THE UNIVERSITY
of EDINBURGH

Robustness of Interaction Control in Robot Swarms

Calum Corrie Imrie

Doctor of Philosophy

Institute of Perception, Action and Behaviour

School of Informatics

University of Edinburgh

2021

Abstract

Robotics is making significant strides in its capabilities and is being integrated into many real-world tasks. Some of these are carried out in unpredictable and dynamic environments, for example, search and rescue operations in a disaster area. Other problems, such as medical applications that require the robots to be very small, have led to significant progress in nanorobotics, in which the processor and actuating abilities are limited due to physical constraints. To solve these problems, we must look to methods that are robust to changes in the environment as well as solutions that draw maximum effect from the abilities of the robot. This may be achieved by utilising the potential of a group of robots often related to the principle of swarm intelligence. In this context, we study emergent phenomena in a group of agents each having simple low-level rules and strongly rely on local communication, resulting in increased flexibility and versatility. Individually the agents cannot perform well. As a collective, however, they are able to complete complex tasks and display implicit cognitive abilities that are critical in the applications of swarm robotics. This Thesis aims to put forward the argument for the strengths of swarm intelligence, and furthermore how it may be incorporated in applications. This Thesis first looks at a swarm having the aim to maximise energy consumption in a fixed-sized environment. The agents can sense the whole environment and augment sensory information with local communication as input for a neural network which is trained by an evolutionary strategy. Although successful for one source, as the number of sources increases frustration builds within the swarm and its effectiveness declines. It is shown that by limiting information through evolving the physical properties, the swarm can avoid this confusion. We have studied models and techniques that do not require explicit learning, and that can adapt as the situation changes. The Thesis first looks at reaction-diffusion equations being deployed onto a swarm and form Turing patterns, which are stable periodic patterns usually either honey-comb spotted patterns or stripes. We then explore how patterns can be induced via the environment. This is deployed onto a virtual Kilobot swarm to demonstrate how Turing patterns can guide robotic systems, even within the limitations of the Kilobot platform. We then present another system which has the aim to separate the agents in the swarm to allow maximum coverage of the environment. Starting with a one-dimensional dynamical system we show how this system operates given a periodic boundary condition. Using this as a foundation, we include a fixed environment, limited range of communication, and finally how the rules can be employed onto individual agents in a two-dimensional environment for maximal coverage. We

compare this to the separation rule in classical swarm systems and show that our system has a smoother distribution allowing it to cover the environment quicker. This can then be taken further to circumvent obstacles or surround objects using the principle of separation.

Acknowledgements

I could not have carried out the following work without the unwavering and enthusiastic support of my supervisor, Dr. Michael Herrmann. I would also like to extend my gratitude to my second supervisor, Dr. Katrin Lohan, for providing fresh perspectives, and to Dr. Georg Martius for giving words of guidance at my annual reviews. I am thankful for the support given by Dr Olaf Witkowski while I was researching evolutionary swarms, and I would like to thank my fellow colleagues, Billy Lyons and Yurdusev Yakup Akan for many stimulating discussions.

Last, and by no means least, many thanks to my sister, Molly, my dad, Corrie, and my mum, Chris, for their endless proof reading, which required a great amount of patience on their part.

This work was funded by the EPSRC: Centre of Doctoral Training in Robotics and Autonomous Systems (Grant number: EP/L016834/1).

Declaration

I declare that this thesis was composed by myself, that the work contained herein is my own except where explicitly stated otherwise in the text, and that this work has not been submitted for any other degree or professional qualification except as specified. Some figures and discussion in Chapter 4 was presented at the Living Machines Conference 2017 (Imrie and Herrmann, 2017). Similarly, contents of Chapter 3 was presented at the Genetic and Evolutionary Computation Conference (GECCO) 2021 (Imrie et al., 2021).

(*Calum Corrie Imrie*)

Table of Contents

List of Figures

Chapter 1

Introduction

Some of the most extraordinary systems found in nature are beautifully simple. Indeed, simple organisms are able to navigate and exploit their incredibly large world, and perform complex tasks. For example, ants are able to find sources of food by simply following a smell left by other ants, and this not only directs them to food but also the quickest route to it. Termites can construct huge structures, and birds flock by merely observing their neighbours movements. We also see wonderful patterns in nature, such as the spots on jaguars, and the stripes on fish. These are all achieved through the simplicity of their rules and the interactions they have with their fellow individuals and environment, which is a phenomena known as self-organisation. With the simplicity of the rules they can be robust to any changes that might occur to the environment. These are traits that would be invaluable in a robotic system, and the research field of swarm robotics aims to capture and harness these features for application-based systems. This Chapter describes the motivation underlying the specific themes and research questions that are investigated, and the over all structure of this Thesis.

1.1 Research questions

The primary aim of this Thesis is to understand how to better exploit the dynamics between the individual agents in simple systems for use in robotic swarms. Agents tend to be simple with limited sensing and actuating capabilities, and it is through interactions with their neighbours that a complex system, the swarm, is formed.

An advantage of robotic systems is that we have control over their design and build, and may impart more information to the system, or changing its dynamics in order to generate different behaviours. Given this flexibility, we start with simple systems and

increase their complexity to provide an insight how to adapt robotic systems to behave in a given manner. We also consider what the robots are actually capable of with regards to replicating the behaviour observed in the simulations. Specifically:

- What are the effects on a swarm of increasing choices?
- Can natural systems give guidance to a swarm through interaction alone?
- What patterns and behaviours may be generated by the same system?
- To what extent can robotic swarms utilise the systems described?

To address these questions we will look at three simple systems as our starting points; an evolutionary swarm with equal choice, a reaction-diffusion system, and an energy function based on the distance between an agent's two neighbours.

1.2 Thesis structure

The Thesis will initially provide an overview of the core fundamentals of swarm robotics in Chapter 2: Swarms: Nature to robotics. The Chapter first explores the idea of self-organisation and the emergent order that can arise in simple systems. An overview of Swarm Intelligence, and how this has been used in robotics to cover a variety of tasks, as well as some insight into the real-world applications, will be discussed.

Chapter 3: Information dynamics in evolving agents, begins by considering a simple swarm that shows the ability to learn how to find a source of energy. This is achieved by an evolutionary algorithm, and the first question explored is the effects of giving more equally valid choices to the system. The swarm starts to deteriorate in performance as it learns to cycle between sources of energy rather than staying at one. It is only through evolving the physical aspects, which leads to the agents evolving to become blind, does the system overcome the difficulty of decision making.

The next Chapter begins with a well-known natural system, reaction-diffusion, that produces periodic patterns that are known as Turing patterns. Not only are the agents able to create Turing patterns by acting as the points in space, they can also be guided by the pattern as well. This includes clustering, forming stripes, and can be influenced further with a simple binary sensor for inducing patterns. This is then shown to also work in a simulated swarm of Kilobots, even with their limitations in communication.

Continuing with the principle of starting with a simple system and building in complexity, in Chapter 5: Maximising an evenly spread swarm in an unknown environment

we start with a simple energy function that when minimised drives the agents to be in the midpoint of their two neighbours. It is shown that the system will always go towards an equidistant configuration, and the system is expanded to include limited agents, and fixed boundaries. This is expanded to a 2-dimensional setup and it is demonstrated that the swarm is able to spread evenly in an environment, including in the presence of obstacles. This same simple behaviour can also be used to surround objects of interest.

In Chapter 6: Conclusions, each Chapter is summarised and an overview of the contribution they each makes to answering the questions posed here is given.

Chapter 2

Swarms: Nature inspiring robotics

Swarm robotics is concerned with understanding how collections or swarms of simple robots utilise simple rules to complete complex tasks. Furthermore, these swarms can adapt to changes within the system, as well as to the environment, and a myriad of different behaviours can be observed. This Chapter gives an overview of the essential building blocks to create such a swarm. First, the concept of self-organisation will be introduced and the systems in nature that inspire swarm intelligence research described. Second, a look at swarm robotics and with the common tasks usually performed by swarms, and consider the burgeoning opportunities of integrating robotic swarms into real world applications.

2.1 Self-organisation

Self-organisation is a fundamental concept in many natural and artificial systems. The principle underlying self-organisation is that a system having simplistic low level rules (or behaviours), may then generate a seemingly high level intelligence with the ability to adapt to changes to perturbations to the system. This is a phenomenon observed in many areas such as biology, physics, chemistry, economics, and philosophy (Haken, 1977).

2.1.1 Emergence

The first formal structure of self-organisation was modelled by Ashby (1947a). Self-organisation is a system in which there is no knowledge of a higher level goal, but instead a collection of lower level goals, which cause simple behaviours in a disor-

dered system. Through the lower level behaviours, however, a pattern and coordination spontaneously appear, giving the impression that there is a higher level goal. A self-organising system may be damaged but has the ability to self-repair.

A high level example of this is human behaviour itself. When encountered with a new environment a human can adapt to the new surroundings with similar but new behaviours. This specific scenario was raised by Ashby (1947b) as he posits that the human brain is a machine which adapts itself to the environment.

Another phenomena in a self-organising system is the concept of emergence, and how complexity can emerge from a system that initially has a lack of structure. The term emergence, as it originates from an intuitive understanding, does not have a strict unified definition in the community (Halley and Winkler, 2008). The closest to such a definition being the system's individual components through their interaction gives rise to a system-wide behaviour, which is perhaps not predictable from the observation of the individual components alone, or as Aristotle phrased it "the whole is something over and above its parts, and not just the sum of them all…" (Halley and Winkler, 2008).

While it applies to higher levels of organisation, it is the study of the low level rules and system dynamics which is of interest when observing the emergence of self-organised systems. Within these dynamics there will usually be an attractor, which will drive the system, and Ashby showed that any system with an attractor will be self-organising, as that attractor is what organises the system (Ashby, 1947a,b).

With emergence there is usually a competition between behaviours, with a single behaviour emerging for the system. In natural systems this can be observed in ants converging on one particular food source (Hölldobler and Wilson, 1990), as well as how languages are formed over time (Berg and Aronoff, 2017). Natural systems having emergent properties have been replicated in simulation. Hashimoto and Ikegami (1996), for example, were able to replicate aspects of emergent language in an artificial setting in which an initial collection of agents had rules of grammar for a language composed of strings of symbols. Through an evolutionary approach, the agents collectively develop an efficient grammar where they can create and understand different words being communicated by different agents. This is an example of how emergence in natural systems can be better understood through simulation, and which allows the testing of hypotheses. The idea of emergence, however, can also be utilised in practical situations, such as the formation of structures in modular self-configurable robots (Bojinov et al., 2000).

There has been work into how to measure the emergence of order, and self-organisation (Fernández et al., 2014), given that an interesting question raised is, if almost all systems can fall under the category of self-organising, when is it useful to label a system as such (Gershenson and Heylighen, 2003).

Furthermore, there is the interesting argument posited by some theorists that emergence is not recognised by the system itself (Corning, 2002), and thus emergence only exists when the system, and subsequent emergent behaviour, is observed. Crutchfield (1994) instead suggests, however, that there are perhaps two kinds of emergence; pattern formation and intrinsic emergence. When pattern formation in a self-organising system emerges the system is not aware of this, and it is instead the observer that takes note. More importantly, the system does not suddenly change behaviour once the outside observer has noticed patterns. Given this, Crutchfield (1994) states that intrinsic emergence is also required to capture the concept of emergence. Intrinsic emergence details that the individual components, while not aware of the pattern formation, utilises the emergent phenomena and adds to the global processing of the system, and thus the system is able to increase its functionality. An example is a capital market, where competitive agents exploit the patterns forming, and the market's prices then emerge to convey all information regarding resources supply, demand, etc (Eugene, 1991) (cited from (Crutchfield, 1994)).

This Thesis does not discuss the implications of the definition of emergence with respect to the results described. It is still, however, perhaps worth the reader considering, particularly intrinsic emergence, while evaluating these data.

2.1.2 Guided self-organisation

In natural systems, self-organisation is an emergent phenomenon, and mathematical models have been developed to better understand the low level dynamics that give rise to the high level ordered system, such as the study of spatial and temporal patterns that emerge in biology (Murray, 2007). Computer simulations have also been employed to simulate the algorithm to observe the similarities between the artificial replication and the observed phenomenon in the natural system, such as, for example, snowflake growth (Demange et al., 2017).

The rules and dynamics for artificial systems, however, do not always have to follow a fixed regime, as artificial systems are not always created to merely replicate another system. Homeostasis was first expressed formally by Claude Bernard in 1865,

though the actual word homeostasis was first used by Cannon (1926), and describes a system that, through modifying its environments, aims to maintain a stable state. If we consider temperature as the state space, the human body aims to keep its body temperature at a stable point (varies with age (Kenney and Munce, 2003)) and will regulate the body if anything was to perturb the system such as infection or hypothermia (Benzinger, 1969; Kiyatkin, 2010). Translating this concept to technology, a room with a thermostat can be considered homeostatic, as the heating appliances attempt to keep the room at a given temperature by either turning on or off the radiators. Here, however, the explicit rules can be modified for a given outcome, the specific temperature, time, occupancy of residents in the room and so on, to achieve a prior desired outcome. This is driving the system to an explicit goal, while not fundamentally altering the systems dynamics, i.e. not losing the strengths of self-organisation of robustness and self repair (Prokopenko, 2013). This is referred to as guided self-organisation.

Empowerment is an information theoretic approach for agents to maximise their control over their actions within their environment (Klyubin et al., 2005). A simple example to illustrate this idea is an agent operating in a grid world, with the boundaries acting as walls. An empowered agent will opt to place themselves in the centre of the grid world as this will maximise the number of successful actions the agent can perform in the next time step (Salge et al., 2014), i.e. it will not fail an action by bumping into a wall. As the environment becomes more complex the agent will naturally explore and exploit the environment. If a closed door is present the agent will open it as it shall create new opportunities, and thus maximise the control and influence of actions in the environment. There is no explicit goal to be achieved and yet the agent exhibits goal-esque behaviour, for example, opening the door, and is therefore able to be robust to changes in the environment and to the agent itself. An additional signal, or objective, could then be integrated with an empowered agent to guide the agent towards a particular behaviour or goal. Lyons and Herrmann (2020) employs empowerment with reinforcement learning by adding an entropy component to the reward function. This is not identical to empowerment as an approximation is calculated instead, which is referred to as γ-empowerment. Overall, this creates a reinforcement learning agent and while solving a problem if the agent becomes uncertain of what to do, be it due to lack of explicit objective or otherwise, the agent will continue to explore in a self-organised manner due to the empowerment component, with a harmonious balance of exploration and exploitation.

This approach can be applied in robotics, a good example being a controller known

as homeokinesis (Der et al., 1999). Homeokinesis can be seen as the complement of homeostasis, where the agent tries to find stability through activity. The aim of the system is to minimise the prediction error of the signals by adapting its controller and model, thereby generating behaviours. This, on its own, causes continuous exploration of the embodiment of the robot and the environment. This can be modified, however, for the robot to be driven towards particular behaviour generation (Martius et al., 2007; Martius and Herrmann, 2010; Martius, 2010), which can then be built upon further to utilise the strengths of the self-organising properties for a particular goal. In the case of homeokinesis, this can be for reinforcement learning (Smith and Herrmann, 2011).

2.1.3 Criticality

The dynamics of self-organising systems are dictated by the tuning of the parameters, which can lead to critical points in the system. In the field of physics, a critical point is when the system is on the verge of a phase transition (Jensen, 1998), a very simple example being water at 0 ^{o}C, at which temperature it can either be liquid or solid. This is an important concept for self-organisation as being near these critical points can lead to not only interesting but also optimised behaviour.

Another well studied model when considering criticality in systems is the sand pile model (Bak et al., 1988). If we consider a grid setup in which each cell can contain grains of sand, and by adding a grain to a cell the 'pile' of sand increases. Now a threshold can be introduced as to when a sand pile in a cell is too large and thus diffuses to neighbouring cells. At the start as grains of sand are added there will be small avalanches throughout the system. As more sand grains are added and distributed, however, one minor avalanche could cause neighbouring cells to also avalanche, and this in turn could cause a chain reaction of avalanches. From a high level perspective, the sand pile model will experience many small avalanches, and few large avalanches, and thus the magnitude of the avalanches will follow a power law distribution.

Shew and Plenz (2013) discuss through a review of work regarding brain functionality, how the cortex of the brain is critical, allowing it to optimise information storage and transfer. If it was supercritical (full neural activity as an example), the system would be overloaded, and if sub-critical (little to no neural activity as an example), would result in sub-optimal behaviour as the speed of information transfer would be considerably lower.

Similar to the line of thought in Sec. 2.1.2, an engineer of artificial systems has

the flexibility to modify said system to improve it in general or guide it to a specific purpose. The investigation of criticality of the system can therefore be beneficial, as it can lead to increased effectiveness in algorithms through an understanding of the dynamics. An example is the work conducted by Gershenson (2012) where they explored random Boolean networks and different methods of guiding the networks to becoming critical. They also discussed the advantages these systems would have for the disciplines of networks and engineering, including maintaining robustness while also achieving rich dynamics.

2.2 Swarms in nature

A swarm in the animal world is one where the individual of the species either usually requires the assistance of others to succeed in the task, or the inclusion of others will greatly benefit all. Swarms in nature includes bees, birds, fish, and ants (Bonabeau and Dorigo, 1999). The individual interactions are integral to the low level rules of the agents, and it is from here that the self-organised behaviour emerges.

2.2.1 Swarm intelligence demonstrated by ants

An example of this system can be found in social insects such as ants (Hölldobler and Wilson, 1990). While searching for food, an ant will adopt the basic rule of following a pheromone, and once the source is found, pick up a piece and take it back to the nest, following the pheromone trail back. The richer the source, the stronger the pheromone the ant will leave behind, and over time, more ants will be attracted to the trail of pheromones produced from the ant who found the closest, richest source. With more ants on this single trail the pheromone trail will become stronger and thus more ants will join this trail. As can be seen, ants will converge on this trail, and thus the ants will have found the best source and be most focused on this source in a relatively short length of time. Any knowledge of the higher level goal of, not only finding food, but also identifying the quickest route, was not known to the ant, instead all it knew was to follow the pheromone trail if present, and produce its own if it found food. Furthermore if the environment was to change, say, an obstacle fell onto the pheromone trail, the ants will not change their overall behaviour. They will still follow the pheromone trail to the obstacle and from here proceed to random searching, and if enough time passes, the trail overall will fade. This can now lead to the ants searching closer to the

nest for another source of food, which implies an implicit high level decision making scheme unknown to all the individuals. This is a powerful capability as the individual's cognitive abilities are limited and are only able to carry out their simple tasks (in the case of the ants, follow pheromone, and bring back food).

2.2.2 Navigating the world

Swarms comprising of the simplest organisms will have very limited sensing capability and thus will have little flexibility on how to navigate the world. Usually it is the detection of one source, and a simple attractive or repellent behaviour is performed by the organism. An example is chemotaxis, which is following a chemical trail (as seen in ants in Sec. 2.2.1), and is seen to be a robust tactic even for the simplest of organisms, bacteria (Alon et al., 1999). Other examples include thermotaxis, which is to follow a temperature gradient (which is thought to also be utilised by sperm to guide them to the oocyte (Bahat and Eisenbach, 2006)), and phototaxis, which is responding to a light stimulus (cockroaches are known to be negatively phototactic as they try to find places to hide (Roth and Willis, 1952)).

2.2.3 Stigmergy

Agents in natural swarms tend to rely on indirect communication, either through body action (birds flocking), or through changing the environment for others to perceive (ants leaving pheromone trails). The latter is known as stigmergy, which was first introduced by Grassé (1959) (cited from (Theraulaz and Bonabeau, 1999)) to explain how termites construct their nests. With every change performed on the nest, the termite workers perceive their local environment differently, which dictates their actions on how to contribute next. This in a sense means that the organisation of the construction is not determined by the termite workers but instead by the structure of the nest.

Jones et al. (2004) reported that another natural system to do this are honeybees. The hive requires the temperature to be sufficient to survive during the winter season, however the individual bees cannot regulate their own temperature. Thus they require others to cluster and increase the hive's temperature, and if the bees are too warm they cool the environment by flapping their wings. The introduction of bees to a cluster is changing the environment via the temperature change the new bee has caused, and thus informing the bees as what to do next.

2.2.4 Collective mind

Swarms are decentralised systems, and furthermore while there might be roles, there is never an explicit leader. They strongly rely on the presence of others as a form of propagation of information throughout the environment. Buhl et al. (2006) investigated a horde of locusts, and while the number of locusts was small there was disorder. When the number began to increase, however, this disorder transitioned into coherent swarm motion. Further, they were able to report that groups of two to seven locusts were only weakly aligned and this significantly increased with groups only slightly larger, meaning they were able to identify the critical density for which this phase change occurred.

Not only does the inclusion of new individuals possibly lead to organised behaviour, it can radically change a swarm's preference. Couzin et al. (2011) experimented with a group of fish, golden shiners, and trained a few to prefer swimming to one option, while the majority of the fish preferred the other option. The minority, however, dictated the overall behaviour of the school of fish, and would always guide the fish to their preference. When they added new uninformed individuals, however, this led to the majority regaining control and the school of fish being governed by majority rule. This fascinating reset of beliefs is perhaps an interesting option to perturb the system and avoid local minima.

2.3 Swarm intelligence

A swarm is considered to be a decentralised collection of agents/entities which have limited abilities. Each individual will be striving to achieve a goal (usually the same goal(s)), and they will primarily rely on local information and communication with their neighbours. There are examples of swarms in nature, specifically the animal kingdom, which have been a source of inspiration for some well known optimisation algorithms.

2.3.1 Ant Colony Optimisation

A number of swarm intelligence algorithms are explicitly inspired by natural swarms. Chakraborty and Kar (2017) reports that the most popular of animal/insect swarm based algorithms is the Ant Colony Optimisation (ACO) algorithm created by Dorigo and Di Caro (1999). ACO in its basic setup is a connected graph where the ants must

find the shortest path between two points. The ants are dispersed and use local heuristics to determine what path to take, and once all ants have ended their run they lay pheromone trails and the strength is determined by the length of the route; shorter routes result in stronger pheromone trails. On subsequent runs the ants are now also encouraged by pheromone trails, and eventually the ants will converge and an optimal path will have been discovered. From this setup ACO has been expanded, such as the Max-Min Ant System (Stützle and Hoos, 2000), where only the best ant updates its system, and Ant Colony System (Dorigo and Gambardella, 1997) where the ants also lay down pheromone trails, not just at the end of the run, but also at every time step. ACOs have been used for a variety of problems such as how best to organise a two-sided assembly line to keep assembly costs to a minimum (Simaria and Vilarinho, 2009), vehicle routing problems (Rizzoli et al., 2004), and mobile wireless networks multihop planning (Ghosh et al., 2019).

2.3.2 Particle Swarm Optimisation

Particle Swarm Optimisation (PSO) is an optimisation technique inspired by the flocking of swarms. A collection of particles is distributed in a search space and each particle has a personal best value, and the swarm has a global best, which the individuals have knowledge of. The parameters determine the behaviour of the overall swarm, whether they will favour exploring their local best or exploiting the global best. Favouring one strongly over the other can lead to the swarm stagnating, or potentially even diverging if the parameters are being adapted in real time inappropriately.

Metaheuristics can be employed (Clerc and Kennedy, 2002) to help find the correct balance and acquire an optimum value. Furthermore, it can be observed that this system will have a critical state, as the system can either converge too soon, or diverge depending upon the setup of the PSO. Erskine and Herrmann (2015) showed that a PSO will peak in performance near or on this critical point.

2.3.3 Turing Learning

The swarm-based algorithms mentioned up to this point have predominantly been for solving data-related problems, but they can also be developed as a research tool for the understanding of animal collectives. A relatively recent idea is Turing Learning (Li et al., 2016), which aims to develop models that best capture the movement data of natural systems. There are two populations; one population are models of the data, and

the second population are classifiers. The models will generate data samples, and the classifiers must then determine which data is the data generated by the natural system, and which was generated by the models. The models then, based on the success of the classifier, learn how to produce more accurate samples. Meanwhile the classifier learns how to better discern the true samples from the model generated samples. This adversarial aspect is what drives the learning, until the models are generating samples that are nearly indistinguishable from the true samples, thereby creating a model for the natural system. The classifiers, since they have been optimised, can also be used after learning for detecting abnormal behaviour in the swarm.

2.3.4 Cellular automata

Cellular automata was first discussed in the 1950s (Bays, 2010), and is a grid cell system where the cells interact with their immediate neighbours and from their neighbours determine what their value should be. Popularity grew when Gardner (1970) published the rules to John Conway's Game of Life. Each cell was considered as alive or empty and the update rules were simple:

1. If an alive cell had two or three alive neighbours it would survive to the next time step.

2. Every alive cell that has more than four or less than two neighbours will die (from overpopulation and isolation respectively), i.e. the cell becomes empty in the next time step.

3. An empty cell becomes alive if it is neighboured by exactly three alive cells.

These simple rules led to stable patterns forming, such as blocks and beehives, as well as oscillating patterns. The simplest example of an oscillating pattern being a blinker, where three vertically aligned alive cells will become three horizontally aligned alive cells, and then oscillate between the two configurations at every time step. This was studied with much rigour (Wolfram, 1984), and there has been research into the applications cellular automata could have in computer science, such as cryptography (Chaudhuri et al., 1997) and image processing (Popovici and Popovici, 2002). Cellular automata can also be used for simulating systems in biology and physics, as demonstrated by White et al. (2007) where they simulated an epidemic with each cell containing a concentration of infected, susceptible, and recovered individuals. This shows an expectant outward moving ring of heavily infected cells as those inside the

ring have recovered, while those outwith are still highly susceptible. They can also simulate reaction-diffusion systems, please see Chapter 4.

2.3.5 Reynold's Boids

Reynolds (1987) developed an artificial life algorithm, Boids, where each agent has three objectives:

- Separation: Steer away from neighbours
- Alignment: Steer towards the same direction as neighbours
- Cohesion: Steer towards the centre of mass of neighbours

From these three simple rules the swarm is able to flock and move as one in a direction, while also not colliding with each other. Individually, the rules are a component of the velocity, and thus, each has a weighting assigned, which dictates the behaviour of the swarm. Increasing the weighting of cohesion and lowering the alignment, for example, will cause the boids to aggregate more.

Due to the rules' simplicity they can be used efficiently in simulations and animations. Hartman and Benes (2006) expand upon Boids with the inclusion of a leadership rule, that counteracts the alignment force, and gives a guidance to the flocking behaviour. This has potential applications in video games and simulating crowd control, as their work is able to simulate hundreds of boids in real time at 30 frames per second.

2.4 Swarm robotics

Typically a robot is defined as a machine that operates autonomously to perform tasks. With the power of AI increasing, robotics is becoming an attractive option for industry (Steil and Maier, 2017; Zanchettin et al., 2018; Evjemo et al., 2020). The number of global units installed had been steadily rising from 2012 to 2018, and even though there was a drop in 2019 (reflecting the economic uncertainty of the US and China trade disputes), there were still 373,240 units installed, worth around USD 13.2 billion for the units alone (International Federation of Robotics, 2020).

A robot is typically considered as a collection of sensors and motors which uses a controller to integrate the two. A sensor gives the robot information about the environment, and these sensors can include infrared, cameras, light, tactile, microphone,

and so on, as well as proprioceptive sensors such as joint angles and internal temperature. The motors manipulate the robot's positioning in the environment, as they apply torques to the joints of a robot, and can be as simple as a servo motor for wheels or a more complicated actuator consisting of a collection of angular motors that together form arms and grippers.

The controller that connects the sensor information to the motor output does not need to be complicated to achieve interesting and 'smart' behaviours. Braitenberg (1986), with his famous thought experiments, demonstrated this by first using a simple two wheeled robot with two sensors (positioned front left and front right) responding to a stimulus, and showing how one to one connections between the sensors and motors (wheels) can generate intelligent navigation. If the left sensor is connected to the left wheel and right to right, the robot will drive towards the stimulus (positive taxi), and if the connections were left sensor to right wheel and right to left the robot will drive away from the stimulus (negative taxi). When the environment has more stimuli, the robot will display more varied and complex behaviours, and the connections of the sensors to motors also plays a role in behaviour generation.

Swarm robotics is aiming to maintain the principles of swarm intelligence of low capabilities, capturing the robustness and adaptive qualities of self-organisation. This usually means that the platforms, simulated or physical, tend to have limited sensor and actuator ability, though interesting behaviour can still be generated, further enhanced by the robot-to-robot interactions.

2.4.1 Common tasks

There are a variety of tasks that are performed by swarms. They can be clustered usually with a starting common behaviour, and this common behaviour, acts as a starting block for building towards a more complex task.

2.4.1.1 Foraging

Foraging is when there is a source with items, usually referred to as food, which the robots are tasked with returning to a 'nest'. For tasks such as these, communication between robots is vital to minimise energy spent in searching and retrieving the food. Shared memory is a technique where agents in a system have access to the same data, and with respect to this, one can turn to stigmergy and apply the principle here.

Fujisawa et al. (2014) decided to stay as close as possible to the idea of how insects utilise stigmergy, i.e. with an evaporating chemical trail, solving a foraging task by making the robots physically leave a trail of ethanol behind for other robots to sense and follow. They showed this working in both the simulations and the physical robots, though reported diminishing returns as the number of robots in the swarm increased. This has been reported as a common flaw by Krieger et al. (2000) (as cited by Fujisawa et al. (2014)) as the robots might start to interfere with each other.

The robots, of course, do not need to act exactly like the natural systems that they are inspired by and can make more use of technological aspects. Hoff et al. (2013) employ a robot swarm in a foraging task where they use two different algorithms; gradient and sweeper. The gradient algorithm has the robots exploring the environment, with a few becoming stationary to act as beacons, where they use a simple hop algorithm to create a gradient for the swarm to follow. Two gradients are stored on the beacons; one towards the food source and one towards the nest. The sweeper acts by having the robots, almost in line, sweep across the environment, and once they have found the food, some of the robots became stationary beacons to guide robots to the source and back to the nest. The gradient is faster for finding food, thus, exploiting the source if said source is positioned initially near the nest, while the sweeper would perform better for sources far away, though the sweeper tends to take much longer. Thus a third algorithm, labelled adaptive, was developed which combines the two; first try the gradient algorithm and if it fails to initially find food, switch to the sweeper.

The foraging task has been expanded as well to include multiple types of food where the swarm must now decide how best to exploit the sources given the choices available (Balch, 1999).

2.4.1.2 Flocking

Flocking is the act of the swarm moving as one in a similar direction, normally seen in the flight of bird and fish collectives. Reynold's Boids (see Sec. 2.3.5) is a simple algorithm that has the advantage of guiding a swarm in a flocking manner, while ensuring that the individuals do not collide or break away from the swarm.

A classic example of this in swarm robotics is the work done by Hauert et al. (2011), which consisted of simulated and physical winged robots with the Boids algorithm implemented. On top of the three rules in Boids they added a fourth force known as migration (Crowther, 2004)(cited by Hauert et al. (2011)). This keeps the robot contained to an area, as the experiments were conducted outdoors, and prevents

them from flying away. They investigated the effects of the communication range as well as the wing turn speed. As the communication range is lowered it becomes harder for the robots to flock effectively, and likewise when the wing turn speed decreases, as the robots cannot keep up with the flow of the flock.

Another common approach is to have the robot swarm learn the flocking behaviour instead. This eliminates the need to tune the parameters which can, as seen above, be dependent upon the physical robot, and can take into consideration all aspects of the task, not just flocking. Yan et al. (2020) researched the role that deep reinforcement learning can play in training a swarm of unmanned air vehicles (UAVs) to flock and reach a goal target. In simulation they had 10 UAVs tasked with reaching a goal position, and obstacles were present in the environment which the UAVs had to avoid. Reinforcement learning requires a reward function from which the system can learn. Here the authors used three components for the reward function. First, if the goal was reached, or how close the UAV was to the goal when a training episode ended. Second, obstacle avoidance, receiving negative reward the closer it got to obstacles. Finally, a flocking maintenance reward, which was determined by the difference between a UAVs heading from the average swarm heading, and the distance from the swarm's centre of mass. The UAVs only used local information to operate, and then the training algorithm used the collected UAVs reward to best determine how to globally train the controller, which here was a deep neural network.

2.4.1.3 Aggregating

Aggregation is the clustering of the agents/robots in a swarm, which is a very useful technique for a swarm to be able to perform. A swarm can perform what is known as free aggregation, meaning they should cluster regardless of location of the cluster, or a swarm can employ environment-mediated aggregation, where there is some influence from the environment that causes clustering behaviour to occur at a particular spot. Another consideration is swarm size, and if the swarm should have a maximum cluster size which would cause sub-clusters to form instead.

Aggregation can be used with particular goals in mind. Schmickl et al. (2006) studied the idea of a 'collective perception', since a robot in their swarm does not have the sensory capabilities to map the whole environment, instead the whole swarm should contribute to build a better global understanding. The goal in this case was for the robots to cluster at certain goal areas, which varied in size. Once the robots have aggregated, the number of robots at each goal area should be proportional to the

size of the goal area, thus the 'collective perception' of the environment, as the swarm now has an idea of the number and sizes of the goal areas. The successful strategy they employed for the aggregation was a trophallaxis inspired technique (trophallaxis is the process of passing liquid food from mouth to mouth, usually performed by social insects), which had the agents passing virtual nectar to neighbours as a way of communicating if they were on the goal area, or near a goal area, as they might have received some virtual nectar from another robot on the goal area.

Another interesting aspect is the compactness of the cluster, and how it might not always be beneficial to be compact. Trianni et al. (2003) evolved a swarm of robots for an aggregation task, initially with a swarm of five robots. Each robot was controlled by a simple perceptron, connecting their sensors to their motors, and the connections are known as genotypes. Different genotypes were allowed to operate on the robots and at the end of each run the best performing genotypes were kept to produce offspring. The fitness of the genotype is determined by how close they were to the swarm's centre of mass over the last 10 seconds. For producing offspring the winning genotypes will create a copy of their genotype and this new genotype will mutate, i.e. their connections will slightly change. Two stable configurations came from this; a static configuration (a compact collective), and a dynamic clustering behaviour where the swarm was not compact and they still moved together. While the dynamic cluster gets a lower fitness overall than the static clustering, the dynamic clustering will actually allow the swarm to scale. By increasing the size of the swarm, the static clustering will result in sub-clusters, as it appears a greedy approach has emerged, while for the dynamic cluster this loose collective is still moving which will allow it to merge with other dynamic clusters and prove to be superior than the static cluster.

2.4.1.4 Exploration and mapping

Swarms are ideal as a means of efficiently exploring an environment. Given the usually large number of agents, they can each individually explore a different part and this can then be used to update a global representation of the map. A traditional mapping problem is simultaneous localisation and mapping (SLAM) which is the task of not only creating a map of one's environment, but also understanding one's positioning. A swarm can be used as a platform to begin generating this map as a combined effort, as suggested by Zou and Tan (2012).

There are nature-inspired techniques as well, stigmergy being quite popular as it is a simple approach to either attract or repel other robots. A particularly interesting

work in this field is the proposal by Duncan (2019) how to perhaps integrate swarm techniques into real world problems. They use a simple pheromone scheme, where each individual robot lays down a virtual pheromone trail of where they have been, and this then acts as a repellent which causes robots to move away. This in turn means that robots should not explore the same area twice, and thus rapidly increases the speed of exploration. The pheromone trails can also diffuse and evaporate over time allowing re-exploration if required (say exploring an environment that is prone to change). The work goes on to describe how the general scheme can operate in the real world, by utilising cloud services and GPS data to create a virtual pheromone map that all robots can refer to.

Another strategy is to combine small and large exploration steps as shown by Vargas et al. (2020). A swarm of drones performs a general exploration task, with the additional goal of having the drones' fields of view of the ground's surface overlap, i.e. continuous coverage. The continuous coverage is achieved by implementing the Boids algorithm onto the swarm, to achieve a tight flocking behaviour. The other component for the velocity is the usage of a Lévy Walk, which, simply put, generates short random steps with few significantly longer steps. This results in the drone swarm performing local flocking searches, with large steps occasionally performed by the drone swarm to locally search in a different part of the environment.

2.4.1.5 Object clustering

Swarms can be tasked with collecting a group of similar items, known as object clustering. This common task has practical applications for the real world, such as rubble collection at a disaster or construction, or even tidying rubbish in the common household. What is particularly welcoming is that there are some classic examples of how a robot swarm can complete a task of this nature using very simple techniques, allowing them to be further built upon for more complicated object clustering goals.

Beckers et al. (2000) tested the idea of stigmergy as a way to indirectly communicate between the agents as to where the agents should form clusters. In this case the stigmergic aspect was the clusters themselves, which would cause the robots to change behaviour depending on what they sensed. Each robot had a C-shaped shovel attached to its front which was used to push the objects around, and the shovel also acted as a sensor to inform the robot if they were carrying one or more objects. The robots would move around the environment randomly, avoiding collisions, and if the shovel sensor detected an object, i.e. the robot is now carrying an object, it would still continue with

its normal behaviour. If, however, the robot detected more than one object, it would 'drop' the objects and move away. Essentially, if a robot is carrying an object it will drop the object when it has encountered a cluster of objects. There is then the random chance that a sub-cluster is pushed by accident by a robot as it performs its random motion, however, it will quickly detect what it is doing and 'drop' the cluster. This keeps the sub-clusters 'active' and will have a chance to merge with other sub-clusters. This simple approach proved to be effective, though in this particular paper suffers from a similar issue of overcrowding as the swarm size increases.

A seminal work within the realm of object clustering by Deneubourg et al. (1991) describes how a robot swarm can perform object clustering with probabilistic behaviour. The robots will move around the environment randomly and if they encounter an obstacle they will have a probability of either picking it up or leaving it there. Similarly robots with an object are also moving around randomly and they too will have a probability, but, in this case, whether they should leave the object they are carrying. The probabilities are dependent upon the local information and short-term memories of what they have perceived. Specifically, the memory stores the number of objects the robot has encountered over a short past. If a robot encounters an object, then the probability of picking up the object decreases for every object it has seen recently, which is implying that there is a cluster of these objects already. Similarly for dropping an object, the more objects a robot has seen the higher the chance the robot will drop the object, i.e. dropping the object at the cluster. The authors of this work actually consider multiple types of objects, allowing clusters of similar objects by allowing each robot to have a different probability for the different types of objects encountered recently.

2.4.1.6 Assembly

Assembly is a construction task which solves a particular problem, therefore, in a sense, it is object clustering with a driven purpose. This can be used for search and rescue after natural disasters, as the environment might have changed requiring a new passage to be built. An example is the work by Magnenat et al. (2012) where the robot is tasked with gathering resources but is surrounded by a ditch. On the robot's side of the ditch it has blocks which it can manipulate with its scoop, and it has the ability to map the environment. In this instance, the robot first observes the environment, and seeing that there is no way across the ditch to the resources, must pick up the bricks to fill the ditch with.

