Decentralized Approach to SLAM using Computationally Limited Robots

by

Vishnu Sudheer Menon

A Thesis

Submitted to the Faculty

of the

WORCESTER POLYTECHNIC INSTITUTE

In partial fulfillment of the requirements for the

Degree of Master of Science

in

Robotics Engineering

by

May 2017

APPROVED:

Professor Carlo Pinciroli, Thesis Advisor

Professor Eugene Eberbach, Thesis Co-Advisor

Professor Michael Gennert, Thesis Committee Member

Abstract

Simultaneous localization and mapping (SLAM) is a challenging and vital problem in robotics. It is important in tasks such as disaster response, deep-sea and cave exploration, in which robots must construct a map of an unknown terrain, and at the same time localize themselves within the map. The issue with single-robot SLAM is the relatively high rate of failure in a realistic application, as well as the time and energy cost. In this work, we propose a new approach to decentralized multi-robot SLAM which uses a robot swarm to map the environment. This system is capable of mapping an environment without human assistance and without the need for any additional infrastructure. We assume that 1) no robot possesses sufficient memory to store the entire map of the environment, 2) the communication range of the robots is limited, and 3)there is no infrastructure present in the environment to assist the robot in communicating with others. To cope with these limitations, the swarm system is designed to work as an independent entity. The swarm can deploy new robots towards the region that is yet to be explored, coordinate the communication between the robots by using itself as the communication network and replace any malfunctioning robots. The proposed method proves to be a reliable and robust exploration algorithm. It is shown to be a self-growing mapping network that is able to coordinate among numerous robots and replace any broken robots hence reducing the chance of system failure.

Acknowledgements

I would like to express my gratitude to my advisors Prof. Carlo Pinciroli and Prof. Eugene Eberbach for their guidance, direction, criticism, and enthusiasm during the course of this work.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Robots have come a long way from being fictional entities in films to being sent for planetary exploration. These robots are given tasks of great difficulty which would be near-impossible for humans to perform. A good example of this is the use of robots to assess radiation levels within the broken reactors of the Fukushima Daiichi nuclear power plant. The radiation levels are too high for humans to withstand, but by using robots, such difficulties are overcome.

Advancement in robotics have led to the development of Multi-Robot Systems (MRS). These systems involve multiple robots performing either a single task or multiple tasks as specified by the user. The advantages that such a system can bring about is evident as the risk of the system failing is reduced and the time for task execution is also reduced resulting in a significant improvement of performance.

MRS are classified into centralized and decentralized systems. A centralized system consists of multiple robots coordinated by a central controller. The central hub controls communication and is the decision making power of the system. On the other hand, a decentralized system gives this decision making power to individual robots. Every robot is tasked with a local goal and involved in local interactions with the environment resulting in a desired emergent behavior of the robotic swarm.

Advantages of robotic swarms are well researched and documented. Brambilla et al. (2013) describes a robotic swarm to be autonomous, able to modify the environment, capable of local sensing and communication, and with no access to global knowledge. According to Şahin (2004), the lack of a centralized coordination system in social insects inspired the development of robotic swarms. He claims that a robotic swarm should be robust, flexible and scalable. The robustness of a swarm

arises from the fact that every robot is simple and dispensable. This is comparable to how a colony of ants does not depend upon the survival of a single ant. The flexibility of the swarm is the ability of an individual robot to switch its behavior depending upon the task it is executing and to efficiently perform tasks even under changing swarm sizes.

Such advantages make the robotic swarm a reliable and efficient solution for numerous problems, such as large-scale coverage and exploration (Sharma and Tiwari, 2016), search and rescue (Tair et al., 2015), agriculture and mapping (Dirafzoon and Lobaton, 2013).

Simultaneous Localization and Mapping (SLAM) is the problem of a mobile robot moving through an area without map information a priori while simultaneously keeping track its location within it. The robot acquires observations of the environment using limited range-finder sensors and estimates its pose according to odometry measurements. The presence of noise in the sensors affects measurements and causes the map to be inaccurate. Hence, the aim of a robot performing SLAM is to create an accurate map given noisy sensor inputs (Fox et al., 1999).

Multi Robot Simultaneous Localization and Mapping (MR-SLAM) expands the application of SLAM to multiple robots. Using multiple robots means that the mapping task can be completed much faster due to parallel coverage at any instant of time. The failure of a single robot will not hamper the completion of the task, hence the system is more robust (Birk and Carpin, 2006).

Using robots to perform exploratory mapping in unknown and risky terrain is a safer alternative to having humans perform the same task. For example, exploration of unmapped mines is a dangerous yet important task Thrun et al. (2004b). Such conditions also pose a risk of damaging the robot, hence using a single robot to map these terrains is inefficient and prone to failure. In such conditions, MRS would prove to be an feasible solution because the risk factor to humans is non-existent and the risk of the system failure is reduced.

In this work, we study a decentralized solution to the problem of mapping large areas with robots that could not individually complete the task, due to memory limitations and high risk of failure.

1.1 Problem Statement

The goal of this work is to map an unknown region using a swarm of robots. We assume that the region to map has numerous obstacles distributed in a uniform manner and that the region is inaccessible to external localization or communication networks. This is to simulate a location that is inaccessible to humans or would be dangerous for human exploration. The region is also free of any infrastructure that assists in map building. Figure 1.1 gives a pictorial representation of the swarm system. The system uses the robots as a transmission network and grows towards the region that has not been explored.

The swarm system used in this work uses a decentralized strategy for communication and deployment. Every robot in the swarm has memory and communication range limitations. These constraints mean that a single robot is incapable of developing the complete map.

Each robot is equipped with a LIDAR for developing grid maps and a communication sensor for interacting with the swarm. Each robot maps the environment until its memory gets completely consumed and then transfers it to the user. On completion of the mapping task, the robot becomes a communication link between the user and new robot that has been assigned to continue the map task.

1.2 Outline

In Chapter 2, we introduce the formulation of the SLAM problem, describe the different types of single robot SLAM algorithms, and explain the need for Multi-Robot SLAM. In Chapter 3, we introduce the approach used in this work for performing Multi-Robot SLAM. It explains the different behaviors a robot can display depending on external factors and the numerous algorithms that are present in the system. The focus of this work is to develop a robust swarm system for mapping unknown regions, and an algorithm responsible for progressively deploying robots to different parts of the environment to map and maintain connectivity with them. In Chapter 4, we discuss the metrics and the results in various experiments and the analysis obtained. In Chapter 5, we discusses the practical use and advantages of this system and also future improvements that can be made.

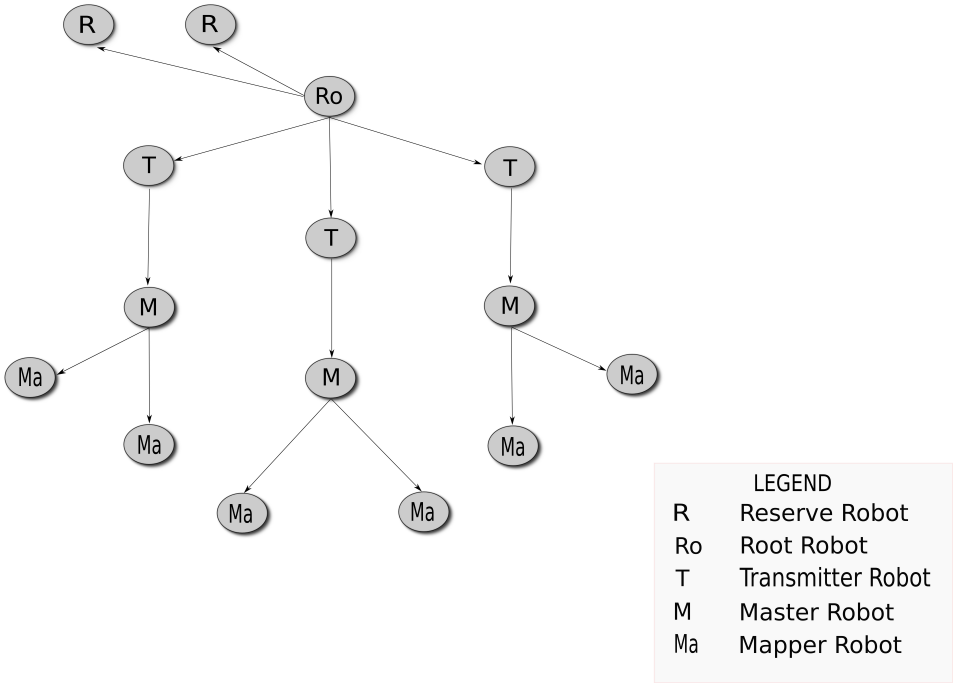

Figure 1.1: Layout of the Swarm System.

Chapter 2

Background

In Chapter 1, we discussed the need for using MRS for solving the SLAM problem. In this chapter, we give a background study about the concepts related to this work. In Section 2.1, we explore the SLAM problem and in section 2.2, we introduce the different types of algorithms in single robot SLAM. In Section 2.3, we compare the state of the art centralized and decentralized strategies in Multi Robot SLAM.

2.1 Introduction to SLAM

SLAM is the ability of a robot to develop an accurate map of the environment and localize itself within that explored map(Thrun et al., 2002). Both the environment and the vehicle position in the map is unknown. Odometers or control inputs are used for estimating motion and landmarks will be sensed and mapped by the robot. This section introduces the SLAM problem along with important concepts necessary to develop an understanding of SLAM problem.

The pose of the robot in a two dimensional map is expressed by its position in **x** and **y** coordinate axis and its orientation with respect to an axis.

$$x = (x_i, y_i, \theta_i) \tag{2.1}$$

The following definition expresses the pose of the robot over multiple time steps.

$$x_{1:t} = \{x_1, x_2, ..., x_t\} \tag{2.2}$$

Here x_1 corresponds to the pose of the robot at time $t = 1$. A probabilistic

formulation expresses the probability of the transition of the robot from one pose to another through the application of a control action.

$$p(x_t|u_t, x_{t-1}) \tag{2.3}$$

The expression 2.3 accounts for the uncertainties such as drift in motion and noise in sensor reading.This probabilistic kinematic model acts as the state transition model for the robot. In equation 2.3, the terms x_t and x_{t-1} represent the robot pose. The term u_t corresponds to the robot motion command asserted from time period $(t-1, t]$.

The observations made by robot are expressed as:

$$z_{1:t} = \{z_1, z_2, ..., z_t\} \tag{2.4}$$

The term z_t corresponds to the observations made by the robot using its on board sensors such as camera or LIDAR which are used for developing the map $\mathbf{m}$. The sensors have a certain amount of noise in their measurements. The Probabilistic formulation showed in expression 2.5 is used for modeling this noise.

$$p(z_t|m, x_t) \tag{2.5}$$

Given the trajectory and a set of observations (Dissanayake et al., 2001), the SLAM problem for a single robot is to find the posterior over the newly developed map. The probabilistic expression of this statement is given as

$$p(m, x_{1:t}|z_{1:t}, u_{1:t}, x_0) \tag{2.6}$$

The expression 2.6 shows that the task of estimating the map is coupled with the task of estimating the trajectory. The SLAM problem can be considered to be of 2 types, full SLAM (Grisetti et al., 2010) and online SLAM (Grisetti et al., 2007). The full SLAM problem requires complete estimation of the robot's trajectory while the online SLAM problem requires estimation of only the posterior pose.

When a complete accurate trajectory is known, the SLAM problem is reduced to a mapping problem. The mapping problem is defined as estimating the map of

the region given the trajectory and observations.

$$p(m|x_{1:t}, z_{1:t}) \tag{2.7}$$

If an accurate map of the region is given, the SLAM problem becomes a localization problem. The localization problem is defined as estimating the posterior over the trajectory of the robot given the map, control inputs and observation.

$$p(x_{1:t}|m, z_{1:t}, u_{1:t}, x_0) \tag{2.8}$$

2.2 Single Robot SLAM

Through the use of a probabilistic framework for motion and perception, a single robot can map an unknown terrain. This section will give a deeper understanding about different maps that can be developed using SLAM and the different types of algorithms in SLAM (Saeedi et al., 2016).

