A new multi-robot search algorithm using probabilistic finite state machine and Lennard-Jones potential function

Md. Shadnan Azwad Khan - 13321076

Supervisor: **Dr. Mohammad S. Hasan**

Co-supervisor: **Dr. Tarem Ahmed**

Submitted to:

Department of Computer Science and Engineering

School of Engineering and Computer Science

BRAC University

Declaration

I, hereby declare that this thesis project, neither in whole or in part, has been previously submitted for any degree. It has been completed based on the knowledge we have acquired and implemented ourselves. Materials of work accomplished by others and the references gained from the official websites are mentioned by reference.

____

Md. Shadnan Azwad Khan

Under the supervision of,

____

Dr. Mohammad S. Hasan

Senior Lecturer

Faculty of Computing, Engineering and Science

Staffordshire University

And co-supervision of,

____

Dr. Tarem Ahmed

Assistant Professor

Department of Computer Science & Engineering

Independent University, Bangladesh

Acknowledgement

I have been, throughout this thesis, fortunate to have the support of our friends, family and mentors. In particular, I would like to thank our supervisor Dr. Mohammad S. Hasan and co-supervisor Dr. Tarem Ahmed for guidance, patience and for taking immense interest in the work. I am grateful for the support from the esteemed faculties and technicians of the School of Engineering and Computer Science. This thesis would not have been possible without their help. The author would also like to acknowledge an Erasmus+ Inter-national Credit Mobility (ICM) fund for Bangladesh awarded to Dr. Mohammad Hasan at Staffordshire University, UK in 2016.

Abstract

Swarm robotics is a decentralized approach to robotic systems. This paper examines the problem of search and rescue using swarm robots. We present as solution a multi-robot search algorithm using probabilistic finite state machine and interaction inspired by Lennard-Jones potential function. The approach utilizes a finite state machine to separate the tasks performed and to change coordination rules according to the circumstances and social probabilities. The approach is tested in various scenarios to test flexibility, scalability and robustness. The performance results are promising and comparison with Robotic Darwinian Particle Swarm Optimization and Glowworm Swam Optimization for algorithmic complexity appear favourable.

Keywords: autonomous robots, multi-robot systems, performance analysis, search and rescue, swarm intelligence

Content

List of Figures

List of Tables

List of Algorithms

Chapter 1

Introduction

Within multi-robot systems swarm robotics is a novel approach, taking as inspiration biological swarms like social insects (bees, ants, or termites), fish schools, bird flocks, or bacteria colonies [1]. It is a decentralised approach to robotics that is based on robustness, flexibility, and scalability [2]. From a computation point of view, swarm intelligence simulates the overall behaviour of the swarm and not the individuals in the swarm it is trying to mimic. Specifically, it is the emergent collective behaviour in decentralized groups of autonomous robots with individuals in the swarm following simple local rules which can produce largely varied and complex behaviour for the swarm [1][3]. The robots themselves are relatively simple in design with limited range of sensors or actuators. The communication between robots is local and limited.

Multi-robot systems essentially have three different coordination paradigms: centralised architecture where decision-making is under control of a single entity, decentralised hierarchical architecture where decision-making is based on negotiated through a hierarchy system implemented locally, and decentralized distributed architecture where each entity is autonomous and decision-making is completely decentralized according to [4]. Multi-robot system is used in the current state of the art in the field of search and rescue for the reconnaissance and rescuing phases of a search and rescue mission as described in [5] to assist human responders and it has been found that distributed approaches avoid bottlenecks due to overflows in communication links as in centralized approaches. Furthermore, in [6] four types of interactions between distributed systems are described: collective where goals are shared and actions advance goals of others but entities are not aware of each other, cooperative where goals are shared and actions advance goals of others and entities are aware of each other, collaborative where entities are aware of each other and advance goals of others but have individual goals, and coordinative where entities are aware of each other and have individual goals but do not advance goals of others.

Global coordinating systems or absolute positioning is not usually implemented for swarms and on-board sensors are very important in methods to locate other robots [7]. Relative positioning techniques are more widespread in the implementations of swarm robots according to [1] and has been implemented in [8] where on-board sensors are used. Relative positioning allows individuals in a swarm to keep their own local coordinates to position themselves.

There are several tasks within the field of swarm robotics or swarm intelligence that are widely researched or areas of interest such as mapping, exploration, foraging, morphogenesis or pattern formation, and search tasks [9]. This paper deals with the problem of search and rescue. Autonomous robots finding targets within an unknown environment is a problem that is suitable for a swarm of robots. The area or environment in question can be dangerous or inaccessible to humans or robots could be deployed as secondary operation aiding humans. A swarm robotics approach has some advantages compared to single robots. It is vastly more efficient and robust in its execution due to a parallel autonomous behaviour of the individual in the swarm and the scalable nature of the swarm itself [1]. Sensory information accrued by multiple robots allow for optimization techniques to be used that improve the solution significantly [10]. Given all its advantages, search and rescue solutions using swarm robotics are relatively few leaving a wide possibility for further research. This paper aims to propose a new algorithm for a swarm of robots carrying out search and rescue operations.

1.1 Thesis Outline

This thesis is organized into several sections:

1. In section II, past research work and related papers are discussed.
2. Section III introduces and explains the proposed algorithm.
3. Section IV describes the simulation experiment and the procedures related to it.
4. In section V, the results of the experiment are compared with the algorithms
5. Finally, section VI summarizes the paper and discusses the performance of the algorithm along with limitations in our experiment.

Chapter 2

Literature Review

Particle swarm optimization (PSO) first proposed in [11] is an evolutionary computation technique that is inspired by social behaviours of foraging swarms. It is a robust and flexible approach that utilises individual fitness to maximise global performance. It considers present states and best performance in the swarm and past best performances of individuals to move towards an optimised solution towards the goal state. Due to its simplicity and low time and space complexity, PSO is easy to implement and has been adapted for swarm robotics despite being created as a solution to optimisation or estimation problem and has been shown to be an efficient algorithm for many applications [12].

