Modelling Personality in Swarm Robotics Based on SPIDER Algorithm

Master Thesis

Author:

Jing Yang

MSc Candidate in Computer Science

Supervisor:

Dr. Seth Bullock

Toshiba Chair in Data Science and Simulation

A dissertation submitted to the University of Bristol in accordance with the requirements of the degree of Master of Computer Science in the Faculty of Engineering.

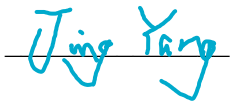
Presented on: 17.01.2022

Abstract

The SPIDER Algorithm is a personality-based algorithm inspired by the social spider *Stegodyphus dumicola*, used to control swarm robots to form communication chains connected to bases in unknown environments. This project has reproduced the SPIDER Algorithm using Python to comprehensively analyse and improve it. The effectiveness, robustness, scalability and flexibility of the SPIDER-hunt Algorithm are fully supported by the experimental results. Are well supported by the experimental results, introducing personality traits while retaining the advantages of distributed control and low computational complexity. Finally, this project explores the suitability of adaptive personality and pragmatic swarm robotics algorithm design in a practical context.

Author's declaration

I declare that the work contained in this dissertation complies with the University's Regulations and Code of Practice for Taught Programs and has not been submitted for any other academic award. Unless otherwise specified in the text, this work is all my own. I have appropriately referenced and acknowledged all material in this dissertation that is not my own. Except for referenced literature, all views stated in the dissertation are those of the author. Except for referenced literature, all views stated in the dissertation are those of the author.

SIGNED:  DATE: 9.10.2022

Acknowledgements

I would like to sincerely thank my project director, Dr. Seth Bullock, for guiding me throughout the project.

I would like to thank the student administrator of SCEEM for their patient care for my special situation. Without your understanding, I would not have been able to fulfill these tasks successfully.

Finally, I would like to express my deep gratitude to my girlfriend and family for their support during my studies.

Table of Contents

List of Figures

List of Tables

1 Introduction

This section includes the motivation and main objectives of the project development, as well as the main structure of the report.

1.1 Motivation

The SPIDER (Swarm Personalty for DEnsity Response) Algorithm is an 'adaptive personality' multi-robot system control algorithm developed by Edmund R. Hunt's team, inspired by the results of a study on the social spider *Stegodyphus dumicola*'s social behaviour patterns.[1]

The study on social spiders that inspired the development of this algorithm is novel and highly intriguing. According to Hunt et al.'s study (2018), there exist both "bold" and "shy" individuals in social spider populations, and the number of bold individuals present has a significant positive correlation with the hunting aggressiveness of the population as a whole. In addition, the personalities of swarming spiders are adaptive, with the boldest individuals changing over time and the boldness of the shy individuals increasing autonomously after the bold individuals are artificially removed.[2]

The SPIDER Algorithm uses decentralised control and gives each robot agent in the cluster a low computational behavioural logic: each robot's current destination from the base dynamically varies according to its own "boldness". Simultaneously, each robot automatically adjusts its boldness according to the number of robots around it and their boldness. Based on this property, the algorithm achieves the ability to shape personalities in response to group density and boldness to maintain a specific population distribution.

Currently, the original developers use the SPIDER algorithm to form long semi-stable chains extending from the base. However, the algorithm has a much broader scope of potential pragmatic applications. The low computational cost and decentralisation of the SPIDER algorithm are highly promising as it opens up research ideas for this project. In addition to the formation of characteristic morphologies such as long chains, the design idea of adaptive personality and the boldness algorithm of response density is also suitable for collective decision-making tasks in swarm robots.

Motivated by this, this project expects to improve how the SPIDER algorithm calculates boldness through further experiments and research. It will also apply this adaptive personality-based algorithm to a classical hunting task, exploring its performance in multiple dimensions such as effectiveness, stability, flexibility and scalability. Based on this, a deeper look is taken at whether single personality differentiation is a practical idea suitable for swarm robot algorithm design.

1.2 Aims and Objectives

Based on the motivations mentioned above, several key objectives were established

to evaluate the project's success:

1. Conduct a comprehensive review of researches on algorithms related to personality traits in the field of multi-robot systems, summarising the characteristics, application directions, strengths and weaknesses of these algorithms.

2. The SPIDER-density Algorithm and SPIDER-boldness Algorithm were re-implemented in a different environment using Python, and the performance of the two SPIDER Algorithms was re-evaluated.

3. Improve the SPIDER Algorithm and apply it to complete a swarm robot's hunting task. Compare the results with the control groups that are not designed with personality in mind to assess algorithm performance.

4. Based on the above results, explore whether a single personality differentiation is a practical idea suitable for the design of swarm robotics algorithms.

1.3 Structure of Dissertation

The report has been divided into five main sections, with each section organised and presented as follows.

1. Background chapter. Introduce12 the research background of this project: a brief introduction to the SPIDER Algorithm, an overview of the field of application of the algorithm in group robotics, and the current state of research on personality-related multi-robot control algorithms.

2. Overview of SPIDER algorithm. Describe the first phase of this project, including the implementation of the SPIDER Algorithm, parameter optimisation methods, analysis methods, results and discussion.

3. Improvement of Algorithm. Describe the second phase of this project, including the implementation of the SPIDER-hunt Algorithm, analysis methods, results and discussion.

4. Evaluation chapter. Evaluate the results obtained in the first two chapters; problems encountered in development and resulting improvements; explore the suitability of adaptive personality and pragmatic swarm robotics algorithm design.

5. Conclusion chapter. Include the main contributions of the project as well as future work.

2 Background

This section provides a detailed and systematic statement of the research background and context of the project. The first section briefly describes the groundwork of the project, the SPIDER Algorithm. The second section provides an overview of the area of application of the algorithm, swarm robotics. In the third part, the current state of research on personality-related multi-robot control algorithms is discussed comprehensively. In the end, there is a quick summary of the chapter content.

2.1 SPIDER Algorithm

The SPIDER Algorithm, developed by Hunt's team, is the cornerstone of this project. Details of the algorithm implementation and evaluation are discussed in Chapter 3, and only a brief background and nature of the work will be given here.

2.1.1 *Stegodyphus dumicola*

Stegodyphus dumicola, colloquially referred to as the African social spider, is a member of the Eresidae or velvet spider family. It is indigenous to the Central and Southern African continents. This species is one of three Stegodyphus spiders that exhibit social behaviours.[3]

Recent studies have shown that *Stegodyphus dumicola* populations are divided into bold and shy individuals, and that partial task differentiation in social life follows personality differences.[4] The number of bold individuals present had a significant positive correlation with hunting aggressiveness for the entire population. Bold individuals moved more frequently away from the population centre in search of food sources and in response to crises. In constrast, shy individuals preferred to stay in the population centre away from crises for less aggressive care work. Most importantly, although the overall distribution of boldness as a trait is relatively stable across the population, it is not completely constant: the boldness “turnover” over time; the bold individuals decrease in boldness after being artificially removed from the population and left on their own, and the shy individuals increase in boldness when surrounded by shy companions. This means that the boldness of spiders is adaptive.[2] The SPIDER Algorithm has been developed to simulate this property.

It is worth noting that one of the better known bio-inspired algorithms related to social spider habits is the Social Spider Optimization (SSO) algorithm, which, although in the same field of population intelligence, is an optimization tool rather than a swarm robot controller and is not relevant to this project.

2.1.2 Overview of the Algorithm

The SPIDER (Swarm Personalty for DEnsity Response) algorithm is a swarm robot control algorithm inspired by the phenotypic plasticity of personality exhibited by

Stegodyphus social spiders. decentralised risk appetite assignment proportional to the swarm's capacity to tolerate prospective losses.

Such algorithms can be used to solve tasks that are impossible or too dangerous for individual robots, in particular to explore and establish distributed communication chains in unknown and dangerous areas. For example, in areas of nuclear radiation leakage, establishing communication chains that can extend to the most dangerous areas in order to collect information, etc.

The algorithm uses decentralised control and gives each robot agent in the cluster a low computational behavioural logic: each robot's current destination from the base dynamically changes according to its own boldness. At the same time, each robot automatically adjusts its boldness according to the number of robots around it and its boldness. Based on such properties, the algorithm, as its full name suggests, enables a novel approach to the exploration and connectivity trade-offs by shaping personality through response density to maintain a population-specific distribution.

The SPIDER algorithm contains two controllers, SPIDER-density and SPIDER-boldness, both of which are associated with boldness as a unique personality. The SPIDER-density controller calculates the boldness value only in relation to the population density within the perceived range of the robot agent. The SPIDER-boldness controller, on the other hand, adds to this the feature of correlating the numerical growth rule of the robot's boldness with the average boldness of its neighbours, making the algorithm more flexible and adaptable to numerous situations.

Hunt et al. (2018) experimented with a simulation of the algorithm on the kilobox platform and optimised the parameters using an elitist genetic algorithm. In addition, they tested the algorithm's robustness through simulated population catastrophes, which looked at the performance of a swarm of robotic bees attempting to regain homeostasis when subjected to large numbers of individual damage. After testing, swarm robots using both controllers regained homeostasis after fluctuations. Swarm robots have three main properties: robustness, flexibility and scalability, and experiments such as this are quite novel and interesting in terms of swarm robot properties.

The SPIDER Algorithm is still only a simple prototype, both in terms of reducing social spiders' behavioural patterns and pragmatism. However, this gives it the property of being computationally simpler than other personality-inspired swarm robot control algorithms. Therefore, rather than further complicating the algorithm, this project prefers to apply this parsimonious logic to other tasks to test its potential. Also, due to the algorithm's excellent performance in robustness tests, the algorithm demonstrates superiority for tasks with group member damage. Based on these advantages, group trapping experiments using the improved algorithm were designed for this project and are described in detail in Chapter 4 of this report.

2.2 Swarm Robotic

Swarm robotics has several unique properties compared to ordinary robotics. This

project is dedicated to the development of a swarm robotics control algorithm. In order to more fully describe the superiority of the algorithm, this section will briefly introduce the definition and characteristics of swarm robotics, application scenarios, and how these elements have inspired the development of the project.

2.2.1 Definition and Brief History

Swarm robotics, also known as "embodied swarm intelligence" [6] is the study of the design, development, and deployment of large groups of robots that collaborate and solve problems or execute tasks cooperatively [13].

The concept of swarm robotics emerged in the late 9801s. At that time, swarm robotics, as an application of the idea of group intelligence to the field of robotics, was just being distinguished from group optimization (or later heuristic algorithms) by the broad concept of group intelligence.

Group bots are thought to have such characteristics as decentralized control, lack of synchronicity, simple and (quasi) identical members to mark their distinction from ordinary groups [8]. These characteristics are significantly correlated with research findings on social insect behaviour patterns during this period. A major feature of early group robotics research was inspired by biological [9], with origins in cellular robotics [10] and later mainly swarming insects capable of exhibiting social behaviour, such as ants [23] , bees [18] and, for this project, social spiders.

2004 In 2007, Erol Şahin et al. defined swarm robotics as the study of how to construct a large number of relatively simple physically embodied agents in such a method that a desirable collective behaviour emerges from the agents' local interactions with one another and with their environment [11]. Their concise and precise definition of concepts and properties has been widely used by subsequent researchers.

During the same period, the situation began to change with the development of biology and robotic hardware, and Amanda J. C. Sharkey argued that, for practical reasons, there was no need to restrict robot performance to the "simple" level of individual agents in a swarm in an archaic manner in order to replicate biological behaviour. This led Sharkey to classify swarm robots as Scalable SR, Practical Minimalist SR, and Nature-inspired minimalist SR, taking into account the performance and algorithmic inspirations of realistic robots in terms of communication [10] The group is divided into Scalable SR, Practical Minimalist SR and Nature-inspired minimalist SR.

Since then, in the last decade of research, the definitions given by scholars have become more comprehensive or broad, and the definition of swarm robots has gradually become as all-encompassing as the "group intelligence" from which it derives. "Biologically inspired" is still mentioned, but it is not always the focus anymore. The development of swarm robots now focuses on the engineering direction, i.e. on solving practical problems. It follows that although the SPIDER Algorithm is inspired by the behaviour of swarming organisms, it need not be limited by the rules of activity of the

organisms themselves and can be boldly adapted to the practical needs of swarming robots themselves.

2.2.2 Properties

In the ideal vision of scientists, swarm robots should have the powerful properties of their reference source, the clusters formed by social insects: robustness, flexibility, scalability.[11]

- Robustness implies that a system is able to maintain operation in the face of environmental disturbances or system failures. System failures may include the loss of some of the robot agent's communication capabilities or mobility. Environmental disturbances may be caused by weather changes, terrain changes, etc. The ability of a swarm robotics system to withstand these conditions relies on three properties: first, a high degree of system redundancy. A robust system is a reliable system, and a reliable system can be created based on unreliable parts through redundancy.[15] . Any loss or failure of an individual can be compensated for by another individual. Individuals do not necessarily have to be completely homogeneous, but they must be substitutable. Secondly, the simplicity of the individual. In fact, individual simplicity is the basis for overall redundancy, although it means that individual robotic agents cannot perform any important tasks, it also means that the loss of individuals is acceptable. Thirdly, decentralised control. There is no central control behind the coordinated movement of social insects [14]. Each insect is an individual decision-maker, and the destruction of one part of it will not have a fatal impact on the overall operation.
- Flexibility implies the ability to cope with different environments and tasks. It requires swarm robotic systems to be able to generate modular solutions flexibly for different tasks in different environments. The fact that the functions of individuals in a swarm do not move in a fine-grained and specific direction means that swarms have the potential to diversify complex tasks through cooperation. ¸ Şahin explains flexibility graphically by using the example of ants, in which the nature of the actions of individuals involved in foraging, forming chains and carrying prey in a swarm are very different. However, the ants are still able to adopt different practices to accomplish each cooperation, which is a sign of their colony's power [11].
- Scalability is the ability of a system to function well when the size of the swarm changes. Scalability relies on a limited range of local communication and decentralised decision-making by individuals in the system. Enhancement of individual functionality affects overall scalability.

In summary, the three main advantageous properties of swarm robots are essentially based on redundancy, decentralised control and individual simplicity, which together constitute the unique advantages of swarm robotics. The performance evaluation of

related research at the software level is also based on the testing and discussion of the algorithms' robustness, flexibility, and scalability.