2.2.1 Grid Maps

A matrix of cells which represent the environment is the easiest solution for two dimensional mapping. The resolution of mapping decides the amount of environment detail that can be captured in each cell. Each cell holds a value which represents the probability of that cell being occupied (Elfes, 1990). Three dimensional data can also be represented in grids, as shown by Hornung et al. (2013), the data is represented as point clouds or voxel grids.

2.2.2 Feature maps

Feature maps utilize distinctive landmarks to represent the map of a region. Each landmark is accompanied by a descriptor which is used for recognizing the landmark after a period of time (Walter et al., 2007). The advantage of using feature map is shown by Williams et al. (2009), through the application of loop closure, which is a method to reduce the cumulative error of the posterior by re-observing a previously observed landmark.

2.2.3 Variants of Single robot SLAM

Algorithms in SLAM can be categorized based on the differences in map representation and data processing techniques.

Feature based SLAM

The algorithms under this category extract features from the environment and maintain the list of features as a map. Feature based SLAM is limited to environments with distinctive features. Features are objects in the environment that are distinguishable and identifiable from different point of views. For example, Smith et al. (2006) used straight lines as features for an Extended Kalman Filter(EKF) and Nieto et al. (2003) used steel poles and trees as features in two separate experiments. Relative motion is acquired by comparing features in subsequent frames. The relative motion of the features is used for estimating the motion of the robot(Mouragnon et al., 2006).

View based SLAM

View based SLAM methods employ a range and bearing sensor that returns the position of the obstacles within a given range. The map is populated with probability values that correspond to the certainty of detecting an obstacle. View based SLAM can be used in places which are void of any features such as mines Baker et al. (2004) and underwater locations Fairfield et al. (2007). View based SLAM differs from feature based SLAM as the former does not extract any feature from the environments. Grisettiyz et al. (2005) performed View based SLAM to develop two dimensional grid maps using Sick PLS range finder and estimated relative motion by comparing whole scans between each time step.

Appearance based SLAM

Appearance based SLAM is used to solve the loop-closure problem. It is used in conjunction with either Feature based SLAM or View based SLAM. Hess et al. (2016) employed loop closure in View based SLAM system by identifying similar scans while revisiting places. In Feature based SLAM systems like Mur-Artal et al. (2015), features were extracted using a camera and then compared with each other for finding loop closing locations.

Filtering based SLAM

Filtering based SLAM techniques include variants of Kalman filter, Information filter and Particle filter based algorithms. These algorithms are derived from Bayes Filter with the goal of estimating the pose of the robot given sensor information.

Kalman Filter based systems assume that the state transitions and the measurement models are linear with added Gaussian noise. A variant of Kalman Filter for non-linear systems is the Extended Kalman Filter(EKF). EKF is a prediction filter that linearizes the non linear systems and estimates the posterior given the observation and control inputs. EKF is a popular filter in SLAM and has been researched and implemented in variety of scenarios. Ahn et al. (2008) used EKF SLAM in an indoor environment, Salvi et al. (2008) performed it for mapping underwater terrain and it also been used in conjunction with other SLAM methods like in (Montemerlo et al., 2002).

Information Filters (IF) are used to overcome disadvantages of EKF in high dimensional maps, where the EKF SLAM tends to become slow and require higher computational power (Aulinas et al., 2008). The IF is implemented by propagating the inverse of the state error covariance matrix which is known as the information matrix. So if the system has high errors then IF will have very sparse information matrix. The landmarks with high confidence will only populate the information matrix. Thrun et al. (2004a) have further reduced the density of the information matrix by performing *sparsification* on the information matrix.

Particle Filters(PF) are used for localization in SLAM. They represent any given distribution by samples drawn from that distribution. Montemerlo et al. (2002) used Particle filters with EKF in a an algorithm called FastSLAM. This method proves to be more scalable than a lone EKF SLAM because it considers each landmark to be independent of another. Hence, each landmark is modeled using a single 2x2 Gaussian and thus reducing the complexity of manipulating a high dimensional covariance matrix. These particles represent the pose of the robot in its own map.Each particle has a weight attached to it, that represents the belief of that particle about its pose and its map. To reduce the space complexity of FastSLAM, Eliazar and Parr (2003) introduced a method to have only a single map for all the particles.

Smoothing based SLAM

Smoothing based SLAM is a full SLAM problem where the entire trajectory is estimated using the observation and motion constraints. The SLAM problem is converted to a least squares optimization problem where the maximum a posteriori (MAP) over the trajectory is estimated (Grisetti et al., 2010). This algorithm is divided into two parts, the front-end and back-end. The front-end of the algorithm acquires the sensor information and converts it into observation and motion constraints. The front-end is also responsible for identifying observed landmarks and identifying loop closure locations. The back-end of the algorithm performs optimization on the observation and motion constraints. It is done using methods such as QR factorization(Kim et al., 2010), Cholskey factorization(Dellaert and Kaess, 2006), or Gauss-Newton factorization(Grisetti et al., 2010). Multiple iterations of the optimization process is performed until the error converges to a minimal value.

2.3 Multi-Robot SLAM

The previous sections explained the various types of algorithms in SLAM. These same algorithms along with a coordination strategy is used for performing Multi-Robot SLAM. The coordination strategy is the approach used by systems to coordinate the task of map building through multiple robots. A single server can be used for collecting and merging the map of multiple robots or the robots can share their map between themselves. These strategies are categorized as centralized and decentralized approaches respectively. A detail study of these approaches is presented in the sections 2.3.1 and 2.3.2.

2.3.1 Centralized Coordination Strategy

A Centralized Coordination Strategy involves the robots sending all their information to a central entity. This central entity fuses all the map information being sent and coordinates the robots in their exploration. Morrison et al. (2016) developed a centralized multi-robot SLAM system called MOARSLAM which uses a central server for storing the maps developed using autonomous robots. Each robot is connected to the server to receive updates regarding locations it should explore and

if the connection fails the robot will continue to map and wait until the connection is available. In Simmons et al. (2000), robots perform maximum likelihood estimation to develop maps using odometry and sensor information. These maps are transmitted to a central server to develop a global map. A similar strategy is followed in (Gil et al., 2010) where visual descriptors of landmarks along with odometry information are transmitted by the robots to a central server for developing the map.

The drawback of a *Central Coordination Strategy* is that the system is dependent on the central unit. The central unit is expected to communicate and coordinate with all the robots in the system and this leads to scalability issues. Moreover, failure of the central unit can cripple the entire system.

2.3.2 Decentralized Coordination Strategy

Decentralized Coordination Strategies enable the use of large number of robots to solve the given problem. However, coordination of these robots without a global entity can prove to be difficult. In this section, we discuss the methods used in Multi robot SLAM systems for sharing data among robots and division of mapping task among the robots.

Communication

In Decentralized SLAM systems, robots develop the global map by sharing the map information with other robots. Howard (2006) introduces a method to have individual robots meet each other and share their sensor data. This data is then integrated with the robots' own information and its map is updated. Fox et al. (2006) implements a variant of this approach by using clusters of robots to map and these robots split into smaller groups in the presence of divergent paths. Carlone et al. (2010) introduces uncertainties in transmission and transmits a preprocessed data of smaller size. Using this strategy, Arturo et al. (2011) developed a variant of FastSLAM for multi robot system called Independent FastSLAM and (Bresson et al., 2015) developed a system that considers noisy transmission and also modeled a Kalman filter to reduce drift.

The drawback of having robots only share information when they meet is that the global map is only created when multiple robots meet and share their sensor information. This assumption makes the map building ability of the system dependent on

the random meeting of the robots and the chance of robots meeting each other becomes less if the environment is long with numerous tunnels.

Task Division

Task division in decentralized SLAM system is needed to allot new regions for robots to explore and map. In single robot systems, Yamauchi (1997) introduces an autonomous exploration algorithm which moves the robot towards the closest frontier for exploration. Frontiers are portions in a robots map that lies between explored and unexplored areas. To coordinate exploration and mapping in multi-robot systems, Fox et al. (2006) extended Yamauchi (1997)'s work by allotting the closest frontiers to each robot. Colares and Chaimowicz (2016) made the robots select a frontier region to explore based on an information factor that depends on the distance the frontier is from the robot, the information that frontier region holds and distance it is from the other robots.

Decentralized robot swarms consists of large number of robots working with little or no knowledge about the work being done by others. Robots might repeat the work done by other robots and this reduces the efficiency of the system. In SLAM systems, if robots do not meet and inform each other of the locations they have mapped, then those locations will get remapped by other robots or clusters of robots. The robot does not have the information regarding the region explored by the part of the swarm that has not been in contact with it. Hence, it might start to map and already explored region.

Chapter 3

Approach

In chapter 2, the challenges in centralized and decentralized Multi-Robot SLAM were introduced and the limitations of the existing systems were discussed. In this chapter, we present our approach to decentralized deployment and mapping. Section 3.1 gives a high level understanding of the working of the swarm system. Section 3.2 describes the robot state machine and details behavior of the robot in each state. The design of the algorithms are explained in section 3.3.

3.1 Overview

The robot with ID 1 is deployed as a *Root robot* and is placed in front of the location that is to be mapped. Only one *Root robot* exists at any given time (see Section 3.2.1) and the remaining robots are deployed as a *Reserve robots* (see Section 3.2.2). The *Root robot* manages the initial deployment of robots and stores the map developed by the robots. The *Reserve robots* wait in the vicinity of the *Root robot*. They move along the swarm network to switch behaviors of either a *Mapper robot* or *Transmitter robot*. *Mapper robots* map the frontiers as defined in Section 3.2.3. The *Mapper robot* switches to the role of **??** when reaching a terminating criteria (see Section 3.2.4). *Master robot* requests for new robots to further map the environment. *Master robots* are responsible for coordinating the *Mapper robots* by allotting them locations to map . The *Master robot* switches to *Transmitter robot* (see Section 3.2.5) when the requested *Mapper robots* complete mapping.

3.2 State Machine Design for Robots

In this section, we introduce the state machines used for modeling the robots' behavior. The state machine consists of 5 robot behaviors, *Root*, *Reserve*, *Master*, *Mapper* and *Transmitter*. As shown in figure 3.1, each robot undertakes a task that is either requested or alloted to it. The robot then switches to the required behavior to execute the task. A more detailed explanation of each behavior is provided later.

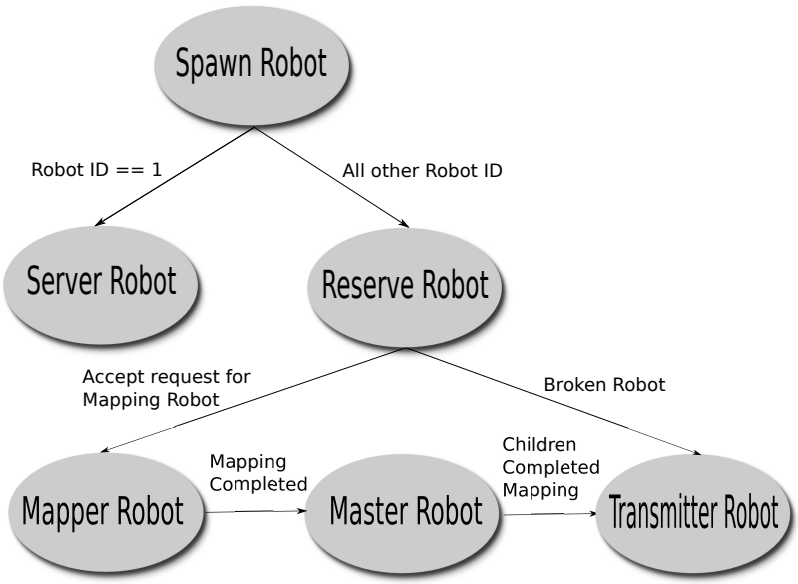

Figure 3.1: Finite State Machine that represents the robot behaviors.

