# Communication Network Discovery and Leader Selection Strategies for Multi-Robot Deployments


Student Name: Shayan Lahiri

IIIT-D-MTech-CS-2010078


Aug 1, 2015


Indraprastha Institute of Information Technology

New Delhi


<u>Thesis Committee</u>

P.B. Sujit (Chair)

A.V. Subramanyam (Internal)

Vikram Gupta (External)

Submitted in partial fulfillment of the requirements

for the Degree of M.Tech. in Computer Science

# Certificate

This is to certify that the thesis titled **”Communication Network Discovery and Leader Selection Strategies for Multi-Robot Deployments”** being submitted by **Shayan Lahiri** for partial fulfillment of the requirements for the degree of Master of Technology in Computer Science & Engineering is a record of the bonafide work carried out by him under my guidance and supervision at Indraprastha Institute of Information Technology, Delhi. This work has not been submitted anywhere else for the award of any other degree.

**Dr. P.B. Sujit**
**Indraprastha Institute of Information Technology, Delhi**

# Abstract

Robots have been successfully deployed during natural disasters to perform remote search and rescue missions. These robots are tasked under human operator supervision. For remote operations, network connectivity is essential. However, there is a scarcity of network infrastructure in a post-disaster scenario and the robots may lose connectivity with the operator. If the environment is dynamic, a robot may be disconnected from the base station, in which case, either it stays at the current location until network is re-established or searches in the environment to re-establish the connection. An intuitive mechanism of returning to the last network connected location may be an inefficient strategy. In this thesis, we use foraging concepts from the animal kingdom to address the problem of connection re-establishment in sparse network coverage scenarios. We use a combination of Lévy walks, past path memory and convex hull concepts to develop an efficient hybrid model that allows the robots to escape from no-network areas. Simulation results are presented that show the superiority of our hybrid model in establishing connectivity with the base station compared to Lévy only search, memory-based search and random search.

It is also important to ensure that the time spent on interaction between the human operator and the robots/agents is as minimal as possible. If the operator has to control a large number of agents, known as a swarm, then it becomes time consuming for him to interact with each agent. The interaction time can be reduced if the operator controls a subset of agents to guide the behaviour of the swarm. This can be even further reduced by removing operator control over selection of agents within a swarm. Hence, we examine how automatic selection of agents, within a swarm, should be done so as to influence the other agents to complete the assigned task. We also find out how many influencing agents should be selected and where they should be located for efficient relocation of the swarm without any swarm fragmentation.

## Acknowledgements

It has been a long journey and this thesis would not have been possible without the influence of many people along the way. My thesis would be incomplete without acknowledging their contribution. I would like to thank Dr. P.B. Sujit for his immense patience in mentoring me and guiding me through the stumbling blocks I encountered along the way. I have no hesitation in acknowledging the fact that without the constant support of Dr. Sujit, my thesis may not have come to fruition in its current form.

My parents and my sister were a constant source of motivation, advice and inspiration for me along the journey. Professor Manohar Khushalani was extremely supportive of me and was always available to listen to me whenever I faced problems along the way. I would also like to thank Stuti Jain for being my rock throughout my thesis work by having unshakeable faith in me and always believing in my abilities. Alvika Gautam and Digvijay Singh gave me valuable inputs throughout my work which enabled me to push ahead whenever I hit a roadblock. My work would be incomplete without the help of Dipto Sarkar, Akshit Gupta, Shubhansh Agrawal and Parikshit Maini for their help with data collection. For this, I will always be grateful to them.

# Contents

# List of Figures

# List of Tables

# Chapter 1

# Introduction

The last decade saw significant developments in the field of unmanned robots. A consequence of this is an increase in their deployment in a wide array of application scenarios among which post-disaster search and rescue, and damage assessment activities are critical. The post-disaster search and rescue operations are carried out by a remote operator who tasks the robots to perform some action. Once tasked, the robots perform the tasks autonomously. Thus, human operator supervision is necessary. For supervision and tasking, the robot must be connected to the base station. A key issue with post-disaster deployment is that the environment is very dynamic and there is a deficiency in network coverage and infrastructure. Even after placing communication infrastructure, communication capability may often be disrupted.

For continuous operation under network disconnection, iridium satellite communication can be used. But this mode of operation is expensive and the data rates required for video transmission is limited. However, due to the presence of multiple robots, the robots can co-operate with each other to form ad-hoc dynamic chains to connect with the base station [1]. Another approach is to record the location of the previous established communication between the agents and return to that location to re-establish the communication [2]. Alternatively, the robot can perform random movements with the hope of re-establishing connection, but, this may be expensive with no guarantee of success.

Communication plays a critical role in cooperative multi-robot systems. The capabilities of these systems can be maximised to achieve better performance with communication [3]. There are several studies that deal with mitigating communication failure during formation flight [4, 5, 6], task allocation [7, 8], and reconnaissance [9]. Most of these works address the issue of communication failure at the agent level, while we address the issue of communication failure at the mission level where the operator is disconnected from the robot team due to a dynamic environment. We assume that the communication system of the robot is working but the robot is unable to establish a link with the base station because of the dynamic environment. Therefore, there is a need to develop network discovery strategies for robots when the connection with base station fails.

Over the years, the number of agents that work together to complete a task has increased as more efficient methods of controlling the agents are discovered. Some of the control strategies exploit knowledge of animal behaviour that have been researched over the years. An example is swarming behaviour in which collective behaviour is displayed by animals of similar size enabling them to aggregate together. It is important for the human operator to get a quick response from the swarm to complete a task in the minimum possible time without any swarm fragmentation occurring.

The human operator has two ways of controlling a swarm - directly and indirectly. In the direct way of controlling a swarm, an informed agent can be placed amongst uninformed agents to influence swarm behaviour [10, 11]. The human operator can use a leader, a mediator or a predator for this purpose. A Leader is an agent that can pull other agents, in its zone of influence, towards itself due to having a higher radius of influence as compared to other agents. A Predator is an agent that repels other agents, in its zone of influence, away from itself. A Mediator is a special kind of agent that can repulse other agents, in its zone of influence, away from it in such a manner that the agents form a torus around the mediator [11]. In the indirect way of controlling a swarm, a beacon can be placed in the swarm environment. The beacon can influence nearby agents to change their behaviour as desired by the human operator [12]. This allows the operator to exert passive influence on the agents unlike the direct way of influencing the swarm in which the operator has to perform explicit selection of the influencing agents.

Two types of leadership exist - Tacit and Explicit Leadership [13]. Tacit leadership uses consensus in which there is no distinction between a leader and a normal agent in the swarm. Every agent will attempt to set its direction based on the average direction of other agents in its zone of influence. Explicit leadership uses flooding in which the influence of the leader takes precedence over influence of other agents. Every agent will attempt to set its direction based on the direction of the leader or the direction of the nearest agent that has already been influenced by the leader.

In this thesis, we develop network discovery strategies based on bio-inspired techniques, specifically, Lévy flight from animal foraging theory. One of the primary reasons for animals to move in their habitat is to search for food. "If the environment is unchanging or quite predictable, animals may develop knowledge of where to locate resources and they will exploit that knowledge" [14]. But this is seldom the case. Resources are often scattered intermittently over the entire landscape, and hence, the animal can not move directly to the resource most of the time. Hence, to forage optimally for sparsely distributed resources, animals often demonstrate a Lévy pattern in their way-point distribution [15]. Nurzaman et al. [16, 17] used the concept of Lévy walks along with other biologically inspired methods for information foraging tasks. Sutantyo et al. [18] used a combination of Lévy walks and artificial potential fields for information foraging tasks. However, in this thesis, we explore the use of Lévy walks to enable semi-autonomous robots to operate in areas of sparse network coverage. The space-filling fractal nature of Lévy flights [19] makes it ideal for network discovery.

We also present the strategies for leader selection in human-swarm control and identify the conditions that enable the swarm to change its orientation to quickly move from one location to another under the influence of leaders within the swarm.

We study the network discovery and leader selection problem independent of each other. This allows us to analyse the individual strategies for each problem and then integrate the strategies once we have identified the best solutions to the two different problems.