The authors state that expanding this with more robots can allow a swarm to build

bridges in a search and rescue environment, which would be appropriate as it does not matter where the bridges need to be, just so long as there is one safe bridge/route for survivors to be able to use. Additional guidance is required if the goal is to build a more specific construction, or a general construction in a more specific location. The use of environmental cues can help with this, such as using light, as shown by Stewart and Russell (2006), to guide where walls should be constructed. In this particular work one of the robots was assigned the role of organiser, and would aim a beam of light, dictated by a prior set of rules, which would guide the other robots during the wall assembling process.

Assembly, however, does not just mean object assembly, but can also refer to the swarm as well in what is known as self assembly. This involves the swarm constructing shapes with their physical selves. An example of this is the work by Rubenstein et al. (2014) where they designated predefined shapes for the robot swarm to make, such as a spanner or a star.

Perhaps more interesting yet is the assembly of the swarm through the act of attaching and detaching. A well known swarm project is the Swarm-Bots project (Groß et al., 2006), which focuses on a swarm of robots known as *s*-bots which have a gripper, allowing the individuals to attach themselves to other individuals. It has been shown that this platform can carry out a multitude of tasks including crossing over holes (Trianni et al., 2006) and transporting goods as a collective (Gross and Dorigo, 2009).

2.4.2 Swarm applications

Swarm robotics is highly attractive to real world applications, given its robustness and flexibility to changes in the environment. NASA, for example, has been interested in swarm intelligence as a way to better explore space, given the little prior knowledge of what could occur to the space craft, as well as the terrain on, say, Mars (Vassev et al., 2012). In particular, NASA invested in a project called Autonomous NanoTechnology Swarms (ANTS), with the aim of incorporating swarm intelligence with asteroid-exploring nanotechnology (Curtis et al., 2000). Nanotechnology would benefit greatly from swarm techniques as the physical size limits its sensing and processing capability.

The GUARDIANS (Group of Unmanned Assistant Robots Deployed In Aggregative Navigation by Scent) project (Penders et al., 2011) was a funded swarm robotics project tasked with assisting firefighters in warehouse fires. A problem with warehouse fires is that the smoke created greatly inhibits the vision of the firefighters, and

the blueprints for warehouses will only show the basic structure, i.e. it will not have the contents of the warehouse on the map, which might include shelves, furniture, and clutter in general. The best approach that firefighters use is to follow the wall and keep a close formation with their fellow firefighters. The GUARDIANS project's aim was to assist with the guiding of the firefighters thus increasing the efficiency with which the warehouse was explored. The robots would operate alongside the firefighters, encircling them and at times acting as beacons to guide the other robots. They would communicate using lights attached to the inside of the firefighters' masks, informing them in which direction to go. The firefighters, however, were sceptical of following the robots when the robots were indicating that they should leave the wall, and so the swarm gave preference to this behaviour. This perhaps indicates that more trust is needed, which could potentially be achieved through the relatively new research field Human Swarm Interaction (HSI) by allowing human operators to have more control over the swarm's behaviour (Ferrer, 2018; Crandall et al., 2017).

Even though the project ended in 2011, there is still interest in swarm robotics to be used in firefighting situations. Vardanega et al. (2020) investigated user case studies of the desire to have swarm robotics in firefighting, storage organisation, and bridge inspection. Overall the consensus was that swarm robotics would be welcome. The demands of each area, however, resulted in a different preference for which tasks should and should not be automated, taking into account cost and safety. The survey concludes with the statement that swarm robotics is a developing field that has yet to be properly exposed to end users, and thus there should be stronger ties between the swarm community and appropriate fields of industry to understand how the swarm approach can be implemented for a desired real world goal.

2.5 Conclusion

Swarm intelligence is the principle of a collection of limited agents that employ simple rules giving rise to a seemingly swarm ordered behaviour, which is a phenomena known as self-organisation. From here the natural systems that exhibit this behaviour can be studied, and can then be replicated for AI and robotic purposes.

This Chapter has gone over some of the fundamental considerations in swarm behaviour, and has covered some of the most common tasks that swarm robotics is able to achieve. It is interesting to note how the works shown in Sec. 2.4.1 have overlap with each other, for example, the work for object clustering overlaps with the work in

object assembly, and the work in flocking can overlap with exploration. This highlights the strength that swarm techniques have with regards to flexibility, and being able to act as components in larger systems. Furthermore, the ability to slightly change the setup to produce a different behaviour is extraordinary in swarms, especially in robotic swarms. Robots are designed with a purpose in mind, and swarm research is demonstrating that several swarm techniques can be integrated together to fulfil specific needs and tasks. Lastly, which of these techniques should be employed and how they should be integrated for real world applications requires the swarm community to form links with end users.

Chapter 3

Information dynamics in evolving agents

As discussed in Chapter 2 swarms are individual agents able to interact locally to overcome individual limitations, and accomplish complex tasks as a collective. In this Chapter we present an investigation of a swarm performing the task of gathering energy from a number of equivalent sources distributed on a 2-dimensional space. Counter intuitively as the number of sources increases we observe that the swarm quickly becomes less efficient, despite more space being covered by reward. Our analysis reveals that this confusion arises because they are detecting multiple sources at once, which comprises a challenge for the evolutionary scheme. Additionally, we also show that negative side effects of the communication among the agents tend to reduce their efficiency by offering too many options. This investigation highlights the problems that emerge from, what appears as, an overabundance of explicit information that causes a natural confusion within the system to emerge.

3.1 Introduction

Artificial life is a research field dedicated to creating simple systems and agents as a test bed for understanding how simple organisms are able to learn and operate, and ultimately grow into more complex systems. Swarms are an interesting system to observe, as while each agent in the swarm may be limited in capabilities, the local rules that govern them result in a self-organisation producing an implicit high level intelligence, and various collective behaviours (Bonabeau and Dorigo, 1999). The strengths of robustness and adaptability to perturbations to the environment ultimately stem from the

low level rules, as well as the local communication. These rules can be tamed further through an evolutionary method, by creating new agents from the highest performing agents, to progressively drive the swarm towards optimal behaviour. Typically, information processed by an agent regarding the environment and its neighbours is limited. Attempting to incorporate more information can lead to confusion that low level rules alone struggle to disseminate, especially if competing with other agents through an evolutionary selection. Birattari et al. (2016) have demonstrated the problems that can arise from over-designing a swarm's controller in a simulator for practical purposes, however there can already be an underlying problem with regards to the agents' capabilities for the task presented.

Specific information is a key component enabling systems to learn. Typically in physical agents information can be acquired through sensors, which is then processed to decide the best course of action. Learning how to process this information can be a difficult task as the system might not know what the search space and environment is with respect to the, perhaps, limited knowledge of the reward scheme or fitness function. Understanding this general learning cycle is key for autonomous robotic applications.

An interesting concept is information overload (Martínez-García et al., 2013; Pitonakova et al., 2016), and whether the system's cognitive capabilities can still disseminate something of value. From here the question could then be expanded to the physical aspects, and if given the choice would the system change its physical self to minimise information. We investigated the effects of additional potential rewards on the swarming agents' behaviour to explore the consequences of information overload through a combination of equally viable choices, unlimited sensing (with respect to sources) and signalling within a heterogeneous swarm. It was demonstrated that the swarm struggles cognitively to optimise its behaviour in the presence of multiple equally viable sources. It is only with the inclusion of manipulating the physical abilities that the swarm is able to overcome the confusion and frustration of multiple choices. This Chapter will also discuss the implications for robotic applications, as mutating the physical aspects of a robot might not be the most realistic solution.

3.2 Background

The design and control of swarms has often been inspired by biological systems. It is an interesting option to use the interaction among the individual agents in order to

gain information as the swarm dynamics compensates the limitations of the individuals. Specifically it is the population of the swarm and how the information propagates through the agents that makes swarm intelligence appealing to study.

Through understanding this, it may also be possible to reduce the effects of obstruction and interference amongst the agents, thus improving the efficiency of the swarm. Therefore, rules governing local behaviour must also include how to interact and function, given not only their own information but the explicit and implicit information provided by neighbours. This has been investigated in different fields such as swarm chemistry (Sayama, 2009), particle swarm optimisation (Garcia-Gonzalo and Fernandez-Martinez, 2012), and robotic swarms (Oh et al., 2017).

It is difficult to understand precisely what the local rules should be to produce the complex behaviour of the swarm. A good example highlighting the issues surrounding this was the work done by Bratton and Kennedy (2007) as they studied to define a standard for particle swarm optimisation (PSO). This work explored the different variations of PSO implemented, ending with the statement that even their 'standard definition' is not the best version given the wide variety of problems. Not only this, an increase in swarm size can lead to error cascades with explicit information being communicated within the swarm (Gauci et al., 2017) so defining how and what agents communicate is vital. Additionally the communication aspect entails that agents have to acquire the ability to predict what the other agents are going to do to better formulate their understanding of the environment and state (Herrmann, 2001).

Biological systems adapt to the environment and perturbations to the system via evolution, which is a searching mechanism where each iteration approaches an optimum. This same strategy can be replicated in artificial systems and can lead to two outcomes; an optimised system and/or a better understanding of the task and environment (Adami, 1998). In particular, the evolution of signalling has been one of great interest as this gives information regarding the environment (Torney et al., 2011).

The advantage of an evolutionary strategy is that the underlying complexities of the task need not necessarily be known, as the strategy itself will find the gradient with respect to optimisation. While there has been work on detecting convergence in evolutionary algorithms, such as Trautmann et al. (2009), it is unclear if the strategy has reached a local optimum. In artificial systems, certain techniques can be employed to help avoid this, such as Liu et al. (2017), where they select solutions and determine them individually based on convergence and distance to competing solutions. This maintains diversity in terms of solutions, the real strength of evolutionary algorithms,

and this promotes the system to continue exploring. This however is a luxury that natural systems do not have, which tend to converge on a single solution, and these do not necessarily help us to replicate the full benefits of evolution for artificial life.

There has been much research into the role of evolution within swarms, both biologically and artificially. However, there is usually a specific environment and agent structure in mind, which is assuming that the strategy has produced not only the best outcome with respect to the swarm's output, but also with the swarm's ability to process incoming information.

3.3 Experimental Setup

The agents are tasked with gathering energy from localised sources. This work expands on Witkowski and Ikegami (2016) who have extensively analysed the swarm's behaviour and evolution for a single source setup. The following describes the setup for these simulations.

3.3.1 Environment

The environment is a 2D bounded circular arena, which means agents are not allowed to leave the boundaries of this environment. Within the arena there are sources which have fixed positions and a fixed radius. Their centres are evenly separated on a circle inside the environment, thus all sources are equidistant from their two immediate neighbouring sources. The sources do not move and have unlimited energy. The number of agents on a source at any given time does not affect the energy gained by the individual agent.

If an agent is inside a source it gains energy q (q_t means energy gain at time t), proportional to their distance to the centre of the source

$$q_t = \omega \left(1 - \frac{||x_t - s_c||}{s_r} \right) \tag{3.1}$$

where x_t is the agent's position, x, at time t, s_c is the source's centre, s_r is the radius of the source, and ω is a scaling factor. If the agent is not inside a source then $q_t = 0$. This is then combined with a loss of energy

$$\Delta Q = q_t - \alpha_c v_{m,t} - d \tag{3.2}$$

where Q is the agent's current energy, v_m is the velocity magnitude, α_c is a scaling constant, and d is a constant to represent a continual cost of sustenance. The term

$\alpha_c v_{m,t}$ defines the cost of moving; the faster an agent the more it costs in energy. The cost of moving is included to investigate whether a balance is created within the system between cost of reaching a source and the reward itself. The cost of sustenance is to help encourage activity within the swarm and prevent stagnation.

3.3.2 Agent

The agents have a position vector x and heading θ, and update their position using a velocity vector v

$$x_{t+1} = x_t + v_t \tag{3.3}$$

The velocity, for the sake of this work, is a unit vector, $\hat{\mathbf{i}}$, scaled by a magnitude and rotated by the agent's heading, θ

$$v_t = R(\theta_t)\hat{\mathbf{i}}v_{m,t} \tag{3.4}$$

where v_m is the velocity magnitude t denotes (discrete) time, and $R(\theta)$ is a rotational matrix about angle θ.

The agents can communicate with each other, where the strength of the transmitted signals scales proportionally to the distance between the two agents. The signals are a value in the range (-1,1) generated from the controller, which is an Elman Network (see Sec. 3.3.3 for details). Signals are perceived categorically dependent on the angle for the sending agent relative to the current heading of the receiving agent. In the present study, generally we use a categorisation into six groups of signals, as this was shown to be suitable to capture the neighbours' information. This is demonstrated in the study by Witkowski and Ikegami (2016) which this work is expanding on.

The agents also have the ability to sense distance to sources. The agents detect if they are first close enough to the edge of a source to trigger the distance sensors, then the closest distance sensor is triggered dependent upon the Euclidean distance to the source's edge

$$s = 1 - \frac{||x - s_{e,i}||}{\ell} \tag{3.5}$$

where $\mathbf{s}_i$ is the position of source i, ℓ is the dynamic sensing range. If the agent is inside the source then $s = 1$ and the specific sensor triggered will be the closest one to the centre of the source, i.e. the direction (or rather the low resolution direction provided by the sensors) is still given to the agent but not how close it is, as the distance detected will be as though the agent was outside the source and touching the edge of the source. The distance sensors are also discretised in a similar manner to the communication.

To keep the agent simple, the distance sensing will be discretised into four sections; front, left, right, and behind (the distance sensing sectors will also be relative to the agent's heading). The number of sources present in the environment at any time will not exceed eight, thus four sectors for the distance sensing is a suitable number to keep the agent simple and give implicit directional information. If two or more sources are within the sensing range, then only the closest will be sensed. This is to make the problem easier for the agent and to not cause confusion with detecting multiple sources in the same sector resulting in a potentially misleading average distance. This will encourage a greedy approach, which is suitable here as the sources are all equally as valid. Fig. 3.1 shows a graphical representation of the communication and sensing of the agent. Both the communication and sensing discritsed sections are relative to the agent's heading.

The agents' sensing range, ℓ, for the following simulations will be at least twice the size of the environment's radius, i.e. the agent will be able to detect all sources at any given position in the environment.

3.3.3 Controller

The agents employ a neural network, specifically an Elman Network, to decide on their action, i.e. to determine the velocity magnitude, heading and the strength of their outgoing signal. There are 11 inputs, c, to the network:

- $c_1 \dots c_6$ for incoming signals, which have been grouped based on the angle of the sender and receiver relative to receiver's heading
- c_7 for the energy change experienced at that time step
- $c_8 \dots c_{11}$ for distance sensing and information from the sources (i.e. can see in front, left, right, and behind)

The values $c_1 \dots c_6$ are the summation of signals, which individually are in the range (-1,1).

There is one fully connected hidden layer, and this layer is also connected to a context layer that feeds directly back into this hidden layer.

Lastly there are 3 output neurons, o:

- o_1 for angular acceleration, used to update the heading

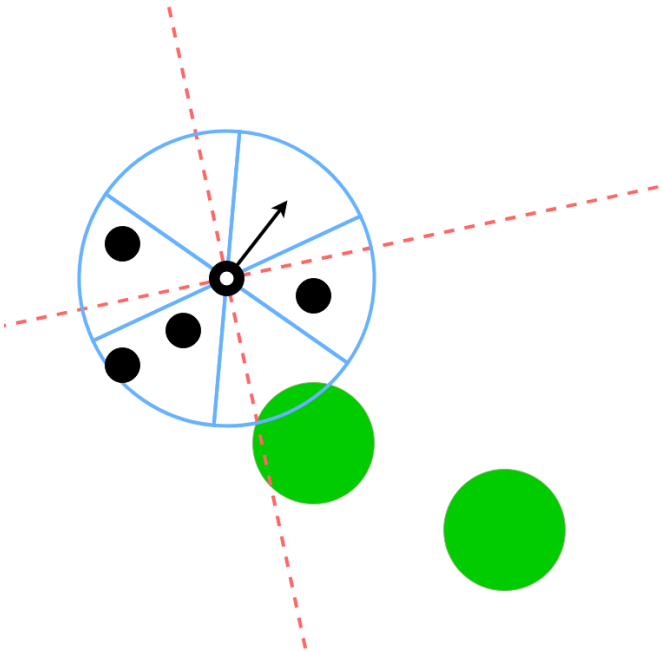


Figure 3.1: An agent (black ring with arrow indicating heading direction) can sense the presence of other agents (filled black circles), if they enter the sensor range (blue circle). For each of the six sectors of the circle (blue lines), the agent can detect only one other agent, and an other agent can trigger only one of the six sectorial sensors. Passive distance sensors indicating food sources (green disks) with respect to main directions (front, back, right and left; as indicated by the red dashed lines). Again, If multiple sources are in the same quadrant, then the agent senses only the closest one, see Sect. 3.3.3 for more detail. The discretised sectors for both the communication and source detection are relative to the agent's heading.

- o_2 for velocity magnitude acceleration
- o_3 for signalling strength

The agents update their heading as

$$\Delta\theta = \alpha_\theta o_1 \tag{3.6}$$

and the agents' velocity magnitude is updated as

$$\Delta v_m = \alpha_v o_2 \tag{3.7}$$

where α_θ and α_v are learning rates. Furthermore $0 \leq v_m \leq v_{max}$, where v_{max} is a pre-defined maximum velocity. If $v_m > v_{max}$ then we set $v_m = v_{max}$. The values used for the parameters presented in Eq. 3.6 and Eq. 3.7 are listed in Table 3.1. The output o_3 is the signal the agent transmits. The evolutionary strategy (see Sec. 3.3.4) will evolve the swarm to develop the ability to communicate and have an understanding of the incoming signals. The concept of collective agents and robots evolving to develop a language has been demonstrated to work for solving shared tasks (Marocco and Nolfi, 2006).

The controller's activation function is a fast sigmoid

$$\sigma(y) = \frac{y}{1+|y|} \tag{3.8}$$

The weights are randomly initialised and are never updated during generational runs.

3.3.4 Evolutionary Strategy

The swarm uses an evolutionary strategy to learn the task, specifically they follow the fitness proportionate scheme. Each generation consists of a number of time steps for when the swarm operates, and once the pre-designated number of time steps is reached the energy gathered is compared to the average energy gathered by the swarm. If the agent has more than the average then they are reserved for the next generation. If they have less than the average, they have a probability of surviving

$$\begin{aligned} p_i &= \frac{Q_i}{\hat{Q}} \\ \hat{Q} &= \frac{\sum_i^N Q_i}{N} \end{aligned} \tag{3.9}$$

where p_i is the probability that agent i is kept for the next generation, $\hat{Q}$ is the average energy gathered by the swarm, and N is number of agents in the swarm. The probability is essentially a fraction of the energy compared to the average energy; higher performing agents having a greater chance of surviving and being carried over to the next generation. The fitness proportionate scheme was chosen here due to the nature of the swarm's communication. The swarm is heterogeneous, as their weight space can greatly vary, and thus the individuals may transmit and interpret signals differently. A harmonious swarm can then emerge due to this, promoting potential exploration, and an overall high performing swarm. If the evolution was, say, always selecting the top X performing agents then this harmonious behaviour will be damaged after each generation due to the shared signalling being greatly replaced each time.

The swarm is then replenished by producing offspring using a roulette scheme. The average energy gathered is recalculated using the surviving swarm, and the probability of that agent producing an offspring is calculated the same as Eq. 3.9. This is repeated until the swarm is at initial size. The offspring produced are identical to the parent with mutations occurring to the weight space of the controller. Every weight in the offspring has a probability to mutate, p_m, and if it does mutate the weight update will be

$$\Delta w_i = U(-b, b) \tag{3.10}$$

where w_i is weight i of the network, U is a uniform distribution function, and the size of the mutations is governed by a parameter that we will set to $b = 1.3$ in the simulations below.

3.3.5 General scenario setup

We tested the swarm with different setups to observe how the swarm's behaviour changes given the increase in the number of sources. The eight different setups are displayed in Fig. 3.2, and the positions of the sources do not change from generation to generation. The sources are also static as mentioned in Sec. 3.3.1. At the start of each generation the agents' position, x, heading, θ, and velocity magnitude, v_m are all randomly generated.

The sources are all identical with regards to the energy output, size, and distance to their immediate neighbouring sources. This was intentional as we wanted to observe how the swarm will behave if the multiple choices it is given are nearly identical. The only difference between the choices are the distance between the agent and the

sources, and intuitively we should expect the agents to learn to gravitate towards the closest source relative to them.

Even though the setup is the same from generation to generation while learning, we do not suspect an over-fitting to occur. This is because the agents have randomised initialisation at the beginning of each generation, as stated above, and all information provided to the agents are relative to the agents' position and heading.

The weights, as previously stated, are randomly initialised at the start of each setup. The initial weights are between -0.4 and 0.4 with uniform probability. With the initial weights quite small, it was thought that $b = 1.3$ could promote significant early exploration.

For each scenario, we ran the simulation five times to allow us to acquire average timecourses. We acknowledge that we should have acquired more, however there were issues with computational limits.

3.3.6 Simulations conducted

We first observed the swarm's success in an environment with one source, and then compared this to an environment with two sources. We then observed the effects of signalling by performing the same setups but with signalling switched off. From here we then increased the number of sources (3..8), and then looked again at the setups without signalling.

The effects of signalling were then investigated further. First, the signalling was turned off from the beginning and compared with the previously attained results. Then we conducted simulations where the swarm started with the ability to signal for a set number of generations, and then signalling was switched off for the same number of generations. We then performed this setup in reverse, i.e. starting with signalling switched off and then retraining after signalling was switched on.

We also observed the swarms in different environments after training. For example, after training a swarm in an environment with one source how will it perform in an environment with multiple sources.

Active Information Storage (AIS) was calculated as well, in order to discern if there is a relationship between the number of sources the swarm were exposed to and the information an individual agent contained.

Finally, the swarm was given the ability to also evolve their distance sensing. This was achieved by allowing the offspring to also mutate the range of the distance sensors,

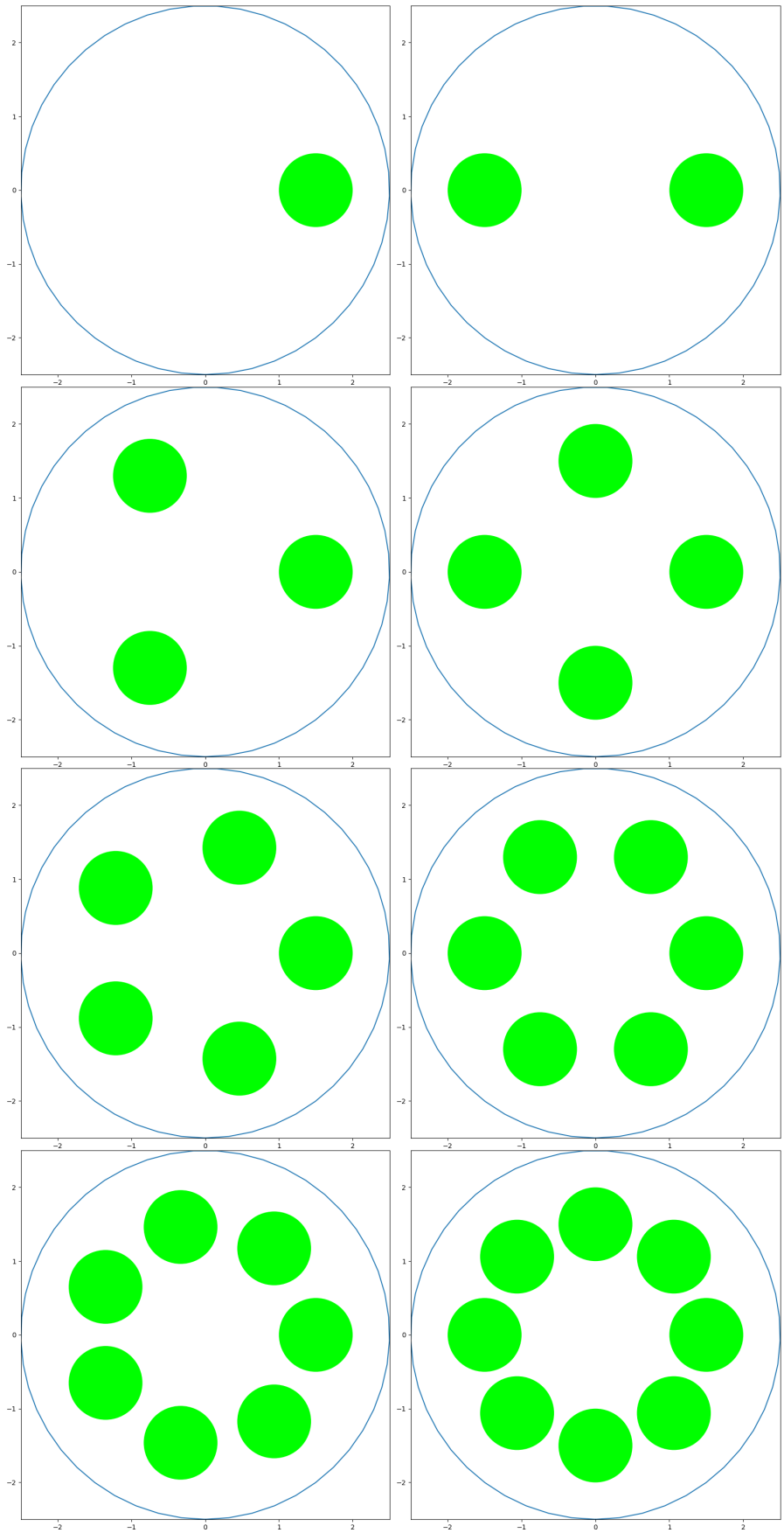


Figure 3.2: The environment setup with one through to eight sources. The sources are the green disks, and the blue circle is the environment's boundary.

ℓ. Up to this point we had been merely observing the cognitive evolution to adapt to the environment and information being received, i.e. the controller. Thus it is appropriate to also consider the physical aspects of the agent and whether there is a preference on how to acquire information, not just how to process information.

3.3.7 Parameter choices

Table 3.1 contains the parameter choices. These parameter choices were used for the majority of the simulations. The main exception is the arena size and ℓ for the simulations of varying the arena size.

As the swarm is only evolving the weights of the controller it is essentially only learning how to process the incoming information. Therefore the learning rates for updating the velocity magnitude and angular velocity are kept low, $\alpha_v, \alpha_\theta = 0.1$, to grant time to the agents to formalise a decision.

The agents also have an initial energy value of 2.0. We wanted to observe if the agents would prefer to retain their starting energy rather than attempt to travel to sources.

3.4 Results

Presented are the results of the simulations conducted as described in Sec. 3.3.6. Overall the swarm has difficulty when more than one equally viable choice is given, and the signalling provides confusion to the swarm. The swarm is able to overcome these issues when the distance sensing range, ℓ, is mutated in the offspring too, as this limits the information input the controller has to process. This is further evidenced by the simulations of pre-trained swarms operating in different environments, where the swarms exposed to multiple sources, but not having their distance sensors constantly triggered, perform the best in other environments.

3.4.1 One source vs. two sources

Let us first consider increasing the number of sources from one to two. Running simulations demonstrates that in both cases the swarm does improve, and is consistently increasing the energy gathered within each generation.

There is, however, a significant drop in energy gathered when the number of sources increases from one to two. This is not intuitive as by adding a source the

Parameter	Value
Swarm size	100
Arena radius	2.5
Source radius	0.5
Source centre distance from centre	1.5
Number of hidden neurons	10
Local communication range	0.5
Maximum velocity magnitude: v_{max}	0.05
Distance sensing range: ℓ	10.0
Angular acceleration step-size: α_θ	0.1
Velocity magnitude step-size: α_v	0.1
Starting agent energy: Q_0	2.0
Velocity energy loss scaling factor: α_c	0.05
Energy decay constant: d	0.05
Source energy gain constant: ω	0.3
Timesteps in generation	4000

Table 3.1: List of parameters predominately used for simulations, unless stated otherwise.

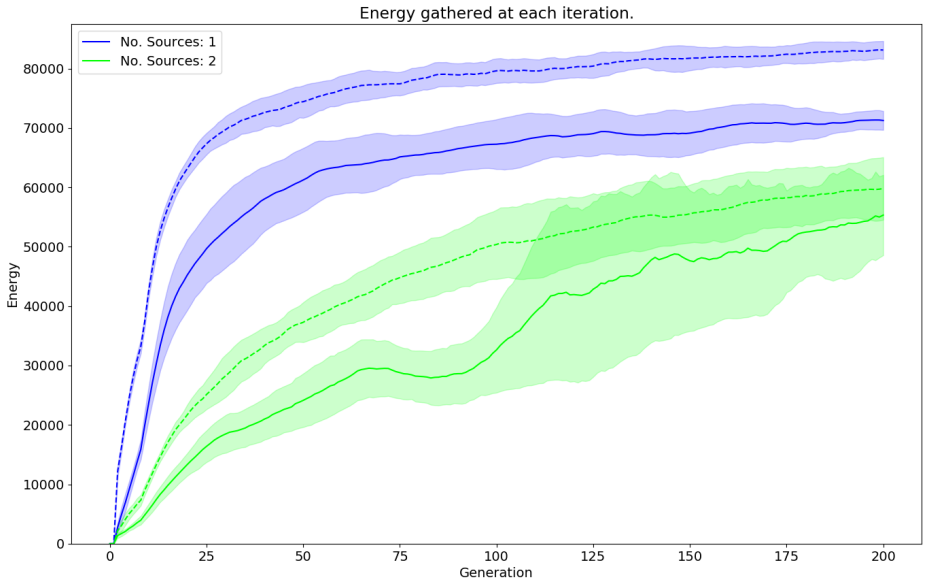


Figure 3.3: A swarm of agents operating in an environment using the parameters stated in Table 3.1. They were simulated for 200 generations per simulation. There were four different settings; one source, two sources, and the same as the previous two but without signalling, i.e. communication range was set to 0. The solid plots are for the simulations where signalling was turned on, and vice versa for dashed. Each setup was conducted five times to acquire an average timecourse. For each simulation a moving average was applied, and afterwards the energy timecourses were averaged.

percentage of area where the agents will be rewarded is doubled. This also means that upon initialisation an agent will be closer to a source when there are two instead of one, meaning less travel and therefore less energy consumption. Simulations were also conducted with signalling switched off, i.e. the communication range was set to 0, and this resulted in a higher gain in energy. See Fig. 3.3 for all presented simulations.

Fig. 3.4 displays trajectories of 10 selected agents that were chosen at random with uniform probability. The trajectories are from the final generation of sample simulations, which here is generation 200. The agents are clearly successful in reaching the centre of the source when there is only one present, however, there is a switching between sources, indicating strong confusion.

A possibility for this behaviour is that when the swarm has reached the centre of a source it still detects the other source, which starts to signal the agent to travel to it. The momentum built from reaching the first source could also contribute to the agent

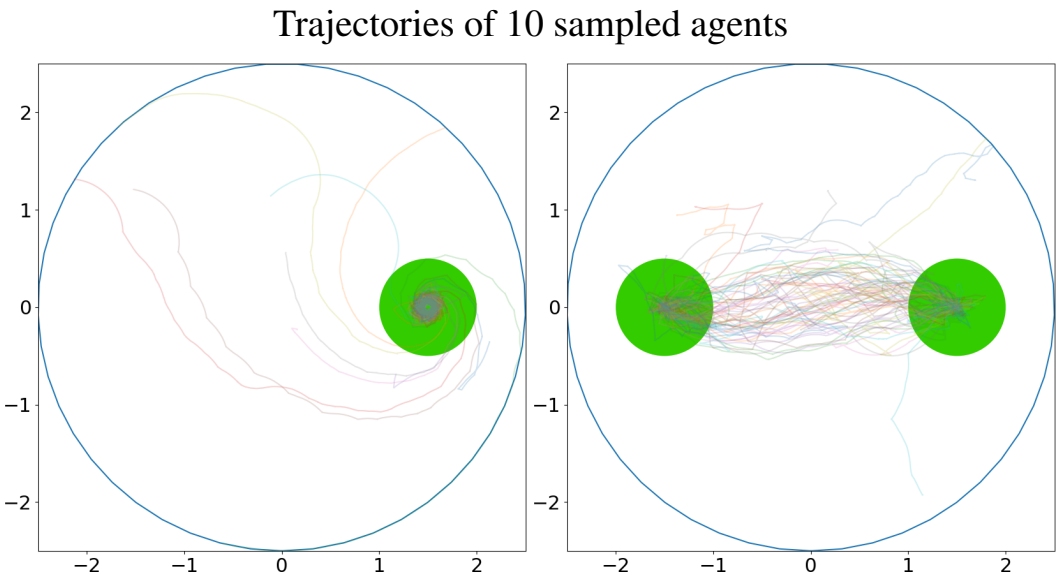


Figure 3.4: In each simulation the swarm was trained with a fixed number of sources. The above are samples to demonstrate the trajectories of 10 randomly chosen agents with varying number of sources. The agents develop a preference to cycle between the sources as more are introduced. With two, however, a cyclic behaviour emerges as the agents are not certain of which source to remain stationary at.

continuing to follow the signal of the other source. At this point the energy change is decreasing (input c_7), however, the network does not seem to consider this as a negative aspect, i.e. the drop in energy change does not prompt the network to inhibit the desire to move away from the source. As long as $c_7 > 0$ then the agent considers this to be satisfactory. Though it should be noted that the agents do exhibit deceleration in the environment with one source, meaning that the agents do identify the centre being the optimal position, which can be seen in the two source setup as well. This however does not explain everything, as the agents will still be detecting the centre of the source they are currently in, which will also result in a stronger signal given the close proximity. Lastly, it is demonstrated that the swarm performs better without signalling, which indicates that the agents are confusing each other with their signals, and cognitively the agents are not able to learn to overcome this. In combination the agents might be experiencing the attractive force of the other source, and the potential repulsive force of neighbours.

Simulations were also conducted varying the arena size, one set where the arena's radius was twice as big, and another where the radius was quadrupled. The position of the sources remained the same, i.e. 1.5 length units from the centre of the environment.

These simulations were tested with and without signalling, see Fig. 3.5. The results presented agree with the intuition that with smaller arenas the swarm will perform better as there is less travel time from random initial positions to sources.

3.4.2 More than two sources

We then investigated the swarm's behaviour by increasing the number of sources to eight, see Fig 3.6. Similar behaviour is observed here as before, except there also appears to be a balance of the total area that is a reward and the number of sources with respect to the swarm's final energy level.

Fig. 3.7 displays sample trajectories on increasing the source number, and we again observe a cyclic behaviour in which the agents are confused as to whether to go or stay. From these trajectories it is observed that the swarm can learn different cyclic strategies. One strategy is to continue on a path never decelerating, and continuously cycle between the sources. Another strategy is that the agents remain in a source for a short time, before the detection of the other sources slowly start to pull at the agents, which in this case will pull the agents towards the centre of the environment. The agent will not travel far before moving to a neighbouring source. This style of behaviour is known as pulsing in swarms (Edwards, 2000).

3.4.3 Effects of retraining signalling

Simulations were performed to observe the effects of signalling, which was achieved by initially training a swarm with signalling switched on or off, and then vice versa for the same number of generations, see Fig. 3.9. The evolutionary strategy, see Sec. 3.3.4, is used throughout both phases.

Firstly, if the swarm is initially trained without signalling then the instance it is switched on the swarm's performance immediately drops. After another 200 generations of learning the swarm is able to improve again, however not to the same level as before. On the other hand, if the swarm's signalling is initially switched on and then switched off, as before, there is a drop in performance. After this, however, there is a significant improvement after the relearning phase.

The results demonstrates that if trained with signalling the agents attempt to utilise it, as when the signalling was switched off the swarm gathered less energy. However, it is also evidenced that the signalling is hindering the agents, and is acting as noise that the agents are attempting to understand. In both instances the swarm is more

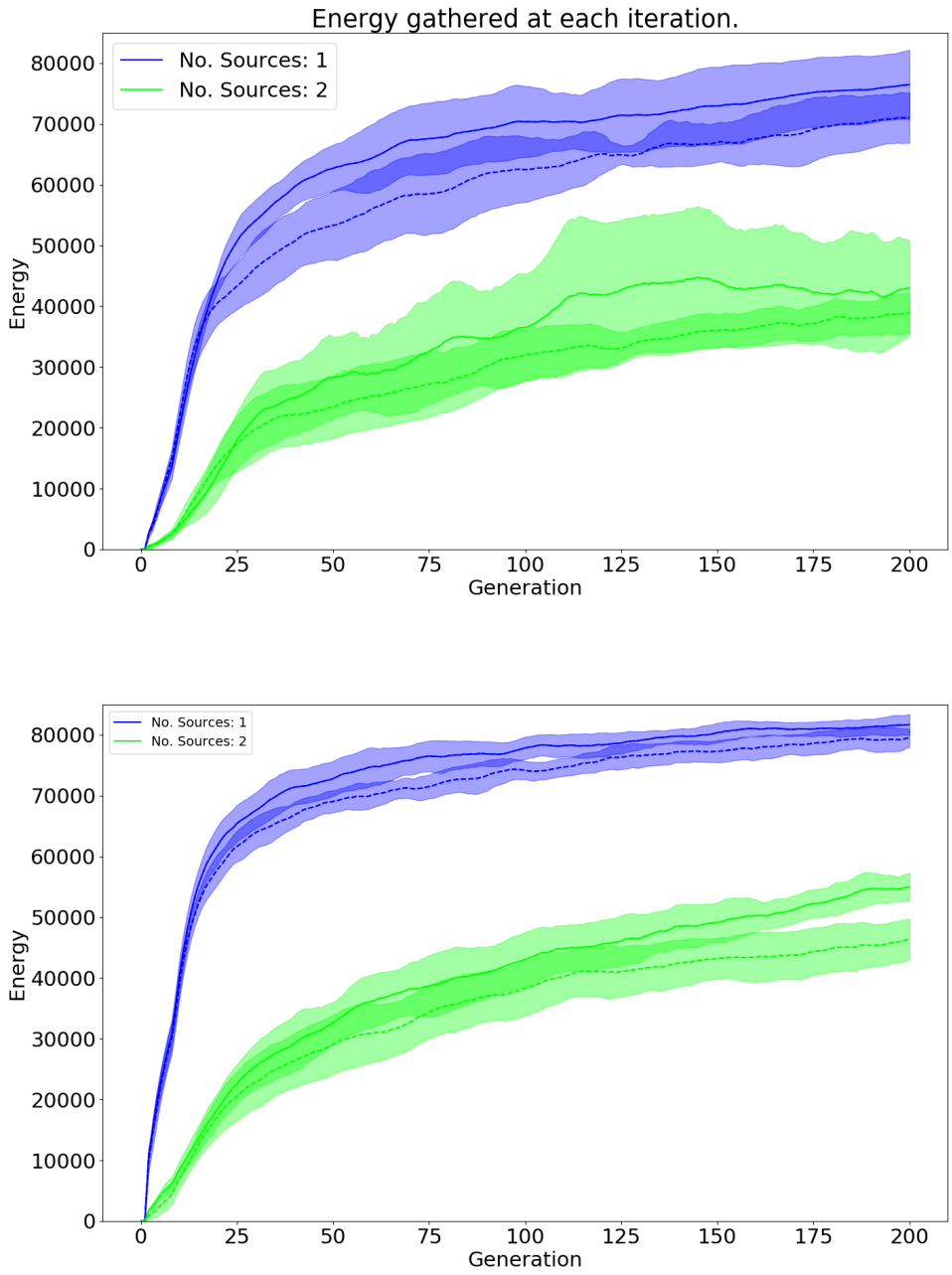


Figure 3.5: The simulations above varied the arena size, following the same parameters as presented in Table 3.1. The radius of the arena, however, is 10.0 for the solid plot, and 20.0 for the dashed plot. To ensure that the agents' received the same intensity of signal given the increase in size, ℓ was set to 20.0 and 40.0 respectively. Signalling was also turned off which is presented in the bottom plot. Each simulation was repeated five times to get an average energy timecourse. The swarm, predictably, has a higher performance for smaller arenas, which is due to less travel time for the agents when initialised. In the case of one source, however, they seem to be on par or better than when in an arena of size 2.5, see Fig. 3.3.