3.2.1 Root Robot Behavior

Robot of ID one is declared as *Root robot*. The responsibility of this behavior is to communicate all the requests regarding additional robots to the *Reserve Robots* and develop the global map of the environment by merging the maps developed by the *Mapper robots*.

3.2.2 Reserve Robot Behavior

Reserve robot behavior are displayed by robots who are not alloted a region to map or requested to become a node in the communication network. textitReserve robot

behavior is the initial behavior exhibited by all robots other the robot assigned to become the textitRoot robot.

Reserve robots can readily switch their behavior depending upon the external request. *Reserve Robots* are used for transforming into either a *Mapper Robot* or a *Transmitter Robot*. The switch into *Mapper Robot* behavior is done on reception of a request for expanding the mapped region and the switch into *Transmitter Robot* is performed to replace a broken robot in the communication network.

3.2.3 Mapper Robot Behavior

Mapper robots are responsible for developing the map of the region alloted to them by the *Master robot*. They perform SLAM using LIDAR to observe the environment and use Odometry Motion Model to estimate their motion. To explore unknown regions, *Reserve robots* switch to *Mapper robots* and start the mapping process. Memory, and communication constraints on the robot limit size of the map it can develop. Memory and communication constraints are modeled by number of grid cells a robot's memory can store and the communication signal strength between *Root* and *Mapper* robots respectively. If the constraints are reached, the *Mapper robots* will identify frontier cells of mapped region and select exploration initialization points for launching new *Mapper robots* robots.

3.2.4 Master Robot Behavior

Master robots are responsible for generating and transmitting a request for new *Mapper robots* in order to increase the mapped area. *Master robots* coordinate with the *Mapper robots* by allotting them regions to map.

When a *Mapper robot* completes its mapping task, it switches to a *Master robots* and launches new *Mapper robots* to map from its frontier regions. Once all the *Mapper robot* completes mapping, the *Master robot* switches to a *Transmitter Robot* in that position.

3.2.5 Transmitter Robot Behavior

The *Transmitter Robot* forms the communication network that transmits information from the frontier regions to the *Root robot*. A *Master robot* becomes a Transmitter robot when all *Mapper robot* that it is coordinating complete their mapping tasks.

Transmitter robot are responsible for acting as the communication link between the *Root Robot* and the *Mapper Robots*. They transmit information such map data, requests for additional robots, and information regarding broken robots. Transmission of information occurs in a chain like fashion, each *Transmitter Robot* broadcasts the message to the robot that was coordinating with it while it was a *Mapper Robot*. This is done to ensure that there won't be a cycle of broadcast data generated between multiple *Transmitter Robot*.

3.3 Algorithms

This section introduces the algorithms that that compose the behaviors presented in section 3.2. Section 3.3.1 explains the algorithm used for receiving and merging maps from different robots. Section 3.3.2 discusses the working of the communication network of the swarm system. Section 3.3.3 discuses the method used by the *Reserve robots* to traverse through the swarm. Section 3.3.4 introduces the method used by the *Master robots* to corrdinate the the *Mapper robots*. Section 3.3.5 discusses the method of identifying and replacing a broken *Transmitter robot*.

3.3.1 Map Development

The Map Development algorithm develops the global map by merging the map developed by the *Mapper robots*. The *Mapper robots* develop maps in their local frame, so it needs to be transformed into the global frame before being merged with the global map. *Transmitter robots* are used for sending the maps to the *Root robot*.

FastSLAM

Mapper robots use FastSLAM algorithm developed by Grisetti et al. (2007) to develop grid maps. An Odometey Motion model is used to estimate the motion of the robot and a LIDAR is used to acquire observational reading. The LIDAR gives the robot a two dimensional view of its surroundings up to a range of 150 cms.

FastSLAM is the application of Rao-Blackwellized Particle Filter to the SLAM problem. Particle filter is used to estimate the pose of the robot(x_k) and also map(m_k). This estimation is based on the current control input(u_k) and the

measurements(z_k) acquired through LIDAR. The combined posterior is given by:

$$P(m_k, x_k | u_k, z_k) \tag{3.1}$$

The principle behind the working of FastSLAM is to use Bayes rule to split the joint posterior into two separate probability distributions. This causes the observations to be independent of trajectory of the robot.The application of Bayes rules simplifies the problem into:

$$P(m_k, x_{0:k} | u_{0:k}, z_{0:k}) = P(m_k | x_{0:k}, z_{0:k}) P(x_{0:k} | z_{0:k}, u_{0:k}) \tag{3.2}$$

The posterior of the trajectory is estimated recursively by a particle filter. Each particle represents a pose of the robot in the world and is equipped with its own map. The particles populates its map using the LIDAR reading. Each particle is weighted based on how accurate it is on estimating the robots pose. Based on the weight factor, particles are resampled and the map of the particle with the highest weight is considered to be the robots map.

Transmission of Map Data

The robot transmits the developed map when it switches to the *Transmitter robot* behavior.The map is transmitted to the *Root robot* in two parts; the first part contains the relative location of the *Mapper robots* in the global frame and the second part contains the raw map data in the *Mapper robot's* local frame.

Transformation of Map Data

As explained in Section 3.3.1, the *Root robot* receives the map data in 2 parts. The first part of the map data forms the transformation equation shown in equation 3.3.

$$\begin{bmatrix} x_{global} \\ y_{global} \\ 1 \end{bmatrix} = \begin{bmatrix} cos(\theta) & -sin(\theta) & x_{relative} \\ sin(\theta) & cos(\theta) & y_{relative} \\ 0 & 0 & 1 \end{bmatrix} \times \begin{bmatrix} x_{local} \\ y_{local} \\ 1 \end{bmatrix} \tag{3.3}$$

The second part of the map data consists of the grid map developed in the *Mapper robot's* local frame. The *Root robot* uses equation 3.3 to transform the received map data into the global frame and adds it to the global map.

Frontier Selection for Deploying Mapping Robots

Once mapping completed, every *Mapper robot* is responsible for identifying points to launch new robots. Initially, each robot identifies the frontier region of its developed map. The frontier region is the border between the mapped and the unknown part of the map.

Figure 3.2 shows the *Master robot*(M) coordinating the *Mapper robots*(Ma). *Master robot* allots each *Mapper robots* a region to map. When they have completed their mapping tasks, the *Master robot* informs them about the locations of the other *Mapper robots*. The *Mapper robots* use this information to select the to select the portion of its frontier that is away from the other robots. As shown in figure 3.2, the **Selected Frontier Region** is the part of the frontier from which new robots will be launched. This region above the line connecting the each *Mapper robots* is selected for launching new robots.

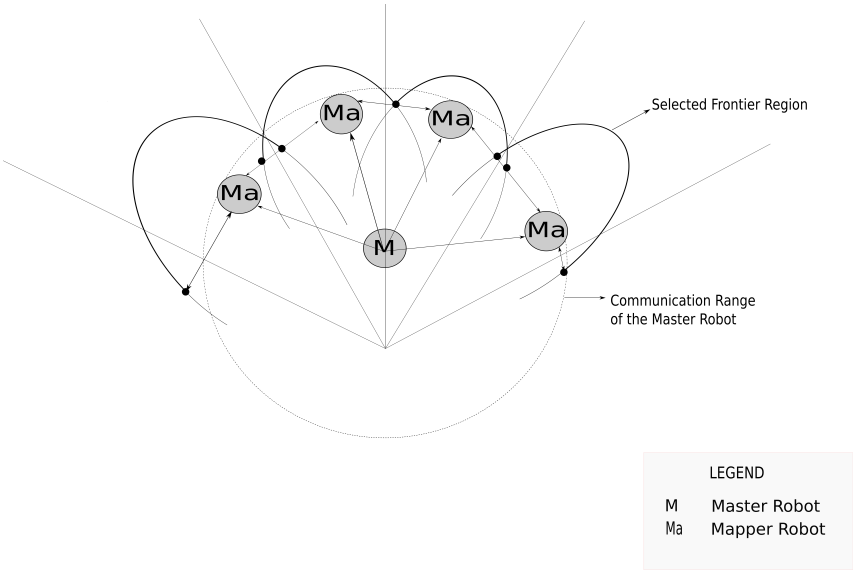

Figure 3.2: Identification of frontiers to launch new robots.

Before the allotment of a frontier region to a *Mapper robot*, the *Master robot* checks whether any other robots are present in that frontier region. The presence of a robot indicates that the frontier region has already been mapped. This check is performed by listening for broadcast noise from that particular frontier region. If the frontier region has a robot mapping it or a group or robots forming the communication network, the broadcast noise will be evident and the *Master robot* will restrain from launching robots to map that frontier region.

3.3.2 Communication Network

The Communication Network is used for the transmission of data through the swarm. An array of *Transmitter robot* forms the communication network.

Range and Bearing Sensor

The Range and Bearing Sensor are used for communication by the robots. It is a line of sight based sensor and robots communicate if they are within each others communication range. Through this mechanism the robots share information related to its position or request for additional robots.

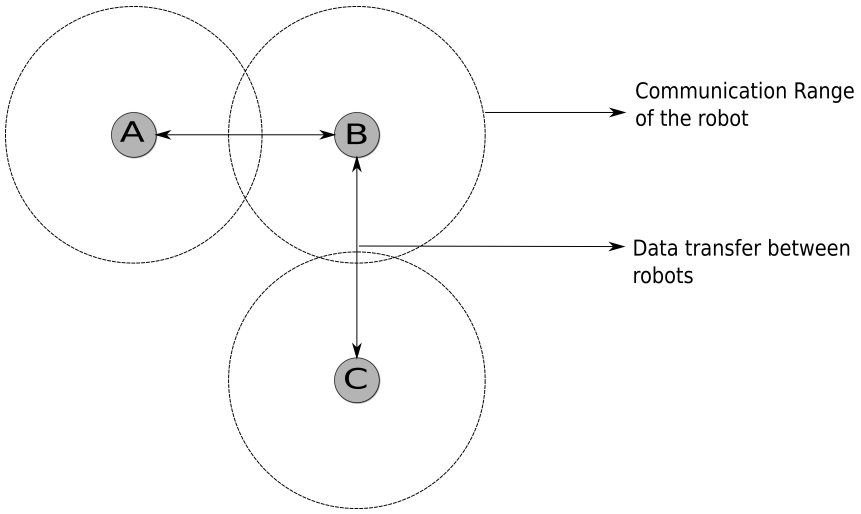

Figure 3.3: Data transmission between the robots using Range and Bearing sensor .

Figure 3.3 shows the working of the Range and Bearing sensor. The robot pairs $\{A, B\}$ and $\{C, B\}$ have an overlapping communication range and will be able to exchange data. The robot pair $\{A, C\}$ wont be able to communicate with each other. Instead they form a communication network with B for exchanging information with each other.

Transmitter Robot Chain

Transmitter robots form the communication network and are responsible for the passage of information along the swarm. Communication is performed in a chain like manner, each *Transmitter robot* communicates with the robot that coordinated it as *Master robot*. Each *Transmitter robot* have limited memory and it informs

the amount of memory available to the robots that want to transmit to it. If a robot wants to transmit more information, it will wait until more memory in the *Transmitter robot* becomes available.

Hierarchy of Task Data

Due to the large amount of information flow in the communication network, each request is tagged by a unique ID. This ID ensures that robots in different behaviors understand each other. Table 3.1 displays the different Transmission Request ID's used for labeling the data transmitted between robots.