2.2.3 Scenarios of Swarm Robotics

The classical scenarios using swarm robots, i.e. the common tasks to be handled, can be divided into four main categories according to group behaviour: spatially organising behaviours, navigation behaviours, collective decision-making and other collective behaviours.[17]

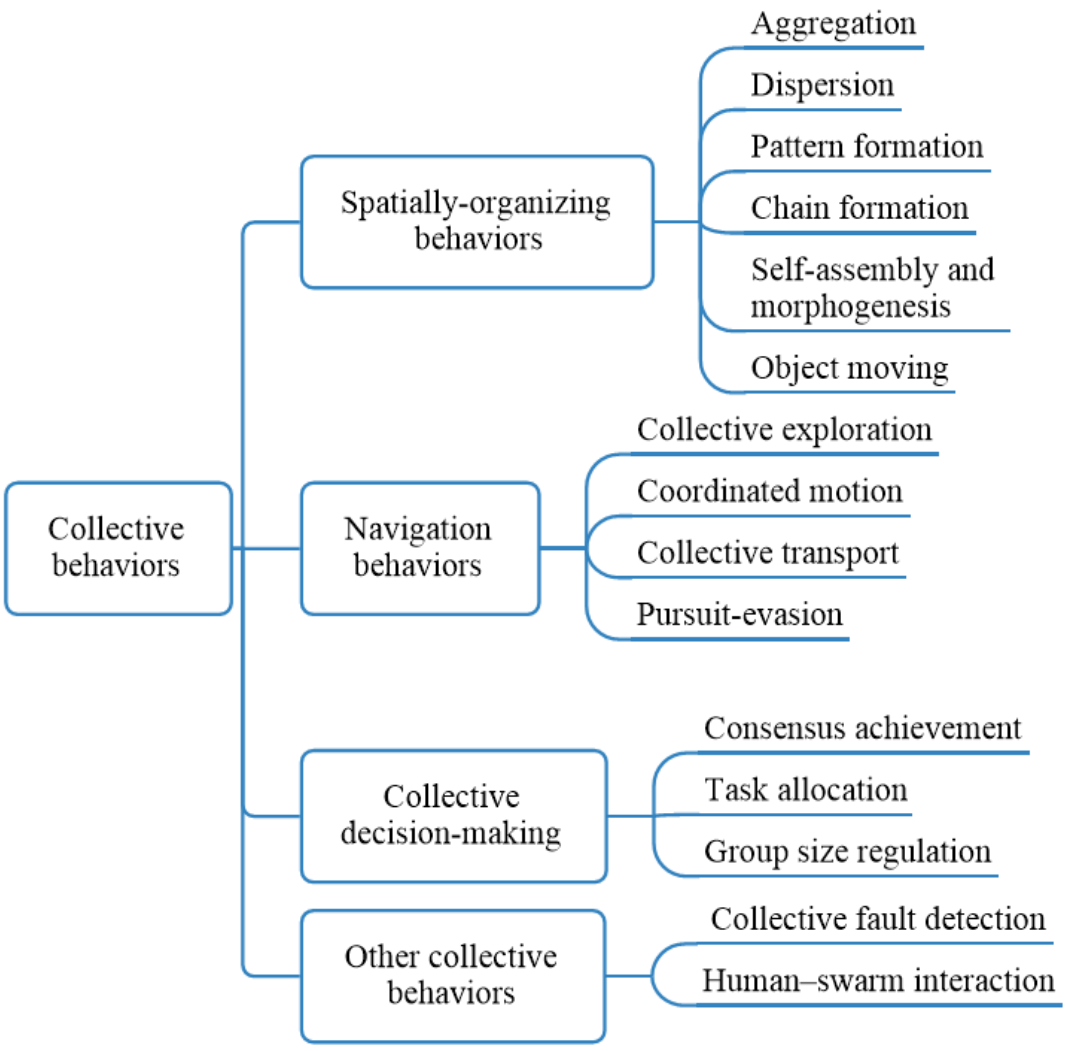


Figure 2-1 : The collective behaviour of swarm robots is classified as spatially-organising behaviours, navigation behaviours, collective decision making, and others.

Research related to spatially-organising behaviours aims to address the problem of allowing the robot itself and its task objects to organise their positions in space in a

specific way. In order of complexity, these are: aggregation[18] , dispersion[19] , pattern formation[22] , chain formation[23] , self-assembly and morphogenesis[29] , object moving[32] .

In coordinating the movement of a cluster of robots, the act of navigation is born. It is the process by which robots cooperatively explore their environment and coordinate their paths and ways of movement. It includes: collective exploration[35] , co-ordinated motion[36] , collective transport, pursuit-evasion[42] .

Collective decision making is a necessary capability for swarm robots to form autonomous systems at the macro level [19]. Making decisions in an undefined situation is a difficult task, and group decisions are more complex than those of a single individual. How to choose the best solution, how to reduce the losses caused by inter-individual conflicts in the process of reaching consensus, and how to spontaneously distribute tasks are all challenges that need to be explored by scientists. collective decision making includes consensus achievement[37] , task allocation[39] , group size regulation[45] .

Where, in practice, there will be no clear boundaries between tasks and combinations of behaviours will occur from time to time. The work underlying this project, the SPIDER Algorithm, can be seen as a combination of collective exploration in navigation behaviours and chain formation in spatially-organising behaviours, the key to which is the collection and transfer of information in groups, and in the implementation of which consists of Area coverage and swarm-guided navigation. Area coverage is used to create a robotic communication network in the environment. The communication network can detect possible hazards within the coverage area or communicate terrain information to the swarm, which is essential for the robot swarm to perform tasks in unknown areas and to obtain information in order to develop countermeasures for specific situations, forming the basis for the swarm's flexibility. Area coverage is often combined with dispersion behaviour to maximise the distance between robots without interrupting communication, with a view to building a communication network with the widest possible coverage [34].

In contrast, the SPIDER-hunt Algorithm, obtained by improving on the SPIDER Algorithm, can be seen as a special kind of pursuit-evasion in which the prey does not move, also known as foraging [19], self-organised by the local and random decisions of individual team members, swarms of robots search and collect food in a shared environment [49]. The swarms search and collect food in a shared environment.

2.3 Personality Traits in Multi-Robotics

Personality-based modelling has been widely explored in a variety of domains, but it has only just been brought to multi-robot scenarios, This section reviews related work using personality as inspiration for design methods.

Since there are not many studies on swarm robots and personality, this part also includes multi-robot systems that are similar to swarm robot systems. Furthermore,

emotion and character are regarded as synonyms for personality.

2.3.1 Definition of Personality

Personality is a coherent model of emotions, behaviours, cognitions and desires over time and space [50]. Typically, people see emotions as an integration of feelings, goals and actions and evaluations at a point in time, and personality as an integration of emotions in time and space.

However, this report's survey of related work will treat emotions and personality more loosely as synonymous. This is because research in the multi-robot field often does not distinguish between these two traits: some studies set up functions describing personality change and then describe the resulting performance of robots as emotions; some studies set up functions describing the emotional change and then describe the emotional orientation of robots over time as personality.

This phenomenon is due to two main reasons: firstly, research combining personality or emotion with multi-robot design is still in its infancy and therefore, there is no clear classification in design thinking. The use of terminology is often confused when inspired by biology or psychology. Secondly, multi-robot systems, especially swarm robotic systems, have an individual simplicity compared to individual functionally complex and complete robots. As a result, researchers often do not work with the goal of creating personalities that are comparable to the functioning of the human brain. The 'personalities' of the individual robots in the group are simpler and do not relate to complex past experiences. On this basis, the simplified personalities are modelled and experimentally represented in a way that is not different from emotions.

2.3.2 Role of Personality

Simplified personality plays a triple role in multi-robot systems: responding to the current state of the individual, regulating the individual's internal behaviour, and acting as a mediator of social interactions. Past researchers have designed personality models for multi-robots that include at least one of these three aspects.

Personality is used to represent the current state of an individual. The states that are often represented by the robot agent through personality are: energy state, operational state, safety state, and interaction state [64]. The energy state can be directly expressed as a percentage of the robot's battery. The energy state can be directly represented by the percentage of the robot's battery. Operational status indicates the level of performance and the current workload of the robot. The safety state refers to the current level of damage and failure of the robot. Interaction status is determined by how often the robot is currently communicating with other robots and the type of response it makes to requests from other agents. These four states are dynamically changing at all times and together determine the suitability of a given robot to continue performing its current task, and researchers have tried to reflect their influence in terms of emotions when designing control structures. A classic example of this [59] is the implicitly

coordinated task-sharing approach based on emotional intelligence proposed in Setting up many unnecessary emotions in a multi-robot system would add significantly to the computational complexity, which would be unwise. With this in mind, the model [59] incorporates four basic emotions as shown in Figure 2-2: joy represents the state of a robot when it has sufficient energy and the workload is bearable; fear is an attribute that increases when a robot is low on energy or faced with too many tasks to handle; anger increases when a robot encounters an obstacle in performing a task or when it asks for help from another robot but does not respond. In practice, when a certain state of the robot grows above a threshold, fear, for example, the robot gives up facing the unachievable task alone and starts to develop a new plan of action, choosing a task to share and sending a help message to a robot in the right state.

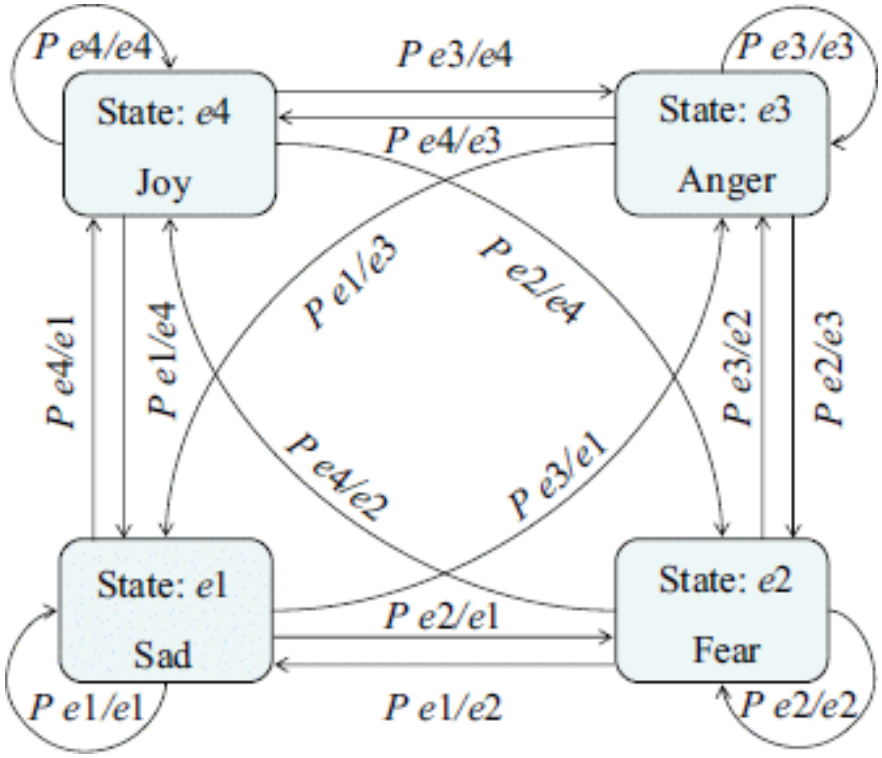


Figure 2-2: Markovian emotion model for four emotional states showing transition probabilities [59]

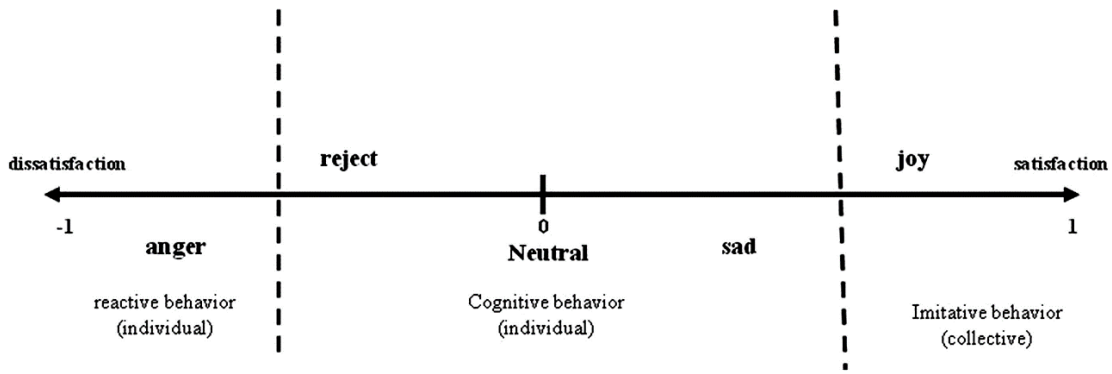


Figure 2-3: Emotional model in AMEB . [64] AMEB abstracts the current state of the robot as a satisfaction index, with higher satisfaction indexes corresponding to higher levels of positive emotions.

Positive emotions such as joy tend to make the robot adopt a cooperative strategy, while negative emotions such as anger tend to make the robot act alone.

A similar design approach is used in AMEB, a fuzzy emotion model for heterogeneous robots [64]. AMEB is a control architecture for heterogeneous robots designed to allow agents in a group to self-organise and form emergent behaviours, with three levels: individual, collective, and systematic learning. In order to provide support to the robotic agents at the individual level and to manage each agent's perceptual capabilities and local decision-making, the researchers have built an emotion model into the software that controls and manipulates the robot's behaviour. As shown in Figure 2-3, AMEB treats the robot's energy state, operational state, safety state and interaction state as fuzzy variables with three values of 'low, medium and high', thus defining a set of fuzzy rules to determine the robot's emotional state. It abstracts the state value as an index of the robot's satisfaction and uses the rules of the fuzzy emotion model to calculate the emotional intensity of the robot, which leads to specific behaviours.

When personality is used to represent the current state of a robot, the attributes that are often present are joy, sadness, anger and other emotions that can be evoked in time and change rapidly. This is in order to be able to accurately reflect the real-time level of the robot, similar to the state of a human at work. Further, there are personality attributes that regulate the magnitude of the impact that these basic emotions or current states can have. At this level, personality regulates the internal behaviour of the robot.

Personality acts as an internal regulator of behaviour. When personality represents internal states, the personality of each individual changes according to the actual information through the same set of rules, i.e. the same state causes the robot to present the same personality. Where personality influences local decisions, on the other hand, robots with different personalities are required to act differently when faced with the same information. The introduction of personality in multi-robot systems helps them to adapt to a dynamic environment in which they can improve their decision-making ability at a given moment. Personality defines the robot's disposition at a given moment, which determines how it responds to a stimulus. In this way, robots can react differently to the same stimulus, transforming collective behaviour into unpredictability and promoting emergence and self-organisation in the system[64] . For example, animals respond more strongly to external stimuli when they experience fear [63]. For example, shy spiders are more likely to flee when confronted with signals from natural predators [69] .

[62] In the multi-agent navigation scenario, three different types of personality models containing "**shy, aggressive, and tense**" were used, as shown in Figure 2-4 As shown, agents with shy personalities will choose to avoid the edges of the group to reach the goal, exhibiting collision avoidance behaviour. More aggressive e agents, on the other hand, tend to pass directly through the dense group of other agents.