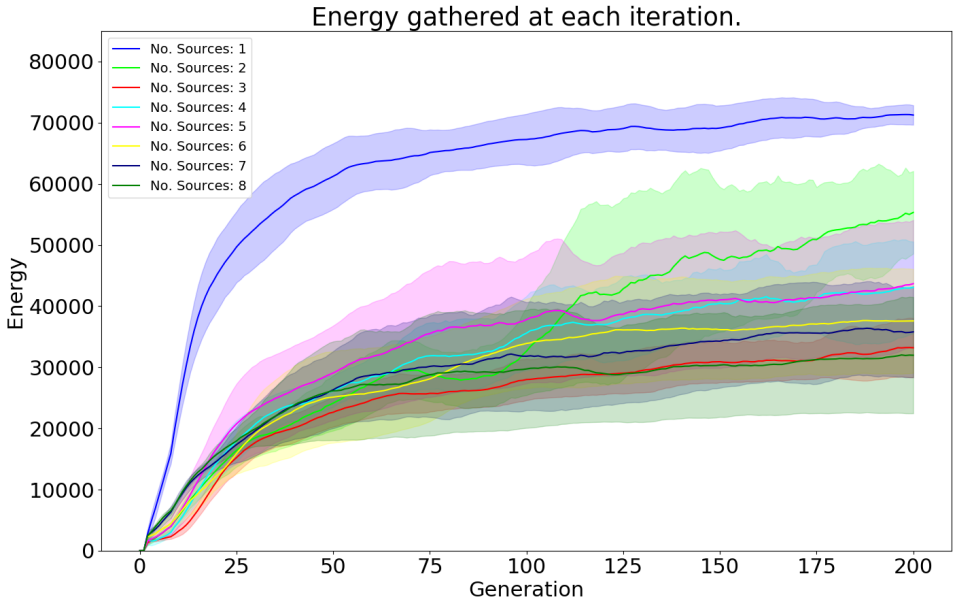


Figure 3.6: An evolutionary swarm must optimise gathering energy by being inside the green disks (see Fig. 3.2). The swarm uses the parameters presented in Table 3.1. The plot is of the total energy gathered at the end of each generation. There is a clear distinction initially in performance when the swarm is training in an environment with only one source, compared to swarms in an environment with multiple sources, and it is clear that there is confusion when more than one source is introduced. Furthermore, in terms of performance, there is a balance between the area covered by all the sources and the number of sources. Each simulation was conducted 5 times to get an average timecourse, with a moving average applied beforehand.

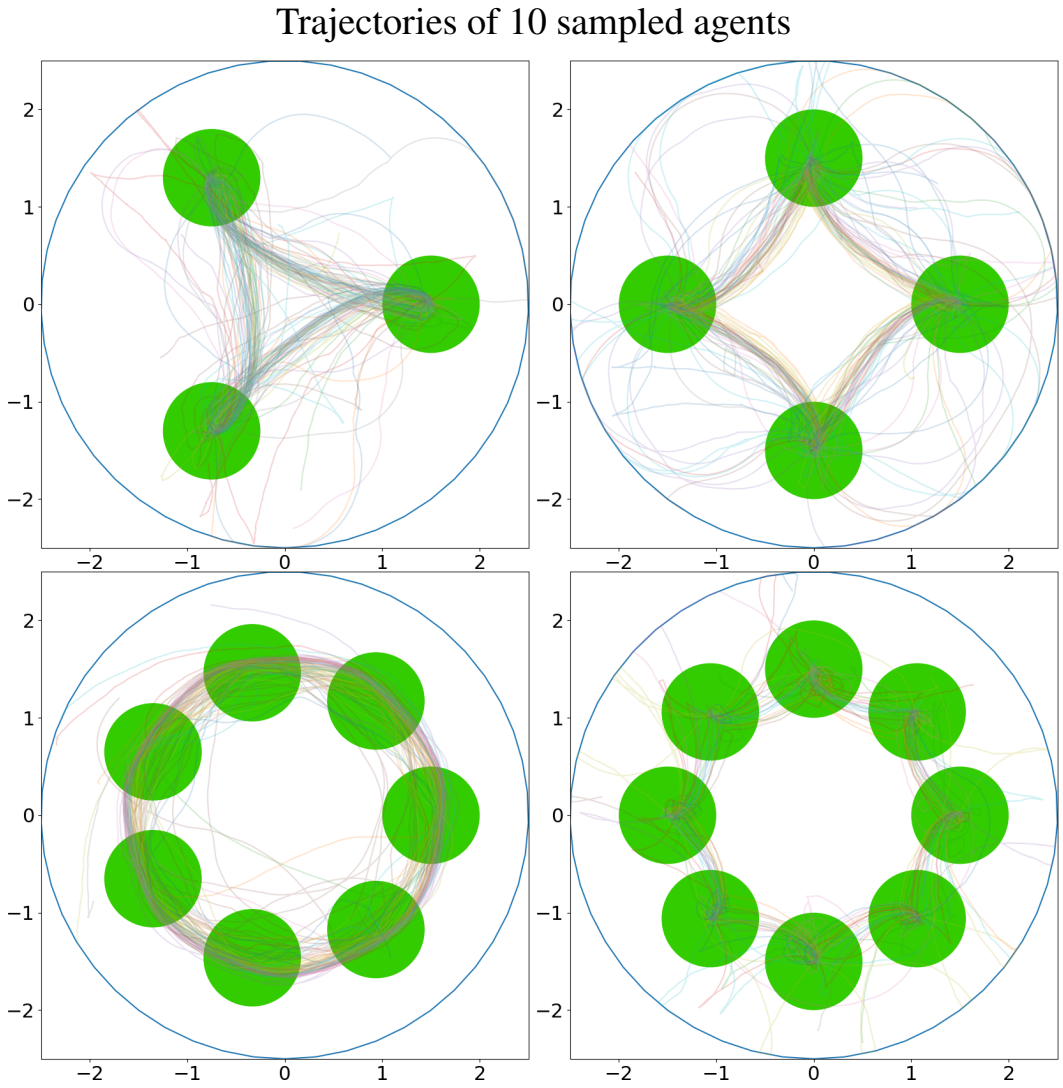


Figure 3.7: For each simulation the swarm was trained with a fixed number of sources. The above are samples to demonstrate the trajectories of 10 randomly chosen agents with a varying number of sources; 3, 4, 7, and 8 are shown. The agents develop a preference to cycle between the sources as more are introduced.

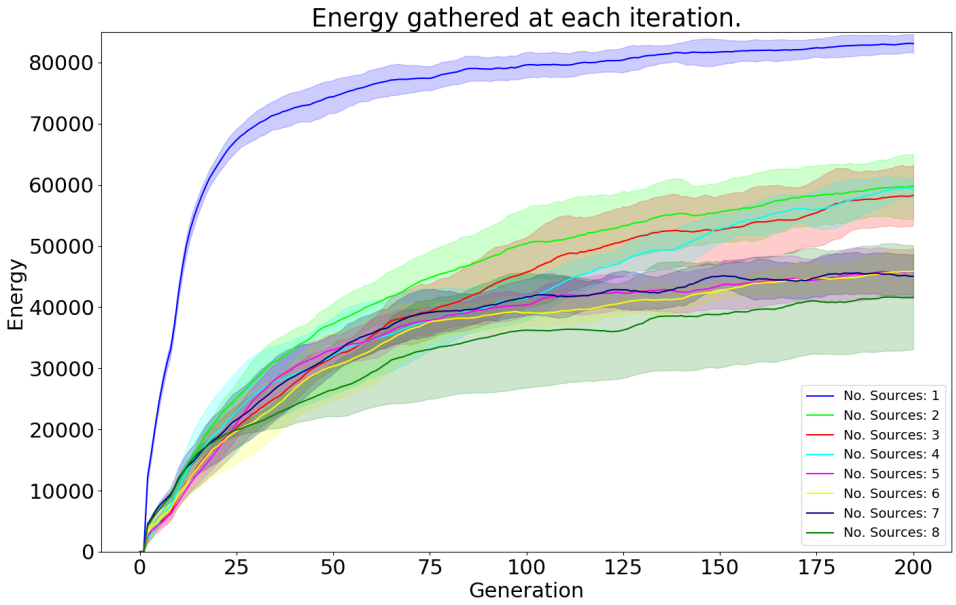


Figure 3.8: The setup is the exact same as that in Fig. 3.6, except communication between agents was turned off (by setting the communication range to 0). Each simulation was conducted five times to acquire an average timecourse. The swarm overall performs better if communication is switched off, and the agents rely on distance sensing.

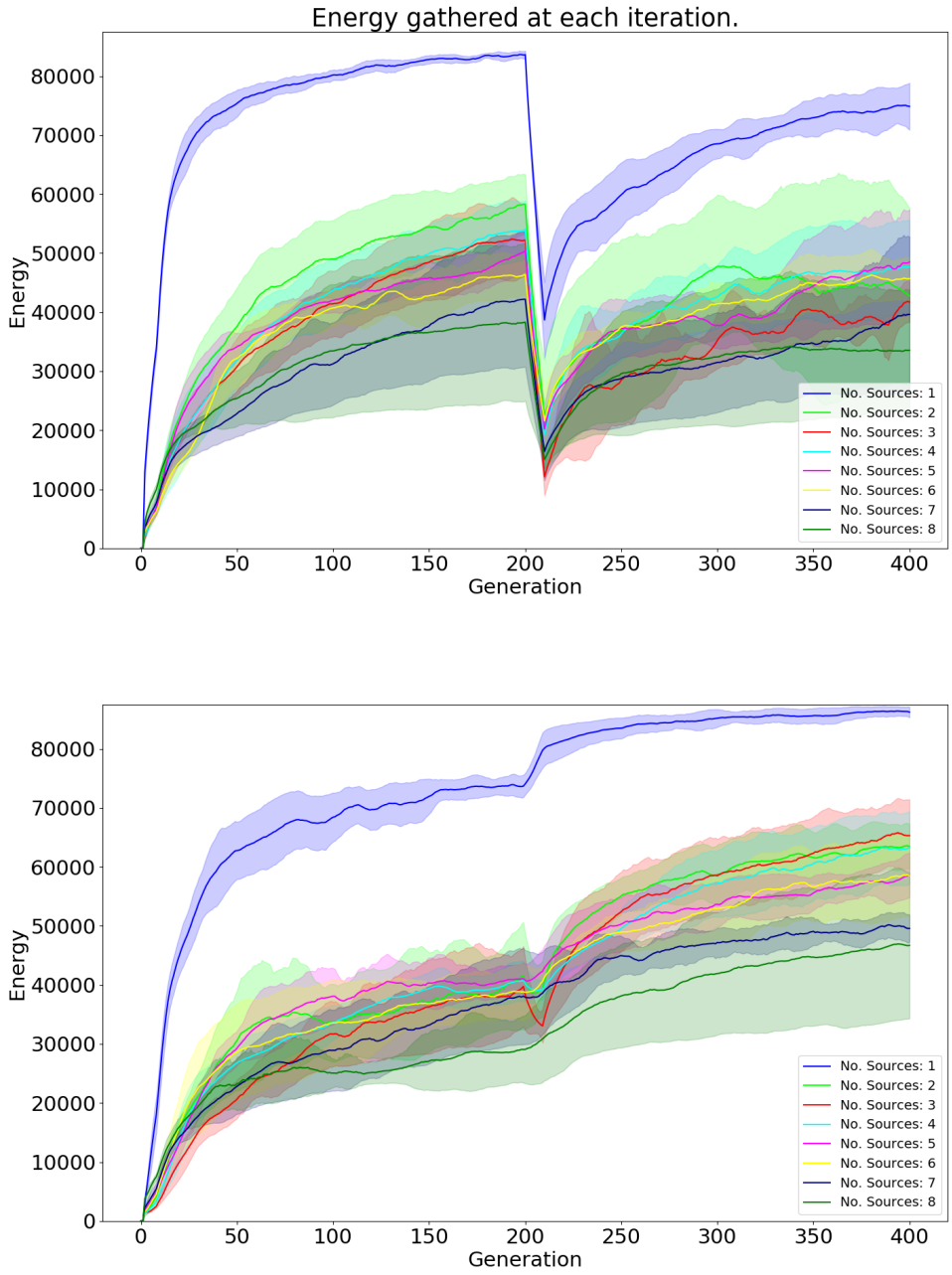


Figure 3.9: The simulations use the parameter choices in Table 3.1. The plot on top shows the simulations initially without signalling for 200 generations. Afterwards the signalling was enabled and the agents were trained for another 200 generations. The bottom plot is the opposite, for the first 200 generations signalling was turned on and then turned off and trained for another 200 generations. When the agents started without signalling, they initially did worse but were then able to learn, however they did not achieve the same performance level. For the reverse, performance did drop when signalling was disabled, but after retraining final performance improved.

successful, regardless of number sources, when signalling is switched off and given time to learn. This could also be an indication that the signalling is encouraging the pulsing behaviour of the agents, as the signalling could be interpreted as a repulsive force by the Elman Network controller, pushing the agent away from the optimal centre of a source and into traversing to another source.

3.4.4 Trained swarms in different environments

The swarms were initially trained in an environment with a fixed number of sources. Afterwards they were placed in environments different from their own, i.e. a different number of sources for one generation, see Fig 3.10.

The results demonstrate that when placed into environments different from the one the swarm evolved in the performance is poor, particular the swarm trained on only one source. The behaviour of this swarm was considerably less organised as it only had one source to ever consider. The opposite can be stated about the swarm trained on eight sources, where it was never fully able to optimise its own behaviour in its training environment. Furthermore, the agents in this swarm were residing in an environment where their distance sensors were consistently triggered as there was increased chance of having multiple sources surrounding said agents, resulting in a significant change of information processing when switching environments.

It is found that the optimum number of sources for training purposes is three. This can be due to the sources not always being seen, which avoids the issue that training on eight sources encounters. Also multiple sources can be detected at the same time, which is how it is perhaps outperforming the swarms trained on fewer sources.

3.4.5 Active Information Storage analysis

A component in a complex system is considered to hold information through its relationship with the other components (Zipser et al., 1993). Active Information Storage (AIS) is an analysis tool for determining the information storage within a distributed system Lizier et al. (2012), and can also be calculated for swarms (Miller et al., 2014). Here we look at the local AIS to determine how useful the past is for predicting the current state, which we calculate for each agent and to determine the local AIS for the swarm we simply take an average. Local AIS for an individual agent is calculated as

$$a(\hat{v}_t) = \log \frac{p(\hat{v}_t|\hat{v}_{t-1})}{p(\hat{v}_t)} \tag{3.11}$$

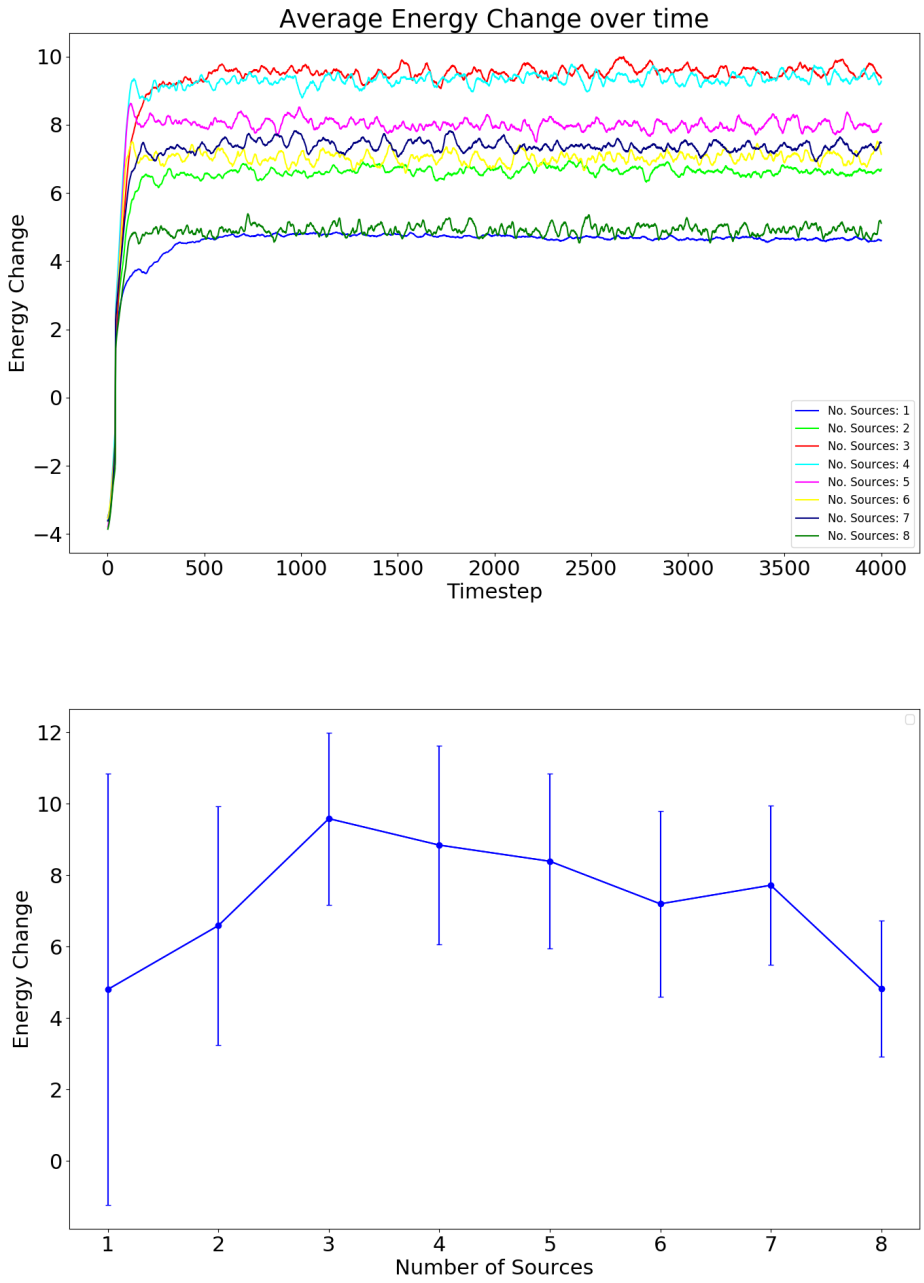


Figure 3.10: The above simulations were initially trained for 200 generations on an environment with a fixed number of sources. They were then tested on environments with a different number of sources for one generation, as well as an environment with the same number of sources as the swarm was trained on. The parameters used are presented in Table 3.1. Five swarms were trained for each initial setup to then allow an average energy timecourse to be calculated. The plot shows the average energy change at each time step across all environments. There is a balance required for the swarm with regards to the number of sources to train on in order to be able to perform, on average, the best in all environmental setups. In this instance it is three.

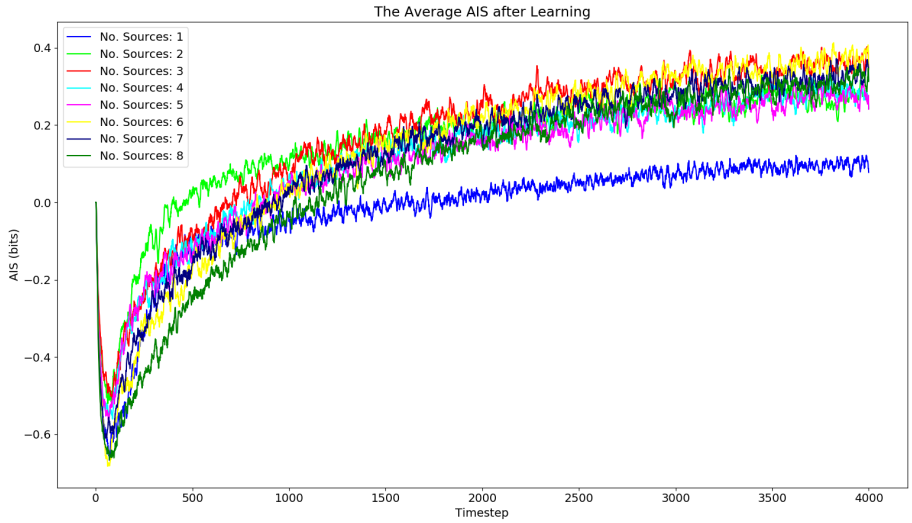


Figure 3.11: The graph presented is the AIS of swarm, as described in Fig. 3.6, in its final generation. While there is no significant information storage, there is a difference between a swarm operating in an environment with only one source compared to the swarm learning in an environment with multiple sources. The timecourse is the average of five full repeated simulations for each setup.

where $\hat{v}_t$ represents the agent's state at time t which here is the change in velocity and the velocity magnitude: $\hat{v}_t = \{v_t - v_{t-1}, |v_t|\}$. The probability distributions are approximated by generating histograms for each generation. The state's representation and histogram approximation is the same approach as that used by Miller et al. (2014). It is possible to have $a(\hat{v}_t) < 0$ meaning that the past is misinformative.

Fig. 3.11 displays the local AIS for the swarms presented in Fig. 3.6. Predictably local AIS is negative at the start, due to initialisation, as there is no data to make a distribution from to calculate accurately the next state. As the simulations continue the swarm starts to store information regardless of the environment they are evolving in. The swarm also stores more information in environments when there are multiple sources, however there is little difference between the setups of two to eight sources. The results do imply that the swarm is attempting to learn and capture the information presented in the environment, however combined with previous results in this investigation it is not enough to fully exploit it.

3.4.6 Evolving the distancing sensing range

The simulations thus far have only considered evolution of the controller and not of the physical capabilities of the agents. Furthermore by switching off signalling the swarm is more successful with the task. Together with results presented in Sec. 3.4.4 there is a potential information overload that the swarm's learning cannot overcome. To investigate this the simulations were conducted again except the range of the distance sensors are now also a parameter that can be mutated. The mutation rule is

$$l_{\hat{n}} = l_n + p_l \tag{3.12}$$

where $\ell_{\hat{n}}$ is the distance sensor range offspring of agent n, and p_l is a uniform distribution for the step size. There is no upper limit to the range however $\ell_{\hat{n}} \geq 0$. For the following results p_l is drawn from the uniform distribution $U(-2,2)$. Each setup was simulated five times to acquire an average timecourse.

Simulations were conducted of the same setup as described in Sec. 3.4.2 with the inclusion of distancing sensing evolution, see Fig. 3.12. There is an overall increase to the accumulated energy, although there still exists the pattern of the swarm performing worse as the number of sources increases beyond one. Observing the swarm's average distance sensing range, the swarms that performed worse are also the swarms that evolved to have, on average, the largest range.

The simulations were ran again except the starting range was set to 1.0, see Fig. 3.13. The distance sensing range for one source evolved to be just under 5, which given the size of the environment allows the agent to always see the source. When increased to two sources the range maintained at around 1, where this would mean that while the agents might not always see a source, they will most likely only ever detect one source when they do detect a source. Beyond two sources the swarm essentially becomes blind, relying solely on the local communication and energy consumption information (i.e. are they currently inside a source or not). Overall this led to a substantial increase in the accumulated energy gain compared to any of the previous simulations.

3.5 Discussion

This research primarily investigated the swarm's ability to evolve and optimise the energy gathered if given infinite distance sensing, so to speak, as well as local communication. The results show that the swarm will be more successful if they lost the ability

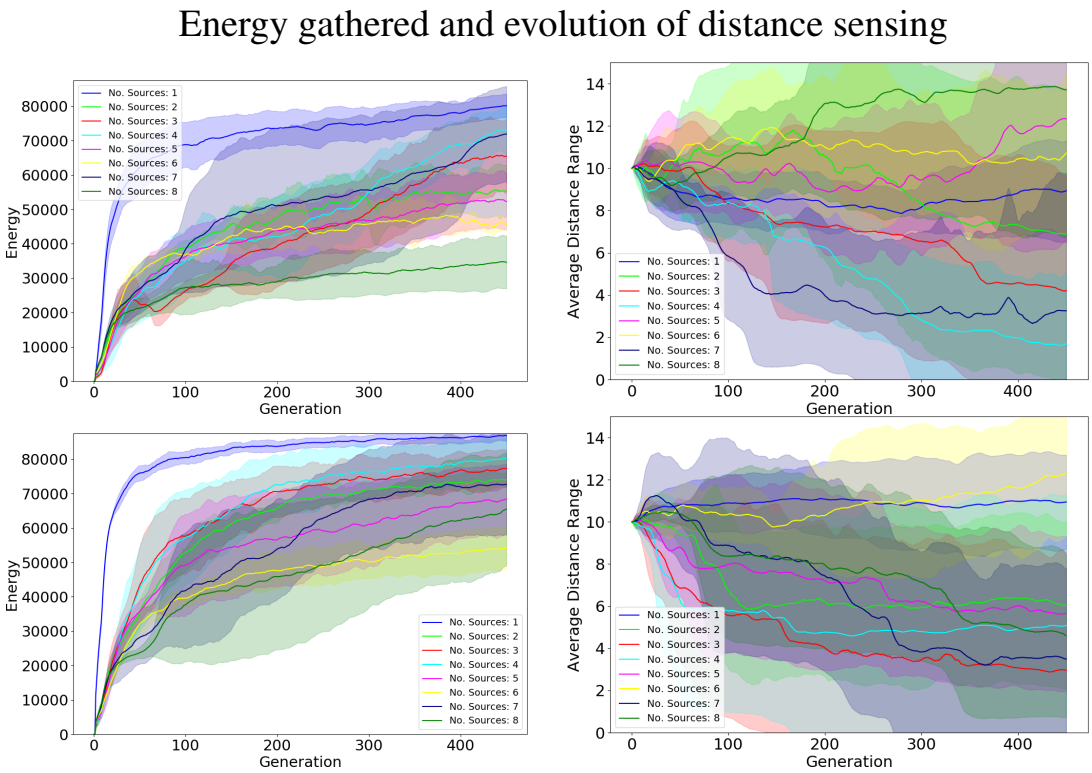


Figure 3.12: Evolutionary swarms performing the same task as in Fig. 3.6, with the parameters in Table 3.1 and $T = 450$. The swarm, however, can now evolve their distance sensor range, ℓ. The left plots represents the accumulated energy, and the plots to the right the average distance sensor range. The top row is when signalling was turned on, and the bottom row is when signalling was turned off. Each setup was repeated five times to acquire an average. While in general the swarm performs better with the inclusion of evolving the sensing ability there is still a drop off in performance when the number of sources increases. We can compare the swarms' performances with their respective average distance sensing range, and it can be observed that if a swarm evolves to have a lower average range they tend to perform better than swarms that have higher ranges, regardless of the number of sources present. For example, the swarms in the top row's plot that were trained on four sources evolved to have a shorter distance range than the swarms trained with two sources, and the swarms trained on four sources performed better than the swarm trained on two sources.

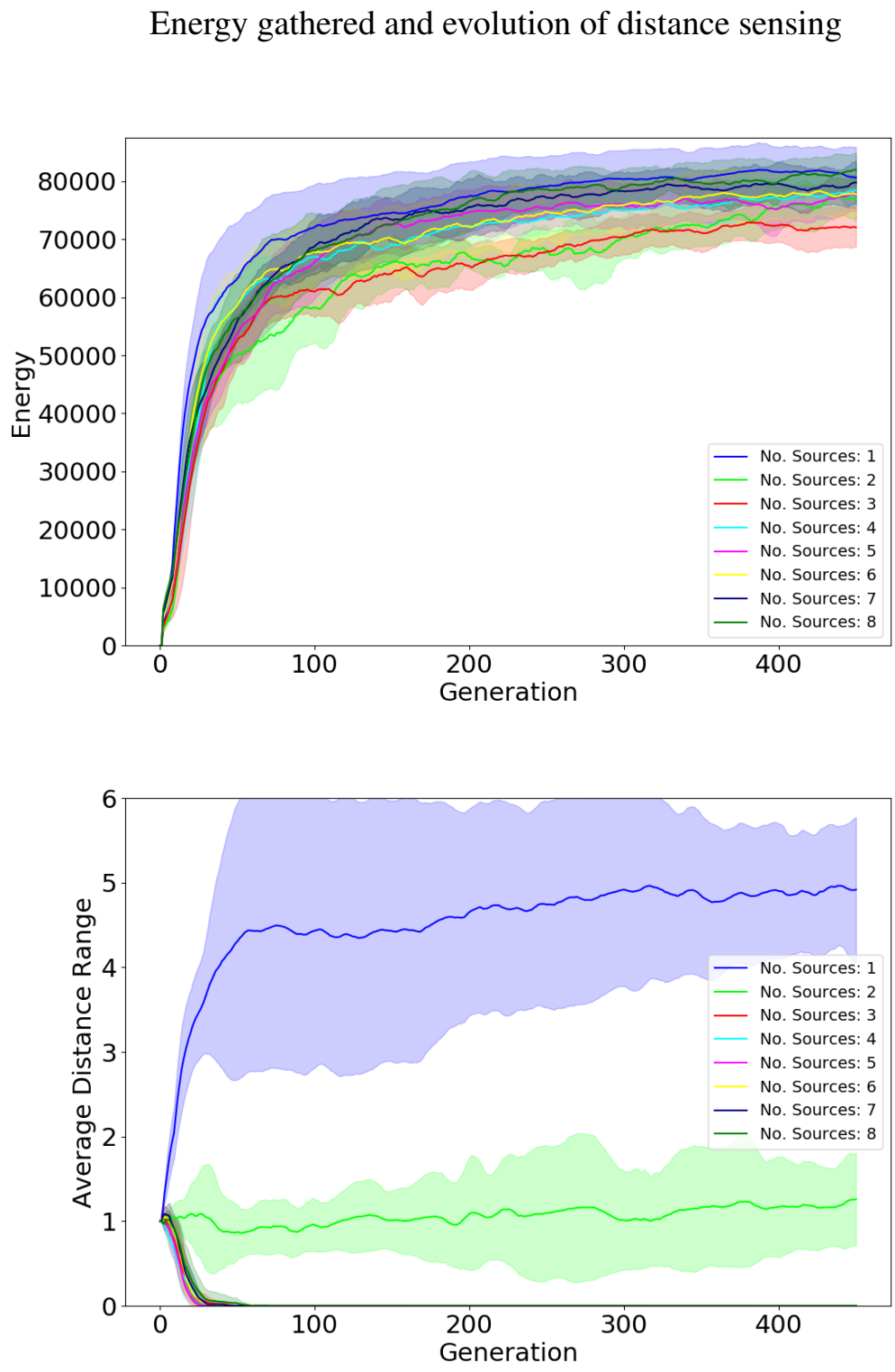


Figure 3.13: The results presented here are from simulations with the same setup as described in Fig. 3.12, except the initial distance sensing range was 1. The graph on the top is of the energy gathered and the graph on the bottom is of the distance sensing range. It is apparent that the swarms with this initial setup performed the best compared to previous simulations presented in this work. With one source the swarm evolved its distance sensing range to just under 5 length units, meaning they were always able to detect the source. With two sources the swarm evolved so that while it could detect a source it would be unlikely that it could detect both at any given time. Furthermore a striking finding is that the swarms evolved to practically remove distance sensing entirely when three or more sources are present.

to communicate, and have less sources to be considered, even if it means a drastic reduction in arena space covered by reward. We then allowed the swarm to evolve their physical capabilities which in this case was the range of their distance sensing. This led to the swarm evolving towards limiting their sensing, even as far as blinding itself. This intuitively is demonstrating that the swarm not only wishes to limit information but choices for decision making, especially for observing equally viable sources.

3.5.1 Agent and environment complexity

There are several steps that should be taken to fully understand this emergent confusion. Firstly the problem can be made more complex, such as increasing the dimensions, to observe if the additional search space gives the swarm the freedom to evolve away from the emerging confusion.

From observing swarms in different environments than the ones they were trained in and the distance sensing evolution observed, we should perhaps investigate further the evolution of the physical capabilities of the agents. The number of distance sensors for example could have an effect on the confusion, and observing if preventing the ability for a sensor to always be triggered helps prevent information overload.

Furthermore, the observation that the range of the distance sensor becomes zero for three or more sources (Fig. 3.13) is striking, but not entirely surprising. Given the initially uniform random positions, a swarm of 100 agents, each with a communication range of 0.5, is likely to cover the entire environment at the start. The results also demonstrate that the signalling is perhaps a source of noise that cannot be minimised, so instead disables the physical distance sensing to thereby reduce the overall information being perceived by the system. Investigation in the future needs to also look at the initial setup of the agents to fully grasp the physical evolution. This in turn does lead to question the crucial need to evolve the entire agent and not just an aspect, i.e. just the controller/cognitive side. This work has highlighted the requirement to further understand the harmony of the physical and cognitive qualities to perceive the state and to understand said detected state.

In the simulations presented there were no obstacles in the environment. If obstacles were to be added, the agent setup would have to change as well; either the obstacles block the view of sources from the distance sensors, or the agent can detect both the obstacles and the sources requiring more input neurons for the controller. In the former case the agents would effectively have their options limited resulting in the

agents remaining at a source, as long as no other source is visible from said source. This would then lead to cyclic behaviours occurring at smaller scales, i.e. cycles between sources that are visible to each other. If the environment was changing every generation and the distance sensing range was evolving as well, then the range would most likely also evolve to be minimal. This will be due to the difficult nature of anticipating the distances between sources that will be visible to each other, unlike that which was observed in Sec. 3.4.6 when one or two sources were present. If the setup was that the agents could detect both the obstacle and the out-of-sight sources at the same time then this might deter agents from going towards sources that require time to circumvent said blocking obstacle. However, if obstacles are not blocking direct paths between sources cyclic behaviour will probably occur. The agents are also operating entirely locally, again meaning agents will perhaps want to have minimal sensing for sources and obstacles due to lack of explicit global information to exploit. Though for obstacles perhaps the source sensing range could be at least large enough to circumvent them efficiently. Out of the two cases discussed here, the latter would be a good step to pursue, as it would be interesting to observe if symmetry breaking occurs for the swarm's performance.

In natural systems, symmetry is usually broken, in part due to a lack of information available, such as ants pheromone trails being competitive to ensure that the ants converge on one food source (Hölldobler and Wilson, 1990). Individuals in animal collectives can also enforce that decisions are made and continue progress rather than potentially waste time deliberating. This behaviour can be seen in schools of fish (Couzin et al., 2011), and in packs of gorillas (Couzin et al., 2005). Evolution in natural systems evolves both the cognitive and physical aspects, which results in the emergent decision making behaviour. This is at times however not fully realised in artificial systems, particularly in physical artificial systems. The controller is usually the only part that undergoes evolution and this can be hindered if the physical aspects of the system is not appropriately selected, which is observed in this Chapter. To fully utilise the concept of evolution there should be a harmony emerging from the controller and the sensors/actuators; the controller learning how best to link the sensors/actuators to the environment, and the sensors/actuators linking the environment to the controller.

3.5.2 Non-equal sources

A crucial aspect of this investigation was that all sources were equally as valid as each other. One approach to making the sources unequal is if they are explicitly rewarding different amounts of energy. In this setup the input neuron that detects energy change, c_7, would play a strong factor in learning the distribution of energy. If the agent was within a low rewarding source, then perhaps the information from c_7 will encourage the agent to cycle to the next source. If the agent was within a high rewarding source then the information from c_7 might dampen the effects produced from detecting other sources. This is, however, with the assumption that the sources and their respective reward levels remain constant between generations. If they do not, then the swarm will not be able to properly learn the distribution; i.e., what is high and what is low? This in turn would lead back into a cyclic behaviour as the agents might never be certain that they are in the highest energy source.

Sources could also be implemented to shrink or grow like plants. In the case of a source shrinking, the swarm would benefit from a good exploration strategy as the swarm, once depleting a source, will need to find another source to obtain energy. The swarm should still remain at a source until it is depleted before traversing to the next source, although this would unlikely happen given the cyclic behaviour emerging. Thus, the distance sensing range, if evolving would evolve to be of minimum range to optimise the time spent at a shrinking source, i.e. minimise information overload. If the sources grew then they could perhaps act as larger attractors becoming closer to the agents, thus leading to cyclic behaviour. In either case, the swarm will most likely benefit from minimising their distance sensing range to avoid detecting too many sources at one time, and performing suboptimally. Similarly, it would be interesting to explore this in conjunction with the setup of varied source energy distributions described as earlier.

Another approach is to change how much energy the source rewards dependent upon the number of agents present, be it either more agents provides more energy or vice versa. In either case the agents will not know how many agents are already occupying the source until they reach said source. The communication between the agents will be key as this will act either an attractive or repulsive force. The agents will, however, still remain uncertain about other sources and the presence of agents at those sources, so explorative behaviour may still be strongly present resulting in cyclic behaviour to find the best source. That said, if the setup was that more agents resulted

in less energy reward then having distance sensing would be important if the agents are needing to know where other sources are if there are too many at one source. This could lead to a cascading effect of agents constantly repelling agents, and resulting in a significant potential drop in overall swarm performance due to the agents travelling from source to source.

3.5.3 Information processing in the swarm

Regarding this thought we can turn to recent work done by Yanagisawa et al. (2019) where they start with Bayesian probability and Shannon information to form a functional model of emotion reliant upon information gain. Too little or too much information gain results in a negative emotional state, which is perhaps what artificial systems are requiring to consider. In this case receiving a vast quantity of information, i.e. negative valency, should encourage the evolutionary learning to move away from the current environmental/agent setup to better tune the controller for general situations. While the opposite extreme should encourage exploration and further understanding of the environment via the cognitive quality.

