a reduction in the uncertainty. The process of optimal 'deployment' and 'search' continue until the average uncertainty over the entire search space is reduced below an arbitrary but fixed value, indicating the targets, if presented, are detected with an acceptable confidence (probability).

One of the very important components in multi-agent search is the search sensor itself and the spatial variation of its effectiveness in performing the search, that is target detection. As we mentioned earlier, we assume the non-uniform effectiveness of the camera within its image frame. We first provide a detailed discussion on the spatial variation of the image quality both in terms of optical resolution and digital quality. We observe that the image quality is higher at the central pixel and degrades away from it. Such a scenario leads us to a non-uniform search effectiveness of the camera. We present an experimental setup to obtain a sensor effectiveness model for a downward-facing camera using the target detection probability. Through a set of target detection experiments carried out using AuRuco markers and triangular-shaped objects as targets, we obtain a sensor effectiveness model for a downward-facing camera in different scenarios. We also establish that an exponential function with two parameters can be used to model the spatial variation of the camera's search effectiveness (that is, the search effectiveness model).

We develop a platform using ROS/Gazebo and Matlab environment for simulation of the proposed multi-quadcopter search strategy in a hybrid centralized-decentralized architecture. The platform developed can be a very useful tool for conducting realistic simulation experiments to validate the proposed search strategy and to make a comparative study of its performance in terms of time required for the search process, with different parameters such as camera search effectiveness functions, sensor range, number of robots, and decide on the right parameters for any given mission.

We provide detailed results of experiments and simulation carried out along with a detailed discussion on the same. First, we present the results of the experiments carried out to obtain the search effectiveness model of the downward-facing camera, which we use in the proposed multi-quadcopter search strategy. Though we used an experimental setup to establish, that in general, an exponential function with two parameters can be as the search effectiveness model of a camera, it can be used to carry out experiments with a specific type of imaging sensor, the type of image processing tools, the kind of environment in which it has to detect the targets, the type of targets that need to be detected, and hence obtain a suitable search effectiveness model.

We present representative results of the simulation experiment carried out using the realistic ROS/Matlab simulation platform, both to demonstrate the simulation platform itself and the proposed search strategy. Finally, we provide a detailed account of simulation experiments carried out to evaluate the effect of the number of search quadcopters and the camera effectiveness parameters on the performance of the proposed multi-quadcopter search strategy, using the simulation platform developed in this work.

The simulation platform developed can be used to carry out experiments using physical AR Drones. The controller used within the simulation environment may be used to control the physical AR Drones. Also, the simulation environment can be used to conduct a large number of simulation and physical experiments to decide on parameters such as the optimal number of quadcopters, type of cameras used (in terms of their search effectiveness, which may be obtained by using the experimental setup based on that used in this work), for a given search scenario. In this sense, the experimental setup and simulation platform developed are useful beyond the sample results provided in this thesis and will surely help the proposed

multi-agent search strategy takes a step forward from theory to experiment and then finally into reality.

Keywords: Multi-robot search, Cooperative search, Quadcopter, Uninhabitated Aerial Vehicles, Voronoi Partitioning

CONTENTS

Title Page . . . .

Declaration . . . .

Certificate . . . .

Abstract . . . .

Contents i

List of Figures v

List of Tables x

1 INTRODUCTION 1

1.1 CENTRALIZED, DECENTRALIZED AND DISTRIBUTED ARCHITECTURES 2

1.2 MULTI-AGENT SEARCH PROBLEM 4

1.3 PREVIEW OF RELATED CONCEPTS 5

1.3.1 Voronoi Partition 5

1.3.2 Centroidal Voronoi Configuration 6

1.3.3 Quadcopter dynamics 7

1.3.4 Robot Operating System 12

1.3.5 Transform(tf) 14

1.3.6 ROS launch files 16

1.3.7 Gazebo simulator 16

1.3.8 Unified Robot Description Format (URDF) 17

1.4 SUMMARY OF THE CONTRIBUTIONS 20

1.5 Organization of the thesis 23

2 LITERATURE REVIEW 25

2.1 TECHNIQUES USED FOR MAS 25

2.2 TYPES OF AGENTS USED IN MAS 26

2.3 TYPE OF SEARCH SENSOR USED IN MAS 27

2.4 TARGETS BEING SEARCHED 28

2.5 APPLICATION DOMAIN FOR MAS 29

2.6 CONTINUOUS VS GRIDDED ENVIRONMENT 29
2.7 SENSOR EFFECTIVENESS/FOOTPRINT USED IN MAS 30
2.8 RESEARCH GAP AND MOTIVATION 40

3 MULTI-QUADCOPTER SEARCH STRATEGY 43
3.1 PROBLEM SETTING 43
3.2 PROPOSED SEARCH STRATEGY 44
3.3 SUMMARY 47

4 SEARCH EFFECTIVENESS OF A CAMERA 49
4.1 CAMERA AS A SEARCH SENSOR 49
4.1.1 Search effectiveness models 50
4.1.2 Image resolution 52
4.1.3 Quantum efficiency 54
4.2 EXPERIMENTAL SETUP 55
4.2.1 Imaging sensor 56
4.2.2 Targets 57
4.3 SUMMARY 58

5 A REALISTIC SIMULATION PLATFORM FOR THE PROPOSED MULTI-QUADCOPTER SEARCH 59
5.1 HYBRID ARCHITECTURE FOR THE PROPOSED STRATEGY 59
5.2 SIMULATION PLATFORM FOR THE PROPOSED SEARCH STRATEGY 63
5.2.1 ROS/Gazebo Simulator 64
5.2.2 Central controller using Matlab 65
5.2.3 Multi-quadcopter system in ROS 65
5.2.4 ARDrone quadcoptor control 65
5.2.5 Multi-ARDrone system control: 67
5.2.6 Control of a quadcopter in Gazebo 68
5.3 SUMMARY 68

6 RESULTS AND DISCUSSIONS 71
6.1 CAMERA EFFECTIVENESS 71
6.1.1 Experiments with markers 71
6.1.2 Experiments using triangle shaped targets 77
6.2 EXPERIMENTS USING THE HYBRID SIMULATION PLATFORM 81
6.2.1 Experiments with five AR.Drones 83
6.2.2 Experiments with three AR.Drones 86
6.2.3 Experiments with different number of AR.Drones 86
6.3 EXPERIMENTS WITH DIFFERENT NUMBER OF QUADCOPTERS AND SENSOR RANGE 88
6.4 SUMMARY 110
7 CONCLUSIONS AND SCOPE FOR THE FUTURE WORK 113
7.1 CONCLUSIONS 113
7.2 Scope for Future work 115
BIBLIOGRAPHY 117
LIST OF PUBLICATION 128

List of Figures

1.1 A distributed multi-agent system. Circles indicate agents that form the nodes and double arrowed lines indicate the edges connecting the agents/nodes forming an adjacency graph. 2
1.2 A multi-robotic system may have (a) a centralized, (b) a decentralized, or (c) a distributed control architecture. 3
1.3 A Voronoi partition generated by nodes $p_1 \ldots p_8$. 6
1.4 A centroidal Voronoi configuration with the corresponding Voronoi partition. 7
1.5 Quadcopter body frame and inertial frame 8
1.6 Example of a ROS network 13
1.7 ROS-Computation graph Level 14
1.8 Structure of tf 15
1.9 Transform tree of two ‘Turtlesim’ robots 16
1.10 Gazebo simulation environment 17
1.11 Representation of a link. 18
1.12 Representation of the joint 19
1.13 Representation of the robot 20

2.1 A (a) flat and (b) non-flat sensor footprint. 31

3.1 Illustration of multiple autonomous quadcopters searching for targets in a search area using downward facing cameras. 43
3.2 Flow chart illustrating typical autonomous multi-UAV search for target detection. 44

4.1 Illustration of a (a) single or (b) multiple autonomous quadcopters searching for targets in a search area using downward-facing cameras. 50
4.2 Different sensor effectiveness models assumed in the literature. . . . 51

4.3 Plot of the MTF values with the distance from the central pixel for a Nikon Nikkor camera. Source:http://pixelsandpaintstrokes.com/tools-technique/mtf-charts/ 52
4.4 Sagittal lines and Meridonial lines. 53
4.5 QE v/s pixel position for 6m pixel size CMOS sensor. 55
4.6 Schematic representation of the experimental setup. 57
4.7 An ArUco marker. 57

5.1 A hybrid centralized-distributed multi-quadcopter search architecture. 61
5.2 A hybrid centralized-decentralized multi-quadcopter search architecture. 63
5.3 A block diagram illustrating the architecture of the simulation platform for multi-quadcopter search strategy using centroidal Voronoi configuration, in two machines. 64

6.1 4x4cm ArUco markers target detection experiment setup 72
6.2 Exponential curve fit over entire data corresponding to Table 6.1. . 73
6.3 4x4cm ArUco markers target detection experiment with salt and pepper noise 74
6.4 Exponential curve fit over the data given in Table 6.2. 75
6.5 5x5cm ArUco markers target detection experiment setup. 75
6.6 Exponential curve fit over the data shown in Table 6.3. 76
6.7 5x5 ArUco marker detection with salt and pepper noise 77
6.8 Exponential curve fit over over the data shown in Table 6.4. 78
6.9 Triangle shapes for target detection experiment setup Table 6.5. . . 78
6.10 Exponential curve fit over the data shown in Table 6.5. 79
6.11 AR Drone (a) outdoor configuration and (b) indoor configuration. 81
6.12 AR Drone model within the Gazebo environment. 82

6.13 Snapshots of the deployment of quadcopters from an initial configuration to a centroidal Voronoi configuration in Gazebo environment. 84

6.14 Snapshots from Matlab showing the process of optimal deployment of quadcopters. The positions of the quadcopters (based on which the Voronoi partition is created), the Voronoi cells, and the corresponding centroids are also shown in each step. . 85

6.15 Reduction in uncertainty density after each "search" performed at the end of optimal "deployment". 87

6.16 Reduction in (a) maximum uncertainty density and ((b) average uncertainty density over Q as the "deployment" and "search" progress. 88

6.17 First four 'deploy and search' steps with three AR.Drones. Figures in the left side show the configuration of the quadcopters at the end of deployment and those on the right side indicate uncertainty density distribution after the search performed following the deployment. 89

6.18 'Deploy and search' steps $5-8$ with three AR.Drones. Figures in the left side show the configuration of the quadcopters at the end of deployment and those on the right side indicate uncertainty density distribution after the search performed following the deployment. . 90

6.19 Exponential function $ke^{-\alpha r^2}$ with $k = 0.6$ and different vakues of α $(0.07, 0.1, 0.3, 0.5, 1)$. 91

6.20 Number of search steps required with $\alpha = 0.07$, for different number of searchers, as shown in Table 6.7. 93

6.21 Reduction in average uncertainty density for different number of quadcopters with $\alpha = 0.07$. 93

6.22 Number of search steps required with $\alpha = 0.1$, for different number of searchers, as shown in Table 6.8. 94

6.23 Reduction in average uncertainty density for different number of quadcopters with $\alpha = 0.1$. 95
6.24 Number of search steps required with $\alpha = 0.3$, for different number of searchers, as shown in Table 6.9. 96
6.25 Reduction in average uncertainty density for different number of quadcopters with $\alpha = 0.3$. 96
6.26 Number of search steps required with $\alpha = 0.5$, for different number searchers, as shown in Table 6.10. 97
6.27 Reduction in average uncertainty density for different number of quadcopters with $\alpha = 0.5$. 98
6.28 Number of search steps required with $\alpha = 1$, for different number searchers, as shown in Table 6.11. 99
6.29 Reduction in average uncertainty density for different number of quadcopters with $\alpha = 1$. 99
6.30 Number of search steps required with $\alpha = 0.07, 0.1, 0.3, 0.5, 1$ and $N = 3, 5, 10, 15, 20, 30, 40, 50$ 100
6.31 Number of searches required with varying α with $N = 3$. 101
6.32 Number of searches required with varying α with $N = 5$. 102
6.33 Number of searches required with varying α with $N = 10$. 102
6.34 Number of searches required with varying α with $N = 15$. 103
6.35 Number of searches required with varying α with $N = 20$. 104
6.36 Number of searches required with varying α with $N = 30$. 104
6.37 Number of searches required with varying α with $N = 40$. 105
6.38 Number of searches required with varying α with $N = 50$. 106
6.39 First search instance with $\alpha = 0.07$ and different N. 107
6.40 First search instance with $\alpha = 0.3$ and different N. 108
6.41 First search instance with $\alpha = 1$ and different N. 109
6.42 First search instance with $\alpha = 0.07$ and $N = 3, 5, 10, 15, 20, 30, 40,$ and 50. 111

+

List of Tables

4.1 The Quantum Efficiency at each point is considered to be the certainty of finding an object at that point in the image. 56

6.1 Probability of detection with location of target using 4×4 ARuco markers. 73
6.2 Probability of detection with location of target using 4×4 AuRuco markers with added salt and pepper noise. 74
6.3 Probability of detection with location of target using 5×5 AuRuco markers. 76
6.4 Probability of detection with location of target using 5×5 AuRuco markers with salt and pepper noise. 77
6.5 Probability of detection with location of target using triangular targets. 78
6.6 Simulation time for a single iteration in the optimal deployment process (t_{it}), total simulation time t_{tot}, and number of search instances ('deploy' and 'search' steps) N_s with different number (N) of quadcopters. 91
6.7 Search with multiple quadcopters with $\alpha = 0.07$ 92
6.8 Search with multiple quadcopters with $\alpha = 0.1$ 94
6.9 Search with multiple quadcopters with $\alpha = 0.3$ 95
6.10 Search with multiple quadcopters with $\alpha = 0.5$ 97
6.11 Search with multiple quadcopters with $\alpha = 1$. 98
6.12 Search using $\alpha = 0.07, 0.1, 0.3, 0.5, 1$ and $N = 3$. 101
6.13 Search using $\alpha = 0.07, 0.1, 0.3, 0.5, 1$ and $N = 5$. 101
6.14 Search using $\alpha = 0.07, 0.1, 0.3, 0.5, 1$ and $N = 10$. 102
6.15 Search using $\alpha = 0.07, 0.1, 0.3, 0.5, 1$ and $N = 15$. 103
6.16 Search using $\alpha = 0.07, 0.1, 0.3, 0.5, 1$ and $N = 20$. 103
6.17 Search using $\alpha = 0.07, 0.1, 0.3, 0.5, 1$ and $N = 30$. 104
6.18 Search using $\alpha = 0.07, 0.1, 0.3, 0.5, 1$ and $N = 40$. 105

6.19 Search using $\alpha = 0.07, 0.1, 0.3, 0.5, 1$ and $N = 50$. 105

7.1 List of Publications based on PhD Research Work 129

CHAPTER 1

INTRODUCTION

Searching for the presence of targets of interest such as survivors in a disaster is an interesting and practically useful and challenging problem. Some of the earliest literature (Stone 1975, Koopman 1980, Lida 1992) on this problem of searching for targets in an unknown environment assume certain restrictive conditions. These seminal contributions were mostly theoretical in nature and applied to a single agent searching for single or multiple, and static or moving, targets. The same task can likely be accomplished more effectively by multiple searchers. But, when multiple agents are involved, coordination between them becomes an important issue.

Multiple robots have been extensively used in performing several tasks cooperatively, in a distributed manner. These distributed multi-robotic systems (MRS) or multi-agent systems (MAS) find application in many fields such as search and rescue, environment monitoring, surveillance, landmine detection, and clearing, etc. Apart from reduced mission time owing to several agents performing the task simultaneously, distributed MRS are also robust to the failure of some of the individual robots. Owing to these advantages, simpler design, and lower cost of individual robots, they are increasingly finding applications in adverse conditions such as in military applications and disasters such as natural calamities.

Searching for survivors in regions affected by a natural calamity in a civilian context, searching for mines or enemy targets in a military context, using sophisticated sensors carried on unmanned vehicles, such as UAVs or UGVs, are some of the very useful problems that need attention of the researchers from the field of multi-robotic systems. Compared to fixed wing UAVs, owing to their maneuverability, ease of takeoff and landing, compactness, lower cost, hovering ability, etc., quadcopters are more suitable for such operations.

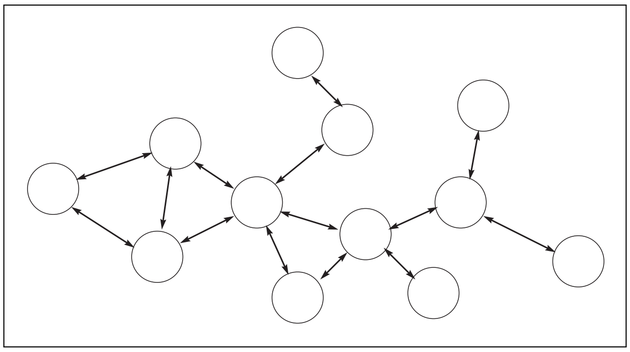


Figure 1.1: A distributed multi-agent system. Circles indicate agents that form the nodes and double arrowed lines indicate the edges connecting the agents/nodes forming an adjacency graph.

1.1 CENTRALIZED, DECENTRALIZED AND DISTRIBUTED ARCHITECTURES

In this section, we preview concepts from centralized, decentralized, and distributed architectures used in networked systems such as a MAS/MRS. In a MAS/MRS, the individual agents influence each other to accomplish the assigned task. They may not (or may) explicitly interact or communicate with the neighboring agents. The action of one agent influences other agents in some form. These influences/interactions happen typically in a spatially distributed form over the adjacency graph formed by the agents, as illustrated in Figure 1.1. Neighboring agents directly influence each other's actions and every agent's action is influenced by every other agent's action if the adjacency graph is connected. Such a distributed influence/interaction leads to a collective behavior.

While the agents interact/influence in a distributed manner, the controller may have centralized, decentralized or distributed architectures, as illustrated in Figure 1.2. In the case of a centralized architecture, a central controller (or a server) controls all the individual agents. It receives necessary information from all the agents and sends control signals to all of them. Failure of the central controller leads to the failure of the entire system. Also, communication is very crucial in a centralized controller. In a distributed controller each agent

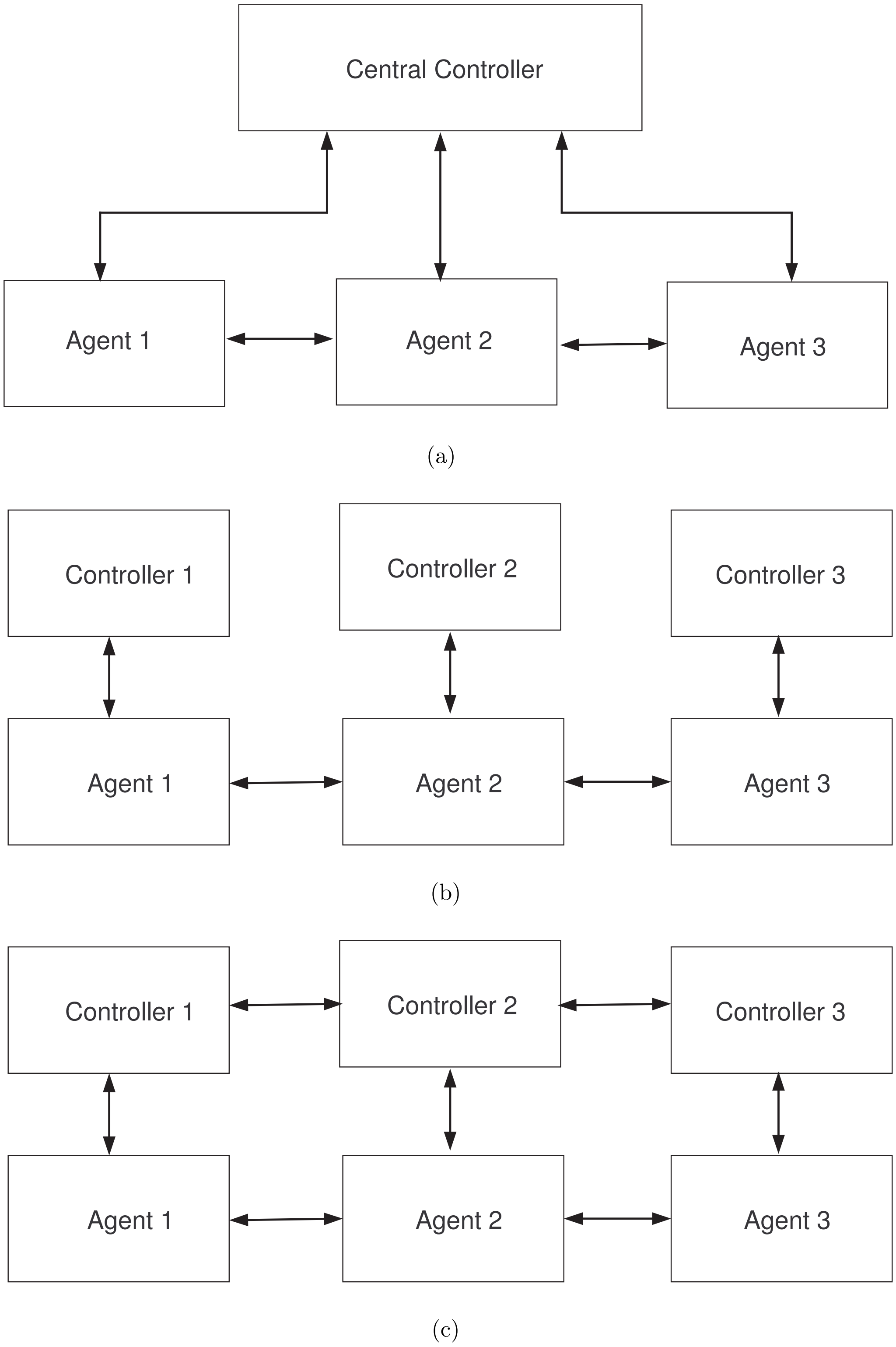


Figure 1.2: A multi-robotic system may have (a) a centralized, (b) a decentralized, or (c) a distributed control architecture.

is controlled by a dedicated controller. Here, controllers communicate amongst themselves to account for the interaction/coupling between the agents. In a typical distributed multi-agent system, a distributed control architecture may not suffer from a single point failure. Thus, this architecture can provide robustness to failure of a few agents/controllers. Further, the computational load is distributed among the agent-level controllers. However, communication overhead is still an issue, though to a lesser extent as compared to the centralized control architecture. In a decentralized control architecture, each agent is controlled by a dedicated controller in a similar way as in the case of the distributed control architecture. However, the controllers do not interact with each other. Thus dependency on communication quality or the communication overhead is not an issue in a decentralized control scheme. However, this architecture cannot account for the interaction between the agents which is typical of any MAS/MRS.

Remark 1 *While a centralized control architecture requires complete communication between the central server with the individual agents, a distributed control architecture requires communication between the agent-level controllers. Both are sensitive to communication quality and delays at different levels. A decentralized control architecture does not require communication between the agent-level controllers.*

1.2 MULTI-AGENT SEARCH PROBLEM

In this thesis, we address the problem of searching an unknown region for the presence of targets of interest using multiple cooperating quadcopters carrying downward-facing cameras. We use a realistic model for search effectiveness of a downward-facing camera, where a suitably defined search effectiveness is maximum just below the center of the camera and degrades monotonically away from the center. We experimentally validate such a search effectiveness model-based target detection probability. The search task is collecting information of interest such as the presence of targets. The lack of information is modeled as an uncertainty density distribution over the search space. The uncertainty is reduced on performing search. Consider for example a problem of searching for survivors in a natural calamity such as an earthquake. When no information about presence or absence of a survivor (target) is available then we assign uncertainty as 1 and if such information is available (or gathered by search), then the uncertainty is 0. If this information is available only partially, then the uncertainty is a value between 0 and 1. The objective of this work is to devise and validate a multi-quadcopter search strategy so as to reduce the uncertainty density. We have developed a platform using MATLAB and Gazebo simulators in ROS environment for realistic simulation of such a system. Such a simulation platform is a definitive step forward toward translating solutions to multi-UAV search problem from theory to laboratory environment, and finally into reality.

1.3 PREVIEW OF RELATED CONCEPTS

Now, we preview a few fundamental concepts that are used in this work. We first preview the concepts of Voronoi partition and Centroidal Voronoi configuration. we then provide an introduction to ROS and its components.

1.3.1 Voronoi Partition

Named after Georgy Voronoi (Voronoi (1907)) and is also called *Dirichlet tessellation* (named after Gustav Lejeune Dirichlet Dirichlet (1850)), the Voronoi partition is a widely used scheme of partitioning a given space based on the concept of "nearness" of points in a set to some finite number of pre-defined locations in the set called 'nodes' or 'generators'. The concept has been used in several fields such as, image processing, CAD, sensor coverage, GIS, multi-robotic coverage, robot path planning, etc.

A *partition* of a set X is a collection of subsets W_i of X with disjoint interiors such that their union is X itself. Let $Q \subset \mathbb{R}^2$, be a convex polygon. Let $\mathcal{P} = \{p_1, p_2, \ldots, p_N\}$, $p_i \in Q$, be a finite set of nodes. The *Voronoi partition* generated by $\mathcal{P}$ is the collection $\{V_i(\mathcal{P})\}_{i \in \{1,2,\ldots,N\}}$ and is defined as,

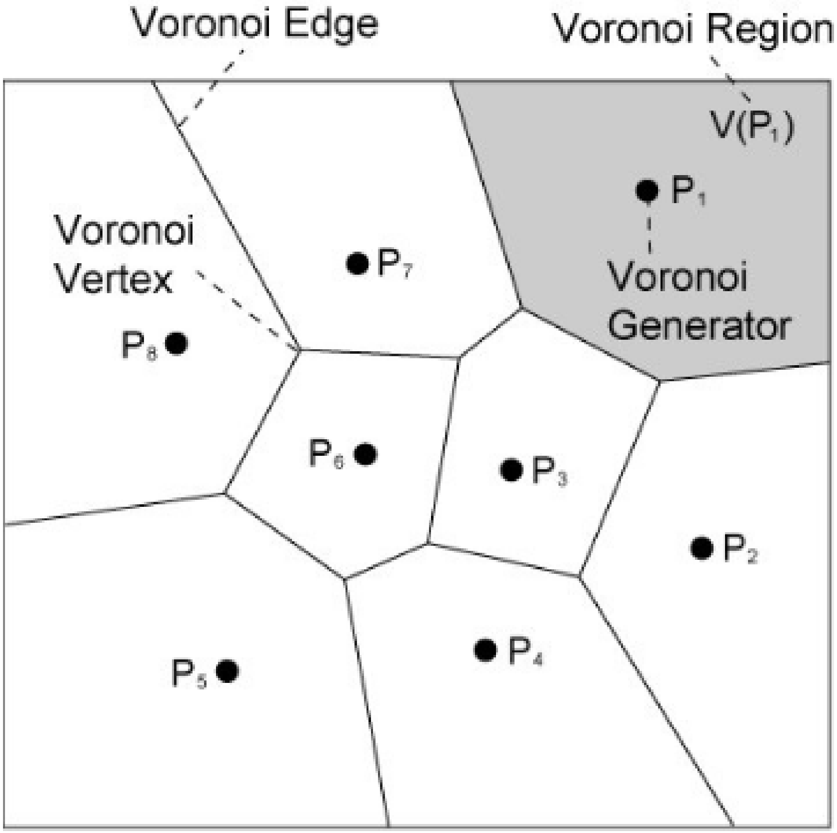


Figure 1.3: A Voronoi partition generated by nodes $p_1 \dots p_8$.

$$V_i(\mathcal{P}) = \{q \in Q | \parallel q - p_i \parallel \leq \parallel q - p_j \parallel, \forall p_j \in \mathcal{P}\}$$

A Voronoi cell V_i is the collection of those points which are closest to the node p_i compared to any other node in $\mathcal{P}$. The intersection of any two Voronoi cells is either null, a line segment, or a point. Also, a Voronoi cell is a topologically connected non-null set. Figure 1.3 illustrates a Voronoi partition.

1.3.2 Centroidal Voronoi Configuration

In centroidal Voronoi configuration, the nodes which generate the Voronoi cells are located at the centroid of the respective Voronoi cells. This concept was proposed in Du et al. (1999). Figure 1.4 illustrates a centroidal Voronoi configuration. It may be noted that a centroidal Voronoi configuration results in more uniform partitioning.

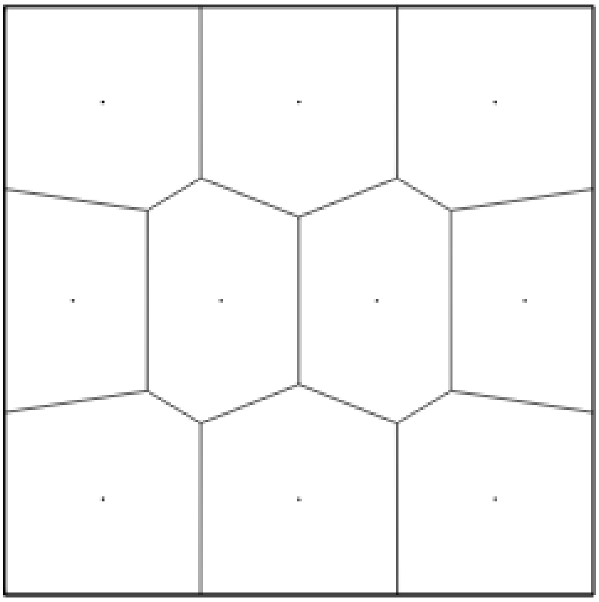


Figure 1.4: A centroidal Voronoi configuration with the corresponding Voronoi partition.

This concept has been used in several locational optimization, facility location (Okabe & Suzuki 1997), optimal sensor deployment Cortes et al. (2004) and multi-robot deployment (Guruprasad & Ghose 2011, 2013). In these problems, the nodes are made to move toward the respective centroids using a gradient based proportional control law (Lloyd's algorithm) while the Voronoi cells and hence the centroids are recomputed as the nodes move. Eventually, the nodes reach the centroid of the respective Voronoi cells, asymptotically. The formulation and theoretical results can be found in (Cortes et al. 2004) and (Guruprasad & Ghose 2011, 2013).

1.3.3 Quadcopter dynamics

As the focus of thesis is on search using multiple quadcopters equipped with donward facing cameras, knowledge of their dynamics is very important in controlling their motion. We discuss the dynamics of quadcopters here.

Quadcopter dynamics is described with the two frames in which it operates as shown in 1.5. The inertial frame is the ground with gravity pointing in the negative z direction. The body frame is attached to the quadcopter and describes its orientation, with the rotor axes pointing in the positive z direction and the arms pointing in the x and y directions.

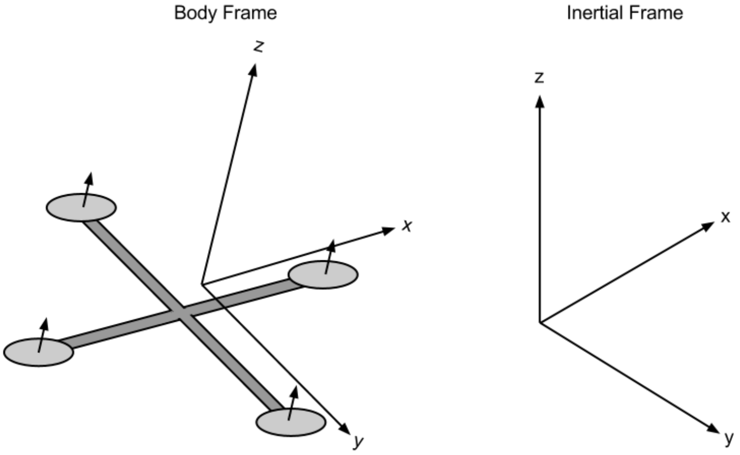


Figure 1.5: Quadcopter body frame and inertial frame

Kinematics

First, we discuss the kinematics in the body and inertial frames. We define the position and velocity of the quadcopter in the inertialframe as $\boldsymbol{x} = (x, y, z)^T$ and $\dot{\boldsymbol{x}} = (\dot{\text{x}},\dot{\text{y}},\dot{\text{z}})^T$, respectively. Similarly, we define the roll, pitch, and yaw angles in the body frame as $\theta = (\phi, \theta, \psi)^T$, with corresponding angular velocities equal to $\dot{\theta} = (\dot{\phi}, \dot{\theta}, \dot{\psi})^T$. However, note that the angular velocity $\omega \neq \dot{\theta}$. The angular velocity is a vector pointing along the axis of rotation, while $\dot{\theta}$ is just the time derivative of yaw, pitch,and roll. In order to convert these angular velocities into the angular velocity vector, we can use the following relation:

$$\omega = \begin{bmatrix} 1 & 0 & -s_\theta \\ 0 & -c_\phi & c_\theta s_\phi \\ 0 & -s_\phi & c_\theta c_\phi \end{bmatrix} \cdot \dot{\theta} \tag{1.1}$$

where, ω is the angular velocity vector in the body frame. The body and inertial frame are relate by a rotation matrix $\mathbb{R}$ which goes from the body frame to the inertial frame. This matrix is derived by using the $Z - Y - Z$ Euler angle conventions and successively undoing the yaw, pitch, and roll.

Moreover, the rotation matrix of the Quadcopters body must also be compensated during position control. The compensation is achieved using the transpose of the rotation matrix.

$$\mathbb{R}(\phi,\theta,\psi) = \mathbb{R}(x,\phi),\mathbb{R}(y,\theta),\mathbb{R}(z,\psi) \tag{1.2}$$

$$\mathbb{R}(x,\phi) = \begin{bmatrix} 1 & 0 & -s_\theta \\ 0 & -c_\phi & c_\theta s_\phi \\ 0 & -s_\phi & c_\theta c_\phi \end{bmatrix} \tag{1.3}$$

$$\mathbb{R}(y,\theta) = \begin{bmatrix} c_\theta & -s_\psi & 0 \\ s_\psi & c_\psi & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{1.4}$$

$$\mathbb{R}(z,\psi) = \begin{bmatrix} c_\psi & -s_\psi & 0 \\ s_\psi & c_\psi & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{1.5}$$

For a given vector $\vec{v}$ the body frame, the corresponding vector is given by $\mathbb{R}\vec{v}$ in the inertial frame.

Motors

All motors on the quadcopter are identical, hence a single motor analysis is mentioned without loss of generality. The adjacent propellers, however, are oriented opposite each other; if a propeller is spinning clockwise, then the two adjacent ones will be spinning counter-clockwise, so that torques are balanced if all propellers are spinning at the same rate. Brushless DC (BLDC) motors are used in most quadcopter applications. The torque produced by the electric motors is given by

$$\tau = K_t(I - I_0) \tag{1.6}$$

where, τ is the motor torque, I is the input current, '$I_0'$is the current when there is no load on the motor, and K_t is the torque proportionality constant. The voltage across the motor is the sum of the back-EMF and some resistive loss:

$$V = IR_m + K_v\omega \tag{1.7}$$

where, V is the voltage drop across the motor, R_m is the motor resistance, ω is the angular velocity of the motor, and K_v is a proportionality constant (indicating back-EMF generatedper RPM). We can use this description of our motor to calculate the power it consumes. The power is

$$P = VI \tag{1.8}$$

$$P = \frac{(\tau + K_t I_0)(K_t I_0 R_m + \tau R_m + K_t K_v \omega)^2}{K_t}$$

As the term $I_0 \ll \tau$ for simplicity, a negligible motor resistance term is assumed. Now, the power is proportional to the angular velocity

$$P \approx \frac{(\tau + K_t I_0) K_v \omega}{K_t}$$

Further simplifying this model, it is assumed that $K_t I_0 \tau$. Thus, the simplified equation for power:

$$P \approx \tau\omega\frac{K_v}{K_t} \tag{1.9}$$

Forces

The power is used to keep the quadcopter aloft. The energy the motor expends in a given time period is equal to the force generated on the propeller times the distance that the air it displaces moves ($Pdt = Fdx$). Equivalently, the power is equal to the thrust times the air velocity

$$P = F\frac{dx}{dt} \tag{1.10}$$

$$P = Tv_h \tag{1.11}$$

By assuming that the vehicle speeds are low, v_h is the air velocity when hovering. It is also assumed that the free stream velocity is zero (the air in the surrounding environment is stationary relative to the quadcopter). Momentum theory gives the equation for hover velocity as a function of thrust

$$v_h = \sqrt{\frac{T}{2A\rho}} \tag{1.12}$$

where, ρ is the density of the surrounding air and A is the area swept by the rotor.