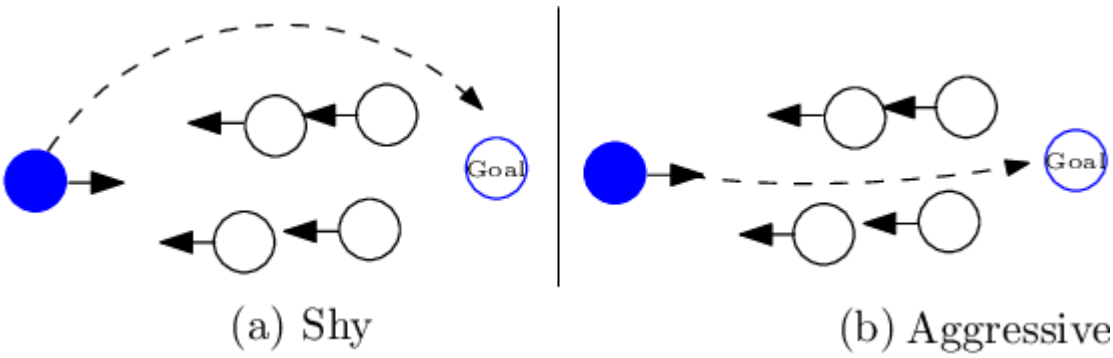


Figure 2-4 (a) Using a shy model, decides to move around the group. (b) Using the aggressive model, instead, makes way between the incoming agents to reach the goal [62]

On the other hand, personality influences are often associated with the idea of game theory when it comes to the local decisions of agents in multi-robot systems[70] . Game theory focuses on the rational behaviour of interdependent intelligence and is a formal representation of the study of competitive and conflicting relationships between intelligence, with the main aim of determining the rational decisions of the intelligence to maximise payoffs or minimise penalties. The personality model combined with game theory has good theoretical extensions, and studies that contrast with the game theoretical approach provide a clearer reality of the strengths and weaknesses of the personality model.[70] Personality traits are defined abstractly as the decision orientations of a robotic agent, such as the agent's propensity to adopt a certain strategy in a zero-sum game. This abstract approach is scalable and can be applied to the implementation of more specific personalities such as the familiar 'bravery', 'curiosity', 'teamwork', etc.[66] An evolutionary multi-robot personality model is proposed to model the behaviour of human users sharing resources in a cloud system, and this model is compared with the Nash Bargaining Solution Sharing model to obtain a personality model with higher performance and user satisfaction.

Common personality traits that act as internal behavioural moderators for robotic agents are: personality factors used to indicate decisional tendencies[70] , frustration[50] , cooperation[57] , curiosity[60] , aggressiveness[62] , generosity[66] . These personalities are more stable but are also adaptive. The use of a fixed personality model throughout the task was only seen in controlled experiments. The researchers hoped that the accumulation of environmental influences would cause changes in personality, spontaneously making the changes in decision making required by the task.

Personality as a mediator of social interaction. In nature, animals are able to experiment with a simple and direct form of communication by exposing emotional states, such as changing body posture, which produces social conditioning effects [63]. In a multi-robot context, social interaction is subdivided into social interaction between robotic agents and human-robot interaction due to the different subjects involved. When personality is used as a mediator of multi-robot social interactions, work focuses on the accuracy of personality presentation and recognition.

As personality traits are not complicated to set up, the way in which agents present their personalities to their colleagues in a multi-robot system is very straightforward: pass on the values of the corresponding personalities with neighbours who come into communication range, or get the emotion sets of all the robots directly in the cloud. This is slightly different from traditional swarm bots. Most swarm bot systems, in the pursuit of distributed control as well as individual simplicity, limit the transfer of information between agents and also avoid allowing the bots to obtain data from the cloud. However, most studies involving personality ignore this. Most swarm robotics algorithms that introduce personality traits are not fully distributed control. Personality is a property that requires more abstract thinking to identify than pheromones, distance, etc. Therefore, no matter how simple the implementation is in the algorithm, the design by default raises the complexity of the robot.

For a classic example of recognition of personality within a robot swarm see[64] . The researchers used an algorithm based on the AR2P system to endow each robot with the ability to recognise emotions. Inspired by the neocortical operational model, the algorithm mimics the workings of the brain to build a set of recognition modules that work at different levels. In [64] the swarm robot architecture, the AR2P algorithm recognises the emotion of another robot in the system based on the set of current state signals from the cloud. This recognition module consists of a low-level recognition model that directly identifies the current state of the robot (battery level, performance level, etc.), and a high-level recognition model that models different emotions based on combinations of different states and their weights.

Personality in the direction of multi-robot-human interaction [[53] ,[65] ,[67]] focuses on how to make a population of robotic bees exhibit emotions that humans can understand. Unlike single complex robots, individual simple multi-robot systems can communicate emotions to humans in human-like language with straightforward expressions, and therefore requires an artistic and suggestive presentation. [65] Attempts have been made to make the robots' 'shy' personalities felt by human volunteers through simple responses to human gaze and gestures. When humans looked at the robots, they would stop moving and look back at them; if they looked away, the robots would continue the skit.[66] The performance potential of the simpler multi-robot system of individuals, the swarm robots, is explored. In this case, personality no longer acts as a deciding factor for the group but becomes the task itself. The authors decided to have the swarm robots arrange themselves in different patterns, moving in specific ways to evoke particular emotions. In terms of the implementation method, this is a simple act of spatial self-organisation by the swarm robots. For example, the robot cluster presents happiness by moving along a circular sine curve, visually arranged as a circle with small ripples; the robots move in a swirling pattern along an expanding radius circle, creating a spiral effect to indicate surprise. Effects such as Figure 2-5 .

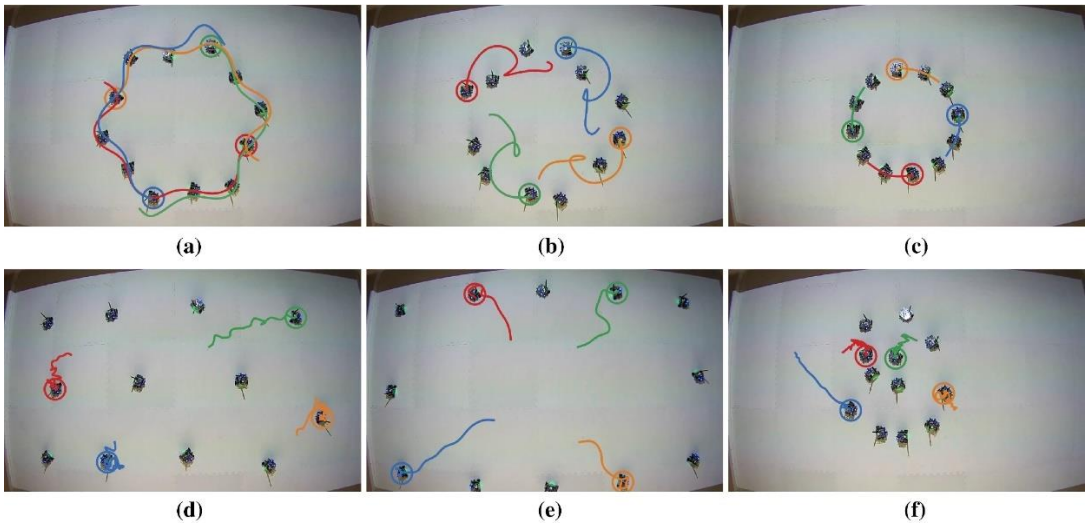


Figure 2-5 : Snapshots of the swarm behaviours implemented on a team of 12 GRITSBot X, taken in the Robotarium with an overhead camera. (a) Happiness, (b) Surprise, (c) Sadness, (d) Fear, (e) Disgust, (f) Anger.[66]

The personalities used for inter-robot social representations are mostly joy, sadness, anger and fear, similar to the personalities described earlier in this section when used to express compressed internal states. These are two closely related traits that are often used to solve problems of intra-group task allocation. Human-robot interaction in multi-robot contexts also starts with the presentation of basic emotions in preparation for more complex stories to be told with multi-robot systems in the future.

2.3.3 Task Areas

Previously we have learned about the main role that personality plays in multi-robot systems and got some rough impressions of the task scenarios that emerge from such research. This section summarises what these tasks have in common. The introduction of personality traits in the design of multi-robot systems has been applied mainly to tasks with three characteristics.

Robots are required to produce a limited number of heterogeneous tasks on a homogeneous hardware base. Both multi-robot systems and swarm robotic systems involve scenarios that require task allocation, where the division of labour and cooperation is required in order to make effective use of a large group. And heterogeneity in personality traits provides the basis for individuals to make different decisions. Personality either absolutely or probabilistically influences the behavioural choices of robots and, with appropriate design, can enable groups to be coordinated into the required number of parts and to perform their respective roles. As a result, personality is heavily used in task allocation and team recruitment tasks [[56] , [58] , [61] , [64]]. Such scenarios are often complex, being combinations of several simple behaviours in which personality is not directly involved in the work, but only as a selection factor.

Tasks that require robotic agents to produce self-interested behaviour. In multi-robot systems, especially swarm robotic systems, the collective good always prevails, created by the high redundancy of clusters and the substitutability of individuals. However, a good system capable of self-organisation also requires a balance between the attrition of individuals and the performance of the whole. This helps both to reduce costs and to make the cluster sustainable. One way to address this issue is to add the attribute of caring for individual interests - personality - to the agents in the group. This gives the agents the ability to assess whether their current state is suitable for continuing the task. Once in a poor state, the agent will either abandon the mission or refuse recruitment to a new mission, wait for help or return to base[54] .

Multi-robot system human-like tasks are required. Personality and emotions are different from ordinary state parameters and are more easily understood by operators and viewers. Therefore, introducing personality traits for multi-robots can both give understandable state feedback to humans and also allow for simple simulation of human groups in certain scenarios. For example, the use of a multi-robot in [17] simulates the resource-sharing behaviour of a group of cloud system users, suggesting suggestions for future construction work. Scenarios in which humans and multi-robots interact also often choose personality as the entry point for communication [[65] , [67]]

2.3.4 Control Methods

The algorithms of multi-robot systems that use personality traits as a design idea are mostly a combination of centralised and distributed control.

The definition of the two control methods in the multi-robot domain is first presented, for which[61] A good formulation is given: in a multi-robot system, the control problem is defined as local when the controller of each robot can be represented as a function combined with neighbourhood information; on the other hand, the control problem is considered as global when the agents need to obtain information about the whole group members, regardless of spatial distances. Further, the control system structure is said to be distributed when the solution is implemented through local interaction only; the centralised nature of the system structure emerges when at least one agent needs to sense global information or communicate with all agents.

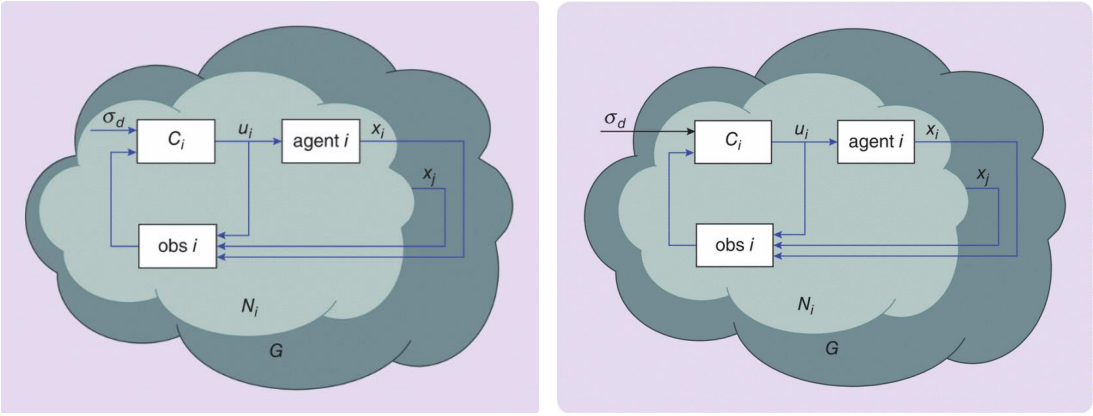


Figure 2-6 The right is Global control problem in a distributed architecture . All lines are thin, blue denoting only local information flows.[61]

Centralised control is suitable for smaller systems, where the cost of computing is proportional to the size of the cluster. The complexity of distributed control is related to the specific algorithm, which is scalable and can be cost effective when scaling up the cluster size. At the same time, distributed control does not have a single point of failure.

In contrast to the fully distributed control pursued by common swarm robotics algorithms, personality-related control algorithms often allow robots to directly access global datasets, directly obtain personality states or other information about all other robots in the group, and also allow robots to broadcast information globally. Most swarm robot systems, in pursuit of distributed control as well as individual substitutability, limit the transfer of information between agents and also avoid allowing robots to obtain data from the cloud. However, most studies involving personalities ignore this. On the one hand, this is due to the fact that researchers have artificially increased the efficiency of communication between robots in order to simplify the simulation process and to focus on examining the role of personalities. For example[57] , robot decision making is still controlled locally, influenced by the density and virtual force of the surrounding robots, but all robots always know exactly where the prey is, regardless of cluster size.[64] All agents in the cluster have direct access to the set of other robot emotions stored in the cloud as well as to the state information of other robots, which is used for two levels of emotion recognition. It also allows bots to broadcast recruitment information globally, in order to be able to move all agents that can participate in the task. On the other hand, this is because algorithms inspired by personality are always designed to boost the complexity of the robot by default. Personality is a property that requires more abstract thinking to recognise than pheromones, distance and other state information, and therefore all by default no longer take into account the problem of restricted communication between agents, allowing a centralised nature of the control method.

Nevertheless, due to the advantages of distributed control in large groups, these studies have almost always adopted distributed control in the solution of some of the tasks. However, as soon as a global information flow exists, the dimensionality of the problem is completely changed. This means that only a subset of the global control is solved using distributed solutions.

2.3.5 Advantages and Issues

After a thorough investigation of the studies related to the introduction of personality traits in multi-robot systems, it is possible to summarise the advantages and issues of these studies compared to other multi-robot systems.

The main advantages of algorithms designed with personality traits in mind are: the ability to intuitively construct heterogeneity in behaviour and decision making in

groups of homogeneous hardware, allowing the system to produce self-organised coordination; the ability to produce desired action jumps after personality changes caused by specific contingencies; the reduction of the contradiction between stability and adaptability in group learning; and making it easier for humans to understand the actions of robotic people.

At the same time, adding personalities to a multi-robot system has three obvious issues: the generation and evolution of personalities lead to increased computational complexity; the expression and interaction of feelings lead to increased hardware requirements; and the reduced degree of distributed control leads to reduced system scalability.

The SPIDER Algorithm, which uses a single personality trait as the key variable, studied in this project, has the advantages of the personality algorithm described above, while retaining the benefits of traditional distributed control of swarm robots, with low computational complexity. It can be seen that the algorithm has a huge potential waiting to be explored.

2.4 Chapter Conclusion

This chapter begins with an overview of the foundations of the project, the inspiration for the SPIDER Algorithm, its main properties and the scenarios in which it can be applied, giving the reader a basic understanding of the work of the algorithm.

The main area covered by this study, the definition, characteristics and classical application scenarios of swarm robotics are then presented. The SPIDER Algorithm and the location of subsequent improvement work can also be found in this section.

Last but not least, this section reviews the current state of research on personality-related multi-robot control algorithms in recent years, covering: the definition of personality in this research direction, the three roles that personality plays in the controller of multi-robot systems, common application scenarios and control methods for personality algorithms, and the advantages and disadvantages compared to algorithms with non-personality elements. A detailed examination ultimately reaffirms the research value of the SPIDER Algorithm.

3 Overview of SPIDER Algorithm

This project is divided into two main phases, the first phase reproduces and evaluates the SPIDER Algorithm. the second phase, improves the SPIDER Algorithm used to complete the hunting task, obtains the new SPIDER-hunt algorithm and evaluates it. In this section, the first phase will be presented.

The first section describes the implementation of the SPIDER Algorithm, including the simulation environment in which the algorithm was run, the performance scoring criteria, and the logic of the SPIDER-density and SPIDER-boldness controllers. The second section briefly describes the elite genetic algorithm for parameter optimisation methods. The third section evaluates the experimental design and results used by the algorithm.