Another aspect of information theory to consider is transfer entropy, which aims to identify the flow of information between systems. Crosato et al. (2018) applied transfer entropy to studying a school of fish and the information cascading through the fish, and whether this was informative or misinformative. This approach can be used here as well as it seems to be demonstrated that the school is able to recover as a whole from misinformation passing through. Recently, Suganuma et al. (2019) proposed that maximising transfer entropy within the evolutionary learning aspect of heterogeneous swarms as a way to divide labour efficiently. This division of labour is akin to the idea of agents following specific roles, such as leadership, for increasing foraging efficiency which is seen in nature (Couzin et al., 2005).

Most fascinating is the overload of information that the evolutionary strategy tries to overcome by attempting to utilise all said available information. The agent could be too advanced for the task, and leads one to question not just evolution of the ability to function but the capabilities themselves. The field of criticality can easily be applied here as there is a crucial balance of explicit information that should be made available to the system and the environment. Systems in a critical state are able to maximise their performance with the given resources (Legenstein and Maass, 2007) and there is evidence of this happening in natural systems, such as the frontal cortex (Shew and

Plenz, 2013). Observing if the swarm approaches a critical state will identify if the swarm's evolution is akin to natural systems, or if it is in fact in a supercritical state, which will require the swarm to expand what it means to evolve from one generation to the next to still achieve goal directed behaviour at perhaps the sacrifice of resources.

3.5.4 Practicality in robots

We need to consider the reality of using this as an approach for robotic swarm applications. As demonstrated here, the swarm can fully optimise its behaviour, if the change of detectable information, i.e. the distance sensing range ℓ in the agents, is considered. Although hardware evolution in robots is also studied using natural computing techniques, such as genetic algorithms and evolutionary strategies, it is not currently an approach that could be deployed in a real-world application. There are, nevertheless, impressive examples of partly simulated physical agents which are promising for practical tasks (Kriegman et al., 2020; Hale et al., 2020), but the evolution of robots with generic abilities still remains a challenge. As shown in Fig. 3.10 the swarms trained in the environment with a similar number of sources as distance sensors was able to generalise the best, but this would require the designer to know what an environment should entail and how to best design the physical robot ahead of time. With each decision made the generality of the solution, and thus the robustness to changes, begins to decrease.

3.6 Conclusion

This work investigated an evolutionary swarm set with the task to gather as much energy as possible from sources, where each agent had the ability to detect distances to sources and communicate locally. This investigation has uncovered that exposing the swarm to more information can inhibit goal direction within the swarm's behaviour, and leads to confusion on how to interact with the environment. When presented with more than one source the swarm's performance significantly drops, and the behaviour of the agents is to cycle between the sources. Furthermore, by switching off signalling, either initially or during said simulation, the swarm is more successful in all environmental setups. The inner swarm communication seems to be acting as noise that the evolutionary algorithm is attempting to disseminate. However, it is of little effect as by switching it off demonstrates that the agents perform better without signalling infor-

mation. It was then observed that allowing the swarm to evolve the distance sensing range greatly improved the overall performance, with the striking result of the swarm reducing said range to 0 with multiple resources present. This is to avoid the controller being given multiple choices on which source to go to.

Future work should include observing the individual agents' capabilities given the task, as well as the criticality of the system to better understand the underlying implicit learning of evolutionary algorithms for low level systems. We also need to learn more about how to quantify the information overload, and track flow of task relevant information in the system.

Furthermore, we need to strongly consider the realism of using this approach on physical robotic swarm applications. This thesis is not saying that learning is not valuable, but given the limitation of what can be done to the physical robots in real time we might want to consider swarm behaviours that can perform a complex task with a general solution, i.e. robust to perturbations, and use these as starting building blocks for solutions to real world problems.

Chapter 4

Turing patterns for robotic swarms

Robotic swarms rely strongly on local interaction to be able to, as a collective, perform complex tasks. Reaction-diffusion systems model the spatial concentrations of chemical species over time, and this can lead to periodic spatial patterns known as Turing patterns. The following Chapter shows that the strength of local communication with a swarm consisting of agents that can, through messaging their internal values to local neighbours and performing individual explicit reaction-diffusion equations, produce Turing patterns in a virtual space, which is not known to the agents. Furthermore, this can be achieved even with the swarm in motion and can recreate these patterns in the physical space. Multiple patterns can be achieved, though stripes require the swarm to be initially closer together. The Chapter then takes this concept further by introducing points of interest and a single sensor to induce a desired stripe pattern. It also shows that the RD system can work in robotic platforms, specifically 'physically realistic' Kilobot simulations.

4.1 Introduction

Swarm robotics is the study of collections of simple robots which can complete limited tasks individually. As a collective, however, they can attain an implicit high level intelligence. Swarm robotics relies strongly on local interaction to determine their behaviour, and each robot employs low level rules. This results in a robust system with adaptive capabilities, be it with respect to the environment or individual robots perishing. In Chapter 3 it was observed how learning can lead to confusion if the system is provided with a plethora of equally viable choices, and aims to restrict the incoming information via 'physical' evolution. To this end, a swarm could benefit

from a self-organising system which does not require anything beyond messaging, at the very least as a starting position for building towards applications.

Jin and Meng (2011) describe how morphogenesis, the biological process of self-organisation for shape development, can be applied to robotics. This field is considered here, specifically reaction-diffusion for swarm organisation in spatial tasks. Reaction-diffusion systems are known to create self-organised periodic spatial patterns, known as Turing patterns. These can explain the patterns observed in nature such as animal skin patterns (Kondo, 2009) and mammalian palates (Economou et al., 2012). These patterns are robust to the injection of small amounts of noise, and different patterns can be formed through parameter selection. Our work takes a reaction-diffusion setup and applies this to a swarm of simulated agents by replacing each point in space with an agent. Not only are the agents able to create a virtual Turing pattern, but can also replicate this pattern in the physical space. The patterns formed are not just hexagonal honey-comb spots, but can also be stripes and inverted spots. Furthermore, the agents and robots in our setup do not have any external sensing, and solely rely on communication to form these patterns in the physical space. By adding a single binary sensor, the swarm can then be given the ability to induce patterns, which the swarm demonstrates by being able to form a stripe pattern, bridging multiple fixed points.

The Chapter first gives a brief overview on Turing patterns, as well as the RD system to be implemented. How RD can be deployed onto a swarm is presented, followed by the patterns that it can form. The swarm is then given a single binary sensor to illustrate how the RD swarm can be built upon to achieve spatial tasks. It is then demonstrated that an RD system can be integrated with robotics as well, and this Chapter uses a physically realistic Kilobot swarm to illustrate this. Lastly a discussion on how this investigation starts with a dynamical system known for self-organising into spatial patterns through simple communication alone can act as a building block for robotic applications, and how to improve the RD swarm.

4.2 Turing patterns

Turing pattern is an emergent periodic pattern from reaction-diffusion systems. Here Turing patterns are described, as well as their potential uses, and how Turing patterns can be simulated in a discretised space.

4.2.1 Background

Reaction-diffusion (RD) systems model the temporal spatial dynamics of the concentrations of multiple species, and for simulation purposes the space is discretised. The reaction component is how the concentration of a species changes given the concentration of all species. Secondly, the diffusion captures the species random motion, and the resulting change in concentration for each discrete space.

This work considers two species RD systems, and one species is normally known as an activator and the other as an inhibitor, as the concentration of the inhibitor species results in reducing the activator's concentration. An example of this is the Klausmeier model (Klausmeier, 1999), in which the concentration of water and plant life are being modelled. The plant life is the inhibitor as it is absorbing the water, i.e. reducing the concentration of water.

Alan Turing in 1952 hypothesised that spatial patterns can emerge from dynamical systems through introducing a diffusing instability, which can then be used to model certain natural phenomena. This hypothesis was demonstrated some time later, and so was named Turing patterns. Reaction-diffusion (RD) systems produce stripe-like or honeycomb-like Turing patterns (Turing, 1952) if two substances are spreading with different diffusion constants, such that the inhibitor is vanishing faster than the activator that in turn has caused its production. This approach was expanded to create an extended Klausmeier model (Siero et al., 2015) which allows the creation of striped, or spotted plant formation, depending upon the rainfall, through the reaction-diffusion system where the species are water and plant. A widely studied model is the Gray-Scott model (Gray and Scott, 1983), which was initially a simple system consisting of simple rules for chemical production such as $A + 2B \rightarrow 3B$, and it demonstrates how this simple homogeneous system can display relatively complex behaviours. This has since been expanded and studied in one, two, and three dimensional models that exhibits pattern formation and Turing patterns (Munafo, 2014).

It is also found in animal skin pigmentation (Barrio and Varea, 2006), for example, the angelfish (*pomacanthus*) has striped patterns on its body that change throughout the juvenile years. The patterns begin as curved, almost ring-like, but change into horizontal lines, and this process can be modelled through a reaction-diffusion system, that forms a stripe pattern that changes. Other animals that this can be seen in are the thirteen lined squirrel, sea urchins and the gila monster lizard.

The application of RD systems in robotics is just at its beginning. Morphogenetic

robotics (Doursat et al., 2013) uses Gene Regulatory Networks (GRN) in order to control multi-robot systems (Guo et al., 2009). Kilobots have been utilised to realise the potential of this approach (Slavkov et al., 2018; Carrillo-Zapata et al., 2019). Although in their setups they only produced spotted patterns, in this Chapter we shall demonstrate that stripes and inverted spots can also be generated. There will also be a difference in how the Kilobots operate. The traditional setup is to have the Kilobots initialised together in a single cluster, with the outside Kilobots moving around the perimeter of the cluster, with the rest of the Kilobots acting as beacon for positional control. In our approach, we show that the Turing patterns can be produced in generic simple agents, not just Kilobots (they are used as a platform to demonstrate this). Thus, we do not use the traditional Kilobot control approach as seen in Slavkov et al. (2018) and Carrillo-Zapata et al. (2019), i.e. all the Kilobots in our setups can be operating at the same time, and the Kilobots do not need to know positional data.

4.2.2 Simulating Turing patterns

Turing patterns can be simulated, but let's first consider the RD equations required. The system will use a two species setup, as commonly seen and described in Sec. 4.2.1.

The general form for RD equations is

$$\begin{aligned} \frac{\partial u}{\partial t} &= \alpha(D_u \triangle u + f(u,v)) \\ \frac{\partial v}{\partial t} &= \alpha(D_v \triangle v + g(u,v)) \end{aligned} \tag{4.1}$$

which describes the spatiotemporal dynamics of an activator u and an inhibitor v. The Laplace operator, $\triangle = \frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2}$, represents the diffusion of u and v, typically with diffusion constants $D_u < D_v$. There is also a step size, α, included here which will be used for this Chapter's simulations.

The dynamics can be realised e.g. by substances in solution, but a large variety of natural systems that follow this dynamics are known (Adam, 2006). The functions $f(u,v)$ and $g(u,v)$ are the reaction models for the respective potentials and may vary for the different applications. Often, the activation function is non-linear and the inhibition is linear. We will use the FitzHugh-Nagumo model (FitzHugh, 1961; Nagumo et al., 1962), as this is known to produce Turing patterns, i.e. choose the reaction terms, $f(u,v)$ and $g(u,v)$, as

$$\begin{aligned} f(u,v) &= \lambda u - \alpha_q u^2 - (1-\alpha_q)u^3 - v \\ g(u,v) &= u - v \end{aligned} \tag{4.2}$$

The parameter λ is a positive constant, and α_q is a constant between 0 and 1. Whether the system produces spot-like or stripe-like patterns is also dictated by the presence or absence of a quadratic term in the reaction models (Ermentrout, 1991), which is governed in Eq. 4.2 by the parameter α_q.

To simulate the system the space will be discretised, and each point in space will contain the concentration values, u and v. Thus let's say the space is discrete and can be represented by an $X \times Y$ array, and $u_{x,y}, v_{x,y}$ is the concentration values of u and v at (x,y). To replicate the diffusion each point will use their immediate four neighbours. The Laplace operator can then explicitly become

$$\begin{aligned} \triangle u_{x,y} &= \sum_{i=x-1}^{x+1} (u_{i,y} - u_{x,y}) + \sum_{j=y-1}^{y+1} (u_{x,j} - u_{x,y}) \\ &= (u_{x-1,y} + u_{x+1,y} + u_{x,y-1} + u_{x,y+1}) - 4u_{x,y} \\ \triangle v_{x,y} &= \sum_{i=x-1}^{x+1} (v_{i,y} - v_{x,y}) + \sum_{j=y-1}^{y+1} (v_{x,j} - v_{x,y}) \\ &= (v_{x-1,y} + v_{x+1,y} + v_{x,y-1} + v_{x,y+1}) - 4v_{x,y} \end{aligned} \tag{4.3}$$

A periodic boundary condition is usually implemented

$$\begin{aligned} u_{X+1,y} &= u_{1,y} \quad & v_{X+1,y} &= v_{1,y} \\ u_{0,y} &= u_{X,y} & v_{0,y} &= v_{X,y} \\ u_{x,Y+1} &= u_{x,1} & v_{x,Y+1} &= v_{x,1} \\ u_{x,0} &= u_{x,Y} & v_{x,0} &= v_{x,Y} \end{aligned} \tag{4.4}$$

While the periodic boundary condition is used here, as we progress to using agents and robots, a free boundary condition will be used instead.

For the purposes of this Chapter's work u and v will have the limits of $0 \leq u, v \leq 1$. At a high level glance at the reaction components, the diffusive parameter D_u is the acceleration which the chemical species diffuses, so if it is low we would expect the points to have a high u due to the slow diffusion. This will then result with an almost equivalent value of v. Since the diffusion of the inhibitor, D_v, is typically much higher, the surrounding points in space will receive a substantial concentration of the species v, which in turns significantly reduces u in those surrounding spaces. This is the competitive element of RD which is a component of the periodic nature of Turing patterns. Increasing D_u will lower the overall u as well as v, meaning less competition. This intuitively would suggest the emergence of spots connecting, leading to striped patterns. The parameter λ is a self generating parameter, meaning the higher λ the

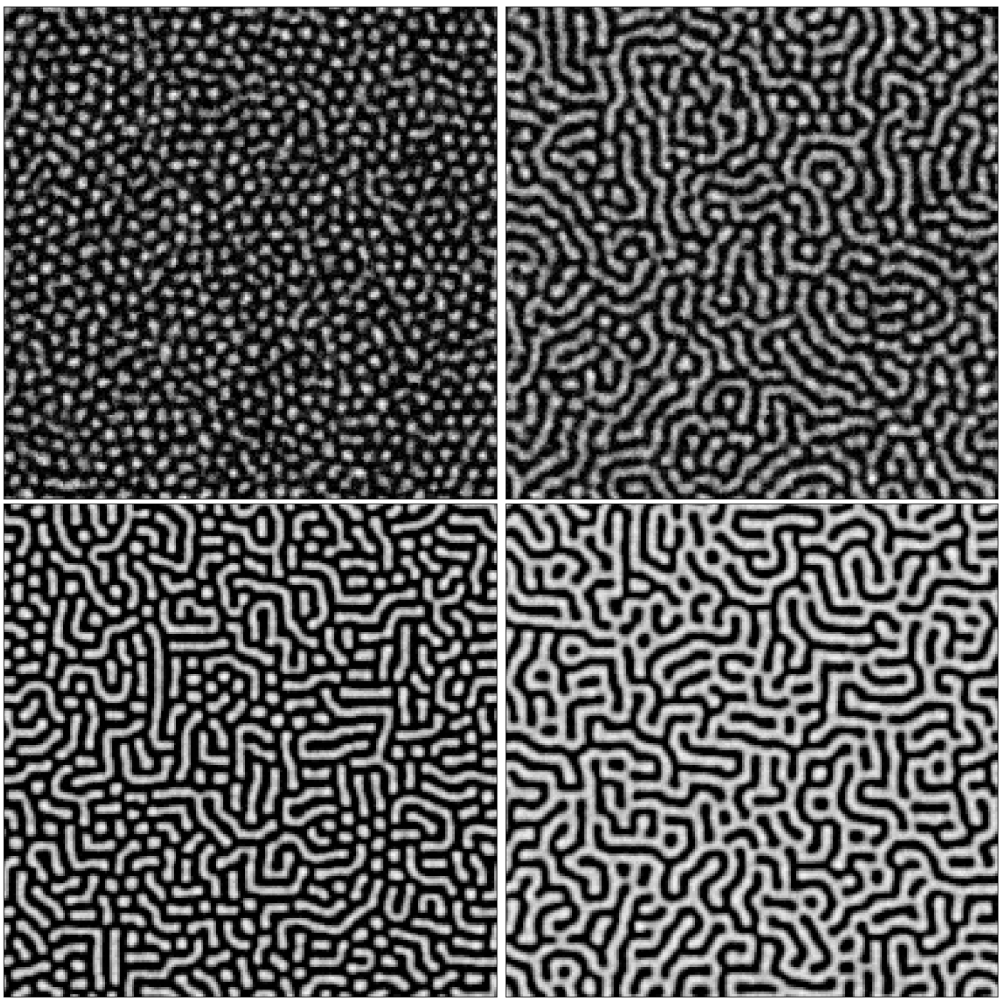


Figure 4.1: Various Turing patterns generated from different conditions. The space is a 200×200 array. The top row is when $\alpha_q = 1$, i.e. the quadratic term of Eq. 4.2 is used instead of the cubic, and the bottom is when $\alpha_q = 0$. The left column is when $\lambda = 1.05$ and $D_u = 0.3$ and for the right $\lambda = 1.1$ and $D_u = 0.5$.

higher the inhibition required, which would result in a lowering of the competitive element. Thus again we would have a similar effect on the system as D_u. Therefore for determining patterns we can initially turn to these parameters.

Fig. 4.1 contains the outcome of simulating the above RD system to produce different Turing patterns under different conditions. It is demonstrated here that the inclusion of the quadratic term (setting $a = 1$), and having lower λ and D_u values, do favour spotted patterns, while striped patterns are more predominant in the opposite case.

4.3 Reaction-diffusion swarms

The RD system now has to be represented by a swarm of moving agents. Here it is described how each point in space can be represented by an agent. First a description of the agent, which will initially have no sensors and rely solely on local communication, followed by the integration of the RD system, and then how the velocity is dictated.

4.3.1 Agent setup

The agents are circular masses, with radius r, occupying space in a 2-dimensional environment. They have position x and is updated via a velocity vector, v

$$x_{t+1} = x_t + v_t \tag{4.5}$$

where t is the timestep. As the agents have mass, they cannot therefore occupy or overlap the same space.

The agents do not have any sensing. They have the ability to communicate to their local neighbours. The local neighbourhood is determined by the Euclidean distance between the centre of the agents. At every time step the agents will broadcast their u and v, and the only other information from this communication will be the distance from which the message was received.

Lastly, as each agent is to represent a point in the RD space, i.e. each will carry values for u and v. This will again be subject to the condition $0 \geq u, v \geq 1$.

4.3.2 RD system setup

Simulations of RD models will discretise the space to allow explicit calculations across said space, as demonstrated in Sec. 4.2.2. To employ an RD system onto a swarm, each point in space is replaced by an agent and they contain the concentration values of the species on their point in space. This means that the system will use a free boundary condition. The agents at each timestep message their neighbours with their concentration values, with this information being used for the diffusion in the RD equations. The agents only know their explicit concentration values, and if the swarm is stationary then by observing the concentrations of each agent we can see the pattern emerging in a virtual space. The more interesting option, however, is to use the dynamics of the agents to represent one of the concentrations, and have the physical distribution of agents influenced by the pattern formed in the virtual space. The latter approach is

what will be the swarm's implicit aim, achieved by the explicit aims of the individuals. Beyond the scope of this investigation are heterogeneous swarms where different agents/robots may represent different species. It should also be noted that the agents cannot explicitly see the virtual space, other than their own point space.

The RD system, Eq. 4.1, will be calculated by each individual agent. The reaction components, Eq. 4.2, can be easily calculated by the individual agents as they have explicit knowledge of their own u and v values. It would be preferable for the swarm to be flexible and thus have the ability to attain different patterns, therefore $\alpha_q = 0$, i.e. the quadratic term will be removed, which takes preference away from clear defined spot patterns.

The diffusion process, however, will need to be updated. In the present context of the RD system only homogeneous second derivatives occur, i.e. directionality is not required for the system. This has the advantage of allowing simple communication methods to be used, such as the simple message receiving scheme described in Sec. 4.3.1. Therefore the diffusion process can be replicated in a swarm by redefining the Laplace operator in Eq. 4.1.

$$\begin{aligned} \triangle u &= -\hat{N}u_0 + \sum_{n=1}^{N}(1-\frac{d_n}{c})u_n \\ \triangle v &= -\hat{N}v_0 + \sum_{n=1}^{N}(1-\frac{d_n}{c})v_n \end{aligned} \tag{4.6}$$

where d_n is the distance between the agent and neighbour n, as described in Sec. 4.3.1, c is the communication range of the agents. The closer an agent is the higher the weighting for diffusion, i.e. diffuse more to closer neighbours as we would expect in regular RD systems. Therefore, $\hat{N}$ is the weighted sum of neighbours

$$\hat{N} = \sum_{n=1}^{N}(1-\frac{d_n}{c}) \tag{4.7}$$

4.3.3 Velocity updates

The agents have a velocity, v_t, which obeys the update rule

$$v_{t+1} = \alpha_v v_t + (1-\alpha_v)\gamma\hat{v}_t + \psi \tag{4.8}$$

where α_v is a positive constant between 0 and 1, and $\hat{v}_t$ is the new velocity computed which is dependent upon the activator's value, u_t. This makes the above an exponential average over the velocities dependent upon u_t. The parameter γ is a velocity factor

that controls the magnitude of velocity, which relies upon the maximum average u_t received. Lastly, it is also possible to denote the noise by ψ which, for simulation speed, is uniform noise. The new velocity, $\hat{v}_t$, is determined by the following

$$\hat{v}_t = \begin{cases} \omega v_t & u_t \geq u_{t-1} \text{ and } p_{0,t} < 0.99 \\ f(v_t) & \text{otherwise} \end{cases}$$
$$f(v_t) = \begin{cases} R(\theta)v_t & \text{if } 0 \leq p_{1,t} \leq \frac{1}{3} \\ R(-\theta)v_t & \text{if } \frac{1}{3} < p_{1,t} \leq \frac{2}{3} \\ v_t & \text{otherwise} \end{cases} \tag{4.9}$$

where ω is a positive constant and $\omega > 1$, and $p_{i,t} \in (0,1)$ with uniform probability. $R(\theta)$ is a 2D rotational matrix with angle θ. Simply put, if the concentration of the activator increases, then the magnitude of the agent's velocity also increases. Otherwise it uses Brownian motion for exploration. To help avoid stagnation, there will be a small probability that the agent uses Brownian motion regardless. The probability for this is 0.01.

As we are wanting the swarm to create Turing patterns in the physical space, the agents should preferably decelerate when they are near a stable pattern. One way to do this is to compare the current activation value, (u_t), to an average value. The velocity can then be dictated by the difference between the two; the closer an agent's activation value is to the agent's average activation value calculated implies that the agent's activation value is stable, and further implies that the agent is part of a formed pattern. Therefore this would need to significantly slow the speed of the agent, counteracting any acceleration being caused. For example if the agent's activation value increased due to noise this should dampened if the activation value is still relatively close to the average value.

An issue, however, is that the virtual RD system being performed is on a space that is constantly changing, due to the agents moving. Specifically, if an agent moves into range of another agent then the amount of agents to diffuse to has now increased which will lead to potential fluctuations in the pattern. Therefore, the activation value to consider is the neighbours', as when new large diffusion occurs, an agent within this new cluster will still contain what is the maximal u value, minimising fluctuations.

The agents compare at each timestep the largest activation value received, $u_{\max,t}$, with the average acquired, $\hat{u}_{\max}$.

$$\hat{u}_{\max,t+1} = \alpha_{\hat{u}}\hat{u}_{\max,t} + (1-\alpha_{\hat{u}})u_{\max,t} \tag{4.10}$$

Turing patterns consist of areas of high concentrations, and so to replicate this in the physical space the agents should be near agents that have high concentrations. This will attract the individual agents to cluster where there are high concentration values, thus translating virtual concentrations with physical agent concentration. This is employed by updating γ

$$\gamma = 1 - \frac{u_{\max}}{\hat{u}_{\max} + k} \tag{4.11}$$

where k is a small positive constant to avoid singularity, which for the following simulations $k = 10^{-4}$, and if $\gamma > 1$ then $\gamma = 1$. Lastly, the agent has a limit for the magnitude of the velocity, $v_{\max}$, i.e. if $v_t > v_{\max}$ then $v_t = v_{\max}$.

4.4 Pattern formation

Using the swarm described in Sec. 4.3 it was implemented in a particle simulator. The following shows the results of the swarm, and the different patterns it was able to achieve.

4.4.1 Environment setup

To demonstrate this swarm **LiquidFun** (Google, 2013) was used, which is a particle physics simulator, see Fig. 4.2. It includes mass/weight and collisions, and given the simplicity of the agents nothing more specialist, such as realistic sensing capabilities, was required.

The agents were placed in a square environment of fixed width with no obstacles present, as we are only concerned with the ability to form patterns with the implicit RD system.

At the start of each simulation each agent had a random initial position, velocity, and concentration value for u and v.

4.4.2 Parameters

Tab. 4.1 displays the parameters selected for the following simulations. Averaging rates pertaining to exponential averaging schemes were kept high, i.e. $\alpha_v = 0.9$ and $\alpha_{\hat{u}} = 0.99$. This was to compensate for a potential fluctuation via diffusion whenever a new agent would enter the neighbourhood.

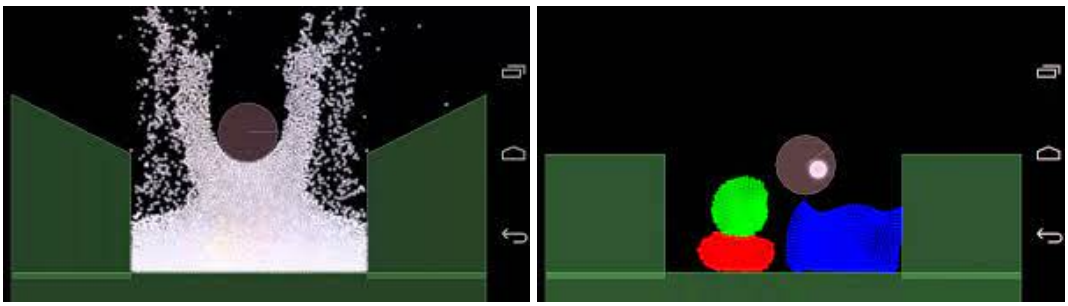


Figure 4.2: Examples of the **LiquidFun** simulator, primarily designed for particle physics in 2D applications (see `https://google.github.io/liquidfun/`). The particles can be subjected to gravity, as well as having different sizes and densities allowing appropriate interactions between different kinds of particles (see right image). The particles can also be grouped together to form complex objects and the connections between the particles can vary from rigid to elastic (see left image). For our setup this simulator provides an appropriate setup as our agents will need to be able to move and undergo collisions with other agents and the surrounding walls.

In the RD simulated system in Sec. 4.2.2, each point was only concerned (or 'communicating') with the four closest neighbours (up, down, left, and right), and increasing this neighbourhood will cause the points to diffuse over a larger area. This would result in the wavelength of the Turing pattern decreasing, i.e. the spots/stripes increasing in size. For computational purposes, the number of agents in the following simulations ranged between 100 and 500, while the number of points in the system space in Fig. 4.1 was 4×10^4. Thus, to be able to observe the Turing pattern as a periodic pattern in this smaller space the wavelength should be reasonably high, but given that the agents strongly rely on communication for guidance the communication range, c, was set to $4r$ meaning agents can communicate with those exactly two agents distant.

4.4.3 Varying diffusivity

To examine the behaviour of the system, simulations were conducted with varying sizes of swarm. It is also important to know if multiple patterns can be attained as discussed in Sec. 4.2.2.

The swarm was initially placed in a environment small enough that the swarm inhabited the entire environment. This essentially removes the ability for the agents to move, and is similar to a grid setup as seen in Sec. 4.2.2. The simulated swarm is able to form Turing patterns, and can vary the pattern by increasing D_u, the diffusivity of u,

Parameter	Value
Agent radius: r	0.1
Communication range: c	0.4
Inhibitor diffusive constant: D_v	1.0
Self-activation rate λ	1.02
Velocity averaging rate: α_v	0.9
Maximum velocity magnitude: $v_{\max}$	0.5
Velocity accelerant: ω	1.1
Heading angle step size: θ	$\frac{1}{6}\pi$
RD step size: α	0.2
RD quadratic presence: α_q	0
Maximum received u memory averaging rate: $\alpha_{\hat{u}}$	0.99

Table 4.1: Parameters used for all simulations presented in Sec. 4.4.3

see Fig. 4.3.

The parameter D_u was varied for the different setups to observe if we could get different patterns; low for spots and high for stripes, similar to that attained for the example system in Sec. 4.2.2. The number of agents was also varied for the two values set for D_u. Each setup was conducted multiple times to acquire an average.

The agents, for low D_u, are able to form spots and maintain moderate stability, see Fig. 4.4 for an example. With high D_u, however, they are not achieving the desired stripe pattern, Fig. 4.5. The average neighbours, velocities, and concentration values of u across the agents and multiple simulations were recorded and are presented in Fig. 4.6. For low D_u, the agents acquire, expectantly, a higher number of neighbours as the swarm size increases. The interesting aspect, however, is that when fewer agents are present, the swarm as a whole remains quite active, and slowly the average number of neighbours increases. These results indicate that for a stable part of the pattern to form it requires a certain number of agents. This is further seen in the small differences in the mean u values and maximal u values, as a swarm of smaller size increases these values, although significantly slower.

Setting D_u to a higher value though did not produce the striped patterns in the physical space, see Fig. 4.5. This is further shown in the repeated simulations as the average velocity for the swarm decreases at a faster rate, see Fig. 4.6.

Since the agents were reacting very quickly with respect to their velocity when

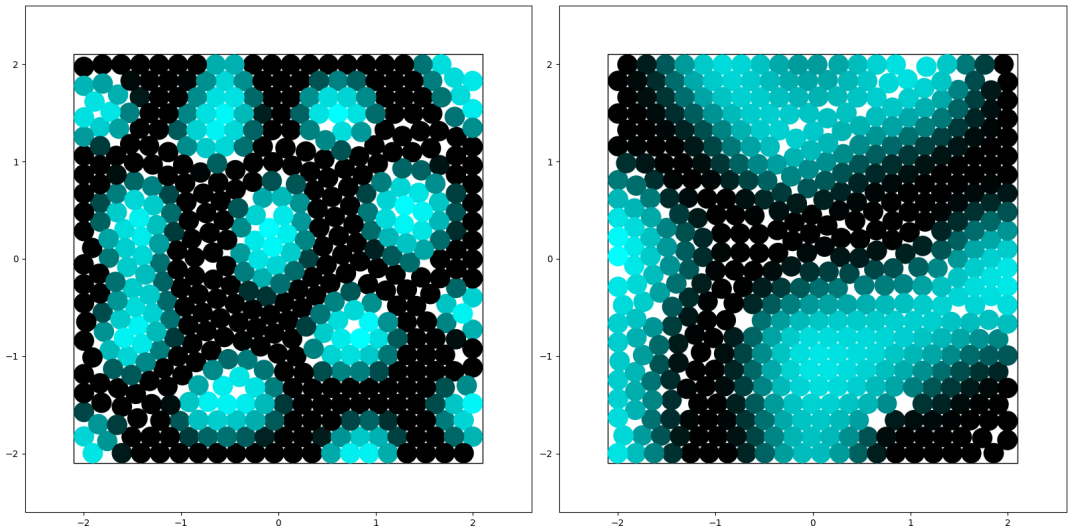


Figure 4.3: Two swarms of 500 agents in a square environment of width 2.2 units. Since each agent has a radius of 0.1 the swarm will inhabit the entire environment (movement not possible), making this setup similar to the grid setup seen in Fig. 4.1, except with a larger area that each agent will diffuse over. The agents in this setup are still communicating their u and v values so that each individual agent can calculate its explicit RD equations. The colours are for display purposes, and are the agents' u values after normalisation. The darker an agent's colour the lower u is, black being near 0 and turquoise being the maximal value. The swarm on the left has $D_u = 0.3$ and $D_u = 0.63$ for the swarm on the right. Both swarms have the same parameters as presented in Table 4.1.

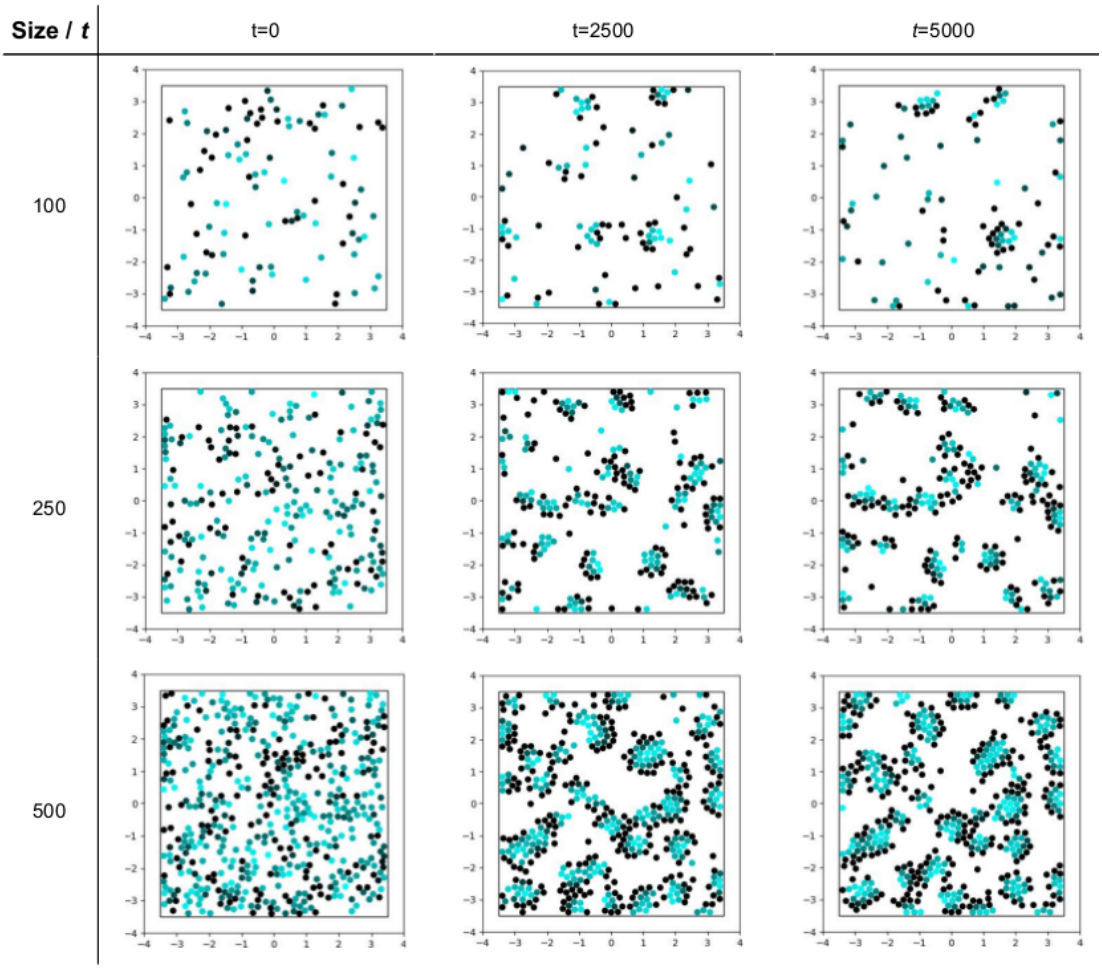


Figure 4.4: Above are the agents' positions for spot formation for varying number of agents. The agents had $D_u = 0.3$, and the other parameters are stated in Table 4.1.

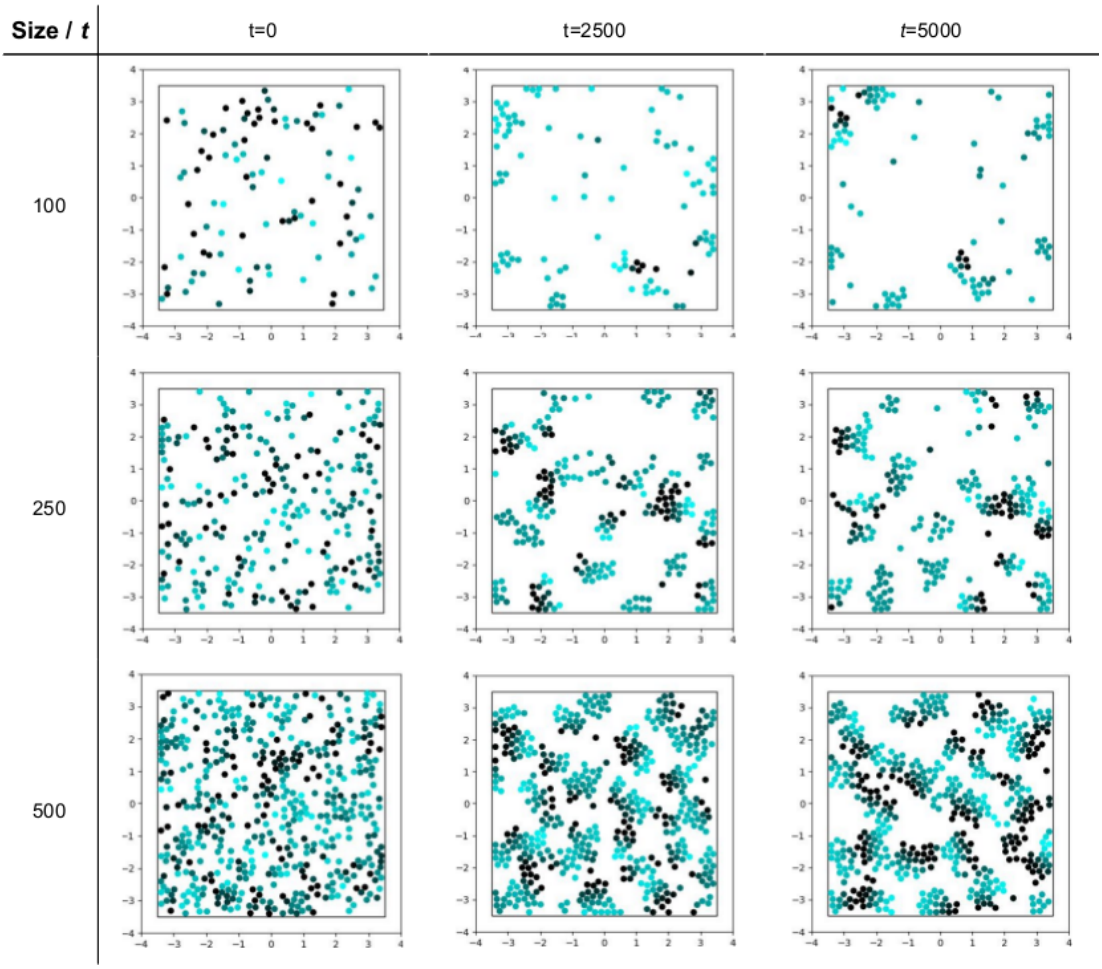


Figure 4.5: Above are the agents positions for attempting stripe formation for varying number of agents. The agents had $D_u = 0.63$, and the other parameters are stated in Table 4.1.