There are many examples of search and rescue approaches in which unknown environments were traversed to locate targets at unknown locations such as in [13] where PSO was used with adaptive RSS weighting factor. Another method was shown in [14] with a search algorithm inspired by PSO used to find targets without precise global information where cartesian geometry was used to unify relative coordinate systems to improve robustness and efficiency.

A distributed approach to multi-robot search was proposed in [15] where PSO was modified inspired by chemotaxis behaviour in bacteria. The approach is tested on dynamic environments for the fitness of individuals and for the swarm globally. Local adaptations based on varying neighbourhood sizes were used to test for the change in global fitness achieved through local interactions. The results show that the approach is adaptive in dynamic environment and continues adaptations throughout changes in the environment without loss in performance.

A study in [12] conducted benchmark experiments for multi-robot search algorithms inspired by swarm intelligence. Five state-of-the-art algorithms are compared using the non-realistic simulator MRSim. The performance is measured using the exploration ratio of the environment and its average of 500 iterations. Robotic Darwinian PSO (RDPSO) is shown to have the best performance in the simulated experiments. Moreover, the RDPSO is further compared with two other best performing algorithms, aggregations of foraging swarm (AFS) and glowworm swarm optimisation (GSO), using a swarm of 14 e-pucks. RDPSO converges to the optimal solution faster and with accuracy GSO closely follows its performance. For RDPSO and GSO, the computational load due to space and time complexities or the communication demands are not significantly higher than the other algorithms.

RDPSO was first proposed in [16] along with robotic PSO (RPSO) as extensions of Darwinian PSO (DPSO) and PSO respectively. The techniques were modified for obstacle avoidance and for multi-robot systems. A simulation demonstrates these algorithms performing distributed exploration. The techniques use dynamic topology to split the swarms into sub-swarms over several iterations. There arises a chance of getting stuck in local minimum that is avoided in the RDPSO but not in the RPSO. RDPSO outperforms RPSO by avoiding local optima using a punish-reward mechanism controlling social exclusion and social inclusion. Therefore, global communication and coordinating system considerations outweigh distance metrics and local minimum when dividing the sub-swarms.

GSO was introduced in [17] as an optimization algorithm that was like but distinct from PSO and Ant Colony Optimisation (ACO). The entities in GSO are thought of as glowworms that carry a fitness value calculated based on their current location called luciferin which they broadcast to their neighbours. An individual in the swarm computes its movements based on an adaptive neighbourhood where it probabilistically moves towards a neighbour with a higher luciferin value. The swarm divides into disjoint groups die to these movements based on local information and selective neighbours allowing it to move towards multiple signal sources.

Foraging robots could use path planning in their environment for efficiency. In [18] virtual ants are implemented that use artificial pheromones within a swarm network. This is achieved by local messages forming chains within the robots deposited with artificial pheromone that help with path selection. The method is tested on 20 real robots and it is shown that the approach can do path selection for more advantageous paths when possible. The robots have minimal abilities since they are only required to use simple communication and behaviours.

A search approach using potential field is shown in [19] where a model based upon Coulomb's inverse-square law is used. The system uses positive charges as obstacles pickup up by sensors and being positive themselves the robot move away from obstacles. The target is also positive since it appears as an obstacle to the sensors and while they are still denoted as a positive charge they are identified using a camera. Once a target has been identified the task is said to be completed. Information sharing could be either pessimistic or optimistic where a robot takes information about an obstacle given by another robot and selects a high charge if pessimistic and low if optimistic. The approach is tested on Player/Stage simulator and the sharing systems are found to have similar performance but both outperform systems without sharing.

Probabilistic finite state machines (PFSM) are also used for swarm robotics applications such as in [20] where a foraging swarm is modelled using PFSM with state searching, grabbing, homing, avoidance, deposit and resting. The mathematical aspects are modelled to mimic macroscopic behaviour while geometric methods are used to derive transitional probabilities of the individuals. Player/Stage simulations of the model show promising results for dynamic situations. PFSM offer flexibility in design of the approach and it also very easy to implement, therefore, suiting swarm robotics.

Chapter 3

The Proposed Algorithm

The algorithm utilises a PFSM where states may change due to three different reasons:

(i) Time boundaries: a maximum time boundary changes the state taking transitional probabilities into consideration as well.

(ii) Transition probabilities: these are generated randomly and updated through social interaction between the swarm due to various scenarios and taking boundaries conditions also into consideration.

(iii) Task events: events pertaining to the swarm task automatically changes the state of the individual robots concerned.

The state transitions are therefore nondeterministic up to an extent allowing for dynamic responses. The states used for this algorithm are: Timeout, Search, and Return. The general algorithm is shown in Algorithm 1. State Transition Diagram pertaining to the state changes are shown in Figure 1.