The thrust is proportional to the square of angular velocity of the motor:

$$P \approx T\sqrt{\frac{T}{2A\rho}} \tag{1.13}$$

$$T = (\frac{K_v k_\tau}{K_t}\sqrt{2\rho A})^2$$

$$T = K\omega^2 \tag{1.14}$$

the thrust is proportional to propeller property and square of propeller rotation speed.

The total thrust of all the motors of the quadcopter on body frame is

$$T_B = (T_1 + T_2 + T_3 + T_4)$$

Propellers have the same size and pitch,so:

$$T_B = k({\omega_1}^2 + {\omega_2}^2 + {\omega_3}^2 + {\omega_4}^2)$$

Hence the forces and torques effected on the quadcopter can be estimated on each motor have the torque on quadcopter around z axis, as the drag forces as following equivalent:

$$F_D = \frac{1}{2}C_D A v^2 \rho$$

Where F_D is the frictional or Drag Force,

ρ is the air density,

C_D is the drag coefficient,

'A' is the area of the blade,

'v' is the velocity of the blade spinning.

Equations of Motion

To compute the torque, considering each rotor contributes about the body z axis. This torque is required to keep the propeller spinning and to provide thrust; it creates the instantaneous angular acceleration and overcomes the frictional drag forces.

In the inertial frame, the acceleration of the quadcopter is due to thrust, gravity, and linear friction. We can obtain the thrust vector in the inertial frame by using the rotation matrix R to map the thrust vector from the body frame to the inertial frame. Therotational equations of motion may be obtained from the Eulers equations for rigid body dynamics. Expressed in vector form, Eulers equations are written as

$$I\dot{\omega} + \omega \times (I\omega) = \tau \tag{1.15}$$

where, ω is the angular velocity vector,I is the inertia matrix, and τ is a vector of external torques.

$$\dot{\omega} = \begin{bmatrix} \dot{\omega}_x \\ \dot{\omega}_y \\ \dot{\omega}_z \end{bmatrix} = I(\tau - \omega \times (I\omega)) \tag{1.16}$$

1.3.4 Robot Operating System

The Robot Operating System (ROS) is an open source framework for robotics software development with its roots at Willow Garage and Stanford University. It consists of modular tools divided into libraries and supports different languages such as C++, Python and LISP. The concepts are described in detail in (Quigley et al. 2009). The ROS specification is at the messaging layer and consists of processes that can be on different hosts. Since it is a global collaborative open-

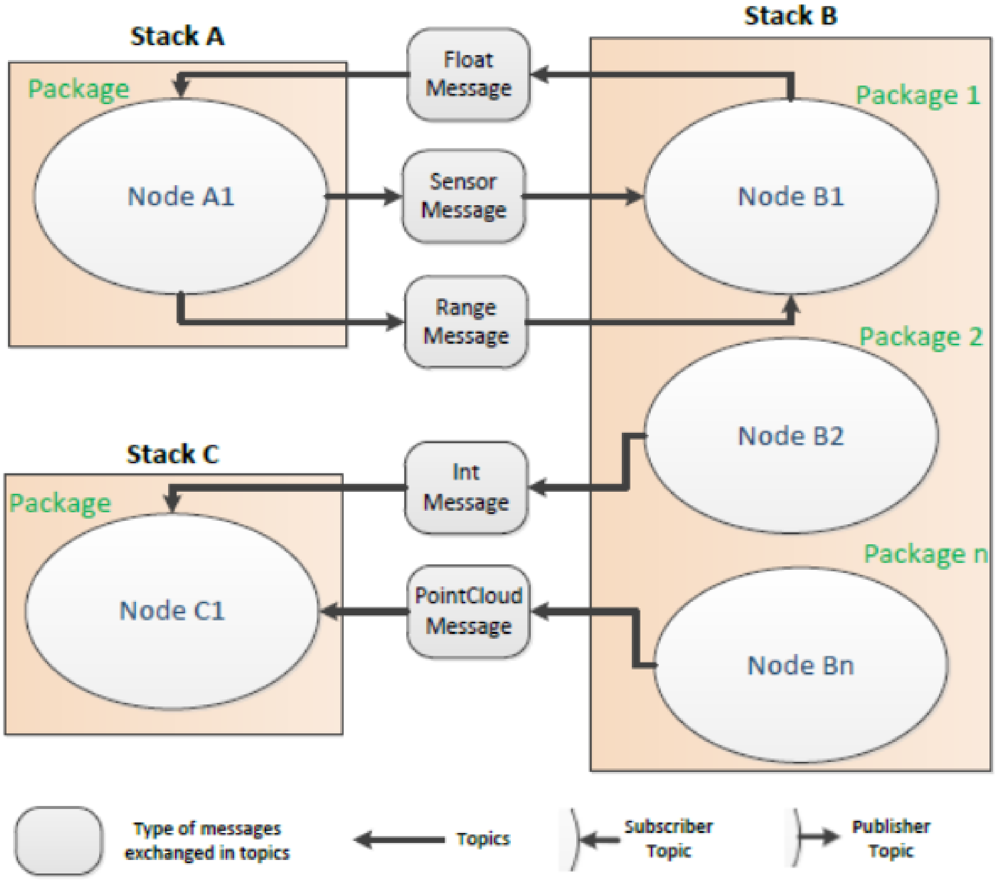


Figure 1.6: Example of a ROS network

source project in active development the code-base as a whole is in constant flux and lots of improvements are done for each new major release. ROS is a meta operating system for robotics that provides libraries and tools to help software developers to quickly and easily create robotic applications. ROS is also completely open source (BSD) and free to use, change and commercialize solutions based on it. In the Figure 1.6, an example of a ROS network is depicted. It is also shown how the network has the flexibility to work with a large variety of integrated solutions, which enables the assignment of particular tasks for specific members of the team. Robots from the same team may have different purposes and perform different tasks, thus heterogeneous teams with different capabilities can coexist. For example, in a search and rescue scenario, the cooperation of a multi-robotic team may arise from performing different tasks with hundreds or thousands of heterogeneous robots. The basic computation graph concepts of ROS are *nodes,*

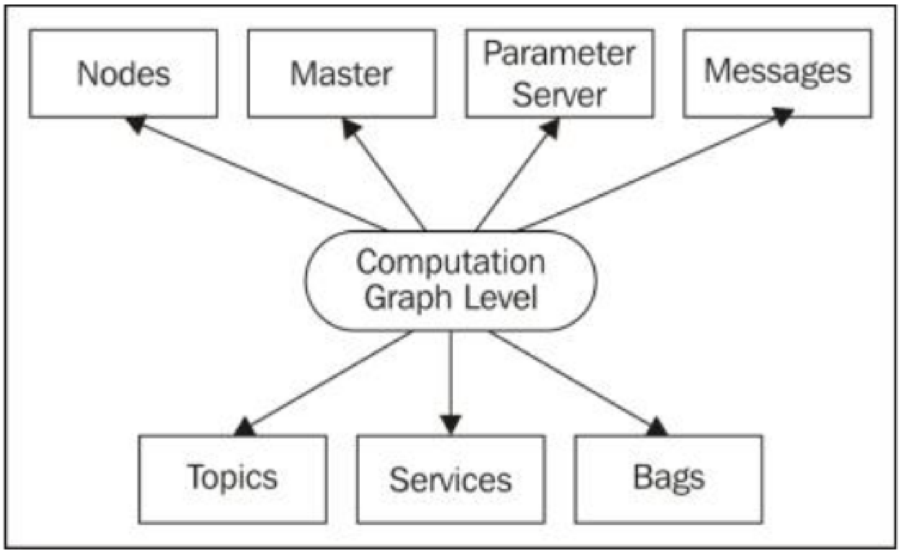


Figure 1.7: ROS-Computation graph Level

master, messages, services, and *topics*, all of which provide data to the graph in different ways. This concept can be summarized in Figure 1.7. The ROS Master provides the name registration and lookup to the rest of the computation graph. Without the Master, nodes would be unable to find each other, exchange messages, or invoke services. ROS is designed to be modular at a fine-grained scale. A robot control system usually comprises many nodes. A ROS node is written with the use of a ROS client library, such as roscpp for libraries in C++ or rospy for libraries in Python. Nodes send out a message by publishing or subscribing it to a given topic. The topic is a name that is used to identify the content of the message. In general, publishers and subscribers are not aware of each other's existence.

1.3.5 Transform(tf)

tf library (Foote 2013) is a ROS package designed to keep track of the coordinate frames and transform data in a robot. While working with robots, it is important that the robot is aware where it is and where the outside world is in relation to itself. The *tf* library is based on the concept of scene graph. Scene graphs are used for rendering 3D scenes as well as in robot simulators. This concept is used in Gazebo, a robot simulator, which is discussed in later sections. The *tf* is also very useful for multi-robot simulation. Scene graphs typically consist of a tree of objects to be rendered. The *tf* is used to transform data from sensor frame to the robot link coordinate frame. Every object is attached to a parent object with a position and other information. Depending on the application the other information can range from visualization meshes for pure rendering to update rules and inertial properties for the simulators. The *tf* package uses the tree data structure of the scene graph.

Transforms and coordinate frames are expressed as a graph with the transforms as edges and the coordinate frames as nodes. In this representation the net transform is the product of the edges connecting any two nodes. Figure 1.8 shows the transform tree of a robot having laser sensor which is connected to the base link of the robot.

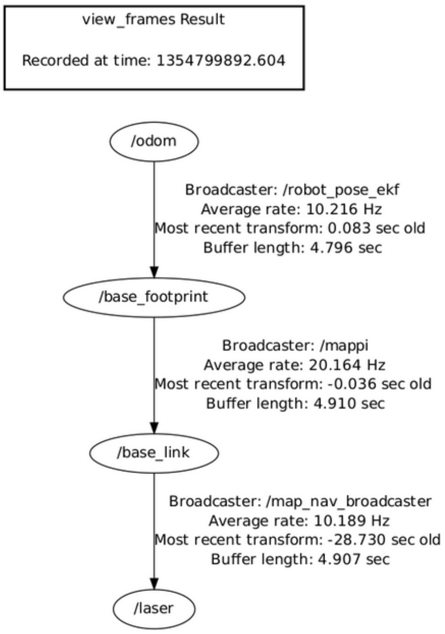


Figure 1.8: Structure of tf

In multi-robot simulation, the *tf* package is used to distinguish transforms of the robots from each other and to relate the robots coordinate frame with the world frame. Figure 1.9 shows the transform tree of two ‘Turtlesim’ robots with respect to world frame.

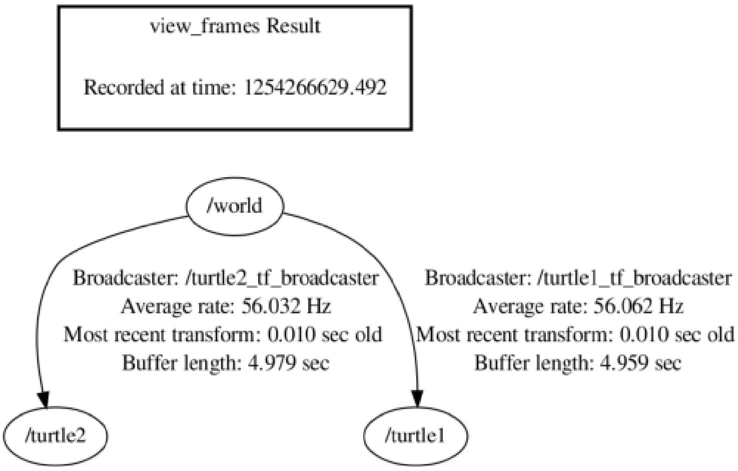

view_frames Result
Recorded at time: 1254266629.492
/world
Broadcaster: /turtle2_tf_broadcaster
Average rate: 56.032 Hz
Most recent transform: 0.010 sec old
Buffer length: 4.979 sec
Broadcaster: /turtle1_tf_broadcaster
Average rate: 56.062 Hz
Most recent transform: 0.010 sec old
Buffer length: 4.959 sec
/turtle2
/turtle1

Figure 1.9: Transform tree of two 'Turtlesim' robots

1.3.6 ROS launch files

roslaunch is a tool for easily launching multiple ROS nodes locally and remotely via SSH, as well as setting parameters on the Parameter Server. The *roslaunch* uses XML files that describe the nodes that should be run, parameters that should be set, and other attributes of launching a collection of ROS nodes.

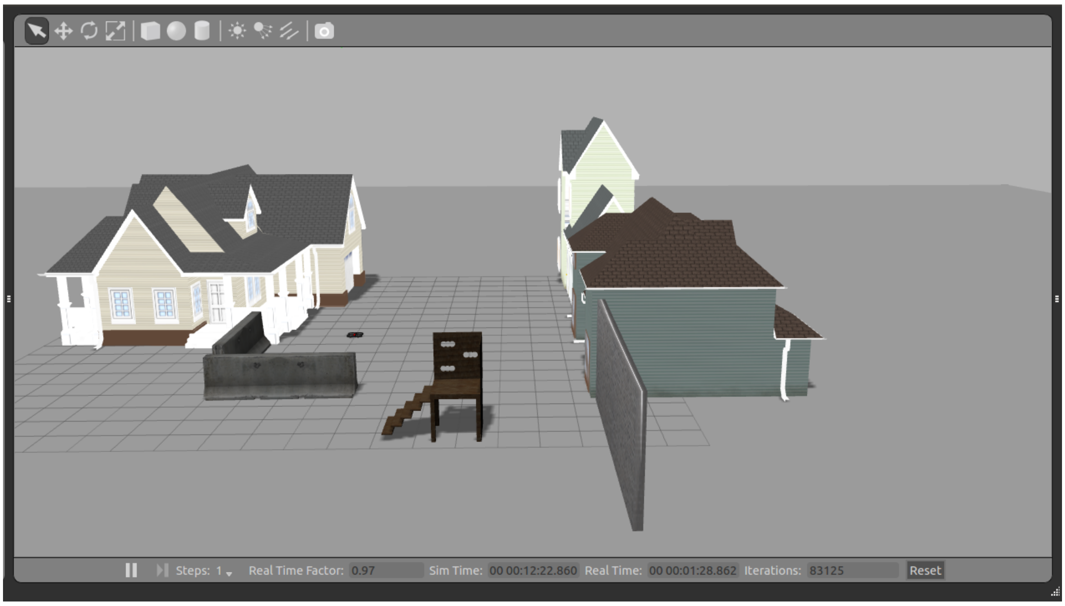
1.3.7 Gazebo simulator

Gazebo is a 3D simulator supported by ROS which is used in applications for a simulating a robot without depending physically on the actual machine. Using Gazebo, simple robot models can be created and tested in various environments. Also, the robot can be placed in different environments such as office cubicles, on the ground, etc. A library is available for the Gazebo simulator using that various pre-built models of robots and worlds which can be included in the simulation. The elements within the Gazebo simulation are

1. World-Collection of models, lights, plugins and global properties
2. Models-Collection of links, joints, sensors, and plugins
3. Links-Collection of collision and visual objects
4. Collision Objects-Geometry that defines a colliding surface

5. Visual Objects-Geometry that defines visual representation
6. Joints-Constraints between links
7. Sensors-Collect, process, and output data from sensors like IMU, Laser sensor, Kinect etc.
8. Plugins-Code attached to a World, Model, Sensor, or the simulator itself

Gazebo supports two types of model files, Simulation Description Format (SDF) and Unified Robot Description Format (URDF). Also 3D model formats such as COLLADA (.dae), Stereolithography (.stl), etc. can be directly opened in the gazebo environment. The detailed description of the concepts mentioned here are available in (http://gazebosim.org/tutorials). Figure 1.10 illustrates a Gazebo environment.



Steps: 1
Real Time Factor: 0.97
Sim Time: 00 00:12:22.860
Real Time: 00 00:01:28.862
Iterations: 83125

Figure 1.10: Gazebo simulation environment

1.3.8 Unified Robot Description Format (URDF)

The URDF is a standard used in modeling robotics in an XML format. The URDF can represent the kinematic and dynamic description of the robot, visual representation of the robot,and the collision model of the robot. Thus using the URDF format, the robot model can be fully described. A typical URDF file will contain the following tags

- Link: The link tag represents a single link of a robot. Using this tag, a robot link can be modelled and the properties of the model can be defined. The modeling includes size ,shape, color etc. or a 3D mesh in the .STL format can be imported to represent the robot link. Dynamic properties of the link such as inertial matrix and collision properties also can be specified in the link tag. The main components of link tag are as shown in Fig 1.11. The syntax for defining link is given below:

  ⟨link name="⟨name of the link⟩"⟩
  ⟨inertial⟩...........⟨/inertial⟩
  ⟨visual⟩ ....⟨/visual⟩
  ⟨collision⟩..........⟨/collision⟩
  ⟨/link⟩

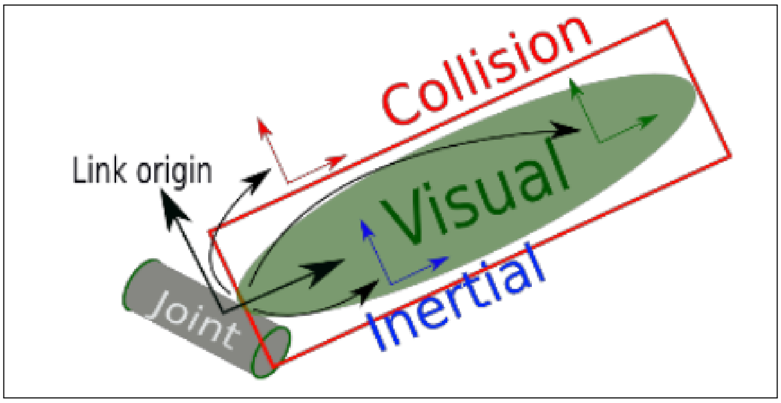


Figure 1.11: Representation of a link.

- Joint: The joint tag is used in the URDF file to represents a robot joint. The kinematics and dynamics of the joint can be specified under this tag. Also the limits of the joint movement and its velocity can be specified. The joint tag supports the different types of joints such as revolute, continuous, prismatic, fixed, floating, and planar. The joint is formed between two links, one is parent and the second one is the child. Figure 1.12 shows representation of a joint in URDF. The syntax used to define a joint is as shown below:-

  ⟨joint name="⟨name of the joint⟩"⟩
  ⟨parent link="link1"/⟩

⟨child link=”link2” /⟩

⟨limit effort.... /⟩

⟨/joint⟩

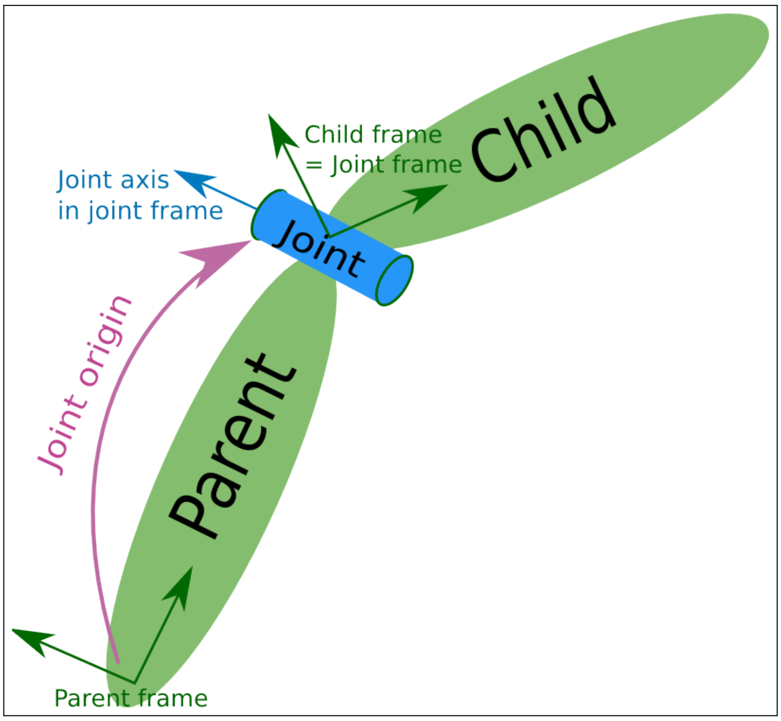


Figure 1.12: Representation of the joint

- Robot: - This tag encapsulates the entire robot model that can be represented using URDF.Inside the robot tag, the name of the robot, the links, and the joints of the robot are defined.The syntax is as follows:-

  ⟨robot name=”⟨ name of the robot⟩”⟩

  ⟨link⟩..... ⟨/link⟩

  ⟨link⟩ ...... ⟨/link⟩

  ⟨joint⟩ .... ⟨/joint⟩

  ⟨joint⟩ ....⟨/joint⟩

  ⟨/robot⟩

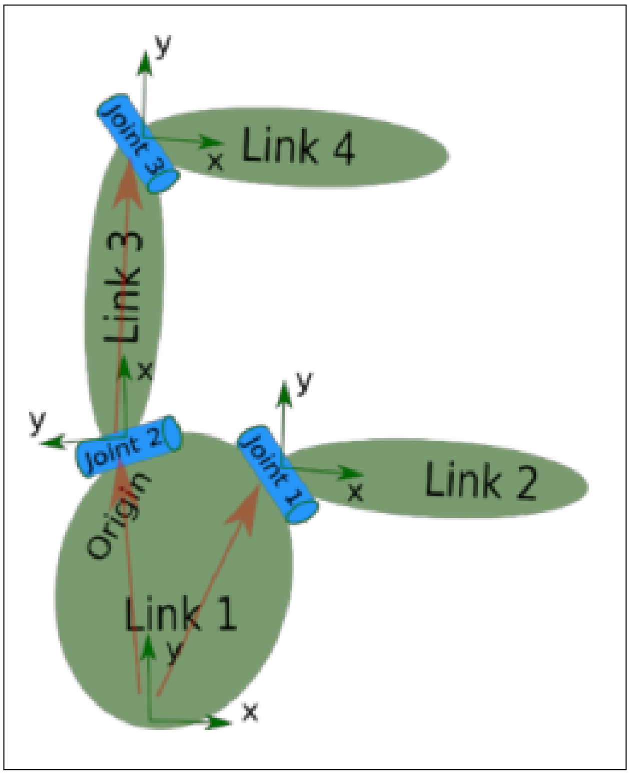


Figure 1.13: Representation of the robot

1.4 SUMMARY OF THE CONTRIBUTIONS

In this thesis we address a practically very useful, and academically challenging and interesting problem of cooperative multi-agent search.

We formulate a multi-agent search strategy using quadcopter UAVs as search agents/vehicles and downward facing cameras mounted on the quadcopters as search agents. Based on practical considerations, we assumed that the search effectiveness of the camera is maximum at the center and degrades away from it, unlike in most of the work in the literature where it is assumed to be constant over entire image frame. The lack of information about presence or absence or the targets of interest in the search space is modelled as an uncertainty density distribution. Here, the uncertainty is 1 when no information on the existence (or absence) of target at a point of interest is available and 0 when it is established that the target is either present or absent at that point. Based on uncertainty density distribution and the monotonically decreasing search effectiveness model, we address and formulate the problem of optimally deploying the quadcopters so as to maximize the uncertainty reduction (and hence information gain). Based on the observation we make on similar problem setting used in the literature, we formulate a ‘deploy’ and ‘search’ strategy using the concepts of centroidal Voronoi configuration, where the quadcopters get deployed into a centroidal Voronoi configuration, shown to be an optimal configuration maximizing the reduction in uncertainty, and then perform search resulting in reduction in the uncertainty. The process of optimal ‘deployment’ and ‘search’ continue until the average uncertainty over the entire search space is reduced below an arbitrary but fixed value, indicating the targets if presented are detected with an acceptable confidence (probability).

One of the very important component in multi-agent search is the search sensor itself and the spatial variation of its effectiveness in performing search, that is target detection. As we mentioned earlier, we assume a non-uniform effectiveness of the camera within its image frame. We first provide a detailed discussion on the spatial variation of the image quality both in terms of optical resolution and digital quality. We observe that the image quality is higher at the central pixel and degrades away from it. Such a scenario leads us to a non-uniform search effectiveness of the camera. We present an experimental setup to obtain a sensor effectiveness model for a downward facing camera using target detection probability. Through a set of target detection experiments carried out using AuRuco markers and triangular shaped objects as targets, we obtain sensor effectiveness model for a downward facing camera in different scenarios. We also establish that an exponential function with two parameters can be used to model the spatial variation of the camera’s search effectiveness (that is, the search effectiveness model).

We develop a platform using ROS/Gazebo and Matlab environment for simulation of the proposed multi-quadcopter search strategy in a hybrid centralized-decentralized architecture. The platform developed can be a very useful tool for conducting realistic simulation experiments to validate the proposed search strategy and to make a comparative study of its performance in terms of time required for the search process, with different parameters such as camera search effectiveness functions, sensor range, number of robots, and decide on the right parameters for any give mission.

We provide detailed results of experiments and simulation carried out along with a detailed discussion on the same. First we present the results of the experiments carried out to obtain the search effectiveness model of the downward facing camera, which we use in the proposed multi-quadcopter search strategy. Though we used the experimental setup to establish, that in general, an exponential function with two parameters can be as the search effectiveness model of a camera, it can be used carry out experiments with specific type of imaging sensor, type of image processing tools, kind of environment in which has to detect the targets, the type of targets that need to be detected, and obtain a suitable search effectiveness model.

We present representative results of the simulation experiment carried out using the realistic ROS/Matlab simulation platform, both to demonstrate the simulation platform itself and the proposed search strategy. Finally, we provide a detailed account of simulation experiments carried out to evaluate the effect of number of search quadcopters and the camera effectiveness parameters on the performance of the proposed multi-quadcopter search strategy, using the simulation platform developed in this work.

The simulation platform developed can be used to carry out experiments using physical AR Drones. The controller used within the simulation environment may be used to control the physical AR Drones. Also, the simulation environment can be used to conduct a large number of simulation and physical experiments to decide on parameters such as the optimal number of quadcopters, type of cameras used (in terms of their search effectiveness, which may be obtained by using the experimental setup based on that used in this work), for a given search scenario. In this sense, the experimental setup and simulation platform developed are useful beyond the sample results provided in this thesis and will surely help the proposed multi-agent search strategy take a step forward from theory to experiment and then finally into reality.

1.5 Organization of the thesis

The thesis is organized as follows. We provide a detailed survey of representative relevant work from the literature in the past on the problem on searching for targets. We focus our attention on work that address the problem of searching for a single or multiple, moving, smart, evading, or stationary targets of interest using multiple cooperating autonomous agents. We group the literature based on the techniques being used, type of agents used, type of sensor used for performing search, the targets being searched, the application domain of focus, the space being search in terms of continuous or gridded, and finally based on the sensor footprint used. After a detailed discussion, we provide the research gap motivating the work carried out in this thesis.

In Chapter 3 we discuss the problem being addressed in this thesis, the multi-quadcopter search using downward facing camera. First we provide a discussion on the problem setting that is used in this work. Next we provide a detailed description of the proposed 'deploy and search' multi-quadcopter search strategy using the downward facing cameras mounted on the quadcopters as search sensors.

The effectiveness of camera as a search sensor is discussed in detail in Chapter 4. Here, we define a search effectiveness function of a search sensor based on the target detection probability. First we discuss the search effectiveness model of the downward facing camera that is used in this work based on the work reported outside the multi-agent search literature, and discuss how they are relevant and useful for the multi-agent search problem addressed in this thesis. Then we present an experimental setup which is used to obtain the search effectiveness model for any downward facing camera, using target detection experiments with markers and triangular shaped objects as targets.

We present a hybrid ROS/Matlab simulation platform for carrying out realistic simulation experiments with multiple quadcopters using the proposed multi-quadcopter search strategy. We first describe the hybrid architecture used for implementation along with the motivation and justification for the architecture used. Then we provide a detailed description of the simulation platform developed as part of this work in chapter 5.

In chapter 6 we present detailed results of experiments and simulation carried out along with a detailed discussion on the same. First we present the results of the experiments carried out to obtain the search effectiveness model of the downward facing camera, which we use in the proposed multi-quadcopter search strategy. Next we present a representative result of the simulation experiment carried out using the realistic ROS/Matlab simulation platform, both to demonstrate the simulation platform itself and the proposed search strategy. Finally, we provide a detailed account of simulation experiments carried out to evaluate the effect of number of search quadcopters and the camera effectiveness parameters on the performance of the proposed multi-quadcopter search strategy, using the simulation platform developed in this work.

Finally the thesis is concluded in Chapter 7 where we summarize the contribution of the work along with a discussion on possible directions for future work.

CHAPTER 2

LITERATURE REVIEW

As we have discussed in the previous chapter, searching for the presence of targets of interest such as survivors in a disaster is an interesting and practically useful and challenging problem, and attracted researchers from a wide variety of fields such as from search theory, motion planning, Unmanned Vehicular (Aerial, ground, surface, and underwater) technology, optimization, sensor technology, sensor fusion. etc. Some of the earliest literature (Stone 1975, Koopman 1980, Lida 1992) on this problem of searching for targets in an unknown environment assume certain restrictive conditions. These seminal contributions were mostly theoretical in nature and applied to a single agent searching for a single or multiple, and static or moving, targets. It is likely that the same task can be accomplished more effectively by multiple searchers. But, when multiple agents are involved, coordination between them becomes an important issue. Benkoski et al. (1991) provide one of the earliest survey of more than 200 papers on search using a single agent.With the advent of autonomous vehicle technology, communication, sensor technology, and advances in the theory of optimization, sensor fusion, the interest is shifted to search using multiple cooperative agents. There is a vast literature on coordinated multi-agent search problem and in this chapter we preview a few representative works from the literature on this problem.

2.1 TECHNIQUES USED FOR MAS

We first provide a brief account of various techniques that are used by researchers in the literature to solve the multi-agent search problem addressed in this thesis. Authors in (Ousingsawat & Earl 2007, Enns et al. 2002, Spires & Goldsmith 1998, Vincent & Rubin 2004, Schlecht et al. 2003) use pre-defined lanes to address the problem of path planning for the agents performing the search. In (Beard & McLain 2003, Flint et al. 2003, Pfister 2003, Flint et al. 2002), the authors use the dynamic programming approach to solve the problem of path planning of agents in a gridded environment optimizing a suitably defined search performance.Yang et al. (2002*b*,*a*, 2007), Strens (2004) use learning methods to aid in solving the cooperative searching problem.Rajnarayan & Ghose (2003), Dell et al. (1996), Kitamura et al. (1993), Sujit (2006), Baum & Passino (2002) use techniques from team theory, graph theory, and game theory to address the problem of coordination between the searching agents. Search theoretic techniques have been used in works such as in (Baum & Passino 2002, Bertuccelli & How 2005). Bayesian approach is used to handle the probability of target detection and sensor fusion in works such as in (Khan et al. 2014, Hu et al. 2013, Bourgault et al. 2003, 2004, Ru et al. 2015, Zhang et al. 2017). Altshuler et al. (2008), Vincent & Rubin (2004), Kothari et al. (2013) use formation flight which converts a single agent sweeping to a multi-agent scenario.

2.2 TYPES OF AGENTS USED IN MAS

The motion capabilities of an agent depends on the kind of the agent such as Ground vehicles: differential wheel or skid steering; UAVs: fixed-wing or rotor-based, etc. For example, a fixed-wing UAV has a constraint on minimum turning radius, while a quadcopter does not have such a constraint. The path planning strategy should be able to handle the requirements and constraints of a specific agent (vehicle) used. Hence, the type of agents dictates the multi-agent problem strategy. Different types of agents ranging from abstract ‘agents’ to ‘UAVs’ have been used in the literature.

Kitamura et al. (1993), Dell et al. (1996), Parker (1997), Polycarpou et al. (2001*a*,*b*), Yang et al. (2002*a*), Finke et al. (2003), Rajnarayan & Ghose (2003), Strens (2004), Sujit (2006), Guruprasad (2009), Guruprasad & Ghose (2011, 2013) refer to the search vehicles as ‘agents’, where in principle any type of autonomous vehicles may used in reality. However, typically most generic strategies presented in these works are more suitable for UAVs carrying suitable sensors as search agents. Authors in (Altshuler et al. 2008, Baum & Passino 2002, Beard & McLain 2003, Beard et al. 2002, Bertuccelli & How 2005, 2006*a*,*b*, Delle Fave et al. 2010, Finke et al. 2003, Flint et al. 2002, Gan & Sukkarieh 2011, Gan et al. 2012, Hu et al. 2013, 2014, Jin et al. 2006, 2003, Khan et al. 2014, Kothari et al. 2013, Lanillos et al. 2014, Li et al. 2018, Manathara et al. 2011, Meng et al. 2014, Millet et al. 2010, Mirzaei et al. 2010, 2011, Sharifi et al. 2013, 2015, Passino et al. 2002, Peng et al. 2010, Pugliese et al. 2016, Riehl et al. 2011, Ru et al. 2015, Scerri et al. 2004, Sujit & Ghose 2004, Sujit & Beard 2008, Waharte et al. 2009, Yang et al. 2017, Yang 2005, Yang et al. 2002*b*, 2007, York & Pack 2012, Zhang et al. 2017) mention using UAVs in general as the autonomous vehicles used as search agents. Li et al. (2018) use a single UAV, Meng et al. (2014) specifically mention fixed-wing UAV as search agents. In (Kuhlman et al. 2017, Pugliese et al. 2016) authors use 'drone' s for a mobile target covering problems. Khan et al. (2015) use MAV (mini UAV) and Engel (2011), Engel et al. (2012), Waharte et al. (2009), Khan et al. (2014) use quadcopters for performing search. A few work such as (Engel 2011, Engel et al. 2012, Waharte et al. 2009, Khan et al. 2014) mention using quadcopter UAVs as search agents.

In (Bourgault et al. 2004, Orgas et al. 2004, Burgard et al. 2002, Parker 1997, 2002, Freda & Oriolo 2007) authors use ground vehicles (AGVs) to perform target search, while AUVs have been used in (Galceran et al. 2015) and ASV/USVs have been used in (Meghjani et al. 2016). In an interesting paper by Kolling et al. (2011), the authors use a human agent guided by the search methodology developed. While most works in the literature use homogeneous search agents and sensors, heterogenous vehicles have been used in (Lum et al. 2006) (UAV + USV), (Mirzaei et al. 2011, Sharifi et al. 2015) (Search UAV + service UAV), (Scerri et al. 2004) (UAV + Munitions – Wide Area Search Munitions), and (Jin et al. 2003, 2006) (UAVs - heterogeneous in the task being performed).

2.3 TYPE OF SEARCH SENSOR USED IN MAS

Along with the type of agent used in the MAS problem formulation, which dictates the kind of path planning, the type of sensor used is also important in a MAS problem formulation. The sensor may also depend on the type of 'target' being searched. For example, a thermal sensor may be suitable for detecting a forest fire. Most works in the literature (for example that by Guruprasad & Ghose (2013)) consider a generic sensor and an abstract form of target detection mechanism. However, a few researchers focus on specific search sensors while formulating and solving the MAS problem. A camera is used in (Engel 2011, Engel et al. 2012, Freda & Oriolo 2007, Hu et al. 2014, Zhang et al. 2017), while radar is used in Li et al. (2018). Electro-optical and IR sensors are considered in York & Pack (2012).

Sun et al. (2016) use downward-facing cameras mounted on UAVs as search sensors. The captured video is transmitted to a ground station, where it is post-processed to detect the targets. The UAV motion planning (guidance/control) is decoupled from the target detection or the search process. Camera mounted UAVs are also used to detect defects in large structures such as bridges automatically (Ikeda et al. 2019).