# Chapter 2

# Related Work

To understand the interaction between human operators and robots/agents, it is necessary to keep track of how Human-Robot interaction (HRI) and Bio-inspired robot team (BIRT) studies have progressed in recent times. While HRI models agent behaviours as per principles that are easy to comprehend by the human psyche (physicomimetics model), BIRT models agent behaviours as per principles of behaviour seen in biology (biomimetics model) in insects such as ants and bees [20]. If we combined the HRI and BIRT research findings, then it would help us design agents that are both robust and responsive to human input [21]. This would give rise to self-organised systems that are robust and scalable [22].

An example of BIRT can be seen in Reynolds' model [23] and Couzin's model [24] which are two well-known flocking models that have been used as the basis for experiments by researchers over the years. Vicsek et al. [25] used a special version of Reynolds' model to study the self-organisation characteristics in flocks which is used to produce emergent behaviour. However, they didn't study how to change flock behaviour by introducing agents into the flock. Jadbabaie et. al. [26] used one feature of Reynolds' model to show that adding an agent, which acts as a leader, to a flock will always cause it to converge to the heading of the agent. Similar work was done by Su et al. [27] as well as Celikkanat and Sahin [28] to show that a flock will eventually converge to a particular heading under the influence of an agent that acts as a leader. Genter et al. [29] showed that a flock can converge to a new heading rapidly under the influence of an agent acting as a leader. These influencing agents guide the swarm by displaying a tendency to move towards the new heading instead of informing the other agents that they have knowledge of the desired heading [28, 30]. Han et. al used a method known as Soft Control to influence the flock by using a Shill which modifies its heading to "trick" the other agents in the flock to change their heading too. A Shill is an external agent which can be introduced into the flock and given behaviour characteristics that are similar to the other agents in the flock thereby ensuring the other agents are unaware that an informed agent is in their midst [10]. Jadbabaie et al., Su et al., Celikkanat and Sahin, Genter et al. and Han et al. added external agents to influence the flock whereas we have chosen agents from within the swarm to act as leaders and quickly guide the swarm to a new heading.

Goodrich et al. [11, 31] made use of Couzin's model in their work on using mediators to shape Couzin-like tori. Goodrich and Pendleton [32] studied the scalability of human-swarm interaction by utilising Couzin's model to represent a UAV flock. The agents considered in our work are bio-inspired agents that follow Couzin model which is actually a biomimetics model.

Kolling et al. studied principles of swarm control such as selection and beacon control which allow human operators to control large swarms. Their studies showed that new human operators work better with selection control than beacon control [12]. Goodrich and Pendleton found that having a Leader-based approach to controlling the behaviour of a swarm worked better than having a Predator-based approach [32]. Amraii et al. also found that Leader-based control is easier for a human operator to manage as compared to a Predator-based control [13]. Based on these prior works and keeping in mind the need to reduce the workload for the human operator, we decided to apply selection control using a Leader-based approach to control the swarm.

Previous work has been done by using artificial potential fields [33, 34, 35] and by using gyroscopic forces [36] to model obstacle avoidance by agents in a swarm. Goodrich has given a comprehensive explanation on utilising potential field concepts to model an agent avoiding a repulsive obstacle field while still being attracted towards a new heading [37]. Hence, we have utilised his equations while modelling obstacle avoidance in our thesis.

# Chapter 3

# Problem Description

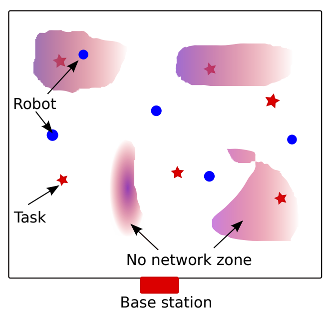
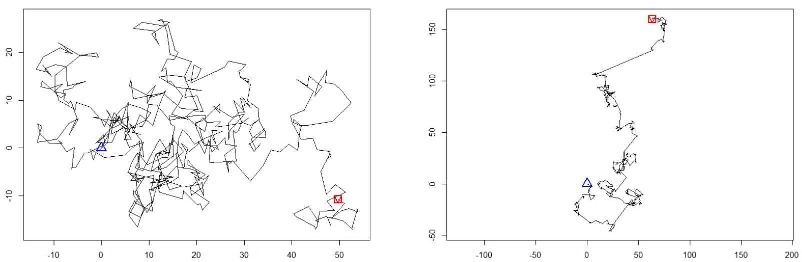



**Figure 3.1:** A multi-robot deployment with task and no-network zones

Consider a multi-robot deployment in a post-disaster scenario as shown in Figure 3.1. The region consists of robots, the available tasks for the human operator and the no-network zones. The no-network zone size, shape and its location are unknown to the operator. The operator has the knowledge of the tasks that are to be carried out and the location of the robots that are connected directly or indirectly to the base station. During the operation, the robots may be assigned tasks that are to be carried out in a no-network zone, or the current location of the robot itself may fall in the no-network zone due to dynamic environmental conditions. In both cases, the robot is unavailable to perform further missions due to network disconnection. This situation prevails until another robot is within communication range of the stranded robot in a no-network region, or the robot itself comes back into communication range. Thus, there is a need to develop and integrate network discovery strategies into robots for effective re-connection with the base station. In this thesis, we use the concepts from Lévy flights, memory-based, and convex hull to create an efficient hybrid model that can discover network




(a) (b)

**Figure 3.2:** (a) The random walk of a robot from a given start (square) and goal (triangle) location (b) The path of the robot using Lévy walk strategy between a source (square) and destination (triangle) locations

connectivity region through exploration. As the area of operation is large, Lévy flights used for foraging by animals is an ideal candidate for finding regions of connectivity in these situations.

Consider the case where we want to improve the efficiency of the work done by the robots in a post-disaster scenario. Instead of having a few robots in the post-disaster scenario, we would have multiple robots which can form a swarm. When the swarm size is small, it is easier for the human operator to manually control the agents and direct their operations as compared to when the swarm size is large. Thus, when the swarm size is large, we need strategies that allow the human operator to control a small section of the swarm which in turn influences the rest of the swarm to do the tasks assigned by the operator. Hence, we come up with a model in which automatic selection of the agents can be done, thereby, only leaving the human operator with the task of selecting the tasks to be done. If control of selecting agents were taken out of the human operator's hands and made automatic instead, then it would reduce his mental workload, on account of him not being required to constantly monitor the environment, and reduce the time taken to complete the tasks. Thus, we also need to study within a swarm, which agents should be automatically selected so as to guide the rest of the swarm to the targets without requiring further input from the human operator. Specifically, we want to see where the selected agents should be located so as to guide the swarm to the targets as quickly as possible while maintaining the swarm integrity.

Lévy walks are inherent in the search patterns of several animal species [14, 38, 39, 40]. Lévy walks are random walks, characterized by a power law distribution of the form:

$$P(l_j) \sim l_j^{-\mu} \qquad where \;\; 1 < \mu \leq 3, \tag{3.1}$$

where $P(l_j)$ is the probability of taking a straight line step of length $l_j$ before changing direction randomly. The power law implies that there will be many short steps interspersed by a few longer jumps, however, the probability of taking a long step is higher than in a normal distribution.

Power law distribution is abundant in the foraging patterns of animals because of the fractal nature of the distribution [19]. The scale free nature of the Lévy motion makes it ideally suited to efficiently search an area where resources are sparsely distributed. We use the concept of Lévy walks to enable the robots to forage for network. We consider the network region as the area of resource and use Lévy walks to enter the network range.

An example of a Lévy walk and the Brownian motion are shown in Figure 3.2. The figure shows a simulation for 500 steps comparing Lévy Walk and Brownian motion. The probability of taking longer steps is significantly higher in the case of Lévy Walks. Hence, a larger area is searched more effectively for sparsely distributed resources.