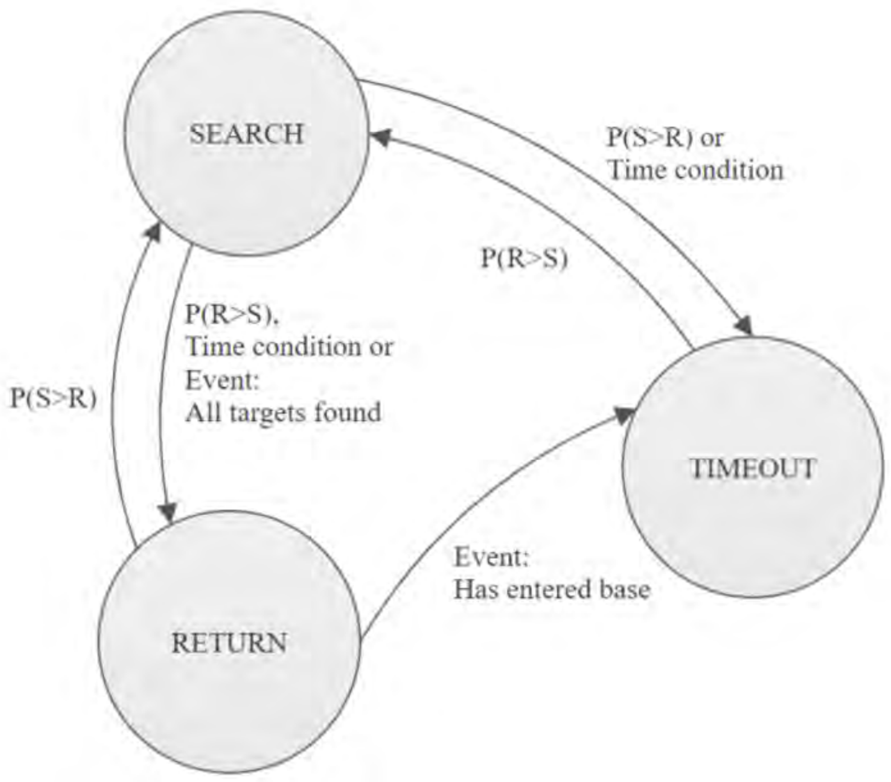

Figure 1: State transition diagram for proposed algorithm

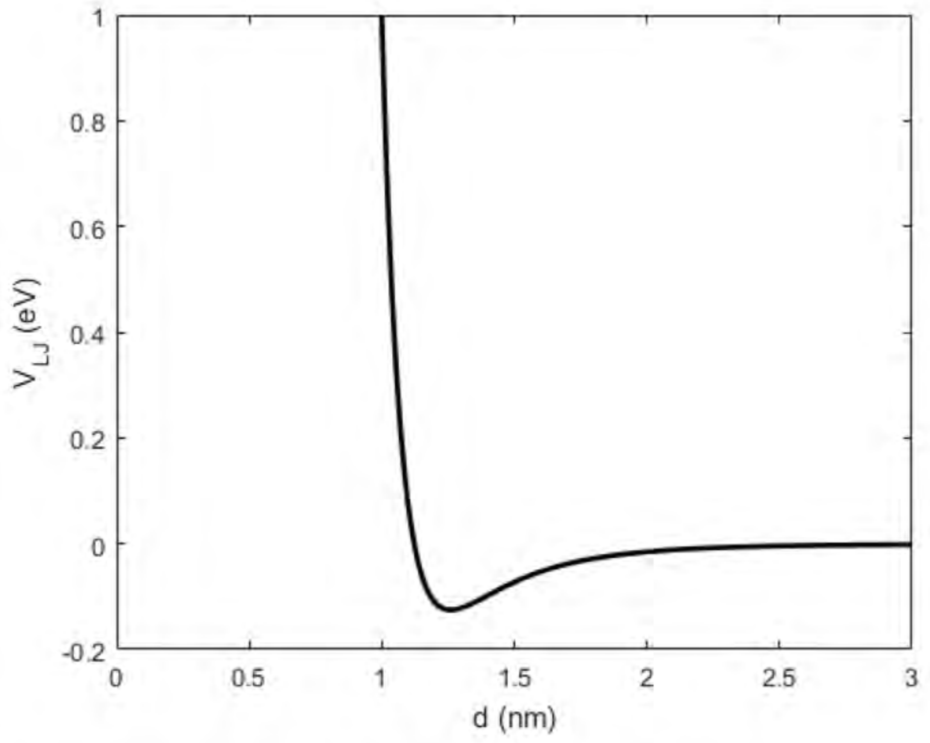

Figure 2: Lennard-Jones potential V_{LJ} of two particles over distance d

Algorithm 1: Lennard-Jones potential probabilistic state machine multi-robot search algorithm for robot n

```
1. Initialise pose <$x_n[0]$, $\varphi_n[0]$> and randomly generated transitional probability set
   $P_T = \{P_{i,j}$ : where i and j are current and future states$\}$
2. Set State = Search
3. Loop (until boundary conditions or all robots found):
4. Set wheel velocity using motion vector
5. Update $P_T$ using neighbouring robot information
6. If (State = Search)
7. If ($T_{SEARCH} < MaxTime_{SEARCH, RETURN}$ and $P_{RETURN, SEARCH} > P_{SEARCH, RETURN}$)
8. Update motion vector using Lennard-Jones potential with neighbouring robots $N_s$
   and search vector towards closest target
9. If obstacle in path
10. Perform obstacle avoidance
11. If near target
12. Give target energy and Set $State_{TARGET}$ = Return
13. TargetAcquired += 1
14. Else
15. Set State = Return
16. If (State = Return)
17. Update motion vector using vector towards base
18. If obstacle in path
19. Perform obstacle avoidance
20. If position is in base
```

```
21. Set State = Timeout
22. If (State = Timeout)
23. If ($T_{TIMEOUT}$ < $MaxTime_{TIMEOUT, SEARCH}$ and $P_{SEARCH, RETURN}$ > $P_{TIMEOUT, SEARCH}$)
24. Set motion vector to zero
25. Randomly generate new set $P_T$
26. Else
27. Set State = Search
28. End Loop
```

3.1 Timeout

This state essentially does a timeout for the robot when it is already back in the initial area or base. It does a clean-up of the transition probabilities and after a given time has passed or if other robots have interacted in the meantime allowing for updates in transition probabilities as the robot in timeout to gathers information and interprets them. Depending on probabilities this state transitions from Return and to Search.

3.2 Search

The main task of the swarm is to search the environment for target robots. It must not only search its environment it must be done in a distributed way which is also optimised by some method.