3.1 Implementation

This experiment was simulated using matplotlib in a Python 3.9 environment, running on an i7-9750H, 2.60 GHz, 16.0 GB, Windows 10.

3.1.1 General Framework

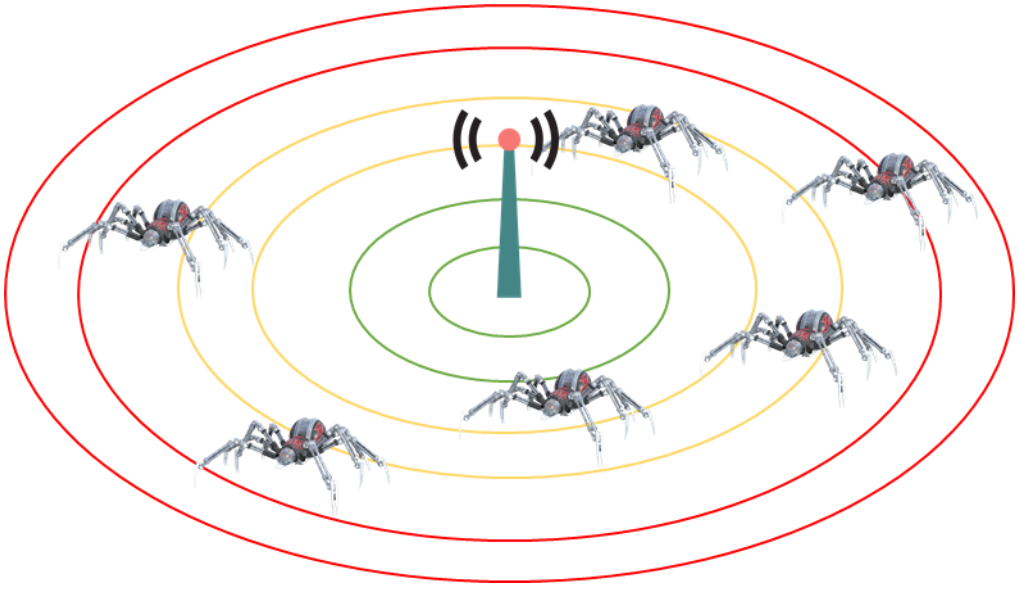


Figure 3-1 : Abstract depiction of the simulation environment for SPIDER Algorithm . Plotting style with reference[57]

The simulation environment set up for the experiments in the reproduction of the SPIDER Algorithm is shown in Figure 3-1 The simulation environment set up for the experiments is shown in the figure. Firstly, this phase of the experiment describes a circular field with different risk levels at different locations, with the centre being the safest area and the further away from the centre of the field the more dangerous it is. A

base robot was placed in the centre of the field, which did not interact with the robot swarm and was only used to receive signals and record scores. At the start of the experiment, the members of the robot swarm move radially from the centre to the edge of the field, starting from around the base robot. They were all able to sense the level of danger at their location and were able to communicate with other robots in their vicinity, transmitting signals of their position.

3.1.2 Experimental Arena

The experimental site is based on matplotlib and is implemented by a polar coordinate system as shown in Figure 3-2 This is shown in the figure. This is a ring of radius 25, which is thus divided into a gradient from 0 to25 different hazards, with rings of different shades of grey to help identify the position of the robot. The green area in the centre is the low-risk area, the yellow area on the periphery is the medium risk area and the red area on the outermost periphery is the high-risk area. The range of the different zones can be adjusted to suit the needs of the experiment. The size of the field and the division into zones can be changed according to the needs of the experiment.

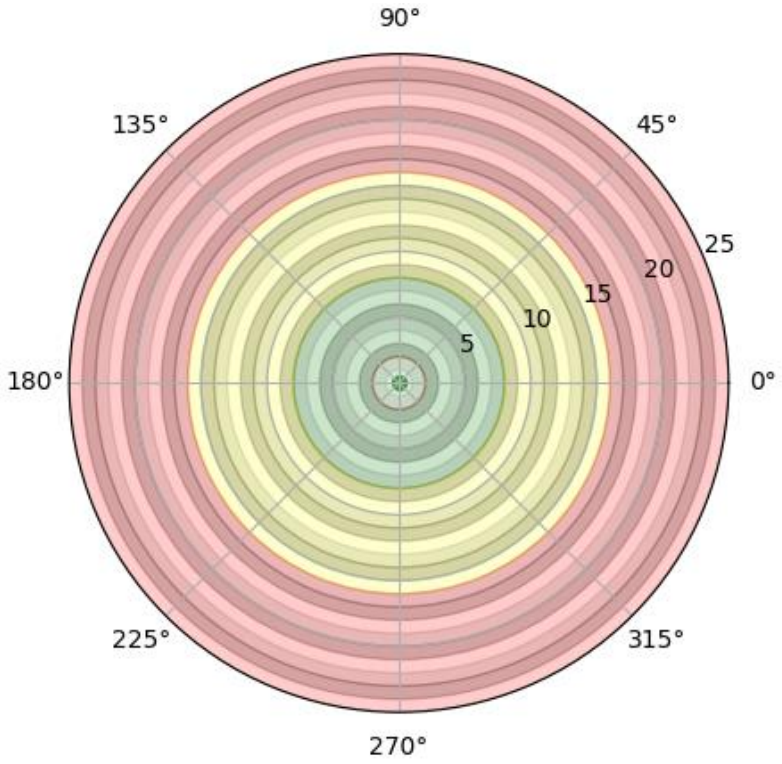


Figure 3-2 : The figure shows a simulated experimental site constructed using matplotlib . This is 25a radius field, with the central green area being the low-risk area, the outer yellow area being the medium risk area and the outermost red area being the high-risk area.

3.1.3 Robot Swarm

This project uses a model that treats the agents in a swarm of robots as mass points.

Each robot is considered to be a particle with a given position at a fixed time. The low-level dynamics of the robots and the control loops required to stabilise them are assumed to be implemented.

There are four basic attributes of a robotic swarm, population size, location, broadcast frequency and local communication distance. The population size of the robot swarm can be set according to the needs of the experiment. Each prime indicates the location where a robot is located. The robots in the swarm are free to stand still or move around the site. The robots within the swarm broadcast their position and other information via a transmitter at a certain frequency. Broadcast messages can be received and recorded by other cluster members within local communication distance.

In the centre of the field, a robot is set up as a base. The base robot has no controller; it does not move and does not affect the movement of the other robots, but has the same local communication distance as the other robots. The base robot is used to record information from the other robots, mainly for performance metric statistics.

3.1.4 Task and Performance Metric

The task of the robot swarm is for the robots to form the longest possible chain of communication outwards from the central location where the base robot is located. A chain is considered to be formed when the distance between two robots is less than the local communication distance.

This experiment expects the robot swarm to balance between exploration and chain formation under the control of the SPIDER Algorithm, rather than being obsessed with link solidity. However, the robots are also not expected to break away from the swarm when attempting to establish communication chains that cover greater distances. If the robots break away from the swarm, they will receive less reward.

To evaluate the performance of the algorithm, the furthest position reached by a chain that can be connected to the centre is recorded as a score, which is added up at regular intervals and the result is the performance metric.

If a chain formed by a robot appears in a more distant area, but is not connected to the base robot in the centre, it does not satisfy the condition of being scored.

Figure 3-3 The two robot chains formed in the current situation are marked in red and green. Both chains are connected to the base robot and are therefore able to transmit signals back to the local area. Both robot chains are4 at their furthest distance from the centre. The current experiment time is 41.5s and the score at this moment is recorded4.

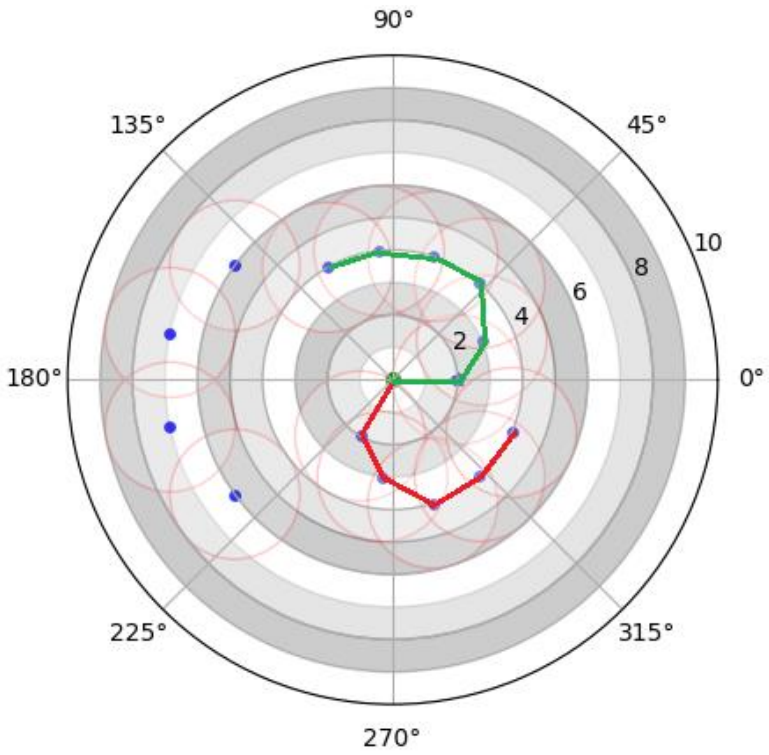


Figure 3-3 : A swarm of robots with population size of 15. In the centre of the field are the base robots that receive signals and record their scores. Each blue prime represents a robot and the red circle indicates the local communication range covered by the robot's probe. The two robot chains formed in the current situation are marked in red and green in the diagram.

3.1.5 SPIDER-Density Algorithm

The SPIDER algorithm consists of two controllers, SPIDER-density and SPIDER-boldness, both of which are associated with boldness as the only personality. SPIDER-density is the initial version of the algorithm, where boldness values are calculated only in relation to the population density within the perceptual range of the robot agent. spider-boldness is an improved version that adds to the initial version the feature of relating the numerical growth rule of the robot's boldness to the average boldness of its neighbours.

The SPIDER-density algorithm is described in the pseudo-code of the algorithm1. The main steps of the algorithm are repeated every 0.5s seconds.

The robot's boldness was limited to between 0to255, which was designed to match the radius of25 the experimental field while the hazard gradient 0grew from to25. Robots with the largest boldness reach the outermost part of the field, i.e. the most dangerous area, while robots with the smallest boldness stay in the centre at a radial distance0.

r_c is the radial distance from the robot's current location to the centre. When initialised, the robot swarm is usually arranged around the base robot at the centre in a ring with a radius of the maximum local communication distance.r_d is the desired radial distance

from the centre, determined by the robot's boldness. The boldness of a robot agent is divided and 10then rounded off to the decimal part to calculate ther_d the value of the For example, if a robot's boldness is215, then itsr_d is21. Note, however, thatr_d is not updated in real time. Only when the robot reaches the previousr_d after it has arrived, it will only be updated according to the current boldnessr_d .

Algorithm 1 SPIDER-density algorithm pseudo-code

```
procedure input parameters a, b for SPIDER-DENSITY (a, b)
Initialize for individual robots separately
    r_c ← robots placed in ring formation
    bold ← set random boldness ∈ [0, 255]
    r_d ← floor (bold/10)
repeat every 0.5s
    for all robots do
        density ← number of neighbors within range
        bold = bold + a × density − b
        bold = limitrange(bold, [0, 255])
        r_c ← travel according to gradient ascent/descent
        if r_c == r_d
            r_d ← floor(bold/10)
        endif
    end for
until experiment time runs out
```

The rule of variation for the value of boldness is that the rate of increase in boldness of a robot is linearly and positively correlated with the number of detectable neighbours by a factor of$a$. And when a robot has no neighbours around it, boldness decreases by a fixed value each time it is updated$b$. According to this mechanism, a robot that is close to a central location because of a smaller boldness can be made to increase its boldness due to the greater population density here, and move next to a more distant area. And as a robot moves away from the central location and the population density around it decreases, the boldness of that robot gradually decreases, updating the lowerr_d , again approaching the central region where the group is located. This simple rule prevents entire populations from piling up close to the base, and also prevents robots from breaking away from the group during exploration.

3.1.6 SPIDER-Boldness Algorithm

The SPIDER-boldness controller adds four new features to the SPIDER-density controller: boldness increase with bold neighbours, 'addictive ' boldness increases, relative boldness increase/decrease, updating desired distance en-route.

The SPIDER-boldness algorithm is described in the pseudo-code of the algorithm2.

The main steps of the algorithm are also repeated every 0.5s seconds.

Now, robots will only increase their boldness if there are bolder neighbours around them.This mechanism is implemented by each robot locally calculating the average of its neighbours' boldness and only increasing the boldness value if that average is greater than its own. This mechanism was inspired by the spider habits observed by the original researchers in [2].

Algorithm 2 SPIDER-boldness algorithm pseudo-code

procedure input parameters a, b, c for SPIDER-BOLDNESS (a, b, c)
Initialize for individual robots separately
  $r_c \leftarrow$ robots placed in ring formation
  $bold \leftarrow$ set random boldness $\in [0, 255]$
  $r_d \leftarrow$ floor $(bold/10)$
repeat every 0.5s
  for all robots **do**
    $mean(neighbor\ boldness) \leftarrow$ average boldness of neighbors within range
    $f(bold) \leftarrow \frac{a}{bold}$; $g(bold) \leftarrow \frac{b}{bold}$; $addict \leftarrow 1$
    if $mean(neighbor\_boldness) > boldness$ **then**
      $bold = bold + f(bold) \times addict - g(bold)$
      $addict = addict \times c$
    else
      $bold = bold - g(bold)$; $addict = 1$
    endif
    $bold =$ limitrange$(bold, [0, 255])$
    if $|r_c -$ floor $(bold/10)| > 2$, $r_d \leftarrow$ floor $(bold/10)$
    $r_c \leftarrow$ travel according to gradient ascent/descent
  end for
until experiment time runs out

In addition, the robot's boldness would have increased faster if it had only increased boldness in the previous cycle. This 'addiction' mechanism was exponentially correlated with the coefficient c is exponentially correlated. This is to allow the process to compensate for the limitation on boldness growth imposed by the previous mechanism.

At the same time, the calculation of the robot's boldness is no longer directly related to neighbour density, but has been changed to a function related to the original boldness.a is still related to the rate of increase in boldness, while$b$ still related to the rate of decrease in boldness.

Finally, the robot was changed to be able to update its destination in a more timely manner. This is to allow the robot to be more agile and responsive to the current

environment, preventing the robot from trying to go to a central area when it is overcrowded or insisting on going to an edge area where no colleagues are.

3.2 Optimization

This study uses an elitist genetic algorithm to optimize the parameters of the SPIDER Algorithm, as shown in the flowchart. Figure 3-4 .

Parameters selected for the SPIDER-density controller a and b the parameters for SPIDER-boldness a and b and c are genes. The fitness score is the average of the performance metric for performing 10 simulation, each simulation taking 2 00s. The selection method is rank-based. due to the specificity of the factors in this algorithm, it is not meaningful to swap the values of the parameters between parents, so the probability of crossover is set to 0. The value jitter of mutation is $\pm 10\%$. The genetic algorithm works for 10 generation.