# Chapter 4

# Network Discovery Strategies

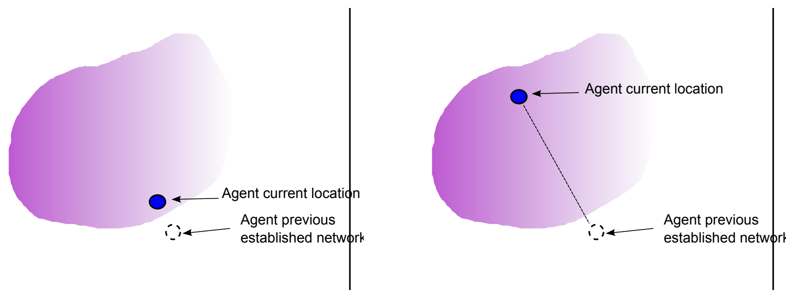
When a robot finds itself in an area devoid of network coverage, it can use several strategies to regain connectivity. These may include random search, memory-based search, Lévy walks or modified versions of Lévy Walk as introduced in this thesis.

## 4.1 Random walk

A simple strategy is to perform a random walk which will enable the robot to escape from the no-network zone. However, the time taken to escape can be large depending on the size of the no-network zone. A path of the robot from a start and goal location is shown in Figure 3.2(a) on page 7.

## 4.2 Memory

Let $p_i(t)$ be the current location of the agent $A_i$ and $\hat{p}_i$ be the last network established location. The agent uses the desired heading as $\theta_i^d = \tan^{-1}(\frac{\hat{y}_i - y_i}{\hat{x}_i - x_i})$ and moves towards $\hat{p}_i$. This is a natural strategy when $\hat{p}_i$ is closer to the boundary of the no-network region as shown in Figure 4.1(a). If the distance between $p_i(t)$ and $\hat{p}_i$ is significant then the strategy may not be the best since, as shown in Figure 4.1(b), there is a shorter path available for the agent to move out of the no-network zone. Also, when the agent is idle and suddenly it loses the network connection, then it has to trace back the previous path it had followed. This implies that the agent has to record the complete history of its path compared to the random walk.




(a) (b)

**Figure 4.1:** (a) Situation where the agent enters the no-network region but has the previous established network location close to the no-network region (b) The agent deep inside the no-network region and the previous known established network connection point is far away

## 4.3 Lévy walk

The agent performs a Lévy walk under two situations (i) whenever it executes a task, and at the end of it finds itself in a no-network region, (ii) the agent is idle on account of being disconnected from the base station. The Lévy walk helps the agent escape from the no-network region.

## 4.4 Hybrid Lévy

There are some advantages in using both memory-based and Lévy walk-based strategies. We introduce a new strategy called as hybrid Lévy, where depending on the distance travelled by the agent in a no-network region, the robot selects either the Lévy walk or the memory-based network discovery strategy or a mixture of both. The larger the time spent in the no-network zone, the lesser the weight given to the memory component. The weight $\omega$ for the memory component is calculated as

$$\omega = 1/e^{\tau}, \tag{4.1}$$

where, $\tau$ is the time spent by the robot in travelling inside a no-network area, and $\omega \geq 0$. The selected direction $\theta_i^d$ is calculated as

$$\theta_i^d = \omega\theta_i^m + (1-\omega)\theta_i^r, \tag{4.2}$$

where, $\theta_i^m$ refers to the direction from which the robot entered the no-network area and $0 \leq \theta_i^r \leq 360$ refers to an angle chosen at random. The direction selected by the robot to perform

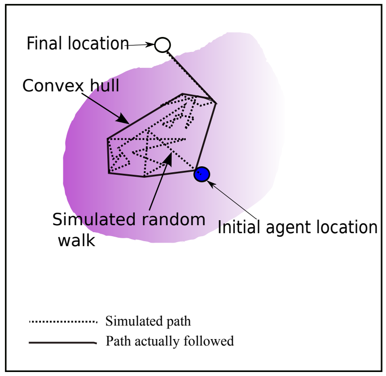



**Figure 4.2:** Visual explanation of the Convex Lévy methodology

a Lévy walk is a combination of a deterministic and a random component. The more time a robot spends travelling in a no-network area; the less certain it is about the easiest escape path. As a consequence, there is an increase in the chances of randomly changing its direction. It is to be noted that in the definition of Lévy walk, the agent changes its direction randomly after taking each step of length $l$ drawn from the power law in equation (3.1). Hence the probability of robot carrying out a "classical" Lévy walk increases with $\tau$.

## 4.5 Convex Lévy

A further modification over the hybrid Lévy model is to optimize energy consumption by utilizing convex hull of the search area. The robot in this case simulates a hybrid Lévy walk and constructs a convex hull around the path it would have traversed and visits only the endpoints of the convex hull instead of travelling the entire path. As illustrated in figure 4.2, this strategy reduces the path travelled and hence is more time saving and energy efficient.

## 4.6 Setup

We consider a bounded region of $50 \times 50$ units for a post-disaster assessment operation. The region is a scaled model. The base station is located outside the region but within network coverage. The robots are placed randomly within 10 units of the base station. Tasks are generated randomly in the world. The network coverage in the region is incomplete, and hence many tasks can be located in no-network regions. The no-network regions can dynamically change in shape and size depicting the real world environments. This simulation has been done using Repast[1] (Recursive Porous Agent Simulation Toolkit) Simphony and a snapshot of the simulation environment is shown in Figure 4.3.

The human operator can visualize the location of the robots that have communication to the base station either directly or indirectly. The operator assigns the tasks to the robot which they perform autonomously.

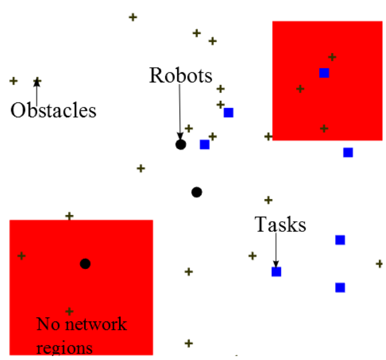



**Figure 4.3:** Screenshot from the Repast Simulation Environment

## 4.7 Results and Analysis

The simulation was run for 30 times for each of the different methods; “Memory”, “Random”, “Lévy”, “Lévy+Memory” (Hybrid) and ‘Convex Lévy‘. For each run, the number of times each robot entered a no-network area, the time spent inside no-network area each time a robot entered it and the total number of tasks completed were recorded. Table 4.1 shows the statistics for the simulations.

**Table 4.1:** Table of summary statistics

| | Total time spent in no-network area | | Number of times in no-network area | | Number of tasks completed | | Time spent per visit | |
|---|---|---|---|---|---|---|---|---|
| **Method** | **Mean** | **S. D.** | **Mean** | **S. D.** | **Mean** | **S. D.** | **Mean** | **S. D.** |
| Memory | 74.92 | 29.83 | 10.20 | 2.86 | 21.20 | 3.80 | 7.21 | 2.15 |
| Brownian Motion | 68.32 | 27.31 | 9.60 | 2.99 | 23.56 | 3.83 | 7.18 | 1.89 |
| Lévy Walk | 56.92 | 18.58 | 11.12 | 2.97 | 25.42 | 3.80 | 5.20 | 1.19 |
| Lévy + Memory (Hybrid) | 48.32 | 16.87 | 10.60 | 3.64 | 25.80 | 4.22 | 4.48 | 1.01 |
| Convex Lévy | 39.12 | 17.93 | 11.46 | 2.77 | 27.50 | 5.05 | 3.37 | 0.80 |

[1]Repast website: http://repast.sourceforge.net/

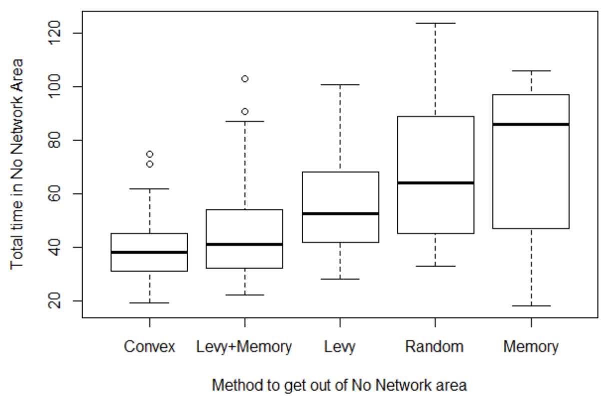



**Figure 4.4:** Box and whisker plot of total time spent in searching for network

| Strategy Selected | Percentage reduction in mean time |
|---|---|
| Memory | 0% |
| Random | 26% |
| Lévy | 35% |
| Lévy+Memory | 52% |
| Convex Lévy | 54% |

**Table 4.2:** Comparison of mean time for each Network Selection Strategy

Figure 4.4 shows the box and whisker plot for the total amount of time spent in no-network area for all the methods. Convex Lévy model takes least amount of time to discover network while the memory dependent model performs the worst. ANOVA test between the different categories reveal that the differences in time spent inside no-network area for the different methods is highly significant with $F_{4,122}$=10.07 and $p < 0.0001$. Post-hoc analysis to understand the differences in results between the methods using Tukey's HSD show that the Convex Lévy model is better than all but the Hybrid model at $p < .05$, whereas the Hybrid model is better than the Random and Memory-based model. Even the simple Lévy-based model outperforms the Memory model at a significance level of $p < .05$. Table 4.2 shows the percentage difference between each strategy based on the mean time spent in a no-network area. The result for Memory model is assumed to be the base case.