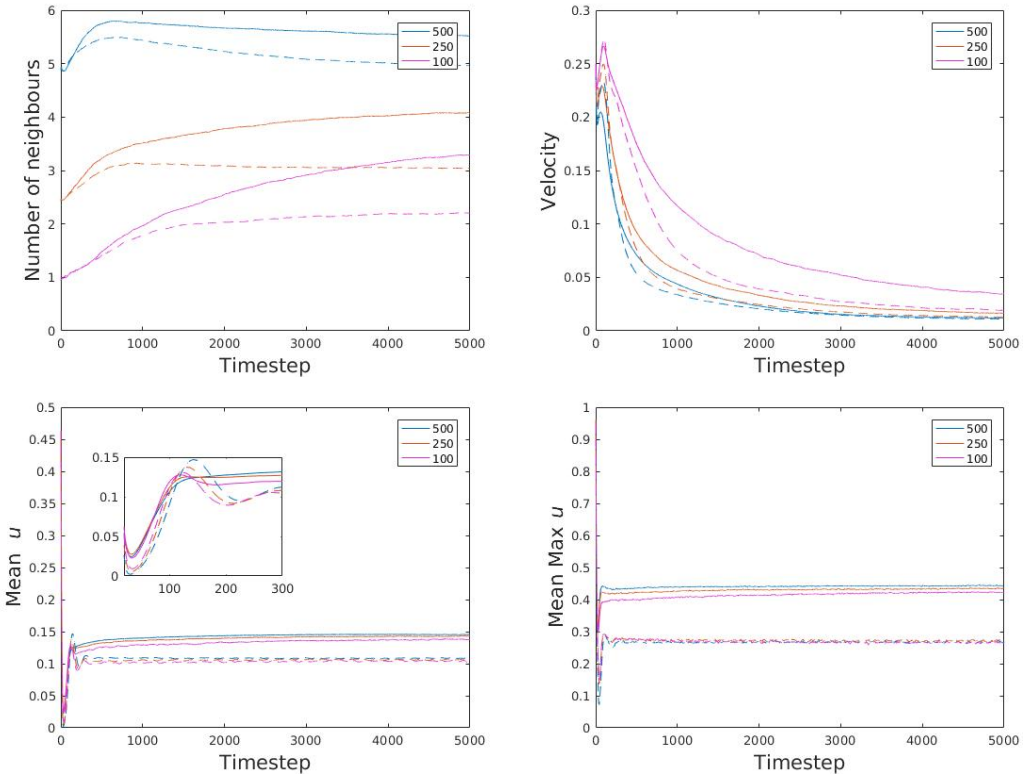


Figure 4.6: Simulations were conducted with $D_u = 0.3$ and $D_u = 0.63$ for various swarm sizes, 500, 250, and 100. For all simulations $\beta = 0$, and the other parameters are stated in Table 4.1. The agents had random initial positions, velocities, u, and v. The square environment has a width of 7.0. Each simulation lasted 5000 timesteps and was repeated 100 times to acquire an average. The top left graph displays the average number of neighbours, the top right the average velocity magnitude, the bottom left is of the average concentration of u, and the bottom right is the maximal concentration of u of the swarm. Solid lines represent simulations where $D_u = 0.3$, and dashed lines for the simulations where $D_u = 0.63$.

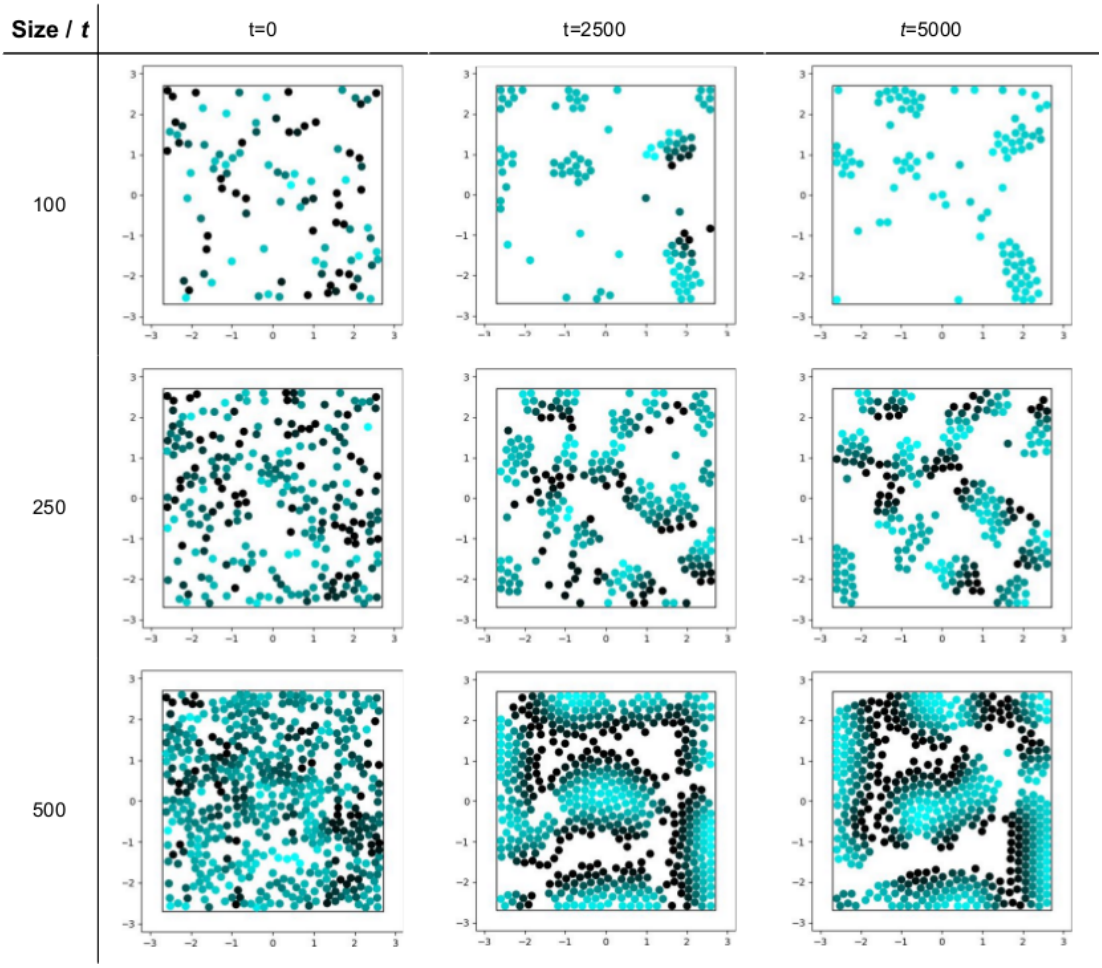


Figure 4.7: The agents positions are under the same settings as in Fig. 4.5 for varying number of agents, however the swarm is in a smaller environment, with the width being 5.4. As the number of agents decreases the stripe pattern is less likely to form.

D_u is set to a higher value, the simulations were conducted again but in a smaller environment. In this smaller setting the swarm was closer to reproducing stripe patterns in the physical space given sufficient swarm size, see Fig. 4.7. Increasing D_u still incurs an increase to the swarm's average deceleration, similar to the larger environment, and does not acquire as many neighbours. For both values of D_u though the average and maximal u increased for the smaller environment, see Fig. 4.8.

4.5 Inducing patterns

The swarm presented in Sec. 4.3 contained agents that could only move and send messages. Even with this setup the swarm can achieve Turing patterns, given enough

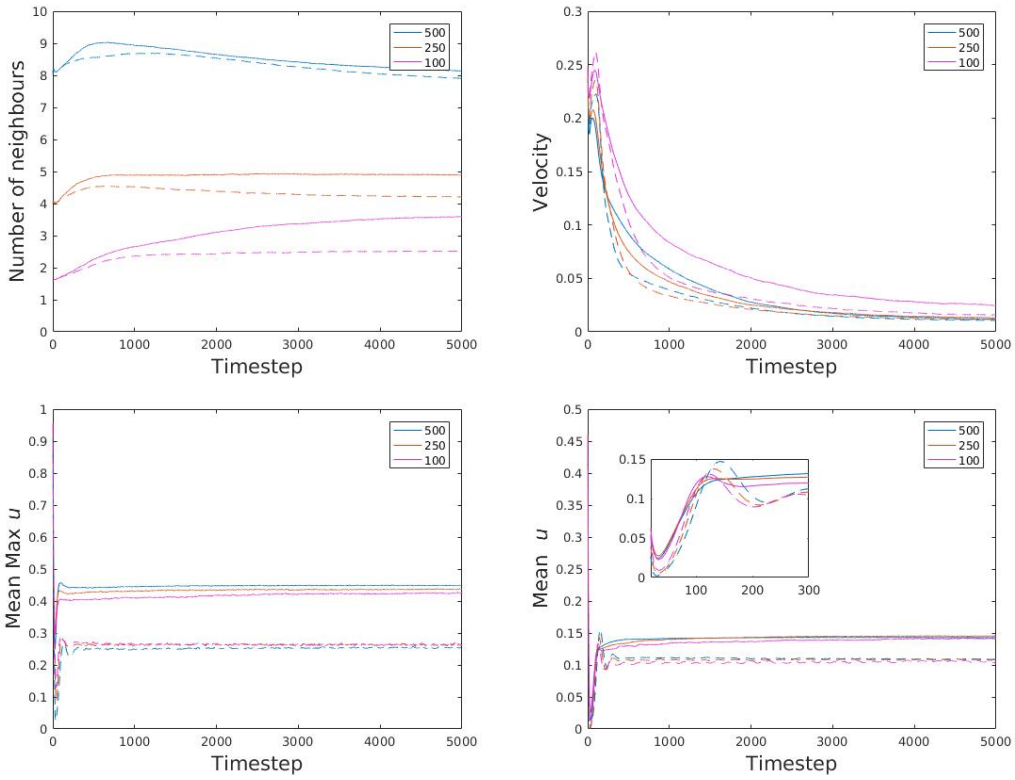


Figure 4.8: The graphs displays the results of conducting the simulations described in Fig. 4.6, however, the square environment had width 5.4.

agents to form the patterns. By using this as a starting building block the swarm can be given further capabilities to be integrated with the RD system.

A common task is to connect two points in space, and swarm robotics can use their physical bodies to connect the two points (Molins et al., 2019). Vázquez-Otero et al. (2012) presented a novel approach in utilising an RD system, using the Fitzhugh-Nagumo system as well for the reaction functions, for path planning in a robot. They defined the inhibition reaction function as

$$\frac{\partial v}{\partial t} = u - v + \beta \tag{4.12}$$

which is identical to the inhibition reaction function in Eq. 4.2 except for the inclusion of the constant β, which is key for connecting points in this algorithm. The algorithm consists of two phases; propagation and contraction. During the propagation phase the algorithm aims to connect the start and goal points by setting $\beta < 0$ which led to a significant increase in the concentration value of u. Once the connection is made, the second phase, contraction, began by having $\beta > 0$, but the regions at the start and goal points were forced to have high u. The algorithm ends when the change is less than a threshold, meaning that all the paths except the one connecting the start and goal points have been deleted. While the algorithm was not able to necessarily produce the shortest route for all the environments, it did however produce the safest route with respect to walls/obstacles. A drawback, however, is the considerable amount of calculations required, which significantly slows the overall completion time.

It is proposed here that an alternative is to have a swarm perform a similar mechanism to connect points in the physical space. By having multiple agents performing their own calculations it should reduce the overall completion time. For this we shall have *points of interest* (POI) that the swarm implicitly aims to connect. The agents will have a simple binary sensor, with range s, which will detect the presence of a POI, and the distance between the agent and POI is the Euclidean distance between the agent's centre and the POI. If the agent does detect a POI then β will be set to a small negative constant. Otherwise $\beta = 0$. The diffusivity, D_u, will be set to a higher value to encourage stripe patterns to connect between the POIs.

Simulations were conducted and the swarm was able to make a bridge between the POIs, see Fig. 4.9. Smaller patterns were forming around the bridge, as to be expected, but the majority of the swarm gravitated towards the connecting bridge. By adding a simple sensor the swarm is able to perform a complex task but still follow the same fundamental behaviour as before.

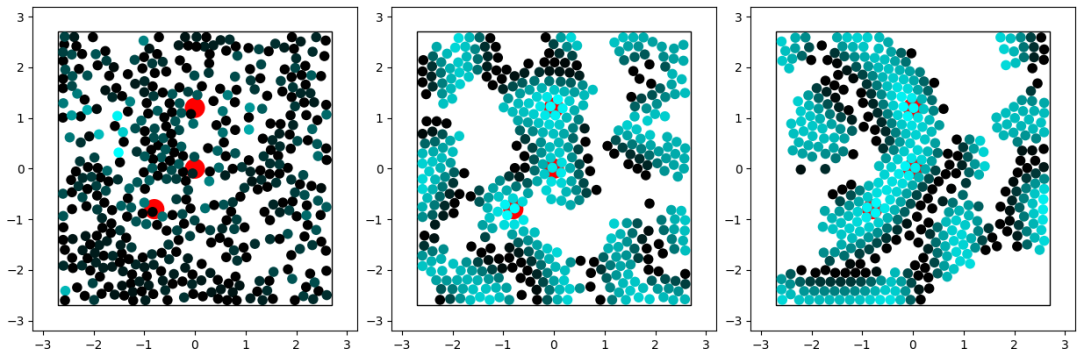


Figure 4.9: The swarm, 450 agents, was in a square environment of width 5.4. The sensor range was $s = 0.2$ and if in range $\beta = -0.05$ else $\beta = 0$. POIs were at (0,0), (-0.8,-0.8), and (0,1.2), and $D_u = 0.5$. It was also desired to have a smaller wavelength to allow clearer observation if the swarm connected the POIs, so $\lambda = 1.05$. All other parameters are the same as in Table 4.1. The images display the swarms behaviour from initialisation to pattern formation. The red circles are the POIs, and the agents can detect said POIs if the centre of the agent is within the circle. The colours of the individual agents are the same as described in Fig. 4.4.

4.6 Turing patterns in Kilobots

This Chapter has thus far demonstrated how a swarm can act as a space for a RD system to operate on, with each agent performing their individual RD calculation. Further yet, even with no sensors the swarm is capable of clustering behaviour, governed by the implicit virtual space to replicate the Turing patterns. Here it is presented that this works in not just the simple simulated agents, but also in realistic simulations of a physical robot platform, in this instance, the Kilobot.

4.6.1 Kilobots

The Kilobot (Rubenstein et al., 2012), is a popular robotic swarm platform which is a three-pronged robot, with the two back prongs mechanically connected to vibrational motors. The Kilobot can sense ambient light by a single sensor, but this will not be used here. Dimidov et al. (2016), for example, used Kilobots to show how diffusive information can assist navigate Kilobots for random walks, and how this approach allows control to improve the exploration behaviour by tuning the parameters to optimise either in an open or a closed environment.

Lastly the Kilobot can send and receive messages by means of IR signals to other

robots in the immediate neighbourhood. The Kilobots reflect IR signals off the floor which is then received by all neighbours in a 10cm radius. When the number of Kilobots attempting to message in the same area increases this in turn increases the likelihood that IR signals will interfere and potentially fail to send.

In addition to the messaging limitation, the messages themselves use limited information, only capable of sending 9 bytes. For the purposes of investigating the minimum information required for the system to operate the Kilobots will only use 1 byte, i.e. only 8 bit per timestep, to implement the diffusive interaction. This means Kilobots can only convey 256 values, [0, 255], thus a mapping is required when sending and receiving potential values

$$\begin{aligned} M_R(\hat{x}) &= \frac{\hat{x}}{255} \\ M_S(x) &= \lceil 255x \rceil \end{aligned} \tag{4.13}$$

where $M_R(\hat{x})$ is a function for converting a received message, $\hat{x}$, and mapping it to the continuous potential space, u and v. The function $M_S(x)$ maps the opposite, taking the Kilobot's current potential value and mapping it to a value that can be encoded for the Kilobot to send as a message.

The Kilobot broadcast at each timestep their u and v values and update their own values depending on the received messages. Similar to before, the Kilobots cannot discern from the message alone where the message is received from, however this is not an issue as shown in the simulated RD swarm setup. Due to limited messaging, each robot will receive a varying number of messages per time step. This is taken into account in the redefinition of the Laplace operator, Eq. 4.1, for a discrete set of robots

$$\triangle u = -Nu_0 + \sum_{n=1}^{N} u_n \tag{4.14}$$

The sum runs over all neighbours of the robot and compares their potential with its own, u_0. Similar to the swarm presented in Sec. 4.3, the operator (4.6) is stochastic, because the robot configuration that sends messages within a time step will typically deviate from a regular grid. It is possible to scale the diffusion constants in order to reduce the effect of the variable distances between neighbours (which is performed in Sec. 4.3.2), but we prefer Eq. 4.14 as it tends to produce a pattern as if the robots were in a regular formation, rather than relative to the embedding space.

The Kilobot swarm is simulated using **ARGoS** (Pinciroli et al., 2011), see Fig. 4.10, using the Kilobot plug-in (Pinciroli et al., 2018) which includes their limited messaging capabilities.

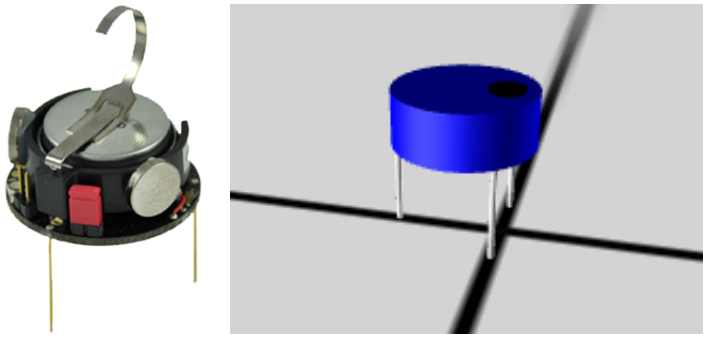


Figure 4.10: A Kilobot and a Kilobot simulated in **ARGoS**.

4.6.2 Scenario Setup

In addition, the RD dynamics needs to be tolerant to discretisation effects. In our **ARGoS** simulation, 225 Kilobots are run, usually in a 15×15 grid, with an initial distance between the robots such that a communication with eight immediate neighbours is possible. A border is present so that when motion is introduced to the Kilobots they do not disperse completely, i.e. given enough time they will encounter another Kilobot. The initial values for the potentials were randomly chosen from [0, 0.1]. Uniform noise with a range [-0.0005, 0.0005] was added to u to escape from chimeric states. Larger noise in combination with the discretisation noise may result in unstructured patterns, or transitions across multiple ground states which may impede subsequent decision making.

4.6.3 Stationary swarm

Fig. 4.11 shows an example run of the set of simulations ran for a stationary swarm, and that a spotted pattern can emerge. Fig. 4.12 displays the location of the Kilobots as well as their activation values. The frequency of the spots in the pattern can be controlled through D_u, which is demonstrated in Fig 4.13. The spots will also remain stable over a long period of time, see Fig. 4.14.

The type of pattern can also be modified, and it is possible to create stripe-like patterns within these simulations. Under the same setup as before but through modifying D_u the system can prefer striped patterns, which is shown in Fig 4.15. It does, however, take a significantly longer time for stable stripes to form in comparison to spot pattern formation. Once formed though, the pattern will remain stable, see Fig. 4.16.

The swarm can also attain different patterns during the same simulation, i.e. changing the parameters in real time can change the pattern. Fig. 4.17 displays two static

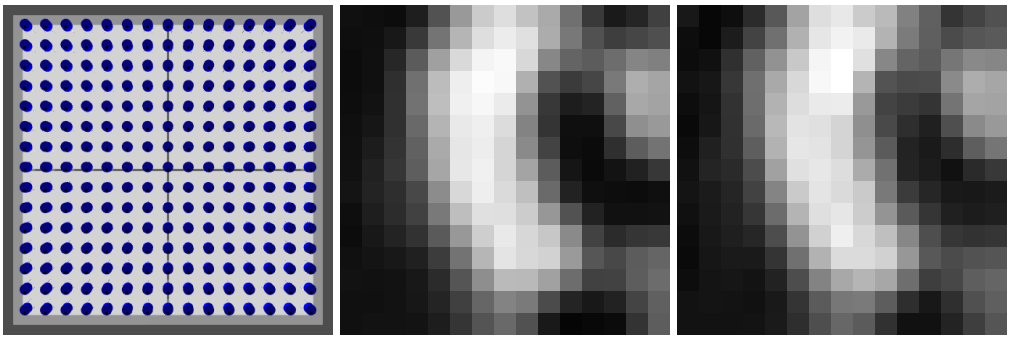


Figure 4.11: The simulation is of the stationary setup described in section 4.6.2. The first image is of the setup itself. The second and third images are the Kilobots' u and v values, respectively, at the end of the simulation. These values were normalised to clearly see the differences between highly and weakly 'activated' Kilobots. In this case the largest values for u and v were, respectively, 0.078 and 0.068. It can be seen quite clearly that there is a single spot, as to be expected since there is the inclusion of a quadratic term which will make the system favour spotted patterns. For this the parameters are $\alpha_q = 1, D_u = 0.5$, $D_v = 1.0$, $\lambda = 1.0$ and $\alpha = 0.2$.

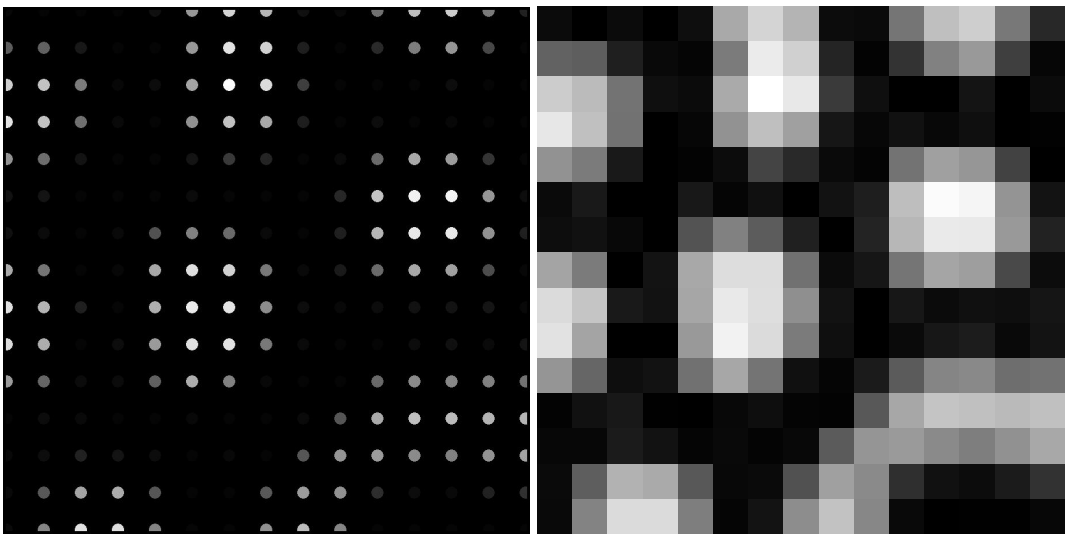


Figure 4.12: Example of a robot configuration in the physical world space. The images on the right are showing each Kilobot's activation and inhibition, respectively, and each circle represents the actual Kilobot's place in the boxed environment, which is shown in Fig. 4.11.

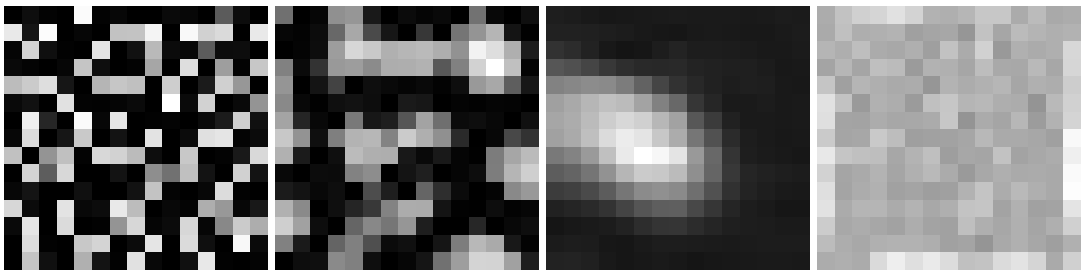


Figure 4.13: The simulations demonstrate the effect of increasing D_u. After the simulations the values were normalised, and the images left of the graph shows the Kilobots' u values. As D_u increases, getting closer to the value of D_v, the spotted pattern becomes less spatially frequent, and also each spot becomes larger in size. The setup is similar to that in Fig. 4.11, except $D_u = 0.1, 0.3, 0.7$ and 0.9 (from left to right).

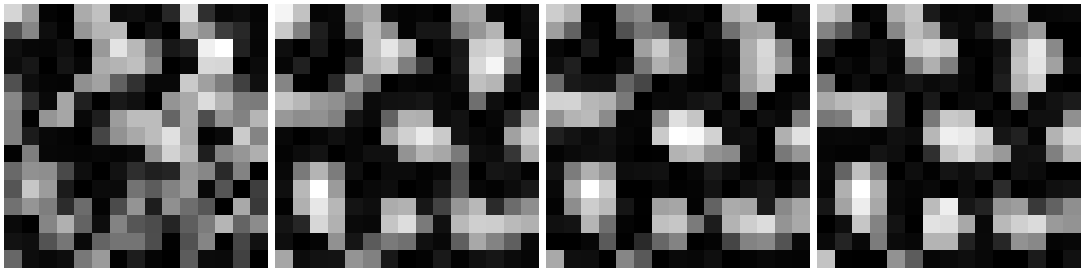


Figure 4.14: Stationary simulation where each image displays the normalised u values. The images are after 100, 400, 700 and 1000 time steps. The evolution of the spotted formations began as a larger spot which over time broke apart to form smaller stable spots. The setup is similar to that in Fig. 4.11 except $D_u = 0.3$.

Kilobot swarms slowly increasing either λ or D_u over time. The Kilobots can change from having no pattern, to spots, to stripes, to inverted spots, all in the same simulation.

4.7 Kilobot guided patterns

Even with the lossy communication, i.e. some messages being lost, and limited messaging size the Kilobot swarm is able to form Turing patterns. From here the Kilobot swarm can now utilise the pattern for achieving spatial tasks.

In principle, two options can be used: The robots can either realise the RD dynamics by estimating the robot concentration from the mutual distances and slow down, or speed up in order to change the pattern. We will follow here instead the simpler option to simulate the RD dynamics by exchanging messages with neighbouring robots to account for the diffusion in Eq. 4.1, while the behavioural consequences become

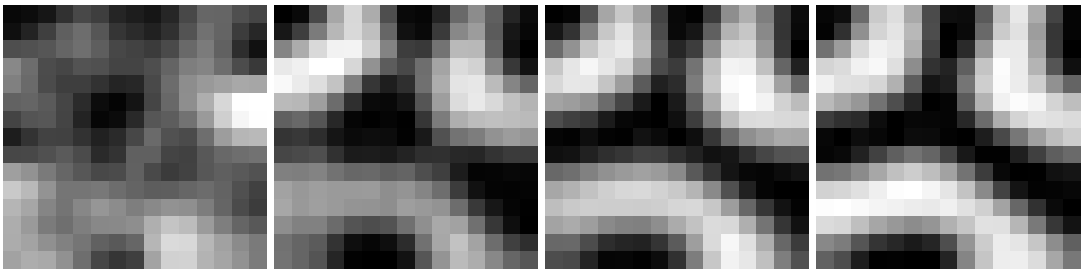


Figure 4.15: The results here show a simulation where $D_u = 0.55$, $D_v = 1.0$, $\Delta t = 0.02$, $\alpha = 0$ and $\lambda = 1$. All other parameters are set as that in Fig. 4.11. The images displays the activation, u, of the simulated Kilobots. From left to right, images show time steps 500, 2000, 3000 and 4000. A pattern containing curved stripes is produced around time step 1000 and is then solidified by time step 4000.

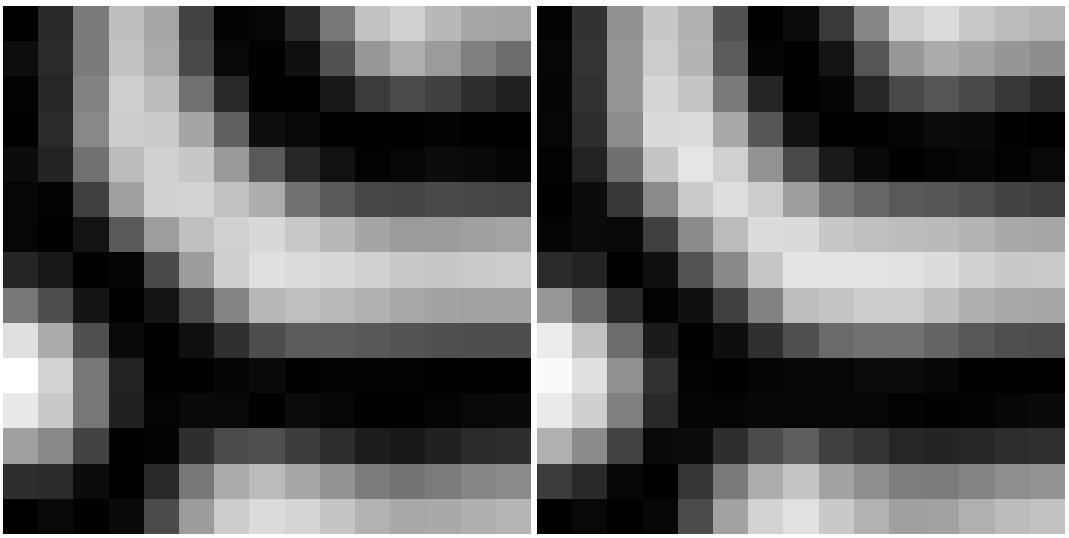


Figure 4.16: A simulation with the same setup as in Fig 4.15 showing the u values after 50000 and 600000 time steps. It can be seen that the pattern, once formed, remains fixed.

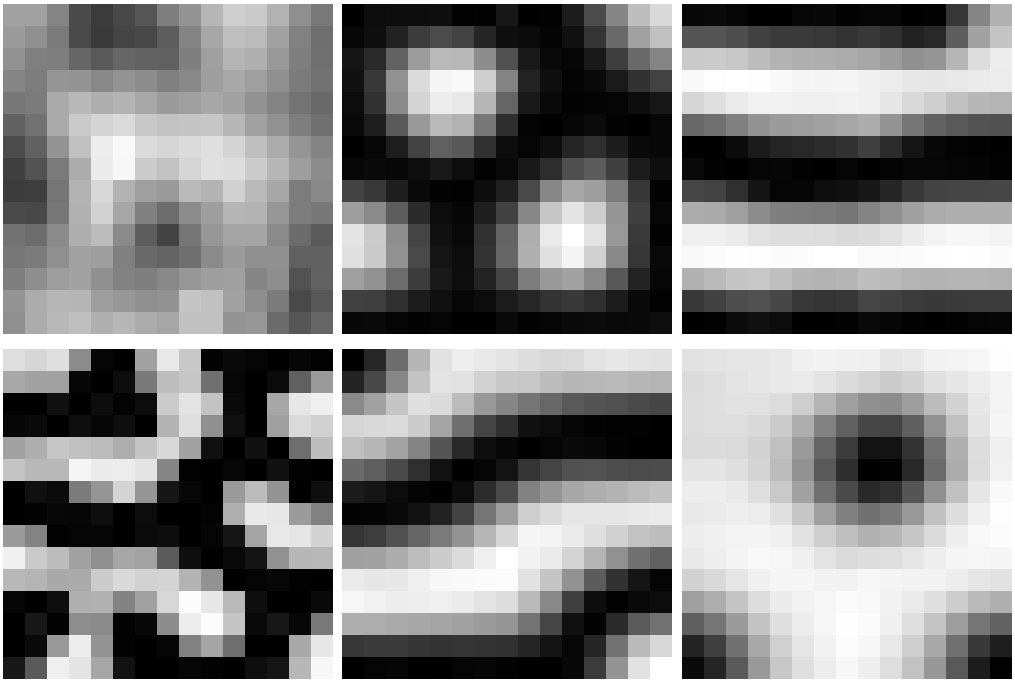


Figure 4.17: A static Kilobot swarm passed messages to each other to replicate an RD system, see Fig. 4.11. Two simulations were conducted which increased either λ or D_u over time. The first row shows the effect of increasing λ, where $\frac{d\lambda}{dt} = 10^{-6}$, and the images are for when $\lambda = 0.906$, $\lambda = 0.954$, and $\lambda = 1.022$ (left to right). The bottom row shows the effect of increasing D_u, where $\frac{dD_u}{dt} = 10^{-6}$, and the images are for when $D_u = 0.317$, $D_u = 0.561$, and $D_u = 0.670$ (left to right).

apparent only after a pattern has formed.

4.7.1 Clustering

A nearly regular configuration of the swarm can be achieved if the robots follow a form of preferential attachment where detachment is also allowed. The system is setup so that it favours spots, and after stable spots are formed a threshold value is used to determine the role of the Kilobot. The Kilobots with high activation will be the centre of these clusters, and the weakly activated will converge towards the highly activated. Let's denote these roles as r_a and r_m, respectively.

They adapt their speed by multiplication dependent upon the number of neighbours they hear from, and their respective roles. The power sent to the motors for r_a and r_m, denoted as p_a and p_m, are updated as follows

$$\begin{aligned} p_a &= p_a a_1^{n_a} a_2^{n_m}, \\ p_m &= p_m m_1^{n_a} m_2^{n_m}, \end{aligned} \tag{4.15}$$

where n_a and n_m are the number of messages received by a Kilobot with the role r_a and r_m respectively, and a_1, a_2, m_1, and m_2 are constants: $a_2 > 1 > a_1 > 0$ and $m_2 > 1 > m_1 > 0$. Random motion is generated for both r_a and r_m. The power can never be below a minimum value, p_{min}. Kilobots will speed up if they do not hear from a r_a, and slow down if they do. If r_m is within a certain distance of a message sent from r_a, then they will change roles to r_a. This will further encourage stable clustering.

The constants in Eq. 4.15 will determine several aspects, such as compactness. The Kilobots have the capability to push a single Kilobot, and so if m_1 is set to be a gradual decline it will promote compactness. The Kilobots will converge on those that are highly active, as they will be surrounded by r_as themselves, and thus will not increase their speed. When Kilobots of role r_m join the clusters and become r_a, then there is more potential for the Kilobots to receive messages from r_a, and remain as a cluster, see Fig. 4.18.

4.7.2 Stripe segmentation

The Turing pattern constructed by the robots can be controlled to some extent. The wavelength of the striped pattern can be modified through the parameter D_u. This would lead to the swarm having the ability to distinctly separate itself into subswarms.

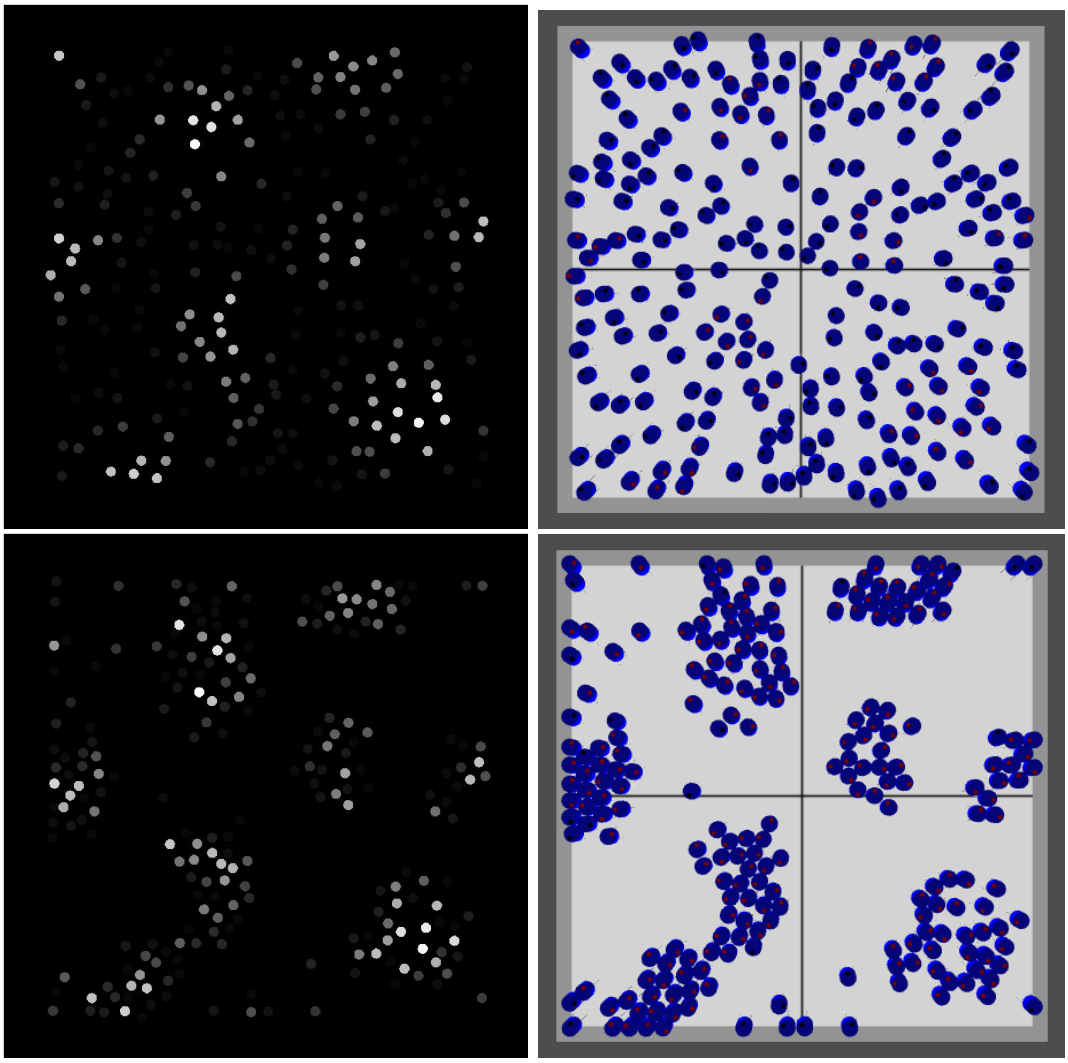


Figure 4.18: The Kilobots were randomly spatially distributed and only sent and received messages for the first 2500 time steps. If a Kilobot's u exceeds a threshold θ, then they are r_a and their LED turns red. Otherwise their role is r_m. Afterwards, the Kilobots will speed up if they do not receive messages from r_a, and slow down otherwise. The top two images display the simulation after 2500 time steps, and the bottom two from time step 4000. The parameters were $D_u = 0.6$, $D_v = 1.0$, $\lambda = 0.97$, $\alpha_q = 1$, $\theta = 0.04$, $p_{min} = 0.01$, $a_1 = 0.1$, $a_2 = 1.1$, $m_1 = 0.9$, $m_2 = 1.1$, and $\alpha = 0.2$.