Request ID	**Transmission between Robots**	**Content of the message**
1	Master robot/Transmitter robot and Reserve robot	Request for additional robots
2	Root robot/Transmitter robot and Reserve robot	Rejecting the Reserve robot as either a Mapper robot or new Transmitter robot
3	Master robot/Transmitter robot and Reserve robot	Accepting the Reserve robot as either a Mapper robot or new Transmitter robot
5	Mapper robot and Transmitter robot/Root robot	Map data containing the position of the Mapper robot in global frame.
6	Mapper robot and Transmitter robot/Root robot	Map data containing grid cell values in the Mapper robot local frame.
7	Transmitter robot / Root robot and Reserve robot	Reserve robot request to initalize its position in the global frame.
11	Root robot and Reserve robot	Request for additional robots

12	Root robot and Reserve robot	Root robot accepts Reserve robot as a Mapper robot.
13	Root robot/Master robot/Transmitter robot and Reserve robot	Reserve robot alerting of its arrival.
14	Root robot/Master robot and Mapper robot	Coordinating the movement of the Mapper robot.
15	Root robot/Master robot and Mapper robot	Mapper robot informs it has completed the mapping task .
16	Root robot/Master robot and Mapper robot	Information regarding Frontier allocation .
17	Root robot/Master robot and Mapper robot	Information regarding Frontier allocation.
18	Mapper robot to other Mapper robots	Informing other Mapper robots that it is mapping in that region .
20	Transmitter robot to other Transmitter robots	Informing its children how much memory it has remaining .
21	Transmitter robot to other Transmitter robots	Checking if its Children are alive.
22	Transmitter robot and Reserve robot	Accepting the reserve robot as a replacement Transmitter robot .

Table 3.1: Unique Transmission Request ID

Transmission Container

Robots use the Range and bearing sensor to listen and reply to numerous other robots. The Robots are restricted to transmit only limited information in each instance. Hence, it uses the transmission container to store the data it needs to

transmit and the data is transmitted based its importance level calculated using equation 3.4.

$$ImportanceFactor = w_1 * L + w_2 * ID + w_3 * T \tag{3.4}$$

Equation 3.4 is used to prioritize the yet to be transmitted messages. Higher the *Importance Factor* for a message, the sooner it gets transmitted. The terms w_1 w_2, and w_3 are weighting factors which can be varied appropriately. **L** is the length of the queue stores the message, **ID** is the Message ID given to that particular request and **T** is the time it have been waiting in the queue.

3.3.3 Robot Deployment Algorithm

Robot Deployment Algorithm is used by *Reserve robots* to traverse the swarm. *Reserve robots* answers the request for additional robots and uses this algorithm to navigate to the robot that is generating the request.

Reserve Robot Deployment

Reserve robots are required to answer the requests for additional *Mapper robots* or a robot to replace a broken *Transmitter robot* . As the swarm grows, *Reserve robots* needs to travel longer distance to answer the requests. The Reserve Robot Deployment algorithm will guide the *Reserve robots* towards their destination.

Section 3.3.2 explained the method of transmission of information using *Transmitter robots*. The *Reserve robots* use the same transmission chain for traveling to the robot generating the request.

Figure 3.4 shows the *Transmitter robots* informing the *Reserve robot* about a request. *Reserve robot* can either be located next to the *Root robot* or anywhere else along the swarm. On hearing a request being broadcasted by *Transmitter robots*, the *Reserve robot* will traverse along the chain of *Transmitter robots* that are repeating the request until it reaches the source of the request.

Figure 3.5 shows the *Reserve robot* reaching the *Master robot* after following the *Transmitter robot* chain. The *Master robot* is alerted by the *Reserve robot* of its arrival and it decides whether the *Reserve robot* can be given a mapping task. If a mapping task is still available, as shown in figure 3.6 the *Reserve robot* switches to a *Mapper robot* and performs the task. If it is not available, the *Reserve robot* will

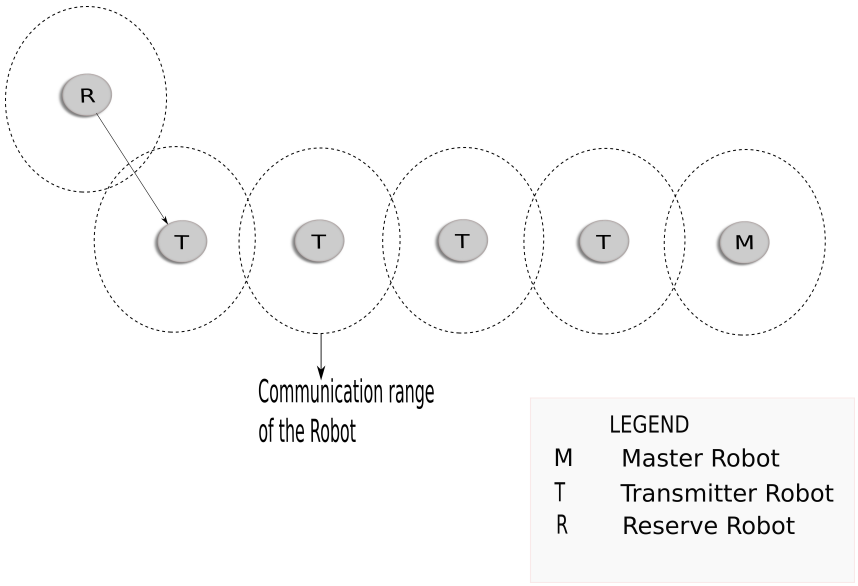

Figure 3.4: Master robot request for additional robots .

wait for a new request.

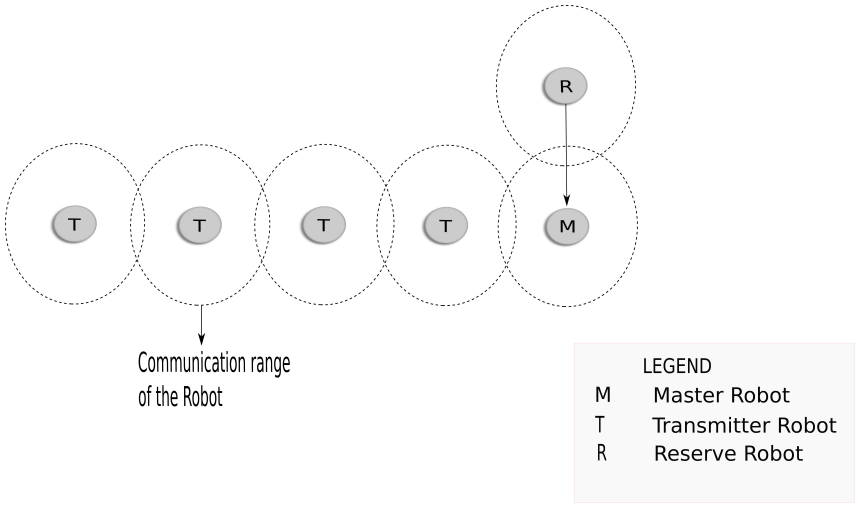

Figure 3.5: Master robot accepts the Reserve Robot.

3.3.4 Autonomous Robot Movement Algorithm

Master robots use the Autonomous Robot Movement Algorithm to coordinate the movement of the *Mapper robots*. The *Master robots* allocates a region for the *Mapper robots* to map. This region can either be completely free or filled with obstacles.

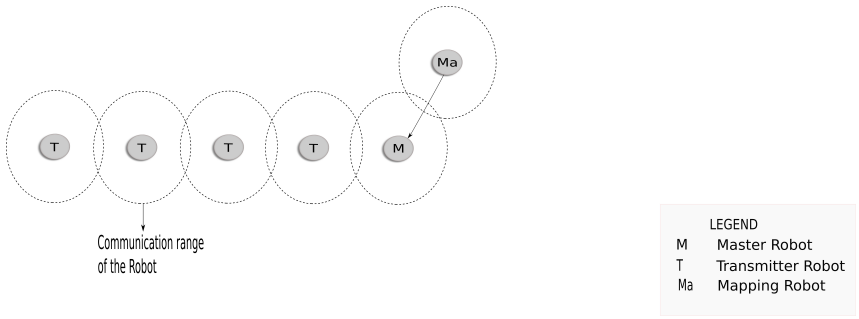

Figure 3.6: Reserve robot converted to a Mapper Robot.

Section 3.3.4 explains the working of the algorithm in presence of free space. Section 3.3.4 explains the working of the algorithm when the robot detects an obstacle.

Mapping in Free Space

There is a high probability of regions without obstacles when mapping a large area. In such regions the *Master robot* transmits a list of target positions for the *Mapper robot* to explore. These positions ensure that the *Mapper robot* move in a curved fashion and then sweeps the free space completely.

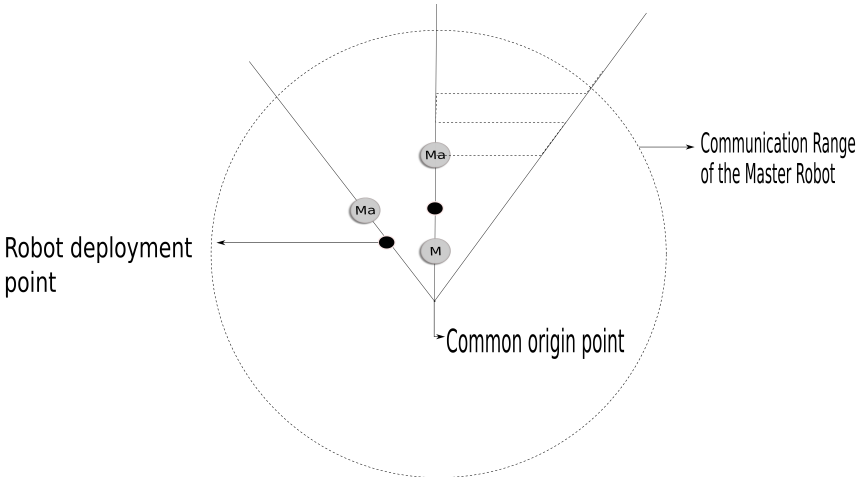

Figure 3.7: Autonomous Robot Movement algorithm.

The *Master robot* controls the movement of multiple *Mapper robots* as shown in figure 3.7. The *Master robot* allots each *Mapper robot* a particular region to map. The region is the space between two unit vectors that passes through a common origin point and the robot's initial launch points. The *Mapper robot* follows a curved trajectory, shown in fig 3.7 as dashed lines, to map the alloted region.This method

ensures that no part of the region remains unmapped. The method ensures that the *Mapper robots* map new regions and also maintain minimal overlap needed for map merging. The *Mapper robots* will map until it reaches its memory or communication limit.

Mapping in presence of Obstacles

On detection of an obstacle by the *Mapper robots*, it scans the surface of the obstacle using LIDAR to search for a corner. If a corner is detected, the *Mapper robot* moves towards the corner and switches to *Master robot* behavior. If a corner is not detected, the *Mapper robot* moves along the side of the obstacle and continue the mapping process.

3.3.5 Robot Replacement Algorithm

The Robot Replacement Algorithm ensures that the communication system is robust and resilient to any damages. If a broken *Transmitter robot* is identified in the communication network, this algorithm ensures that they are quickly replaced. Section 3.3.5 explains the method used for identifying the broken *Transmitter robot*. Section 3.3.5 explains the method used for replacing the broken *Transmitter robot*.

Identification of broken robots

Each *Transmitter robot* in the communication network is linked with its parent and multiple children *Transmitter robots*. The responsibility of the parent *Transmitter robots* is to listen to the information being broadcasted by its children and also to check if they are working.

Figure 3.8 shows a functioning communication network composed of working *Transmitter robots*. The communication network starts from the *Root robot* and grows with each *Transmitter robot* being linked to one or more children *Transmitter robots*.

Figure 3.9 shows a broken communication network due the absence of a *Transmitter robot*. The *Transmitter robot* labeled as **T1** is responsible for identifying the problem and requesting additional robots to repair it. The parent *Transmitter robots* will ping the child if it remains unresponsive for fixed period of time. If the

child robot fails to respond, then the parent robot considers it broken and requests a new robot as its replacement.