The memory-based model has the widest variance in Figure 4.4. This is expected, as shown in Figure 4.1(a) and 4.1(b) where the time taken for the memory-based escape strategy between two extreme cases is significant. The box and whisker plot also shows that the variance of the

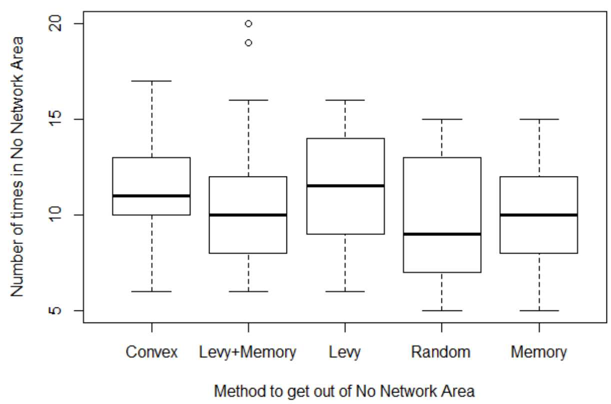



**Figure 4.5:** Box and whisker plot showing the number of times the robot went into the no-network areas

Convex Lévy model to be the lowest, indicating that it has the most reliable performance.

For simulations such as the one here, where the robots are semi-autonomous in nature, there may be significant amount of subjectivity present on behalf of the operator. In biased cases, it may be possible that the allocation of targets were done in ways that made the robots go into no-network areas more often in some cases. Thus, box and whisker plots are also made that shows the number of times the robot entered into a no-network area for each method. Figure 4.5 confirms that the experiments are free from such bias on behalf of the operator ($F_{4,122}$=1.475, p=0.214).

When a robot enters into a region with no-network to complete a task, it must get back into network coverage to receive the next assignment. The efficiency, with which the robot is able to search for network autonomously, directly affects the number of tasks completed. The variation in the number of tasks completed by the robots when using the five different foraging methods is shown through the box and whisker plot in Figure 4.6. Again, we can see that the Convex Lévy model performs the best while memory-based model performs the worst. The difference in performance amongst the methods is highly significant with $F_{4,122}$=8.416, p=0.00000497. Table 4.3 shows the percentage difference between each strategy based on the number of tasks completed. The result for Memory model is assumed to be the base case.

Similar results were obtained by using Bayesian Statistics. The results are given in Table 4.4. The analysis was performed drawing 11000 samples with a burn-in of 1000 samples using Markov-Chain-Monte-Carlo (MCMC) method. Uninformative priors were used to keep the posterior distributions unaffected by the prior distributions.

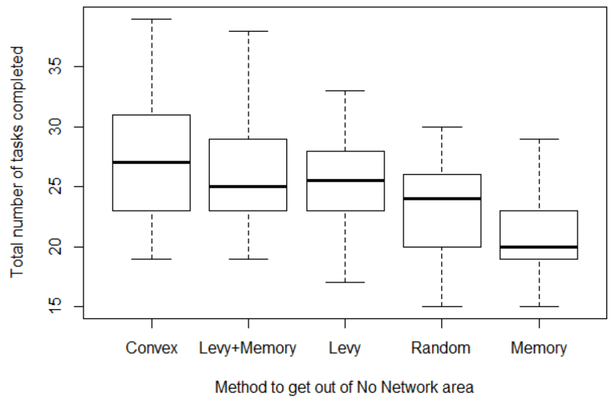



**Figure 4.6:** Box and whisker plot of the number of tasks completed

| Strategy Selected | Percentage increase in number of tasks completed |
|---|---|
| Memory | 0% |
| Random | 20% |
| Lévy | 25% |
| Lévy+Memory | 25% |
| Convex Lévy | 35% |

**Table 4.3:** Comparison of Number of tasks completed for each Network Selection Strategy

| Evaluation Criteria | Methods | Difference in means | HDI low | HDI high | Difference in SD | HDI low | HDI high |
|---|---|---|---|---|---|---|---|
| Total time in no-network area | **Convex - Memory** | -37.32 | -52.41 | -23.68 | -15.77 | -26.79 | -5.61 |
| | **Convex - Random** | -28.97 | 42.23 | -15.47 | -13.19 | -23.70 | -3.75 |
| | **Convex - Levy** | -17.74 | -27.21 | -8.13 | -3.97 | -11.40 | 3.31 |
| | **Convex - Hybrid** | -8.36 | -19.41 | 2.78 | -6.67 | -15.35 | 2.25 |
| | **Hybrid - Memory** | -28.93 | -44.68 | -11.89 | -8.78 | -20.58 | 3.16 |
| | **Hybrid - Random** | -20.46 | -35.53 | -6.19 | -6.26 | -17.86 | 5.18 |
| | **Hybrid - Levy** | 0.19 | -2.21 | 2.54 | 0.36 | -1.56 | 2.15 |
| | **Levy - Memory** | -19.49 | -34.77 | -4.65 | -11.73 | -23.42 | -0.79 |
| | **Levy - Random** | -11.19 | -25.31 | 2.49 | -8.98 | -19.76 | 1.36 |
| Number of tasks completed | **Convex - Memory** | 6.26 | 3.57 | 8.85 | 1.26 | -0.71 | 3.40 |
| | **Convex - Random** | 3.78 | 1.09 | 6.40 | 1.23 | -0.86 | 3.20 |
| | **Convex - Levy** | 1.92 | -0.63 | 4.58 | 1.30 | -0.64 | 3.38 |
| | **Convex - Hybrid** | 1.718 | -0.95 | 4.50 | 0.96 | -1.12 | 3.03 |
| | **Hybrid - Memory** | 4.52 | 2.14 | 6.91 | 0.34 | -1.53 | 2.16 |
| | **Hybrid - Random** | 2.04 | -0.39 | 4.45 | 0.29 | -1.58 | 2.13 |
| | **Hybrid - Levy** | 0.17 | -2.21 | 2.33 | 0.41 | -1.38 | 2.22 |
| | **Levy - Memory** | 4.29 | 2.02 | 6.45 | -0.03 | -1.78 | 1.67 |
| | **Levy - Random** | 1.86 | -0.40 | 4.23 | -0.08 | -1.92 | 1.62 |

**Table 4.4:** Summary of Bayesian Statistical Analysis using MCMC simulations. The results are given at 95% Highest Density Interval (HDI). The rows having a significant difference in mean are highlighted.

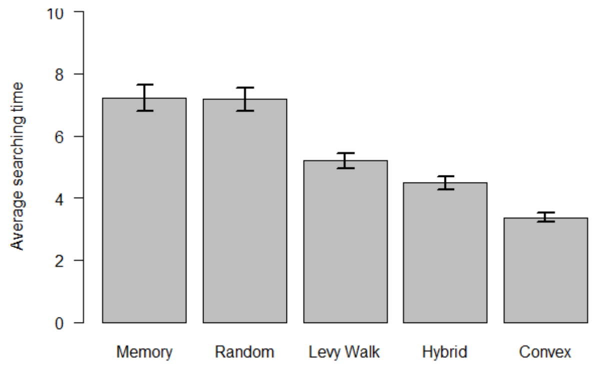



**Figure 4.7:** Bar plot showing the average search time for each of the five methods. The vertical bars show the standard errors.