For this task, the robots in the state of Search work out Lennard-Jones potential its neighbouring search robots and calculates a target distance between itself and its neighbours. Two-dimensional motion vectors are calculated using a generalised form of the Lennard-Jones potential which is modified for maximum distance between individuals as they search for the targets. The Lennard-Jones potential function is usually expressed as,

$$V_{LJ} = 4\varepsilon\left[\left(\frac{\sigma}{d}\right)^{12} - \left(\frac{\sigma}{d}\right)^{6}\right] \qquad (1)$$

Where the potential well is defined by ε, the inter-particle distance where potential is zero by σ, the distance between particles by d. Figure 2 shows the graph for Lennard-Jones potential between two charged particles.

The targets themselves are given the higher potential so that each target can be optimally visited by its closest search robots. Finding robots in a neighbourhood changes the transitional probabilities of the neighbouring swarm robots.

There is also an obstacle avoidance method in place so that robots never come close enough due to the potential function. The targets once found, are given energy to go back to the base. Figure 1 shows the transition of this state while lines 6 to 15 show in further detail the method implemented to search for the targets.

3.3 Return

Allowing the normal rules of time boundaries and transitional probabilities in effect, returning occurs for the whole swarm only when all the targets have been found and returned to the base or the global time boundary is crossed. Upon transition to the state of Return, a robot calculates its vector to the base and adds it to an obstacle avoidance vector given an obstacle appears in its path. It takes the shortest path back considering its environment is dynamic and has other moving robots in it. Obstacle avoidance due to interactions between robot's updates transition probabilities. Upon returning to base the state transitions to Timeout.

Chapter 4

Simulation and Experimental Setup

4.1 ARGoS overview

ARGoS as outlined in [20] is a modular, pluggable, multi-physics engine simulator capable of simulating large heterogeneous swarm robotics in real time with efficiency and flexibility in its design. It can use multiple threads and multiple physics engines and robots can move freely from the simulated space of one engine to another with ease and transparency. Distinctly, ARGoS is implemented in a way that every entity is implemented as a plug-in with easy interface to include custom plug-ins. In simulations, it has been able to simulate up to 10,000 wheeled robots in real-time with full dynamics in place.

A controller interacts with the sensors and actuators of a robot through a control interface. The sensors and actuators, in turn, measure or change the entities in the simulated 3D space.

Loop functions or hooks designed to occur at certain events further modify or interact with the simulated space while physics engines run the mechanics of the simulation and visualization module renders the graphical display of it. A simulation configuration file is used to set up the simulation using all the modules and plug-ins as required. Figure 3 shows the design aspects of ARGoS.

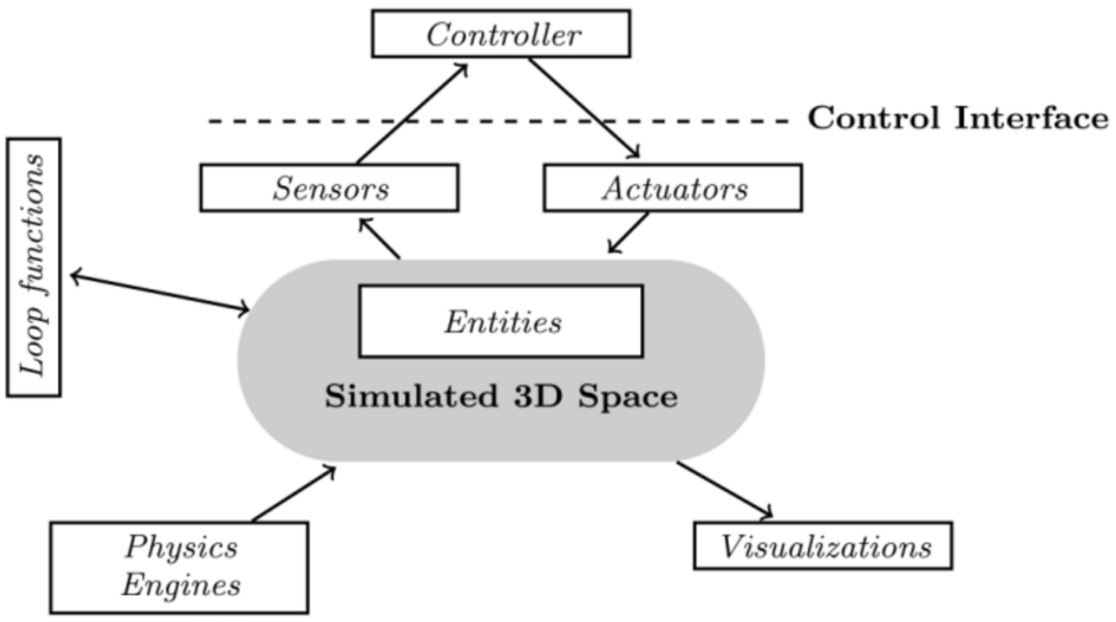

Figure 3: The modular design of ARGoS [20]

4.2 Swarm task definition

The goal of the swarm for search and rescue for the simulation is defined as identifying, locating, approaching and returning the targets to initial location of the search and rescue robots. With ending criteria for successful completion being discovery of all target footbots dispersed in the environment.

4.3 Robot model and control

The footbot modelled after the marXbot [21] is already implemented in ARGoS and is used for the simulation for the validation of the proposed approach. The sensors and actuators in the footbot include RFID read and write device, ceiling or frontal camera, gripper, etc. However, the simulation utilises the following sensors and actuators from the footbot:

- Differential steering actuator
- Omnidirectional camera
- Range and bearing sensor
- Range and bearing actuator
- Ground sensor
- Proximity sensor

- Light sensor
- LED actuator

The controller used for the footbot comes with predefined structures inherited from ARGoS. The controller is required to specify a control method that is called every time step as well as an initialising method. Methods are also required to reset or destroy data or memory for the controller. Other than that, the controller can have any number of secondary methods to aid the control. For the case of this paper, methods for the PFSM function and the behaviour for each state including motion control and optimisation have been designed.