Since the parameters that maximise the performance of the robot swarm differ at different population sizes. Therefore, parameter optimisation was carried out separately for population sizes n = 32, 64 and 1.28

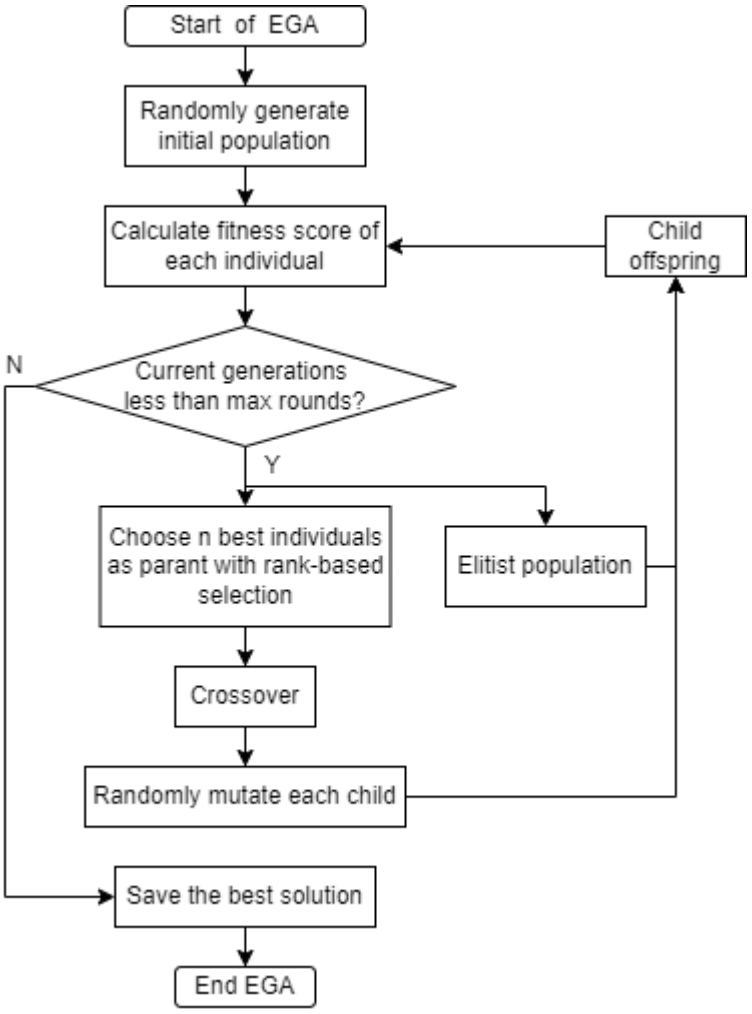


Figure 3-4 : Flowchart of the Elitist Selection Genetic Algorithm

3.3 Analysis

3.3.1 Analysis Methods

To assess the performance of the SPIDER-density and SPIDER-boldness controllers after reproduction on the new platform, data were collected under the following conditions.

To analyse the performance differences between the two controllers at different population sizes and the relationship between population size and robot swarm boldness: the average score and average boldness of the two controllers were counted for 10 repetition of the simulation at population sizes of n = 32, 64, 128, using parameters optimised by genetic algorithms, with an experimental time of 002s.

Analysis of the differences in energy consumption between the two controllers in simulations: The average movement of each robot over time was recorded for simulations with population sizes of n=32, 64, 128 for both controllers using parameters optimised by genetic algorithms for an experimental time of 002s.

Analysis of personality fluctuations and characteristics of robot activity patterns in simulations with both controllers: using parameters optimised by genetic algorithms, the simulation of both controllers 128 at a population size of n = 32,64, was recorded for a particular robot's r_c , boldness, and the average boldness of the entire swarm over time, for an experimental period of 002s.

Testing the robustness of the two controllers: Using parameters optimised by genetic algorithms, the average boldness and average movement of each robot over time was recorded for both controllers in a population catastrophe simulation with population size n = 128 and halved every200 s. The experimental time was 600s.

3.3.2 Results and Discussion

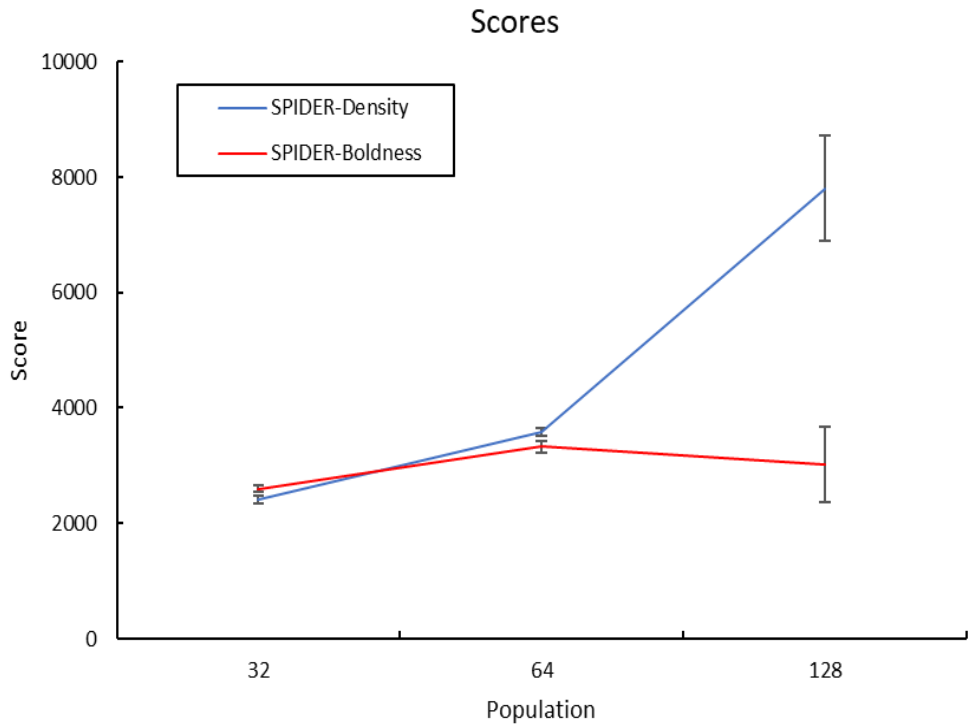


Figure 3-5: Performance metric score comparison for the SPIDER-density controller and SPIDER-boldness controller (95% confidence).

As shown in Figure 3-5 The performance of the SPIDER-density controller improves significantly as the population size of the robot swarm increases, while the SPIDER-boldness controller subsequently underperforms and even falls back at n=128.

The reason for the poor performance of the SPIDER-boldness controller may be related to the fact that the genetic algorithm optimises the parameters to obtain only a locally optimal solution.

Average Boldness of SPIDER Algorithms				
Population	SPIDER-density		SPIDER-boldness	
	mean	s.d.	mean	s.d.
32	67.5213	0.832257	47.70636	1.921598
64	90.79306	1.143994	56.08084	1.584072
128	97.65398	2.149494	119.9487	7.679254

Table 3-1 : Average boldness and standard deviation of 2controllers.

As

Average Boldness of SPIDER Algorithms				
Population	SPIDER-density		SPIDER-boldness	
	mean	s.d.	mean	s.d.
32	67.5213	0.832257	47.70636	1.921598
64	90.79306	1.143994	56.08084	1.584072
128	97.65398	2.149494	119.9487	7.679254

Table 3-1 the average robot boldness increases with population size for both controllers. As the population grows further, the increase in average robot boldness for the SPIDER-density controller slows down, while the increase for the SPIDER-boldness controller becomes larger. At the same time, the SPIDER-density controller yielded more concentrated data over multiple replications, while the SPIDER-boldness controller yielded more fluctuating results.

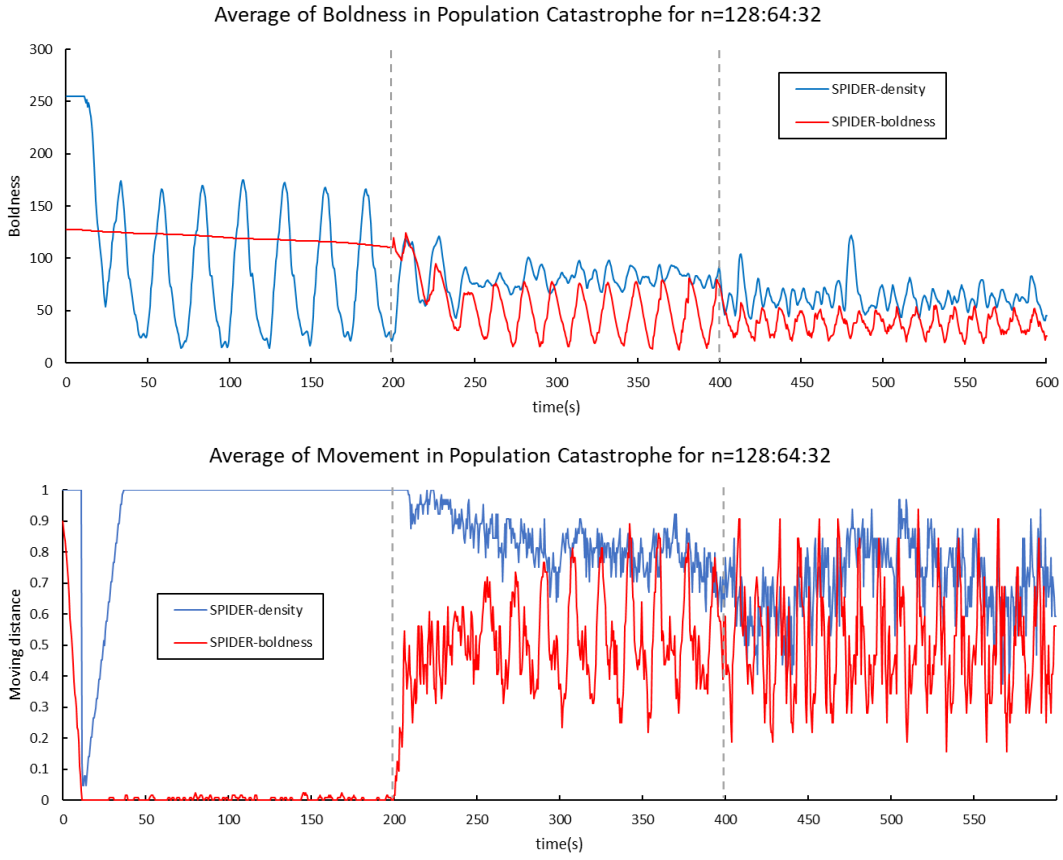


Figure 3-6 Average of boldness (top) and moving distance (bottom) in population catastrophe for 2 controllers . The initial population is n=128, half of the population is randomly eliminated at 200s, at which point n=64, and the general population is randomly eliminated again at 400s, at which point n=32. n=32.

As shown in Figure 3-6 The average movement of the SPIDER-boldness controller is significantly lower than that of the SPIDER-density controller, and the rate of return

to steady state is similar for both controllers.

It is worth noting that at a population size of n = 128, the robot population barely moves after the initial location assignment. Also, the boldness of the robots does not change significantly. However, the robot swarm did not score well at this point. This implies that the genetic algorithm is multi-gene and that the boldness calculation for large populations may not be ideal when performing parameter optimisation. In the next phase of experiments in this project, the improved SPIDER Algorithm is planned to be applied to larger populations, which makes the genetic algorithm inappropriate for the next phase of parameter optimisation.

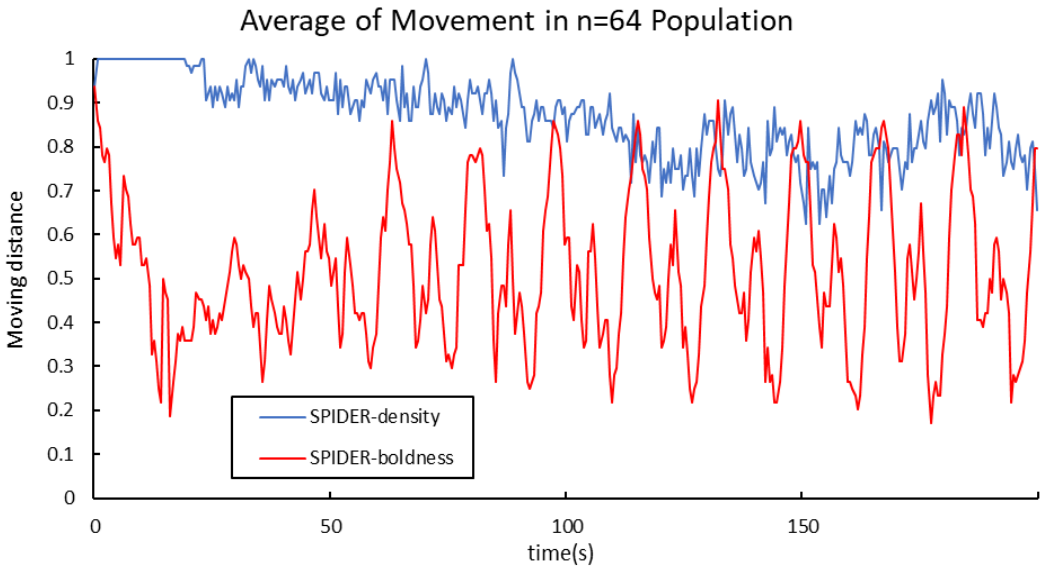


Figure 3-7 : Average movement of each robot over time for 2 controllers for population size n=64.