Another important aspect of comparison between the methods of searching for network is the average time spent in searching for network. Note that we have already established that the experiment is free from operator bias. Hence, the average number of times the robots searched for network is similar in all the five cases. The bar plot in figure 4.7 shows the average time spent in foraging for network per search along with the standard deviation. The Convex Lévy model has the best performance.

The rate at which the no-network regions change can affect the performance evaluation. Since Convex Lévy model performed the best, we have analyzed the effect of change in no-network configuration in the world for it. Figure 4.8 shows the performance of the Convex Lévy model for different network switching times. If the network changes too quickly then staying idle is the best strategy. When the landscape changes seldom then the performance gradually saturates. Thus, Convex Lévy approach is a safe option even when robots are deployed in a scenario where the network is relatively stable.

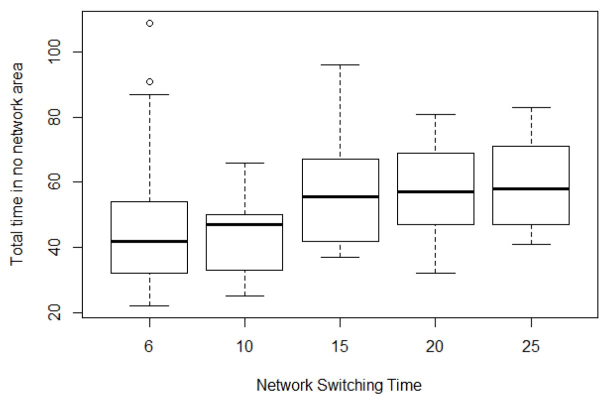



**Figure 4.8:** Box and whisker plot of time spent searching for network at different network dynamicity for the Convex Lévy approach

# Chapter 5

# Leader Selection Strategies

## 5.1 Couzin's model

The agents in our simulation for leader selection follow the swarm behaviour as outlined by Couzin et al. [24]. To achieve flocking, every agent in the swarm must model its behaviour as per the following rules:

**Rule 1** - Agents always attempt to maintain a minimum distance between each other. This behaviour takes highest precedence for any agent.
**Rule 2** - If agents are not attempting to maintain a minimum distance between each other, then they tend to be attracted to each other and orient themselves with that of nearby agents so that all the agents will end up moving in the same direction.

To achieve this, the model introduces 3 different concentric zones - Zone of Repulsion (ZOR), Zone of Orientation (ZOO) and Zone of Attraction (ZOA). If any agent ends up in the ZOR of another agent, then Rule 1 applies. If none of the agents are in the ZOR of each other, then Rule 2 applies provided there exists agents that are either in the ZOO or ZOA of another agent. The agents can only interact with other agents that exist in one of these three concentric zones. The full explanation on Couzin's model is covered in Couzin et al.'s paper. [24]

## 5.2 Leader Strategies

Before outlining the strategies that we considered, we give the definition of a convex hull in the context of the work to find an effective leader selection strategy. A convex hull is the smallest convex set that contains all the agents in the swarm within it. The vertex of the convex hull represents an agent that lies on the boundary of the swarm. We utilised a QuickHull implementation to compute the convex hull of the swarm by taking the agent coordinates as a finite set of points in a plane. The full explanation on the implementation of the algorithm is given in [41]. The leader selection strategies considered by us were:

**Strategy 1** - First we calculate the centroid of the swarm inside the convex hull. Then we take a point, say $p_{goal}$, that is 1000 units away from the centroid in the direction of $\theta$' (which is the desired direction that the swarm should move in). We calculate the distance of each agent, on the vertex of the convex hull, from $p_{goal}$ and choose the agent that is nearest to $p_{goal}$ as the first leader. The next leader selected will be the second closest agent. This goes on till either we have the required number of leaders or till we can't continue on account of having made all the agents on the convex hull as Leaders.

**Strategy 2** - Agents that are located on the vertex of the convex hull are not considered for selection as a leader. Amongst the remaining agents located inside the convex hull, the number of agents present in the Radius of Attraction $R_a$ of each agent is calculated. The agent with the highest number of agents in its $R_a$ is selected as a leader. Then the agent with second highest number of agents in its $R_a$ is also selected as a leader and so on. This goes on till we have the required number of leaders.

**Strategy 3** - In this strategy, we use a combination of Strategy 1 and Strategy 2 with 50% of the leaders coming from Strategy 1 and 50% of the leaders coming from Strategy 2.

**Strategy 4** - The first leader is selected amongst the agents, at the vertex of the convex hull, that is closest to $p_{goal}$. Once this agent $A_i$ has been selected, it is removed from consideration for selecting the next leader as well as all the other agents that lie within the $R_a$ of $A_i$. From the remaining pool of agents that are left, the agent that has the highest number of agents in its $R_a$ is selected as the next leader. Then this leader, as well as the agents in its $R_a$, is also removed from consideration for selecting the next leader. This goes on till either we have the required number of leaders or we can't continue on account of having removed all agents from consideration.

**Strategy 5** - This strategy is inspired by the works of S.K. Ganjugunte [42] and P. Maini [43], and makes use of a NP-hard Hitting Set problem by reducing it to a geometric version of the problem. Let the coordinates of all the agents in the swarm be taken as a set of points X. Let set S be different subsets of X such that every element *s* in S is a subset of X. We have to find the minimum set of points H from the set X such that H contains at least one point from each $s$. In the geometric version, the elements of the set S are discs in 2D space. Hence, all points in set X which lie within a specific disc would belong to the element $s$ of set S. Thus, after generating the hitting set, it would imply that all discs have been hit. Each disc in our problem is said to have a radius of $R_a$. The solution to the Hitting Set problem would be the minimum set of points from X that hits each disc. This would imply that each point is at most '$R_a$' distance away from a point in the computed hitting set. The agents to be picked as leaders are picked from this hitting set.

## 5.3 Setup

This simulation has been done using Eclipse IDE and a snapshot of the simulation environment is shown in Figure 5.1. We used two environments for our simulations. One environment contains no obstacles and is unbounded thereby giving the agents, in the swarm, the freedom to move anywhere within the environment. The second environment is also unbounded but contains one obstacle lying in the path of the swarm thereby limiting the freedom of movement of the agents as they approach the obstacle. We simulated a swarm using 50 agents that were randomly generated in an unbounded region of $500 \times 500$ units with the following criteria:
1) The distance between each agent should be greater than their Radius of Repulsion $R_r$
2) The position of an agent should be within Radius of Orientation $R_o$ of at least 1 other agent
3) All the agents should have the same orientation

100 different swarms were generated with agents strictly following the behavioural rules outlined by the Couzin model. The agents also had a noise parameter, $\sigma$, added to simulate the stochastic effects on their decision making abilities [24].

Human operator influence was added via a means of deciding the Leader selection strategy among the options available to him. The selected leaders were programmed to follow the rules outlined by Couzin model with the addition of a goal bias factor $\omega$' programmed into them. The desired direction in which the swarm had to be redirected was defined as $\theta' = \theta \pm \Delta\theta$ where $\theta$ is the current orientation of all the agents and $\Delta\theta$ is the angle by which the orientation has to be changed. All agents and leaders were assumed to be massless and having no size difference between them. Also, it has been assumed that the human operator can only interact with the leaders.