Apart from the suitability of a search sensor for a given target of interest, the details and characteristic of a given sensor also affect the problem formulation and the solution/search strategy. Formulations based on generic sensors assume certain generic conditions, which may not be equally applicable to all specific sensors. Even when a camera is used in a multi-agent search problem, its orientation (front-facing or downward-facing), characteristics such as range, image quality, spatial variation of the image quality, etc., affect both the formulation and the search strategy. In this thesis, we focus on downward-facing camera as the search sensor and consider its characteristics that affect the target detection capability while formulating the multi-agent search strategy.

2.4 TARGETS BEING SEARCHED

Very few authors address the problem of searching for specific targets such as Sharifi et al. (2015), who address detection of forest fire, while most authors address generic targets. Authors in (Dell et al. 1996, Freda & Oriolo 2007, Lum et al. 2006, Parker 1997, 2002, Ru et al. 2015, Bertuccelli & How 2006*a*,*b*, Hu et al.

2014, Ru et al. 2015) address the problem of multiple moving targets, Grundel (2005) and Riehl et al. (2011) address the problem of searching for a single moving target by multiple agents. While Kolling et al. (2011), Vincent & Rubin (2004), Altshuler et al. (2008) address a problem of searching for smart or evading targets, Millet et al. (2010) address the problem of searching for targets that pop-up. Most other researchers consider multiple stationary targets. Maki et al. (2019) address the problem of tracking sea turtles using an AUV.

2.5 APPLICATION DOMAIN FOR MAS

Most researchers do not address any specific application domain. Jin et al. (2003, 2006) specifically address a battlefield scenario. Much other research work considers searching in a risky or hazardous environment (such as (Flint et al. 2003, Beard & McLain 2003, Vincent & Rubin 2004), etc.). Though several authors (such as Guruprasad & Ghose (2011)) mention a scenario of natural disaster, Sharifi et al. (2015) (forest fire) Kuhlman et al. (2017) (earthquake) specifically mention a civilian application involving a natural disaster.

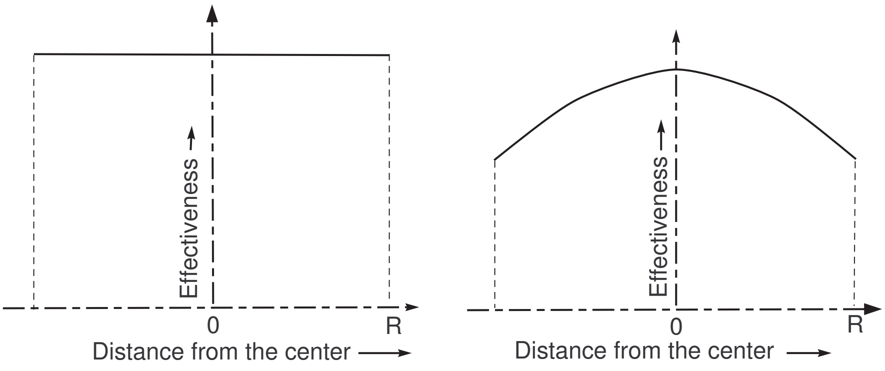
2.6 CONTINUOUS VS GRIDDED ENVIRONMENT

In most situations the search space is gridded into small cells for implementation of the MAS strategies from a practical implementation perspective. However, the decision-making process, such as deciding on an optimal path for the search agents may use a continuous space or a gridded space. A search strategy involves primarily two sub-tasks: Search, that is target detection, and Path planning for the agent. Path planning has to take into account the search effectiveness in some way. For example, the agent path should minimize the search time and maximize a suitably defined search effectiveness (such as rate of information gain). Most researchers using techniques such as dynamic programming, graph theory, search theory, game theory, team theory, etc. depend on a grid-based representation of the search space. The decision making process is fundamentally to decide on which is the next best cell the agent should move. However in Guruprasad (2009), Guruprasad & Ghose (2011, 2013) use a continuous space for this decision making process. Here the authors use the principles from locational optimization problems in continuous space. Works that use pre-defined lanes may also use continuous space for path planning.

As discussed here, most works in the literature use discretized space (gridded) for 'planning' the next move of the search agents. While such a method is suitable for optimization methods such as 'dynamic programming' and useful for ground robots, a path planned in continuous space is more suitable for UAVs. Instead of planning a path using grid-based or 'discretized space' and then converting the same into a smooth path in continuous space, as in the most works in the literature, Guruprasad & Ghose (2011) base their planning directly in the continuous space. As the suitability of 'discrete' or 'continuous' space depends on the problem formulation and the solution provided in the individual works in the literature, they may not be directly comparable in a quantitative sense.

2.7 SENSOR EFFECTIVENESS/FOOTPRINT USED IN MAS

The search component of the problem which is associated with target detection using the onboard search sensors is also an important component of a MAS strategy. The spatial search effectiveness of the sensors or the sensor footprint plays an important role in formulating and solving the MAS problem. Figure 2.1 shows different types of sensor footprint (spatial search effectiveness) that are typically used in the MAS literature. In Figure 2.1(a), any region that comes under the search sensor is considered searched, in the sense that the target is detected if it exists. The area of sensor footprint is typically considered as a cell. In a few works, the sensor footprint may cover several gridded cells. The problem of path planning for such scenarios is identical to exhaustive search (such as in Spires & Goldsmith (1998)) or area coverage problems(such as in Choset (2001)). In most literature, on MAS a sensor footprint as shown in 2.1(a) is used where a single search over a cell is not sufficient to detect the target



(a) (b)

Figure 2.1: A (a) flat and (b) non-flat sensor footprint.

completely. In these situations, the target detection probability (modeled in a wide variety of techniques in the literature), provide that the target is present in a given cell will increase with number of passes over the cell. Most researchers use this model. A more practically realistic spatial effectiveness model for a search sensor is as illustrated in Figure 2.1(b). The effectiveness of the sensor is maximum directly below it and monotonically decreases away from its center. Such models have been used in sensor network literature to solve optimal sensor deployment problem such as in (Cortes et al. 2004, Schwager et al. 2009). Guruprasad (2009), Guruprasad & Ghose (2011, 2013) and Delle Fave et al. (2010) use exponentially decreasing sensor effectiveness models. While Schwager et al. (2009) and Delle Fave et al. (2010) use it for a downward-facing camera, Guruprasad & Ghose (2011) use such as model for a generic sensor.

In the following, we provide a brief survey of a few representative work form the literature in a partial chronological order.

Flint et al. (2002) address the problem of cooperative control for multiple autonomous UAV's searching for targets. Here the authors assume that *a priori* data about target distribution is available, and use a dynamic programming technique to solve the problem. In (Flint et al. 2003) author consider a problem of searching and uncertain and risky environment. The authors provide methods to incorporate a priori and dynamic information about the targets and to incorporate threats in a three-dimensional environment into a computationally feasible dynamic programming approach. A concept of “search gain” for a cell which authors define as the number of expected real targets that will be detected by a sensor sweep of that cell is used in the problem formulation. A search map indicating the probability of target detection is updated as the search progresses. In (Jin et al. 2003, 2006) author consider a heterogenous team of UAVs, where each group of UAVs perform tasks such as search, combat, attack, and battle damage assessment, in a search and destroy mission over a battlefield. Each class of UAVs has its sensing and attack capabilities concerning different target types. A simple cooperative approach to this problem, based on distributed assignment mediated through centralized mission status information has been presented. In (Beard et al. 2002) authors address a problem of coordinated target assignment and intercept for UAVs. Beard & McLain (2003) use a team of unmanned air vehicles (UAVs) to cooperatively search, an area of interest that contains regions of opportunity and regions of potential hazard. The objective of the UAV team here is to visit as many opportunities as possible while avoiding as many hazards as possible.

In (Bourgault et al. 2003) a decentralized Bayesian approach to coordinating multiple autonomous sensor platforms searching for a single non-evading target is presented. Here each decision maker builds an equivalent representation of the target state PDF through a Bayesian DDF network enabling them to coordinate their actions without exchanging any information about their plans. In (Bourgault et al. 2004) authors address the problem of coordinating a team of multiple heterogeneous sensing platforms searching for a single lost target, using decentralized Bayesian negotiation approach. Decentralized cooperative planning is achieved via anonymous negotiation based on the communication of expected observed information. Scerri et al. (2004) present a problem coordinating a large number of wide Area Search Munitions (WASMs), which are part UAV and part munition.

Sujit & Ghose (2004), Sujit (2006) use concepts of graph theory and game theory to solve the problem of coordinated multi-agent search. Here, the authors partition the search space into cells and use graph theory to model the connectivity of cells and use concepts from game theory to devise trajectories of agents that can move one cell at a time. Each cell is associated with an uncertainty value modelling the lack of information on the presence or absence of the target in that cell. Also, the effectiveness of a search sensor is considered maximum at the cell which coincides with its center and lower in the surrounding/neighboring cells. In (Sujit & Ghose 2004) propose a search algorithm based on the k-shortest path algorithm that maximizes the effectiveness of the search in terms of searching through the maximum uncertainty region, given a constraint on the endurance time of the UAV and on the location of the base station from which the UAVs operate. In (Sujit & Ghose 2009) authors propose negotiation schemes for the multi-agent search problem.

Vincent & Rubin (2004) design and analyze the performance of cooperative search strategies for UAVs searching for moving, possibly evading, targets in a hazardous environment. Here the UAVs work together by arranging themselves into a flexible flight configuration that optimizes their integrated sensing capability.

Yang et al. (2004), Yang (2005) present a decentralized control model for cooperative search using multiple UAVs and achieve online cooperation among vehicles by treating the possible paths of other vehicles as "soft obstacles" to be avoided. Further using the approach of "rivaling force" between vehicles to enhance cooperation, each UAV takes into account the possible actions of other UAVs such that the overall information about the environment is increased. Yang et al. (2007) present a practical framework for online planning and control of a group of UAVs for cooperative search based on two interdependent tasks: (i) incrementally updating cognitive maps used as the representation of the environment through new sensor readings; (ii) continuously planning the path for each vehicle based on the information obtained through the search. The authors formulate the cooperative search problem and develop a decentralized strategy based on an opportunistic cooperative learning method (introduced in their earlier work (Yang et al. 2002*b*,*a*)), where the emergent coordination among vehicles are enabled by letting each vehicle consider other vehicles actions in its path planning procedure. Authors ensure that physically feasible paths for the vehicles are generated, where constraints on aerial vehicles, including physical maneuverabilities are considered. The proposed strategy also guarantees a complete search of the environment and is robust to a partial loss of UAVs. Bertuccelli & How (2005) address a problem of Multi-UAV search for environment with imprecise probability maps using the search the theoretic formulation in a gridded environment. The authors propose an approach to calculate the minimum number of looks needed to achieve a given level of confidence of target existence.

In (Bertuccelli & How 2006*a*) the authors extend the work for the case of moving targets. Here the authors consider probabilistic target motion, creating Uncertain Probability Maps (UPMs) that take into account both poor knowledge of the probabilities and the propagation of their uncertainty through the environment. In (Bertuccelli & How 2006*b*) authors use a discrete-state, discrete-time Markov chain-like model to model the target motion.

Grundel (2005) considers a problem of constrained path planning for one or two agents in search of a single randomly moving target such that we maximize the probability of intercepting the target at some time in its trajectory. The authors assume that the agents operate in a receding horizon optimization framework with some finite planning horizon.

Lum et al. (2006) address the occupancy-based map searching using heterogeneous teams of autonomous vehicles such as UAVs and USVs. The problem of agents converging on the targets and searching the unexplored regions is formulated as a model predictive control problem. Trodden & Richards (2008) also use a distributed model predictive control strategy.

Altshuler et al. (2008) address the cooperative hunters problem, where a swarm of UAVs)is used for searching one or more evading targets, which are moving in a predefined area while trying to avoid a detection by the swarm. Here the authors use the geometric flight configurations/formation of UAVs to optimize their integrated sensing capabilities.

Schwager et al. (2009) address a problem of optimal coverage for multiple hovering robots with downward-facing cameras, though not for a search problem. Waharte et al. (2009) address a problem of coordinated search with a swarm of UAVs, based on a distributed, grid-based probabilistic environmental model. The authors use quad-copters as search agents. Delle Fave et al. (2010) considers the coordination of a team of UAVs that are deployed to search for a moving target within a continuous space. The authors present an online and decentralized coordination mechanism based on the max-sum algorithm. The authors use a non-uniform sensor effectiveness which is maximum directly below it and decreases exponentially away from it.

Peng et al. (2010) use a decentralized optimization search method based on distributed model predictive control (DMPC) for the problem of cooperative area search for UAVs. Here a centralized on-line optimization decision of the whole multiple UAV system is decomposed into the decentralized optimization of several single UAV subsystems under the framework of DMPC, and a Nash optimality and particle swarm optimization (PSO) based algorithm is implemented to the solution of the decentralized optimization.

Millet et al. (2010) present a method for a team of multiple UAVs with finite communication range to perform an efficient continuous search of an area of interest to find pop-up targets. As agents plan their motion to search high probability regions of the area of interest, they simultaneously strive to maintain a connected communication topology with the other agents.

Mirzaei et al. (2011) addresses a problem of cooperative multi-UAV search with communication delay. In this work, a decentralized approach is proposed to solve a cooperative multi-vehicle search and coverage problem in uncertain environments. Two different types of vehicles are used for search and coverage tasks. The task of service vehicles here is to spread out over the environment to optimally cover the terrain. Authors use locational optimization techniques to assign Voronoi regions to vehicles. In (Sharifi et al. 2015) authors use a similar setting for a problem of a cooperative multi-vehicle search and coverage problems in an uncertain environment such as forest fire monitoring and detection.

Gan & Sukkarieh (2011) use a team negotiation technique using a decentralized gradient-based optimization algorithm for coordinating a team of autonomous sensor agents searching for targets in a large scale environment. In this work, the target state is represented in its belief form which is a Probability Density Function (PDF) over the target state space. in Gan et al. (2012) focus on the addresses the problem of explicitly avoiding inter-agent collisions in a team negotiation process. Team negotiation is performed using a decentralized gradient-based optimization approach, while safety distance constraints are designed and handled using Lagrangian multiplier methods. The novelty of work as claimed by the authors is the demonstration of a decentralized form of inter-agent collision avoidance in the loop of the agents real-time group mission optimization process.

In Kolling et al. (2011) address the problem of searching for moving targets by human agents. To solve this problem the authors use an annotated height map, a graph representation, and search strategies based on worst-case assumptions about all targets.

Manathara et al. (2011) address a problem where multiple UAVs have search support and prosecute operations. In this work, the authors proposed decentralized sub-optimal (polynomial time) and decentralized optimal coalition formation algorithms that generate coalitions for a single target with low computational complexity.

Riehl et al. (2011) present a receding horizon cooperative search algorithm for multiple UAVs searching for a mobile target using model predictive approach and dynamic graphs. The authors reduce the continuous search problem into a sequence of optimizations on a finite, dynamically updated graph, whose vertices represent waypoints for the searchers and edges indicate potential connections between the waypoints. The optimization criterion measures the probability of finding the target per unit travel time.

York & Pack (2012) present a comparative study to evaluate the merits of a cooperative unmanned aerial sensor (UAVs carrying search sensors) against a single system with equivalent capabilities. The authors here quantify the advantage of multiple cooperative UAVs over a single UAV using a case study of searching and detecting ground targets with electro-optical (EO) and infrared (IR) signatures.

Kothari et al. (2013) present a distributed target-centric formation control strategy for multiple UAVs in the presence of target motion uncertainty. The formation is maintained around a target using a combination of a consensus protocol and a sliding mode control law.

Hu et al. (2013) address the problem of cooperative search for multiple stationary ground targets by a group of UAVs with limited sensing and communication capabilities. Each agent keeps an individual probability map and updates the map individually with measurements according to the Bayesian rule. A nonlinear transformation of the probability map is introduced to simplify the computation by linearizing the Bayesian update. A consensus-like distributed fusion scheme is proposed for multiagent map fusion. in Hu et al. (2014) the authors use an airborne camera on each of the UAVs.

Khan et al. (2014) focus on strategies for merging occupancy probabilities of target existence in multi-UAV cooperative search. Here the authors assume that small-scale UAVs (e.g., quadrotors) with communication range limitations move in a given search region following pre-defined paths to locate a single stationary target. The proposed merging strategies perform Bayes updates of the occupancy probabilities while considering realistic limitations in sensing, communication and UAV movement all of which are important for small-scale UAVs. Khan et al. (2015) considers a network of autonomous MAVs cooperatively searching for targets. The objective is to minimize the search time while considering sensing and communication limitations. Authors formulate the cooperative search as a traveling salesman problem. Authors consider two sub-problems: information merging as in Khan et al. (2014) and decision-making. Authors establish that depending on the availability of information and capability of making decisions, the MAVs can search an area more efficiently if information merging and decision-making are distributed.

In Lanillos et al. (2014) the authors address the problem of coordination of a team of autonomous sensor platforms searching for lost targets under uncertainty. A real-time receding horizon controller in continuous action space is developed based on a decentralized gradient-based optimization algorithm and by using the expected observation as an estimate of future rewards. Meng et al. (2014) design control logic and optimize flight paths for fixed-wing UAVs that are required to cooperatively search for potential targets in the area of operation (AO) and keep monitoring and tracking the found targets according to a certain predefined minimal revisit time.

Ru et al. (2015) propose a distributed Multi-UAVs cooperative search control method for moving target to reduce the impact of uncertainties caused by unknown motion parameters on the searching plan and improve the efficiency. The target the probability map is updated, based on detection results obtained from onboard sensors, using Bayesian theory. A Gaussian distribution of target transition probability density function is introduced to obtain the prediction probability of the existence of moving target, and then target probability map is further updated in real-time. A performance index function combining target cost, environment cost, and cooperative cost is constructed, and the cooperative searching problem can be transformed into a central optimization problem. Then a distributed model predictive control method is presented to obtain control command of each UAV.

Galceran et al. (2015) address the problem of coverage path planning for autonomous underwater vehicles (AUVs) for inspection of three-dimensional underwater structures. In Meghjani et al. (2016) authors address the problem of searching multiple non-adversarial targets using a mobile searcher (USVs or ASVs) in an obstacle-free environment for marine applications (such as marine environmental monitoring, drifting debris, or lost divers in open water) where the targets drift on the ocean surface. The authors propose three classes of search strategies, namely, data-independent, probabilistic and hybrid search. The data independent search strategy follows a pre-defined search pattern and schedule. The probabilistic search strategy is guided by the estimated probability distribution of the search target. The hybrid strategy combines data-independent search patterns with a probabilistic search schedule.

Zhang et al. (2017) address the cooperative search problem for a team of UAVs with limited field-of-view (FOV) and available overload constraints. The author establishes the models of the environment, UAV, image sensor, and communication, and then propose a modified distributed information fusion strategy based on the Bayesian rule. The authors propose a distributed gradient-based optimization method for path planning involved in the cooperative search taking into account the available overload constraint of the UAV. The authors address explicit collision avoidance by establishing reasonable safety distance constraints using the Lagrange multiplier.

In Yang et al. (2017), a problem of Multi-UAV search in an environment which is completely unknown and is dynamically changing using Ant Colony theory. Kuhlman et al. (2017) consider a disaster scenario such as an earthquake or floods, where finding survivors a few hours sooner results in a dramatic increase in saved lives. The authors propose using drones to expedient rescue operations. Entropy is used to quantify the uncertainty. The authors present an anytime algorithm, based on best first branch and bound, for autonomous multipass target search in natural environments.

In addition to the target search problem, object detection using UAV mounted cameras is useful in patrolling and other related applications (Zhou et al. 2019). Several works in the literature focus on image processing and target detection using UAVs (Mohan et al. 2017).

2.8 RESEARCH GAP AND MOTIVATION

We make a few observations based on the literature review:

1. Most work in the literature either consider a generic agent, a generic UAV, or fixed-wing UAVs as search agents. Thus, the path planning strategy is either too general or too specific for a given agent (such as fixed-wing UAV). In fact, most practically oriented work such as in (Beard & McLain 2003, Zhang et al. 2017) use the path planning specific to fixed-wing UAVs, while theoretical work such as in (Guruprasad & Ghose 2011) are specific in nature as far as the type of agents are considered, though the work is more suitable for aerial vehicles.

2. Though the quadcopter UAVs have become very popular and economical recently, there has not been substantial work on quadcopters being used for multi-agent search problems in the literature barring a few exceptions such as in (Engel 2011, Engel et al. 2012, Waharte et al. 2009, Khan et al. 2014). Even when it comes to practical applications of drones, they are remotely triggered than being autonomous. This warrants both theoretical and practically oriented research on use of quadcopters in autonomous multi-agent search, the problem that is addressed in this thesis.

3. Though downward-facing cameras is one of the convenient (in terms of being suitable and economically viable) search sensors for applications where targets are visually detectable, very few work in the literature, such as in (Engel 2011, Engel et al. 2012, Freda & Oriolo 2007, Hu et al. 2014, Zhang et al. 2017), focus on such vision-based search. Most researchers do not focus on a specific sensor, though the problem setting suits such a sensor (for example, work by Guruprasad & Ghose (2011)).

4. When searching a large geographical area with a downward-facing camera, it is natural to expect that the effectiveness of the camera as a search sensor, in terms of image resolution/quality is not uniform across the image frame.

As used in a few literature such as in (Guruprasad & Ghose 2011, Delle Fave et al. 2010), the image quality is expected to be highest at the central pixel (that is, directly below the agent/camera), and degrade monotonically away from it. However, in most work in the literature, both when a camera is used as search sensor, (Zhang et al. 2017), for example, and any other generic search sensor is used, a flat sensor footprint is typically assumed, which may not be suitable in most practical scenarios unless the imaging sensor has sufficiently high-quality image so that even a target located in the corner of the image frame is detectable with the same ease as that directly below the camera.

5. Even when a non-uniform sensor footprint/search effectiveness model is used for a camera, authors such as Guruprasad & Ghose (2011), Delle Fave et al. (2010) use generic and intuitive exponential functions to model the effectiveness of sensor/camera without any justification for the use, except that the function is tunable.

6. Most work in the literature except for a few such as (Guruprasad & Ghose 2011) use a gridded space for path planning. Path planning here is reduced to which is the next cell (or a few next cells in order) to which the agent should move. Such a "plan and act" at each cell strategy for path planning may not be suitable for several applications, apart from possibly generating non-smooth paths. In fact we may observe that some authors provide additional strategies for converting the cell to cell path to a smooth path suitable for the motion of the agents. A path planning at continuous space as proposed by Guruprasad & Ghose (2011) does not suffer from such limitations and is more suitable when a non-flat sensor effectiveness is used.

With these observations in the background, in this work we propose a multi-agent search strategy using quadcopters as search agents, downward-facing cameras as search sensors, and formulate the problem in a continuous space. We assume a non-unform search effectiveness model for the camera for devising a search strategy. We then obtain a suitable search effectiveness model for downward-facing camera using target detection experiments, apart from collecting relevant information form the work outside the multi-agent search literature. In an attempt to move a step closer to implementation of the multi-quadcopter search strategy in reality, we develop a realistic simulation platform.

CHAPTER 3

MULTI-QUADCOPTER SEARCH STRATEGY

In this chapter, we describe the multi-quadcopter search strategy using downward-facing camera proposed in this thesis.

3.1 PROBLEM SETTING

In this section, we describe the problem formulation. Figure 3.1 illustrates the problem setting addressed in this work. We consider a problem of searching a region of interest $Q \in \mathbb{R}^2$, for the presence of targets of interest. Number and location of the targets is unknown. The targets are assumed to be stationary and not an adversary or hazardous. N quadcopters equipped with downward-facing cameras, communication, and other necessary equipment should cooperatively search Q, the search space, and detect all the targets present. The configuration of agents at any given time t is $P(t) = (p_1(t), p_2(t), \dots, p_N(t)) \in Q^N$, with $p_i \neq p_j$, whenever $i \neq j$. Here, $p_i(t)$ is the projection of the position of the i-th agent at

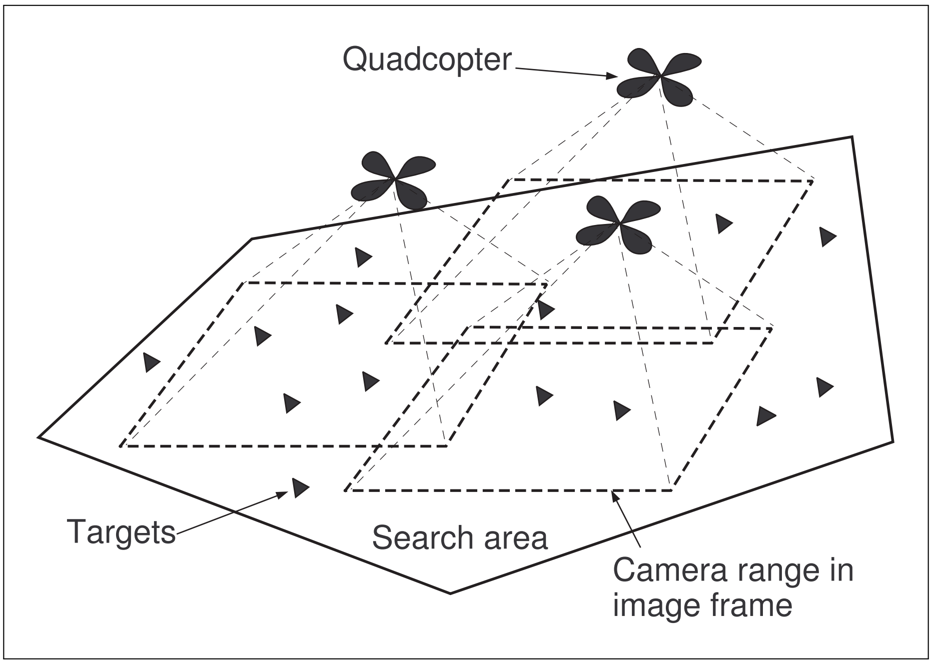


Figure 3.1: Illustration of multiple autonomous quadcopters searching for targets in a search area using downward facing cameras.

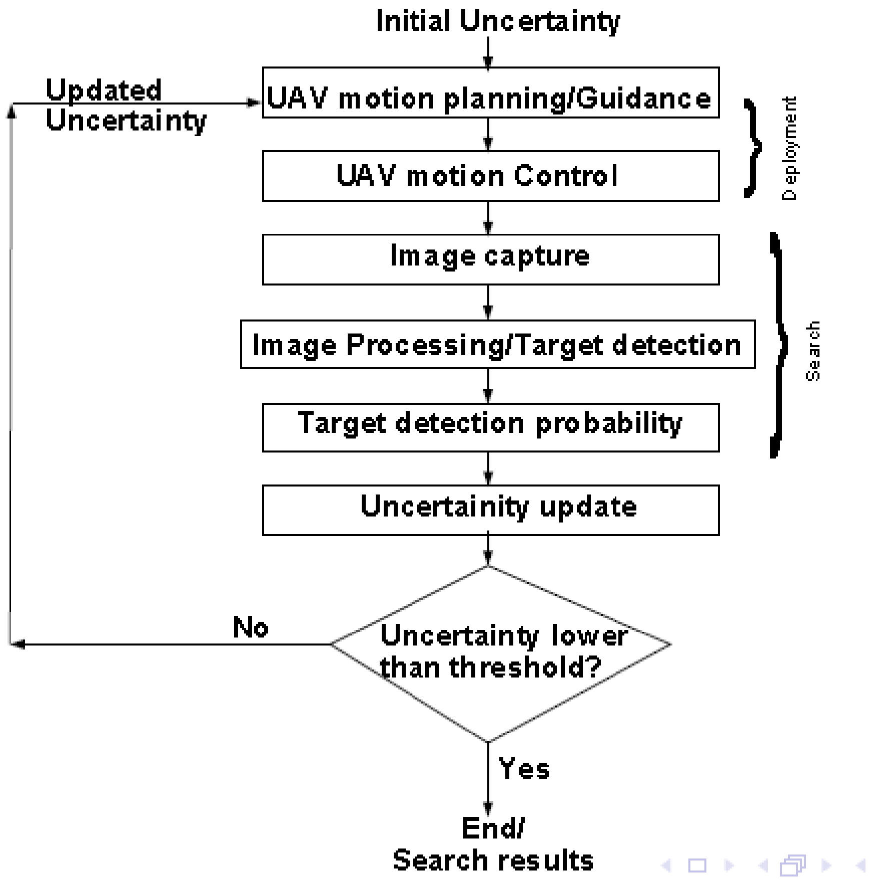


Figure 3.2: Flow chart illustrating typical autonomous multi-UAV search for target detection.

time t on Q. The actual position of the quad-copter includes its altitude along with orientation. While this complete positional and orientational information is required for the quad-copter control, the search problem formulated in this thesis requires only the its projection on Q.

3.2 PROPOSED SEARCH STRATEGY

In this section we discuss the proposed cooperative search strategy. We use downward-facing cameras as search sensors. Figure 3.2 provides flow chart illustrating a typical autonomous multi-UAV search strategy.

The resolution, and hence, the information content of the image, is typically not uniform throughout the image frame, as we will discuss in the next chapter in detail. This non-uniform image quality across the image frame leads to non-uniform search effectiveness as the downward-facing cameras are used for target detection in this work. Thus, we observe that the sensor model that is used in (Guruprasad & Ghose 2011) suits the problem being addressed in this work. As in (Guruprasad & Ghose 2011, Sujit & Ghose 2009), we use $\phi : Q \rightarrow [0, 1]$, to define the uncertainty density distribution representing lack of information. At the end of the search process, the uncertainty density approaches zero, indicating the complete information on the presence (or absence) of targets in each point in Q is available. The Voronoi partition results in natural cooperation amongst the search agents requiring minimal communication between them. Motivated by these observations, we use the problem formulation proposed in (Guruprasad & Ghose 2011) as the basis for the search strategy to be used by multi-quadcopter systems in this work.

Search effectiveness of each downward-facing camera is assumed to be a strictly decreasing function of $\|p_i - q\|$, at any point $q \in Q$. After deployment, the sensors gather information about Q, reducing the uncertainty density according to,

$$\phi_{n+1}(q) = \phi_n(q) \min_i \{\beta(\|p_i - q\|)\} \tag{3.1}$$

where, $\phi_n(q)$ is the uncertainty density at the n-th search step; $\beta : R \mapsto (0, 1)$ is the search effectiveness of the camera, a strictly increasing function of the Euclidean distance from the agent, and acts as a factor of reduction in uncertainty by the sensors. At a given $q \in Q$, only the agent with the smallest $\beta(\|p_i - q\|)$, that is, the agent that can reduce the uncertainty by the largest amount performs the search. This is the equivalent of each agent searching within its Voronoi cell, computed based on P as the node-set.

The deployment of the quadcopters in Q should maximize the search effectiveness, or equivalently, maximize the reduction in uncertainty ϕ in any given iteration. Thus, the following objective function is maximized.

$$\begin{aligned}\mathcal{H}^n(\mathcal{P}) &= \int_Q \Delta\phi_n(q)dq \\ &= \int_Q \max_i\{(\phi_n(q) - \beta(\| p_i - q \|)\phi_n(q))\}dq \\ &= \int_Q(\phi_n(q) - \min_i\{\beta(\| p_i - q \|)\phi_n(q)\})dq \\ &= \sum_i \int_{V_i} \phi_n(q)(1 - \beta(\| p_i - q \|))dq\end{aligned} \tag{3.2}$$

The gradient is given by

$$\begin{aligned}\frac{\partial \mathcal{H}^n}{\delta p_i} &= \int_{V_i} \phi_n(q)\frac{\partial}{\partial p_i}(1 - \beta(r_i))dq \\ &= -\tilde{M}_{V_i}(p_i - \tilde{C}_{V_i})\end{aligned} \tag{3.3}$$

Here $\tilde{M}_{V_i}$ and $\tilde{C}_{V_i}$ are interpreted respectively as the mass and the centroid of V_i with $\tilde{\phi}_n(q) = -\phi_n(q)\frac{\partial f(r_i)}{\partial (r_i)^2}$, as density, $f(\cdot) = 1 - \beta(\cdot)$, and $r_i = \|p_i - q\|$. Note that as f is a monotonically strictly decreasing function, $\frac{\partial f}{\partial({r_i}^2)} < 0$, implying, $\tilde{\phi}(q) \geq 0$. Thus, necessary condition for optimality is $p_i = \tilde{C}_{V_i}$. Hence, the centroidal Voronoi configuration, where the ith agent is located at C_{V_i}, the weighted centroid of V_i, the Voronoi cell containing p_i, based on a $\tilde{\phi}(q) = -2\phi(q)\frac{\partial f}{\partial({r_i}^2)}$, as density, is the optimal configuration maximizing the search effectiveness. Equivalently, the information gain (of the target distribution) is maximized if the search is performed when the agents (and hence the search sensors) are located at the weighted centroid of corresponding Voronoi cells. This forms the basis for a control law based on Lloyd's algorithm.

$$u_i(t) = -k_{prop}(p_i(t) - \tilde{C}_{V_i}(t)) \tag{3.4}$$

Where, k_{prop} is a positive gain. The control law makes the agents (quadcopters) move toward the centroid of their respective Voronoi cells. It has been shown in (Guruprasad & Ghose 2011), if the agents are assumed to be point masses, then the control law (3.4) successfully makes the agents reach the centroidal Voronoi (optimal) configuration, asymptotically. However, in reality, the quadcopters, which are used as agents in this work, have complex nonlinear dynamics, and a point mass assumption is highly unrealistic. In the case of quadcopters, we still use the control law given in Eqn. (3.4), however at the outer loop. An inner-loop or low-level controller is responsible for ensuring that the robot moves toward the respective centroids. Now each quadcopter follows the strategy shown in Algorithm 1.

```
Step 1 Compute Voronoi cell $V_i$ based on $P(t)$.
Step 2 Compute centroid $C_{V_i}$ of $V_i$ using $\phi$ as density.
Step 3 Move toward $C_{V_i}$.
Step 4 If $p_i \approx C_{V_i}$ GOTO Step 5 else GOTO Step 1.
Step 5 Perform search (Update $\phi$)
Step 6 If average uncertainty is less than a preset value GOTO Step 7, else GOTO
       Step 1.
Step 7 END
```

Algorithm 1: "Deploy" and "search" strategy followed by each quadcopter.

3.3 SUMMARY

In this chapter we described the proposed multi-quadcopter search strategy using downward-facing cameras mounted on the quadcopters as search sensors. Here we have assumed that the search effectiveness of the camera is maximum at the center and degrades away from it. Such an assumption leads us to the use of the Voronoi partitioning scheme which results in optimal task partitioning and the centroidal Voronoi configuration as optimal deployment configuration maximizing the uncertainty reduction or equivalently the information gain. In the next chapter we present an experimental setup that we use to obtain the search effectiveness model for a downward-facing camera using target detection probability.

CHAPTER 4

SEARCH EFFECTIVENESS OF A CAMERA

In this chapter, we discuss the camera as a search sensor, define and discuss its spatial search effectiveness, and finally provide an experimental setup to obtain the search effectiveness of the camera using object detection.

4.1 CAMERA AS A SEARCH SENSOR

In this section, based on the literature, we discuss some of the properties of a camera relevant to the search problem. In most conditions, such as search and rescue operations in a natural calamity hit region, the targets such as survivors requiring immediate assistance or the amount of damage to be assessed, are typically visually detectable. A camera is suitable in most similar scenarios. UAVs such as quadcopters are usually equipped with a frontal camera and a downward-facing camera. Figure 4.1 illustrates a single-robot scenario.

In most situations in an aerial search, as illustrated in Figure 4.1, downward-facing cameras are more suitable. In a typical single-UAV or a multi-UAV search problem, the cameras capture the image of a large area. The search effectiveness function of a camera captures the spatial variation of effectiveness of the camera in terms of target detection capability. We define the search effectiveness of a sensor as:

$$f(q) = p(q) \tag{4.1}$$

where, $q \in Q$ is the point of interest in $Q \subset \mathbb{R}^2$, the search area, $p(q)$ is the probability of detection of the target of interest present at q. If we assume that the search sensor is anisotropic in nature then we have:

$$p(q) = f(r) \tag{4.2}$$

where, $r = \|C - q\|$, and $C \in Q$ is a point directly below the sensor.