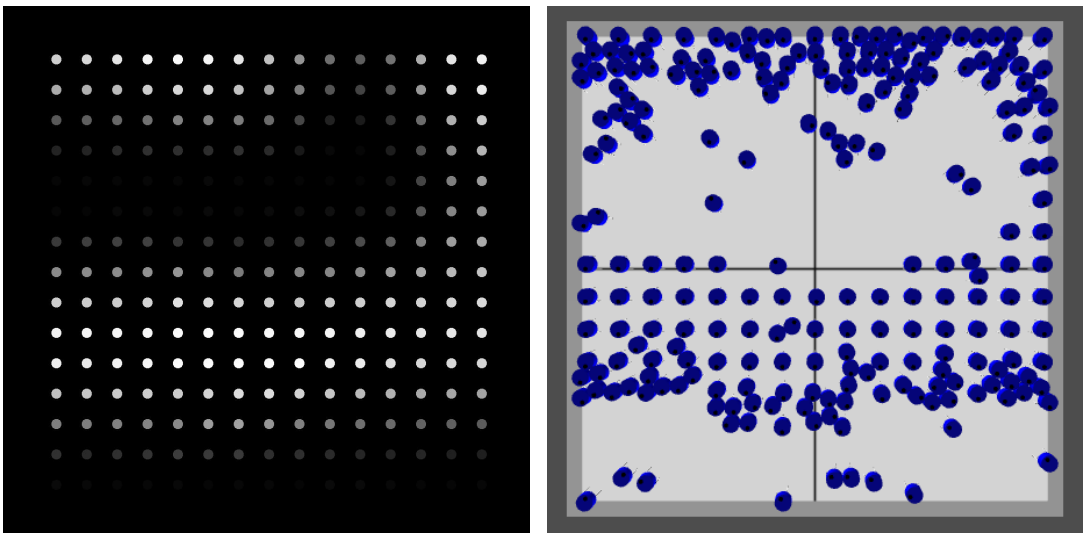


Figure 4.19: The stripes here were formed within 15000 time steps (left). After the formation of the pattern, the Kilobots with low u values randomly moved and would slow down when in contact with other Kilobots, which can be seen after 370 time steps of stripes formation (right). The parameters were similar to that in Fig. 4.18 except $D_u = 0.63$, $\alpha_q = 0$.

In this experiment, D_u was set to form at most two full parallel stripes through the swarm. Kilobots that have activations u below a certain threshold, would begin to randomly move until they arrive near other robots, then they slow down.

Once the stripes are formed the Kilobots will cluster into distinct groups. Fig. 4.19 displays an example simulation of this. In this particular case the stripe was curving, meaning that the other stripe would barely form as it was parallel to this stripe. The initial layout of the Kilobots have them all facing forward. Though the stripes can form in different directions, which will result in different cluster formations, the stripes will always remain still, thus the other Kilobots will tend to stop around them as they will be consistently slowing down within the stripes presence.

4.7.3 Ring formation

It is also possible for the Kilobots, with some success, to form rings based on the information provided by the RD system. Similar to that, in the stripe segmentation procedure, the Kilobots initially act solely as a platform for the RD system. The system is setup so that it favours spots, and after stable spots are formed a threshold value is used to determine the role of the Kilobot. These roles will be denoted, as that in section 4.7.2, r_a and r_m. Kilobots with the role of r_a will form the shape of the ring,

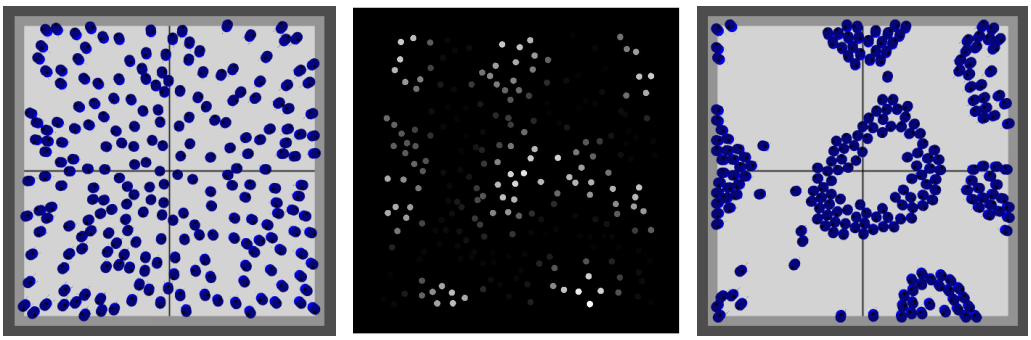


Figure 4.20: The Kilobots were initially randomly spatially distributed and were stationary for the first 10^4 time steps, which is seen in the left image. The spots formed after this period can be seen in the middle. The Kilobots were then given a role dependent upon u. If $u > \theta$ then they are r_a, else r_m. A Kilobot r_m would move randomly and will slow down if they hear from a r_a, and speed up otherwise. r_a will speed up and turn left if they hear from a r_a, and slow down and move forward otherwise. r_a also has a small chance of moving randomly. The parameters were $D_u = 0.55$, $D_v = 1.0$, $\alpha_q = 0$, $\lambda = 0.97$, $\theta = 0.135$, $a_1 = 1.8$, $a_2 = 0.8$, $m_1 = 0.2$, and $m_2 = 1.05$.

and r_m will fill in the gaps.

Random motion is generated for r_m while for r_a there is a probability of 0.2 to perform a random movement, otherwise the movement depends on the messages received. If r_a receives a message from another active robot r_a' then it shall turn left, otherwise it will move forward. Due to the lossy communication, it is unlikely that both of the two active Kilobots will receive a message from each other on the same time step, and thus will result in a repulsion force. They follow the same power updates as that in Eq. 4.15, but here $a_1 > 1 > a_2 > 0$.

The setup described above takes advantage of the spotted pattern created since the spots, i.e. r_a, will immediately repel each other to create a loose ring. The other Kilobots will then, through random movement, converge onto these loose structures. The size of the ring is controlled through a_1 and a_2, as this determines the initial repelling force within each of the spots produced.

The majority of spots will lead to ring formations, though difficulty arises with spots that are present at the borders. These will mostly transform into semi-rings and more cluster-like groupings. Fig. 4.20 displays an example simulation on a randomly spatially distributed swarm.

4.8 Discussion

From the simulations, both on the particle agents from **LiquidFun** and the simulated Kilobot swarm in **ARGoS**, it is shown that it is possible to realise Turing patterns in a swarm. There are aspects of this system to consider, as this Chapter's aim was to present the foundations of an RD swarm which could then, eventually, be expanded to robotic swarm applications.

4.8.1 Varying patterns

From the results presented it is clear that the agents have a difficult time in forming the stripe patterns if there is an insufficient number of agents neighbouring each other. Also, by increasing the diffusivity the swarm's average u value will be closer to the maximal value, meaning that the agents will be more likely to receive these values, which quickens the decrease in velocity, preventing physical pattern formation. One approach that might help overcome this is to reduce the wavelength of the stripe pattern by increasing λ as this will have a higher maximal u present in the system, though this cannot be a significant increase otherwise no Turing pattern will form in the virtual space.

Regarding the maximal potential values, these remained fairly consistent regardless of swarm size and, in a local respective, neighbourhood size. This does indicate that the solutions are stable regardless of swarm size. Furthermore, providing that all the agents in the swarm are using the same parameters then it should be possible for all agents to know what the maximal potential values will be in the RD system. This will then remove the need for the swarm to learn from sampling neighbours, as performed in Eq. 4.10, and incorporate smoother velocity policies.

4.8.2 Optimising velocity policy

The number of neighbours was considered for forming patterns; decrease the velocity, the more neighbours an agent/robot has. This will lead to further attraction between the agents, and the RD system will dictate the clusters' evolution, be it spots or stripes. Furthermore, in the instance of stripes this could lead to a dominant stripe emerging, as the probabilities of remaining stationary can be dependent upon the number of neighbours, as a result of the Matthew effect potentially occurring (Merton, 1968). There is an issue, however, of the concentration of agents no longer representing the concen-

tration of the potential values. Physical clusters may form from sub-clusters where the agents have already formed a Turing pattern, and with the number of neighbours now a component for the deceleration, it means that the sub-clusters meeting point could be where the agents' u values are low, even close to 0, and the agents will then freeze due to the neighbour count.

The clustering behaviour, such as Correll and Martinoli (2011), can be used as a precondition for the simulation, because it can gather all robots to enable a message exchange within the swarm. This would assist greatly with the instances where the swarm size is low and scattered, see Fig. 4.4 and 4.5.

The velocity policy can also be benefited from agents having more capabilities. Allowing the agents to detect the direction of the messages, for example, can lead to a significant improvement by simply having the agents move towards their locally detected maximal u. This also encourages agents to be compact as well.

4.8.3 Simulation to physical platform

This work has demonstrated that not only are Turing patterns possible in a particle swarm, but also in a realistic simulated robot swarm, specifically a Kilobot swarm. Furthermore, the underlying system, the FitzHugh-Nagumo model, did not require an extensive change. The main change briefly investigated was the inclusion of a quadratic term to promote spots (or removed to promote stripes), but this was unnecessary as the other parameters were able to achieve this (see Fig. 4.17). Even with the Kilobot's communication difficulties, pattern formation can be decided upon (spots or stripes) through parameter choice, and how these patterns form, such as the number of spots. The random initialisation of the distribution of u and v amongst the swarm did not cause any issues with the pattern formation; if the parameters were set for a specific pattern then that pattern will emerge regardless of the random initialisation. That said it would be interesting to investigate different forms of initialisation, for example not having a minimum/maximum on u and v, and subsequent investigation into potential chaotic states.

The stripe segmentation used a very basic rule for velocity. It was to show how the stripes can separate a swarm and that the wavelength can be controlled to determine the number of splits. Once the stripes have been formed then it is relatively simple to have control of the swarm by allocating the movement to those below/above a threshold that can be found analytically. The Kilobots can also communicate the maximum value

of the potential during formation of a pattern, and use this information to determine the threshold (e.g. as half the maximum). This can also split them into types which again can be communicated to each other, and thus have more specific segmentation behaviours. For example, all non-active Kilobots should avoid active Kilobots and form their own cluster.

4.8.4 Integration of control

The other formation demonstrated was the ring formation. The borders hinder ring formation, but as stated in section 4.6.2, the borders contain the Kilobots' Brownian motion and eliminate the chance of the swarm dispersing. One solution to this specific situation would be the use of lighting, which the Kilobots could detect and move away from. This would provide a non-obstacle repellent that would keep the Kilobots contained and potentially increase the success of ring formation. Another approach would be to change the Kilobots' motion so that it can home in early on either Kilobots that have the role r_a or have seen r_a. While this is difficult given the lack of directional information, it is possible to gain this information using past data. Interestingly this is where the RD system can provide additional information such as diffusivity, which will allow the Kilobots to move with the diffusion, thus a smoother spatial transition.

For general scenarios a similar approach can be employed as shown in Sec. 4.5 of using a simple binary sensor to manipulate the reaction functions. While it was demonstrated in this Chapter on how to form patterns with the POIs, the detection of a POI could incur an opposite effect, causing $\beta > 0$. POIs would now act as a repellent and instead the swarm would form patterns around the POIs instead of on them. This again does not require the velocity scheme, which is essentially a sampling search, to change.

4.8.5 Robotic platforms

In order to realise the experiments with real Kilobots, finite battery life needs may be problematic. Obtaining straight stripes (Fig. 4.19), would be around two hours real time, which considerably exceeds the typical battery life of 20 minutes, although in most of the performed experiments the robots move only for part of the time.

Therefore other robotic platforms may be preferable. A more capable robotic swarm would also have improved communication abilities, and thus reduce the time of pattern formation due to lossy communication. An interesting platform would be an

aerial drone, as this would increase the physical dimensions, and thus be fascinating to observe whether the RD system would have to be adapted to still attain Turing patterns, or whether it will be able to self organise appropriately.

4.8.6 Applications

The spotted pattern, see Fig. 4.14, can be controlled with respect to the number of spots, and this produces distinct teams within the swarm. This would be ideal for team-based tasks, including swarm exploration as this would allow groups of robots to detach from the swarm and conduct their exploration within their own group.

4.8.7 Criticality

As demonstrated in Fig. 4.17, the swarm is able to change patterns in real time through changing their parameters. Turing patterns require the correct parameters to be chosen to be able to form, thus they too will have a critical point with respect to the particular pattern formed. The critical behaviour in this swarm allows it to easily transition between one pattern to the other, thus increasing the range of information and behavioural capabilities. Understanding the criticality of the system will allow us to design the system to generate more complex reactive behaviours in response to the environment, and provide alerts as to when a transition will occur (Scheffer et al., 2009).

4.9 Conclusion

Swarm robotics is making positive steps, and this is due to the robustness of the underlying low level rules. This Chapter has shown how to take reaction-diffusion, a dynamical system usually applied to modelling natural systems, which produces Turing patterns, and deploy it onto a swarm. The swarm is capable of replicating Turing patterns with simple messages, and due to the diffusion relying only on second order derivatives, which are homogeneous, directionality is not required information. When randomly spatially distributed, the swarm is able to form Turing patterns in the physical space, which is similar to that in an implicit virtual space (which the agents are not aware of). The swarm employs a velocity rule that is reliant upon the maximal u received via messaging. Other than knowing the distance of where the message was sent from there are no other sensors, meaning the agents of the swarm use a guided

Brownian motion to search. At no point does any agent have knowledge of where they are in the world or the number of agents within the swarm.

The RD swarm is able to form spots, however, there is the issue of consistent stripe patterns being constructed if there is an insufficient number of agents. This can be solved by either increasing the number of agents, or increasing the capabilities of the individuals to enforce an early clustering for the RD system to create a striped pattern for the physical agents to follow. This Chapter demonstrates an example of this by displaying an example of inducing patterns with the use of only a single binary sensor.

This investigation also demonstrates an RD system deployed on a realistic Kilobot swarm simulation. Furthermore, they have the capability of creating both spotted and striped patterns with control of how the patterns will be formed. This could lead to applications such as swarm separation and team formation. Through the information from the final pattern that was formed, the Kilobots have the capability to cluster and to form teams and boundaries. Although the Turing patterns offer only a few formation types, we have demonstrated that they offer great potential. Further work will show that a larger manifold of behaviours will be achievable by scheduling or adapting the parameter values or in combination with other techniques.

From here the investigation can begin to look into applications, and the inclusion of higher level forms of information available to the swarm.

Chapter 5

Maximising an evenly spread swarm in an unknown environment

A common task for swarms is to explore an environment, as the large number of agents gives them the advantage of covering a wide area at once. The formation of swarms rely on interactions between neighbouring agents, but while spreading to also maintain close proximity to the other agents in the swarm. This Chapter tackles the issue of having to maximise spreading in the environment, while remaining evenly spread amongst their neighbours using only their neighbours' positions to guide them. The Chapter first defines an energy function forcing the agents to be equidistant and shows this may be used to create a swarm on a 1-dimensional environment. From the eigenvalue analysis, it is known that minimising the energy function will always drive the swarm to an equidistant configuration, which is also demonstrated in simulations. From here, the system can be modified to include fixed boundaries and a limited communication range, and finally adapted to a 2-dimensional setting. Simulations are conducted and display the spreading behaviour in a circular environment in which obstacles may also be present. Finally, the system proposed here does not only allow the agents to spread out for the purposes of coverage, but also to spread evenly around an object of interest, i.e. surround, in what can be an arbitrary shape.

5.1 Introduction

Swarms are collections of robots that, individually, are quite simple, but as an overall system can perform complex tasks. The benefit of multiple robots is the ability to cover a large space, which in turn can help with, for example, exploration, which could be

a vital pre-cursor to carrying out potential future tasks, such as in search and rescue where the environment might be unknown (Michael et al., 2014; Marjovi et al., 2009; San Juan et al., 2018).

The notion of coverage does not necessarily involve simply maximising the exploration of the total environment, but may also be task-dependent, such as encircling an object in which the agents conversely cover the minimal space to complete the task. A good example of the former situation being the safe passage of migrants (Vaidis and Otis, 2020). Here a good coverage of the surroundings by this vulnerable group will provide maximal sensing, and thus ample time to react to danger. Although coverage is a positive aspect, there is a balance that is required for making sure the swarm is still connected, as the swarm could be relying on direct communication.

This Chapter explores an energy function which is created for agents in a swarm to be equidistant, which is first designed for a 1-dimensional swarm, with perfect information, i.e. they know how far away their immediate neighbours are at all times. This starting system is slowly updated to include boundaries, and the agents are then operating with a limited communication range, to which a new condition is given if the agent cannot detect a neighbour on either (or both) side(s). It is then observed that this spreading behaviour could be considered as solving another task, that of covering the perimeter of a circle, i.e. encircling an object. Simulations are then conducted to observe what is understood from the agents' positions from this encircling, and the effects of the swarm size and communication range. This is then transferred over to a 2-dimensional swarm and similar simulations are conducted with comparisons to a basic separation rule used in Boids. Lastly, the swarm adopts the idea of surrounding an object in a 2-dimensional setting with arbitrarily shaped objects.

5.2 Swarm spread

Coverage is a general problem that interests many fields. Jin and Tang (2010) investigated optimal path planning for a farm surface, which requires successful coverage of the farmland and respective fields to best decompose the regions and accurately describe the time (and potential resources) to traverse between these regions and subregions. There is also great interest in sensor networks (Kong et al., 2013) and how to best place each sensor/node with regards to coverage, as this can be used for establishing communication across the environment or for being able to observe and monitor the entire environment. This applies to many environments each with their

own specific challenges such as underwater environments (Sandeep and Kumar, 2017) or uneven/rocky terrain (Liu and Ma, 2011).

There is also the concept of mobile wireless sensor networks, where the network is deployed and attempts to distribute itself accordingly for maximum coverage. Cortes et al. (2004) used Voronoi partitions to solve the task of distributing a collection of vehicles, which can act as a sensor network across a source. In the initial setup, the source produces a gradient that the vehicles can detect, which is essentially a Gaussian density function for the vehicles to distribute over. The authors aim for an equidistant formation over the source as this would be the most effective approach for maximum coverage. To achieve this, they use Voronoi partitions, and the vehicles then find the centre of their partition using Lloyd's algorithm. They continue this process until each vehicle has stabilised, resulting in an equidistant formation. The vehicles compute the Voronoi partitions and the Voronoi cells' centres individually, meaning that this is a decentralised system. This can also work for other pattern formations, such as elliptical density and perimeter functions. The disadvantage, as stated by the authors, is the requirement for the exact calculation of the Voronoi diagrams and respective centres, which can result in slow convergence times. Swarm systems consist of agents that are simple, and calculations such as these will prove difficult. This is particularly true in the field of swarm nanorobotics, as the physical size of the robot limits its processing capability. The calculations would be even more demanding if the nanorobots were operating in complex 3D environments such as the human body for medical applications (Soto et al., 2020; Saadeh and Vyas, 2014).

Swarm techniques have been utilised to help with exploration such as PSO (Kumar et al., 2017). In particular pheromone trails, which are an indirect form of communication, can be used to navigate the best route (Ducatelle et al., 2011), or can also be used as a repellent to force agents away from previously seen areas (Schroeder et al., 2017; Sauter et al., 2005).

A swarm can also be given prior knowledge of the environment to help guide their actions. Correll and Martinoli (2007) designed a swarm with the goal of exploring all the 'blades' using a spanning-tree coverage approach. The robots had prior knowledge of what the environment was, specifically the concept of the 'blades', and were able to use them as vertices to then explore. This is appropriate for inspection tasks such as pipe and bridge maintenance, though, if the environment is unknown then the system should have a general approach, to allow flexibility to the solution.

A good trait to have in exploration is to have minimal overlap of agent coverage,

and this could be achieved by maximally spreading out the agents. Özdemir et al. (2019) employs an evolutionary algorithm for a swarm of robots where the fitness function is defined by how much of the space is occupied by at least one robot. The only information the robot has for its controller is whether it can directly see another robot in its line of sight. The learning process teaches the robot to ensure that it does not, leading to the robots moving away from each other and out of sight. This proves to be a simple yet effective strategy for dispersing the robots and covering as much of the environment as possible.

That being said it would also be beneficial if the individuals in the swarm maintain contact with each other as much as possible. This is because with some robot swarm applications the local or global information might not be available, thus the robots' direct interactions are crucial for information and behavioural changes to propagate through the swarm (Hauert et al., 2009).

Obute et al. (2019) implemented a virtual pheromone scheme to keep a robot swarm close to the nest. The robots would wander from the nest exploring the local environment. The further away from the nest, the higher the chance they would return using a chemotaxis-inspired approach of following the pheromone's gradient. This also worked for a moving nest, with few robots getting lost as the nest travelled.

Unlike the above, the proposed system here is also aiming for equidistant formation, similar to Cortes et al. (2004), as there will only be one type of agent, i.e. a homogeneous swarm. An equidistant formation not only spreads the agents out for coverage, but also keeps the swarm together for communication.

5.3 Energy Function

First, let's define that there are N agents on a one dimensional plane of length L denoted by vector x, which is sorted according to their position. The system of agents can move on this plane denoted by the velocity vector v, and has the update rule

$$x_{t+1} = x_t + v_t \tag{5.1}$$

and x_{t+1} is then resorted.

An equidistant formation can be described by the minimum of

$$E = \sum_{n=1}^{N} \left(\frac{x_{n+1} - x_n}{2} \right)^2 \tag{5.2}$$

with $x_{N+1} \equiv x_1$ and $x_0 \equiv x_N$. The boundary condition for the agents' positions will be a periodic boundary condition

$$f(x_n) = \begin{cases} x_n - L, & \text{if } x \geq L \\ x_n + L, & \text{if } x < 0 \\ x_n, & \text{otherwise} \end{cases} \tag{5.3}$$

To minimise we can take the derivative with respect to the agent's position x_n

$$\begin{aligned} \frac{\partial E}{\partial x_n} &= \left(-\frac{1}{2} \cdot 2 \left(\frac{x_{n+1} - x_n}{2} \right) \right) + \left(\frac{1}{2} \cdot 2 \left(\frac{x_n - x_{n-1}}{2} \right) \right) \\ &= x_n - \frac{x_{n+1} + x_{n-1}}{2} \end{aligned} \tag{5.4}$$

The agents can be viewed as a dynamical system

$$\begin{pmatrix} \Delta x_1(t+1) \\ \vdots \\ \Delta x_{n-1}(t+1) \\ \Delta x_n(t+1) \\ \Delta x_{n+1}(t+1) \\ \vdots \\ \Delta x_N(t+1) \end{pmatrix} = -\alpha \begin{pmatrix} 1 & -\frac{1}{2} & 0 & \cdots & \cdots & 0 & -\frac{1}{2} \\ -\frac{1}{2} & 1 & -\frac{1}{2} & 0 & 0 & \cdots & 0 \\ 0 & \ddots & \ddots & \ddots & \ddots & \ddots & \vdots \\ \vdots & 0 & -\frac{1}{2} & 1 & -\frac{1}{2} & 0 & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & 0 & -\frac{1}{2} & 1 & -\frac{1}{2} \\ -\frac{1}{2} & 0 & \cdots & \cdots & 0 & -\frac{1}{2} & 1 \end{pmatrix} \begin{pmatrix} x_1(t) \\ \vdots \\ x_{n-1}(t) \\ x_n(t) \\ x_{n+1}(t) \\ \vdots \\ x_N(t) \end{pmatrix} \tag{5.5}$$

where α is a step size. To further minimise the distance between neighbours the agent's objective can be the midpoint between its two neighbours

$$E = \sum_{n=1}^{N} \left(x_n - \frac{x_{n+1} + x_{n-1}}{2} \right)^2 \tag{5.6}$$

with again $x_{N+1} \equiv x_1$ and $x_0 \equiv x_N$. Minimising the energy with respect to the agent's position

$$\begin{aligned} \frac{\partial E}{\partial x_n} &= 2 \left(x_n - \frac{x_{n+1} + x_{n-1}}{2} \right) + 2 \left(x_{n-1} - \frac{x_n + x_{n-2}}{2} \right) + 2 \left(x_{n+1} - \frac{x_{n+2} + x_n}{2} \right) \\ &= 3x_n - 2(x_{n-1} - x_{n+1}) + \frac{x_{n-2} + x_{n+2}}{2} \end{aligned} \tag{5.7}$$

with appropriate conventions for x_{n-2} and x_{n+2}. The dynamics is stationary for

$$0 = 3x_n - 2(x_{n-1} + x_{n+1}) + \frac{x_{n-1} + x_{n+1}}{2} \ \forall n$$

Therefore, under gradient descent ($\alpha > 0$ any deviations from the equidistant configuration of positions will obey the relation

$$\begin{pmatrix} \Delta x_1(t+1) \\ \vdots \\ \Delta x_{n-2}(t+1) \\ \Delta x_{n-1}(t+1) \\ \Delta x_n(t+1) \\ \Delta x_{n+1}(t+1) \\ \Delta x_{n+2}(t+1) \\ \vdots \\ \Delta x_N(t+1) \end{pmatrix} = -\alpha \begin{pmatrix} 3 & -2 & \frac{1}{2} & 0 & \cdots & \cdots & 0 & \frac{1}{2} & -2 \\ -2 & 3 & -2 & \frac{1}{2} & 0 & \cdots & \cdots & 0 & \frac{1}{2} \\ \frac{1}{2} & -2 & 3 & -2 & \frac{1}{2} & 0 & \cdots & \cdots & 0 \\ 0 & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & & \vdots \\ \vdots & \ddots & \frac{1}{2} & -2 & 3 & -2 & \frac{1}{2} & \ddots & \vdots \\ \vdots & & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & 0 \\ 0 & \cdots & \cdots & 0 & \frac{1}{2} & -2 & 3 & -2 & \frac{1}{2} \\ \frac{1}{2} & 0 & \cdots & \cdots & 0 & \frac{1}{2} & -2 & 3 & -2 \\ -2 & \frac{1}{2} & 0 & \cdots & \cdots & 0 & \frac{1}{2} & -2 & 3 \end{pmatrix} \begin{pmatrix} x_1(t) \\ \vdots \\ x_{n-2}(t) \\ x_{n-1}(t) \\ x_n(t) \\ x_{n+1}(t) \\ x_{n+2}(t) \\ \vdots \\ x_N(t) \end{pmatrix} \tag{5.8}$$

As the above is stating how the positions are needing to be changed then the derivative of the energy function at timestep t can also be interpreted as a velocity, i.e. $v_{n,t} = \frac{\partial E_t}{\partial x_{n,t}}$. The immediate neighbours act as the initial force while the second neighbours act as the momentum. This function will be the building block for defining the velocity updates for the following setups presented here. It should be noted that this function will be needed to be adapted for specific situations as seen throughout this Chapter, to take into account of neighbours potentially switching places and expanding to a 2-dimensional swarm.

5.3.1 General form of solution

Eigenvalue analysis is generally used for solving systems (Wong et al., 1988), and understanding their convergences. Olfati-Saber et al. (2007) uses eigenvalue analysis for determining when a multi-agent system will reach a consensus, as it treats the system as a networked graph. Given our current system setup, we opted to use a similar approach and also use eigenvalue analysis for determining the behaviour of this system, specifically for determining when the system will converge and under what conditions.

The transition matrix in Eq. 5.8 is a circulant matrix,and thus has the eigenvalues

$$\lambda_j = c_0 + c_{N-1}\omega^j + c_{N-2}\omega^{2j} + \cdots + c_2\omega^{(N-1)j},\ j = 0,\ldots,N-1$$

where $\omega = \exp\left(\frac{2\pi i}{N}\right)$, c_0 denotes the diagonal elements, and c_{k+1} stands for each of the elements in the line parallel to the diagonal and to line with the elements c_k. Here, all c_k are zero, except $c_0 = 3$, $c_1 = c_{N-1} = -2$, and $c_2 = c_{N-2} = \frac{1}{2}$, such that by

$$\begin{aligned}\lambda_j &= 3-2\left(\omega^j+\omega^{(N-1)j}\right)+\frac{1}{2}\left(\omega^{2j}-\omega^{(N-2)j}\right),\ j=0,\ldots,N-1\\ &= 3-4\cos\left(2\pi\frac{j}{N}\right)+\cos\left(4\pi\frac{j}{N}\right)\\ &= 2\left(1-\cos\left(2\pi\frac{j}{N}\right)\right)^2\geq 0\end{aligned}$$

we find that all eigenvalues of the matrix in Eq. 5.8 are positive if $j \neq 0$. This means that the gradient descent with respect to Eq. 5.7 stabilises the evenly spread configuration. As the dynamics is linear, there are no other stable configurations, although practically we need to fix a convention on the numbering of the agent. For large N, some eigenvalues will be near zero and small to fourth order, such that the dynamics are very slow for eigenvalues corresponding to coherent deviations of larger groups of agents from the stationary configuration, i.e. equidistant formation.

Consider for example $N = 12$, and let's assume that six agents are displaced to the right and six to the left, i.e. there is one wider gap in the configuration and, opposite to it, one narrower one. The respective eigenvalue is $2\left(1-\cos\left(2\pi\frac{1}{12}\right)\right) \approx 0.0359$, which implies a significant time scale for the decay towards the stable configuration for a simple system. The learning rate, α in Eq. 5.8, could be optimised in order to compensate some of the effect, however, if this value is too large then the system will not converge to the desired equidistant configuration.

5.4 1-Dimension

Sec. 5.3 has defined an energy function and the stability analysis displays that the swarm will drive towards the stable configuration of equidistant formation. This will now be transferred onto a swarm in a 1-dimensional environment with the system integrated. The swarm will begin with complete information about the environment, and will follow the periodic boundary condition. From here the environment will become bounded and the agents will receive less information in the form of a limited communication range. This will act as the first step towards realising this as a system for physical robots.

5.4.1 Complete information

The system can be described by the following dynamical system

$$v_{t+1} = -(Ax_t + b) \tag{5.9}$$

where A is a $N \times N$ transition matrix, as seen in Eq. 5.8. The agents on the extreme edges will need to incorporate the boundary condition when calculating their velocity. For example, the agent closest to 0 will need to observe the two agents closest to the opposite boundary (i.e. L) as being on the other side, which is done by subtracting L from their positions, and vice versa. This then leads to the vector b being added, to account for the boundary condition

$$b = L\begin{bmatrix} \frac{3}{2} & -\frac{1}{2} & 0 & \dots & 0 & \frac{1}{2} & -\frac{3}{2} \end{bmatrix}^T \tag{5.10}$$

The positions are updated

$$x_{t+1} = x_t + \alpha_v v_t \tag{5.11}$$

where α_v is a step size. The positions obey the periodic boundary in Eq. 5.3, and x is sorted after each timestep to account for this.

Fig. 5.1 shows an example swarm, $N = 10$, representing the above system, and furthermore the swarm achieving equidistant formation. The velocity graph indicates that the swarm performs large initial steps with a rapid decline to decreasingly smaller steps. This will be with the agents on the 'inside' already being near the optimum position with respect to their neighbours, while the 'outer' neighbours are slowly pushing outwards.

As the swarm size increases the time for the swarm to reach the equidistant configuration increases exponentially, see Fig. 5.2. This agrees with the analysis performed in Sec. 5.3.1, specifically when observing the eigenvalues of the system. As mentioned, the step size, α_v, could be increased to speed the overall process, however with these simulations it was found that $\alpha_v > 0.2$ caused the system to be unstable.

Overall the system will converge, though slowly as N increases, and is in agreement with the analysis performed in Sec. 5.3.1.

5.4.2 Bounded environment

With the same system presented in Sec. 5.4.1 the periodic boundary condition will now be replaced with hard environment boundaries. Agents' positions can no longer be

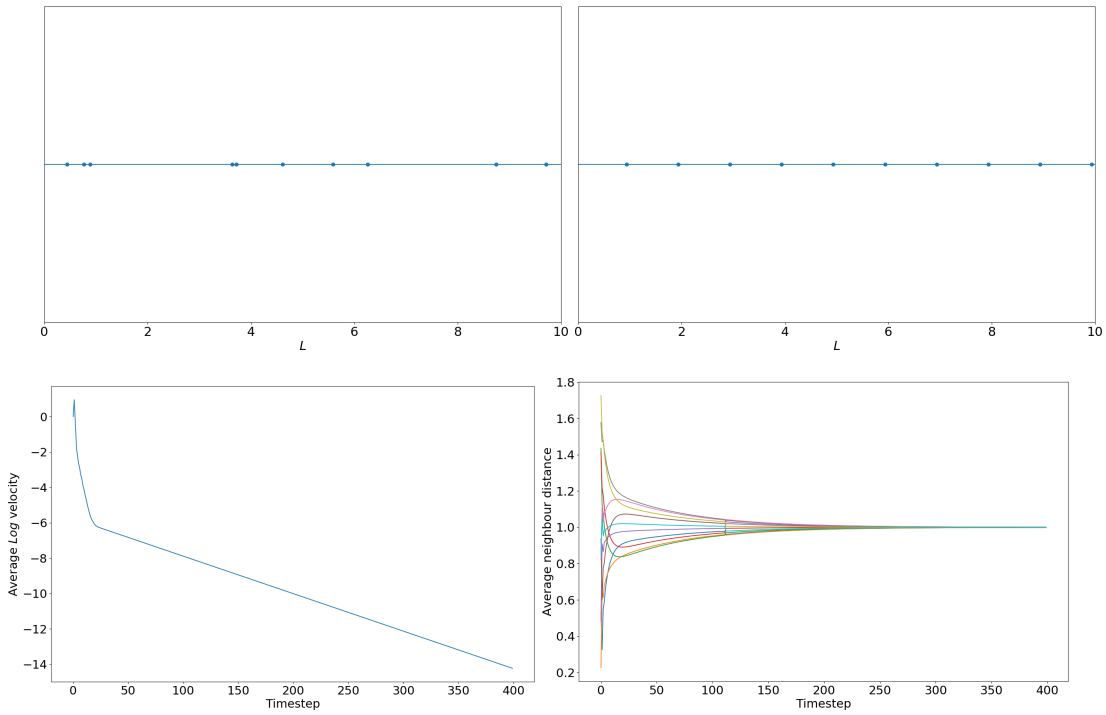


Figure 5.1: An example swarm of 10 agents representing the dynamical system described in Sec. 5.4.1 in a 1-dimensional environment with $L = 10.0$. The top two plots displays the swarm's initial and end positions. The bottom left plot is the averaging log velocity of the swarm, and it is evident that the swarm is approaching 0 velocity, i.e. there is no constant swarm drift even when the equidistant formation has been achieved. It can be calculated that the distance between neighbours should be $\frac{L}{N} = 1.0$. The graph on the bottom right shows each agent's average distance from their two neighbours, and it is displaying that all agents do go towards the equidistant configuration.

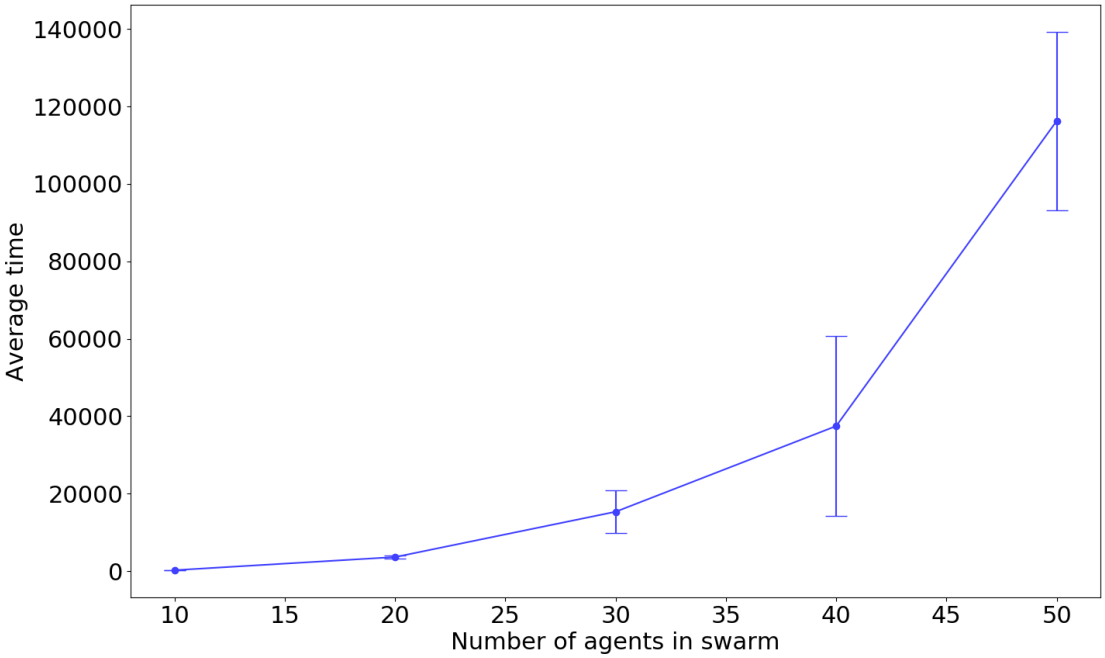


Figure 5.2: Swarms of various sizes using the system presented in Sec. 5.4.1 in an environment where $L = 10.0$. The optimum for the swarm is when the distance between neighbours is $\frac{L}{N}$ and it is considered complete. Intuitively this is when the swarm has evenly divided the environment, with the space between the neighbours being the evenly divided sections. In the case where $N = 5$ this would be when the distance between neighbours for all agents would be 2. For analysis purposes, this will be considered complete when all distances are within 1% of this value. The plot above shows that as the swarm size increases the time for convergence is increasing exponentially. Each simulation was conducted 10 times to acquire an average. The error bars indicate that initialisation plays an important role with regards to the speed of the system.

outwith these boundaries, i.e. $x = 0$ if $x < 0$ and $x = L$ if $x > L$. The agents x_2 and x_{N-1} will detect the boundary and treat them as though they were secondary neighbours. The agents on the extreme edges (x_1 and x_N) will detect the boundaries and treat them like regular primary neighbours. However, as there is no agent beyond the boundary they will create a virtual agent to act as a secondary neighbour. The agent will assume that they, the boundary and the virtual secondary neighbour are equidistant. This means that the secondary neighbour is the same distance from the boundary as the agent is from the boundary, e.g. if $x_N = L - 0.1$ then the first neighbour is the boundary with position L and the virtual secondary neighbour will have position $L + 0.1$.