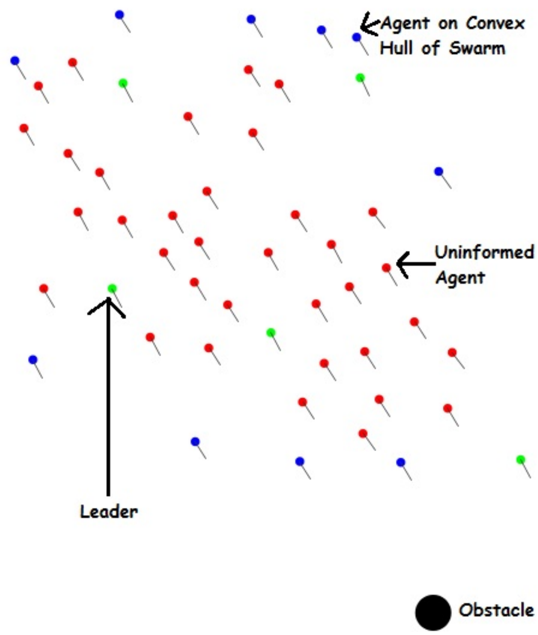



**Figure 5.1:** Screenshot from the Eclipse Environment

## 5.4 Results and Analysis

In the experimental setup, we have N agents with each agent $n_i$ having radius R and a speed $v_i$ in a continuous two-dimensional space. At any time step *t*, each agent has a position in the environment given by $p_i(t) = (x_i(t), y_i(t))$ and an orientation $\theta_i(t)$. Since the position and orientation are a function of time, hence, at every time step *t*, the value of $\theta_i(t)$ may change and the new position coordinates of the agent would be $x_i(t) = x_i(t-\Delta t) + v_i\cos(\theta_i(t))$ and $y_i(t) = y_i(t-\Delta t) + v_i\sin(\theta_i(t))$ where $\Delta t$ is the spacing between each time step.

At each time step *t*, each agent will determine where it needs to be positioned at the next time step $(t+\Delta t)$. For this, the agents need to decide their direction of movement $d_i(t+\Delta t)$ using the behaviour rules defined in Couzin model. To set the direction of an agent $A_i$ based on the agents $n_r$ in its ZOR, the following formula applies:

$$d_r(t+\Delta t) = -\sum_{j\neq i}^{n_r} \frac{r_{ij}(t)}{|r_{ij}(t)|}$$

where $r_{ij} = (c_j - c_i)/|(c_j - c_i)|$ is the unit vector in the direction of neighbour $A_j$. To set the direction of an agent $A_i$ based on the agents $n_o$ in its ZOO, the following formula applies:

$$d_o(t+\Delta t) = \sum_{j=1}^{n_o} \frac{v_j(t)}{|v_j(t)|}$$

where $v_j$ is the direction vector of agent $A_j$. To set the direction of an agent $A_i$ based on the agents $n_a$ in its ZOA, the following formula applies:

$$d_a(t+\Delta t) = \sum_{j \neq i}^{n_a} \frac{r_{ij}(t)}{|r_{ij}(t)|}$$

If there are agents only in the ZOO of agent $A_i$, then $d_i$(t + Δt) = $d_o$(t + Δt). Likewise, if there are agents only in the ZOA of agent $A_i$, then $d_i$(t + Δt) = $d_a$(t + Δt). However, if agents exist in both ZOO and ZOA of agent $A_i$, then $d_i$(t + Δt) = ½ [$d_o$(t + Δt) + $d_a$(t + Δt)]. In case agent $A_i$ has no other agents nearby or the forces of all the other agents cancel each other out, then $d_i$(t + Δt) = $v_i$.

Once the agent has decided its direction at the next time step, it is rotated clockwise or anti-clockwise by an angle $\sigma$ so as to simulate stochastic effects in the agents. The value of $\sigma$ is kept small so as ensure the agent movements are smooth [25].

Out of the N agents in the swarm, K agents will be leaders who have knowledge of the desired direction of the swarm and remaining N-K agents will be uninformed agents who have no desire to go in any particular direction but instead want to maintain social interactions with each other [30]. The human operator has control over what the value of K should be for Leader strategy 1, 2, 3 and 4. For Leader strategy 5, the algorithm determines the optimal number of leaders required to guide the swarm with the new orientation. The leaders have the same $R_r$,$R_o$ and $R_a$ as the uninformed agents. In addition to that, they have a weighting value $\omega$' which denotes their preference for moving in a desired direction $g_i$. Thus, the equation to determine the direction of the leader at the next time step is:

$$d_i{}'(t+\Delta t) = \frac{\hat{d}_i(t+\Delta t) + \omega g_i}{|\hat{d}_i(t+\Delta t) + \omega g_i|}$$

where $\hat{d}_i(t+\Delta t) = d_i(t+\Delta t) \,/\, |d_i(t+\Delta t)|$.

Monte Carlo experiments were carried out by generating 100 simulation sets with each set containing randomly generated details related to the position of the agents and their initial orientation. The orientation was the same for all the agents so as to simulate a stable swarm. The 100 sets were repeated for all the experiments so as to reduce the variance factor. At first we carried out 10 simulations and for each simulation, K,$R_r$,$R_o$,$R_a$,V,$\omega$',$\sigma$,$\Delta\theta$ were calculated till

| Parameter | Description of Parameter | Values |
| --- | --- | --- |
| R | Radius of Agents | 10 |
| N | Number of Agents | 50 |
| K | Number of Leaders | 4 or 6 |
| $R_r$ | Radius of Repulsion | 30 |
| $R_o$ | Radius of Orientation | 120 |
| $R_a$ | Radius of Attraction | 170 |
| V | Speed of agents | 2 |
| $\omega$' | Leader bias towards goal direction | 0.6 |
| $\sigma$ | Stochastic noise | 1.5 |
| $\Delta t$ | Time step | 0.1 |
| $\Delta\theta$ | Number of degrees by which goal direction differs from initial swarm direction | 45 |

**Table 5.1:** Experimental settings for various variables

we found the optimum values that gave best performance. The optimum settings are given in Table 5.1. Thereafter, 100 simulations were carried out for 9 different experiments with each simulation terminating upon every agent in the swarm aligning themselves within 0.2 radians of $\theta$'. The time taken to complete the maneuvre was noted with each time step $\Delta t$ being 0.1 seconds. The simulations were carried out for the following experiments:

1) K=4, Leader strategy 1
2) K=6, Leader strategy 1
3) K=4, Leader strategy 2
4) K=6, Leader strategy 2
5) K=4, Leader strategy 3
6) K=6, Leader strategy 3
7) K=4, Leader strategy 4
8) K=6, Leader strategy 4
9) Leader strategy 5

A simulation was deemed successful if the following conditions were satisfied:
1) All agents aligned their orientation within 0.2 radians of $\theta$' within a timeframe of 10 seconds
2) No agents broke away from the swarm

| Configuration Number | K | Leader Strategy | Number of successful simulations |
|---|---|---|---|
| 1 | 4 | 1 | 51 |
| 2 | 6 | 1 | 74 |
| 3 | 4 | 2 | 80 |
| 4 | 6 | 2 | 80 |
| 5 | 4 | 4 | 89 |
| 6 | 6 | 4 | 91 |
| 7 | Decided by algorithm | 5 | 92 |

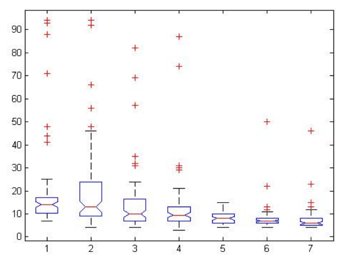
**Table 5.2:** Simulation results

In the experiments it was found that the results for Leader strategy 3 were almost identical to the results achieved with Leader strategy 2, hence, we omitted Leader strategy 3 from the final analysis. The results are shown in Table 5.2.