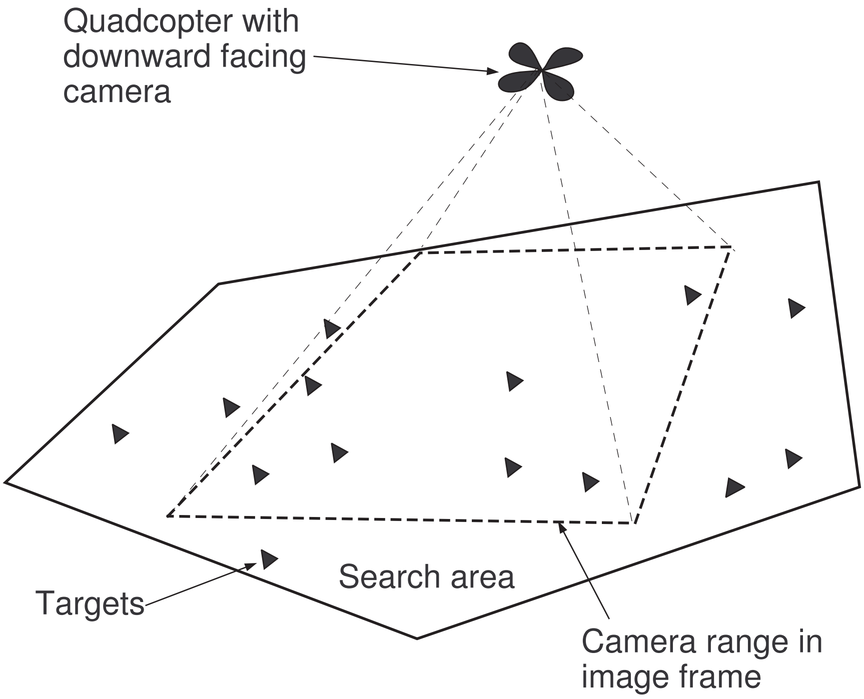

Quadcopter with downward facing camera
Search area
Targets
Camera range in image frame

Figure 4.1: Illustration of a (a) single or (b) multiple autonomous quadcopters searching for targets in a search area using downward-facing cameras.

4.1.1 Search effectiveness models

One of the vital aspects of search is modelling the search process itself. Typically, a metric such as uncertainty density, such as in Sujit & Ghose (2004), is used to model the lack of information on the presence or absence of the target of interest at a point in space. Search is modelled as the process of reducing the uncertainty density by way of information gathering and processing through the search sensor. How the uncertainty reduces upon capturing the image and further processing for target detection over the region covered by the image frame is referred to as the *search effectiveness* (Sujit & Ghose 2004, 2005, Guruprasad & Ghose 2011) model of the given search sensor/camera. Such a model is instrumental in both modelling the search process itself and planning UAV trajectories. Figure 4.2 shows a few typical sensor effectiveness models used in the multi-robot/UAV search or sensor coverage literature.

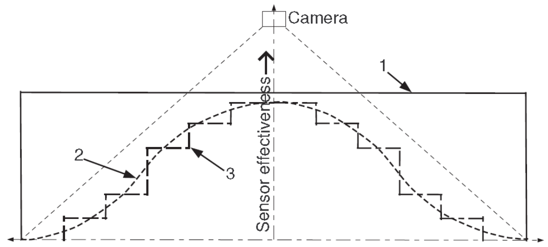


Figure 4.2: Different sensor effectiveness models assumed in the literature.

Successful target detection depends on the image quality. Most works on multi-agent search such as in (Zhang et al. 2017, Beard & McLain 2003) use a flat effectiveness function. The flat sensor effectiveness model marked as '1' in Fig 4.2 implies that the uncertainty density at any point that comes under the camera's field of view (FOV) is reduced uniformly. A special case of this model is when uncertainty at every point that comes under the camera's FOV is reduced to zero. That is, complete information (on the presence or absence of the targets of interest) at all the points covered by the camera's FOV is gathered at a single stretch. Such a model for the camera may be suitable only for situations which provide clear and high contrast images. Either a high resolution camera is used or the targets are distinguishable in the search area. This may not be true in many practical scenarios. Some works in the multi-robotic search literature assume that the sensor's search effectiveness is maximum directly below it and decreasing away from the center, as that marked '2' or '3' in Fig 4.2. Authors such as Sujit & Ghose (2004, 2009), Guruprasad & Ghose (2011), Delle Fave et al. (2010) devise a plan for the agent motion, assuming a sensor effectiveness model marked '2' (Guruprasad & Ghose 2011) or '3' (Sujit & Ghose 2004) in Fig 4.2. In the context of distributed environmental monitoring, authors in Schwager et al. (2011) address a visual sensor coverage problem using multiple cameras mounted on UAVs. Though every point within a camera's FOV is considered covered, the authors use the minimum information per pixel principle as a cost function for the camera placement. Cortes et al. (2004) also use a similar sensor effectiveness function for sensor coverage optimization problem. An exponential function is used in (Guruprasad & Ghose 2011), for a generic search sensor.

A camera has an optical system, a photo-sensor (such as CCD or CMOS), and signal processing systems. The original scene may get degraded in each of these stages. While some of the factors affect the image quality more or less uniformly throughout the image frame, some factors affect the image quality non-uniformly across the frame.

4.1.2 Image resolution

Optical resolution describes the ability of an imaging system to resolve details in the object that is being imaged. While the number of pixels (or Megapixels) as a measure of resolution tells us how many units the image is composed of and the smallest unit of the image, it does not give us a complete idea of the camera's ability to resolve detail in the object being imaged. The optical transfer function (OTF) which describes the spatial variation of the light signal as a function of spatial frequency gives a better measure of resolution.

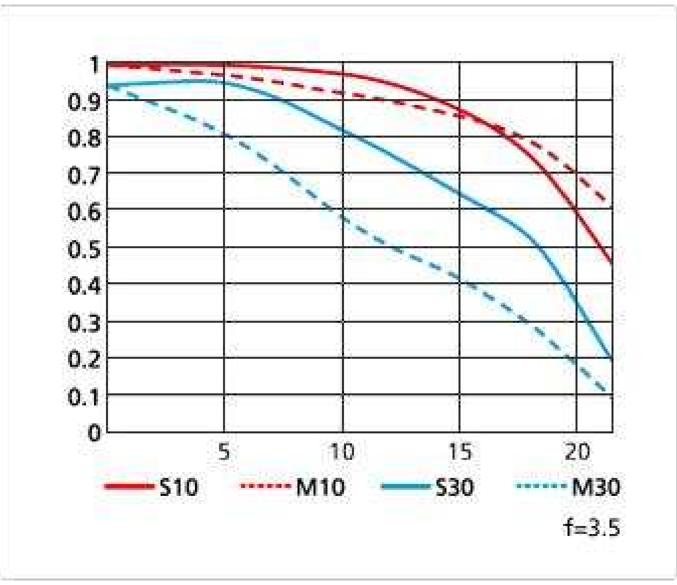


Figure 4.3: Plot of the MTF values with the distance from the central pixel for a Nikon Nikkor camera. Source:http://pixelsandpaintstrokes.com/tools-technique/mtf-charts/

The variation of the camera's resolution is measured along a diagonal direction from the center of the image frame to a corner. Sagittal lines are those which run parallel to this diagonal direction and meridional lines are perpendicular to it. This is illustrated in Fig 4.4.

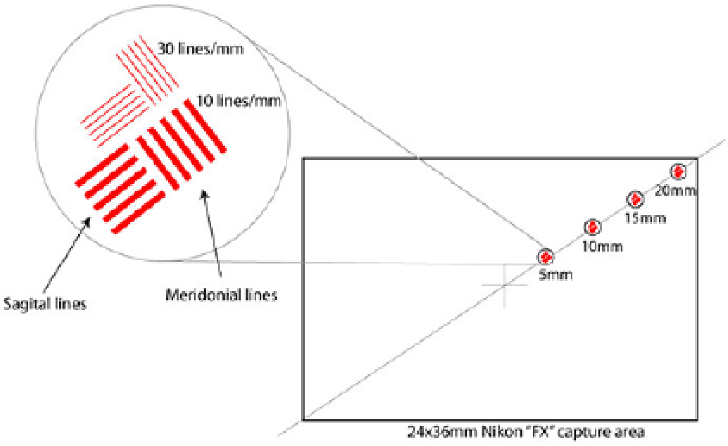


Figure 4.4: Sagittal lines and Meridonial lines.

OTF is given by (Smith 2000),

$$OTF(\xi, \eta) = MTF(\xi, \eta) \times PTF(\xi, \eta) \tag{4.3}$$

Here, ξ and η are spatial frequency in x and y directions respectively, and MTF is the magnitude component and PTF is the phase component. The phase component (PTF) is typically not captured by the sensor. Thus, the important measure with respect to imaging systems is the Modulation Transfer function (MTF). Phase is critically important to adaptive optics and holographic systems. The overall MTF of the system is given by (Sukumar et al. 2008),

$$\begin{aligned} MTF_{sys}(\xi, \eta) &= MTF_{atm}(\xi, \eta) \times MTF_{lens}(\xi, \eta) \\ &\quad \times MTF_{sensor}(\xi, \eta) \times MTF_{trans}(\xi, \eta) \end{aligned} \tag{4.4}$$

Here, the subscripts 'sys', 'atm', 'lens', 'sensor', 'trans', represent the MTF due to atmospheric conditions, the lens or the optical system, the photo-sensor, and finally image processing and image transfer.

Figure 4.3 shows MTF value against the distance from the central pixel. Non-uniformity in sagittal and meridional MTF is due to the camera astigmatism. This leads to anisotropic effectiveness. We may observe that the astigmatism effect is typically negligible.

4.1.3 Quantum efficiency

The number of pixels and the size of the pixel are factors that affect the resolution of a camera. By reducing the pixel size and increasing the number of pixels the resolution can be improved. But reducing the pixel size reduces the quantum efficiency (QE) of the sensor and increases the effect of vignetting. Vignetting, also known as “light fall-off” is common in optics and photography, which in simple terms means darkening of image corners when compared to the center. Light travels through a narrow tunnel in going from the chip surface to a photo-detector in a CMOS sensor. This especially causes problems when light is incident at oblique angles since the narrow tunnel walls cast a shadow on the photodetector which will severely reduce its effective QE. Quantum efficiency (QE) is another metric that provides us a measure of the effectiveness of the camera. While MTF describes the spatial variation of light as a function of spatial frequency, QE describes the ratio of the total photons incident on the system to the number of photons producing charge carriers. Note that MTF is related to the optical system, while QE is related to the photo-detective sensor. Chen (2003) has presented some experimental results showing the variation in the quantum efficiency of a camera with a CMOS sensor at various distances from the central pixel using simulation software. We have used these results to plot the curve in MATLAB. Figure 4.5 shows QE v/s distance from a central pixel for a 6µm pixel size CMOS sensor.

We tabulated the values obtained in Chen (2003) in Table 6.6 and used these values to obtain the following expression as variation of QE with the distance from the central pixel:

$$f = 0.8708 - 0.0714 r_{pix}^2 \tag{4.5}$$

Here, r_{pix} is the distance of the pixel in question from the central pixel. Variation of both MTF and QE across the image frame indicate that the effectiveness of the camera is maximum at the center and reduces with increase in distance from the

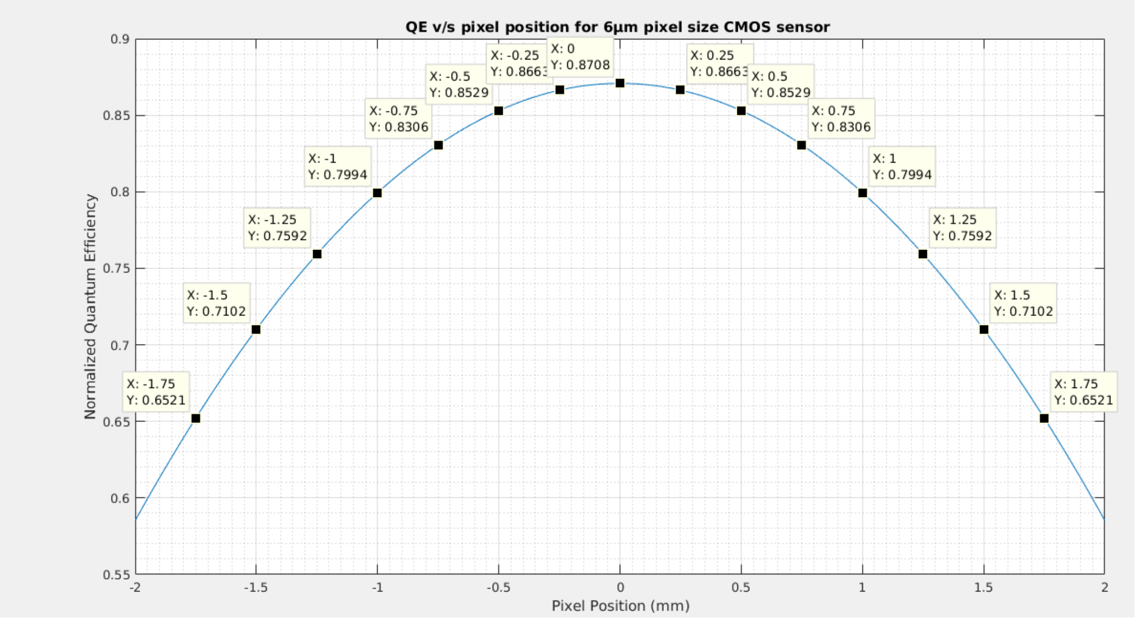


Figure 4.5: QE v/s pixel position for 6m pixel size CMOS sensor.

central pixel, in terms of image quality.

Remark 2 *The purpose of discussion on the Quantum Efficiency (and MTF) here is to understand the variation of image quality across the image frame that affects the search effectiveness of the camera being discussed in chapter, in terms of the target detection probability. This lays the foundation for the experimental investigation provided in the following section.*

4.2 EXPERIMENTAL SETUP

In this section we discuss the experimental technique used in this work to obtain the search effectiveness model of a downward-facing camera. A schematic representation of the experimental setup is shown in Figure 4.6. A downward-facing camera is attached at a height of h above the floor. Target shapes or markers are kept on the floor at various distances from the center of the camera. Several snapshots are of the targets are captured. The targets are identified by processing captured images. Corresponding to each target location, the number of successful detection after image processing are tabulated. Thus, we obtain the

Sl. No.	Pixel Position (mm)	Normalized Q.E.
1	-1.75	0.6521
2	-1.5	0.7102
3	-1.25	0.7592
4	-1	0.7994
5	-0.75	0.8306
6	-0.5	0.8529
7	-0.25	0.8663
8	0	**0.8708**
9	0.25	0.8663
10	0.5	0.8529
11	0.75	0.8306
12	1.0	0.7994
13	1.25	0.7592
14	1.5	0.7102
15	1.75	0.6521

Table 4.1: The Quantum Efficiency at each point is considered to be the certainty of finding an object at that point in the image.

probability of target detection at different locations on the image frame. In this work, we assume the imaging sensor is isotropic and hence we consider only the distance of the target location from the central pixel in the image frame. Target locations are marked $-7, \ldots, 0, \ldots 7$, with 0 indicating that the target is directly below the camera.

4.2.1 Imaging sensor

We used a simple web camera to conduct the experiments. The USB powered camera has a CMOS Sensor of 1/6 inch in size, a maximum resolution of 1600 × 1200 pixels, frame rate of 30fps, fixed focal length with the minimum focusing distance of 0.05m. For our experiments we have used even a lower resolution of 640 × 480 pixels. We have used a basic camera with low resolution as in reality even though a high-resolution advanced camera may be used, the height at which the camera captures the image is typically much higher and also the detection of the target is much more complex than detecting simple markers on a flat floor.

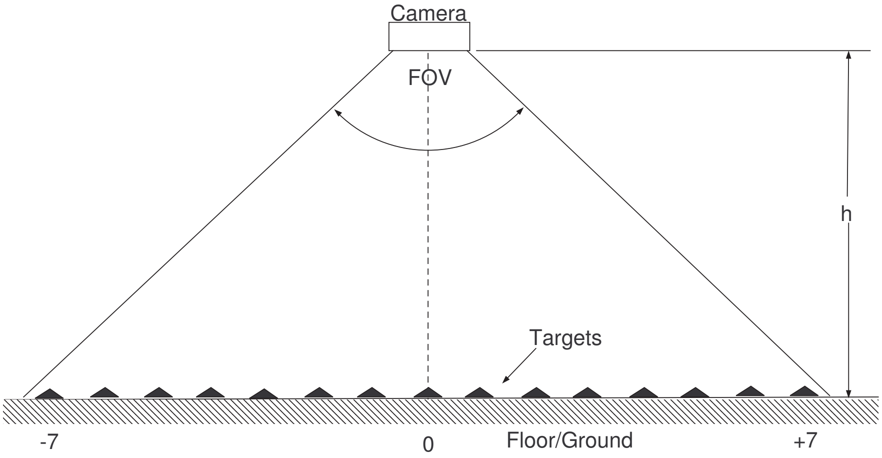


Figure 4.6: Schematic representation of the experimental setup.

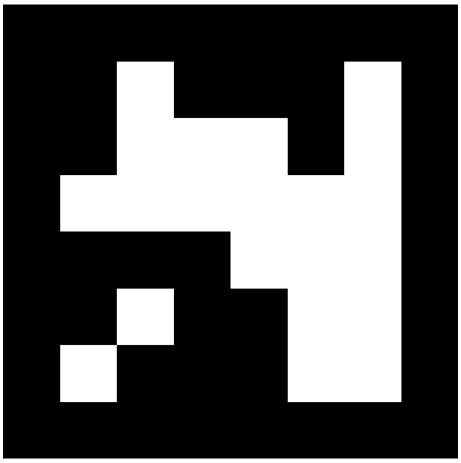


Figure 4.7: An ArUco marker.

4.2.2 Targets

We used ArUco markers and triangular shapes as targets. The ArUco module is based on the ArUco library, a popular library for detection of square fiducial markers (Munoz-Salinas 2013, Muoz-Salinas & Medina-Carnicer 2018). An AurUco marker used in our experiments is shown in Figure 4.7.

These Markers are binary square fiducial markers which can easily and uniquely be identified at a distance. By its design, it is easy for detection even in presence of noise. Marker detection process involves Thresholding, Contour filtering, Bits extraction, Marker Identification, and Corner Refinement. These markers may be used to model target detection in real scenario, where the detection probability is high. Detection of polygonal objects such as triangle based on corner detection can also be used to model the target detection. Unlike the ArUco markers, detection probability of triangular objects as targets are more prone to noise. In this work we used both the ArUco Markers and triangular shaped objects as targets.

4.3 SUMMARY

In this chapter we discussed the spatial variation of the image quality both in terms of optical resolution and digital quality. We have observed that the image quality is higher at the central pixel and degrades away from it. Such a scenario leads us to a non-uniform search effectiveness of the camera. We also presented an experimental setup proposed in this work to obtain a sensor effectiveness model for a downward-facing camera using the target detection probability. We present the results in Chapter 6 with a detailed discussion. In the next chapter we present a realistic hybrid centralized-decentralized simulation platform for the proposed multi-quadcopter search strategy developed using ROS and Matlab.

CHAPTER 5

A REALISTIC SIMULATION PLATFORM FOR THE PROPOSED MULTI-QUADCOPTER SEARCH

In this chapter, we describe a realistic simulation platform developed in this work for the proposed search strategy using multiple quadcopters carrying downward-facing cameras as the search sensors.

5.1 HYBRID ARCHITECTURE FOR THE PROPOSED STRATEGY

The proposed multi-quadcopter search strategy has several components. Primary components are i) the spatial task partitioning, ii) optimal deployment configuration (CVC), iii) control law for deployment, and finally iv) search leading to uncertainty reduction. In this section we first provide a discussion on spatial distribution property of the individual components and the search strategy as a whole. We make the following observation:

1. We use Voronoi partitioning scheme to partition the search space Q into V_i, the Voronoi cells based on the current configuration (that is, position of search agents) P. The task of searching Q using N quadcopters is now partitioned into N single quadcopter search tasks, where the task of performing search within a Voronoi cell is allotted to the corresponding quadcopter. Though each quadcopter searches within the corresponding Voronoi cell, the Voronoi cell itself depends on the location of the neighboring quadcopters (nodes). That is, V_i does not depend only on p_i, the location of the corresponding quadcopter, but the position of all the quadcopters which are its neighbors in the *Delaunay graph* $\mathcal{G}_D$, where two agents i and j are considered neighbors if and only if $V_i \cap V_j \neq \emptyset$. Also, if the ith quadcopter has information about the location of other quadcopters which are its neighbors within $\mathcal{G}_D$, then in principle, it can compute the V_i, the corresponding cell on its own (see (Guruprasad & Dasgupta 2012*b*,*a*) and reference therein).

Thus, in the perspective of spatial task partitioning, the multi-quadcopter search strategy is spatially distributed within the Delaunay graph $\mathcal{G}_D$.

2. Now consider the optimal deployment configuration, that is, the centroidal Voronoi configuration. Here too, though the solution for the optimal deployment for each quadcopter is the centroid of the corresponding Voronoi cell, as we have discussed before, the Voronoi cell itself is not independent of other quadcopter's locations. That is, C_{V_i} is not just the function of p_i but P (to be more specific, $N_D(P) \subset P$, the set of neighbors of p_i in $\mathcal{G}_D$). In this sense, the optimal deployment configuration is spatially distributed in $\mathcal{G}_D$.

3. The third component of optimal deployment is the control law (Eqn. (3.4)) that each of the quadcopter's uses to achieve the centroidal Voronoi configuration. Here, the gradient that is used in the control law, and the control law itself may also be observed to be spatially distributed in $\mathcal{G}_D$.

4. The final component is the search task, which is gathering information within the corresponding Voronoi cells by the quadcopters using the downward-facing cameras and hence, the reduction in the uncertainty density. Here, as we discussed earlier, the multi-quadcopter search task is converted into multiple single quadcopter search task by the spatial task partitioning achieved by Voronoi partitioning. Here the task is decentralized (as opposed to distributed nature of above three components) in nature. However, once a step of search is completed, during the next deployment process, each quadcopter needs to have access to the updated uncertainty density distribution (in order to compute the centroid) within the corresponding new Voronoi cell (as the positions of the agents change due to their motion during the deployment process). This may be achieved in two ways: First, all the agents (quadcopters) communicate the updated uncertainty (and the target probability distribution map, not addressed in this work) to a central server/information provider. In this scenario, a

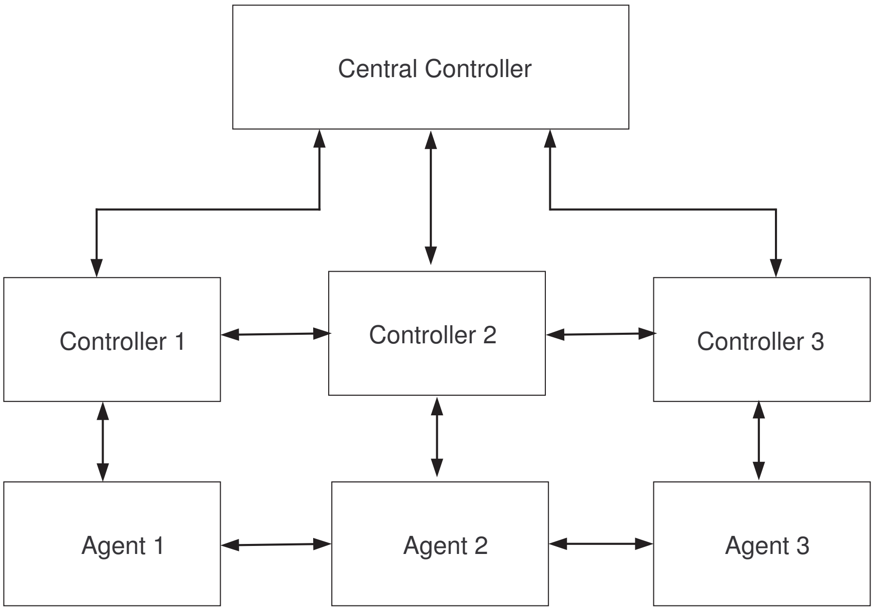


Figure 5.1: A hybrid centralized-distributed multi-quadcopter search architecture.

centralized architecture is used for information storage/retrieval. Second, the quadcopters may use (multi-hop) distributed communication to exchange the updated uncertainty density distribution. This scenario results in a spatially distributed system. However, in reality, the first scenario where a central server is used to store the results of the search (in form of uncertainty density distribution and target detection probability) is more useful. Any search mission during a natural calamity would have a central monitoring station which may act as such a server.

The first three components of the multi-quadcopter search strategy proposed in this work using a distributed architecture, and the last component (that is, search) using a centralized architecture for convenience (note that the last component too can use a distributed architecture in principle) results in a hybrid centralized-distributed system. Thus, though the proposed multi-quadcopter search strategy is amenable for a completely distributed implementation, in a practical scenario, it makes sense to implement it in a hybrid centralized-distributed architecture, where a central server may be used to store the updated uncertainty density (and the target detection probability distribution) and provide this information to quadcopter on demand (to compute the centroids and also update the density after performing search). The computation of the Voronoi cells, their centroids (using the uncertainty density provided by the central server), and hence the control law may be computed by individual controllers using only the information about the location of the neighboring quadcopters. Locational information may be obtained by distributed communication among the quadcopters. In this work, we assume such a hybrid architecture for the proposed search strategy, as illustrated in Figure 5.1.

However, when a central server is used to store the uncertainty density, it makes sense to use its services to compute the Voronoi cells and their centroids. Once each quadcopter controller has information about the centroid of the corresponding Voronoi cell, it may compute the control law independently. Thus, we may use a centralized architecture for computing the spatial partitioning (Voronoi cells) and the optimal deployment configuration (the centroids), and then use decentralized architecture for computing the control law to achieve optimal deployment (CVC) and to perform the search task (updating uncertainty density and the target probability density). Such a hybrid centralized-decentralized architecture is illustrated in Figure 5.2. In this work, we use this hybrid architecture for the implementation of the proposed search strategy within a simulation environment.

Remark 3 *Theoretically, the performance of the proposed multi-quadcopter search strategy does not depend on the architecture (distributed, hybrid centralized-distributed, or hybrid centralized-distributed) in which it is implemented. This is true in reality as long as communication is perfect and communication delay is negligible. In fact, any distributed (or decentralized) system can be implemented in a centralized architecture (converse may not be true). Note also that a distributed system such as the multiple quadcopters performing cooperative search addressed in this work may not be controlled using a purely decentralized architecture as there is no provision for accounting for*

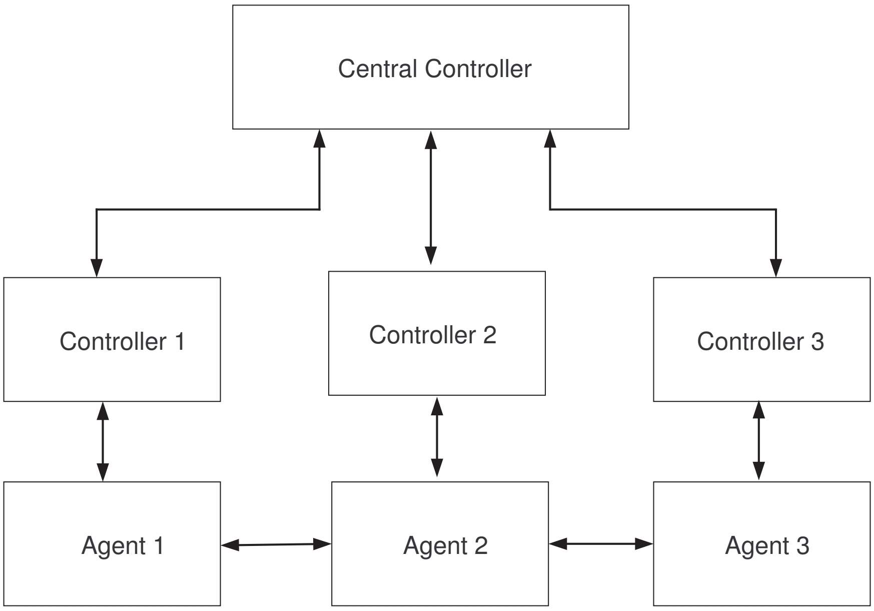


Figure 5.2: A hybrid centralized-decentralized multi-quadcopter search architecture.

mutual interactions (influence) between the agents. It requires either a purely distributed architecture, or a centralized architecture, or a hybrid of centralized and distributed (or decentralized) architecture.

In this work, we do not address the issues related to the communication between the agents and/or the controllers, though they are important in a practical scenario, instead assume a perfect and instantaneous communication. The focus of this work is on the implementation of the proposed search strategy with *realistic agent (quadcopter) dynamics* in a simulation platform, which is a step closer to implementation of the search strategy on physical quadcopters.

5.2 SIMULATION PLATFORM FOR THE PROPOSED SEARCH STRATEGY

In this section, we discuss the platform developed for realistic simulation of the proposed multi-quadcopter search strategy using MATLAB and ROS/Gazebo platforms. We use a hybrid centralized-decentralized architecture illustrated in Figure 5.2 Block diagram illustrating the simulation platform developed as

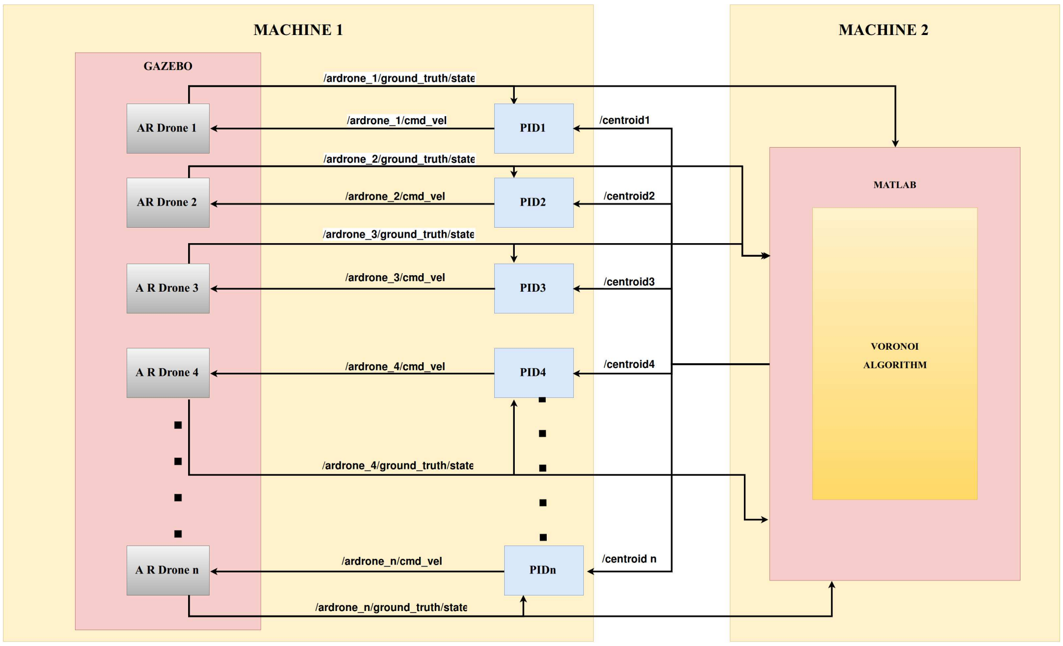


Figure 5.3: A block diagram illustrating the architecture of the simulation platform for multi-quadcopter search strategy using centroidal Voronoi configuration, in two machines.

shown in Figure 5.3. A computer labeled 'Machine 1' in Figure 5.3 simulates a decentralized control of a multi-quadcopter system in ROS/Gazebo environment, while a computer labeled 'Machine 2' operates in a centralized manner, where a single Matlab program maintains the updated uncertainty density form all the quadcopters, and computes the Voronoi cells along with their centroids.

5.2.1 ROS/Gazebo Simulator

The Gazebo is a 3D simulator supported in ROS for realistic simulation of robots. Models of the environment and the robot is created within Gazebo. A library of pre-built models of many robots and world (environment) is available, apart from the option for creating a model from scratch. The models contain both visual information and the physical information such as, inertia, colliding surface, gravity, etc., making the simulation closer to reality. The models created may also be used to control a physical robot, apart from the simulation.

5.2.2 Central controller using Matlab

The central control used is implemented in Matlab[1]. The Matlab program running in 'Machine 2' communicates with the ROS environment in 'Machine 1' to obtain the current position and orientations of the quadcopters, and the current uncertainty density updated by each of the quadcopters. Voronoi cells are computed based on the current configuration of the multi-quadcopter system, which is read (subscribed) from the topics */ardrone_i/ground_truth/state*. Their centroids are also computed based on the current uncertainty density distribution. Voronoi cell, its centroid, and the uncertainty density are communicated to each of the quadcopters through the ROS environment running in 'Machine 1'. While a topic *centroid_i* is used to communicate the centroid of the ith cell, the uncertainty density is set as a global variable. The two machines communicate over Wi-Fi[2] network using 'topics' in ROS.

5.2.3 Multi-quadcopter system in ROS

The decentralized control of multiple quadcopters is implemented in 'Machine 1' in ROS environment. The communication between the quadcopters and central controller 'Machine 2' is handled by ROS in the form of topics.

5.2.4 ARDrone quadcoptor control

An ARDrone is one of the most popular and widely used quadcopters both by researchers and hobbyists. The kinematic and dynamic model of ARDrone as a URDF Unified (Robot Description Format) description, which includes geometric structure, mass, inertia, gravity, etc. is used by Gazebo for simulation. Thus, it may be noted that the dynamics of the quadcopters discussed in Chapter 1 are

---

[1] In place of Matlab we may use any other programming environments such as Python.

[2] The communication between the central controller ('machine 2') and the quadcopter and its controllers within the ROS environment ('machine 1') is over WIFI network when 'machine 1' and 'machine 2' run on two different computers and is through the ROS topics when both machines run on a single computer. We have tested both these methods.

utilized in simulation using the Gazebo environment, unlike in optimal deployment formulation provided such as in (Cortes et al. 2004, Guruprasad & Ghose 2011), where a simple first-order dynamics are assumed in establishing convergence of agent trajectories using simple proportional control law.

A transform(TF) (Foote 2013) library created in ROS keeps track of the quadcopter state (position and orientation) and relative position and orientation of various components including the camera. A package known as *ardrone_autonomy*, developed by Mani Monajjemi and other contributors (Autonomy Laboratory, Simon Fraser University), is used to control the ARDrone in simulation. The same package can also be used to control a physical ARDrone in the real world. This package creates a ROS system for acquiring data from the sensors of the drone. Motion control of the ARDrone can be achieved by publishing a topic called *cmd_vel*, which is equivalent to providing the desired velocity to the quadcopter. In our case, the target point for the quadcopter is the Voronoi cell centroid, which is provided by the central controller 'Machine 2' (Matlab program). To move the quadcopter toward the centroid, we use a simple PID control law. A quadcopter control is achieved by following the algorithm shown in Algorithm 2. Note that the Algorithm 2 performs most steps in the complete search algorithm 1, except for computation of Voronoi cell (partition) and the cell centroids, which is carried out by the

```
Step 1 Publish current state.
Step 2 Obtain Voronoi cell and centroid from central controller (MATLAB).
Step 3 Move toward centroid.
Step 4 If sufficiently close to centroid GOTO Step 5; Else GOTO Step 1.
Step 5 Update uncertainty density and publish the uncertainty density.
Step 6 If average uncertainty density is below preset value GOTO Step 7; Else
       GOTO Step 1.
Step 7 END.
```

Algorithm 2: Algorithm for each ARDrone quadcopter.

central controller.