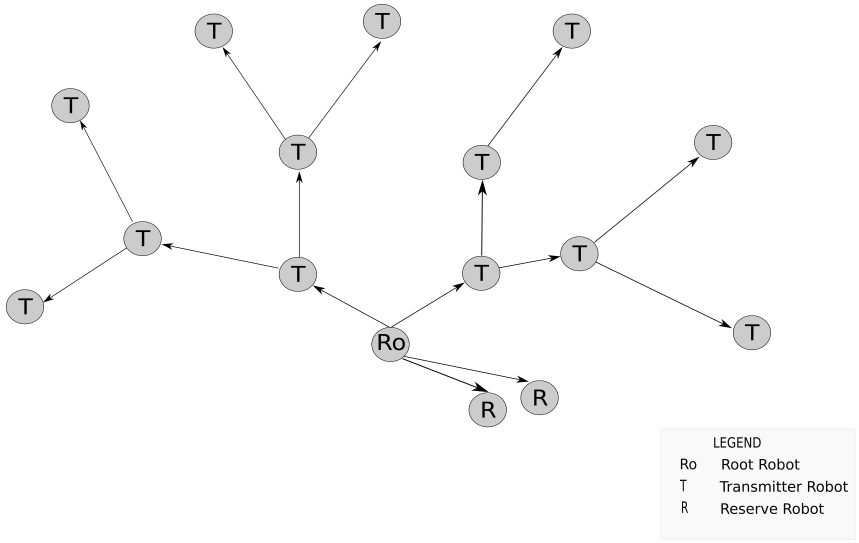

Figure 3.8: Transmission Chain without a broken robot.

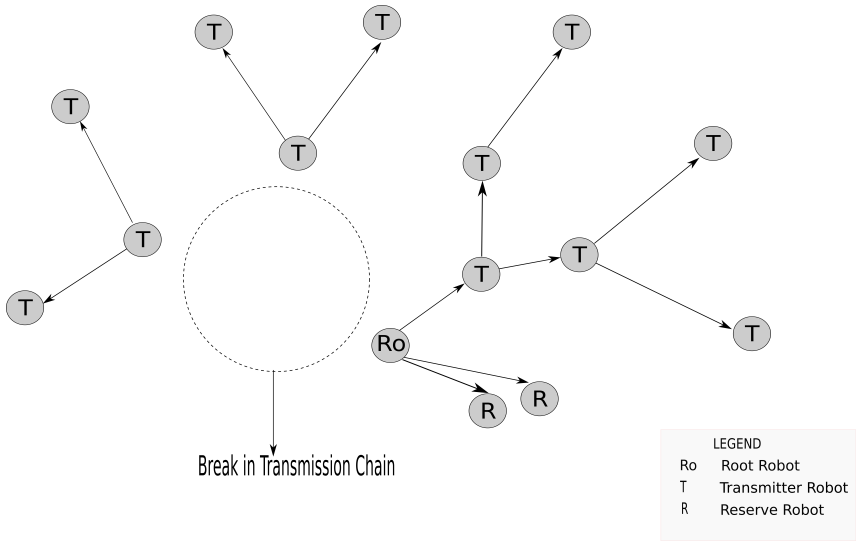

Figure 3.9: Transmission Chain identifies a broken robot.

Replacement by a Reserve Robot

Reserve robots are summoned for replacing the broken *Transmitter robot*. Once a parent *Transmitter robot* request for a replacement, the request will be carried along

the transmission network until a *Reserve robot* answers it. The *Reserve robot* will employ the Robot Deployment algorithm explained in Section 3.3.3 to reach the parent *Transmitter robot*.

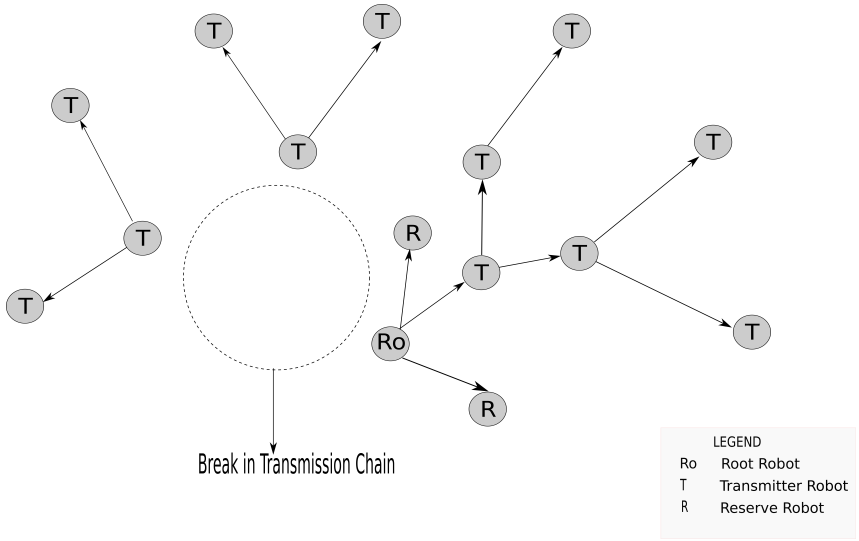

Figure 3.10: Reserve robot moving to replace the broken robot.

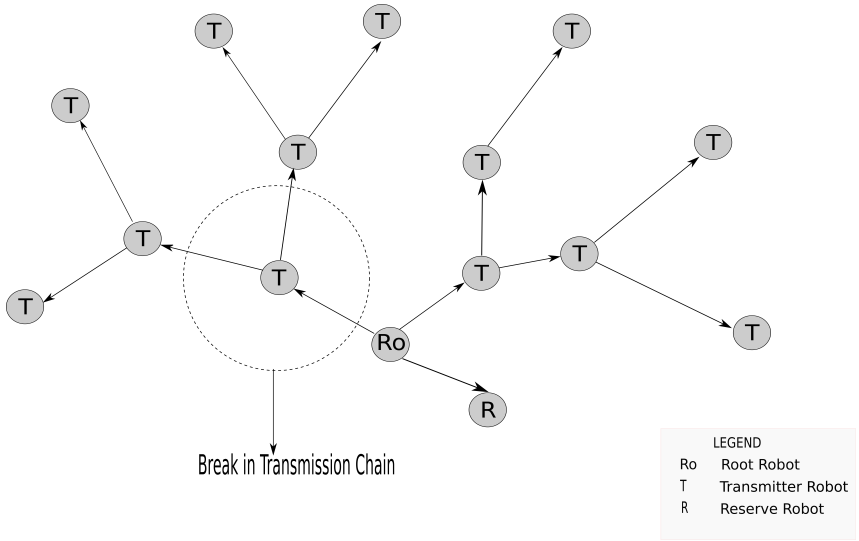

Figure 3.11: Fixed communication network.

Chapter 4

Experimental Validation

Chapter 3 discussed the working of the swarm and the different algorithms that are used in the swarm network. Chapter 4 validates this work by testing it in different sized environments. Section 4.1 discusses the different parameters that are used for evaluating this work. Section 4.3 shows the compiled results from the experiments.

4.1 Parameters Analyzed

4.1.1 Size of the map developed

The total size of the map developed by the robots in terms of number of grid cells mapped is a parameter that is used for evaluating this work. This parameter validates the ability of the *Mapper robots* to map regions filled with obstacles and also the ability of the communication network to transmit information from multiple *Mapper robots* to the *Root robot* where it is merged with the global map.

4.1.2 Number of robots used for map development

The robots are used for mapping and also for constructing the communication network. This parameter represents the number of robots that are deployed for developing the map.

4.1.3 Time taken for map development

This parameter represents the time taken by the swarm system to explore the entire environment, map the region and transmit the map back to the *Root robot* using the constructed communication network.

4.2 Experiment Setup

4.2.1 Foot-bot

The foot-bot was built in the Swarmanoid Project (Dorigo et al., 2013). It is a differential drive robot of diameter 13 cm and height 28 cm. The foot-bot is used in this experiment for implementing this work. The velocity commands given to the robot are used for estimating its motion. A distance scanner which can measure up to a distance of 300mm is used for observing the environment. 24 IR sensors mounted around the robot is used avoiding obstacles in its path. Communication between robot is achieved using a range and bearing sensor.

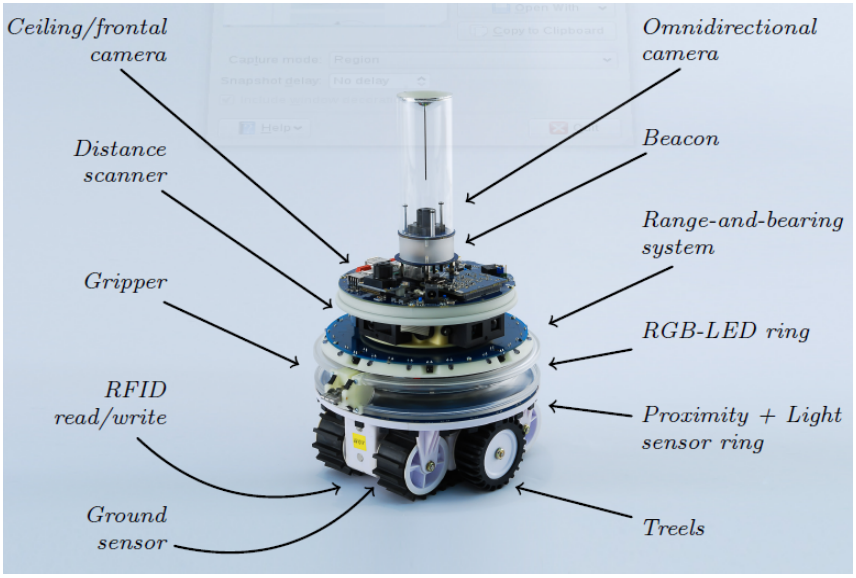

Figure 4.1: Foot-bot (Source: IRIDIA)

4.2.2 Memory limitations of the robot

The memory capacity of the robot is a parameter that is varied in the experiments. The memory capacity is the number of grid cells each robot can map. This factor limits the area each robot can map. 100, 200 and 300 are the three different values of memory capacity chosen for the experiments.

4.2.3 Separation angle between frontier regions

The *Master robot* allots frontier regions to the *Mapper robots* to map as explained in Section 3.3.1. The angle of separation between the frontiers determines the size of each frontier alloted to the *Mapper robot*. 0.4, 0.9 and 1.2 are the three different values of the separation angle chosen for the experiments.

4.2.4 Test Environment

The robots map 3 different environments of varying size. Each environment is cluttered with obstacles. The description of each environment is given below:

Environment One

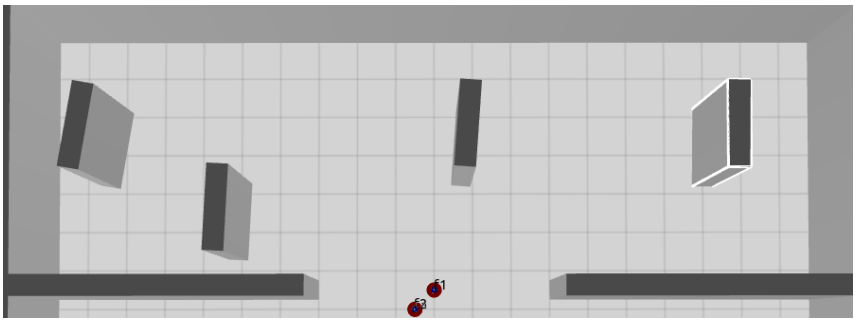

Figure 4.2: Environment One.

Figure 4.2 is the first environment which is of size $20m \times 6.5m$. The complete map of this environment consists of 1500 grid cells with a resolution of 30 cm.

Environment Two

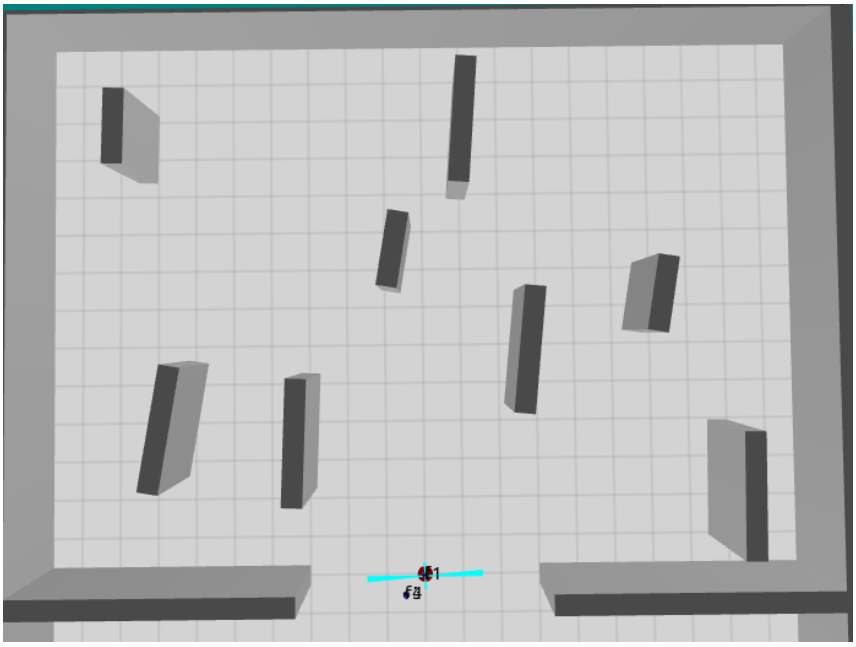

Figure 4.3: Environment Two.