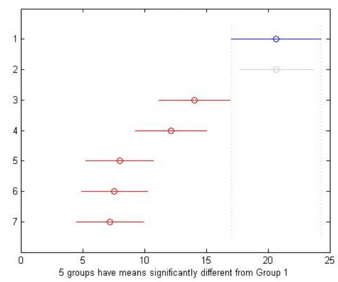
The results were analysed using one-way ANOVA and Multiple Comparisons test which are shown in Figure 5.2. The difference in the mean time taken to change the direction of the swarm from its original heading $\theta$ to the new heading $\theta$' was found to be highly significant with $F_{550}$=16.56 and $p<0.0001$. The mean time was the least for Leader strategy 4 and 5. This is unsurprising as it is expected that higher the number of agents that are directly influenced by the leaders, the faster the swarm will change its orientation. The percentage difference between the mean time taken for Leader strategy 1 and Leader strategy 5 is 65% and in the real world, where speed is crucial, this could be the difference between a success and a failure. Table 5.3 shows the percentage difference between each strategy based on the mean time taken to orient the swarm in a new direction. The result for Leader Strategy 1 is assumed to be the base case.

| Strategy Selected | Percentage reduction in mean time |
|---|---|
| Leader Strategy 1 | 0% |
| Leader Strategy 2 | 30% |
| Leader Strategy 4 | 65% |
| Leader Strategy 5 | 65% |

**Table 5.3:** Comparison of mean time for each Leader Strategy



(a)




(b)

**Figure 5.2:** (a) One-way ANOVA results with X axis representing the configuration in Table 5.2 and Y axis representing the time taken to complete the simulation (b) Multiple Comparisons results with X axis representing the mean time taken and Y axis representing the configuration in Table 5.2

In addition to this, we carried out experiments with an obstacle in the environment to observe if the swarm can successfully navigate around an obstacle after being influenced by the leaders. Previously we were concerned with how long it would take to maneuvre a swarm to an orientation of $\theta$'. Now, we are concerned with how long it will take for the entire swarm to navigate around the obstacle and whether the swarm integrity is maintained. We created a large obstacle in the path of the swarm along orientation $\theta$' with radius of influence = $2R_r$. We noted the time taken from the first agent being repelled by the obstacle to the last agent passing by the obstacle. The obstacle applies potential field concepts to determine by how much the agents should be repelled from it. The formulas used for the repulsion effect of the obstacle were taken from the formulas provided by Goodrich [37]. We used a combination of the formulas to simulate attraction behaviour towards $\theta$' and repulsion behaviour away from the obstacle. Thus, when the agent strikes the obstacle, it follows a radial path around the obstacle towards $\theta$'. To determine whether the agent had to go along the radial path in clockwise or anti-clockwise direction, we determined at what angle the agent struck the obstacle with respect to $\theta$' using

Algorithm 1, given in table 5.4, which can be visualised in Figure 5.3.

| **Algorithm 1** |
|---|
| $\eta$ = Angle between $\theta$ and obstacle |
| $\eta 1$ = Angle between $\theta$ and agent |
| $\phi$ = $\eta 1$ - $\eta$ |
| if $\phi < 0$ |
| turn anti-clockwise |
| else |
| turn clockwise |
| end if |

**Table 5.4:** Details of Algorithm 1

Considering that we can observe the effect of the obstacle on the swarm only if the swarm has been successful in orienting itself towards $\theta$' from $\theta$, we consider only the sets that gave us successful simulations previously. The results are shown in Table 5.5.

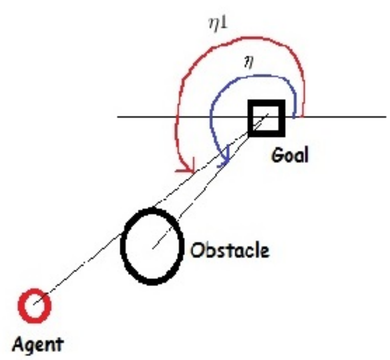


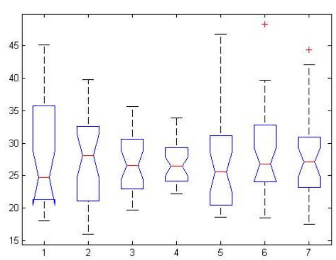

**Figure 5.3:** Visualising Algorithm 1

The results were analysed using one-way ANOVA and Multiple Comparisons test which are shown in Figure 5.4. The difference in the mean time taken for the entire swarm to successfully go around the obstacle was found to be insignificant with $F_{123}$=0.14 and p<0.9. The mean time taken was between 27-28 seconds for all the Leader configurations. But of greater concern is the low success rate in navigating around the obstacle. In many simulations, it was observed that when agents were getting repulsed away from the obstacle, they ended up in the path of other agents and inside their ZOR. This led to the agents applying the repulsion rules so as to maintain the minimum required distance between each other and this had a cascading

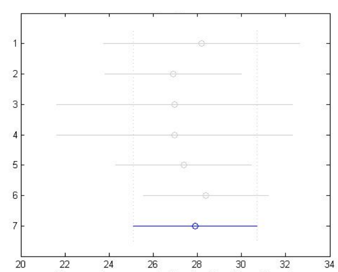
| Configuration Number | K | Leader Strategy | Number of successful simulations |
|---|---|---|---|
| 1 | 4 | Leader strategy 1 | 11 (22% success rate) |
| 2 | 6 | Leader strategy 1 | 23 (31% success rate) |
| 3 | 4 | Leader strategy 2 | 8 (10% success rate) |
| 4 | 6 | Leader strategy 2 | 8 (10% success rate) |
| 5 | 4 | Leader strategy 4 | 23 (26% success rate) |
| 6 | 6 | Leader strategy 4 | 28 (31% success rate) |
| 7 | Decided by algorithm | Leader strategy 5 | 29 (32% success rate) |

**Table 5.5:** Simulation results with obstacle

effect on the rest of the agents that were close to them. This ultimately affected the stability of the swarm and prevented the entire swarm from getting through the obstacle without any change in their orientation.



(a)




(b)

**Figure 5.4:** (a) One-way ANOVA results with X axis representing the configuration in Table 5.5 and Y axis representing the time taken to complete the simulation (b) Multiple Comparisons results with X axis representing the mean time taken and Y axis representing the configuration in Table 5.5

# Chapter 6

# Conclusion

We proposed several network discovery strategies for multi-robot deployments in a human-centered post-disaster scenario. The results show that the Convex Lévy model using Lévy walk with memory and convex hull is the best model for efficient network discovery. We have carried out statistical and Bayesian simulation analysis for different types of strategies under different metrics - search time, number of times agents entered into no-network zone, number of tasks serviced, and the total time spent in no-network zone, and time taken in searching for the network on every visit.

We have also shown that it is possible to guide a swarm to a goal by making use of existing agents within the swarm with the use of Couzin's model. By utilising existing agents as leaders, we can influence the rest of the uninformed agents to change their orientation towards a desired goal. We have shown the different strategies for selection of Leaders from within the agents in a swarm. Based on these Leader strategies, we have carried out ANOVA and Multiple Comparison tests to show the relative capabilities of these strategies in terms of the time taken to change the orientation of the swarm without disturbing the stability of the swarm.

If we place an obstacle in the path of the swarm, then the mean time taken to go around the obstacle is almost independent of the Leader strategy that is adopted. However, the success rate was found to be quite low with the best being around 30%. This low success rate in navigating around the obstacle and maintaining the swarm integrity is a cause for concern. Further studies need to be done to determine how to improve upon the success rate.

The present work has been based on several assumptions as indicated in Chapters 4 and 5. The study could, in future, be expanded by introducing new variables, or modifying the assumptions as made in this thesis, in order to come up with a more comprehensive analysis.

In future, we would also like to integrate a swarm model by deploying automatic selection of leaders with the network discovery methods of multi-robot deployments as outlined in Chapter 4 of this thesis. This would allow us to perform automatic selection of the leaders and deploy them to guide the swarm towards the tasks selected by the human operator. One could also further extend this approach, and introduce the influence of a mediator to split the swarm into sub-swarms which could then concurrently carry out different tasks thereby reducing the overall time taken to achieve the goal of the operator.