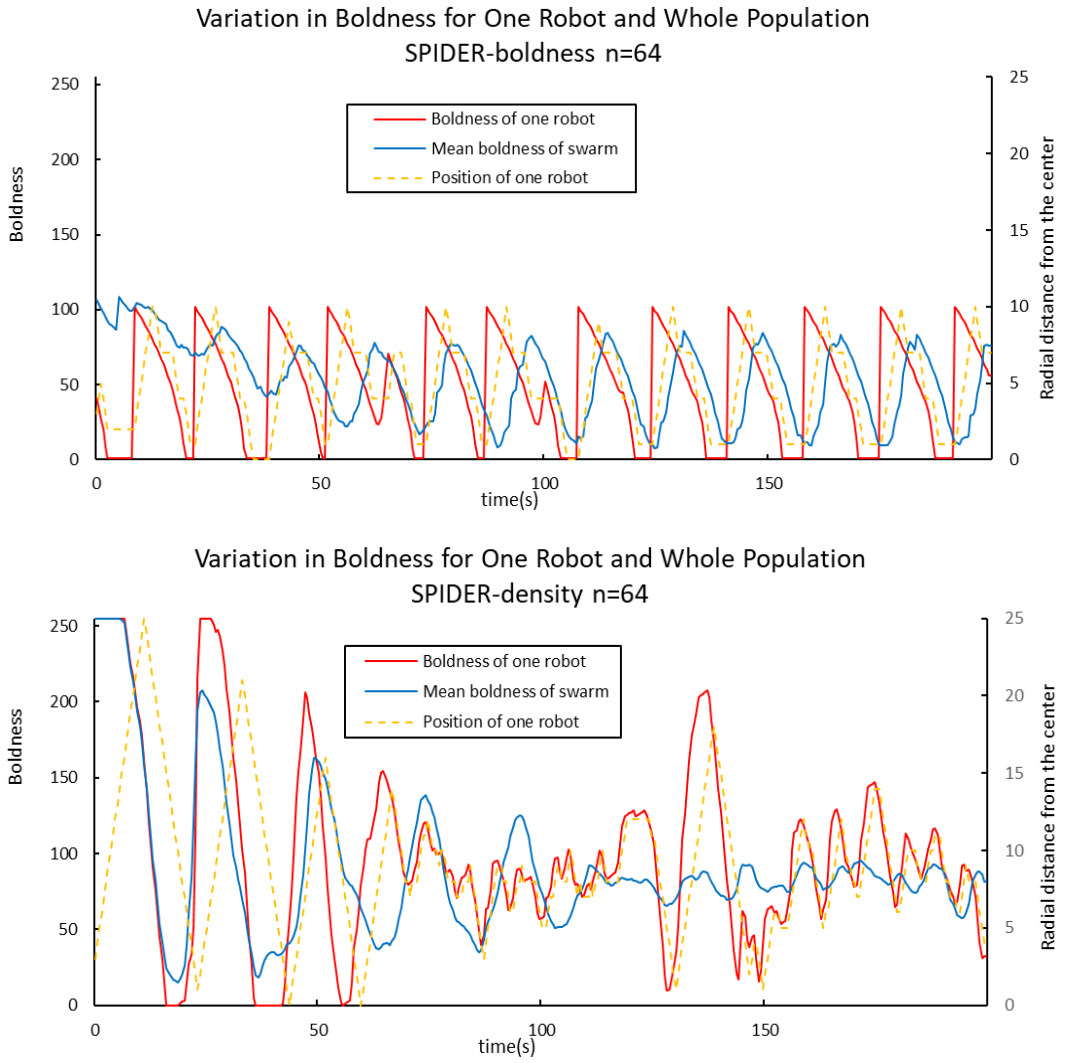


Figure 3-8 : Population size n=64 for a particular robot's r_c The average boldness of a particular robot, boldness, and the average boldness of the entire swarm over time. The top panel shows the SPIDER-boldness controller and the bottom panel shows the SPIDER-density controller.

One simulation for each controller was selected when the scores of the two controllers were similar (n=64) to compare the difference in the average distance travelled by their robots over time, as Figure 3-7 The mean distance travelled for the SPIDER-boldness controller was significantly lower than that for the SPIDER-density controller.

As shown in Figure 3-8 The robot with the SPIDER-boldness controller moves more regularly, with less movement, and does not move farther from the centre. The robot with the SPIDER-density controller moves more unpredictably and repeatedly over long distances from the centre to the edge of the field.

3.4 Chapter Conclusion

This section illustrates the two controller implementations of the SPIDER Algorithm, provides pseudo-code for the SPIDER-density algorithm and SPIDER-boldness, methods for optimising and testing the SPIDER Algorithm, and experimental results and discussion.

It is clear from the experiments that the SPIDER-density algorithm performs better when the population size is increased; the SPIDER-boldness algorithm consumes less energy, and the robot's position is more stable for the same performance. The genetic algorithm does not perform well when optimised for large populations with multiple parameters and may give locally optimal solutions.

4 Improvement of SPIDER Algorithm

This project is divided into two main phases. The first phase reproduces and evaluates the SPIDER Algorithm. The second phase improves the SPIDER Algorithm used to complete the hunting task, obtains the new SPIDER-hunt algorithm and evaluates it. In this section, the work of the second phase is presented.

The first section describes the implementation of the SPIDER-hunt algorithm, including the simulation environment, the task and performance metric, and the logic of the controller. The second section evaluates the experimental design and results used by the algorithm.

4.1 Implementation

4.1.1 General Framework page1

To run the SPIDER-hunt algorithm, the experimental environment constructed for this phase was a rectangular field, where the distance from any location within the field to the central area was no longer correlated with the level of danger. At the start of the experiment, the robot swarm starts moving randomly once it is released from the base in the centre of the field and no further information needs to be sent to the base thereafter. Prey will be refreshed at random locations on the field. The prey does not move, but has different danger levels. Robots that find prey will attempt to capture it. Robots that fail the hunt attempt are scrapped and removed from the field. After the current prey has been successfully captured, the next prey will be refreshed at a random location in the field.

4.1.2 Robot Swarm and Experimental Arena

The experimental site is based on matplotlib and is implemented by a Cartesian coordinate system as shown in Figure 4-1 as shown. This is a rectangular area of 50x50. The size of the field can be changed according to the needs of the experiment.

The robot swarm is the same as described in Sec.3.1.3 and is still considered as an ensemble of particles with a given position at a fixed time. Figure 4-1 The yellow and green masses in the middle represent the robots, when a swarm of population size n=256 is active in the field, again the population size can be modified to suit the experiment.

The red diamond in the diagram represents prey. When prey is within the detection range of a robot, the robot records the location of the prey and broadcasts this information to its neighbours at a certain frequency. The robot that receives the prey location information will travel to the location of the prey and act as a relay in the diagram to continue sharing the location of the prey to its neighbourhood robots.

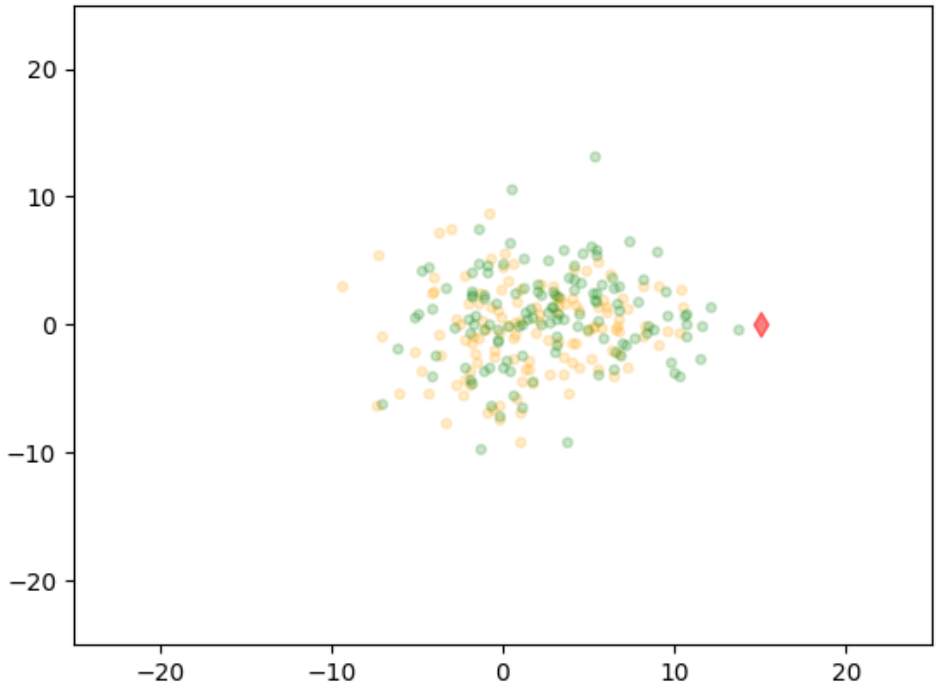


Figure 4-1 : Simulation environment for running the SPIDER-hunt algorithm . The figure shows a rectangular field of 50x50, with the red diamond representing the prey. At this point, a swarm of robots with a population size of n=256 is moving around the field, and the yellow and green dots indicate these robots. The yellow dots indicate that the robot's current boldness is greater than the boldness needed to launch an attack and will attempt to hunt when it encounters prey, while the green dots indicate that the robot's boldness is low and will choose to wait cautiously for its partner when faced with prey.

4.1.3 Task and Performance Metric 1.5 pages

A swarm of robots hunts for prey of unknown danger level D in the environment; if a robot is around prey when the number of robots$N \leq D$ When an attack is launched, that robot is damaged; when$N > D$ time the hunt is attempted, the hunt is successful and the swarm score is increased by one. However, if a robot waits too long for a companion to ensure safety, the efficiency of the hunt is reduced.

The task expects the robot swarm to catch as much prey as possible with as little loss as possible, balancing the relationship between loss and efficiency and adaptively regulating the swarm's decisions in the face of unknown risks. performance metric is the number of prey caught by the swarm in a given time L. Also, the number of robots scrapped during the simulation, N_{loss}, is counted.

4.1.4 SPIDER-Hunt Algorithm 2 pages

The SPIDER-hunt algorithm is described in the pseudo-code of the algorithm3. The main steps of the algorithm are repeated every 0.5s of a second.

Boldness-sensitive adaptive decision mechanism. The method for calculating

boldness is essentially the same as the SPIDER-boldness algorithm. However, in this case the size of the robot's boldness no longer determines the robot's destination coordinates, but is related to whether the robot attempts to hunt or not. When the robot's boldness is greater than the predetermined boldness required for an attack, the robot around the prey will initiate the attack, otherwise the robot will continue to wait around the prey.

Algorithm 3 SPIDER-hunt algorithm pseudo-code

procedure input parameters a, b, c for SPIDER-HUNT(a, b, c)
Initialize for individual robots separately
  $(x_r, y_r) \leftarrow$ robots placed in ring formation in the center area
  $bold \leftarrow$ set random boldness $\in [0, 255]$
repeat every 0.5s
  for $i \in Robots$ {for all robots} **do**
    Calculate boldness of robot i:
      $mean(neighbor\ boldness) \leftarrow$ average boldness of neighbors within range
      $f(bold) \leftarrow \frac{a}{bold}$; $g(bold) \leftarrow \frac{b}{bold}$; $addict \leftarrow 1$
      if $mean(neighbor\_boldness) > boldness$ **then**
        $bold = bold + f(bold) \times addict - g(bold)$
        $addict = addict \times c$
      else
        $bold = bold - g(bold)$; $addict = 1$
      endif
      $bold =$ limitrange$(bold, [0, bold])$
    if robot i is at the prey location **then**
      if robot *i* has just reached prey position **then**
        $bold$ of the robot *i* is reduced by f
      Robot i signals the prey location to nearby robots.
      if $bold > M$ **then** the robot tries to grab the prey.
        if there are more than D robots nearby it will succeed. Otherwise, robot i will fail and be destroyed. $bold$ of the robots nearby is reduced by f.
      else
        robot i makes a move:
          if robot i knows the prey location **then** it signals the prey location to nearby robots and steps towards the prey. Otherwise, it makes a random move.
      endif
  end for
until experiment time runs out

Death Threat. When a robot is scrapped due to a failed hunt, the neighbours of that robot will be affected by its damage signal and the boldness is reduced f . This process simulates a 'death threat', where a trial and error by a fellow robot will effectively alert other robots to act with caution when faced with an unknown level of danger. This prevents robots from being scrapped in bulk in the presence of high-risk prey.

Caution. When a robot first arrives near its prey, its boldness decreases f . Robots that have strayed from the swarm, even if they have spotted prey, are unable to pass information about the prey's location to the swarm in time to attract more robots. This feature is designed to prevent robots that have left the swarm but still have a large enough boldness to attack alone and scrap when they see prey.

4.2 Analysis

4.2.1 Analysis Methods

In order to test the effectiveness of the SPIDER-hunt algorithm, two other boldness-independent controllers were used for the same task and their performance was compared to that of the SPIDER-hunt algorithm. The two controllers used as a baseline are: direct hunting, where the robot attacks whenever it encounters prey, and fixed waiting time, where the robot encounters prey and then waits around for a preset fixed amount of time before attacking.

In addition to the task of refreshing prey at random locations again, the test also includes the task of sequentially refreshing a set of prey at a fixed location with a fixed number of prey. If the robot swarm hunts all prey early, the simulation can end when the last prey is successfully hunted. This is partly to close the gap in superior performance and partly to exclude situations where the remote location of the prey refresh makes the robot swarm spend too much time searching for the prey, resulting in inaccurate results. However, the danger level of the prey remains random. Adaptive unknown danger levels are an important criterion to examine.

The data were collected from the initial population size of n=256,128 and 64 the results were obtained by repeating the simulation three 10times for the SPIDER-hunt, direct hunting, and fixed waiting time controllers, all with an experimental duration of 1000s. The waiting time for the fixed waiting time controller was 5 s. The parameters used for the hunt controller were a=1.4, b=1, c=1.2. Since the experimental results in Chapter 3 showed that the genetic algorithm did not give satisfactory results for the parameter optimisation of the SPIDER-boldness algorithm, this phase of the work did not use the genetic algorithm for parameter optimisation, but used empirically set parameters for the experiments.

In addition, to observe the variation in boldness of a single robot and the average boldness of the entire robotic swarm, the variation in boldness of a robot and the average boldness of the swarm overtime was counted for a single simulation using the SPIDER-hunt controller at an initial population size of n=256. The duration of the experiments

was both 1000s.

4.2.1Results and Discussion

Population	SPIDER-Hunt		Direct hunting		Fixed waiting time	
	Score	Number of robots destroyed	Score	Number of robots destroyed	Score	Number of robots destroyed
256	11.1	40.5	3	239.9	11.6	45.5
128	3	14.6	1.1	120.9	1.8	72.8
64	0.6	13.8	0.3	35.4	0.9	35.7

Table 4-1 : Scores and number of destroyed robots for the three controllers applied to different initial population sizes when the prey position is randomly refreshed.

As shown in

Population	SPIDER-Hunt		Direct hunting		Fixed waiting time	
	Score	Number of robots destroyed	Score	Number of robots destroyed	Score	Number of robots destroyed
256	11.1	40.5	3	239.9	11.6	45.5
128	3	14.6	1.1	120.9	1.8	72.8
64	0.6	13.8	0.3	35.4	0.9	35.7

Table 4-1 As shown, the SPIDER-hunt controller and the fixed waiting time controller performed similarly in the random hunting simulation when the initial population size was large. As the initial population size decreases, the fixed waiting time controller's score decreases more and more robots are lost. Direct hunting controller loses almost all robots at larger population sizes, but after the initial population size decreases, the attrition level gradually approaches that of the fixed waiting time controller.

Population	SPIDER-Hunt			Direct hunting			Fixed waiting time		
	Score	Number of robots destroyed	Total time(s)	Score	Number of robots destroyed	Total time(s)	Score	Number of robots destroyed	Total time(s)
256	10	2.6	202.15	7.1	220.2	831.1	10	9.8	216.45
128	4.1	2.1	1000	2.1	126.9	1000	5.9	124.4	1000
64	0.8	2.7	1000	0.5	61.6	1000	1.2	62.7	1000

Table 4-2 : The number of scores and destroyed robots for each of the three controllers applied to different initial population sizes of robot swarms when the prey location and number are fixed. The number of prey is fixed at 10.

As

Population	SPIDER-Hunt			Direct hunting			Fixed waiting time		
	Score	Number of robots destroyed	Total time(s)	Score	Number of robots destroyed	Total time(s)	Score	Number of robots destroyed	Total time(s)
256	10	2.6	202.15	7.1	220.2	831.1	10	9.8	216.45
128	4.1	2.1	1000	2.1	126.9	1000	5.9	124.4	1000
64	0.8	2.7	1000	0.5	61.6	1000	1.2	62.7	1000

Table 4-2 shown, both the SPIDER-hunt controller and the fixed waiting time controller, in the fixed prey simulation, were able to complete the hunt faster when the initial population size was larger. The fixed waiting time controller lost almost all robots when the initial population size was reduced. The SPIDER-hunt controller, on the other hand, maintains a very small amount of attrition regardless of the population size, demonstrating excellent scalability and robustness.