4.4 Simulation setup and Data logging

Footbot numbers and environment area are chosen as deemed adequate for task and for the simulation being run by ARGoS. To test for different scenarios and their effects on the proposed approach some variables are selected, these include:

- Different environments designed to test flexibility (for 5 by 5m, 7.5 by 7.5m and 10 by 10m environments and corresponding areas of 25, 56.25 and 100m^2)
- Search footbot numbers which are varied to test scalability (tested for 10, 20, 30 search footbots)

- Target footbot numbers that are varied to test robustness (tested for 10, 20, and 30 target foot-bots)

The data logged for each scenario using loop functions are:

- Energy of the swarm at each time step
- Time taken to locate each target
- Total target footbots located

The simulation is carried out for each scenario from changing the variables above. The simulation design is as follows:

(i) The environment is fixed, and footbot number is fixed while target numbers are changed

(ii) If environment has not been tested with full range of footbots, the footbot number is changed and simulation started from step (i) again

(iii) If another environment is left, select new environment and start from step (i) again

(iv) End simulation and collect data for each separate simulation scenario

Once concluded, the data collected is then used for performance analysis and to test the different factors associated with the variables. The performance is analysed using the following equation where higher values mean better performance,

$$P_{STA} = \frac{1000}{E_{total}T_{total}} \times \frac{(N_{T,found})^2}{N_{T,total}} \times \frac{A\,/\,A_1}{N_{S,total}} \qquad (2)$$

Where $N_{T,\,total}$ is the total number of targets, E_{total} is the total energy expended by swarm after finding the last target, T_{total} is the time taken in seconds to find the last target, $N_{T,\,found}$ the targets that have been found, A the explorable area of the environment, and $N_{S,\,total}$ the total number of robots in the swarm. Each scenario is analysed using the performance measure P_{STA}. A_1 is 1 m^2 making P_{STA} unitless.

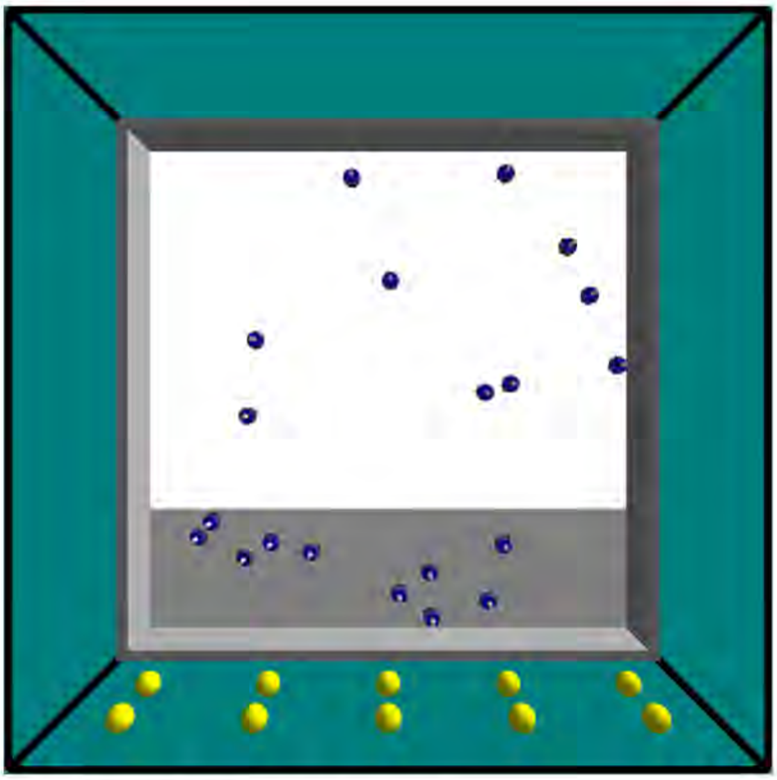
Figure 4: Map 1 with 10 search and rescue bots each

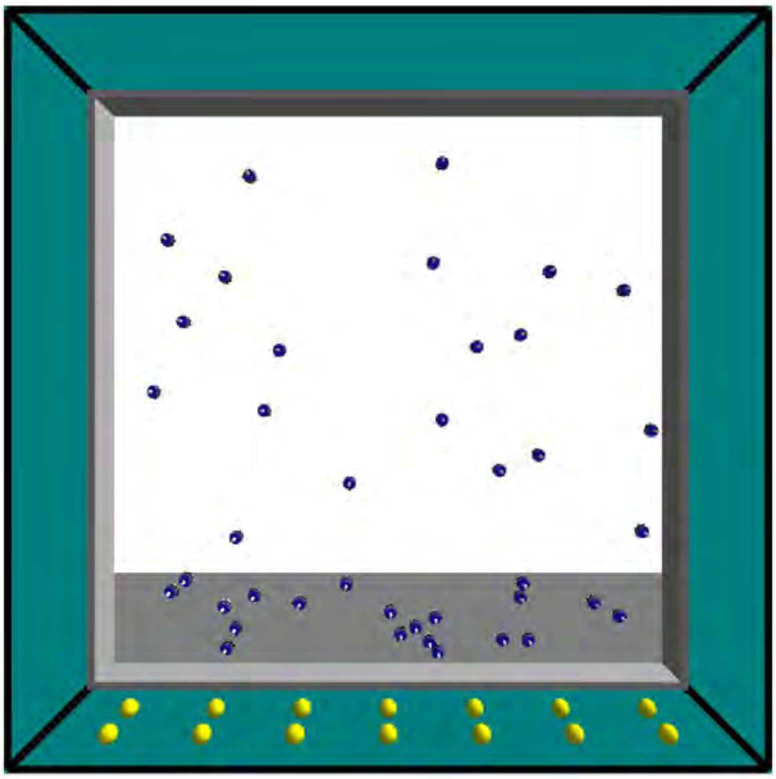
Figure 5: Map 2 with 20 search and rescue bots each