To accommodate for this the transition matrix is updated with the following changes:

- $A(1) = \begin{pmatrix} 2.5 & -2 & \frac{1}{2} & 0 & \cdots & \cdots & 0 & 0 & 0 \end{pmatrix}$
- $A(2) = \begin{pmatrix} -2 & -3 & -2 & \frac{1}{2} & 0 & \cdots & \cdots & 0 & 0 \end{pmatrix}$
- $A(N-1) = \begin{pmatrix} 0 & 0 & \cdots & \cdots & 0 & \frac{1}{2} & -2 & 3 & -2 \end{pmatrix}$
- $A(N) = \begin{pmatrix} 0 & 0 & 0 & \cdots & \cdots & 0 & \frac{1}{2} & -2 & 2.5 \end{pmatrix}$

Vector b is redefined as well

$$b = L \begin{bmatrix} 0 & \dots & \frac{1}{2} & -1 \end{bmatrix}^T \tag{5.12}$$

Simulating this swarm, see Fig. 5.3, shows a similar behaviour as seen in Fig. 5.2; the time to reach equidistant formation increases exponentially with increasing N. The time, however, with fixed boundaries is significantly longer than with the periodic boundary condition. This is due to the agents on the extreme edges achieving optimum position with the virtual secondary agent already equidistant at all times, meaning the swarm will have a slow filtering process from the start at the swarm's boundaries. The variance however is significant, strongly implying the importance of initialisation; a possibility being the more agents starting nearer to the boundaries, the longer the time to achieve optimal configuration.

5.4.3 Limiting information

The swarm will now only detect neighbours within a certain range, r. This means that the velocity update rule used up to this point, Eq. 5.8, will need to be updated and will

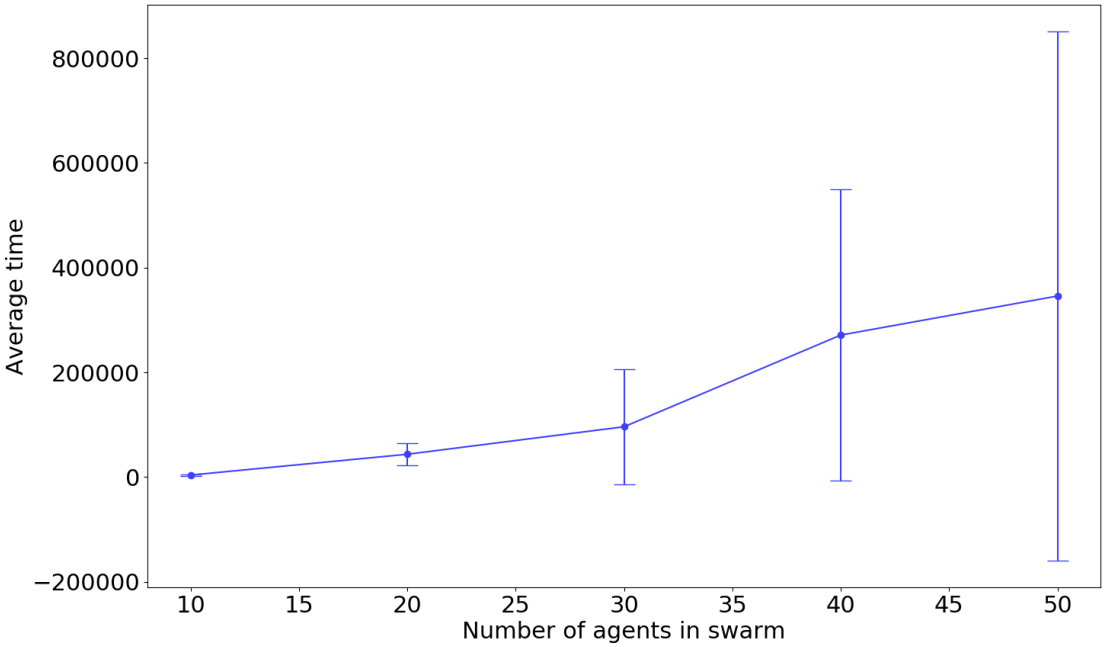


Figure 5.3: Swarms of various sizes similar to the setup in Fig. 5.2 except the boundaries are now fixed and the agents treats them as neighbours as described in Sec. 5.4.2. The swarm follows a similar pattern; as N increases, the time it takes to reach the desired configuration increases exponentially. It is also significantly longer with the introduction of fixed boundaries, however there is a larger variance, signifying further the importance of initialisation.

be performed on each agent locally. The agents will also no longer know what their global position in the environment is, and thus will observe neighbours positions and the boundaries relative to itself. The velocity rule therefore will assume the agent's position to be the origin, i.e. 0, and thereby can use the distances between neighbours for calculating the velocity update

$$v_{n,t+1} = -\alpha_v \left(-2(s_{n,r,t} - s_{n,l,t}) + \frac{s_{n,rr,t} - s_{n,ll,t}}{2} \right) \tag{5.13}$$

where

$$\begin{aligned} s_{n,l} &= x_{n-1} - x_n \\ s_{n,ll} &= x_{n-2} - x_n \\ s_{n,r} &= x_n - x_{n+1} \\ s_{n,rr} &= x_n - x_{n+2} \end{aligned} \tag{5.14}$$

which is the distances between the detected neighbours to the left (s_l, s_{ll}) and right (s_r, s_{rr}). The agents will also have the ability to discern between boundary and agent and apply the appropriate velocity update described earlier, i.e. if the agent detects the boundary as a primary neighbour (s_l or s_r) then the agent will calculate the secondary neighbours' distances in a similar manner as to that presented in Sec. 5.4.2. If the agent does not detect a secondary neighbour, agent or boundary, then the distance is set to the communication range, r, and if the agent does not detect a primary neighbour then both neighbours to that side will be set to r.

From Eq. 5.13 it can be quickly observed that if the agent has no neighbours on either side then $v_{t+1} = 0$. If an agent has neighbours on one side and not the other, the agent will always move away from the detected agent as $|r| \geq |s|$, except for when $r = s$ when the agent will want to stay still. This means that each sub-cluster of agents will expand out as far as their communication range r. Therefore, let's say a swarm begins with N agents clustered together so that it is a connected network, and let's also say that the agents are in an environment of infinite length. The agents on the extreme edges will move away as they do not detect agents on one side, and this in turn will force their neighbours to move to become equidistant in this new configuration. This will continue until all agents are at distance r from their primary neighbours, also meaning that the agents will not have detectable secondary neighbours. This means that the agents cover $r(N+1)$ of space. Thus to adequately cover a finite space of L: $r = \frac{L}{N+1}$. Fig. 5.4 displays example swarms with various $r \in \{0.7\left(\frac{L}{N+1}\right), \left(\frac{L}{N+1}\right), 2\left(\frac{L}{N+1}\right)\}$, i.e. insufficient to cover the environment, adequate to cover the environment, and abundance to cover the environment.

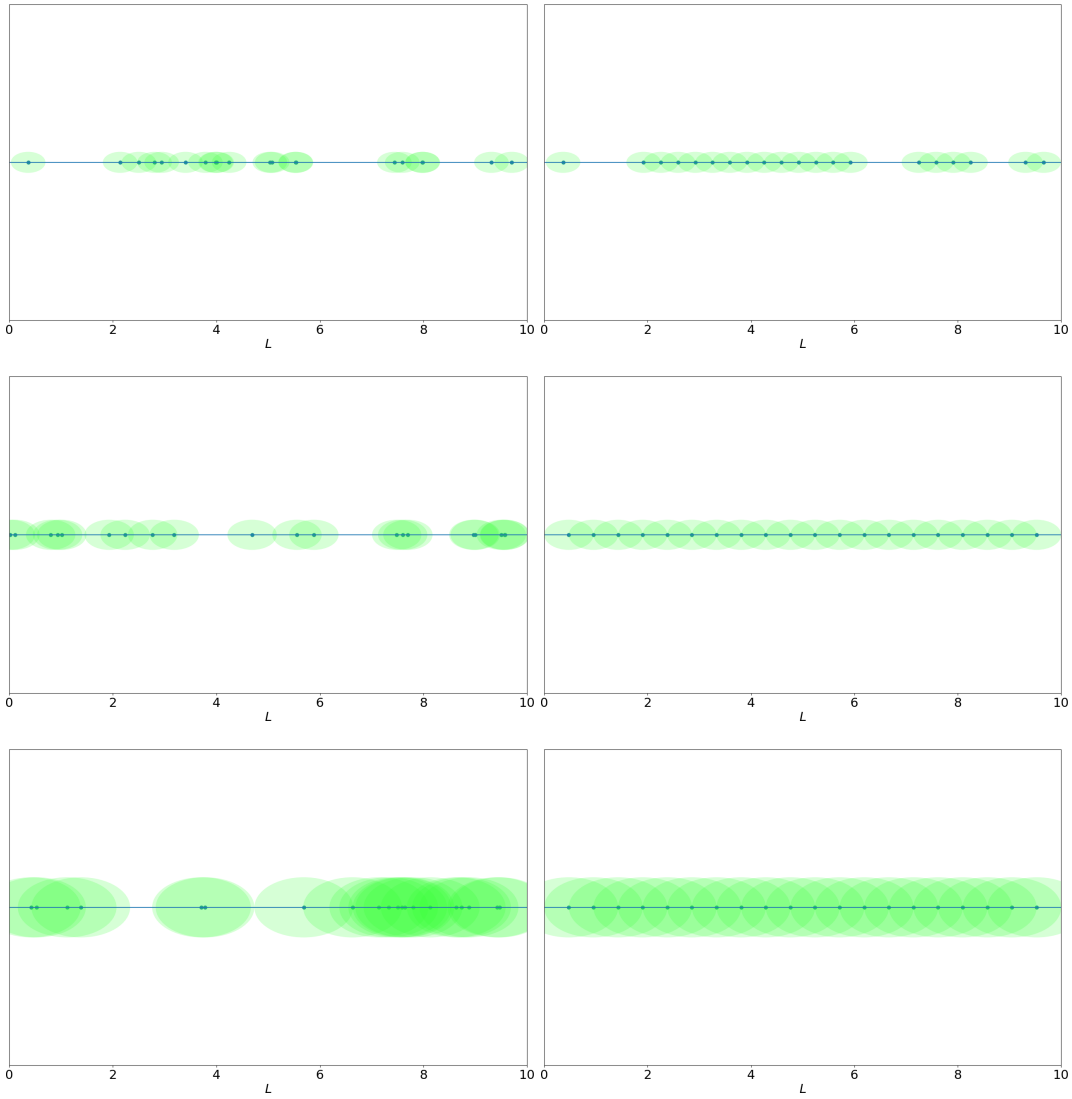


Figure 5.4: Example swarms in a 1-dimensional environment where $N = 20$ and $L = 10$. They have limited range for detecting neighbours and follow the setup presented in Sec. 5.4.3. The agents are uniformly distributed at initialisation, and the left column displays the starting positions, and the right are the final positions, i.e. when the swarm has stopped moving. Each row represents a simulation where the swarms differed with their communication range, r, represented by the faded green circles. Agents (and boundaries) are only detectable if they are within the green circles. From top to bottom; $r \in \{\left(\frac{7}{21}\right), \left(\frac{10}{21}\right), \left(\frac{20}{21}\right)\}$. For the latter two setups they will have identical solutions for the equidistant formation.

If $r < \frac{L}{N+1}$ the swarm will still separate themselves from whichever cluster they are in at initialisation, including boundaries, thereby no agent's communication range is unnecessarily overlapping with another, meaning maximal coverage possible (while still able to communicate).

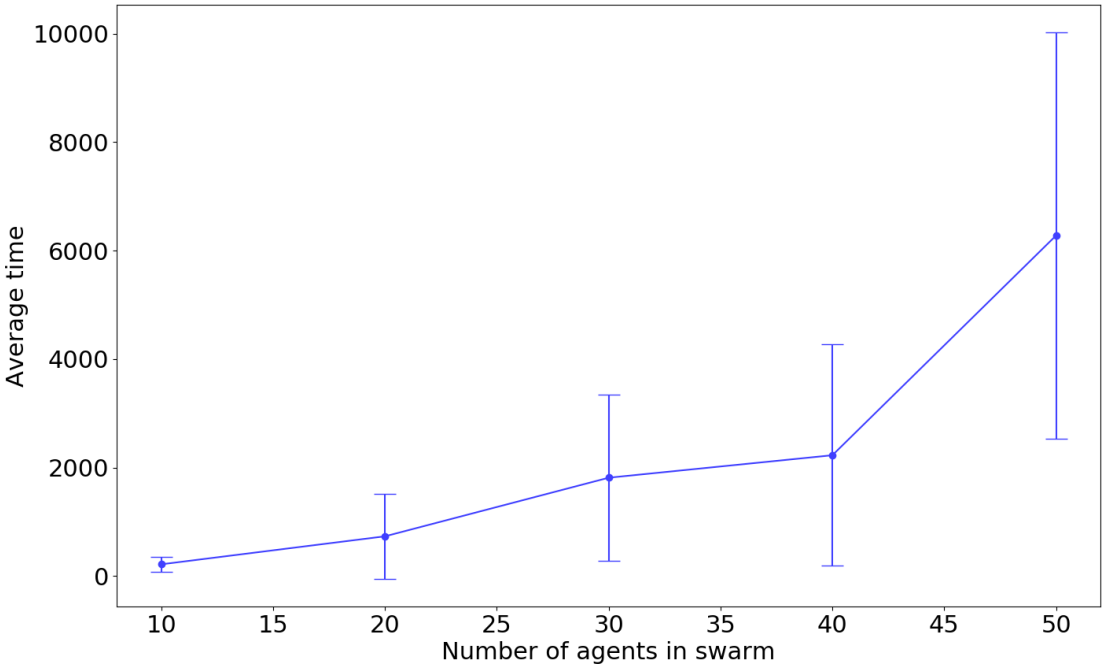
If $r > \frac{L}{N+1}$ then the agents will be closer to what is presented in Sec. 5.4.1, i.e. closer to having perfect information. What is perhaps interesting to note in this case is if the agent is aware of size of the swarm, N, they can determine the length of the environment, L. This can be achieved once the agent has settled and detects no further change to their neighbours' positions, which can act as an indication that the swarm has achieved the desired configuration. From the distance to their neighbours they can calculate the length as $L = s(N+1)$ where s is the distance of either primary neighbour (as they should be nigh identical).

Simulations were conducted with varying the swarm size with the ideal communication range, $r = \frac{L}{N+1}$, and varied r for a swarm where $N = 10$ and $L = 10$, see Fig. 5.5. The time to achieve the equidistant formation follows the same pattern as before; an exponential increase in time with increasing N. The actual times, however, is significantly faster than previous setups with perfect information (see Figs. 5.2 and 5.3). With smaller r they are less likely to detect neighbours on both sides, and the function acts as a smoothing separation, which is faster to perform than finding the midpoint as in this instance the agent has two explicit moving forces. This is further evidenced by the simulations where r was varied and there is a clear increase to the time to achieve equidistant configuration as r increases.

5.4.4 Surrounding in 1-dimension

The environment can be reshaped to be a circle, and the agents of the swarm are fixed to the perimeter of this circle, i.e. still a 1-dimensional spatial environment for the swarm. For all simulations and results presented here the radius of the environment will be 1. The agents' velocity will now update their angle position on the circle. The agents, however, will still use the explicit distances between agents, which now will be a Euclidean distance of their (x, y) position. The agents will still have limited range for communication, see Sec. 5.4.3.

The purpose here is to show an implicit surrounding behaviour which can be integrated when the swarm's dynamics is fully expanded to two dimensions. See Fig. 5.6 for examples.



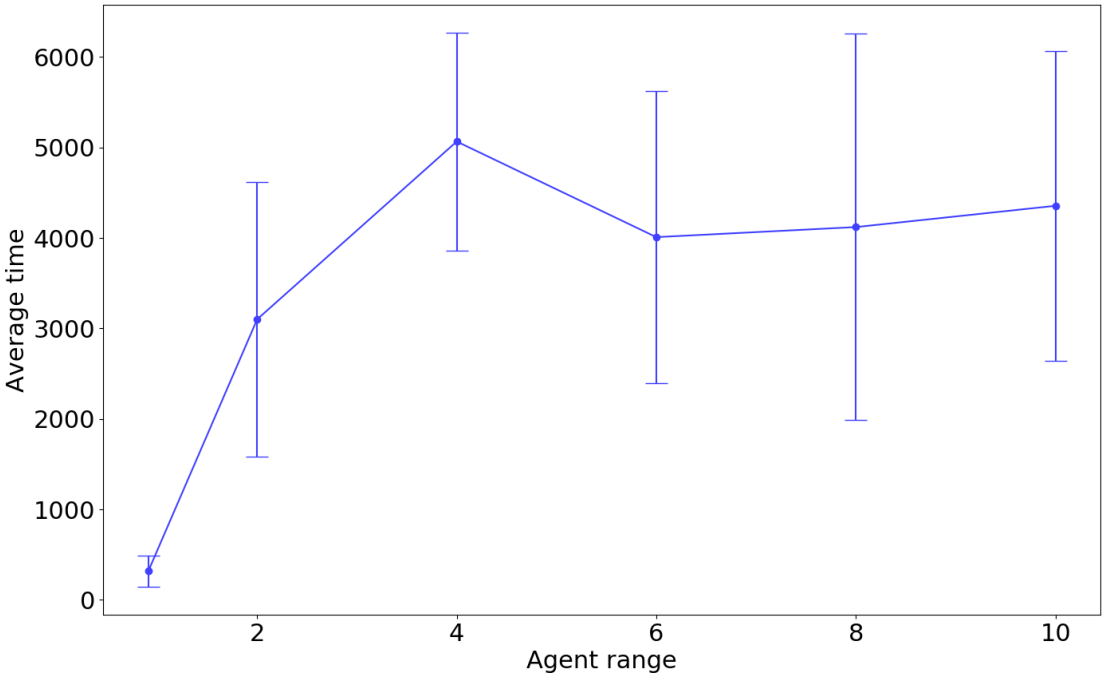


Figure 5.5: Swarms of various sizes were simulated with a limited range of detecting neighbours, be it agents or boundaries, and follow the setup presented in Sec. 5.4.3. The swarm inhabited an environment where $L = 10$, and $r = \frac{10}{N+1}$. The top plot shows the time for all distances to arrive within 1% of the ideal equidistant formation, i.e. all distances are $\frac{10}{N+1}$. The bottom plot is of a swarm with $N = 10$ in an environment of $L = 10$ with various values for r. All setups were conducted 10 times to get the averages presented.

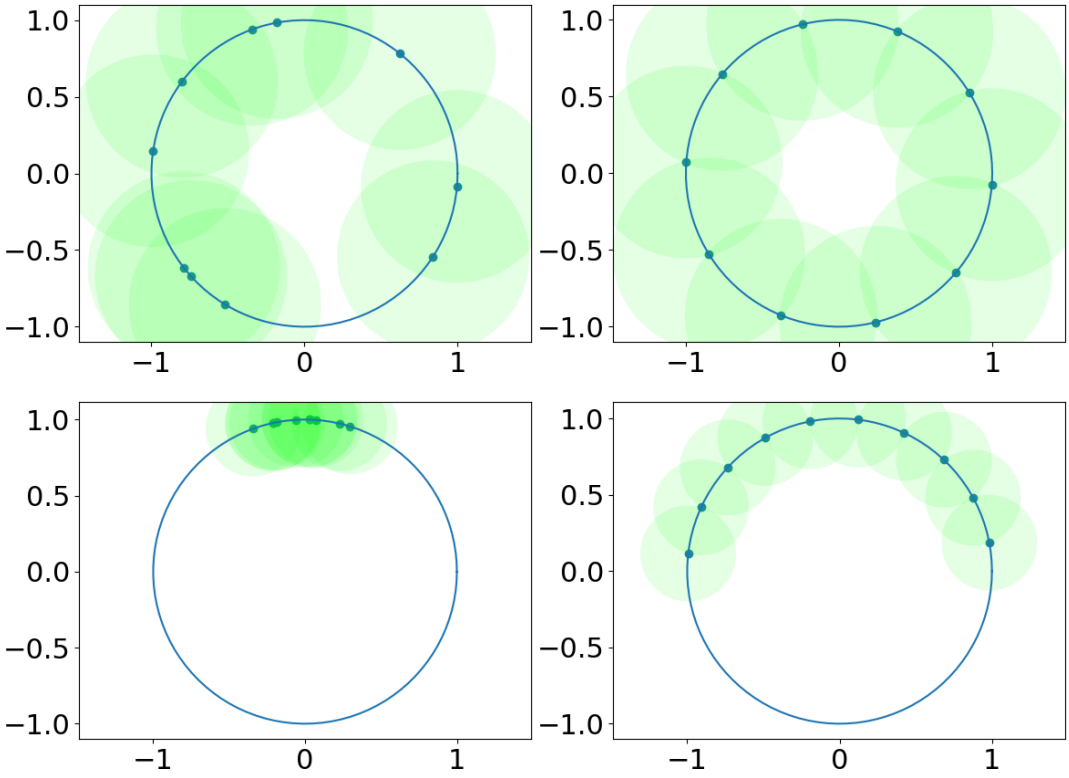


Figure 5.6: Examples of a swarm, $N = 10$, spreading on a perimeter of a circle with radius 1. The top shows the swarm that were initially randomly distributed on the perimeter with $r = \frac{2\pi}{10}$, and the bottom with the swarm clustered initially at the top and spreading as far as possible with $r = \frac{\pi}{10}$.

As before, a desire is to have the swarm adequately cover the environment, which is the circumference of the circle; $C = 2\pi r_c$ in this particular setup (where r_c is the radius of the circle and was agreed earlier $r_c = 1$). Thus $r = \frac{C}{N}$ to cover the space is the assumption (the denominator is N as there is no longer a boundary to consider). However, the agents will be able to detect neighbours through the perimeter of the circle, thus meaning that the communication range is larger than required following the above logic. Furthermore, given the nature of the problem, while the swarm will spread evenly around the environment, they will not fully capture the shape of the environment, see Fig. 5.7, which can be a concern if the aim is to surround an object or determine the shape. As N increases, though, this will alleviate the problem, but overall it does ultimately show that the size of the swarm can be crucial for spatial tasks of this kind.

5.5 2-Dimensions

Here the swarm will now operate in a 2-dimensional closed environment. Firstly, how the velocity rule is updated to account for the increase in number of neighbours an agent can now have, along with dealing with the boundary. This will then be compared with a lite version of Boids, and lastly with the inclusion of obstacles.

5.5.1 Expanding to multiple dimensions

First, it should be noted that the setup will build upon the agents having limited communication range, see Sec. 5.4.3.

In the 1-dimensional setup the agents had opposing forces from both directions which updated their velocity. One way to carry this over is to discretise the communication area in a way that there are opposing forces. In each discretised area the agent will only consider the two closest neighbours and apply their distances (relative to said agent) for calculating that force. This would also consider the boundary and, as will be discussed later, obstacles as well (Sec 5.5.4).

Another approach would be to view the problem in a similar manner but in continuous space. This means that the agent's relative distance to every neighbour within the agent's communication range is viewed as separate 1-dimensional planes. In this continuous approach there will be no 'detectable' second primary neighbour or secondary neighbours, and instead will have the communication range substituted in, as seen in

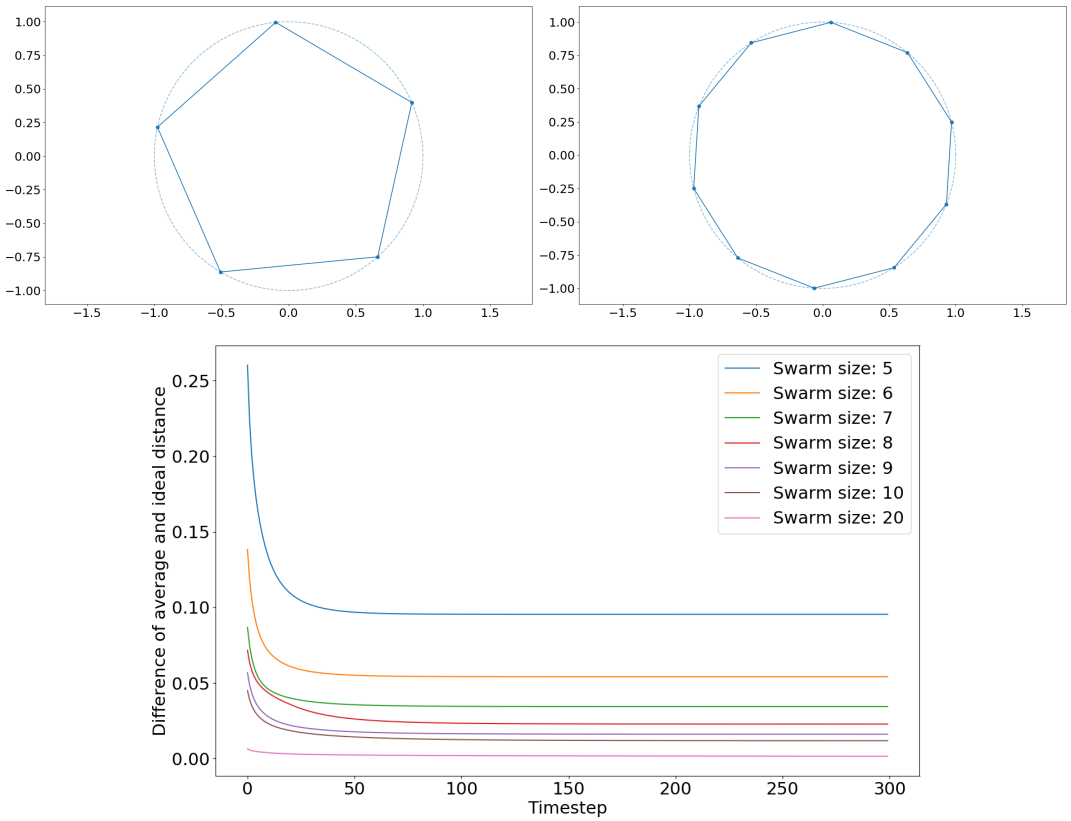


Figure 5.7: Swarms spreading around the perimeter of a circle where $r = \frac{C}{N}$. The top row shows examples of the final configuration of swarms, with the dotted line being the perimeter of the circle and the connected line being an implicit shape derived by the swarm ($N = 5$ for the left and $N = 10$ for the right). The plot on the bottom shows the difference between the average and ideal distance to capture the shape of the circle, and expectantly as N increases the swarm is able to capture more of the desired shape.

Sec 5.4.3. See Fig. 5.8 for illustrative examples of the above two approaches.

A problem with the continuous approach, though, is the boundary is only detected once, rather than the entire length/area of the boundary being detected, thus only one force from the boundary is created that can be opposed by many neighbours. This, therefore, requires an explicit weighting function to balance this. If the space was discretised then there would be no need for a boundary condition as, if the boundary was detected (and it was the only 'neighbour' in that section), it would only have one opposing force generated from a maximum of two neighbours in the opposing direction (remembering that the boundary counts as two neighbours, itself and twice itself).

For this investigation the continuous approach will be implemented, as the discrete approach might not be the most realistic when integrating with robotic applications. Let's say that $\hat{v}_t$ is what the new velocity should be, calculated at timestep t. Given that each 1-dimensional distance plane the agent observes will only have one 'detectable' neighbour, the opposing force will be r and the secondary neighbours will cancel each other out. Thus

$$\begin{aligned}\hat{v}_t &= \sum_{n=1}^{N_r} R(\theta_n)\hat{\mathbf{i}}\left(2(s_n - r) - \frac{r-r}{2}\right) \\ &= 2\sum_{n=1}^{N_r} R(\theta_n)\hat{\mathbf{i}}(s_n - r)\end{aligned} \tag{5.15}$$

where N_r is the number of agents in communication range r and $R(\theta_n)$ is a 2-dimensional rotation matrix which will rotate around the angle between the agent and neighbour n, and $\hat{\mathbf{i}}$ is a unit vector. The difference $(s_n - r)$ is essentially the magnitude of force, hence the need for the rotation to the appropriate direction.

Next the agents will need to consider the boundary. In the continuous approach the agents will only use the closest point of the detectable boundary, as this is most akin to the vast majority of simple sensory approaches on physical robots. This does not reflect, therefore, the total force that the boundary should place upon the agent in comparison to the neighbours. To this end a weighted sum will be used

$$\bar{v}_t = w_b b_t + (1 - w_b)\hat{v}_t \tag{5.16}$$

where w_b is an exponential weight, and b_t is the velocity component from detecting the boundary

$$b_t = -(-2(s_b - r) + \frac{2s_b - r}{2}) \tag{5.17}$$

where s_b is the Euclidean distance to the closest point of the detected boundary. The

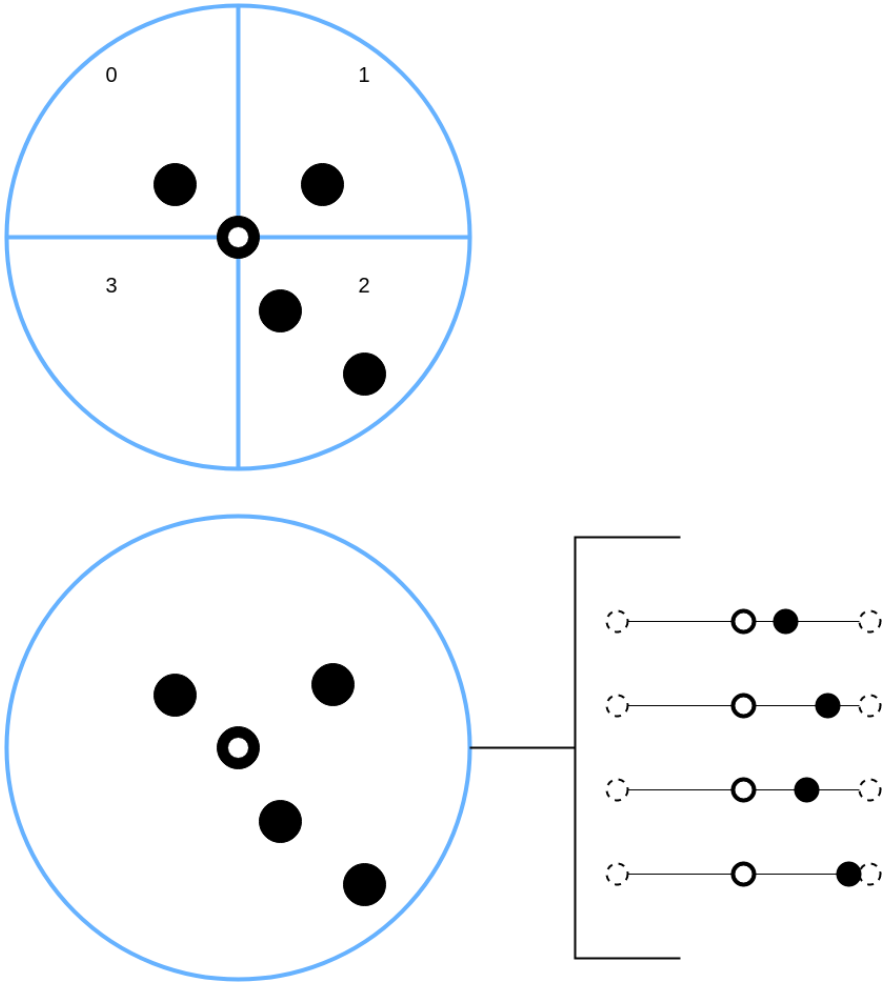


Figure 5.8: Illustrative examples of the approaches discussed when implementing the system for 2-dimensions. The blue circle is representative of the communication range area. The top is if the agent discretises the communication area. Here the agent would have a primary neighbour in area 0 with the value being the distance between the agent and the neighbour, the secondary being set to r as there are no other detectable neighbours. Two neighbours are detected in area 2 and this will be the opposition to area 0. Similar with area 1 and 3 (one primary explicit neighbour in area 1, with all other values set to r. Each force is then rotated appropriately. If more than two neighbours are present in an area then the closest two will be picked for primary and secondary. The area discretisation can be any size as long as it is an even number for direct opposite balance. The bottom graphic illustrates the continuous approach. Every detectable neighbour is translated to a 1-dimensional plane where the distances are the 1-dimensional positions of the primary neighbour. Every other value is set to r (represented by the dashed circles as virtual neighbours on the edges). Again, rotation is appropriately applied and the sum of the forces is used.

weighting, w_b is calculated as

$$w_b = e^{-\alpha^2 N_r} \tag{5.18}$$

The more neighbours an agent has, the closer to the boundary it will prefer to be. The parameter α needs to be chosen beforehand for equidistant formation, which is difficult as the individual agents within the swarm will not necessarily know how many neighbours it will expect to have on average. The new velocity is then updated as an exponential average

$$v_{t+1} = \begin{cases} \alpha_v v_t + (1-\alpha_v)\bar{v_t}, & \text{if boundary detected} \\ \alpha_v v_t + (1-\alpha_v)\hat{v_t}, & \text{otherwise} \end{cases} \tag{5.19}$$

Lastly, to be closer to robots, a maximum velocity magnitude will be implemented, v_{max}, i.e. if $|v_t| > v_{max}$ then $v_t = v_t \cdot \frac{v_{max}}{|v_t|}$.

For all simulations presented in the rest of this Chapter $\alpha_v = 0.8$, $\alpha = 0.4$, and $v_{max} = 0.1$.

5.5.2 Uniformly spreading out in circular environment

The swarm is in a circularly bound 2-dimensional environment with radius 1. Similar to before, to fully cover the environment the agents will need to have a sufficient communication range. Fig. 5.9 shows examples of swarms spreading out if they were clustered in the centre, as well as uniformly randomly spatially distributed. They behave in a similar manner to that observed when the swarm was in 1-dimension.

By increasing the communication range, the swarm increases the spread, see Fig. 5.10 for examples. As the communication range increases then each agent will have more neighbours, meaning they will be pressed further into the boundary. This can be adjusted by increasing α to give more weighting to the boundary, but there must be a balance to have equidistant formation, including with the boundary.

The weighting term is crucial for the swarm to appropriately take into account the boundary. If α is set too low in Eq. 5.18 then the swarm will be significantly repelled by the boundary, resulting in a non-even spread of the agents, clustered in the centre of the environment. If α is set too high then the agents will be pushed onto the boundary as they favour the repulsion of neighbours. Fig. 5.11 contains examples that illustrates this. Regardless, in either case the swarm will still be equidistant from their neighbours where possible.

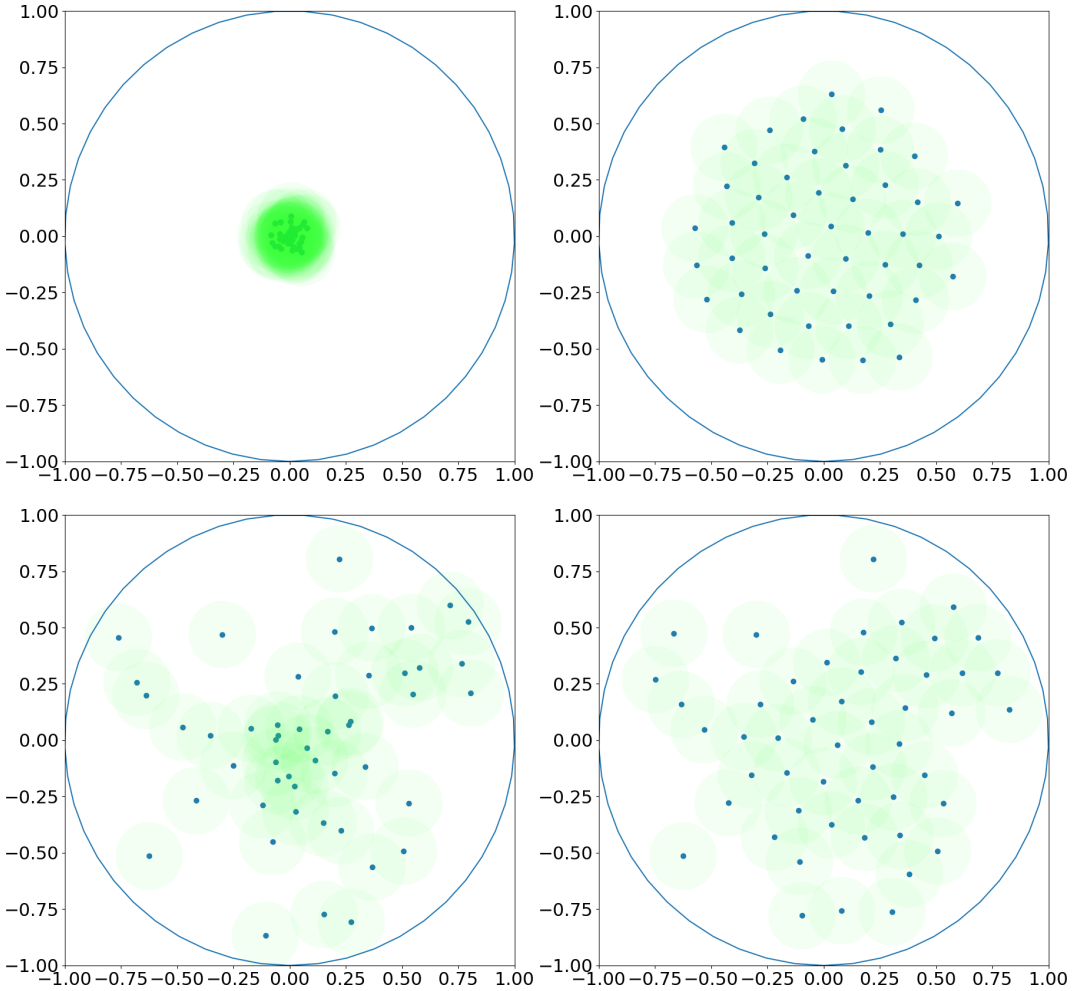


Figure 5.9: A swarm where $N = 50$ and $r = 0.15$ spread out in a 2-dimensional environment. The top row is when the agents are clustered initially in the centre, while the bottom is when the swarm is initialised randomly across the environment. Since r is small the swarm will not cover the whole environment, however, they do spread out as much as possible while maintaining contact with neighbours.