Remark 4 *The problem formulation and the search strategies presented in this thesis do not use the orientational information as we assume the cameras to be isotropic. However, this information is used for controlling the individual quadcopters (both in simulation or hardware). The controllers use this information to arrest the yaw motion of the quadcopters.*

5.2.5 Multi-ARDrone system control:

We use a TF (transform) tree in the place of a simple TF to handle multiple quadcopters. Each quadcopter will have a namespace such as *ardrone_1*. Now multiple instances of the topics (such as the state of a quadcopter) are created distinguished using the namespace. The states of the quadcopters and the corresponding controllers are now handled by these distinguishable topics within the ROS. For example topics */ardrone_1/ground_truth/state)*, */ardrone_1/cmd_vel*, and *ardrone_1/bottom/image_raw*, relate to state (position and orientation), velocity command input, and raw image data from the downward-facing (bottom) camera, corresponding to the 'ARDrone 1'. Similar topics are used for all the ARDrones distinguishing them by the namespace used. These topics are used for communication between i) each quadcopter controller and the corresponding quadcopter, and ii) each quadcopter/quadcopter controller and the central controller. Though the state of the quadcopter is 6 dimensional vector, only the values of P_x and P_y, the x, and y positions are used here, as we want the quadcopters to always fly horizontally at a fixed altitude. Each of the quadcopters follows the Algorithm 2. With this, we achieve a truly hybrid centralized-decentralized control within the simulation environment.

5.2.6 Control of a quadcopter in Gazebo

The MATLAB node calculates the Voronoi partitions and the centroids for the corresponding cells. The ARDrone quadcopter requires a controller to move it towards a specified position. The centroids of the corresponding Voronoi cell is published by the MATLAB node into the *centroid_i* topics. A controller subscribes to the *centroid_i* values and publishes the required velocity to the *cmd vel* of the corresponding to the ARDrone. A positive value of the proportional control value drives the AR drone towards the corresponding centroid. The position values given by the ARDrone quadcopter via */ground truth/state* topic are conveyed to the controller. The velocity of the quadcopter is represented as tilt angles.

Remark 5 *Simulation of quadcopters' motion as part of 'optimal deployment' into the CVC using ROS/Gazebo environment has mainly two advantages:*

- *First, the simulation carried out is more realistic in the sense that the dynamics of the quadcopters are taken into account, unlike the simulations carried out using Matlab (or similar environments) using a point mass assumption for the search agents (quadcopters) such as those carried out in Guruprasad & Ghose (2011, 2013). The claim that the search agents can be successfully deployed into CVC in Guruprasad & Ghose (2011) is valid only the search agents are assumed to be point masses.*
- *Second, the programs within ROS/Gazebo environments that are used to control the quadcopters in simulation, can in principle be used directly on the quadcopters.*

We claim that the simulation platform is 'realistic' in this sense.

5.3 SUMMARY

In this chapter, we presented a realistic platform developed for simulation of the proposed multi-quadcopter search strategy. The platform developed is realistic in two senses: First, the actual dynamics of the quadcopters along with a realistic world model is considered in the simulation; Second, the programs used to control the quadcopters within the simulation environments maybe used in principle to control the physical quadcopter. In the next chapter we present detailed results of experimental and simulation carried out along with a discussion on the same.

CHAPTER 6

RESULTS AND DISCUSSIONS

In this chapter, we present detailed results of experiments and simulation carried out along with a detailed discussion on the same. First, we present the results of the experiments carried out to obtain the search effectiveness model of the downward-facing camera, which we use in the proposed multi-quadcopter search strategy. Next we present a representative result of the simulation experiment carried out using the realistic ROS/Matlab simulation platform, both to demonstrate the simulation platform itself and the proposed search strategy. Finally, we provide a detailed account of simulation experiments carried out to evaluate the effect of the number of search quadcopters and the camera effectiveness parameters on the performance of the proposed multi-quadcopter search strategy, using the simulation platform developed in this work.

6.1 CAMERA EFFECTIVENESS

In this section we provide the results of the target detection experiments to obtain the search effectiveness model of the camera for target detection. First we provide results of experiments using ArUco markers and then we present those using triangular shaped targets.

6.1.1 Experiments with markers

Now we present the results of experiments conducted using ArUco markers. We used 4×4cm and 5×5cm ArUco markers for the experiments. With the plain floor in the background, the images captured were nearly noiseless. We have added noise in the image space to simulate a mosaic/noisy background to simulate such a scenario in reality.

Table 6.1 shows the target detection probability of 4×4 ArUco markers based on their position relative to the central pixel in the image frame as illustrated in Figure 4.6. Here, The total number of detection attempts is 2000, the second

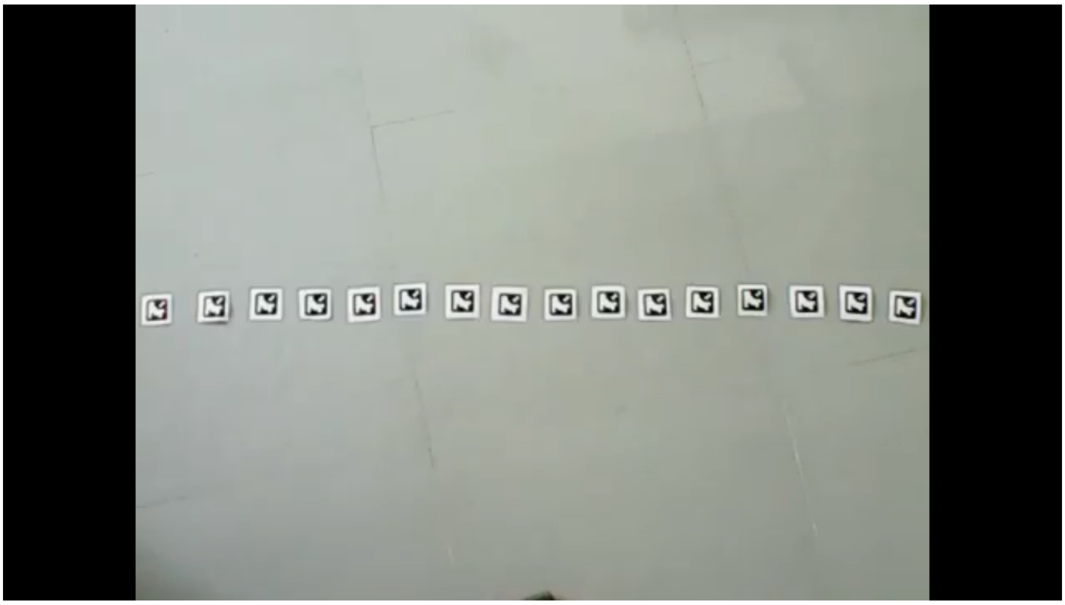


Figure 6.1: 4x4cm ArUco markers target detection experiment setup

column provides the number of successful target detection, and the third column provides the target detection probability. The position 0 represents a point directly below the camera. Figure 6.2 shows the plot of target detection probability with the distance from the central pixel, along with the best fit exponential curve. The equation used for exponential function is:

$$f(x) = k \exp(-\alpha r^2) \tag{6.1}$$

where, r is the distance from the central pixel, $f(\cdot)$ is the probability of the target detection, and k and α are the parameters. For the data shown in Table 6.1, the values of exponential function parameters obtained were $k = 0.5124 \quad (0.4435, 0.5812)$ and $\alpha = 0.07242 \quad (0.04988, 0.09495)$, with 95% confidence bounds. Goodness of the fit metrics as given by Matlab are: **SSE** = 0.0409; **R-square** = 0.9225; **Adj R-Sq** = 0.9166; **RMSE** = 0.0561. Here, **SSE** stands for the Sum of Squares due to Error, **R-square** is is the square of the correlation between the response values and the predicted response values, **Adj R-Sq** is the **R-square** adjusts it based on the residual degrees of freedom, **RMSE** is the Root Mean Square Error.

Location	#of detection	Detection Probability
0	1487	0.5199
1	1217	0.4255
2	1180	0.4126
3	1045	0.3654
4	233	0.0815
5	152	0.0531
6	131	0.0458
7	47	0.0164

Table 6.1: Probability of detection with location of target using 4 × 4 ARuco markers.

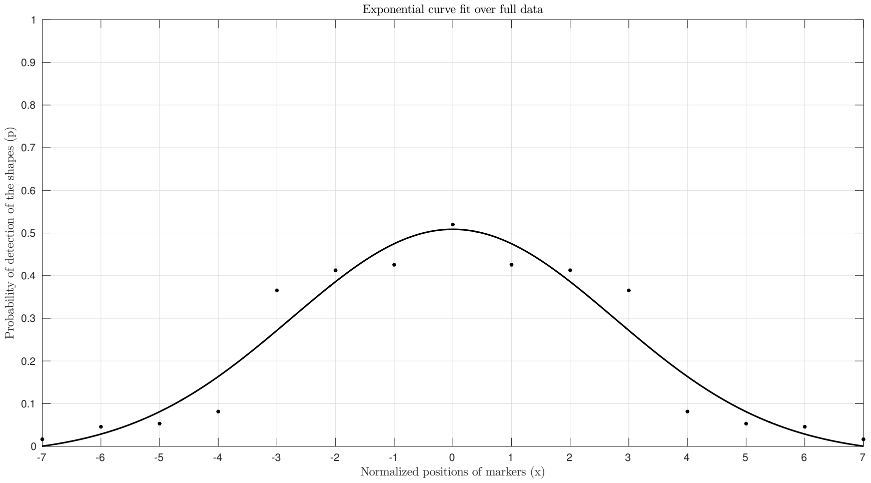


Figure 6.2: Exponential curve fit over entire data corresponding to Table 6.1.

Now we present the experimental results using 4 × 4cm Aruco markers and added salt and pepper noise (salt = 0.9 and pepper = 0.1). Table 6.2 shows the target detection probability with the location of the target. Figure 6.4 shows the plot of target detection probability with the distance from the central pixel, along with the best fit exponential curve. Corresponding curve fitness metrics are: **SSE**= 9.7061; **R-square** = 0.9317; **Adj R-Sq** = 0.9264; **RMSE** = 0.0086

Now we provide the results using Aruco marker of size 5 × 5. Table 6.3 shows the target detection probability with the location of the target. Figure 6.6 shows the plot of target detection probability with the distance from the central

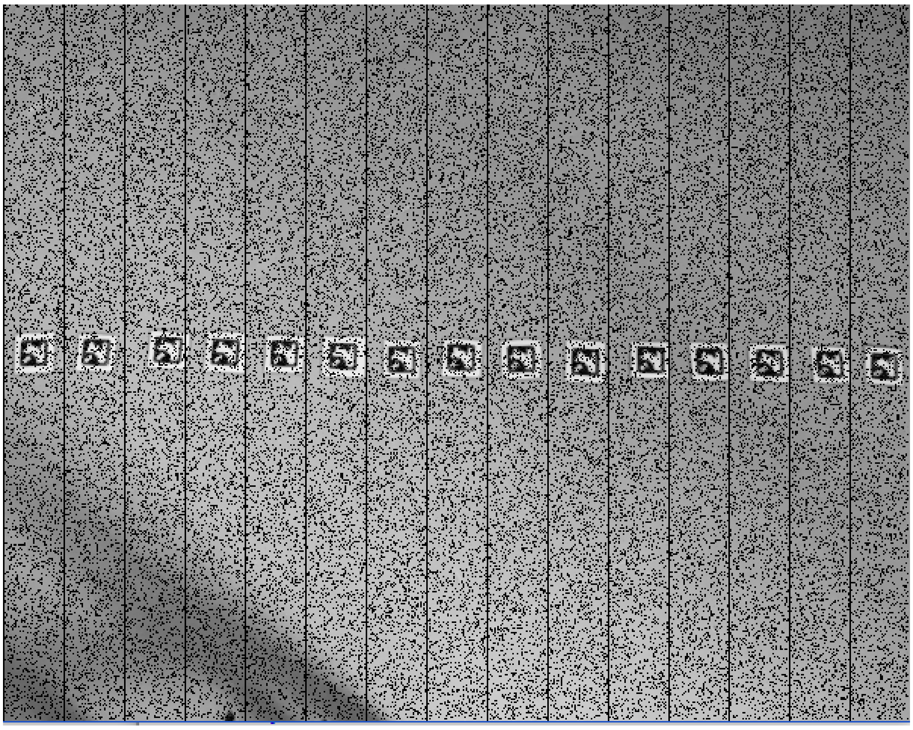


Figure 6.3: 4x4cm ArUco markers target detection experiment with salt and pepper noise
6.2.

Location	#of detection	Detection Probability
0	493	0.1724
1	391	0.1367
2	374	0.1308
3	359	0.1255
4	291	0.1017
5	269	0.0941
6	213	0.0745
7	180	0.0629

Table 6.2: Probability of detection with location of target using 4 × 4 AuRuco markers with added salt and pepper noise.

pixel, along with the best fit exponential curve. The coefficients obtained are $k = 0.6386$ and $\alpha = 0.07276$. Corresponding curve fitness metrics are: **SSE** = 0.01551; **R-square** = 0.9792; **Adj R-Sq** = 0.9776; **RMSE** = 0.03454.

Finally, we present the experimental results using 5 × 5cm ArUco markers and added salt and pepper noise (salt = 0.9 and pepper = 0.1). Table 6.4 shows

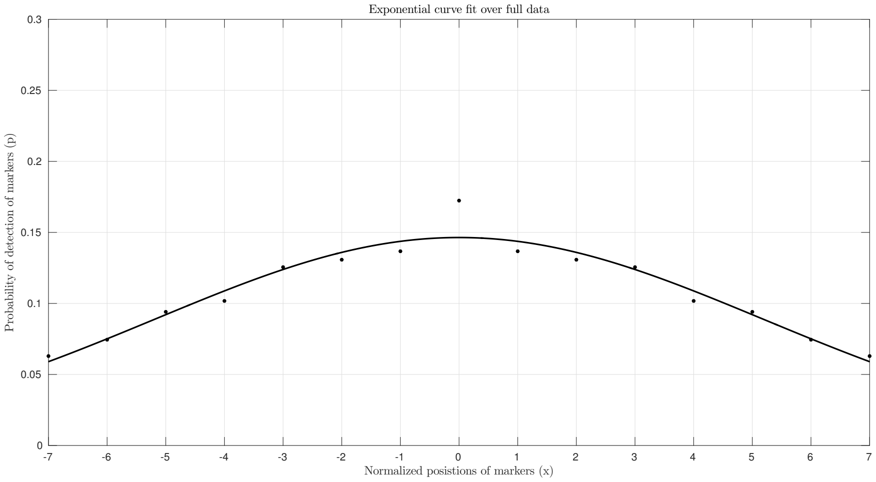


Figure 6.4: Exponential curve fit over the data given in Table 6.2.

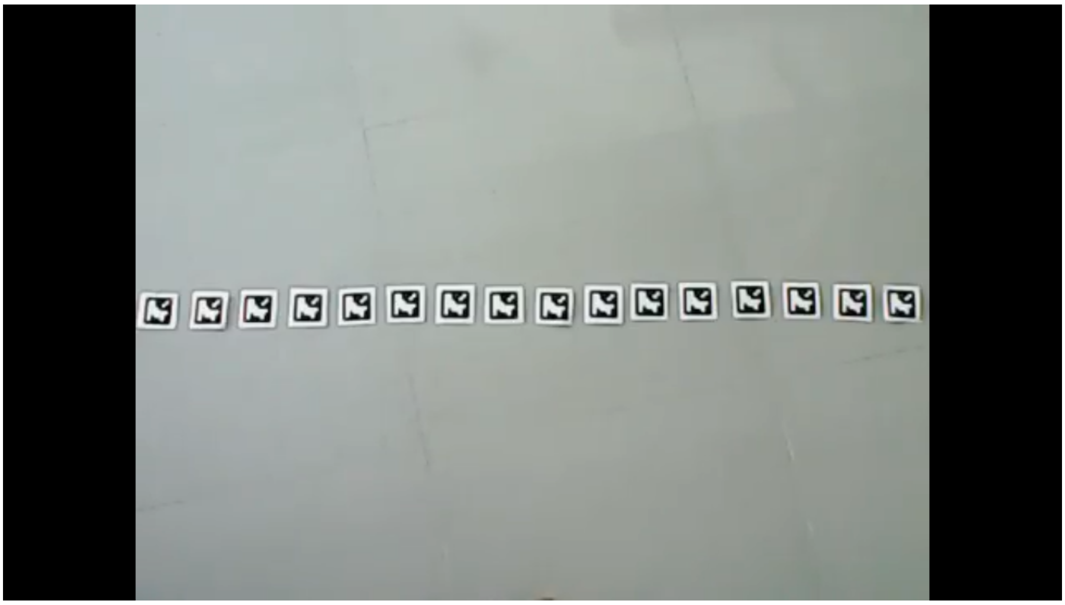


Figure 6.5: 5x5cm ArUco markers target detection experiment setup.

the target detection probability with the location of the target.

Figure 6.8 shows the plot of target detection probability with the distance from the central pixel, along with the best fit exponential curve. Corresponding curve fitness metrics are: **SSE** = $1.2580E-5$; **R-square** = 0.9839; **Adj R-Sq** = 0.9827; **RMSE** = $9.8374E-4$.

Now if we compare Tables 6.1 and 6.2 (or Figures 6.2 and 6.4) corresponding to 4 × 4cm markers, we observe that with added noise the target

Location	#of detection	Detection Probability
0	1997	0.6983
1	1613	0.5640
2	1403	0.4906
3	802	0.2804
4	703	0.2458
5	313	0.1094
6	111	0.0388
7	37	0.0129

Table 6.3: Probability of detection with location of target using 5 × 5 AuRuco markers.

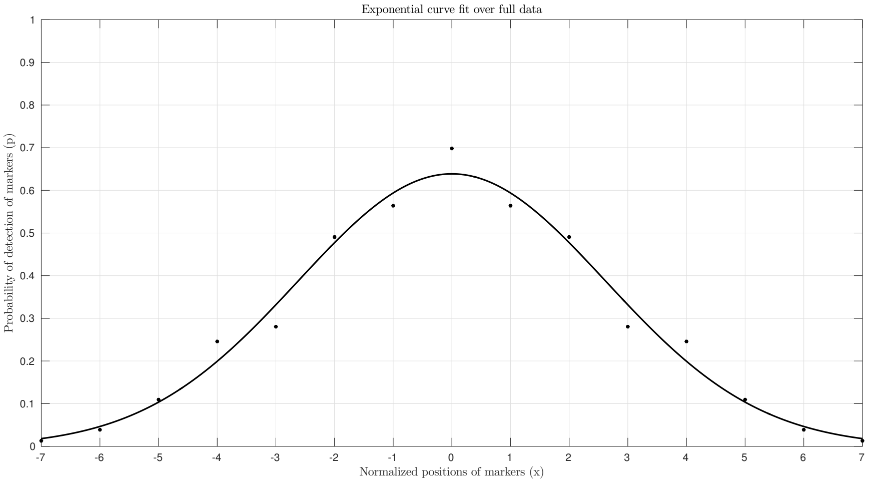


Figure 6.6: Exponential curve fit over the data shown in Table 6.3.

detection probability curve gets flatter and the maximum probability reduces from 0.5199 to 0.1724. A similar observation may be made in the case of 5 × 5cm markers. Further we may also observe with decrease in marker size from 5 × 5cm to 4 × 4, apart from decrease in the maximum detection probability from 0.6983 to 0.5199, the target detection probability distribution curve gets flatter. However, effect of noise on 5 × 5cm marker is more prominent than that on the 4 × 4cm markers in terms of reduction in maximum detection probability and flatter target detection probability distribution curve.

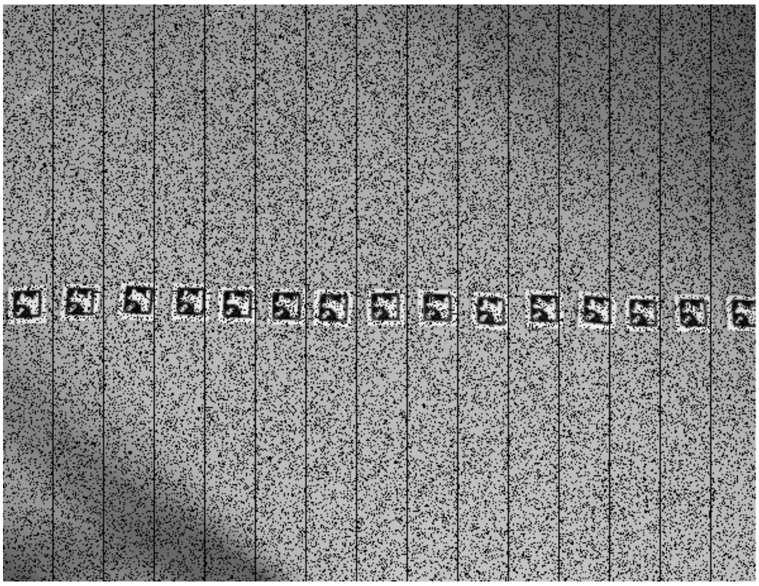


Figure 6.7: 5x5 ArUco marker detection with salt and pepper noise

Location	#of detection	Detection Probability
0	123	0.0430
1	112	0.0392
2	107	0.0374
3	98	0.0343
4	92	0.0322
5	81	0.0283
6	69	0.0241
7	54	0.0189

Table 6.4: Probability of detection with location of target using 5×5 AuRuco markers with salt and pepper noise.

6.1.2 Experiments using triangle shaped targets

Now we provide the results using triangular shapes as targets. Table 6.5 shows the target detection probability with the location of the target,corresponing to a scenarion shown in Figure . 6.10 shows the plot of target detection probability with the distance from the central pixel, along with the best fit exponential curve. The corresponding curve fitness metrics are: **SSE** = 0.0011; **R-square** = 0.9880; **Adj R-Sq** = 0.9870; **RMSE**= 0.0095.

In comparison with the results obtained with the ArUco markers, we may

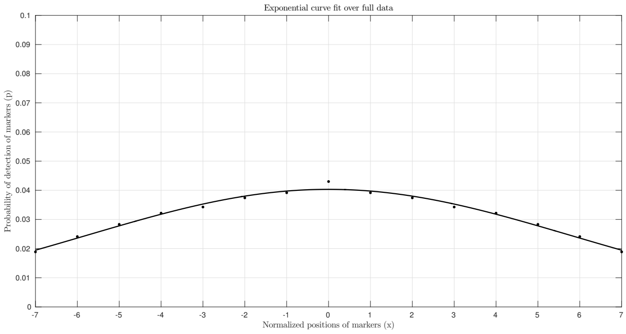


Figure 6.8: Exponential curve fit over over the data shown in Table 6.4.

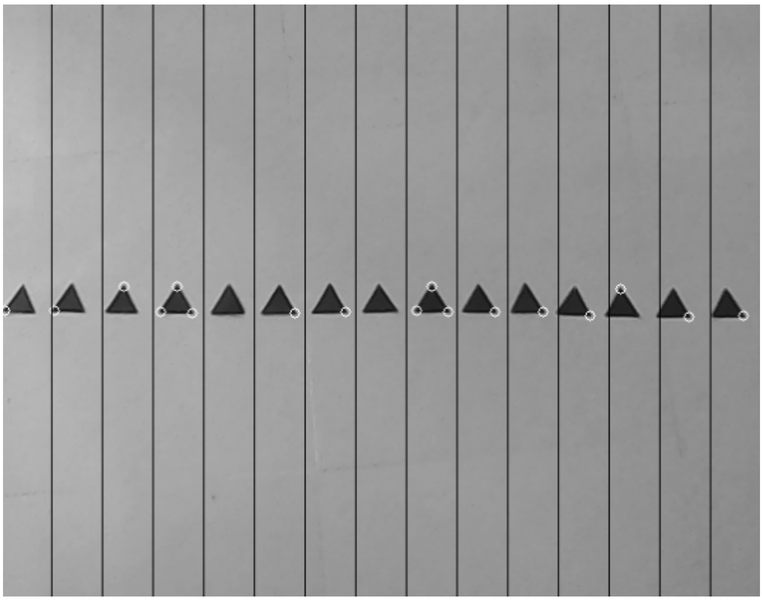


Figure 6.9: Triangle shapes for target detection experiment setup Table 6.5.

Location	#of detection	Detection Probability
0	1137	0.3976
1	1063	0.3717
2	1046	0.3657
3	926	0.3238
4	839	0.2934
5	656	0.2294
6	533	0.1864
7	471	0.1647

Table 6.5: Probability of detection with location of target using triangular targets.

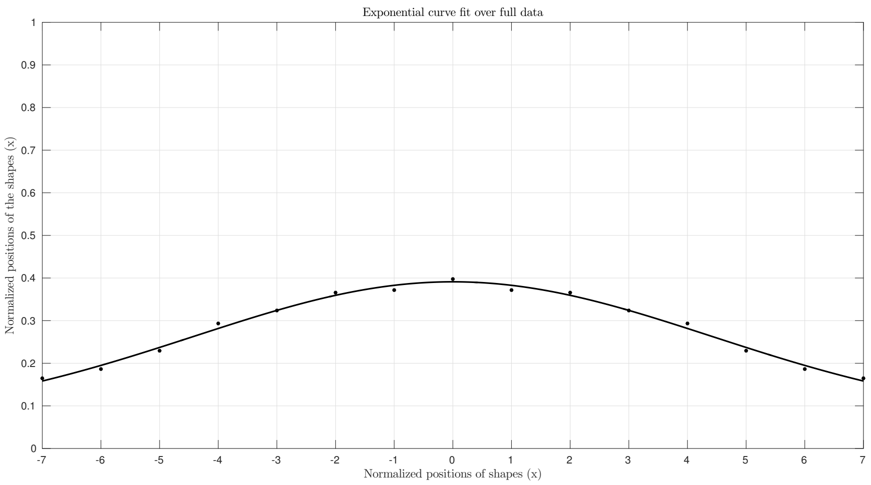


Figure 6.10: Exponential curve fit over the data shown in Table 6.5.

observe that the target detection probability distribution curve with the triangular targets is relatively flatter than that obtained with ArUco markers. We may note that a triangular shape has least maximum target detection probability, while a 5×5 ArUco marker has the highest. However, the effect of noise on flatness of target detection probability is most prominent with the 5×5 ArUco markers. These observations are interesting and may be generalized with more number of experiments with different kind of targets and noise level.

The primary purpose of conducting the above experiments is to obtain the typical variation of the probability of target detection with its position relative to the center of the camera, which in turn can be used in devising deployment/path planning and search strategies for single or multi-quadcopter search using these cameras. The experiments were conducted with usual indoor lighting conditions and hence the illumination across the image frame may not be uniform. Further, exact values of the parameters of the curve fitting over the experimental data depend on several conditions such as nature of the targets, the image processing algorithm used, camera resolution, height of camera, etc. It is not possible to provide a search effectiveness model which fits any camera and any situation. However, it may be observed from the experimental results, that the probability of detection of target is the highest when the target is located directly below the camera and decreases monotonically as the target is moved away from the center. Also, it was observed that an exponential function may be used to fit a curve over the target detection probability data. Thus for any general situation we may use:

$$p(q) = ke^{-\alpha r^2} \tag{6.2}$$

where, $r = \|C - q\|$ is the distance of the target from C the point directly below the sensor. This implies that, if the camera detects a target at $q \in Q$, then the probability of the target present at q, $p(q) = f(r)$. In other words, Eqn. (6.2) gives the confidence level on the camera's ability to detect the target at different points within its FOV. The Eqn. (6.2) models the search effectiveness of a downward facing camera as given in Eqn (4.1). The parameters k and α may be obtained through experiments similar to the one conducted in this chapter, but with the given camera, the environmental conditions, and the targets.

It is interesting to note that the authors in (Guruprasad & Ghose 2011) used such an exponential function, intuitively, and (Delle Fave et al. 2010) used a similar model (with only one parameter α) in an indirect manner to, model the search effectiveness of a search sensor in a multi-UAV/robotic search problem, while most others used either flat effectiveness function (a special case of exponential when α is very small) or some monotonically decreasing function (such as $-r^2$ in (Cortes et al. 2004) in a sensor deployment problem). The observation from the work carried out in this work justifies the use of such a function when downward facing cameras are used as search sensors. The observations from the experiments conducted in this work can also be used to justify a flat sensor effectiveness function (such as that used in (Zhang et al. 2017, Beard & McLain 2003) and several other works) when the target detection probability distribution curve is nearly flat (that is, a very low value of α) as in the case of target detection with added noise.

6.2 EXPERIMENTS USING THE HYBRID SIMULATION PLATFORM

In this section we provide results of simulation experiments carried out using the ROS/Matlab hybrid simulation platform, both to validate the simulation platform developed and demonstrate the proposed search strategy using the multi-quadcopter system.

We use Parrot AR Drone 2.0 quadcopter, one of the very popular affordable quadcopters for our simulation experiments. The simulation platform (Gazebo) uses a full physical model using the URDF file. As mentioned earlier, the controllers within Gazebo simulation environment may be used to control physical AR Drones, and hence experiments with physical quadcopters may also be conducted using the hybrid ROS/Matlab platform developed primarily for simulation purpose. Figure 6.11 shows an outdoor configuration and an indoor configuration of a Parrot AR Drone quadcopter and Figure 6.12 shows its model within the Gazebo simulation platform.

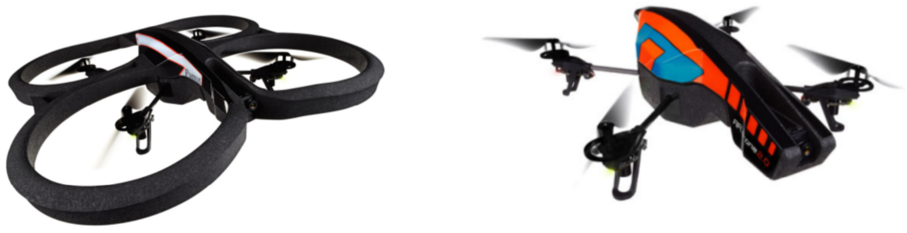

(a) (b)

Figure 6.11: AR Drone (a) outdoor configuration and (b) indoor configuration.

Following are some features of the AR Drone quadcopter:

- The AR Drone 2.0 is powered by brushless DC motors/engine with three-phase current controlled by a micro-controller. It automatically detects the type of engines that are plugged in and automatically adjusts engine controls. It detects if all the engines are turning or are stopped.
- The AR.Drone 2.0 uses a charged 1000mAh, 11.1V LiPo batteries to fly.

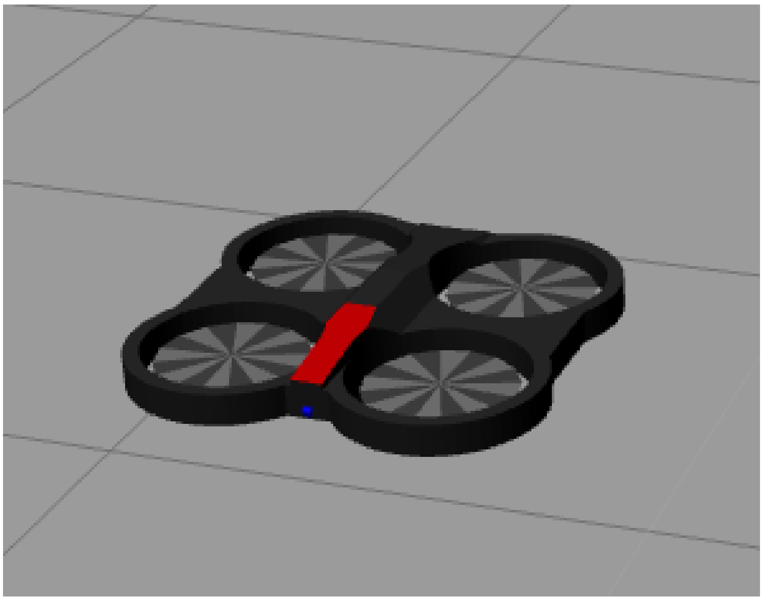


Figure 6.12: AR Drone model within the Gazebo environment.

While flying, the battery voltage decreases from a full charge (12.5 Volts) to low charge (9 Volts). The AR.Drone 2.0 monitors the battery voltage and converts this voltage into a battery life percentage (100% if battery is full, 0% if the battery is low).

- The AR.Drone 2.0 has several motion sensors. They are located below the central hull. The AR.Drone 1.0 features a 6 DOF, MEMS-based, miniaturized inertial measurement unit. It provides the software with pitch, roll and yaw measurements. Inertial measurements are used for automatic pitch, roll, and yaw stabilization, and assisted tilting control.

- An ultrasound telemeter provides with altitude measures for automatic altitude stabilization and assisted vertical speed control.

- A camera aiming towards the ground provides with ground speed measures for automatic hovering and trimming. The frontal camera is a CMOS sensor with a 90 degrees angle lens. In this work we use the downward facing camera.

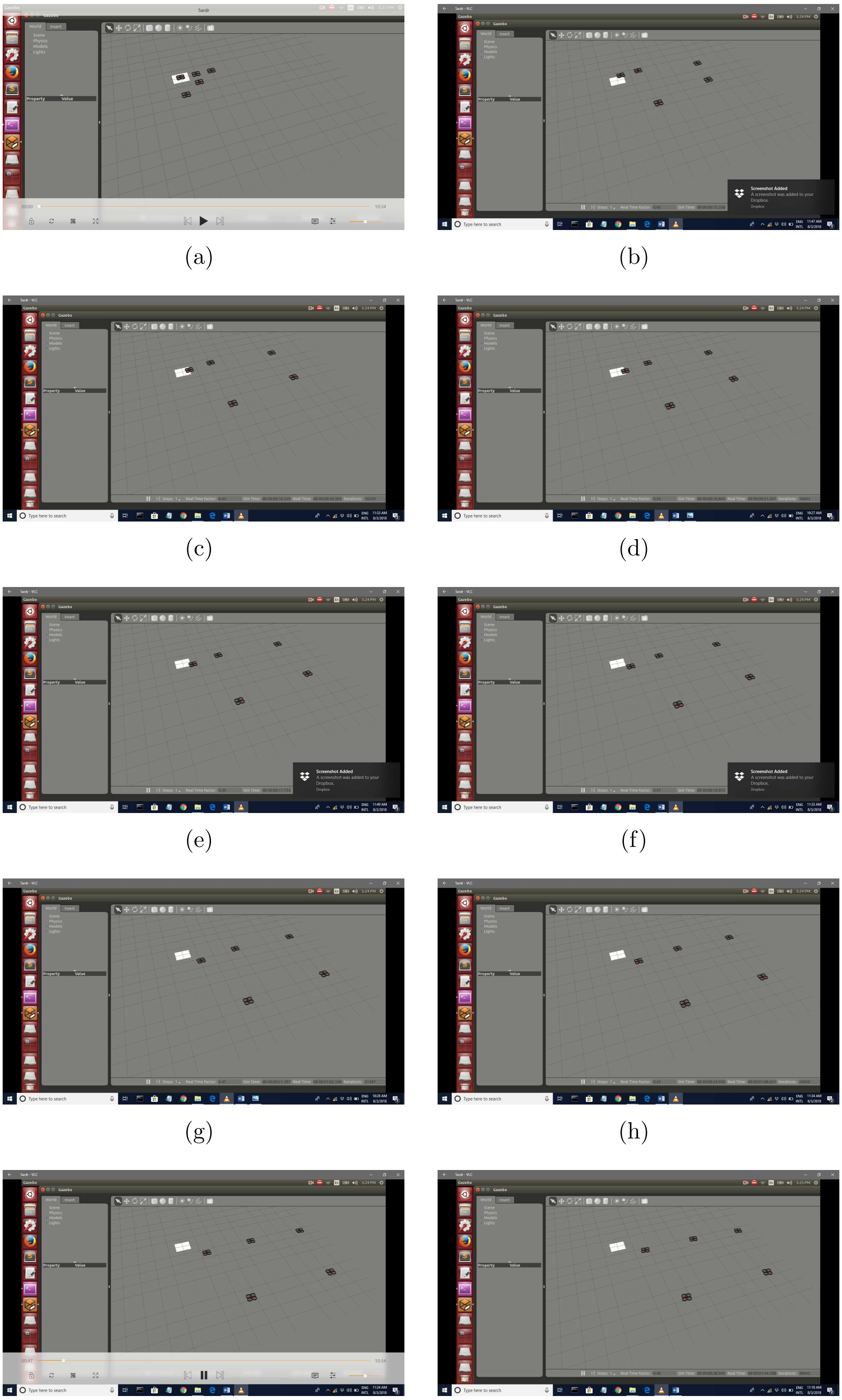
- The AR.Drone 2.0 adds a (in comparison to AR.Drone 1.0) 3 DOF to the IMU with a 3 axis magnetometer. It also adds a pressure sensor to allow altitude measurements at any height.