Figure 4.3 is the second environment which is of size $20m \times 13.5m$. The complete map of this environment consists of 3000 grid cells with a resolution of 30 cm.

Environment Three

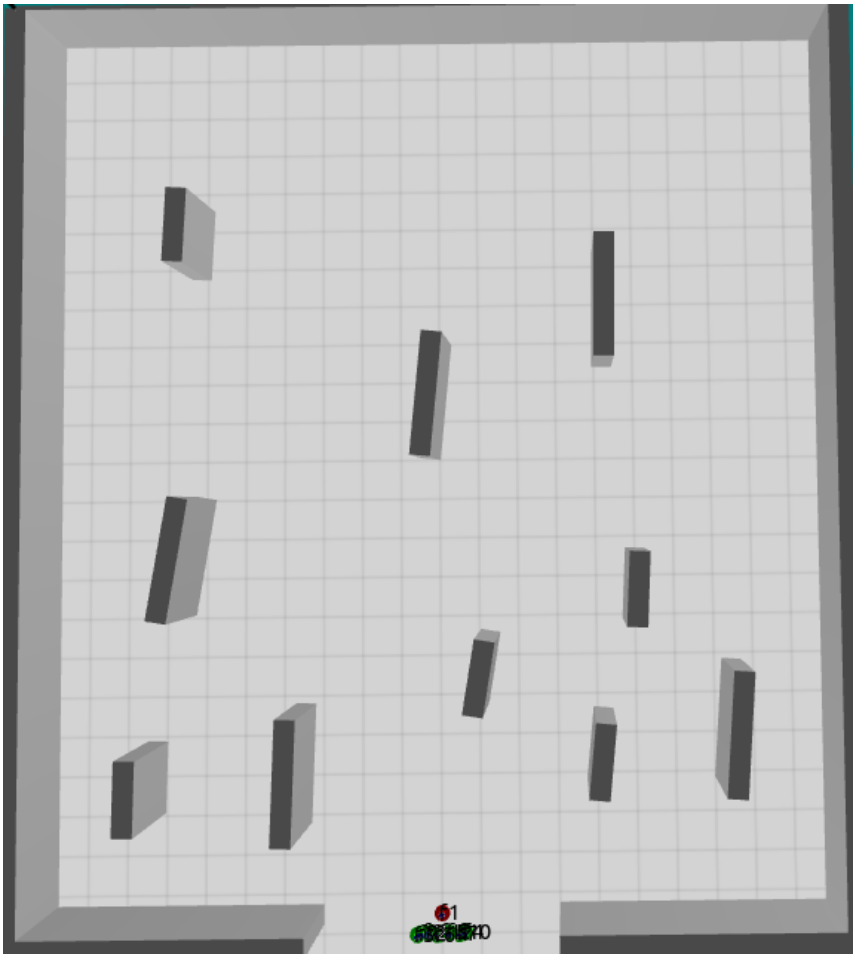

Figure 4.4: Environment Three.

Figure 4.4 is the third environment which is of size $20m \times 22.5m$. The complete map of this environment consists of 5000 grid cells with a resolution of 30 cm.

4.2.5 ARGoS - Autonomous Robots Go Swarming

ARGoS (Pinciroli et al., 2012) is a multi-robot simulator developed in the Swarmanoid Project (Dorigo et al., 2013). ARGoS has a modular and parallel design that enables accurate and efficient simulations of large-scale robot swarms.

4.3 Results

This section shows results for the various set of experiments described in Section 4.2. Each experiment is represented through four graphs. The first graph displays the map developed by the swarm network. The second, the third and the fourth are box plots with data from 50 experiments by varying the memory capacity of the robot and the angle of frontier region. The second graph compares the map developed by the swarm. The third graph compares the number of number of robots that were deployed to map the environment. The fourth graph shows the time taken by the swarm system to map the environment.

4.3.1 Environment One

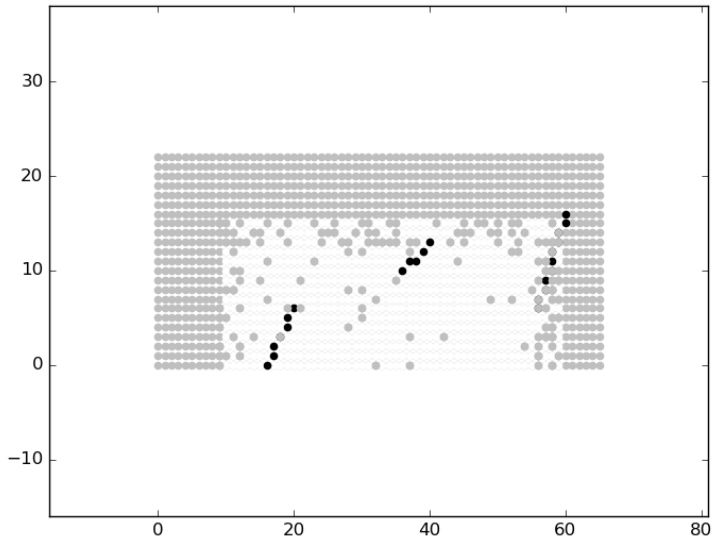

Figure 4.5: Map of Environment One developed by the Swarm Network.

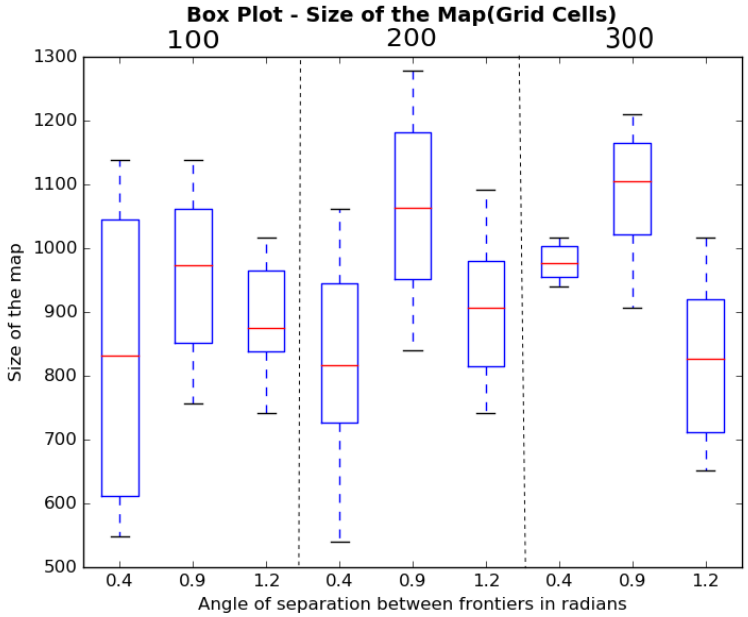

Figure 4.6: Number of Grids Mapped in Environment One by the Swarm Network.

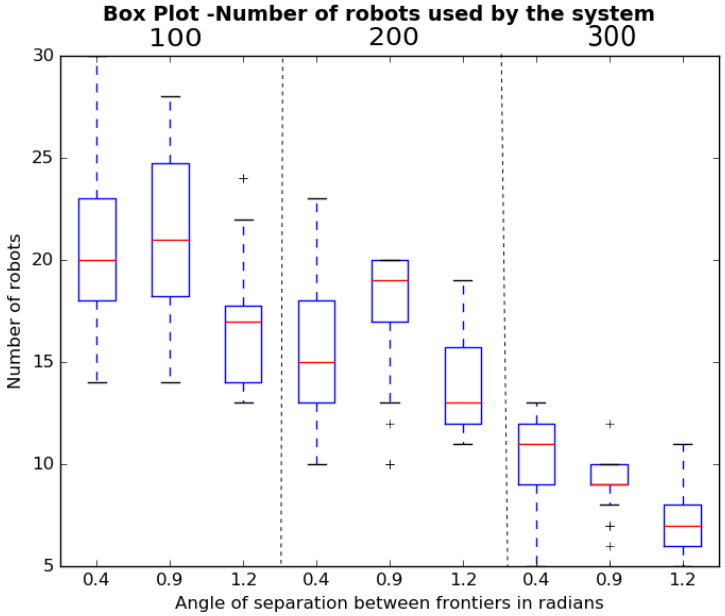

Figure 4.7: Robots Deployed in the Swarm Network to map Environment One.

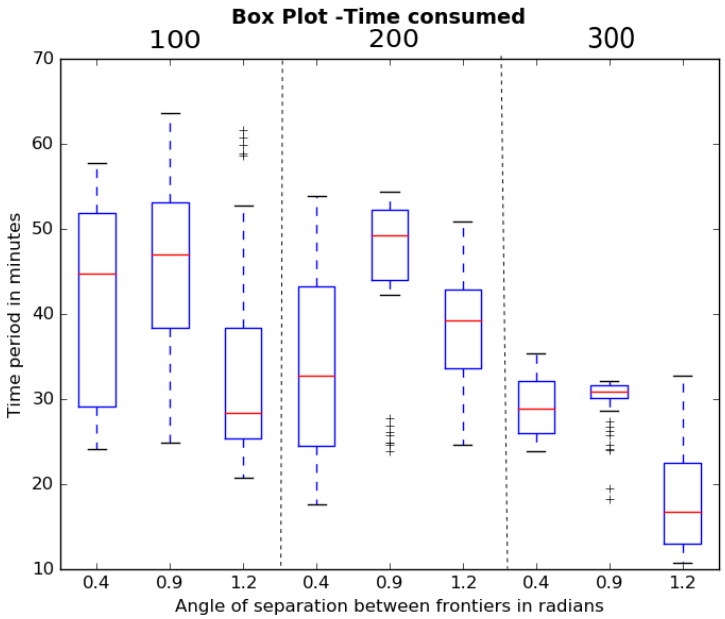

Figure 4.8: Time taken by the Swarm Network to map Environment One.

4.3.2 Medium Sized Environment

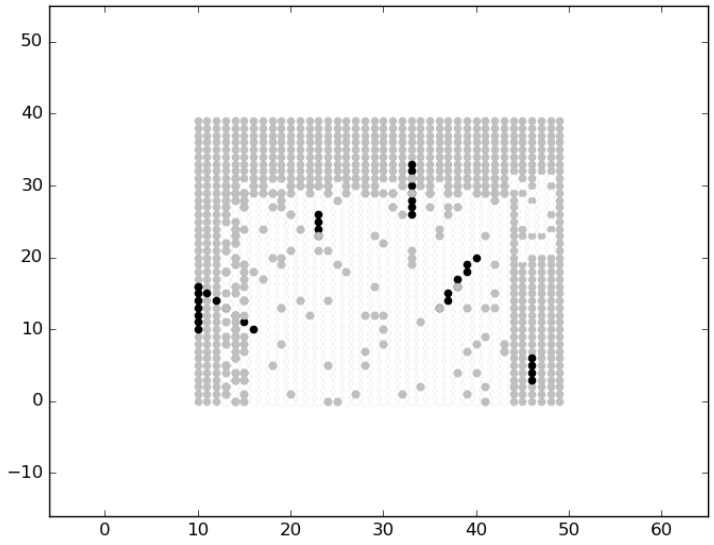

Figure 4.9: Map of Environment Two developed by the Swarm Network.