These experimental results demonstrate the effectiveness of the SPIDER-hunt controller. The SPIDER-hunt controller is able to hunt efficiently when the population is large and avoid losses when the population is small and the hunting conditions are unfavourable.

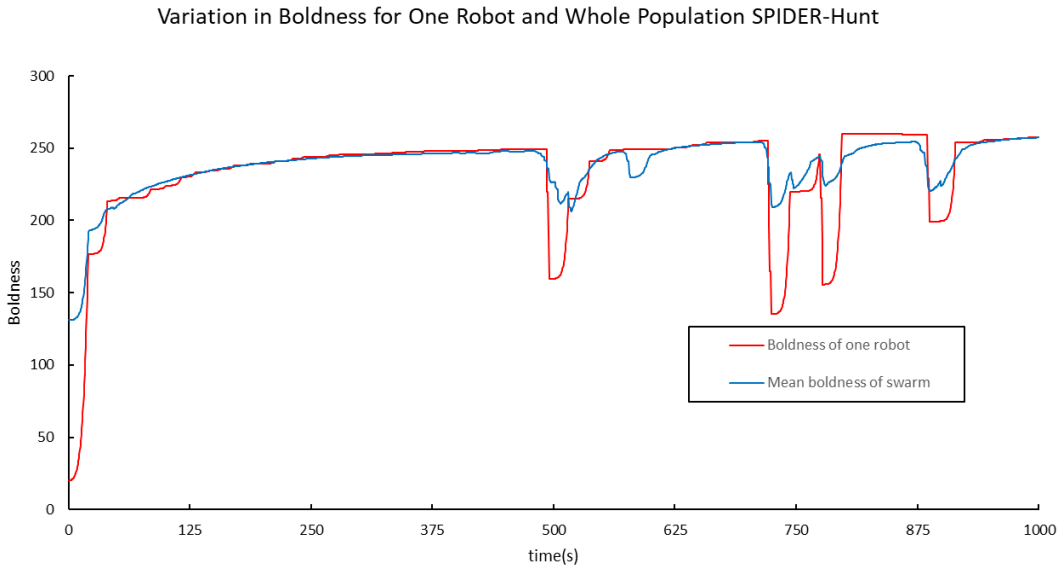


Figure 4-2 : Random refreshing of prey position, variation in boldness for one robot and whole population when the SPIDER-Hunt controller is applied to a robot swarm with an initial population size of 256.

As Figure 4-2 As shown, both the individual robot's boldness and the average boldness of the entire swarm levelled off after a certain value and were stable. Although there were several fluctuations due to cautionary characteristics or death signals from companions, they all eventually returned to steady-state values.

4.3 Chapter Conclusion

This section provides an implementation of the SPIDER-hunt algorithm, analysis methods, experimental results and discussion.

Comparing the performance of the SPIDER-hunt controller with two other controllers without the boldness mechanism, it was found that the SPIDER-hunt controller is able to achieve both efficient hunting when the population is large and effective loss avoidance when the population is small and hunting conditions are unfavourable, with good scalability and flexibility.

5 Evaluation

5.1 SPIDER Algorithm

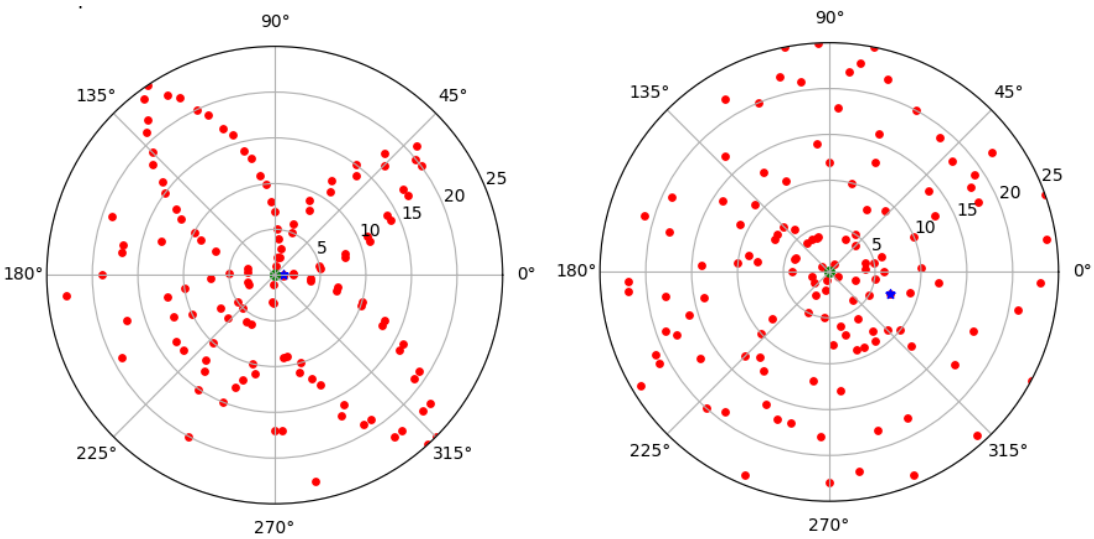


Figure 5-1 : Performance of the two SPIDER controllers when simulated with parameters optimised by a genetic algorithm, n=128. Left is the SPIDER-density controller with a score of 25 at the current moment. Right is the SPIDER-boldness controller with a score of 12 at the current moment.

The results of the genetic algorithm work are quite interesting when using the two controllers optimized for the SPIDER Algorithm. For the SPIDER-density algorithm, the optimal parameters tended to have the robotic swarm move repeatedly in the radial direction, forming a long spiral-like dynamic chain extending from the centre to the edges. Such a movement, although energy-intensive and the boldness of the individuals changing too quickly for the initial simulation of social spider habits, gives almost a perfect performance metric.

For the SPIDER-boldness algorithm, the optimal parameter tends to make the swarm of robots almost immobile once it initially arrives from the release position, giving a random distribution in a space-like effect. Since the boldness of each robot in the swarm is initialised with a random value, this distribution is closer to the behavioural pattern of social spiders in nature, but the score is low and unstable.

The study of swarm robotics could consider more engineering applications and should not be limited by the biological inspiration that inspired it. Accordingly, the SPIDER-density algorithm has better performance for the task of forming long dynamic chains from a central safe area to a distant location.

5.2 SPIDER-Hunt Algorithm

Figure 5-2 : Performance of the SPIDER-hunt algorithm in the simulation. The red diamond is the prey. The yellow and green translucent dots represent the robot. The yellow dots indicate that the robot's current boldness is greater than the boldness needed to launch an attack and will attempt to hunt when it encounters prey. A green dot means that the robot has a low boldness and will wait cautiously for a partner when faced with prey.

As shown in Figure 5-2, a swarm of robots using the SPIDER-hunt controller is shown moving towards the prey. At this point, the robots that reach the prey first enter a low-boldness wait state due to caution or the death of a companion, effectively avoiding attrition when the robots are not dense enough to try to hunt. The robots in the gap are mostly in a higher boldness state, and as they approach the low boldness robots in the waiting state, they will increase their boldness and re-enter the hunting state, realising the decision-making process of "support increases, the probability of successful hunting increases, and hunting is attempted".

The SPIDER-boldness method of calculating boldness in hunting tasks is used instead of the SPIDER-density calculation. This is because if the boldness of a robot is only related to the density of its neighbours, it can result in increased losses when faced with a high-risk prey that cannot be hunted successfully even if the entire swarm is gathered, even if the robots attempting to hunt are constantly destroyed, the boldness of the surrounding robots will still rebound. In contrast, when using a boldness-sensitive algorithm, if no new bold individuals enter detection range, the boldness of the group and the individuals will stabilise and not grow forever. This is reflected in the decision-making process of "not attempting to attack extremely dangerous objects when no new support is available, thus reducing losses".

The flexibility of the SPIDER-hunt algorithm is demonstrated by its ability to make flexible decisions for different situations. The SPIDER-hunt algorithm also shows excellent scalability and robustness by maintaining a very small amount of attrition regardless of the size of the population.

The SPIDER-hunt algorithm has performed as well as expected through the testing of the system.

5.3 Personality and Swarm Robotic

Research related to the idea of a swarm robot algorithm designed with personality traits in mind, in addition to having, for example, "the ability to intuitively build heterogeneity in behaviour and decision making in a hardware homogenous group, allowing the system to generate self-organised coordination; to generate desired action jumps after personality changes caused by specific contingencies; to reduce the conflict between stability and adaptability in group learning contradictions, while making it easier for humans to understand the actions of robotic people." The advantages of this approach also have the problem of increased computational complexity and reduced scalability.

The research in this project offers new ideas to break this situation. In the past, when introducing personality traits for swarm robots, it was often possible to go for personality complexity. However, the research in this project demonstrates that a single personality with a targeted algorithm can perform equally well on a given task. Such an algorithm would combine the advantages of personality algorithms with the distributed control of traditional swarm robots.

6 Conclusion

6.1 Main Contribution

The project confirms an acceptable success by making these contributions.

A comprehensive review of research on algorithms related to personality traits in the field of multi-robot systems summarises the characteristics, application directions, strengths and weaknesses of these algorithms.

The SPIDER-density Algorithm and SPIDER-boldness Algorithm were re-implemented in Python in a different environment and the performance of both SPIDER Algorithms was re-evaluated.

The effectiveness, robustness, scalability and flexibility of the SPIDER-hunt Algorithm are well supported by the experimental results. The results are well supported.

It is explored whether a single personality differentiation is a practical idea suitable for the design of swarm robotics algorithms.

6.2 Limitations

There are two main limitations of this project.

The genetic algorithm did not work for a large enough number of generations or a large enough population size. This may have been responsible for the poor parameter optimisation results, and the SPIDER-boldness algorithm failed to perform with the same excellent performance as in the original authors' study.

The three swarm robot controllers involved in this study have all only been simulated in a virtual environment that omits the effects of many physical factors. To truly validate the usefulness of the algorithms, the controllers need to be extended to a more simulated platform to test their performance or run on a real machine.

6.3 Future Work

Despite the many promising achievements of the SPIDER-hunt algorithm, it is still a prototype.

For future work, the first consideration for developers is to choose other metaheuristics to optimise the parameters for the SPIDER-hunt algorithm. In addition, this algorithm could be combined with other research results on the hunting habits of social spiders, such as the one mentioned in [4], where spiders in small populations attack their prey

more decisively than spiders in large populations. Scientists speculate that this counter-intuitive phenomenon may be due to the fact that spiders in small populations rely more on speed rather than numbers to hunt. Adding this feature to the SPIDER-hunt algorithm could improve the problem of low prey capture in small populations.

The results of this study as a swarm robot algorithm can be simulated in the future on platforms with a higher degree of simulation, selecting the right components for real-world testing, etc.

References

[1] E. R. Hunt, G. Jenkinson, M. Wilsher, C. P. Dettmann, and S. Hauert, "SPIDER: a Bioinspired Swarm Algorithm for Adaptive Risk-Taking, " *The 2020 Conference on Artificial Life*, 2020, doi: 10.1162/isal_a_00279.

[2] E. R. Hunt, B. Mi, C. Fernandez, B. M. Wong, J. N. Pruitt, and N. Pinter-Wollman, "Social interactions shape individual and collective personality in social spiders," *Proceedings of the Royal Society B: Biological Sciences*, vol. 285, no. 1886, p. 20181366, Sep. 2018, doi: 10.1098/rspb.2018.1366.

[3] "Stegodyphus dumicola," *Wikipedia*, Dec. 30, 2021. https://en.wikipedia.org/wiki/Stegodyphus_dumicola#cite_note-refname7-11 (accessed Jan. 16, 2022).

[4] C. M. Wright, C. N. Keiser, and J. N. Pruitt, "Personality and morphology shape task participation, collective foraging and escape behaviour in the social spider Stegodyphus dumicola," *Animal Behaviour*, vol. 105, pp. 47-54, Jul. 2015, doi: 10.1016/j. anbehav.2015.04.001.

[5] E. Bonabeau, M. Dorigo, and G. Theraulaz, *Swarm intelligence : from natural to artificial systems*. New York ; Oxford: Oxford University Press, 2010.

[6] M. Dorigo, M. Birattari, and M. Brambilla, "Swarm robotics," *Scholarpedia*, vol. 9, no. 1, p. 1463, 2014, doi: 10.4249/ scholarpedia.1463.

[7] G. Dudek, M. Jenkin, E. Milios, and D. Wilkes, "A taxonomy for swarm robots," *IEEE Xplore*, Jul. 01, 1993. https://ieeexplore.ieee.org/document/583135 (accessed Sep. 18, 2021).

[8] G. Beni, "From Swarm Intelligence to Swarm Robotics," *Swarm Robotics*, pp. 1-9, 2005, doi: 10.1007/978-3-540- 30552-1_1.

[9] A. J. C. Sharkey, "Swarm robotics and minimalism," *Connection Science*, vol. 19, no. 3, pp. 245-260, Aug. 2007, doi: 10.1080/09540090701584970.

[10] T. Fukuda, T. Ueyama, Y. Kawauchi, and F. Arai, "Concept of cellular robotic system (CEBOT) and basic strategies for its realization, " *Computers & Electrical Engineering*, vol. 18, no. 1, pp. 11-39, Jan. 1992, doi: 10.1016/0045-7906(92)90029-d.

[11] E. Şahin, "Swarm Robotics: From Sources of Inspiration to Domains of Application," *Swarm Robotics*, pp. 10-20, 2005, doi: 10.1007/978-3-540-30552-1_2.

[12] M. Dorigo, M. Birattari, and M. Brambilla, "Swarm robotics," *Scholarpedia*, vol. 9, no. 1, p. 1463, 2014, doi: 10.4249/ scholarpedia.1463.

[13] M. Dorigo, G. Theraulaz, and V. Trianni, "Swarm Robotics: Past, Present, and Future," *Proceedings of the IEEE*, vol. 109, no. 7 , pp. 1152-1165, Jul. 2021, doi:

10.1109/jproc.2021.3072740.

[14] S. Camazine and E. Al, *Self-organization in biological systems*. Princeton: Princeton University Press, 2003.

[15] P. Baran, "Reliable Digital Communications Systems Using Unreliable Network Repeater Nodes," www.rand.org, Jan. 1960, Accessed: Sep. 20, 2021. [Online]. Available: https://www.rand.org/pubs/papers/P1995.html.

[16] A. Cheraghi, S. Shahzad, and K. Graffi, "Past, Present, and Future of Swarm Robotics." Accessed: Sep. 20, 2021. [Online].

[17] M. Brambilla, E. Ferrante, M. Birattari, and M. Dorigo, "Swarm robotics: a review from the swarm engineering perspective," *Swarm Intelligence*, vol. 7, no. 1, pp. 1-41, Jan. 2013, doi: 10.1007/s11721-012-0075-2.

[18] J. Hereford, "BEECLUST swarm algorithm: analysis and implementation using a Markov chain model," *International Journal of Innovative Computing and Applications*, vol. 5, no. 2, p. 115, 2013, doi: 10.1504/ijica.2013.053185.