- The AR.DRone creates its own wifi network with an ESSID and self allocates a free, odd IP address.

6.2.1 Experiments with five AR.Drones

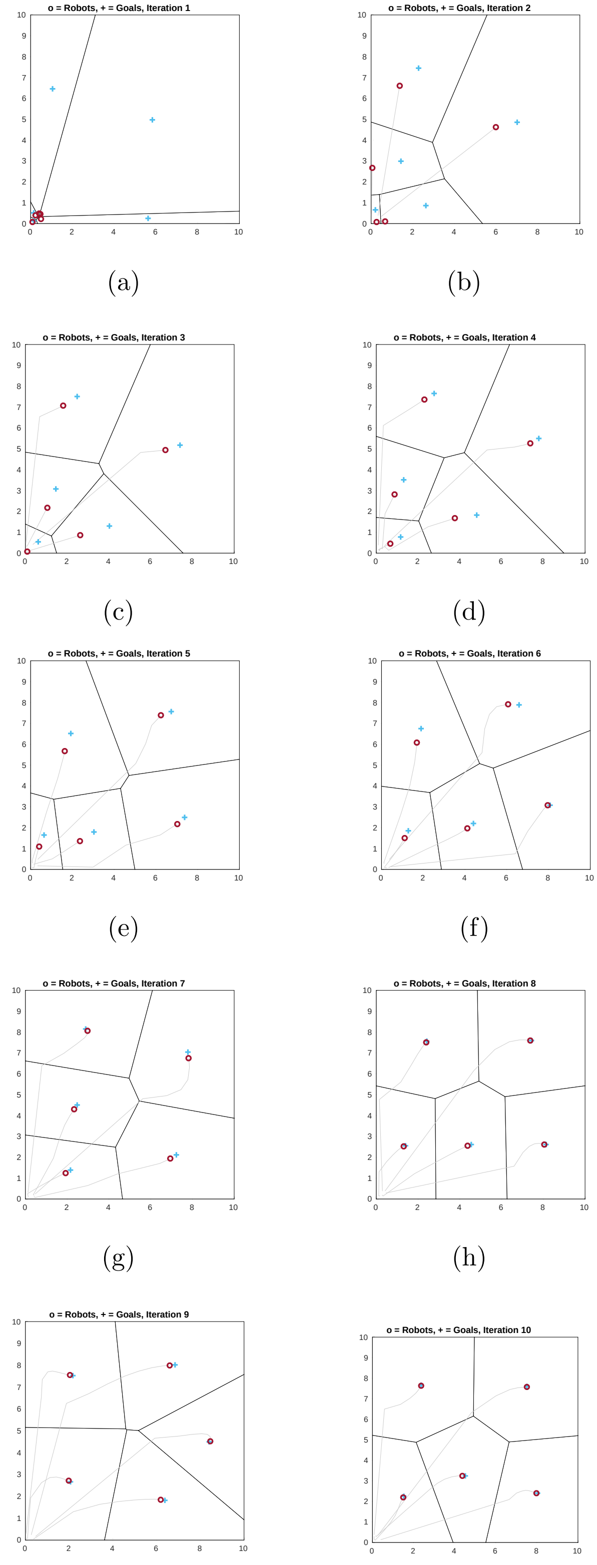
We conducted simulation experiments with 5 quadcopters. Initial configuration of the quadcopters within Gazebo simulator is shown in Figure 6.13 (a). Figure 6.13 shows a few snapshots of the process of optimal deployment. The optimal configuration at the end of deployment step is shown in Figure 6.13 (j). Figures 6.14(a)-(j) show the corresponding snapshots from the Matlab, corresponding to that shown in Figures 6.13. While the actual quadcopters are shown within the simulation environment (in 'machine 1' running ROS) in Figure 6.13, the representative positions as points, the Voronoi cells and the corresponding centroids as computed by the Matlab program (in 'machine 2') are shown in Figure 6.14. Observe, that in the final configuration as shown in Figure 6.14 (j), quadcopter are sufficiently close to the centroids of the Voronoi cells. That is, the quadcopters are deployed into a centroidal Voronoi configuration, an optimal configuration maximizing the search effectiveness as discussed earlier. Note that, though the theoretical results proving that the proportional control law given by Eqn. (3.4) makes the agents reach the centroidal Voronoi configuration asymptotically, is valid only for point mass agents and not the quadcopters. The simulation results demonstrate that the same control law successfully deploys the quadcopters too into the optimal configuration. Here, the nonlinear dynamics of the quadcopters are taken care of by the low level controllers within the flight control modules inbuilt into the ARDrones.

At the end of a optimal deployment search is performed reducing the uncertainty density. The sequence of optimal "deployment" and "search" continue until the average uncertainty is reduced below a specified level. We have assumed an uniform initial uncertainty density as shown in Figure 6.15 (a).



(a) (b) (c) (d) (e) (f) (g) (h) (i) (j)

Figure 6.13: Snapshots of the deployment of quadcopters from an initial configuration to a centroidal Voronoi configuration in Gazebo environment.



(a) (b) (c) (d) (e) (f) (g) (h) (i) (j)

Figure 6.14: Snapshots from Matlab showing the process of optimal deployment of quadcopters. The positions of the quadcopters (based on which the Voronoi partition is created), the Voronoi cells, and the corresponding centroids are also shown in each step.

Figures 6.15 (b) - (f) show how in each search step the uncertainty density is reduced.

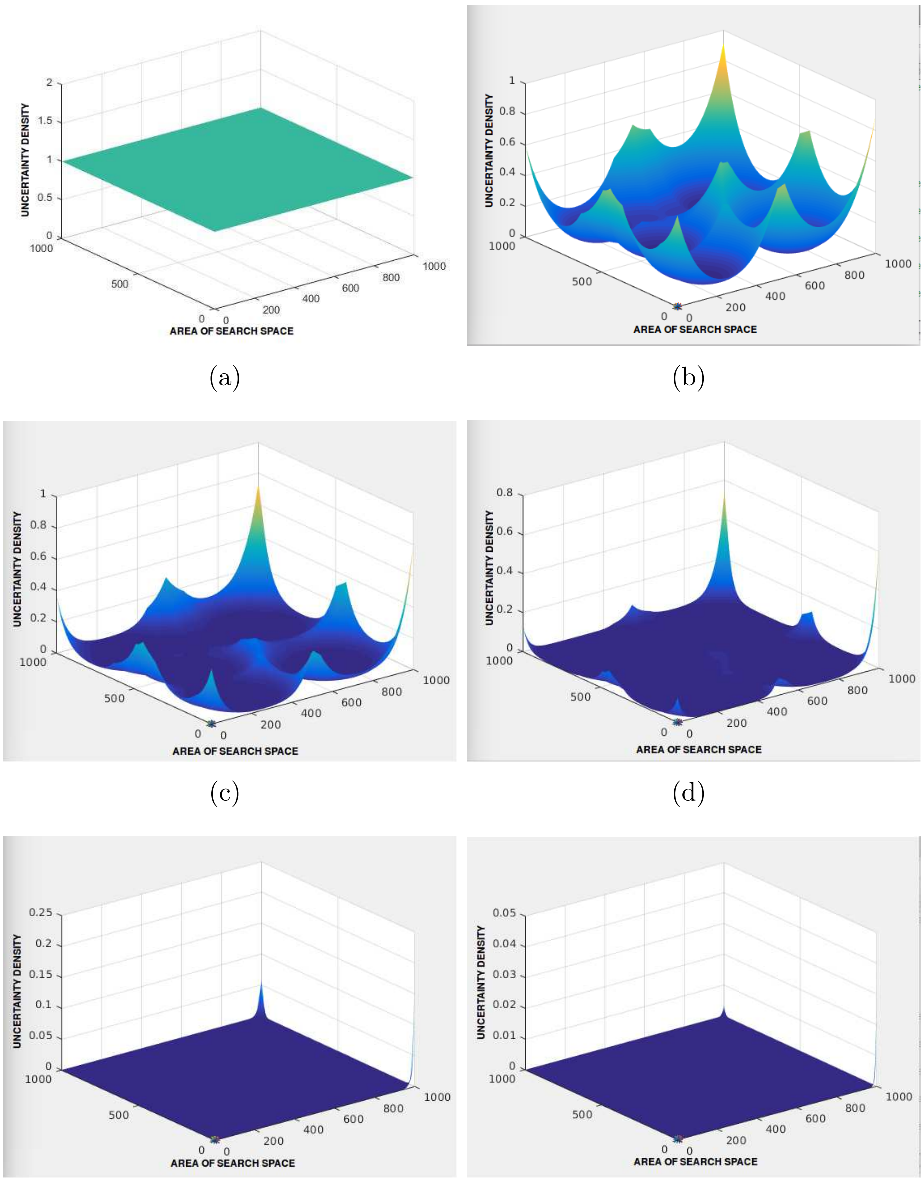
Finally in Figures 6.16(a) and (b) we show how the maximum and average uncertainty ϕ over the search space Q is reduced as the “deployment” and “search” steps/iterations progress, demonstrating that the proposed “deploy” and “search” strategy for multiple downward facing camera mounted ARDrone quadcopters successfully search the space Q.

6.2.2 Experiments with three AR.Drones

In another simulation experiment, we considered only three AR.Drones. In this case the Ar.Drone required 10 search (or ‘deploy and search’) steps to completely search the area (in the sense that the average uncertainty is reduced below the specified threshold). Figures 6.17 and 6.18 show the configuration of the AR.Drones at the end of the corresponding ‘deployment’ (left side) and the uncertainty density distribution after the corresponding ‘search’ (on the right). We may observe that at each ‘deploy and search’ step, the AR.Drone are in an optimal configuration when the searh is performed, as the AR.Drone get the nearly uniformly distributed over the search space as evidenced by nearly uniform size of the corresponding Voronoi cells.

6.2.3 Experiments with different number of AR.Drones

We have carried out simulation experiments using $5, 10, 15, 20, 25, 30, 35, 40$ and 45 quadcopters. Table 6.6 shows the t_{it}, the time consumed for one iteration in the optimal deployment process (Computation of Voronoi cells, their centroids, movement of quadcopters by a small step toward the respective centroids), t_{tot}, the total time required to complete the simulation (involves several deployments iterations and search instances on successful optimal deployment), and N_s, the number of searches (or the ‘deploy’ and ‘search’ steps) to successfully reduce the uncertainty below an acceptable value. As expected t_{it} and t_{tot} increase with the number of quadcopters. In the case of a truly

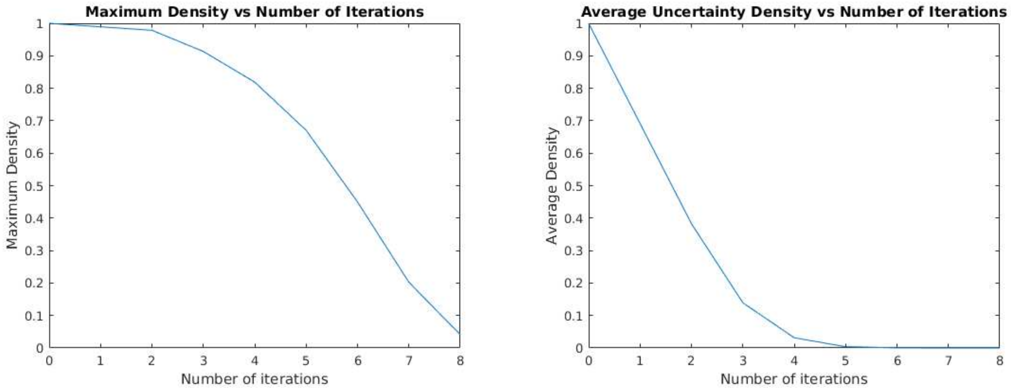


(a) (b)

(c) (d)

(e) (f)

Figure 6.15: Reduction in uncertainty density after each “search” performed at the end of optimal “deployment”.



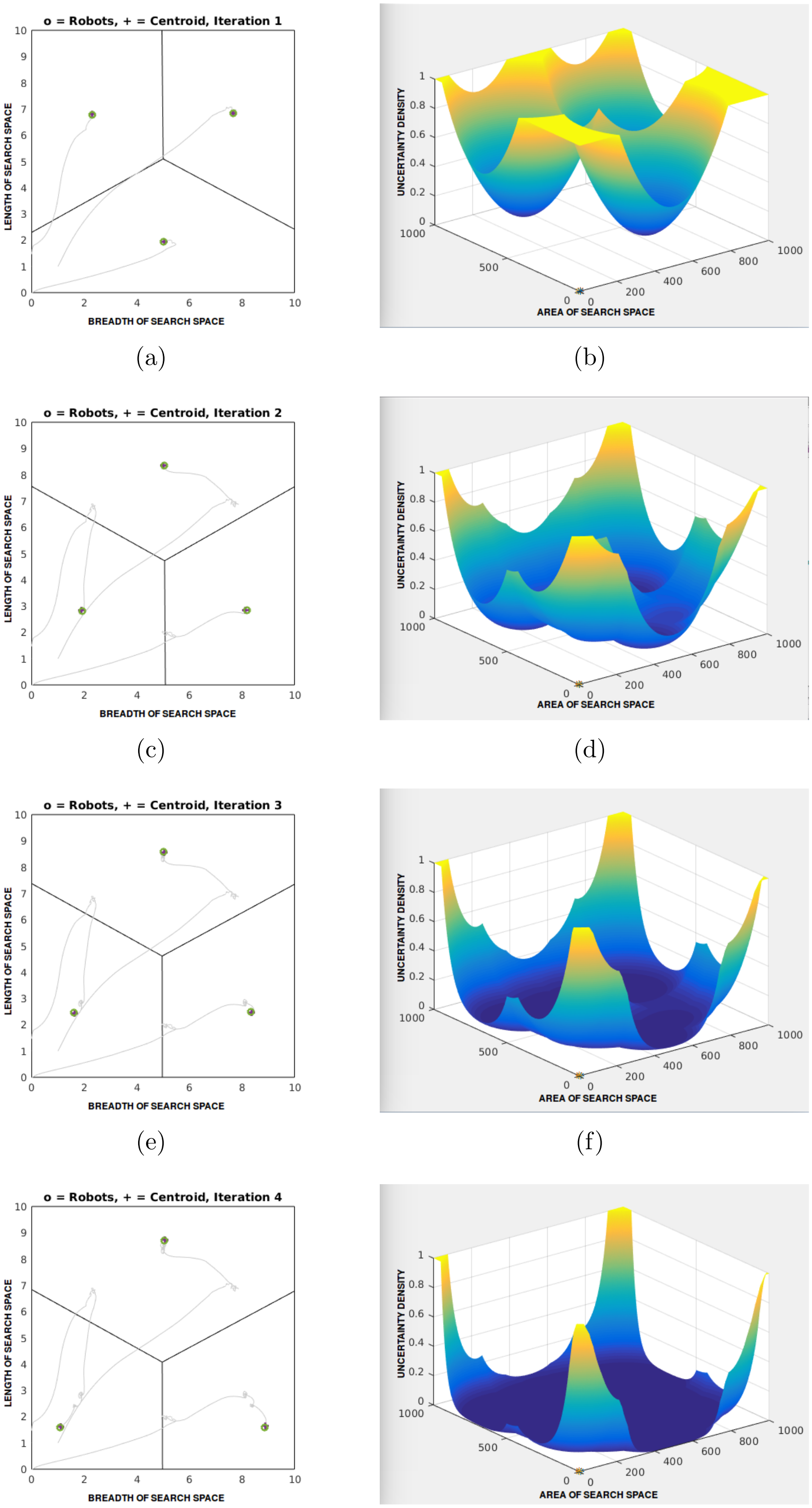
(a) (b)

Figure 6.16: Reduction in (a) maximum uncertainty density and ((b) average uncertainty density over Q as the “deployment” and “search” progress.

distributed (or decentralized) implementation using physical quadcopters, the computational time is independent of the number of quadcopters and is truly scalable. However with a centralized control scheme (‘machine 2’) the computational time increases with number of quadcopters. Though the ‘machine 1’ implements a decentralized architecture, a single computer has to perform all the computations and hence, in a simulation platform, the computational time increases with the number of quadcopters even for a distributed/decentralized implementation. Finally, The number of search instances required initially decreases with the increase in the search agents (quadcopters) (from 5 to 10) as expected, however, after $N = 10$, the number of search instances saturate at a value of 3. Increasing the number search agents is effective only when the sensor range is smaller. These simulation experiments are carried out only to demonstrate that the platform developed can handle large number of quadcopters. Evaluating the performance of the search strategy with number of search agents and other parameters is beyond the scope of this paper.

6.3 EXPERIMENTS WITH DIFFERENT NUMBER OF QUADCOPTERS AND SENSOR RANGE

Now we present the results of a set of simulation experiments carried out using the platform development to investigate the effect of the number search



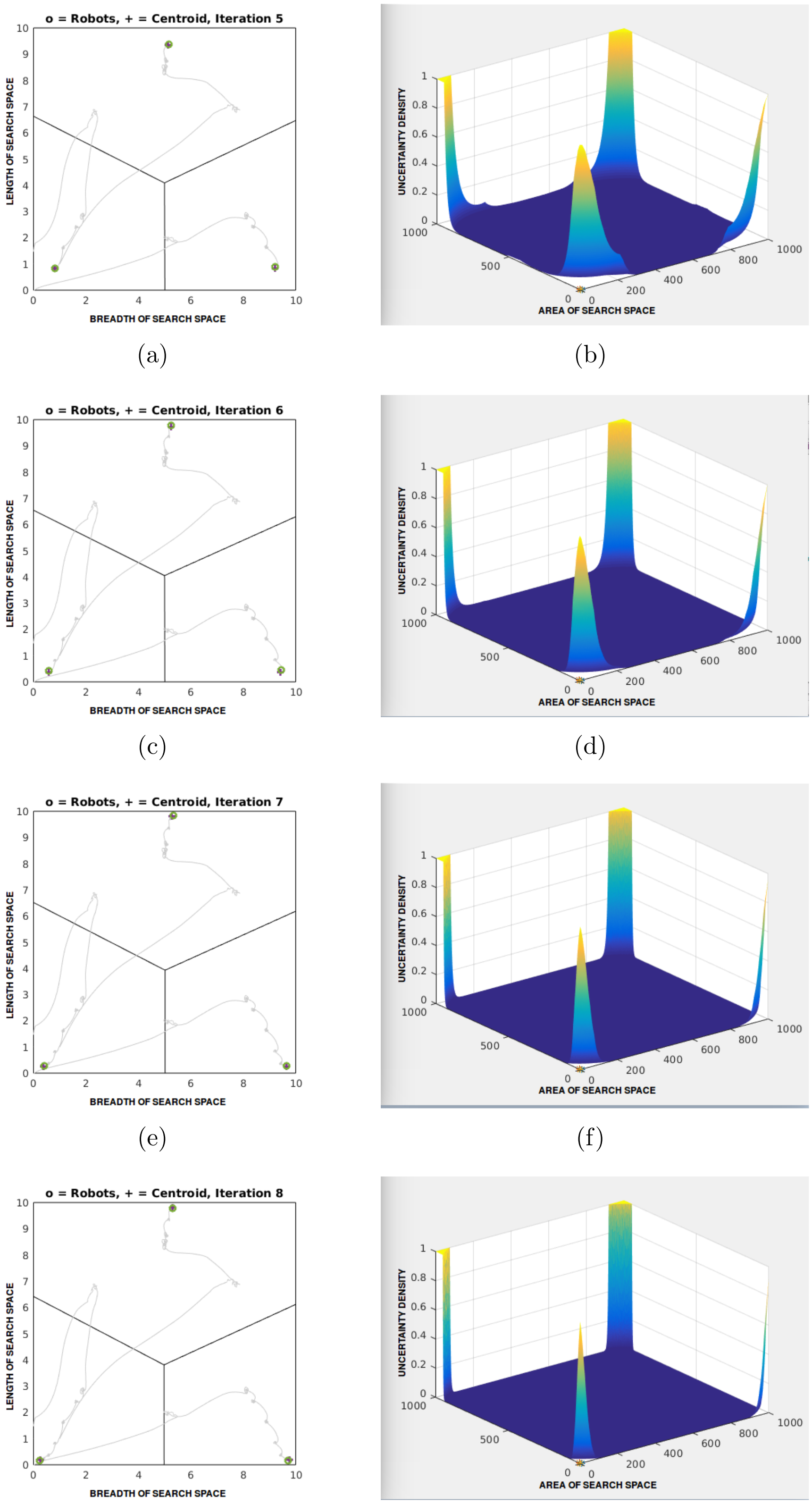
(a) (b)

(c) (d)

(e) (f)

(g) (h)

Figure 6.17: First four ‘deploy and search’ steps with three AR.Drones. Figures in the left side show the configuration of the quadcopters at the end of deployment and those on the right side indicate uncertainty density distribution after the search performed following the deployment.



(a) (b)

(c) (d)

(e) (f)

(g) (h)

Figure 6.18: ‘Deploy and search’ steps 5−8 with three AR.Drones. Figures in the left side show the configuration of the quadcopters at the end of deployment and those on the right side indicate uncertainty density distribution after the search performed following the deployment.

N	5	10	15	20	25	30	35	40	45
$t_{it(s)}$	0.9	1.1	1.3	1.5	1.7	2.0	2.3	2.5	2.6
$t_{tot}(s)$	53.0	51.6	128.1	115.4	151.7	167.2	321.3	352.5	383.7
N_s	6	3	3	3	3	3	3	3	3

Table 6.6: Simulation time for a single iteration in the optimal deployment process (t_{it}), total simulation time t_{tot}, and number of search instances ('deploy' and 'search' steps) N_s with different number (N) of quadcopters.

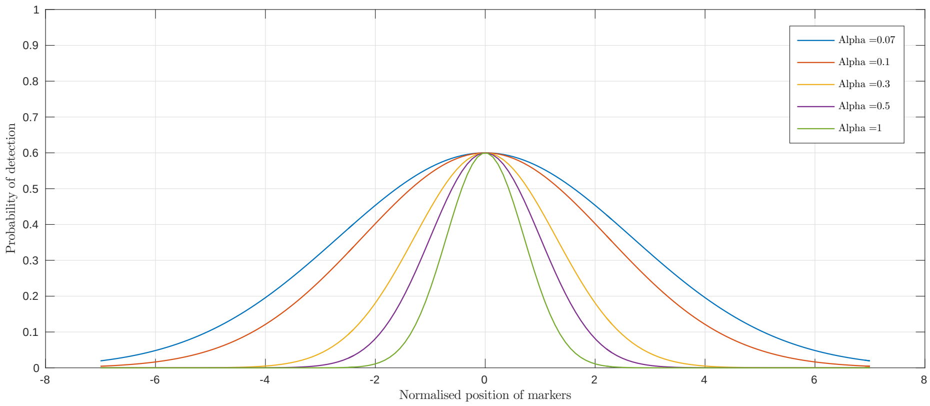


Figure 6.19: Exponential function $ke^{-\alpha r^2}$ with $k = 0.6$ and different vakues of α $(0.07, 0.1, 0.3, 0.5, 1)$.

agents and the search sensor (camera) parameter. In these experiments we did not record the motion of quadcopters in Gazebo environment, but focus on the effectiveness of the search in terms of the number of 'deploy and search' steps required to accomplish the assigned search task.

In this experiment. we consider $3 - 50$ robots and in the sensor effectiveness model $ke^{-\alpha r^2}$, use $\alpha = 0.07, 0.1, 0.3, 0.5$, and 1, which effectively leads to reducing sensor range, as shown in Figure 6.19. In these experiments we keep the value of $k = 0.6$.

Experiments with fixed α

Now we present results of simulation experiments carried out to demonstrate how variation in α, or equivalently the sensor range affects the search performance for a given number of quadcopters performing search.

First we consider $\alpha = 0.07$ and vary N from $3 - 50$ and present the

N	Search Steps	Deployment iterations	Time for simulation(s)
3	6	10, 1, 1, 19, 3, 3	13.39
5	5	14, 1, 1, 13, 2	12.59
10	5	17, 1, 1, 6, 11	25.94
15	4	20, 1, 23, 2	85.22
20	4	36, 1, 13, 2	93.26
30	4	30, 1, 12, 3	124.68
40	4	26,1, 21,3	223.38
50	4	27, 2, 3, 4	298.57

Table 6.7: Search with multiple quadcopters with $\alpha = 0.07$

.

corresponding results. The Table 6.7 tabulates the number of total number of search steps required to complete the search, the number of iterations for each 'deployment' step, and the total simulation time.

Figure 6.20 shows the number of searches required for different the number of search agents (quadcopters) for $\alpha = 0.07$, that is, with the camera search effectiveness of $0.6e^{-0.07r^2}$. We observe that there is a reduction in the number of searches required when we increase the number of quadcopters from 3 to 15. With 3 quadcopter searching the given region we need 6 search steps to successfully reduce the uncertainty to the acceptable level, while require only 5 searches with 5 and 10 quadcopters, and the number of search required gets further reduced to 4 by using 15 quadcopters. This is expected as the primary motivation for using multiple agents for search is to reduce the mission time. However, beyond 15 quadcopters, any increase in the number of quadcopters does not reduce the number of required searches.

Figure 6.21 shows how the average uncertainty density reduces for different number of quadcopters with $\alpha = 0.07$. We may observe that the rate of uncertainty reduction increases with the number of quadcopters as expected.

Next we present similar results with $\alpha = 0.1, 0.3, 0.5$ and 1. Tables 6.8, 6.9, 6.10, and 6.11 tabulate the number of total number of search steps required to complete the search, number of iterations for each 'deployment' step, and the total

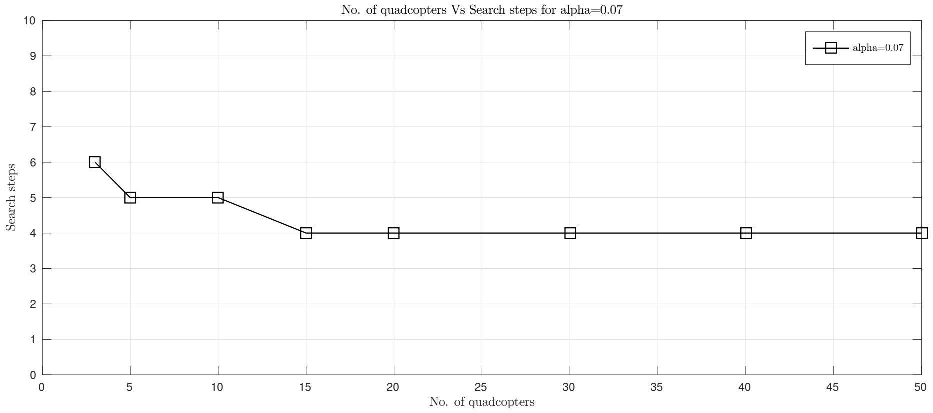


Figure 6.20: Number of search steps required with $\alpha = 0.07$, for different number of searchers, as shown in Table 6.7.

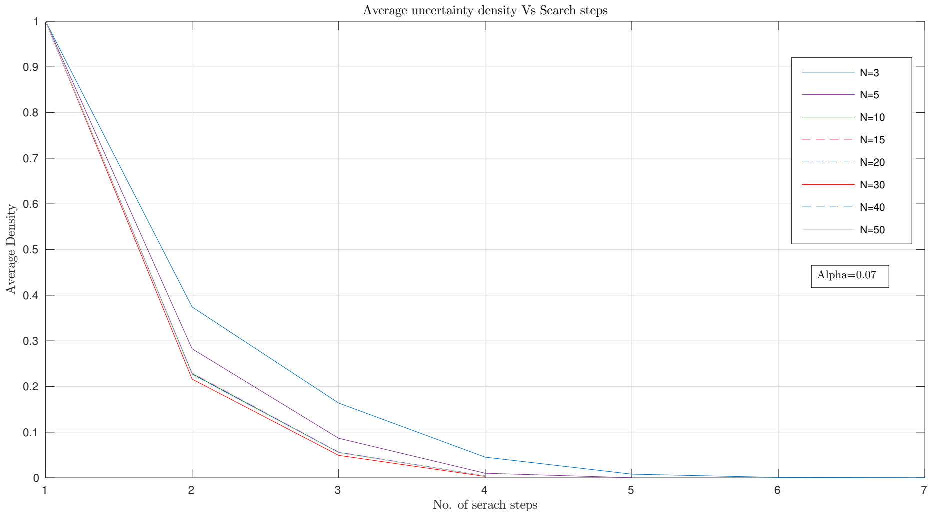


Figure 6.21: Reduction in average uncertainty density for different number of quadcopters with $\alpha = 0.07$.

N	**Search Steps**	**Deployment iterations**	**Total simulation time(s)**
3	8	11, 1, 1, 16, 3, 3, 3, 3	25.25
5	5	16, 1, 1, 11, 2	32.37
10	5	23, 1, 14, 3, 4	53.96
15	4	39, 2, 5, 8	70.86
20	5	25, 1, 20, 1, 17	161.96
30	5	26, 2, 13, 4, 5	163.46
40	4	23, 22, 1, 2	212.97
50	4	35, 1, 12, 2	250.5

Table 6.8: Search with multiple quadcopters with $\alpha = 0.1$

.

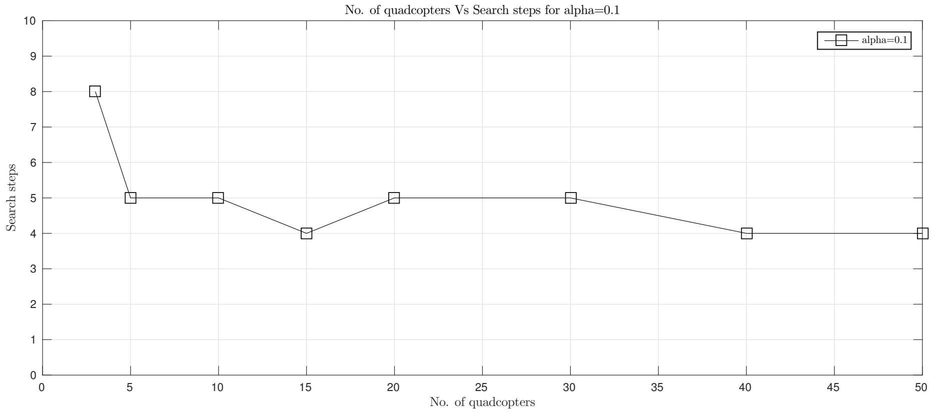


Figure 6.22: Number of search steps required with $\alpha = 0.1$, for different number of searchers, as shown in Table 6.8.

simulation time, for $\alpha = 0.1, 0.3, 0.5$ and 1, respectively. Figures 6.22,6.24,6.26, and 6.28 show the plot of number of requited searches for different number of quadcopters used, for $\alpha = 0.1, 0.3, 0.5$ and 1, respectively. Figures 6.23,6.25,6.27, and 6.29 show the reduction in average uncertainty density as search progresses for different N, corresponding to $\alpha = 0.1, 0.3, 0.5$ and 1, respectively.

Finally Figure 6.30 provides a consolidated result in terms of number of searches required vs the number of quadcopters used for different α.

From the Tables 6.7, 6.8, 6.9, 6.10, and 6.11, Figures 6.20,6.22,6.24,6.28 and Figures 6.21,6.23,6.25,6.27,6.29, and finally Figure 6.30 we may observe following:

1. Increasing N, the number of quadcopters to perform search in general

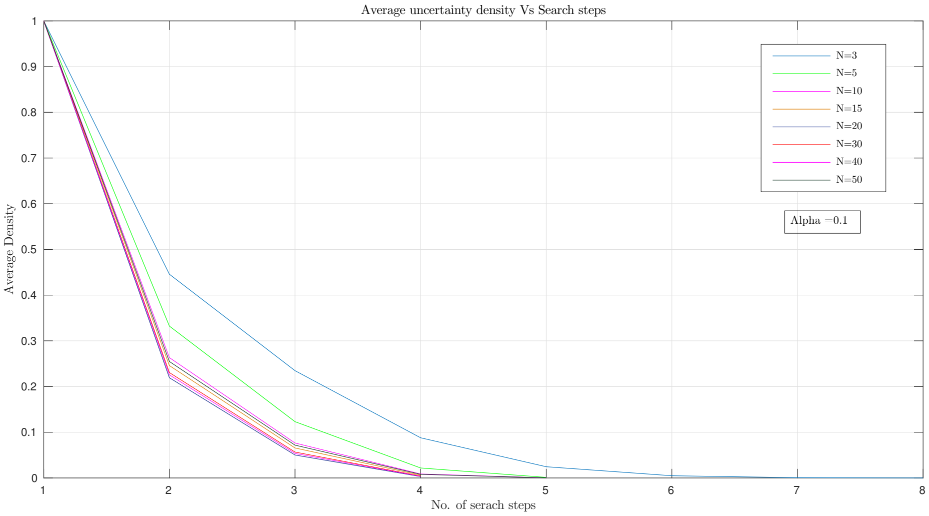


Figure 6.23: Reduction in average uncertainty density for different number of quadcopters with $\alpha = 0.1$.

N	**Search Steps**	**Deployment steps**	**Total Deployment time(sec)**
3	14	10, 1, 1, 1, 15, 3, 3, 3, 3, 4, 4, 2, 1, 1	19.16
5	9	14, 1, 6, 9, 3, 2, 5, 3, 3	18.79
10	7	26, 16, 4, 5, 3, 3, 12	39.75
15	6	24, 14, 7, 4, 6, 12	52.92
20	6	27, 22, 1, 13, 12, 3	84.7
30	6	24, 21, 1, 10, 4, 18	112.68
40	5	35, 8, 2, 17, 3, 16	155.63
50	6	27,24, 2, 12, 4, 16	198.76

Table 6.9: Search with multiple quadcopters with $\alpha = 0.3$

.

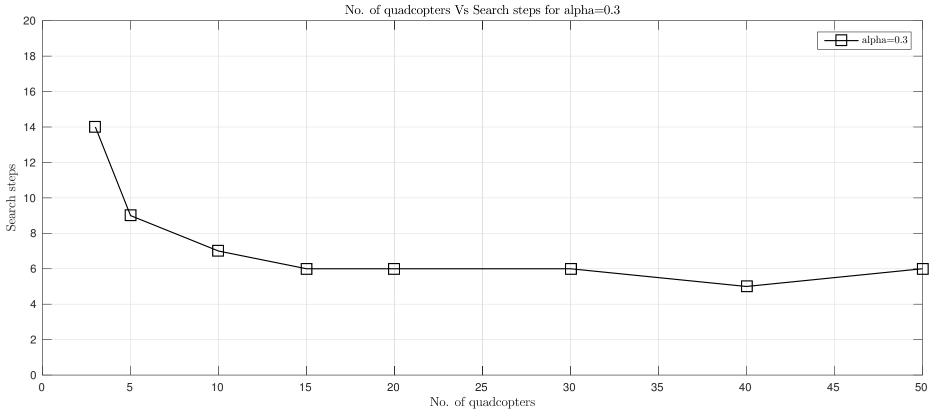


Figure 6.24: Number of search steps required with $\alpha = 0.3$, for different number of searchers, as shown in Table 6.9.