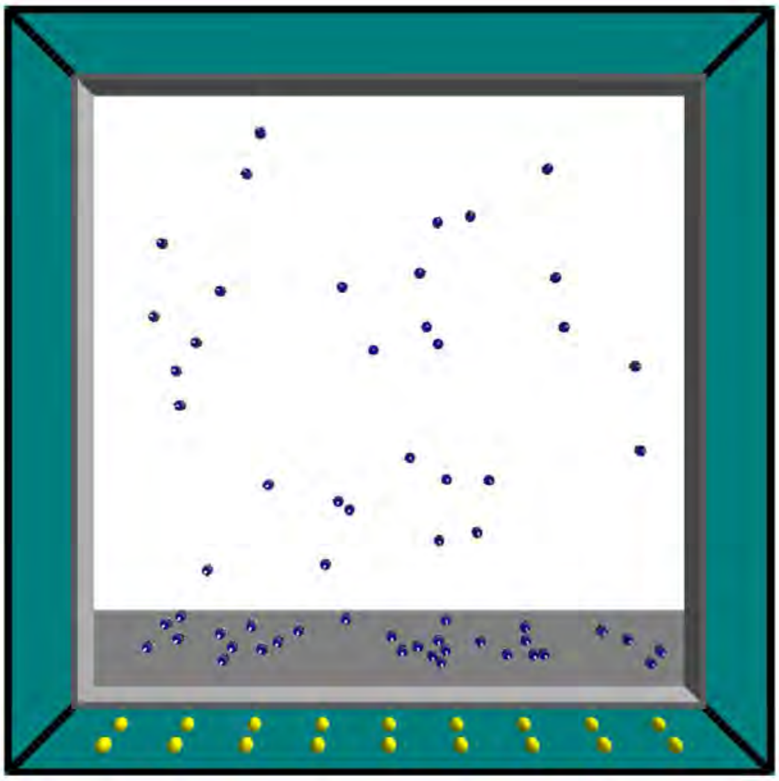

Figure 6: Map 3 with 30 search and rescue bots each

Chapter 5

Experimental Results and Evaluation

The algorithm is tested on 27 scenarios with performance measures for each. To test flexibility, the environment is kept fixed while search and target robots are varied giving 9 scenarios for each environment. The performance is averaged for each environment, likewise the same is done for scalability and robustness test using search and target robots. Space, time, and communication complexities are also compared to RDPSO and GSO (the best performing algorithms as shown in [11]). Other aspects such as obstacle avoidance and sub-optima avoidance are also compared.

5.1 Flexibility

Three different environments are used of areas 25, 56.25, and 100m^2. The results shown in Figure 7 show that performance is highest for smaller environments. This is anticipated since the communication range and interaction vectors benefit from a shorter range, however, it is surprising to see the performance rise for the largest search area. One possibility is that the larger area allows for more distributed search manoeuvres that smaller areas do not. Smaller portioned groups arise in the larger areas allowing for less computational complexity.

5.2 Scalability

Figure 8 shows the average performance for the varying swarm sizes of 10, 20 and 30. The best case is for the lowest swarm size, however, after a dip at 20, it seems to be plateauing around 30. It has a positive slope near the end. The algorithm does execute with all the robots discovered by the end and appears scalable for the most. The swarm had a difficult time avoiding sub-optima for the swarm size of 20 which might explain its poorer performance.

5.3 Robustness

Varying the target size has a very noticeable and apparent effect, Figure 9 shows a very clear positive slope with an increase in performance with increasing target size. This is due to the motion vectors depending on targets in some situations to escape sub-optima due to an aspect of the Lennard-Jones potential. The proposed algorithm proves robust.