[19] Heiko Hamann and Springerlink (Online Service, *Swarm Robotics: A Formal Approach*. Cham: Springer International Publishing, 2018.

[20] "File:Population distribution.svg - Wikimedia Commons," *Wikimedia.org*, 2020. https://commons.wikimedia.org/wiki/File:Population_distribution.svg (accessed Sep. 27, 2021).

[21] M. A. Kessler, "Self-Organization of Sorted Patterned Ground," *Science*, vol. 299, no. 5605, pp. 380-383, Jan. 2003, doi: 10.1126/science.1077309.

[22] B. Shucker and J. K. Bennett, "Scalable Control of Distributed Robotic Macrosensors," *Distributed Autonomous Robotic Systems 6*, pp. 379-388, doi: 10.1007/978-4-431-35873-2_37.

[23] R. Mayet, J. Roberz, T. Schmickl, and K. Crailsheim, "Antbots: A Feasible Visual Emulation of Pheromone Trails for Swarm Robots, " *Lecture Notes in Computer Science*, pp. 84-94, 2010, doi: 10.1007/978-3-642-15461-4_8.

[24] R. A. Russell, "Heat trails as short-lived navigational markers for mobile robots," *IEEE Xplore*, Apr. 01, 1997. https://ieeexplore.ieee.org/document/606882 (accessed Oct. 08, 2021).

[25] P. M. Maxim, W. M. Spears, and D. F. Spears, "Robotic Chain Formations," *IFAC Proceedings Volumes*, vol. 42, no. 22, pp. 19 -24, 2009, doi: 10.3182/20091006-3-us-4006.00004.

[26] R. Groβ and M. Dorigo, "Evolution of Solitary and Group Transport Behaviors for Autonomous Robots Capable of Self-Assembling, " *Adaptive Behavior*, vol. 16, no. 5, pp. 285-305, Oct. 2008, doi: 10.1177/1059712308090537.

[27] C. Anderson, G. Theraulaz, and J.-L. . Deneubourg, "Self-assemblages in insect

societies," *Insectes Sociaux*, vol. 49, no. 2, pp. 99-110, May 2002, doi: 10.1007/s00040-002-8286-y.

[28] "The chemical basis of morphogenesis," *Philosophical Transactions of the Royal Society of London. series B, Biological Sciences*, vol. 237, no. 641, pp. 37-72, Aug. 1952, doi: 10.1098/rstb.1952.0012.

[29] "Swarm-bots project," www.swarm-bots.org. http://www.swarm-bots.org/ (accessed Oct. 09, 2021).

[30] N. Eliot, D. Kendall, A. Moon, M. Brockway, and M. Amos, "Void Reduction in Self-Healing Swarms," *direct.mit.edu*, Jul. 01, 2019. https://direct.mit.edu/isal/proceedings/isal2019/87/99204 (accessed Oct. 09, 2021).

[31] M. Rubenstein and W.-M. Shen, "Scalable self-assembly and self-repair in a collective of robots," *IEEE Xplore*, Oct. 01, 2009 . https://ieeexplore.ieee.org/abstract/document/5354716 (accessed Oct. 09, 2021).

[32] M. Resnick, Turtles, termites, and traffic jams : explorations in massively parallel microworlds. Cambridge, Mass.: Mit Press, [Ca, 2002.

[33] J. Chen and M. Gauci, "Segregation in Swarms of E-Puck Robots Based on the Brazil Nut Effect," *AAMAS '12: Proceedings of the 11th International Conference on Autonomous Agents and Multiagent Systems*, vol. 1, pp. 163-170, Jun. 2012.

[34] D. Payton, M. Daily, R. Estowski, M. Howard, and C. Lee, "Pheromone Robotics," *Autonomous Robots*, vol. 11, no. 3, pp. 319 -324, 2001, doi: 10.1023/a:1012411712038.

[35] A. Howard, M. J. Matarić, and G. S. Sukhatme, "Mobile Sensor Network Deployment using Potential Fields: A Distributed, Scalable Solution to the Area Coverage Problem," *Distributed Autonomous Robotic Systems 5*, pp. 299-308, 2002, doi: 10.1007/978-4-431-65941 -9_30.

[36] M. Ballerini *et al.*, "Interaction ruling animal collective behavior depends on topological rather than metric distance: Evidence from a field study," *Proceedings of the National Academy of Sciences*, vol. 105, no. 4, pp. 1232-1237, Jan. 2008, doi: 10.1073/ pnas.0711437105.

[37] J. Wessnitzer and C. Melhuish, "Collective Decision-Making and Behaviour Transitions in Distributed Ad Hoc Wireless Networks of Mobile Robots: Target-Hunting," *Advances in Artificial Life*, pp. 893-902, 2003, doi: 10.1007/978-3-540-39432-7_96.

[38] G. Theraulaz, E. Bonabeau, and J-N. Denuebourg, "Response threshold reinforcements and division of labour in insect societies, " *Proceedings of the Royal Society of London. series B: Biological Sciences*, vol. 265, no. 1393, pp. 327-332, Feb. 1998, doi: 10.1098/rspb.1998.0299.

[39] W. Agassounon and A. Martinoli, "Efficiency and robustness of threshold-based

distributed allocation algorithms in multi-agent systems," *Proceedings of the first international joint conference on Autonomous agents and multiagent systems part 3 - AAMAS '02*, 2002, doi: 10.1145/545056.545077.

[40] C. F. Chung and T. Furukawa, "A reachability-based strategy for the time-optimal control of autonomous pursuers," *Engineering Optimization*, vol. 40, no. 1, pp. 67-93, Jan. 2008, doi: 10.1080/03052150701593133.

[41] J. P. Hespanha, H. J. Kim, and S. Sastry, "Multiple-agent probabilistic pursuit-evasion games," *IEEE Xplore*, Dec. 01, 1999. https://ieeexplore.ieee.org/document/831290 (accessed Oct. 09, 2021).

[42] L. Schenato, S. Oh, S. Sastry, and P. Bose, "Swarm Coordination for Pursuit Evasion Games using Sensor Networks," *IEEE Xplore* , Apr. 01, 2005. https://ieeexplore.ieee.org/abstract/document/1570487 (accessed Oct. 09, 2021).

[43] T. Yasuda, K. Ohkura, T. Nomura, and Y. Matsumura, "Evolutionary swarm robotics approach to a pursuit problem," *IEEE Xplore*, Dec. 01, 2014. https://ieeexplore.ieee.org/abstract/document/7009182 (accessed Oct. 09, 2021).

[44] A. L. Christensen, R. O'Grady, and M. Dorigo, "From Fireflies to Fault-Tolerant Swarms of Robots," *IEEE Transactions on Evolutionary Computation*, vol. 13, no. 4, pp. 754-766, Aug. 2009, doi: 10.1109/tevc.2009.2017516.

[45] C. Melhuish, O. Holland, and S. Hoddell, "Convoying: using chorusing to form travelling groups of minimal agents," *Robotics and Autonomous Systems*, vol. 28, no. 2-3, pp. 207-216, Aug. 1999, doi: 10.1016/s0921-8890(99)00018-4.

[46] M. Brambilla, C. Pinciroli, M. Birattari, and M. Dorigo, "A reliable distributed algorithm for group size estimation with minimal communication requirements," *IEEE Xplore*, Jun. 01, 2009. https://ieeexplore.ieee.org/document/5174691.

[47] A. M. Naghsh, J. Gancet, A. Tanoto, and C. Roast, "Analysis and design of human-robot swarm interaction in firefighting," *IEEE Xplore*, Aug. 01, 2008. https://ieeexplore.ieee.org/document/4600675 (accessed Oct. 11, 2021).

[48] J. D. Hasbach and M. Bennewitz, "The design of self-organizing human-swarm intelligence," *Adaptive Behavior* , p. 105971232110175, Jul. 2021, doi: 10.1177/10597123211017550.

[49] Y. Song, X. Fang, B. Liu, C. Li, Y. Li, and S. X. Yang, "A novel foraging algorithm for swarm robotics based on virtual pheromones and neural network," *Applied Soft Computing*, vol. 90, p. 106156, May 2020, doi: 10.1016/j.asoc.2020.106156.

[50] W. R. Revelle and K. Scherer, "Personality and emotion," *Oxford Companion to Emotion and the Affective Sciences*, pp. 304 -305, 2009, Accessed: Oct. 02, 2021. [Online]. Available: https://www.scholars.northwestern.edu/en/publications/personality-and-emotion.

[51] T. Shibata, K. Ohkawa, and K. Tanie, "Spontaneous behavior of robots for

cooperation. Emotionally intelligent robot system, " *IEEE Xplore*, Apr. 01, 1996. https://ieeexplore.ieee.org/document/506527 (accessed Oct. 05, 2021).

[52] D. Yingying, H. Yan, and J. Jing-ping, "Self-organizing multi-robot system based on personality evolution," *IEEE Xplore*, Oct. 01, 2002. https://ieeexplore.ieee.org/document/1176414 (accessed Oct. 05, 2021).

[53] T. Fong, I. Nourbakhsh, and K. Dautenhahn, "A survey of socially interactive robots," *Robotics and Autonomous Systems*, vol. 42, no. 3-4, pp. 143-166, Mar. 2003, doi: 10.1016/s0921-8890(02)00372-x.

[54] R. R. Murphy, C. L. Lisetti, R. Tardif, L. Irish, and A. Gage, "Emotion-based control of cooperating heterogeneous mobile robots, " *IEEE Transactions on Robotics and Automation*, vol. 18, no. 5, pp. 744-757, Oct. 2002, doi: 10.1109/TRA.2002.804503.

[55] S. Sato, A. Nozawa, and H. Ide, "Characteristics of Behavior of Robots with Emotion Model," *IEEJ Transactions on Electronics , Information and Systems*, vol. 124, pp. 1390-1395, Jan. 2004, doi: 10.1541/ieejeiss.124.1390.

[56] A. Gage, "Multi-robot task allocation using affect - ProQuest," www.proquest.com. https://www.proquest.com/docview/ 305122462?pq-origsite=gscholar&fromopenview=true.

[57] S. Givigi and H. Schwartz, "Swarm Robot Systems Based on the Evolution of Personality Traits," *undefined*, 2007, Accessed: Oct. 05, 2021. [Online]. Available: https://www.semanticscholar.org/paper/Swarm-Robot-Systems-Based-on-the-Evolution-of-Givigi-Schwartz/ 1472847848bdad1000c5fdd429f123c415c438c7.

[58] S. C. Banik, K. Watanabe, and K. Izumi, "Task Allocation with a Cooperative Plan for an Emotionally Intelligent System of Multi-Robots, " *IEEE Xplore*, Sep. 01, 2007. https://ieeexplore.ieee.org/abstract/document/4421131 (accessed Oct. 05, 2021).

[59] S. C. Banik, K. Watanabe, M. K. Habib, and K. Izumi, "An emotion-based task sharing approach for a cooperative multiagent robotic system, " *IEEE Xplore*, Aug. 01, 2008. https://ieeexplore.ieee.org/document/4798729 (accessed Oct. 05, 2021).

[60] T. Kuremoto, M. Obayashi, K. Kobayashi, and L.-B. Feng, "Autonomic Behaviors of Swarm Robots Driven by Emotion and Curiosity, " *Lecture Notes in Computer Science*, pp. 541-547, 2010, doi: 10.1007/978-3-642-15615-1_63.

[61] B. Fang, L. Chen, H. Wang, S. Dai, and Q. Zhong, "Research on Multirobot Pursuit Task Allocation Algorithm Based on Emotional Cooperation Factor," *The Scientific World Journal*, vol. 2014, p. e864180, Jul. 2014, doi: 10.1155/2014/864180.

[62] J. Godoy, I. Karamouzas, S. Guy, and M. L. Gini, "Moving in a Crowd: Safe and Efficient Navigation among Heterogeneous Agents, " *Semantic Scholar*, 2016. https://www.semanticscholar.org/paper/Moving-in-a-Crowd%3A-Safe-and-Efficient-Navigation-Godoy-

Karamouzas/6996de08e8e09db8d0aaec39bd9ce0495d5274f5 (accessed Oct. 05, 2021).

[63] J. Guzzi, A. Giusti, L. M. Gambardella, and G. A. Di Caro, "A model of artificial emotions for behavior-modulation and implicit coordination in multi-robot systems," *Proceedings of the Genetic and Evolutionary Computation Conference*, Jul. 2018, doi: 10.1145/ 3205455.3205650.

[64] A. Gil, E. Puerto, J. Aguilar, and E. Dapena, "An emotional model for swarm robotics," *CLEI Electronic Journal*, vol. 22, no. 3, pp. 6:1-6:20, Dec. 2019, doi: 10.19153/cleiej.22.3.6.

[65] R. LC, "NOW YOU SEE ME, NOW YOU DON'T," *Proceedings of the Fifteenth International Conference on Tangible, Embedded, and Embodied Interaction*, Feb. 2021, doi: 10.1145/3430524.3442448.

[66] B. Chen, Y. Zhang, and G. Iosifidis, "Resource Sharing in Public Cloud System with Evolutionary Multi-agent Artificial Swarm Intelligence," *Service-Oriented Computing - ICSOC 2020 Workshops*, pp. 240-251, 2021, doi: 10.1007/978-3- 030-76352-7_25.

[67] M. Santos and M. Egerstedt, "From Motions to Emotions: can the Fundamental Emotions be Expressed in a Robot Swarm?," *International Journal of Social Robotics*, Jul. 2020, doi: 10.1007/s12369-020-00665-6.

[68] "Game theory: Analysis of conflict," *Long Range Planning*, vol. 25, no. 2, p. 130, Apr. 1992, doi: 10.1016/0024-6301(92) 90230-y.

[69] E. R. Hunt, B. Mi, C. Fernandez, B. M. Wong, J. N. Pruitt, and N. Pinter-Wollman, "Social interactions shape individual and collective personality in social spiders," *Proceedings of the Royal Society B: Biological Sciences*, vol. 285, no. 1886, p. 20181366, Sep. 2018, doi: 10.1098/rspb.2018.1366.

[70] S. N. Givigi and H. M. Schwartz, "A game theoretic approach to swarm robotics," *Applied Bionics and Biomechanics*, vol. 3, no. 3 , pp. 131-142, Jan. 2006, doi: 10.1533/abbi.2005.0021.

[71] "Interconnected dynamic systems: An overview on distributed control," *IEEE Control Systems*, vol. 33, no. 1, pp. 76 -88, Feb. 2013, doi: 10.1109/mcs.2012.2225929. -88, Feb. 2013, doi: 10.1109/mcs.2012.2225929.

[72] M. Friedmann, "Simulation of autonomous robot teams with adaptable levels of abstraction," *Semantic Scholar*, 2010. https://www.semanticscholar.org/paper/Simulation-of-autonomous-robot-teams-with-adaptable-Friedmann/ 014e1c7b82fd1399453874032fe422863c0f05ce (accessed Dec. 14, 2021).

[73] L. Ba, Y. Li, and M. S. Yang, "Modelling and Simulation of a Multi-Resource Flexible Job-Shop Scheduling," *International Journal of Simulation Modelling*, vol. 15, no. 1, pp. 157-169, Mar. 2016, doi: 10.2507/ijsimm15(1)co3