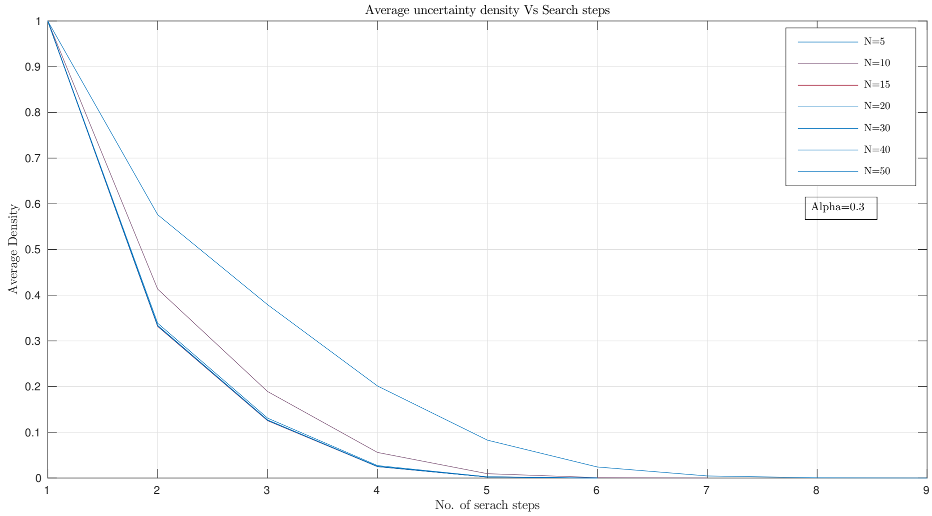


Figure 6.25: Reduction in average uncertainty density for different number of quadcopters with $\alpha = 0.3$.

N	Search Steps	Deployment steps	Total Deployment time(sec)
3	20	12, 1, 1, 1, 1, 1, 11, 4, 3, 2, 2, 3, 3, 3, 3, 3, 1, 1, 1, 1	41.86
5	13	18, 1, 1, 1, 9, 2, 2, 2, 3, 6, 4, 8, 3	51.41
10	9	29, 15, 1, 8, 3, 7, 9, 5, 4	110.9
15	8	23, 12, 1, 19, 6,10, 2, 1	106.4
20	7	36, 7, 9, 18, 8, 12, 3	143.79
30	7	32, 16, 11, 3, 10, 21, 17	160.53
40	7	25, 21, 9, 6, 18, 20, 33	784.29
50	7	33, 6, 8, 7, 14, 27, 20	791

Table 6.10: Search with multiple quadcopters with $\alpha = 0.5$

.

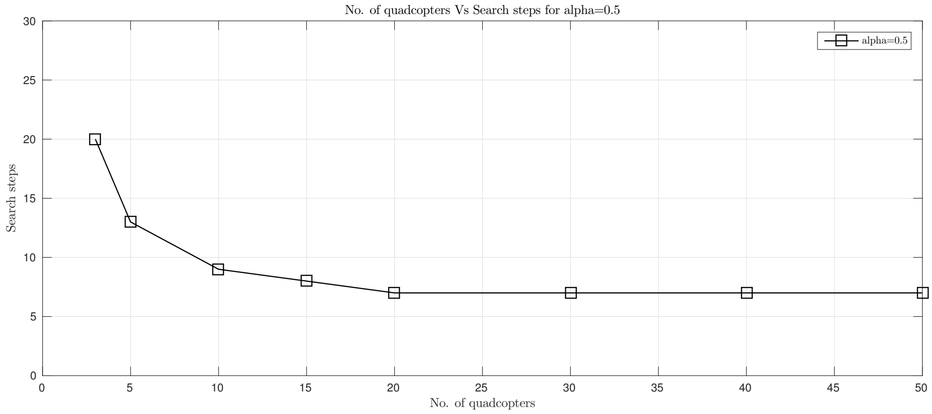


Figure 6.26: Number of search steps required with $\alpha = 0.5$, for different number searchers, as shown in Table 6.10.

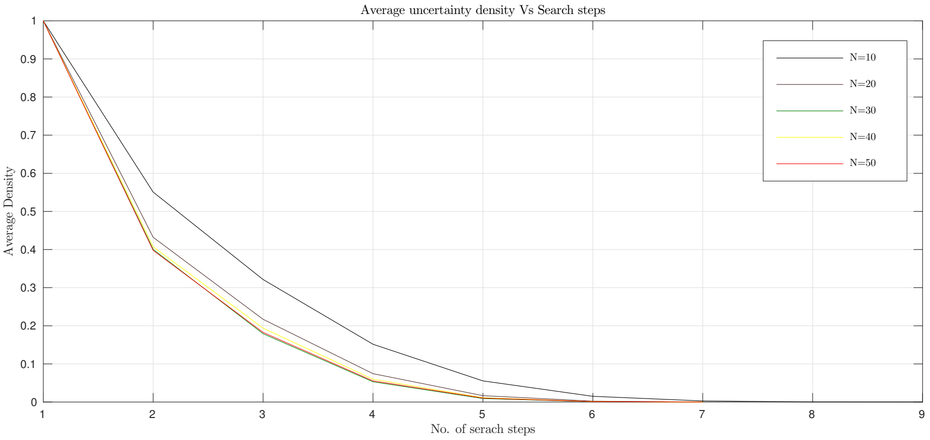


Figure 6.27: Reduction in average uncertainty density for different number of quadcopters with $\alpha = 0.5$.

N	**Search Steps**	**Deployment steps**	**Total Deployment time(sec)**
3	36	10, 1, 1, 1, 1, 1, 1, 1, 1, 1, 12, 2, 2, 2, 2, 2, 2, 2, 3, 3, 3, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 7, 1	36.37
5	22	17, 1, 1, 1, 1, 11, 1, 1, 2, 1, 2, 2, 5, 6, 2, 6, 2, 3, 2, 2, 1, 1	32.93
10	13	28, 13, 2, 7, 6, 10, 4, 6, 6, 3, 15, 1, 1	62.27
15	11	29, 13, 8, 2, 8, 8, 2, 11, 2, 1, 1	69.02
20	10	39, 1, 17, 7, 6, 14, 3, 6, 2, 2	94.98
30	10	27, 14, 14, 7, 16, 15, 14, 8, 1, 26	204.88
40	10	25, 18, 8, 3, 8, 36, 10, 6, 15, 4	239.5

Table 6.11: Search with multiple quadcopters with $\alpha = 1$.

.

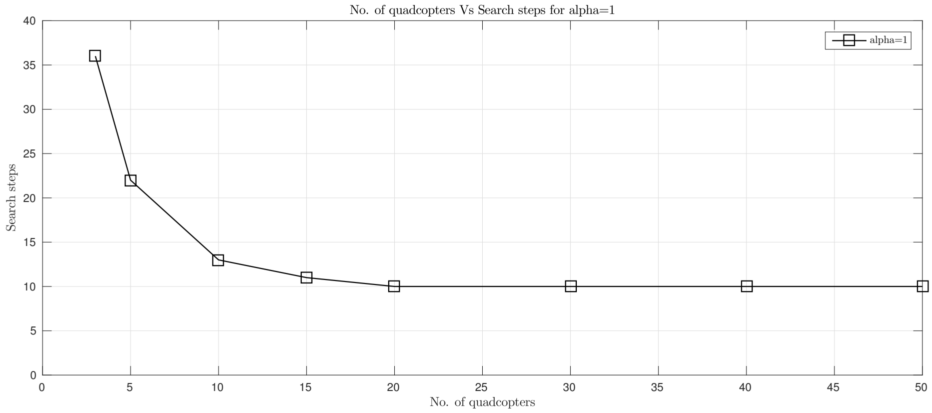


Figure 6.28: Number of search steps required with $\alpha = 1$, for different number searchers, as shown in Table 6.11.

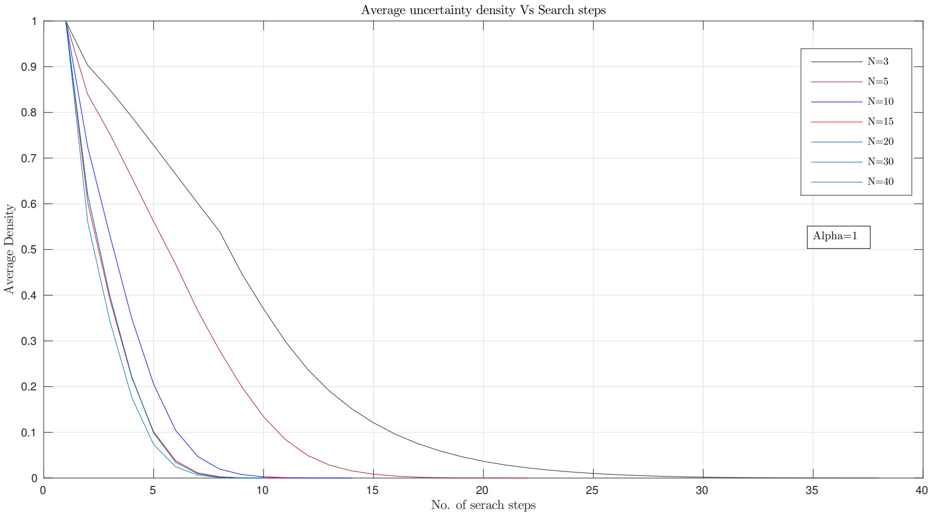


Figure 6.29: Reduction in average uncertainty density for different number of quadcopters with $\alpha = 1$.

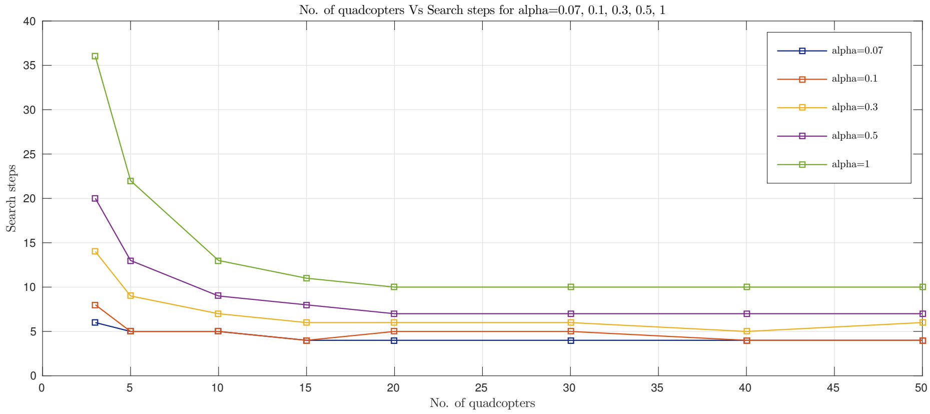


Figure 6.30: Number of search steps required with $\alpha = 0.07, 0.1, 0.3, 0.5, 1$ and $N = 3, 5, 10, 15, 20, 30, 40, 50$

reduces the number of requires searches.

2. However, the performance in terms of the number of required searchers does not improve beyond some value of N, which depend on the value of α.
3. With increase in the value of α, which is effectively a reduction in the range of the camera, the value of N, where a saturation in the performance in terms of the number of required searchers occurs increases.
4. Thus, as the sensor range is reduced, it is advisable to use a large number of quadcopters to achieve a faster search.

Experiments with fixed number of quadcopters

Now we present a similar results with fixing the number of quadcopter performing search at a values from $\{3, 5, 10, 15, 20, 30, 40, 50\}$ and vary the parameter α.

Tables 6.12, 6.13, 6.14, 6.15, 6.16, 6.17, 6.18, 6.19 tabulate the number of total number of search steps required to complete the search, number of iterations for each 'deployment' step, and the total simulation time, for $3, 5, 15, 20, 30, 40,$ and 50 quadcopters performing search with different α. Figures 6.31, 6.32, 6.33, 6.34, 6.35, 6.36, 6.37, 6.38 show variation of the number of search steps required with variation of α, $N = 3, 5, 10, 15, 20, 30, 40,$ and 50, respectively.

N=3	Search Steps	Deployment iterations	Total simulation time(s)
$\alpha = 0.07$	6	10, 1, 1, 19, 3, 3	13.39
$\alpha = 0.1$	8	11, 1, 1, 16, 3, 3, 3, 3	25.25
$\alpha = 0.3$	14	10, 1, 1, 1, 15, 3, 3, 3, 3, 4, 4, 2, 1, 1	19.16
$\alpha = 0.5$	20	12, 1, 1, 1, 1, 1, 11, 4, 3, 2, 2, 3, 3, 3, 3, 3, 1, 1, 1, 1	41.86
$\alpha = 1$	36	10, 1, 1, 1, 1, 1, 1, 1, 1, 1, 12, 2, 2, 2, 2, 2, 2, 2, 3, 3, 3, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 7, 1	36.37

Table 6.12: Search using $\alpha = 0.07, 0.1, 0.3, 0.5, 1$ and $N = 3$.

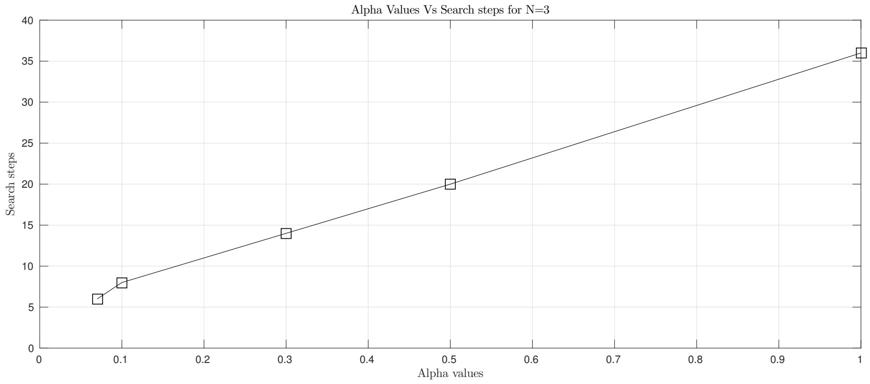


Figure 6.31: Number of searches required with varying α with $N = 3$.

N=5	Search Steps	Deployment iteration	Total simulation time(s)
$\alpha = 0.07$	5	14, 1, 1, 13, 2	12.59
$\alpha = 0.1$	5	16, 1, 1, 11, 2	32.37
$\alpha = 0.3$	9	14, 1, 6, 9, 3, 2, 5, 3, 3	18.79
$\alpha = 0.5$	13	18, 1, 1, 1, 9, 2, 2, 2, 3, 6, 4, 8, 3	51.41
$\alpha = 1$	22	17, 1, 1, 1, 1, 11, 1, 1, 2, 1, 2, 2, 5, 6, 2, 6, 2, 3, 2, 2, 1, 1	32.93

Table 6.13: Search using $\alpha = 0.07, 0.1, 0.3, 0.5, 1$ and $N = 5$.

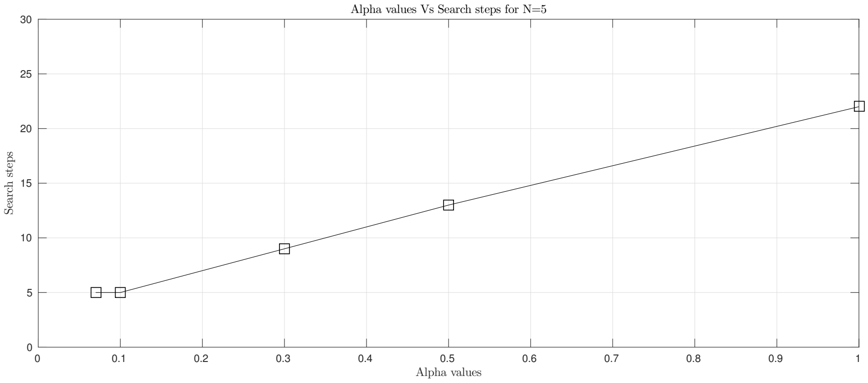


Figure 6.32: Number of searches required with varying α with $N = 5$.

N=10	**Search Steps**	**Deployment iterations**	**Total simulation time(s)**
$\alpha = 0.07$	5	17, 1, 1, 6, 11	25.94
$\alpha = 0.1$	5	23, 1, 14, 3, 4	53.96
$\alpha = 0.3$	7	26, 16, 4, 5, 3, 3, 12	39.75
$\alpha = 0.5$	9	29, 15, 1, 8, 3, 7, 9, 5, 4	110.9
$\alpha = 1$	13	28, 13, 2, 7, 6, 10, 4, 6, 6, 3, 15, 1, 1	62.27

Table 6.14: Search using $\alpha = 0.07, 0.1, 0.3, 0.5, 1$ and $N = 10$.

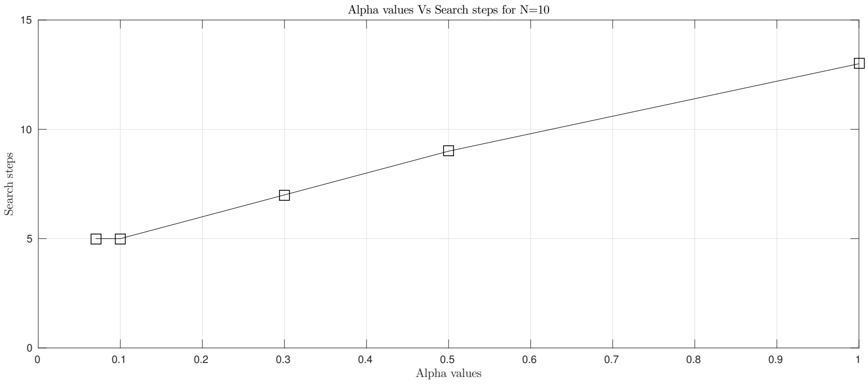


Figure 6.33: Number of searches required with varying α with $N = 10$.

N=15	**Search Steps**	**Deployment iterations**	**Total simulation time(s)**
$\alpha = 0.07$	4	20, 1, 23, 2	85.22
$\alpha = 0.1$	4	39, 2, 5, 8	70.86
$\alpha = 0.3$	6	24, 14, 7, 4, 6, 12	52.92
$\alpha = 0.5$	8	23, 12, 1, 19, 6,10, 2, 1	106.4
$\alpha = 1$	11	29, 13, 8, 2, 8, 8, 2, 11, 2, 1, 1	69.02

Table 6.15: Search using $\alpha = 0.07, 0.1, 0.3, 0.5, 1$ and $N = 15$.

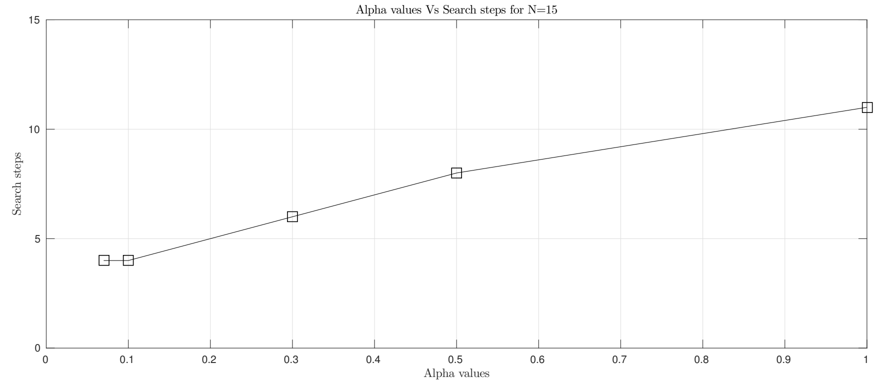


Figure 6.34: Number of searches required with varying α with $N = 15$.

N=20	**Search Steps**	**Deployment iterations**	**Total simulation time(s)**
$\alpha = 0.07$	4	36, 1, 13, 2	93.26
$\alpha = 0.1$	5	25, 1, 20, 1, 17	161.96
$\alpha = 0.3$	6	27, 22, 1, 13, 12, 3	84.7
$\alpha = 0.5$	7	36, 7, 9, 18, 8, 12, 3	143.79
$\alpha = 1$	10	39, 1, 17, 7, 6, 14, 3, 6, 2, 2	94.98

Table 6.16: Search using $\alpha = 0.07, 0.1, 0.3, 0.5, 1$ and $N = 20$.

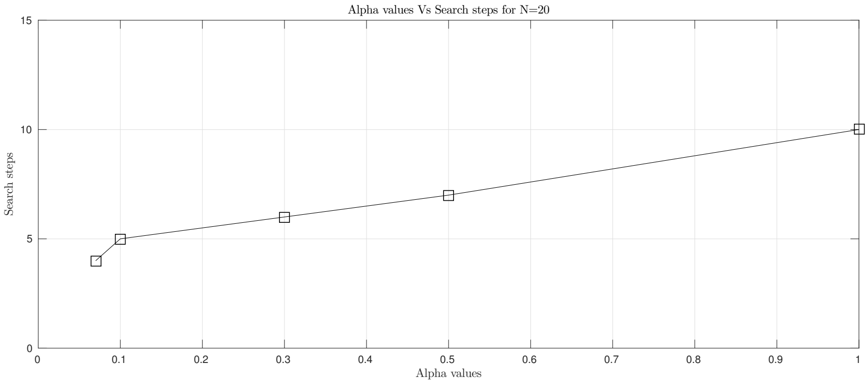


Figure 6.35: Number of searches required with varying α with $N = 20$.

N=30	**Search Steps**	**Deployment iterations**	**Total simulation time(s)**
$\alpha = 0.07$	4	30, 1, 12, 3	124.68
$\alpha = 0.1$	5	26, 2, 13, 4, 5	163.46
$\alpha = 0.3$	6	24, 21, 1, 10, 4, 18	112.68
$\alpha = 0.5$	7	32, 16, 11, 3, 10, 21, 17	160.53
$\alpha = 1$	10	27, 14, 14, 7, 16, 15, 14, 8, 1, 26	204.88

Table 6.17: Search using $\alpha = 0.07, 0.1, 0.3, 0.5, 1$ and $N = 30$.

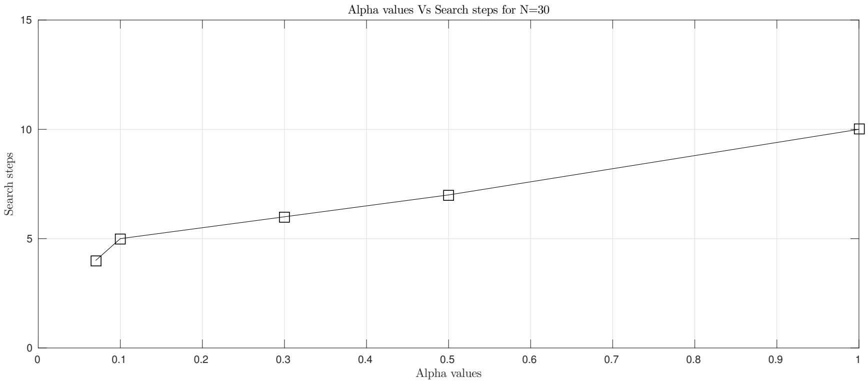


Figure 6.36: Number of searches required with varying α with $N = 30$.

N=40	**Search Steps**	**Deployment iterations**	**Total simulation time(s)**
$\alpha = 0.07$	4	26,1, 21,3	223.38
$\alpha = 0.1$	4	23, 22, 1, 2	212.97
$\alpha = 0.3$	5	35, 8, 2, 17, 3, 16	155.63
$\alpha = 0.5$	7	25, 21, 9, 6, 18, 20, 33	784.29

Table 6.18: Search using $\alpha = 0.07, 0.1, 0.3, 0.5, 1$ and $N = 40$.

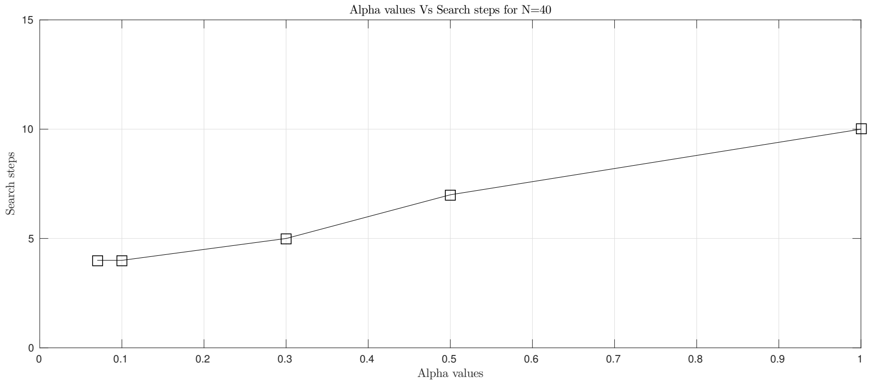


Figure 6.37: Number of searches required with varying α with $N = 40$.

N=50	**Search Steps**	**Deployment iteration**	**Total simulation time(s)**
$\alpha = 0.07$	4	27, 2, 3, 4	298.57
$\alpha = 0.1$	4	35, 1, 12, 2	250.5
$\alpha = 0.3$	6	27,24, 2, 12, 4, 16	198.76
$\alpha = 0.5$	7	33, 6, 8, 7, 14, 27, 20	791

Table 6.19: Search using $\alpha = 0.07, 0.1, 0.3, 0.5, 1$ and $N = 50$.

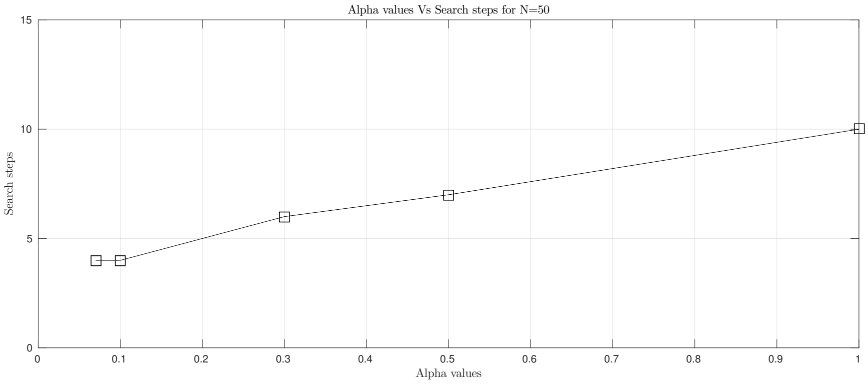


Figure 6.38: Number of searches required with varying α with $N = 50$.

It can be observed from Tables 6.12, 6.13, 6.14, 6.16, 6.17, 6.18, 6.19 and Figures 6.31, 6.32, 6.33, 6.34, 6.35, 6.36, 6.37, 6.38, that as the α is increased for a given N, the number of search steps required increase monotonically. This is intuitive as with lower α, the sensor range is higher, and hence the reduction in uncertainty density in any search step is higher.

Uncertainty reduction with varying α and N

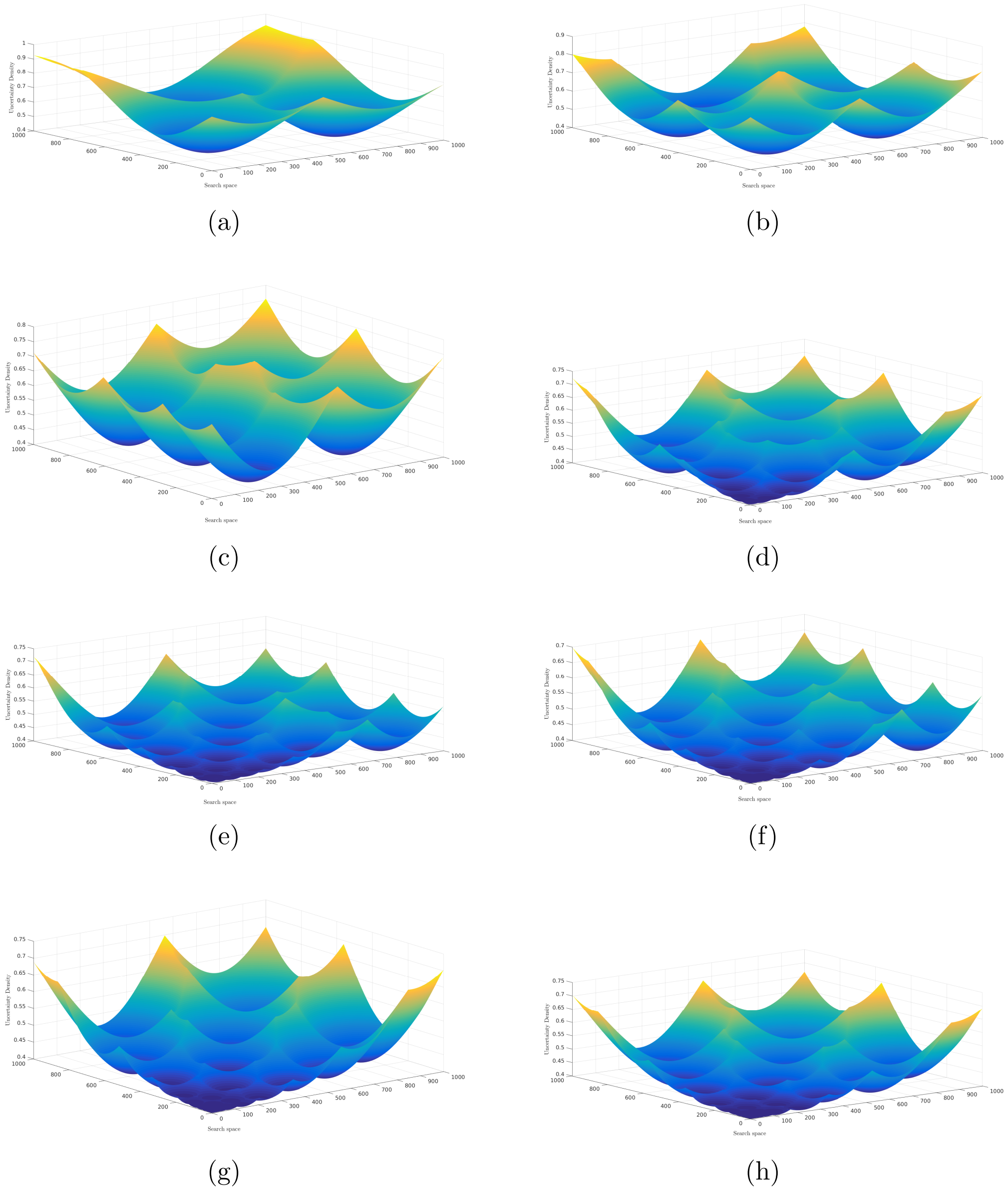
Here we show how the uncertainty density gets reduced in the (first) search instance when α and N area varied, using a few representative results.

Figure 6.39 shows uncertainty density reduction in each search step with $\alpha = 0.07$, for different number of quadcopters.

Figure 6.40 shows uncertainty density reduction in each search step with $\alpha = 0.3$, for different number of quadcopters.

Figure 6.41 shows uncertainty density reduction in each search step with $\alpha = 1$, for different number of quadcopters.

We may observe from Figures 6.39-6.41 that with lower α (as in Figures 6.39, with $\alpha = 0.007$) the sensor coverage of each camera is higher ,leading to better overall coverage of the area, and hence much quicker uncertainty reduction, compared to that with higher values of α (as in Figures 6.41, with $\alpha = 1$, for example.). We have provided only representative results with $\alpha = 0.07, 0.3$, and 1 as it gives us how α affects search performance and helps deiced the optimal



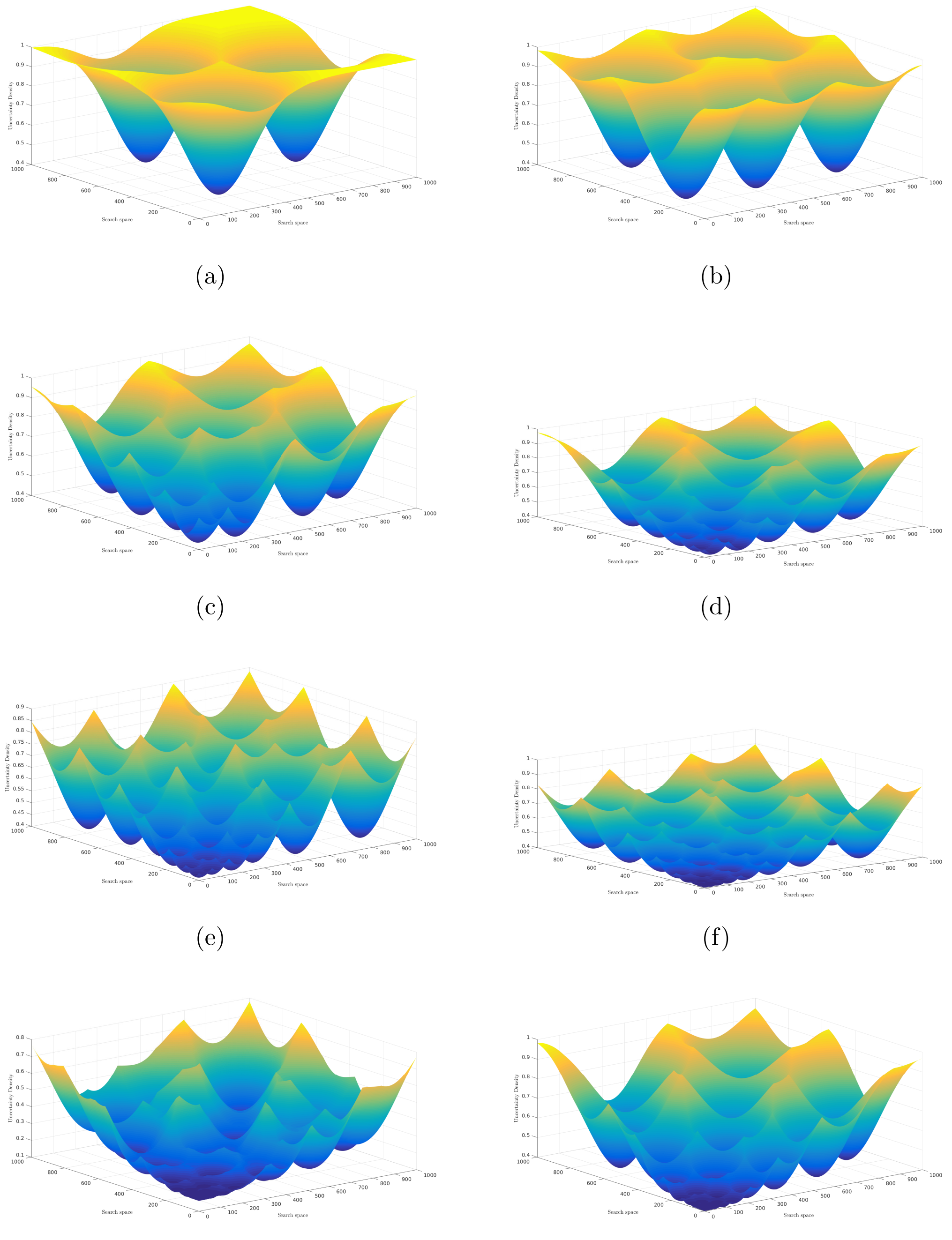
(a) (b)

(c) (d)

(e) (f)

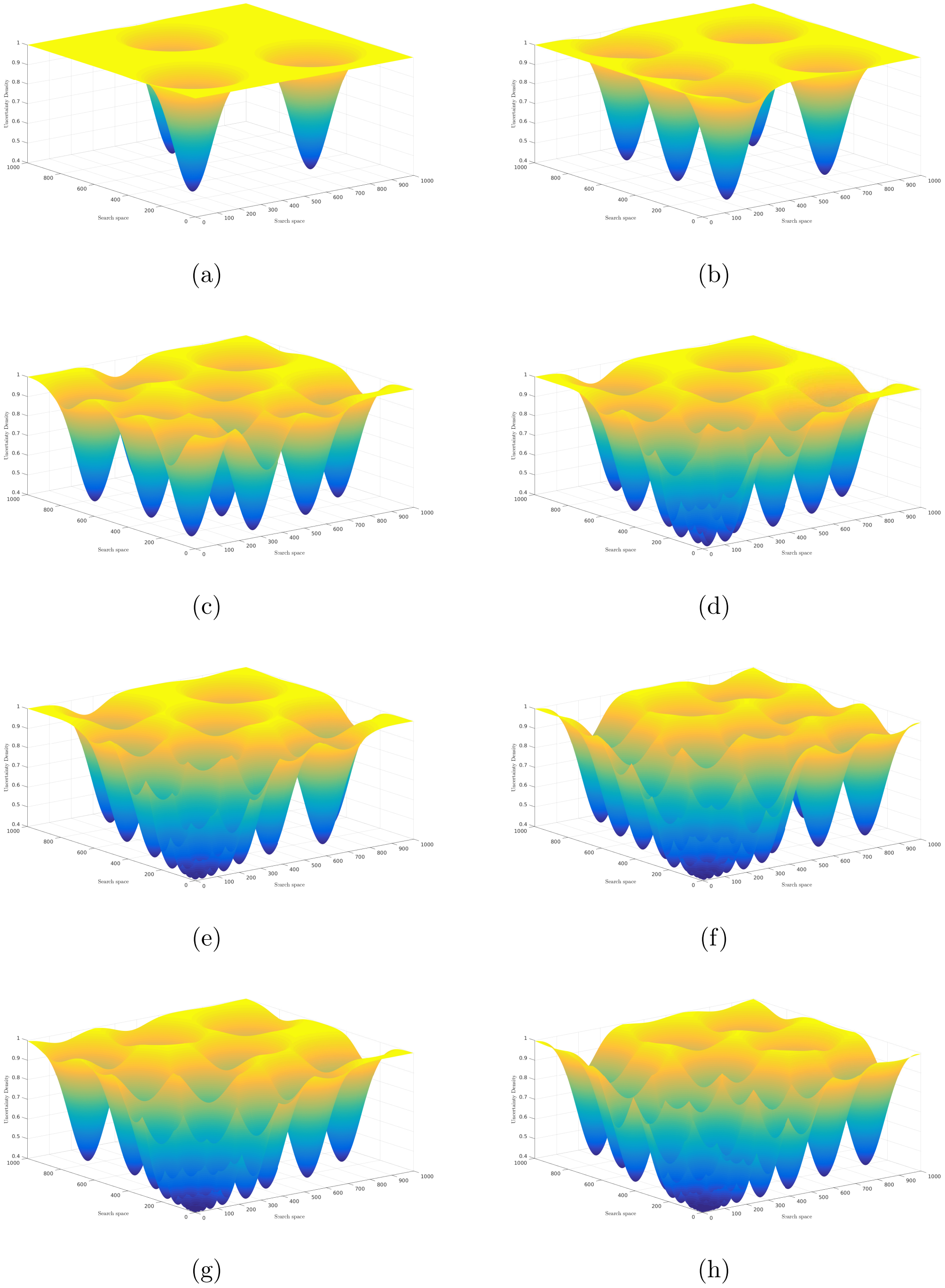
(g) (h)

Figure 6.39: First search instance with $\alpha = 0.07$ and different N.