# Bibliography


[1] P. Basu and J. Redi. **Movement control algorithms for realization of fault-tolerant ad hoc robot networks**. *IEEE Network*, **18**(4):36–44, July 2004. 1

[2] P. Ulam and R.C. Arkin. **When good communication go bad: communications recovery for multi-robot teams**. In *Robotics and Automation, 2004. Proceedings. ICRA'04. 2004 IEEE International Conference on*, **4**, pages 3727–3734. IEEE, 2004. 1

[3] T. Balch and R.C. Arkin. **Communication in reactive multiagent robotic systems**. *Autonomous Robots*, **1**(1):27–52, 1994. 1

[4] L. Pollini, F. Giulietti, and M. Innocenti. **Robustness to Communication Failures within Formation Flight**. In *Proc. of the American Control Conference*, pages 2860 – 2866, 2002. 1

[5] R.K. Mehra, J.D. Boskovic, and Sai-Ming Li. **Autonomous formation flying of multiple UCAVs under communication failure**. In *Proc. of the IEEE Position Location and Navigation Symposium*, pages 371 – 378, 2000. 1

[6] J. Shin, H.J. Kim, S. Kim, and Y. Yoon. **Formation flight control under communication failure**. In *Proceedings of the 1st international conference on Robot communication and coordination*, RoboComm '07, pages 47:1–47:4, Piscataway, NJ, USA, 2007. IEEE Press. 1

[7] D. Dionne and C. A. Rabbath. **Multi-UAV Decentralized Task Allocation with Intermittent Communications: the DTC algorithm**. In *Proc. American Control Conf. ACC '07*, pages 5406–5411, 2007. 1

[8] P. B. SUJIT AND J.B. SOUSA. **Multi-UAV task allocation with communication faults**. In *American Control Conference (ACC), 2012*, pages 3724–3729, 2012. 1

[9] A. WAGNER AND R.C. ARKIN. **Multi-robot communication-sensitive reconnaissance**. In *Robotics and Automation, 2004. Proceedings. ICRA '04. 2004 IEEE International Conference on*, **5**, pages 4674–4681 Vol.5, 2004. 1

[10] J. HAN, M. LI, AND L. GUO. **Soft control on collective behavior of a group of autonomous agents by a shill agent**. *Systems Science and Complexity*, 2006. 2, 5

[11] SHIN-YOUNG JUNG AND M. A. GOODRICH. **Multi-Robot Perimeter-Shaping through Mediator-Based Swarm Control**. *IEEE*, 2013. 2, 5

[12] A. KOLLING, S. NUNNALLY, AND M. LEWIS. **Towards Human Control of Robot Swarms**. *HRI*, 2012. 2, 5

[13] S.A. AMRAII, P WALKER, M. LEWIS, N. CHAKRABORTY, AND K. SYCARA. **Explicit vs Tacit Leadership in Influenceing the Behaviour of Swarms**. *IEEE*, 2014. 2, 5

[14] A.M. REYNOLDS. **Cooperative random Levy flight searches and the flight patterns of honeybees**. *Physics Letters A*, **354**(5-6):384 – 388, 2006. 3, 7

[15] G.M. VISWANATHAN, S.V. BULDYREV, S. HAVLIN, M.G. DA LUZ, E.P. RAPOSO, AND H.E. STANLEY. **Optimizing the success of random searches.** *Nature*, **401**(6756):911–4, October 1999. 3

[16] S.G. NURZAMAN, Y. MATSUMOTO, Y. NAKAMURA, S. KOIZUMI, AND H. ISHIGURO. **Biologically inspired adaptive mobile robot search with and without gradient sensing**. In *2009 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 142–147. IEEE, October 2009. 3

[17] S.G. NURZAMAN, YOSHIO MATSUMOTO, Y. NAKAMURA, S. KOIZUMI, AND H. ISHIGURO. **Yuragi-based adaptive searching behavior in mobile robot: From bacterial chemotaxis to Levy walk**. In *2008 IEEE International Conference on Robotics and Biomimetics*, pages 806–811. IEEE, February 2009. 3

[18] D.K. SUTANTYO, S. KERNBACH, P. LEVI, AND V. A. NEPOMNYASHCHIKH. **Multi-robot searching algorithm using Lévy flight and artificial potential field**. In *2010 IEEE Safety Security and Rescue Robotics*, pages 1–6. IEEE, July 2010. 3

[19] Z. CHENG AND R. SAVIT. **Fractal and nonfractal behavior in Levy flights**. *Journal of Mathematical Physics*, **28**(3):592, 1987. 3, 8

[20] E. SAHIN. **Swarm Robotics: From sources of inspiration to domains of application.** *Lecture Notes in Computer Science*, 2005. 4

[21] M.A. GOODRICH, B. PENDLETON, P.B. SUJIT, J. PINTO, AND J.W. CRANDALL. **Toward Human Interaction with Bio-inspired Teams (Extended Abstract)**. *AAMAS*, 2011. 4

[22] E. BONABEAU, M. DORIGO, AND G. THERAULAZ. **Swarm Intelligence From Natural to Artificial Systems**. *Oxford University Press*, 1999. 4

[23] C.W. REYNOLDS. **Flocks, herds and schools: A distributed behavioral model**. *SIGGRAPH*, 1987. 4

[24] I.D. COUZIN, J. KRAUSE, R. JAMES, G.D. RUXTON, AND N.R. FRANKS. **Collective Memory and Spatial Sorting in Animal Groups**. *Elsevier Science Ltd*, 2002. 4, 18, 20

[25] T. VICSEK, A. CZIROK, E. BEN-JACOB, I. COHEN, AND O. SOCHET. **Novel type of phase transition in a system of self-driven particles**. *PHYS REV LETT*, 1995. 4, 22

[26] A. JADBABAIE, J. LIN, AND A. MORSE. **Coordination of groups of mobile autonomous agents using nearest neighbor rules**. *IEEE Transactions on Automatic Control*, 2003. 4

[27] H. SU, X. WANG, AND Z. LIN. **Flocking of multi-agents with a virtual leader**. *IEEE Transactions on Automatic Control*, 2009. 4

[28] H. CELIKKANAT AND E. SAHIN. **Steering self-organized robot flocks through externally guided individuals**. *Neural Computing and Applications*, 2010. 4

[29] K. GENTER, S. ZHANG, AND P. STONE. **Determining Placements of Influencing Agents in a Flock**. *AAMAS*, 2015. 4

[30] I.D. COUZIN, J. KRAUSE, N.R. FRANKS, AND S.A. LEVIN. **Leadership by numbers: Decision making in animal groups**. *Nature Publishing Group*, 2005. 4, 22

[31] Shin-Young Jung, M. A. Goodrich, and D. S. Brown. **Shaping Couzin-like Torus Swarms through Coordinated Mediation**. *IEEE*, 2013. 5

[32] B. Pendleton and M. A. Goodrich. **Scalable Human Interaction with Robotic Swarms**. *American Institute of Aeronautics and Astronautics*, 2013. 5

[33] O. Khatib. **Real-time obstacle avoidance for manipulators and mobile robots**. *The International Journal of Robotics Research*, 1986. 5

[34] H.G. Tanner. **Flocking with Obstacle Avoidance in Switching Networks of Interconnected Vehicles**. *IEEE International Conference on Robotics and Automation*, 2004. 5

[35] R. Olfati-Saber and R.M. Murray. **Flocking with obstacle Avoidance: Cooperation with limited communication in mobile networks**. *IEEE Conference on Decision and Control*, 2003. 5

[36] D.E. Chang, S. Shadden, J. Marsden, and R. Olfati-Saber. **Collission avoidance for multiple agent systems**. *IEEE Conference on Decision and Control*, 2003. 5

[37] M.A. Goodrich. **Potential Fields Tutorial**. 2000. 5, 25

[38] I. Rhee, M. Shin, S. Hong, K. Lee, S.J. Kim, and S. Chong. **On the Levy-Walk Nature of Human Mobility**. *IEEE/ACM Transactions on Networking*, **19**(3):630–643, June 2011. 7

[39] G. Ramos-Fernandez, J.L. Mateos, O. Miramontes, G. Cocho, H. Larralde, and B. Ayala-Orozco. **Lévy walk patterns in the foraging movements of spider monkeys (Ateles geoffroyi)**. *Behavioral Ecology and Sociobiology*, **55**(3):223–230, January 2004. 7

[40] G. M. Viswanathan, V. Afanasyev, S. V. Buldyrev, E. J. Murphy, P. A. Prince, and H. E. Stanley. **Lévy flight search patterns of wandering albatrosses**. *Nature*, **381**(6581):413–415, May 1996. 7

[41] A. Hristov. **http://www.ahristov.com/tutorial/geometry-games/convex-hull.html**. 2010. 19

[42] S.K. Ganjukunte. **Geometric Hitting Sets and Their Variants**. *Duke University*, 2011. 19

[43] P. Maini. **A Distributed Strategy for Human-in-the-loop Task Servicing using Multiple Robots with Stationary Base Station Connectivity Constraint**. *IIIT Delhi*, 2014. 19