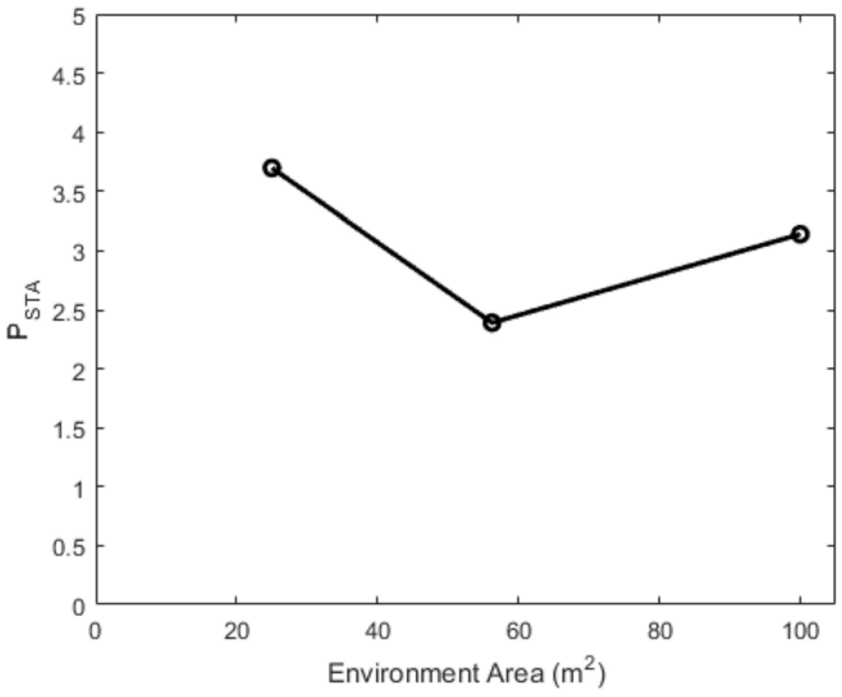

Figure 7: Flexibility test for average performance over environment area

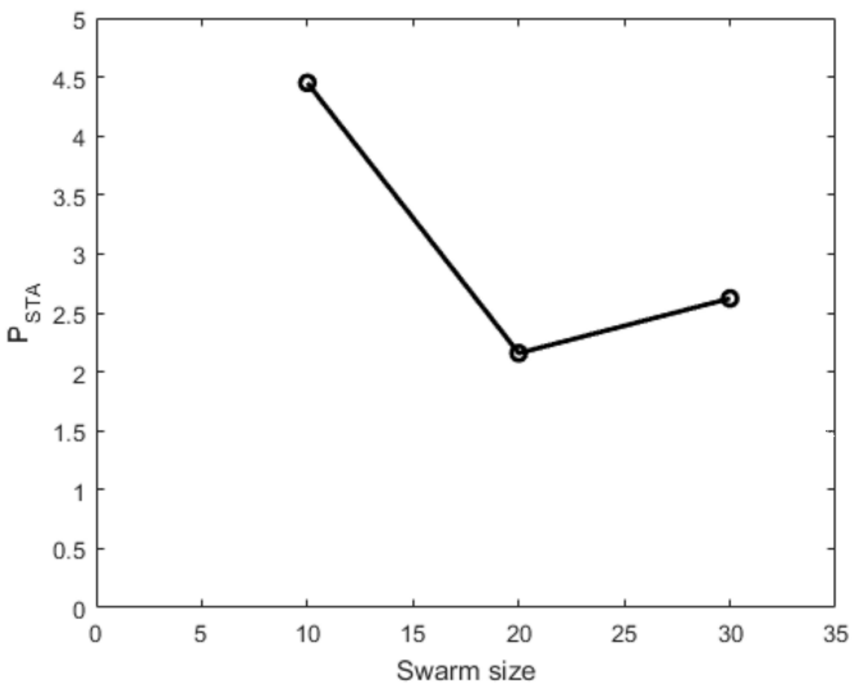

Figure 8: Scalability test for average performance over swarm size

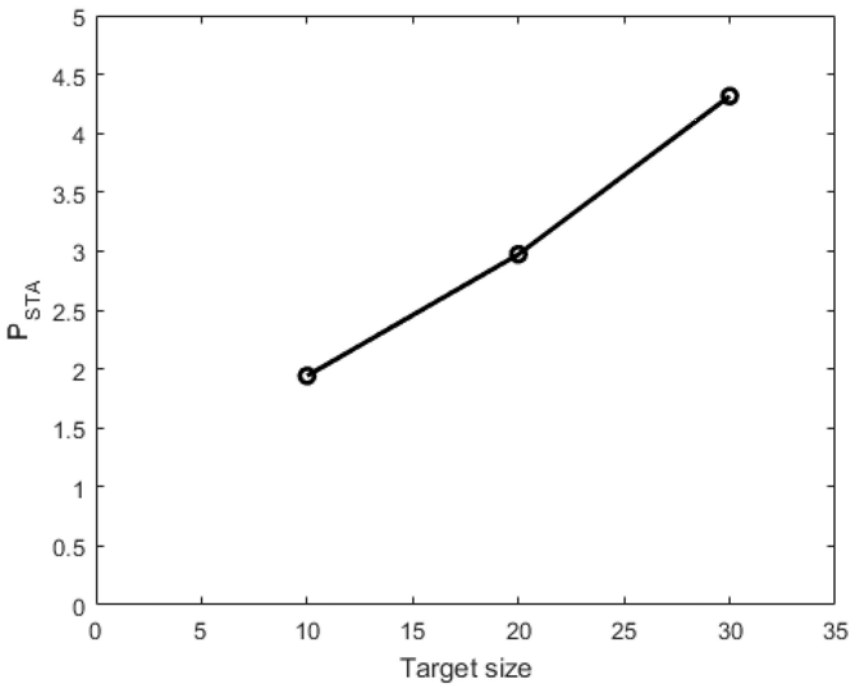

Figure 9: Robustness test for average performance over environment area

5.4 Algorithmic complexity and feature comparison

Further validation of the proposed algorithm can be shown through complexity analysis for some of the leading multi-search algorithms. RDPSO and GSO are compared with the proposed algorithm. Table 1 shows the summary for the comparison. RDPSO has robot dynamics using fractional calculus and sub-optima avoidance while the other two do not have any implement of such kind. The initial deployment of the GSO and proposed algorithm are Random while RDPSO and proposed algorithm based on Lennard-Jones potential utilise artificial repulsion for obstacle avoidance. The communication methods are broadcast for both GSO and proposed while RDPSO uses Ad hoc multi-hop for communication. However, communication complexity for all three is the same and depends on N_S which is the neighbouring swarm size. Computation complexity is higher for RDPSO as well as memory complexity since it uses a fractional order series R_A. Memory complexity for GSO and proposed only depend on fixed number of values taken from previous iteration hence the lower complexity.