(a) (b) (c) (d) (e) (f) (g) (h)

Figure 6.40: First search instance with $\alpha = 0.3$ and different N.



(a) (b) (c) (d) (e) (f) (g) (h)

Figure 6.41: First search instance with $\alpha = 1$ and different N.

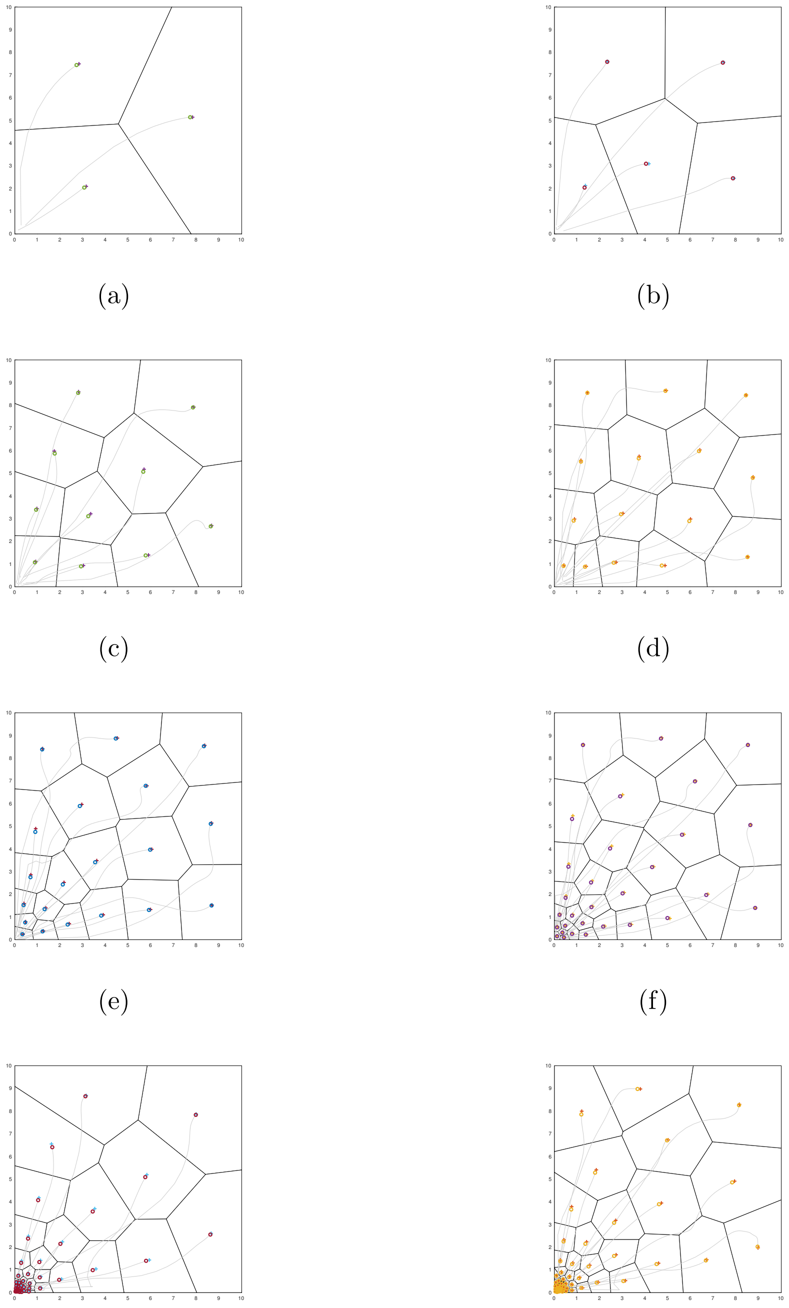
number of quadcopter required.

Optimal deployment with different N

Here we show using the first optimal deployment to show how the number of robots affect the area coverage and hence the search performance. Figure 6.42 shows first configuration of the quadcopters at the end of first deployment step with $\alpha = 0.07$ and different N. We may observe that with lower N (say 3) the Voronoi cells are of more uniform size as compared to that with higher N (say $40, 50$). This non non-unfirmity of deployment affects the search performance. This is due to the way we implement the Lloyd's algorithm based optimal deployment. We stop the deployment process when the quadcopters are closer to the respective centroids by a predefined tolerance d_{tol}. We have used same value of d_{tol} for all values of N. A more uniform deployment may be obtained by reducing this tolerance for scenario with higher N. Another important observation from these Figures is that the area of each Voronoi cell reduces as N increases. Hence, a camera with lower range (that is, higher value of α) is suitable for higher N.

6.4 SUMMARY

In this chapter, we presented detailed results of experiments and simulation carried out along with a detailed discussion on the same. We presented the results of the experiments carried out to obtain the search effectiveness model of the downward-facing camera, which we use in the proposed multi-quadcopter search strategy. We then presented a representative result of the simulation experiment carried out using the realistic ROS/Matlab simulation platform, both to demonstrate the simulation platform itself and the proposed search strategy. Finally, we provided a detailed account of simulation experiments carried out to evaluate the effect of the number of search quadcopters and the camera effectiveness parameters on the performance of the proposed multi-quadcopter search strategy, using the simulation platform developed in this work.



(a) (b)

(c) (d)

(e) (f)

(g) (h)

Figure 6.42: First search instance with $\alpha = 0.07$ and $N = 3, 5, 10, 15, 20, 30, 40,$ and 50.

CHAPTER 7

CONCLUSIONS AND SCOPE FOR THE FUTURE WORK

In this chapter, we summarize the contributions of this thesis and also provide a brief discussion on the scope for future work. First, we provide a summary of the main contributions of this thesis.

7.1 CONCLUSIONS

In this thesis we addressed a practically very useful, and academically challenging and interesting problem of cooperative multi-agent search.

We formulated a multi-agent search strategy using quadcopter UAVs as search agents/vehicles and downward-facing cameras mounted on the quadcopters as search agents. Based on practical considerations, we assumed that the search effectiveness of the camera is maximum at the center and degrades away from it, unlike in most work in the literature where it is assumed to be constant over the entire image frame. The lack of information about presence or absence or the targets of interest in the search space are modelled as an uncertainty density distribution. Here, the uncertainty is 1 when no information on the existence (or absence) of the target at a point of interest is available and 0 when it is established that the target is either present or absent at that point. Based on uncertainty density distribution and the monotonically decreasing search effectiveness model, we addressed and formulated the problem of optimally deploying the quadcopters to maximize the uncertainty reduction (and hence information gain). Based on the observation we made on similar problem setting used in the literature, we formulate a 'deploy' and 'search' strategy using the concepts of centroidal Voronoi configuration, where the quadcopters get deployed to a centroidal Voronoi configuration, shown to be an optimal configuration maximizing the reduction in uncertainty, and then perform search resulting in a reduction in the uncertainty. The process of optimal 'deployment' and 'search' continues until the average uncertainty over the entire search space is reduced below an arbitrary but fixed value, indicating the targets if presented are detected with acceptable confidence (probability).

One of the very important components in multi-agent search is the search sensor itself and the spatial variation of its effectiveness in performing the search, that is target detection. As we mentioned earlier, we assumed non-uniform effectiveness of the camera within its image frame. We first provided a detailed discussion on the spatial variation of the image quality both in terms of optical resolution and digital quality. We observed that the image quality is higher at the central pixel and degrades away from it. Such a scenario leads us to a non-uniform search effectiveness of the camera. We presented an experimental setup to obtain a sensor effectiveness model for a downward-facing camera using target detection probability. Through a set of the target detection experiments carried out using AuRuco markers and triangular-shaped objects as targets, we obtain a sensor effectiveness model for a downward-facing camera in different scenarios. We also established that an exponential function with two parameters can be used to model the spatial variation of the camera's search effectiveness (that is, the search effectiveness model).

We developed a platform using ROS/Gazebo and Matlab environment for simulation of the proposed multi-quadcopter search strategy in a hybrid centralized-decentralized architecture. The platform developed can be a very useful tool for conducting realistic simulation experiments to validate the proposed search strategy and to make a comparative study of its performance in terms of time required for the search process, with different parameters such as camera search effectiveness functions, sensor range, number of quadcopters, and decide on the right parameters for any given mission.

We provided detailed results of experiments and simulation carried out along with a detailed discussion on the same. First, we present the results of the experiments carried out to obtain the search effectiveness model of the downward-facing camera, which we use in the proposed multi-quadcopter search strategy. Though we used the experimental setup to establish, that in general, an exponential function with two parameters can be as the search effectiveness model of a camera, it can be used to carry out experiments with a specific type of imaging sensor, type of image processing tools, kind of environment in which has to detect the targets, the type of targets that need to be detected, and obtain a suitable search effectiveness model.

We presented representative results of the simulation experiment carried out using the realistic ROS/Matlab simulation platform, both to demonstrate the simulation platform itself and the proposed search strategy. Finally, we provided a detailed account of simulation experiments carried out to evaluate the effect of the number of search quadcopters and the camera effectiveness parameters on the performance of the proposed multi-quadcopter search strategy, using the simulation platform developed in this work.

7.2 Scope for Future work

The simulation platform developed can be used to carry out experiments using physical AR.Drones as the controller used within the simulation environment may be used to control the physical AR.Drones. Also, the simulation environment can be used to conduct a large number of simulation and physical experiments to decide on parameters such as the optimal number of quadcopters, type of cameras used (in terms of their search effectiveness, which may be obtained by using the experimental setup based on that used in this work), for a given search scenario. In this sense, the experimental setup and simulation platform developed are useful beyond the sample results provided in this thesis and will surely help the proposed multi-agent search strategy takes a step forward from theory to experiment and then finally into reality. The future work can focus on these aspects.

Though in at the formulation level, there is no restriction on the kind of quadcopter (or even hexacopter) that can be used, the simulation platform developed is based on Parrot AR.Drone. This restriction may be overcome by suitable modifications in the programs.

Other practical aspects that can be added to the simulation/experimental platform is drone to drone communication, drone to central server communication, incorporation of target detection and the target detection probability into the search (which is now limited to uncertainty density update).

Bibliography

Altshuler, Y., Yanovsky, V., Wagner, I. A. & Bruckstein, A. M. (2008), 'Efficient cooperative search of smart targets using UAV swarms', *Robotica* **26**(4), 551–557.

Baum, M. & Passino, K. (2002), A search-theoretic approach to cooperative control for uninhabited air vehicles, *in* 'AIAA Guidance, Navigation, and Control Conference and Exhibit', Paper No. 2002–4589.

Beard, R. W. & McLain, T. W. (2003), Multiple UAV cooperative search under collision avoidance and limited range communication constraints, *in* 'Proceedings of the IEEE Conference on Decision and Control', Vol. 1, pp. 25–30.

Beard, R. W., McLain, T. W., Goodrich, M. A. & Anderson, E. P. (2002), 'Coordinated target assignment and intercept for unmanned air vehicles', *IEEE transactions on robotics and automation* **18**(6), 911–922.

Benkoski, S. J., Monticino, M. G. & Weisinger, J. R. (1991), 'A survey of the search theory literature', *Naval Research Logistics* **38**(4), 469–494.

Bertuccelli, L. F. & How, J. P. (2005), Robust UAV search for environments with imprecise probability maps, *in* 'Proceedings of the 44th IEEE Conference on Decision and Control', pp. 5680–5685.

Bertuccelli, L. F. & How, J. P. (2006*a*), Search for dynamic targets with uncertain probability maps, *in* 'American Control Conference, 2006', pp. 6–14.

Bertuccelli, L. F. & How, J. P. (2006*b*), Uav search for dynamic targets with uncertain motion models, *in* 'Proceedings of the IEEE Conference on Decision and Control', pp. 5941–5946.

Bourgault, F., Furukawa, T. & Durrant-Whyte, H. F. (2003), Coordinated decentralized search for a lost target in a bayesian world, *in* 'Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems', Vol. 1, pp. 48–53.

Bourgault, F., Furukawa, T. & Durrant-Whyte, H. F. (2004), Decentralized bayesian negotiation for cooperative search, *in* 'Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems', pp. 681–686.

Burgard, W., Moors, M. & Schneider, F. (2002), Colaborative exploration of unknown environment with team of mobile robots, *in* M. Betz, J. Hestzberg, M. Ghallab & M. Pollack, eds, 'Lecture Notes in computer Science 2466: Advances in Plan-Based Control of Robotc Agents', Springer Verlag.

Chen, T. (2003), Digital camera system simulator and applications, PhD thesis, Stanford university.

Choset, H. (2001), 'Coverage for robotics–a survey of recent results', *Annals of mathematics and artificial intelligence* **31**(1-4), 113–126.

Cortes, J., Martinez, S., Karatas, T. & Bullo, F. (2004), 'Coverage control for mobile sensing networks', *IEEE Transactions on Robotics and Automation* **20**(2), 243–255.

Dell, R., J. N. Eagel, G. H. A. M. & Santos, A. G. (1996), 'Using multiple searchers in constrained-path, moving-target search problems', *Naval Research Logistics* **43**, 463–480.

Delle Fave, F. M., Xu, Z., Rogers, A. & Jennings, N. R. (2010), Decentralised coordination of unmanned aerial vehicles for target search using the max-sum algorithm, *in* 'AAMAS 2010 Workshop on Agents in Real Time and Environment', pp. 35–44.

Dirichlet, G. L. (1850), '"ber die reduktion der positiven quadratischen formen mit drei unbestimmten ganzen zahlen"', *Journal fr die Reine und Angewandte Mathematik* **40**, 209–227.

Du, Q., Faber, V. & Gunzburger, M. (1999), 'Centroidal voronoi tessellations: applications and algorithms', *SIAM Review* **41**(4), 637–676.

Engel, J. (2011), Autonomous camera-based navigation of a quadrocopter, PhD thesis, TUM University.

Engel, J., Sturm, J. & Cremers, D. (2012), Camera-based navigation of a low-cost quadrocopter, *in* 'Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems', Vol. 320, pp. 2815–2821.

Enns, D., Bugajski, D. & Pratt, S. (2002), Guidance and control for cooperative search, *in* 'Proceedings of the American Control Conference', pp. 1923–1029.

Finke, J., Passino, K. M. & Sparks, A. (2003), Cooperative control via task load balancing for networked uninhabited autonomous vehicles, *in* 'Proceedings of the IEEE Conference on Decision and Control', Vol. 1, pp. 31–36.

Flint, M., Fernandez-Gaucherand, E. & Polycarpou, M. (2003), Cooperative control for uav's searching risky environments for targets, *in* '42nd IEEE International Conference on Decision and Control', Vol. 4, IEEE, pp. 3567–3572.

Flint, M., Polycarpou, M. M. & Fernandez-Gaucherand, E. (2002), Cooperative control for multiple autonomous UAV's searching for targets, *in* 'Proceedings of the 41st IEEE Conference on Decision and Control', Vol. 3, pp. 2823–2828.

Foote, T. (2013), "tf: The transform library", *in* 'Proceedings of the IEEE International Conference on Technologies for Practical Robot Applications (TePRA)', Open-Source Software workshop, pp. 1–6.

Freda, L. & Oriolo, G. (2007), 'Vision-based interception of a moving target with a nonholonomic mobile robot', *Robotics and Autonomous Systems* **55**(6), 419–432.

Galceran, E., Campos, R., Palomeras, N., Ribas, D., Carreras, M. & Ridao, P. (2015), 'Coverage path planning with real-time replanning and surface reconstruction for inspection of three-dimensional underwater structures using autonomous underwater vehicles', *Journal of Field Robotics* **32**(7), 952–983.

Gan, S. K., Fitch, R. & Sukkarieh, S. (2012), Real-time decentralized search with inter-agent collision avoidance, *in* 'Proceedings of the IEEE International Conference on Robotics and Automation', pp. 504–510.

Gan, S. K. & Sukkarieh, S. (2011), Multi-UAV target search using explicit decentralized gradient-based negotiation, *in* 'Proceedings of the IEEE International Conference on Robotics and Automation', pp. 751–756.

Grundel, D. A. (2005), Searching for a moving target: optimal path planning, *in* 'Networking, Sensing and Control, 2005. Proceedings. 2005 IEEE', pp. 867–872.

Guruprasad, K. R. (2009), Multi-Agent Search using Voronoi Partition, PhD thesis, Department of Aerospace Engineering, Indian Institute of SCience, Bengaluru, India.

Guruprasad, K. R. & Dasgupta, P. (2012*a*), A distributed algorithm for computation of exact voronoi cell in a multi-robotic system, *in* 'Proceedings of the Third International Conference on Emerging Applications of Information Technology', pp. 13–18.

Guruprasad, K. R. & Dasgupta, P. (2012*b*), Distributed voronoi partitioning for multi-robot systems with limited range sensors, *in* 'IEEE/RSJ International Conference on Intelligent Robots and Systems', pp. 3546–3552.

Guruprasad, K. R. & Ghose, D. (2011), 'Automated multi-agent search using centroidal voronoi configuration', *IEEE Transactions on Automation Science and Engineering* **8**(2), 420–423.

Guruprasad, K. R. & Ghose, D. (2013), 'Performance of a class of multi-robot deploy and search strategies based on centroidal voronoi configurations', *International Journal of Systems Science* **44**(4), 680–699.

Hu, J., Xie, L., Lum, K.-Y. & Xu, J. (2013), ‘Multiagent information fusion and cooperative control in target search’, *IEEE Transactions on Control Systems Technology* **21**(4), 1223–1235.

Hu, J., Xie, L., Xu, J. & Xu, Z. (2014), ‘Multi-agent cooperative target search’, *Sensors* **14**(6), 9408–9428.

Ikeda, T., Minamiyama, S., Yasui, S., Ohara, K., Ichikawa, A., Ashizawa, S., Okino, A., Oomichi, T. & Fukuda, T. (2019), ‘Stable camera position control of unmanned aerial vehicle with three-degree-of-freedom manipulator for visual test of bridge inspection’, *Journal of Field Robotics* **36**(7), 1212–1221.

Jin, Y., Liao, Y., Minai, A. A. & Polycarpou, M. M. (2006), ‘Balancing search and target response in cooperative unmanned aerial vehicle (uav) teams’, *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)* **36**(3), 571–587.

Jin, Y., Minai, A. A. & Polycarpou, M. M. (2003), Cooperative real-time search and task allocation in uav teams, *in* ‘Decision and Control, 2003. Proceedings. 42nd IEEE Conference on’, Vol. 1, pp. 7–12.

Khan, A., Yanmaz, E. & Rinner, B. (2014), Information merging in multi-UAV cooperative search, *in* ‘Proceedings of the IEEE International Conference on Robotics and Automation’, pp. 3122–3129.

Khan, A., Yanmaz, E. & Rinner, B. (2015), ‘Information exchange and decision making in micro aerial vehicle networks for cooperative search’, *IEEE Transactions on Control of Network Systems* **2**(4), 335–347.

Kitamura, Y., Chauang, Z. B., Tatsumi, S. & Deen, S. M. (1993), A cooperative search scheme for dynamic problems, *in* ‘Proceedings of the IEEE Conference on System, Man and Cybetnetics’, pp. 120–125.

Kolling, A., Kleiner, A., Lewis, M. & Sycara, K. (2011), Computing and executing strategies for moving target search, *in* ‘Proceedings of the IEEE International Conference on Robotics and Automation’, pp. 4246–4253.

Koopman, B. O. (1980), General operation of search, *in* ‘Search Theory and Applications’, 2nd edn, Pergamon Press.

Kothari, M., Sharma, R., Postlethwaite, I., Beard, R. W. & Pack, D. (2013), ‘Cooperative target-capturing with incomplete target information’, *Journal of Intelligent & Robotic Systems* **72**(3-4), 373–384.

Kuhlman, M. J., Otte, M. W., Sofge, D. & Gupta, S. K. (2017), ‘Multipass target search in natural environments’, *Sensors* **17**(11), 2514.

Lanillos, P., Gan, S. K., Besada-Portas, E., Pajares, G. & Sukkarieh, S. (2014), ‘Multi-uav target search using decentralized gradient-based negotiation with expected observation’, *Information Sciences* **282**, 92–110.

Li, J., Chen, J., Wang, P. & Li, C. (2018), ‘Sensor-oriented path planning for multiregion surveillance with a single lightweight UAV SAR’, *Sensors* **18**(2), 548.

Lida, K. (1992), *Studies on Optimal Search Plan*, Vol. 70, Springer-Verlag.

Lum, C., Rysdyk, R. & Pongpunwattana, A. (2006), Occupancy based map searching using heterogeneous teams of autonomous vehicles, *in* ‘AIAA Guidance, Navigation, and Control Conference and Exhibit’, Paper No. 2006–6196.

Maki, T., Horimoto, H., Kofuji, K. & Ishihara, T. (2019), ‘Tracking a sea turtle by an auv with a multi-beam imaging sonar: Toward robotic observation of marine life,’, *International Journal of Control, Automation, and Systems* **18**, 801–816.

Manathara, J. G., Sujit, P. & Beard, R. W. (2011), ‘Multiple UAV coalitions for a search and prosecute mission’, *Journal of Intelligent & Robotic Systems* **62**(1), 125–158.

Meghjani, M., Manjanna, S. & Dudek, G. (2016), Multi-target search strategies, *in* ‘Proceedings of the IEEE International Symposium on Safety, Security, and Rescue Robotics’, pp. 328–333.

Meng, W., He, Z., Teo, R., Su, R. & Xie, L. (2014), ‘Integrated multi-agent system framework: decentralised search, tasking and tracking’, *IET Control Theory & Applications* **9**(3), 493–502.

Millet, T., Casbeer, D., Mercker, T. & Bishop, J. (2010), Multi-agent decentralized search of a probability map with communication constraints, *in* ‘AIAA Guidance, Navigation, and Control Conference’, Paper No. 2010–8424.

Mirzaei, M., Gordon, B., Rabbath, C.-A. & Sharifi, F. (2010), Cooperative multi-uav search problem with communication delay, *in* ‘AIAA Guidance, Navigation, and Control Conference’, Paper No. 2010–8420.

Mirzaei, M., Sharifi, F., Gordon, B. W., Rabbath, C. A. & Zhang, Y. M. (2011), Cooperative multi-vehicle search and coverage problem in uncertain environments, *in* ‘Proceedings of the IEEE Conference on Decision and Control and European Control Conference’, pp. 4140–4145.

Mohan, M., Silva, C. A., Klauberg, C., Jat, P., Catts, G., Cardil, A., Hudak, A. T. & Dia, M. (2017), ‘Individual tree detection from unmanned aerial vehicle (uav) derived canopy height model in an open canopy mixed conifer forest’, *Computers and Electrical Engineering* **4**, 1–17.

Munoz-Salinas, R. (2013), ‘Aruco: Augmented reality library from the university of cordoba’, http://www.uco.es/investiga/grupos/ava/node/26.

Muoz-Salinas, F. J. R.-R. R. & Medina-Carnicer, R. (2018), ‘Speeded up detection of squared fiducial markers’, *Image and Vision Computing* **76**, 38–47.

Okabe, A. & Suzuki, A. (1997), ‘Locational optimization problems solved through voronoi diagrams’, *European Journal of Operations Research* **98**(3), 445–456.

Orgas, U. Y., Dagci, O. H. & Ozgnuner, U. (2004), Cooperative control of mobile robots for target search, *in* ‘Proceedings of the IEEE International Conference on Mechatronics’, pp. 123–128.

Ousingsawat, J. & Earl, M. G. (2007), Modified lawn-mower search pattern for areas comprised of weighted regions, IEEE, pp. 918–923.

Parker, L. E. (1997), 'Cooperative motion control for multi-target observation', *Proceedings of the IEEE/RSJ Conference on Intelligent Robots and Systems. Innovative Robotics for Real-World Applications. IROS'97* **3**, 1591–1597.

Parker, L. E. (2002), 'Distributed algorithms for multi-robot observation of multiple moving targets', *Autonomous robots* **12**(3), 231–255.

Passino, K. M., Polycarpou, M. M., Jacques, D., Pachter, M., Liu, Y. & Y. Yang, M. Flint, M. B. (2002), Cooperative control for autonomous air vehicles, *in* R. Murphey & P. M. Pardalos, eds, 'Cooperative control and optimization', Vol. 66, Kluwer Academic Publishers, pp. 233–271.

Peng, H., Shen, L. & Zhu, H. (2010), 'Multiple UAV cooperative area search based on distributed model predictive control', *Acta Aeronautica et Astronautica Sinica* **31**(3), 593–601.

Pfister, H. (2003), Cooperative control of autonomous vehicles using fuzzy cognitive maps, *in* '2nd AIAA" Unmanned Unlimited" Conf. and Workshop & Exhibit', Paper No. 2003–6506.

Polycarpou, M. M., Yang, Y. & Passino, K. M. (2001*a*), 'Cooperative control of distributed multi-agent systems', *IEEE Control Systems Magazine* **21**, 1–27.

Polycarpou, M. M., Yang, Y. & Passino, K. M. (2001*b*), A cooperative search framework for distributed agents, *in* 'Proceedings of the IEEE International Symposium on Intelligent Control', pp. 1–6.

Pugliese, L. D. P., Guerriero, F., Zorbas, D. & Razafindralambo, T. (2016), 'Modelling the mobile target covering problem using flying drones', *Optimization Letters* **10**(5), 1021–1052.

Quigley, M., Conley, K., Gerkey, B. P., Faust, J., Foote, T., Leibs, J., Wheeler, R. & Ng, A. Y. (2009), "ROS: an open-source robot operating system", *in* 'ICRA Workshop on Open Source Software'.

Rajnarayan, D. G. & Ghose, D. (2003), Multiple agent team theoretic decision making for searching unknown environment, *in* 'Proceedings of the IEEE Conference on Decision and Control', pp. 2543–2548.

Riehl, J. R., Collins, G. E. & Hespanha, J. P. (2011), 'Cooperative search by uav teams: A model predictive approach using dynamic graphs', *IEEE Transactions on Aerospace and Electronic Systems* **47**(4), 2637–2656.

Ru, C. j., Qi, X. m. & Guan, X. n. (2015), 'Distributed cooperative search control method of multiple uavs for moving target', *International Journal of Aerospace Engineering* **2015**.

Scerri, P., Liao, E., Lai, J., Lewis, M. & Sykara, L. K. (2004), Coordinating very large group of wide area search munitions, *in* D. Grundel, R. Murphey & P. Pardalos, eds, 'Recent Developments in Cooperative Control and Optimization', Kluwer Academic Publishers.

Schlecht, J., Altenburg, K., Ahmed, B. M. & Nygard, K. E. (2003), Decentralized search by unmanned air vehicles using local communication, Vol. 2, pp. 757–762.

Schwager, M., Julian, B. J., Angermann, M. & Rus, D. (2011), 'Eyes in the sky: Decentralized control for the deployment of robotic camera networks', *Proceedings of the IEEE* **99**(9), 1541–1561.

Schwager, M., Julian, B. J. & Rus, D. (2009), Optimal coverage for multiple hovering robots with downward facing cameras, *in* 'Robotics and Automation, 2009. ICRA'09. IEEE International Conference on', pp. 3515–3522.

Sharifi, F., Mahboubi, H., Aghdam, A. G. & Zhang, Y. (2013), A coverage strategy with guaranteed collision avoidance in multi-agent systems using navigation functions, *in* 'AIAA Guidance, Navigation, and Control (GNC) Conference', Paper No. 2003–5109.

Sharifi, F., Mirzaei, M., Zhang, Y. & Gordon, B. W. (2015), 'Cooperative multi-vehicle search and coverage problem in an uncertain environment', *Unmanned systems* **3**(01), 35–47.

Smith, W. J. (2000), *Modern Optical Engineering*, 3 edn, McGraw-Hill Professional.

Spires, S. V. & Goldsmith, S. Y. (1998), *Exhaustive search with mobile robots along space filling curves*, Springer-Verlag, pp. 1–12.

Stone, L. D. (1975), *Theory of Optimal Search*, Academic Press.

Strens, M. J. A. (2004), 'Learning multi-agent search strategies', *Journal of the Society for Artificial Intelligence and the Simulation of Behaviour* **1**(5), 245–258.

Sujit, P. B. (2006), Search strategies for muliple autonomous agents, PhD thesis, Department of Aerospace Engineering, Indian Institute of Science, Bangalore, India.

Sujit, P. B. & Ghose, D. (2004), 'Search using multiple uavs with flight time constraints', *IEEE Transactions on Aerospace and Electronic Systems* **40**(2), 491–509.

Sujit, P. B. & Ghose, D. (2005), *in* 'Proceedinsg of the American Control Conference', pp. 2995–3000.

Sujit, P. & Beard, R. (2008), 'Multiple uav exploration of an unknown region', *Annals of Mathematics and Artificial Intelligence* **52**(2-4), 335–366.

Sujit, P. & Ghose, D. (2009), 'Negotiation schemes for multi-agent cooperative search', *Proceedings of the Institution of Mechanical Engineers, Part G: Journal of Aerospace Engineering* **223**(6), 791–813.

Sukumar, V., Hess, H. L., Noren, K. V., Donohoe, G. & Ay, S. (2008), Imaging system MTF-modeling with modulation functions, *in* 'in Proceedings of 34th Annual IEEE Conference of Industrial Electronics (IECON)'.

Sun, J., Li, B., Jiang, Y. & Wen, C. (2016), 'A camera-based target detection and positioning uav system for search and rescue (sar) purposes', *Sensors* **16**, 1778.

Trodden, P. & Richards, A. (2008), Multi-vehicle cooperative search using distributed model predictive control, *in* 'AIAA Guidance, Navigation and Control Conference and Exhibit', Paper No. 2008–7138.

Vincent, P. & Rubin, I. (2004), "a framework and analysis fo cooperative search using UAV swarms", *in* 'Proceedings fo the ACM Symposium on Applied Computing', pp. 79–86.

Voronoi, G. (1907), '"nouvelles applications des paramtres continus la thorie des formes quadratiques"', *Journal fr die Reine und Angewandte Mathematik* **133**.

Waharte, S., Trigoni, N. & Julier, S. (2009), Coordinated search with a swarm of uavs, *in* '2009 6th IEEE annual communications society conference on sensor, mesh and ad hoc communications and networks workshops', IEEE, pp. 1–3.

Yang, F., Ji, X., Yang, C., Li, J. & Li, B. (2017), Cooperative search of UAV swarm based on improved ant colony algorithm in uncertain environment, *in* 'Proceedings of the IEEE International Conference on Unmanned Systems', pp. 231–236.

Yang, Y. (2005), Cooperative search by uninhabited air vehicles in dynamic environment, PhD thesis, University of Cincinnati.

Yang, Y., Minai, A. A. & Polycarpou, M. M. (2002*a*), Analysis of opportunistic method for cooperative search by mobile agents, *in* 'Proceedings of the IEEE Conference on Decision and Control', pp. 576–577.

Yang, Y., Minai, A. A. & Polycarpou, M. M. (2002*b*), Decentralized cooperative search in UAVs using opportunistic learning, *in* 'Proceedings of the AIAA Guidance, Navigation and Control Conference', Paper No. 2002–4590.

Yang, Y., Minai, A. A. & Polycarpou, M. M. (2004), Decentralized cooperative search by networked UAVs in an uncertain environment, *in* 'Proceedings of the American Control Conference', Vol. 6, pp. 5558–5563.

Yang, Y., Polycarpou, M. M. & Minai, A. A. (2007), 'Multi-UAV cooperative search using an opportunistic learning method', *Journal of Dynamic Systems, Measurement, and Control* **129**(5), 716–728.

York, G. & Pack, D. J. (2012), 'Ground target detection using cooperative unmanned aerial systems', *Journal of Intelligent & Robotic Systems* **65**(1-4), 473–478.

Zhang, M., Song, J., Huang, L. & Zhang, C. (2017), 'Distributed cooperative search with collision avoidance for a team of unmanned aerial vehicles using gradient optimization', *Journal of Aerospace Engineering* **30**(1), 04016064.

Zhou, Y., Rui, T., Li, Y. & Zuo, X. (2019), 'A uav patrol system using panoramic stitching and object detection', *Computers and Electrical Engineering* **30**, 1–9.

Table 7.1: List of Publications based on PhD Research Work

Sl.No	Title of the paper	Authors	Name of the Journal	Month, Year of publication
1	Effectiveness of a camera as a UAV mounted search sensor for target detection: an Experimental investigation	Jeane Marina D'Souza, K R Guruprasad,	International Journal of Control, Automation and Systems	November 2020 (Accpeted)
2	A realistic simulation platform for multi-quadcopter search using downward facing cameras	Jeane Marina D'Souza, K R Guruprasad, Akshar Padman	Computers & Electrical Engineering, Special Issue on Recent trends in flocking control and communication for unmanned vehicles	March 2019
3	Optimal Deployment of Camera Mounted UAVs Performing Search	Jeane Marina D'Souza, Siddhartha Suresh Rao K R Guruprasad	International Journal of Engineering and Technology April 2018, Volume Number 7, No 2.21 (2018) Special Issue 21, Pages: 161-165	April 2018

BIODATA

1. **Name** Jeane Marina D' Souza
2. **Father's Name** John Alex D' Silva
3. **Date of Birth** 11/7/1981
4. **Nationality** Indian

5. Address:

402, Beacon Apartments
Kulai Post, Surathkal
Karnataka (State)-575019, India.

6. Mobile Number: +91-7975733393

7. E-mail id.: jeanemdsilva@gmail.com

8. Academic Credentials:

Qualification	University	Institution	CGPA	Year of Passing
PhD (*Pursuing*)	NITK	National Institute of Technology Karnataka (NITK), Surathkal	8.0 / 10	2020
M.Tech in Control Systems	Manipal University	MIT, Manipal	8.26 / 10	2013
B.E in Electrical & Electronics Engineering	Manipal Academy of Higher Education	MIT, Manipal	2.3 / 4	2005

8. Professional Credentials: 6 years of academic teaching experience for UG and PG courses at Department of Instrumentation and Control Engineering, MIT, Manipal University.

I declare that the above information is true and correct to best of my knowledge.

Jeane Marina D' Souza