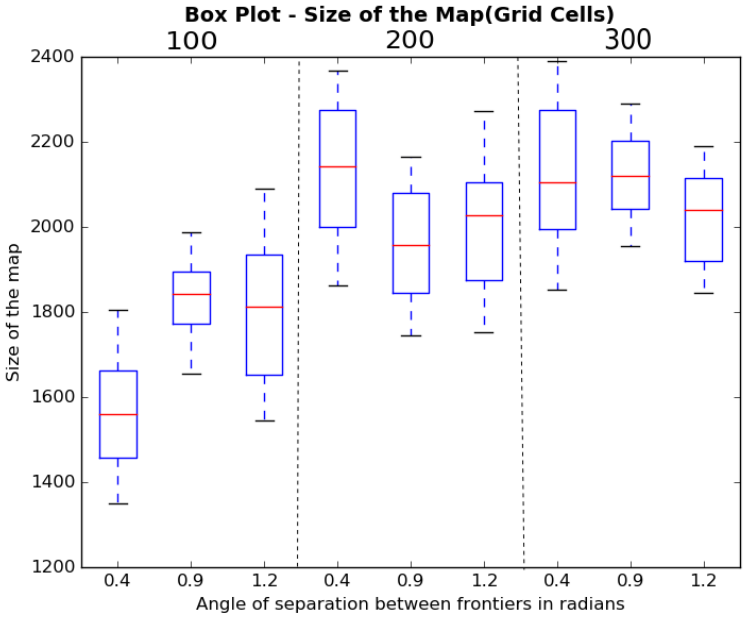

Figure 4.10: Number of Grids Mapped in Environment Two by the Swarm Network.

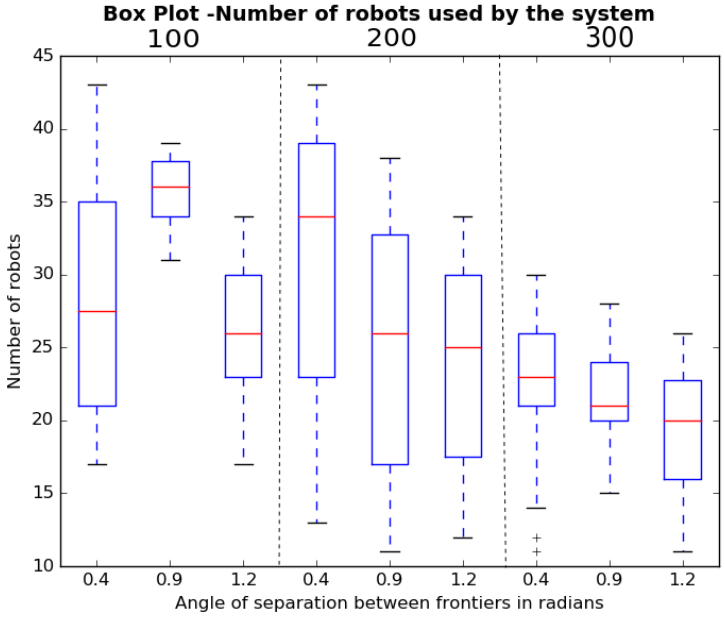

Figure 4.11: Robots Deployed in the Swarm Network to map Environment Two.

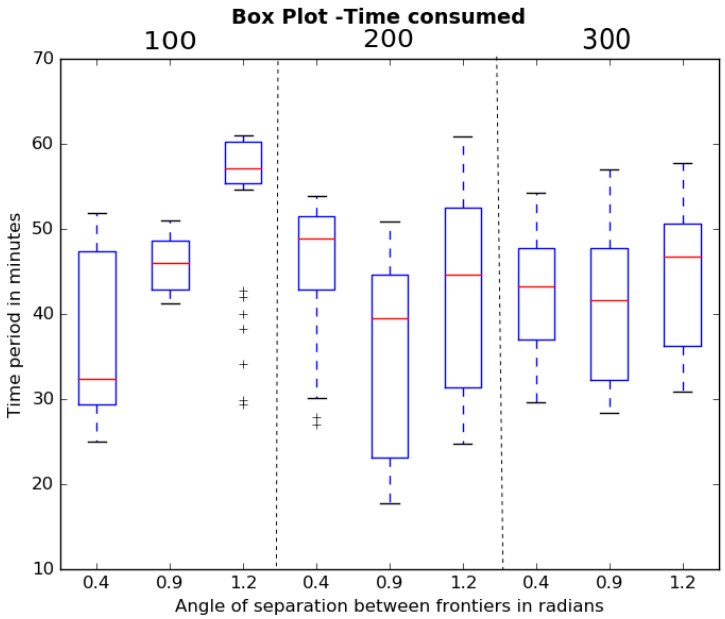

Figure 4.12: Time taken by the Swarm Network to map Environment Two.

4.3.3 Large Sized Environment

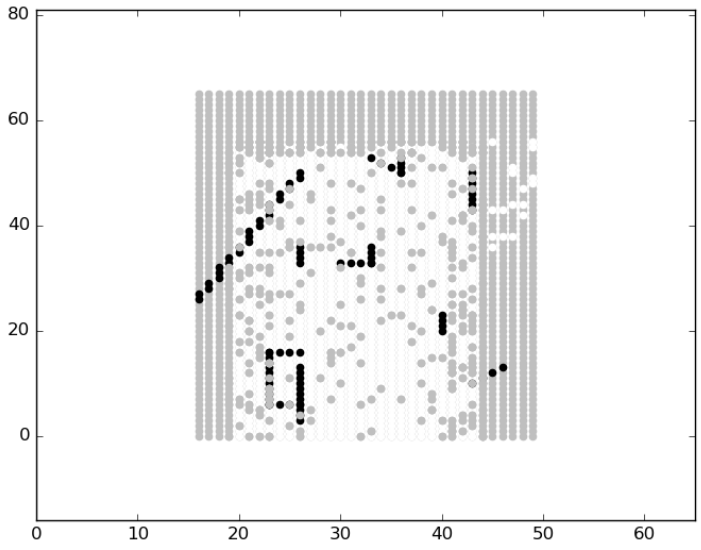

Figure 4.13: Map of Environment Three developed by the Swarm Network.

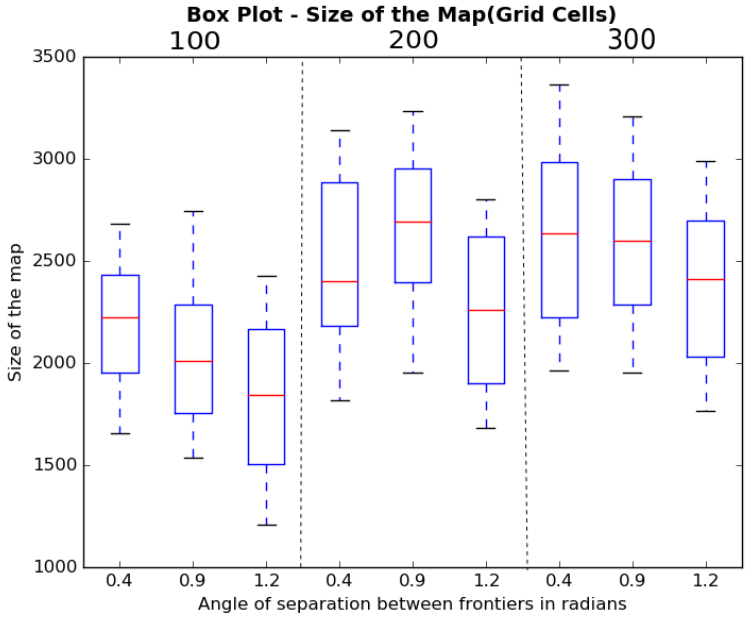

Figure 4.14: Number of Grids Mapped in Environment Three by the Swarm Network.

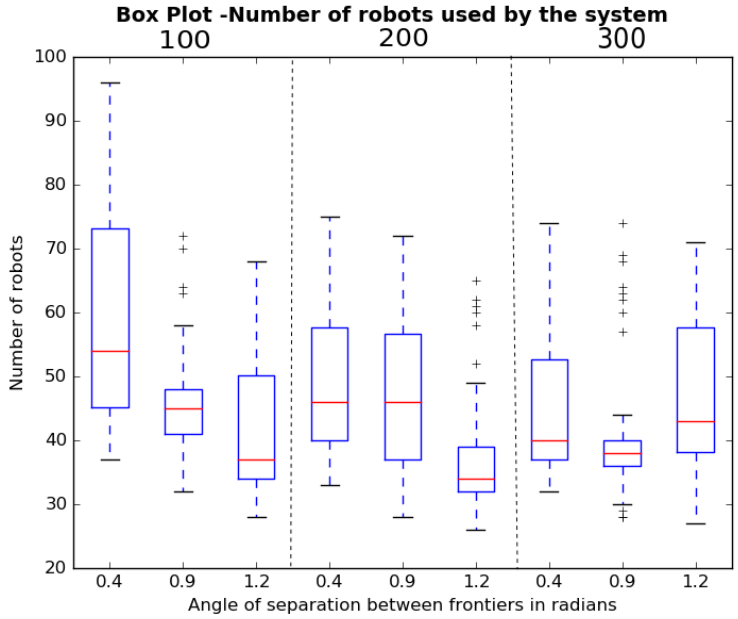

Figure 4.15: Robots Deployed in the Swarm Network to map Environment Three.

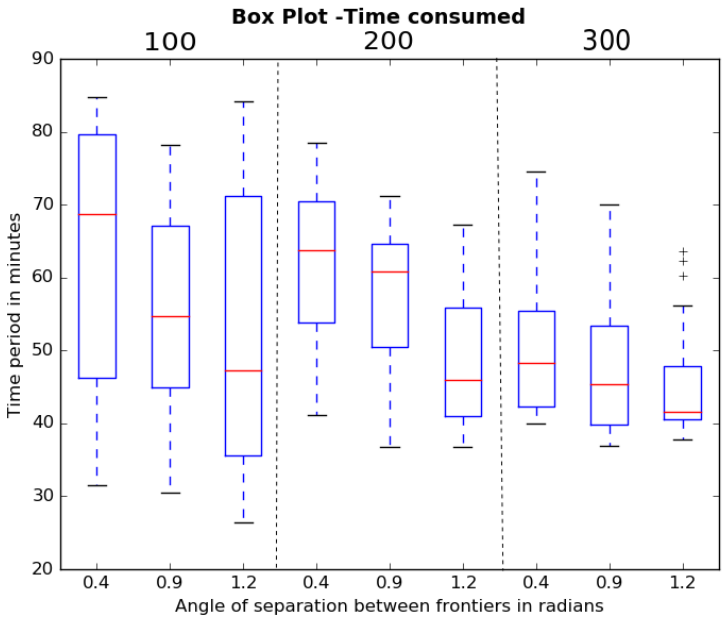

Figure 4.16: Time taken by the Swarm Network to map Environment Three.

4.4 Analysis of results

This section gives a summary of the results that were shown in Section 4.3.

4.4.1 Quality of the map developed by the swarm

Figures 4.5, 4.9, and 4.13 are the maps developed by the swarm. The white cells represent free space, the grey cells represent unknown regions and the black cells represent obstacles.

The presence of grey regions within the map and inaccurate position of obstacles in the map are due to the high uncertainty present in the developed map. Uncertainty in the developed map can occur due to several factors such as inaccurate motion and observation model, accumulation of error due to progressive deployment of robots and inaccurate map merging due to wrong initial pose information.

4.4.2 Size of the map developed by the swarm.

Figures 4.6, 4.10, and 4.14 are the box plots of the size of the map developed by the swarm. The figures show that swarms with robots having high memory capacity develops a larger map. With each robot having a high memory capacity, less robots are required to make the map of the environment and so less robots are deployed.

4.4.3 Number of robots in the swarm

Figures 4.7, 4.11, and 4.15 are the box plots of the total number of robots deployed in the swarm. With larger memory capacity and larger angle of separation between frontiers, the number of robots deployed decreases significantly. With larger memory capacity, each robot maps a larger region and by increasing the angle of separation between frontiers, lesser robots are requested for expanding the swarm network.