Table 1: Summary of multi-search algorithms in comparison with the proposed algorithm

Aspect	RDPSO [11][15]	GSO [11][16]	Proposed algorithm
Robot dynamics	Fractional calculus	–	–
Initial deployment	EST	Random	Random
Obstacle avoidance	Artificial repulsion	Low-level control	Artificial repulsion
Communication	Ad hoc multi-hop	Broadcast	Broadcast
Sub-optima avoidance	Reward punish-ment	–	–
Multiple dynamic sources	Dynamic partitioning and fuzzy adaptive behaviour	Partitioning	–
Computational complexity	$O(2N_S)$	$O(N_S)$	$O(N_S)$
Memory complexity	$O(R_A)$	$O(1)$	$O(1)$
Communication complexity	$O(N_S)$	$O(N_S)$	$O(N_S)$

Chapter 6

Concluding Remarks

The proposed algorithm has shown promise as a proper swarm implementation for Lennard-Jones potential and PFSM for the task of multi-robot search for multiple targets. It is adequately scalable, flexible and robust and shows similar algorithmic complexity to other algorithms of this nature. Due to the nature of PFSM, modifications can be made with ease to allow for higher functionality. It does have issues with sub-optima and, has not been tested extensively. Furthermore, it has some trouble with very dynamic environments. There is scope for further work in a real environment with real robots for dynamic multiple targets. Practical testing would reveal more information and remains as future work along with an optimisation technique for the parameters.

References

[1] Y. Tan and Z. Zheng, “Research Advance in Swarm Robotics,” *Def. Technol.*, vol. 9, no. 1, pp. 18–39, 2013.

[2] E. Sahin, “Swarm robotics: From sources of inspiration to domains of application,” *Swarm Robot.*, pp. 10–20, 2004.

[3] A. J. C. Sharkey and N. Sharkey, *The application of swarm intelligence to collective robots*. 2006.

[4] R. P. P. Rocha, “Building volumetric maps with cooperative mobile robots and useful information sharing: a distributed control approach based on enthropy,” 2006.

[5] M. S. Couceiro, D. Portugal, and R. P. Rocha, “A Collective Robotic Architecture in Search and Rescue Scenarios,” *Proc. 28th Annu. ACM Symp. Appl. Comput. - SAC ’13*, pp. 64–69, 2013.

[6] L. E. Parker, “Distributed Intelligence: Overview of the Field and its Application in Multi-Robot Systems,” *J. Phys. Agents*, vol. 2, no. 1, pp. 5–14, 2008.

[7] J. Borenstein, H. R. Everett, L. Feng, and D. Wehe, “Mobile robot positioning: Sensors and techniques,” *J. Robot. Syst.*, vol. 14, no. 4, pp. 231–249, 1997.

[8] I. Kelly and A. Martinoli, “A scalable, on board localisation and communication system for indoor multi robot experiments,” *Sens. Rev.*, vol. 24, no. 2, pp. 167–180, 2004.

[9] L. Bayindir, “A review of swarm robotics tasks,” *Neurocomputing*, vol. 172, pp. 292–321, 2016.

[10] G. Beni, “From Swarm Intelligence to Swarm Robotics,” *Swarm Robot.*, vol. 3342, pp. 1–9, 2005.

[11] J. Kennedy and R. Eberhart, “Particle swarm optimization,” *Neural Networks, 1995. Proceedings., IEEE Int. Conf.*, vol. 4, pp. 1942–1948 vol.4, 1995.

[12] M. S. Couceiro, P. A. Vargas, R. P. Rocha, and N. M. F. Ferreira, “Benchmark of

swarm robotics distributed techniques in a search task," *Rob. Auton. Syst.*, vol. 62, no. 2, pp. 200–213, 2014.

[13] K. Derr and M. Manic, "Multi-robot, multi-target particle swarm optimization search in noisy wireless environments," in *Proceedings - 2009 2nd Conference on Human System Interactions, HSI '09*, 2009, pp. 81–86.

[14] Q. Zhu, A. Liang, and H. Guan, "A PSO-inspired multi-robot search algorithm independent of global information," in *IEEE SSCI 2011 - Symposium Series on Computational Intelligence - SIS 2011: 2011 IEEE Symposium on Swarm Intelligence*, 2011, pp. 74–80.

[15] J. Pugh and A. Martinoli, "Distributed Adaptation in Multi-Robot Search using Particle Swarm Optimization," in *Proceedings of the 10th International Conference on the Simulation of Adaptive Behavior*, 2008, pp. 393–402.

[16] M. S. Couceiro, R. P. Rocha, and N. M. F. Ferreira, "A novel multi-robot exploration approach based on particle swarm optimization algorithms," *Safety, Secur. Rescue Robot. (SSRR), 2011 IEEE Int. Symp.*, pp. 327–332, 2011.

[17] K. N. Krishnanand and D. Ghose, "Glowworm swarm optimization for simultaneous capture of multiple local optima of multimodal functions," *Swarm Intell.*, vol. 3, no. 2, pp. 87–124, 2009.

[18] A. Campo *et al.*, "Artificial pheromone for path selection by a foraging swarm of robots," *Biol. Cybern.*, vol. 103, no. 5, pp. 339–352, 2010.

[19] J. L. Baxter, E. K. Burke, J. M. Garibaldi, and M. Norman, "Multi-robot search and rescue: A potential field based approach," in *Autonomous Robots and Agents*, vol. 76, 2007, pp. 9–16.

[20] W. Liu, A. F. T. Winfield, and J. Sa, "Modelling Swarm Robotic Systems: A Case Study in Collective Foraging," *Proceeding Towar. Auton. Robot. Syst.*, pp. 25–31, 2002.

[21] M. Brambilla, E. Ferrante, M. Birattari, and M. Dorigo, "Swarm robotics: A review from the swarm engineering perspective," *Swarm Intell.*, vol. 7, no. 1, pp. 1–41, 2013.

[22] C. Pinciroli *et al.*, "ARGoS: A modular, parallel, multi-engine simulator for multi-robot systems," *Swarm Intell.*, vol. 6, no. 4, pp. 271–295, 2012.

[23] M. Bonani *et al.*, "The marXbot, a miniature mobile robot opening new perspectives for the collective-robotic research," in *IEEE/RSJ 2010 International Conference on Intelligent Robots and Systems, IROS 2010 - Conference Proceedings*, 2010, pp. 4187–4193.