4.4.4 Time taken to develop the map by the swarm

Figures 4.8, 4.12, and 4.16 are the box plots of the time taken to develop the map by the swarm. The time taken for constructing the map reduces with an increase in memory capacity of the robots and this occurs because less robots are required to reach the extreme ends of the swarm. Time waste due to the robots getting stuck between obstacles or other robots within the swarm is reduced.

Chapter 5

Discussion and Future Works

5.1 Conclusions

An approach to decentralized SLAM using a robot swarm is proposed in this work. The swarm network is created using robots with limited memory and communication capabilities. In this work, a robot swarm constructs a global map of the environment starting from individual maps of the robots that are significantly smaller than the global map. We presented the various algorithms and robot behaviors used for developing the algorithm.

Results showed the maps developed by the swarm in three different experiments. A different sized environment with different number of obstacles were used in each experiment. With an increase in size of the region to explore and map, the uncertainty present in the swarm gradually increases and it affects the quality of the developed maps. The results also explained the properties of our approach in differently sized environments.

5.2 Future Works

This thesis tackles a decentralized strategy to perform SLAM using robot swarms. Some of the future research areas to be explored which follow the work done in this thesis are discussed below.

- **Map Merging:** This work merges individual robot maps using relative transformation between the robots. This method of map merging does not consider the uncertainties in the robot's initial pose and this increases

the error in the developed maps. An image processing algorithm can be used to match the features in the environment and produce a new relative transformation between the maps and it can be used for merging the maps.

- **Quicker Reserve Robot Deployment:** Another possible improvement of this work is to deploy robots only on receiving requests from other robots. A new method can be explored where the robots are automatically deployed to the part of the swarm that is growing. This will reduce the time wasted waiting for robots to travel to growing regions.

- **Trim the Swarm Network:** Another possible improvement tackles how to distribute robots that map unknown areas. Once the robots have completed the mapping task, they assist in deploying new robots to grow the swarm network. If no new regions can be found for deploying robots, that particular part of the swarm can be disbanded and each of those robots can move towards the growing part of the swarm system.

Bibliography

Ahn, S., Choi, J., Doh, N. L., and Chung, W. K. (2008). A practical approach for ekf-slam in an indoor environment: fusing ultrasonic sensors and stereo camera. *Autonomous robots*, 24(3):315–335.

Arturo, G. A., Mó, B. G., Miguel, J. C., Scar, R. G., and David, U. G. (2011). Cooperative simultaneous localisation and mapping using independent rao blackwellised filters.

Aulinas, J., Petillot, Y. R., Salvi, J., and Lladó, X. (2008). The slam problem: a survey. In *CCIA*, pages 363–371. Citeseer.

Baker, C., Morris, A., Ferguson, D., Thayer, S., Whittaker, C., Omohundro, Z., Reverte, C., Whittaker, W., Hahnel, D., and Thrun, S. (2004). A campaign in autonomous mine mapping. In *Robotics and Automation, 2004. Proceedings. ICRA'04. 2004 IEEE International Conference on*, volume 2. IEEE.

Birk, A. and Carpin, S. (2006). Merging occupancy grid maps from multiple robots. *Proceedings of the IEEE*, 94(7):1384–1397.

Brambilla, M., Ferrante, E., Birattari, M., and Dorigo, M. (2013). Swarm robotics: a review from the swarm engineering perspective. *Swarm Intelligence*, 7(1):1–41.

Bresson, G., Aufrère, R., and Chapuis, R. (2015). A general consistent decentralized simultaneous localization and mapping solution. *Robotics and Autonomous Systems*, 74:128–147.

Carlone, L., Ng, M. K., Du, J., Bona, B., and Indri, M. (2010). Rao-blackwellized particle filters multi robot slam with unknown initial correspondences and limited communication. In *Robotics and Automation (ICRA), 2010 IEEE International Conference on*, pages 243–249. IEEE.

Colares, R. G. and Chaimowicz, L. (2016). The next frontier: combining information gain and distance cost for decentralized multi-robot exploration. In *Proceedings of the 31st Annual ACM Symposium on Applied Computing*, pages 268–274. ACM.

Dellaert, F. and Kaess, M. (2006). Square root sam: Simultaneous localization and mapping via square root information smoothing. *The International Journal of Robotics Research*, 25(12):1181–1203.

Dirafzoon, A. and Lobaton, E. (2013). Topological mapping of unknown environments using an unlocalized robotic swarm. In *Intelligent Robots and Systems (IROS), 2013 IEEE/RSJ International Conference on*, pages 5545–5551. IEEE.

Dissanayake, M. G., Newman, P., Clark, S., Durrant-Whyte, H. F., and Csorba, M. (2001). A solution to the simultaneous localization and map building (slam) problem. *IEEE Transactions on robotics and automation*, 17(3):229–241.

Dorigo, M., Floreano, D., Gambardella, L. M., Mondada, F., Nolfi, S., Baaboura, T., Birattari, M., Bonani, M., Brambilla, M., Brutschy, A., et al. (2013). Swarmanoid: a novel concept for the study of heterogeneous robotic swarms. *IEEE Robotics & Automation Magazine*, 20(4):60–71.

Elfes, A. (1990). Occupancy grids: A stochastic spatial representation for active robot perception. In *Proceedings of the Sixth Conference on Uncertainty in AI*, volume 2929.

Eliazar, A. and Parr, R. (2003). Dp-slam: Fast, robust simultaneous localization and mapping without predetermined landmarks. In *IJCAI*, volume 3, pages 1135–1142.

Fairfield, N., Kantor, G., and Wettergreen, D. (2007). Real-time slam with octree evidence grids for exploration in underwater tunnels. *Journal of Field Robotics*, 24(1-2):03–21.

Fox, D., Burgard, W., Dellaert, F., and Thrun, S. (1999). Monte carlo localization: Efficient position estimation for mobile robots. *AAAI/IAAI*, 1999(343-349):2–2.

Fox, D., Ko, J., Konolige, K., Limketkai, B., Schulz, D., and Stewart, B. (2006). Distributed multirobot exploration and mapping. *Proceedings of the IEEE*, 94(7):1325–1339.

Gil, A., Reinoso, ı., Ballesta, M., and Juliá, M. (2010). Multi-robot visual slam using a rao-blackwellized particle filter. *Robot. Auton. Syst.*, 58(1):68–80.

Grisetti, G., Kummerle, R., Stachniss, C., and Burgard, W. (2010). A tutorial on graph-based slam. *IEEE Intelligent Transportation Systems Magazine*, 2(4):31–43.

Grisetti, G., Stachniss, C., and Burgard, W. (2007). Improved techniques for grid mapping with rao-blackwellized particle filters. *IEEE transactions on Robotics*, 23(1):34–46.

Grisettiyz, G., Stachniss, C., and Burgard, W. (2005). Improving grid-based slam with rao-blackwellized particle filters by adaptive proposals and selective resampling. In *Robotics and Automation, 2005. ICRA 2005. Proceedings of the 2005 IEEE International Conference on*, pages 2432–2437. IEEE.

Hess, W., Kohler, D., Rapp, H., and Andor, D. (2016). Real-time loop closure in 2d lidar slam. In *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1271–1278.

Hornung, A., Wurm, K. M., Bennewitz, M., Stachniss, C., and Burgard, W. (2013). Octomap: An efficient probabilistic 3d mapping framework based on octrees. *Autonomous Robots*, 34(3):189–206.

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

Kim, B., Kaess, M., Fletcher, L., Leonard, J., Bachrach, A., Roy, N., and Teller, S. (2010). Multiple relative pose graphs for robust cooperative mapping. In *Robotics and Automation (ICRA), 2010 IEEE International Conference on*, pages 3185–3192. IEEE.

Montemerlo, M., Thrun, S., Koller, D., Wegbreit, B., et al. (2002). Fastslam: A factored solution to the simultaneous localization and mapping problem. In *Aaai/iaai*, pages 593–598.

Morrison, J. G., Gavez-Lopez, D., and Sibley, G. (2016). Scalable multirobot localization and mapping with relative maps: Introducing moarslam. *IEEE Control Systems*, 36(2):75–85.

Mouragnon, E., Lhuillier, M., Dhome, M., Dekeyser, F., and Sayd, P. (2006). Monocular vision based slam for mobile robots. In *Pattern Recognition, 2006. ICPR 2006. 18th International Conference on*, volume 3, pages 1027–1031. IEEE.

Mur-Artal, R., Montiel, J. M. M., and Tardos, J. D. (2015). Orb-slam: a versatile and accurate monocular slam system. *IEEE Transactions on Robotics*, 31(5):1147–1163.

Nieto, J., Guivant, J., Nebot, E., and Thrun, S. (2003). Real time data association for fastslam. In *Robotics and Automation, 2003. Proceedings. ICRA'03. IEEE International Conference on*, volume 1, pages 412–418. IEEE.

Pinciroli, C., Trianni, V., OGrady, R., Pini, G., Brutschy, A., Brambilla, M., Mathews, N., Ferrante, E., Di Caro, G., Ducatelle, F., et al. (2012). Argos: a modular, parallel, multi-engine simulator for multi-robot systems. *Swarm intelligence*, 6(4):271–295.

Saeedi, S., Trentini, M., Seto, M., and Li, H. (2016). Multiple-robot simultaneous localization and mapping: A review. *Journal of Field Robotics*, 33(1):3–46.

Şahin, E. (2004). Swarm robotics: From sources of inspiration to domains of application. In *International workshop on swarm robotics*, pages 10–20. Springer.

Salvi, J., Petillo, Y., Thomas, S., and Aulinas, J. (2008). Visual slam for underwater vehicles using video velocity log and natural landmarks. In *OCEANS 2008*, pages 1–6. IEEE.

Sharma, S. and Tiwari, R. (2016). A survey on multi robots area exploration techniques and algorithms. In *Computational Techniques in Information and Communication Technologies (ICCTICT), 2016 International Conference on*, pages 151–158. IEEE.

Simmons, R., Apfelbaum, D., Burgard, W., Fox, D., Moors, M., and et al. (2000). Coordination for multi-robot exploration and mapping. In *IN PROCEEDINGS OF THE AAAI NATIONAL CONFERENCE ON ARTIFICIAL INTELLIGENCE*. AAAI.

Smith, P., Reid, I. D., and Davison, A. J. (2006). Real-time monocular slam with straight lines.

Tair, H. A., Taha, T., Al-Qutayri, M., and Dias, J. (2015). Decentralized multi-agent pomdps framework for humans-robots teamwork coordination in search and rescue. In *2015 International Conference on Information and Communication Technology Research (ICTRC)*, pages 210–213.

Thrun, S. et al. (2002). Robotic mapping: A survey. *Exploring artificial intelligence in the new millennium*, 1:1–35.

Thrun, S., Liu, Y., Koller, D., Ng, A. Y., Ghahramani, Z., and Durrant-Whyte, H. (2004a). Simultaneous localization and mapping with sparse extended information filters. *The International Journal of Robotics Research*, 23(7-8):693–716.

Thrun, S., Thayer, S., Whittaker, W., Baker, C., Burgard, W., Ferguson, D., Hahnel, D., Montemerlo, D., Morris, A., Omohundro, Z., et al. (2004b). Autonomous exploration and mapping of abandoned mines. *IEEE Robotics & Automation Magazine*, 11(4):79–91.

Walter, M. R., Eustice, R. M., and Leonard, J. J. (2007). Exactly sparse extended information filters for feature-based slam. *The International Journal of Robotics Research*, 26(4):335–359.

Williams, B., Cummins, M., Neira, J., Newman, P., Reid, I., and Tardós, J. (2009). A comparison of loop closing techniques in monocular slam. *Robotics and Autonomous Systems*, 57(12):1188–1197.

Yamauchi, B. (1997). A frontier-based approach for autonomous exploration. In *Computational Intelligence in Robotics and Automation, 1997. CIRA'97., Proceedings., 1997 IEEE International Symposium on*, pages 146–151.