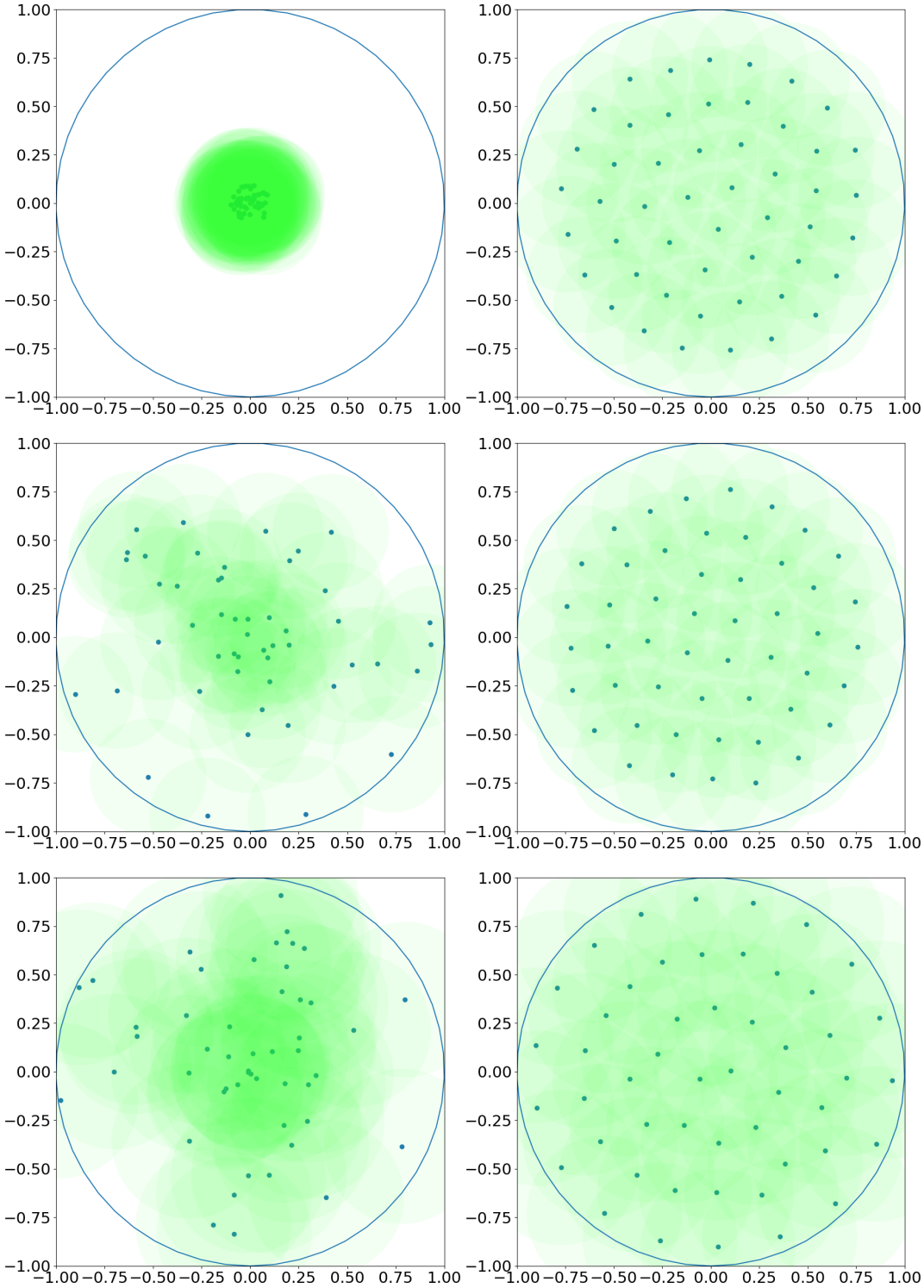


Figure 5.10: Similar to the setup in Fig. 5.9 except for the top two rows $r = 0.3$ and the bottom row $r = 0.4$. If r is large enough the swarm will cover the environment, but as r continues to increase the agents will begin to be pressed into the boundary.

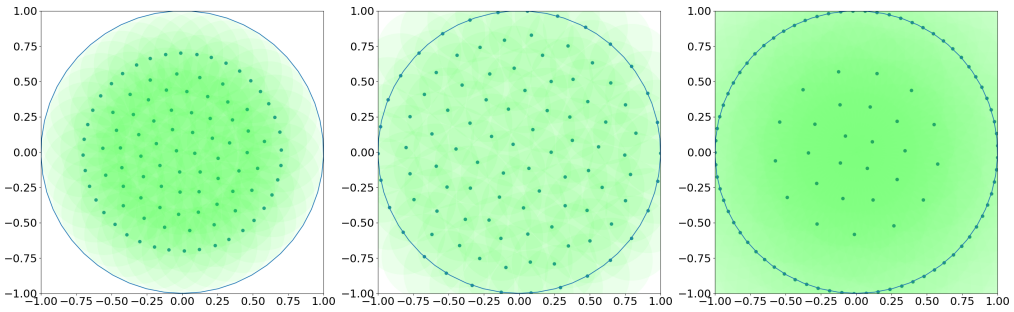


Figure 5.11: Plots of 100 agents performing the spreading behaviour similar to that in Fig. 5.10. Here, however, the parameter α in the weighting function (Eq. 5.18) is varied to show how the system can fail to spread the agents into an even distribution, even if the communication range is large enough to cover the entire environment, like that observed in Fig. 5.10. In the left image, $\alpha = 0.1$, meaning that the boundary will have a greater influence than the neighbours, is set too small, resulting in an outer layer of agents preventing the rest of the swarm from spreading. The agents inside this ring still remain equidistant from each other. If α is set too high, observed in the middle image where $\alpha = 0.8$, then the neighbours will have a greater influence and push the outer most layer of agents onto the boundary. To illustrate this issue further, say if the agents' communication range was increased with a high α then more agents will be pushed to the boundary and act as significant repulsive force, containing a sub-swarm. The agents in this sub-swarm will still be equidistant to each other, like in the other cases. The communication ranges for the left and middle simulations were $r = 0.3$, and $r = 1.0$ for the right simulation.

Simulations were produced of swarms of various sizes clustered in the centre to observe how much area coverage changes with different values of N, see Fig. 5.12 for more details on the setup. As the swarm size increases the area covered also increases, as expected. What is interesting, however, is that the timecourse of swarms is quite similar with when they converge, implying that the system does not slow dramatically as N increases. Also the final area covered is linear as N increases, allowing robust predictability if deploying this onto a robot swarm.

5.5.3 Comparing to Boids lite

As seen in Sec. 2.3.5, the Boids algorithm generates flocking behaviour within a swarm using three simple components for updating an agent's velocity; separation, cohesion, and alignment. A Boids system will be implemented for the setup presented here to act as a comparison to the system described. As separation is a prime objective, the Boids swarm will only utilise the separation function, and will be referred to as Boids lite for the rest of this Chapter. The separation rule, with respect to the agent, is

$$v_s = -\sum_{n=1}^{N_r} R(\theta_n)\hat{\mathbf{i}}s_n \tag{5.20}$$

where s_n is the Euclidean distance between the agent and neighbour n, which is then rotated via the 2-dimensional rotation matrix $R(\theta_n)$ around the angle between the agent and neighbour n. The Boids lite swarm will follow the same weighting function as the energy function swarm when detecting the boundary, see Eq. 5.16.

Simulations were conducted with Boids lite and the presented energy function with identical swarm size starting in the centre of the environment, see Fig. 5.13. Let's first look at the instance where communication range, r, is large enough so that the swarms spread fully across the environment. The swarm utilising the energy function will cover the area in a quicker time than the Boids lite. This could be a result of the boundary condition which the Boids lite swarm struggles with.

Let's now observe when r is small, meaning that at least the swarm utilising energy function will not cover the whole environment, an example displayed in Fig. 5.9. The Boids lite swarm will cover the same area in a similar time as the swarm with the energy function, however the Boids lite swarm will cover significantly more of the environment (Fig. 5.13). The outward force on the Boids lite swarm, caused by moving away from other agents, does not have a virtual neighbour to act against it, as is the case in the energy function (which implicitly acts as a way to find the midpoint). This will

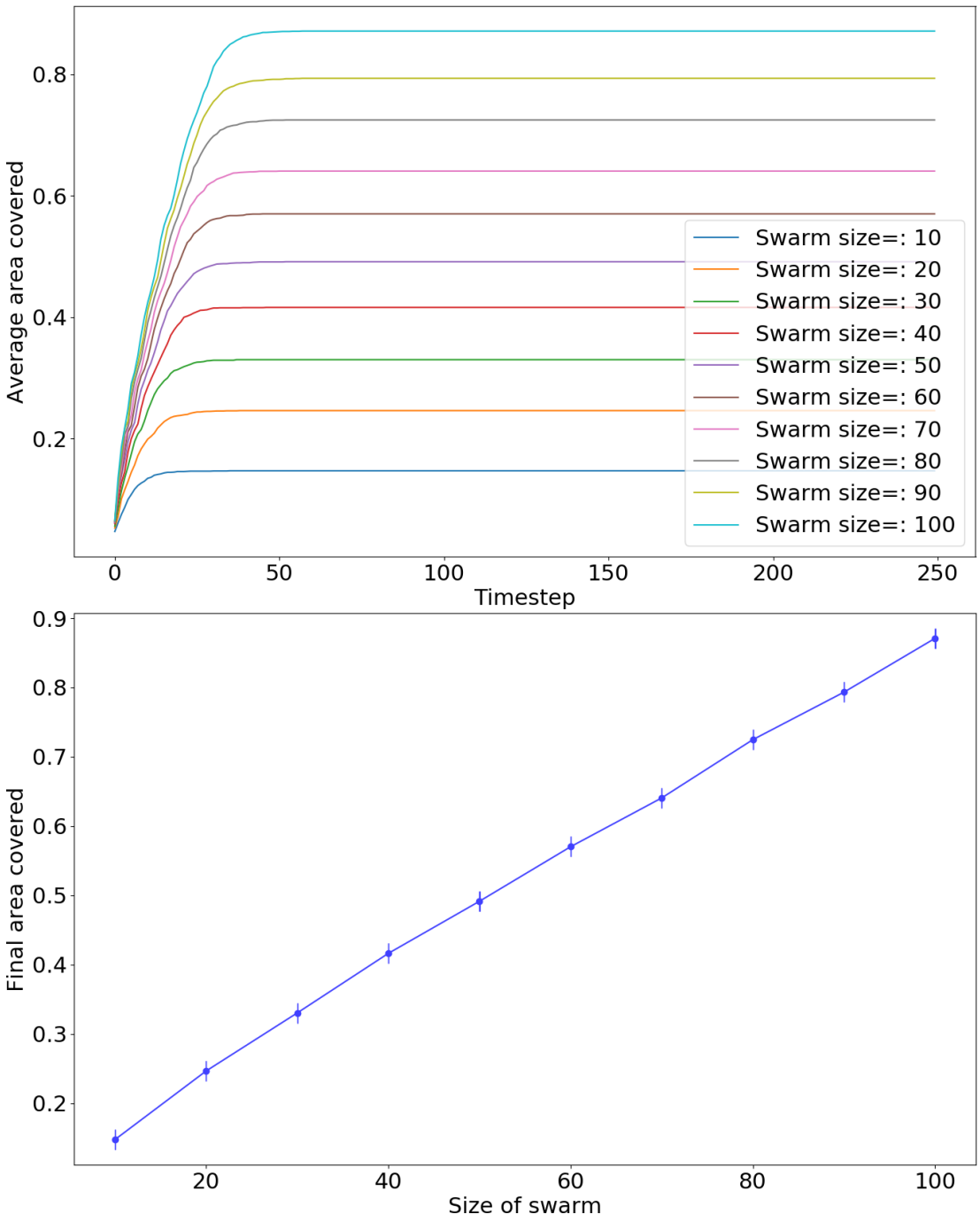


Figure 5.12: Swarms of various sizes initialised clustered in the centre, i.e. randomly spatially distributed within a small circle, radius being 0.1, in the centre of the environment which is a circular environment with radius 1. All swarms had the same communication range of $r = 0.15$. The area covered was measured by what area of the environment was within an agent's communication range at each timestep, which is measured between 0 (no coverage) to 1 (full coverage, all of the environment is observed by at least one agent). The top graph shows the average timecourse for each swarm, and the bottom shows the average final area covered with error bars. Each simulation was conducted 10 times to acquire an average.

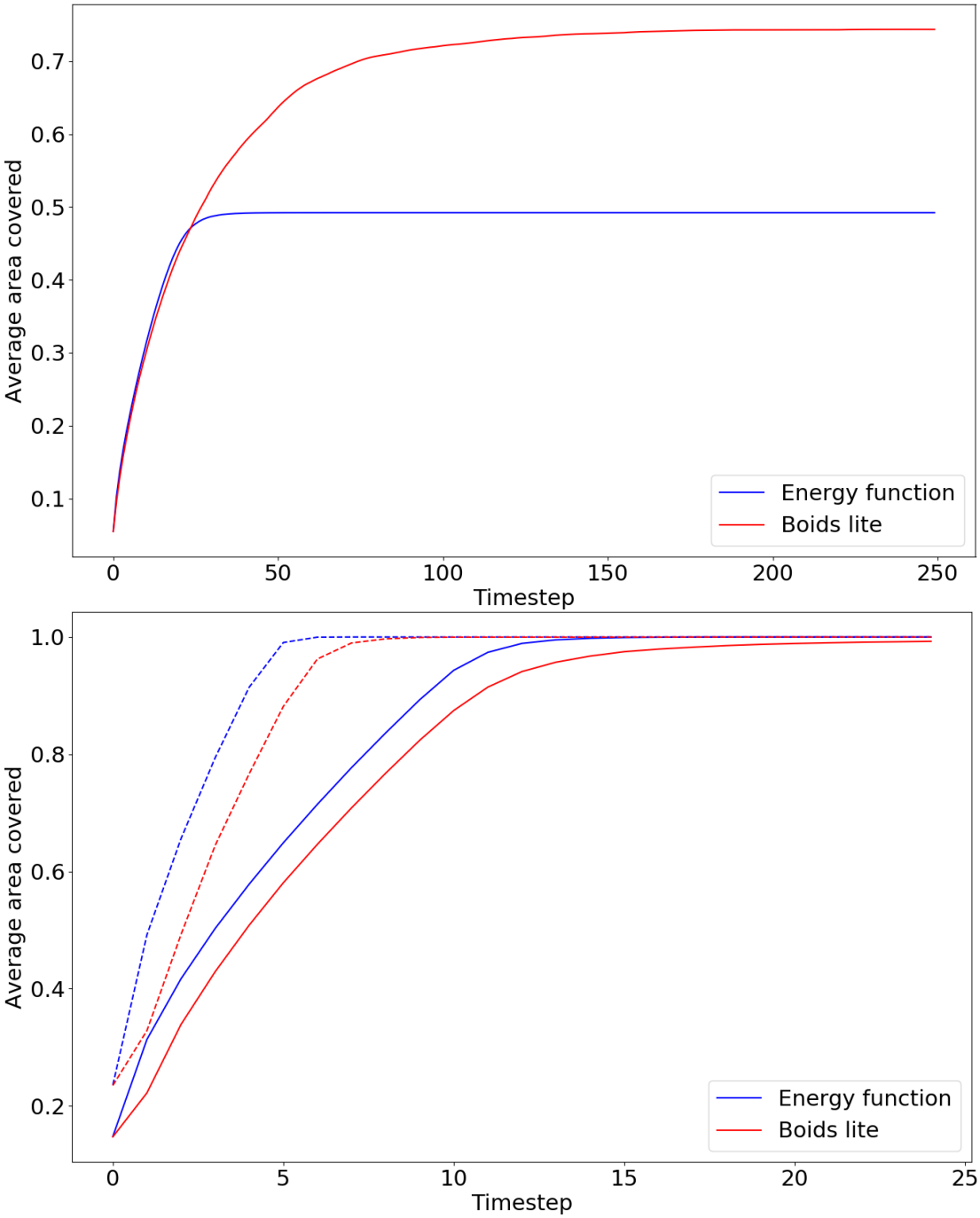


Figure 5.13: Swarms operating within a circular environment, radius 1, utilising the energy function described in Sec. 5.5.1 and Boids lite rules were clustered, i.e. randomly spatially distributed within a small circle of radius 0.1. The size of the swarms were $N = 50$, and the plots display the average timecourse of area coverage, which is how much of the environment is observed by at least one agent. The top plot shows when $r = 0.15$, and the bottom plot is for $r = 0.3$ for the solid plots and $r = 0.4$ for the dashed plots. Each simulation was conducted 10 times to acquire an average.

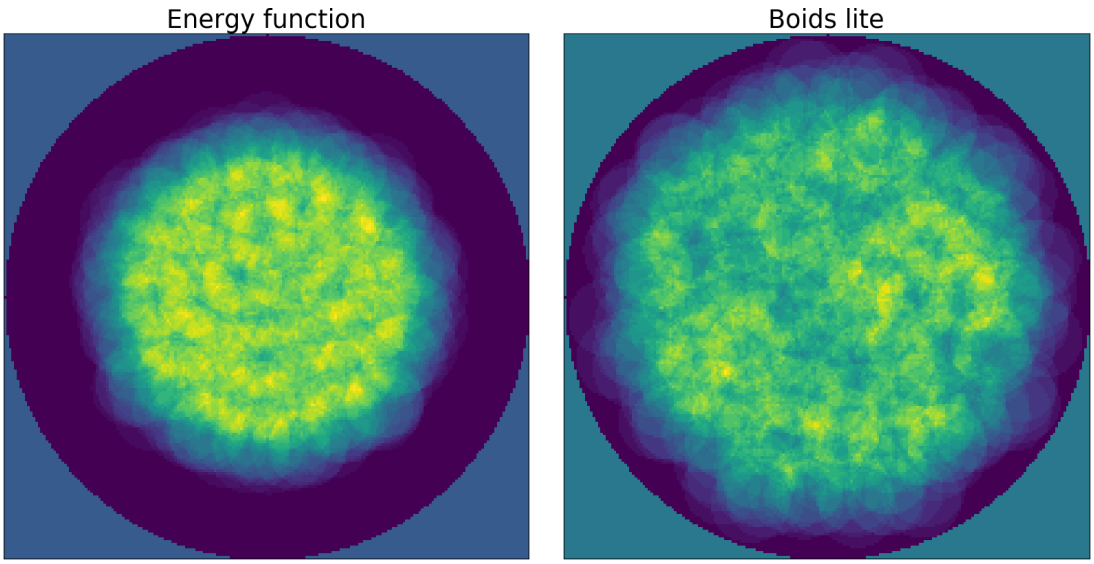


Figure 5.14: Heat maps of average area covered, where coverage is defined as the area seen by an agent within their communication range, see Fig. 5.10 for an illustraive reference. The data presented here is in Fig. 5.13 where $N = 50$ and $r = 0.15$.

result in a larger coverage, however, the Boids lite swarm will become disconnected.

The average heat maps of the area covered, Fig. 5.14, shows that the swarm with the energy function is spread uniformly from the centre as there is an even spread of area covered within the centre, i.e. overlapping communication areas. With the Boids lite though, there is no clear connectivity as the swarm just spreads out resulting in an uneven spread, thereby making it less predictable as well. This is further evidenced by the average number of neighbours and average distance of those neighbours, see Fig. 5.15. The swarms see a decline of average neighbours at the same rate, as well as the average neighbour distance, however the Boids lite swarm continues outwards and is significantly out of range of other agents.

5.5.4 Obstacles within the environment

The swarm presented in Sec. 5.5.1 can also incorporate obstacles and perceive them in a similar manner as the boundary, see Eq. 5.17. Fig. 5.16 contains examples of swarms spreading around the obstacle to form an equidistant formation. Here the obstacle is represented by a circle that the agent cannot enter.

More complicated obstacles were constructed by allowing obstacles to overlap on initialisation. The swarm is still able to spread outwards, an example seen in Fig. 5.17.

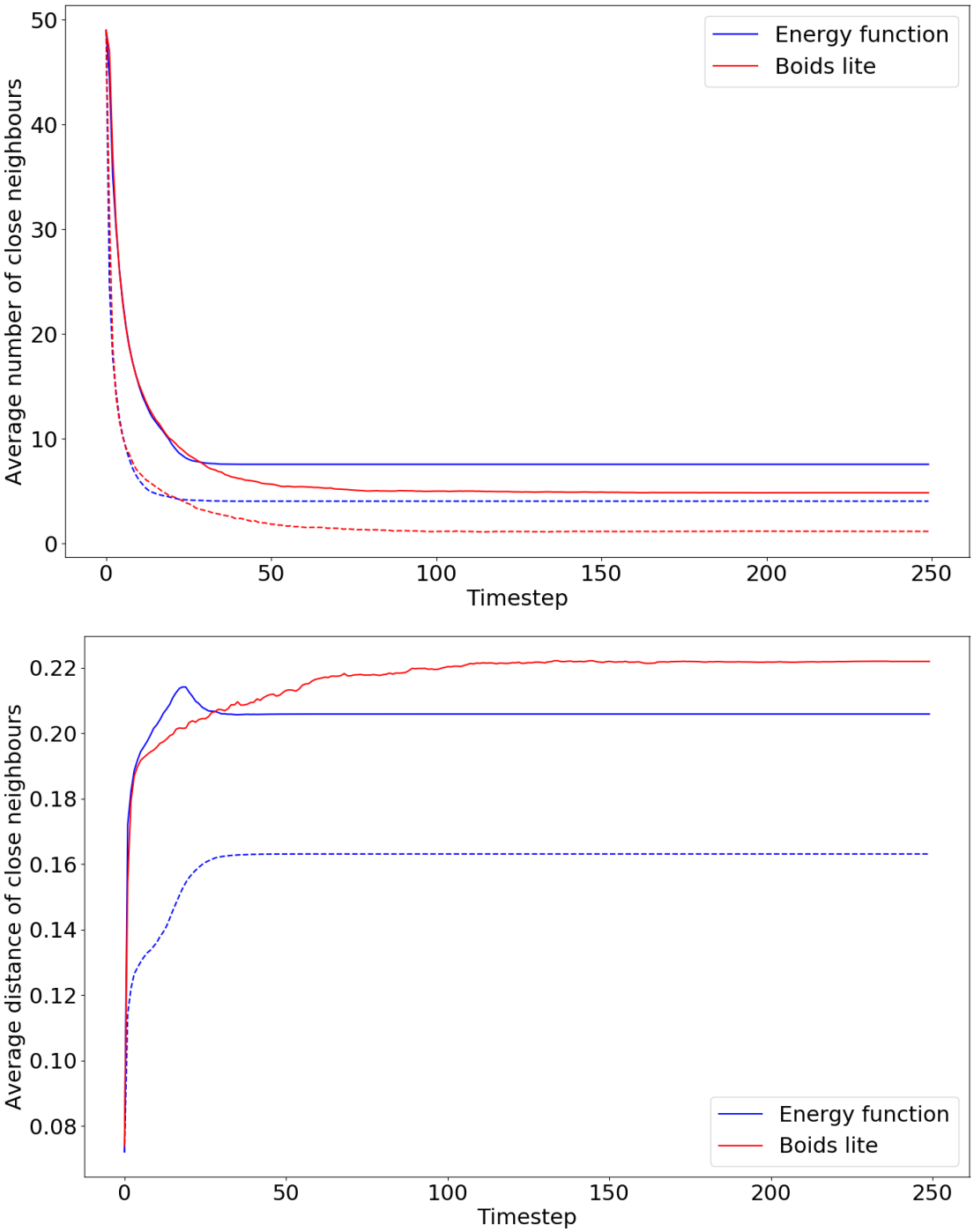


Figure 5.15: Timecourses of the average number of neighbours and the average distance of neighbours of the simulations observed in Fig. 5.13 where $N = 50$ and $r = 0.15$. The solid plots are when neighbours are considered for those within $2r$ and the dashed $1.25r$. The agents in the Boids lite swarm had no neighbours when $1.25r$ was considered, thus an average distance could not be plotted for this case for the Boids lite swarm.

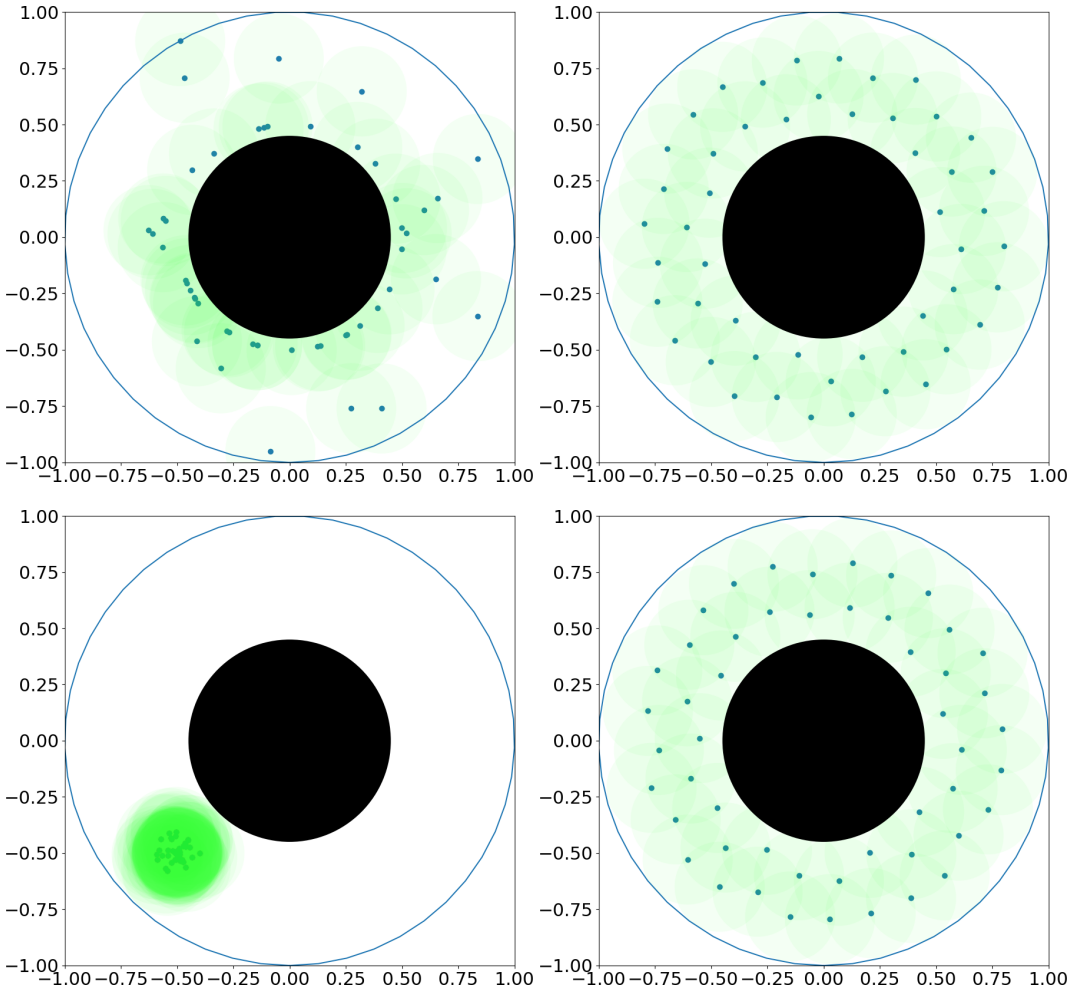


Figure 5.16: Examples of swarms spreading out in an environment where an obstacle in the centre is present, where $N = 50$ and $r = 0.2$.

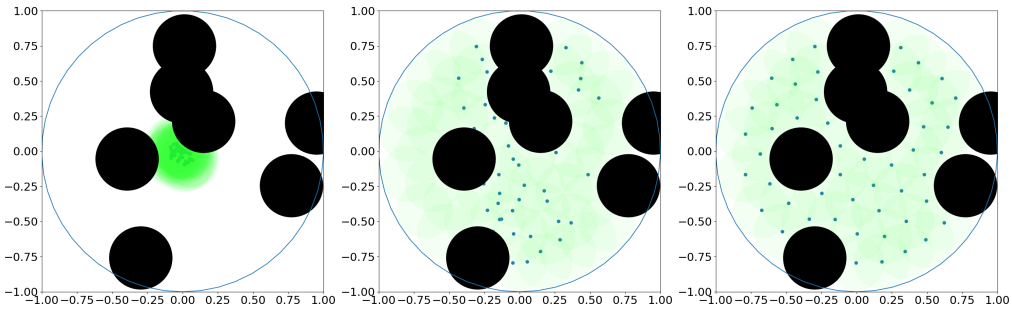


Figure 5.17: An example of a swarm initialised in the centre and then spreading outwards, and ending in a equidistant configuration. Here $N = 50$ and $r = 0.2$. The obstacles were randomly distributed across the environment.

5.6 Surrounding unknown objects

Sec. 5.5 has demonstrated how the swarm from Sec. 5.4.3 can be expanded to 2-dimensions, and the swarm's ability to spread out in an environment, including with obstacles present. Sec. 5.4.4 fixed the 1-dimensional swarm onto the perimeter of a circle, and the swarm was still performing the spreading behaviour dictated by the energy function, but from a 2-dimensional perspective it can be observed as the swarm as surrounding a circle. Barnes et al. (2009) uses a collection of potential fields to guide a swarm which can then follow circle, elliptical, and wedge formations. This is then tested by having a simulated robotic encircle a moving convoy of vehicles. The objective here is similar with the extension that the swarm will be able to surround shapes of arbitrary length, thus allowing it to remain in its equidisant formation. Here it will be presented how the swarm from Sec. 5.5 can be injected with global information to find the object of interest, and the spreading behaviour is then favoured to surround said object. Brief examples of static and morphing objects of interest are then shown.

5.6.1 Setup

There will be two minimisation tasks for the agents; minimise distance to the object of interest, *OOI*, and minimise the energy function (achieved via the spreading behaviour). To keep the overall behaviour simple the agent will have two explicitly different modes; **gravitate** and **spread**.

The agents will now have an omnidirectional sensor that can detect the object of interest with range r_o. If the *OOI* is within range r_o the agent will receive signal strength

$$w = \frac{s_o}{r_o} \tag{5.21}$$

which will inform the agent how close they are to the *OOI*, where s_o is the closest Euclidean point of the *OOI*. The agents will also now have global information regarding their position and the centre of mass of the *OOI*. If the agent does not detect the *OOI* then it will **gravitate** towards the *OOI*, meaning it will update its velocity as

$$\bar{v}_t = x_t - O \tag{5.22}$$

where O is the centre of mass of the *OOI*.

Once the agent detects the *OOI* it will **spread**, which means that it will incorporate

$\hat{v}_t$ (see Eq. 5.15) and will weight $\bar{v}_t$ with respect to the sensor signal strength w.

$$v_{t+1} = (1-\alpha_v)v_t + \alpha_v \begin{cases} \hat{v}_t + w\bar{v}_t, & \text{if } OOI \text{ detected within } r_o \\ \bar{v}_t, & \text{otherwise} \end{cases} \tag{5.23}$$

For the rest of this Chapter $\alpha_v = 0.8$ and $r_0 = 0.1$.

5.6.2 Results

Presented here are brief examples to illustrate how the swarm from Sec. 5.6.1 performs. Fig. 5.18 displays a simulation where the *OOI* was a perfect circle of radius 0.5 in the centre of the environment. The swarm was as initially clustered together bottom left and moved towards the *OOI* and began to spread around. It behaves in the same manner as the regularly spreading swarm seen prior, the agents will continue to move away from each other till equidistant formation has been achieved.

The swarm though was not sufficient in size with its communication range to fully surround the *OOI* (similar to Fig. 5.6). Another cluster of agents was added and they perform in a similar manner, and moving towards the already surrounding swarm they then integrate themselves into the first set of agents, and together they envelope the whole *OOI*. When the second set of agents integrate they slowly push the first set around, demonstrating that a surplus of agents does not affect the overall configuration of evenly spreading around the *OOI*.

In a similar manner as Fig. 5.7 even if the swarm surrounds an obstacle the number of agents might be lacking for us to use the agents' positions to understand the shape. Fig. 5.19 demonstrates this by displaying the connected positions of the swarms of various sizes when they have surrounded a more complex shaped *OOI*.

The swarm is not only robust to increasing number of agents, but also if the *OOI* changes shape over time, the swarm can adapt to it. A complex *OOI* was created by overlapping several circular obstacles. The *OOI* morphed over time by randomly increasing or decreasing the radius of each component that made up the *OOI*. A simulation demonstrating this can be seen in Fig. 5.20.

5.7 Discussion

This Chapter has demonstrated how starting from a 1-dimensional system which minimises the energy function Eq. 5.7, driving the system to an equidistant formation, can

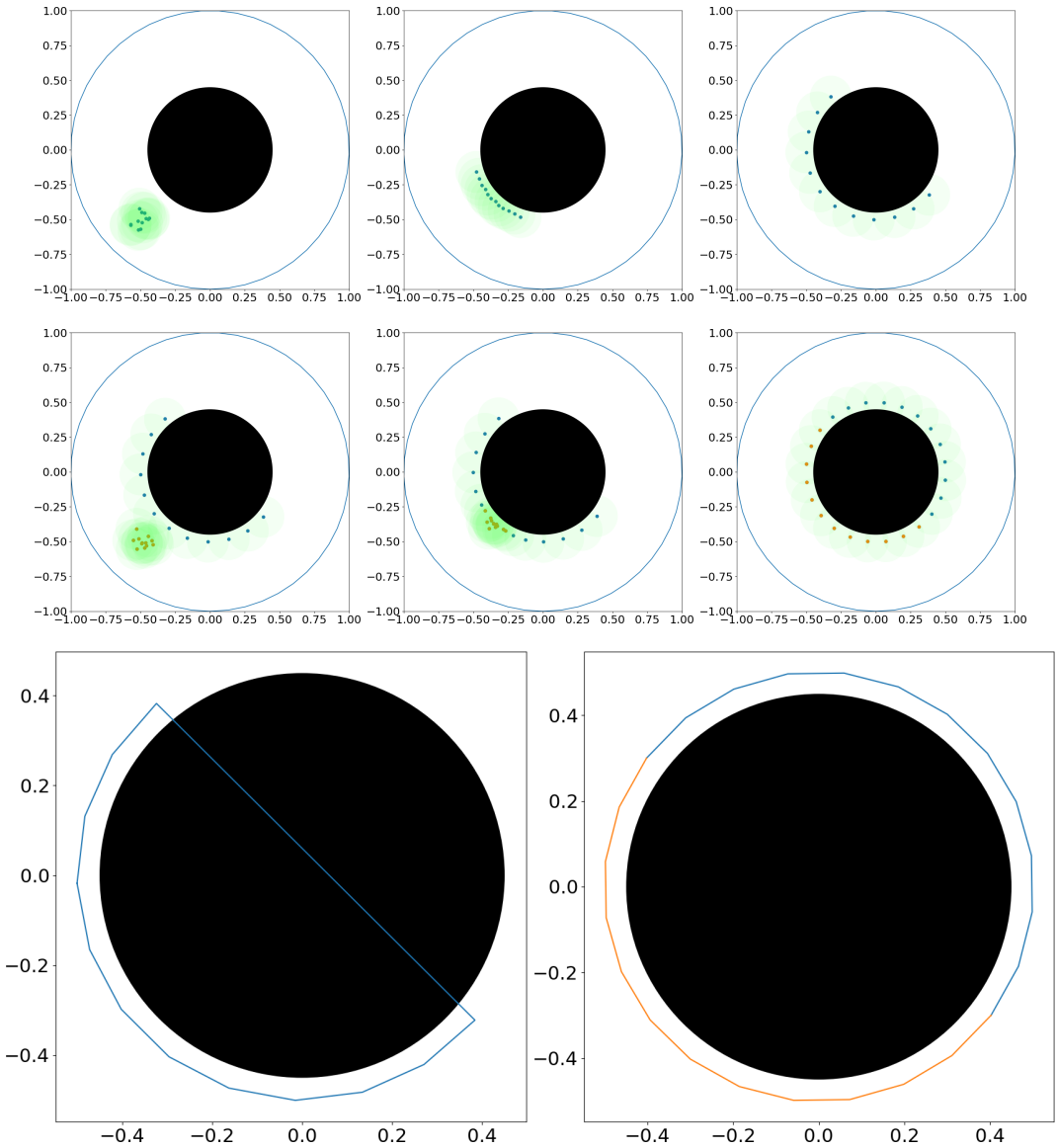


Figure 5.18: A swarm of size $N = 12$ with $r = 0.2$ was randomly distributed in a circle of radius 0.1, origin (-0.5,-0.5), i.e. bottom left of the environment. The *OOI* is a perfect circle of radius 0.5 is in the middle of environment. Following the system described in Sec. 5.6.1 the swarm moves towards the *OOI* until it has detected the *OOI*. From here the swarm begins to spread around the obstacle till equidistant formation has been achieved. As can be seen the swarm surrounds half of the *OOI*. A second cluster of 12 agents (the orange dots) is added in the same manner as the first set, and they too move towards the *OOI* and by integrating with the first set are able to fully surround the obstacle. The bottom two images shows a plot of the agents connected, i.e. an understanding of shape of the *OOI*. The left is when the first swarm has reached equidistant formation, and the right is when the second set of agents was added (blue is the first set of agents, and orange is the second).

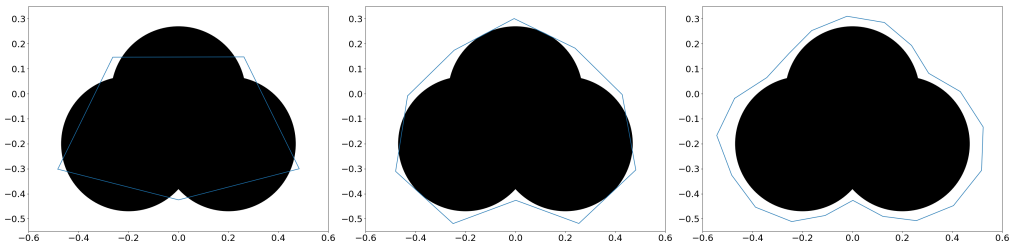


Figure 5.19: Swarms of sizes $N = 5, 10, 20$ all with $r = 0.5$ were randomly spatially distributed in an circular environment, radius 1. They surrounded the object and the images above shows the connected positions. As the number of agents increases the accuracy also increases.

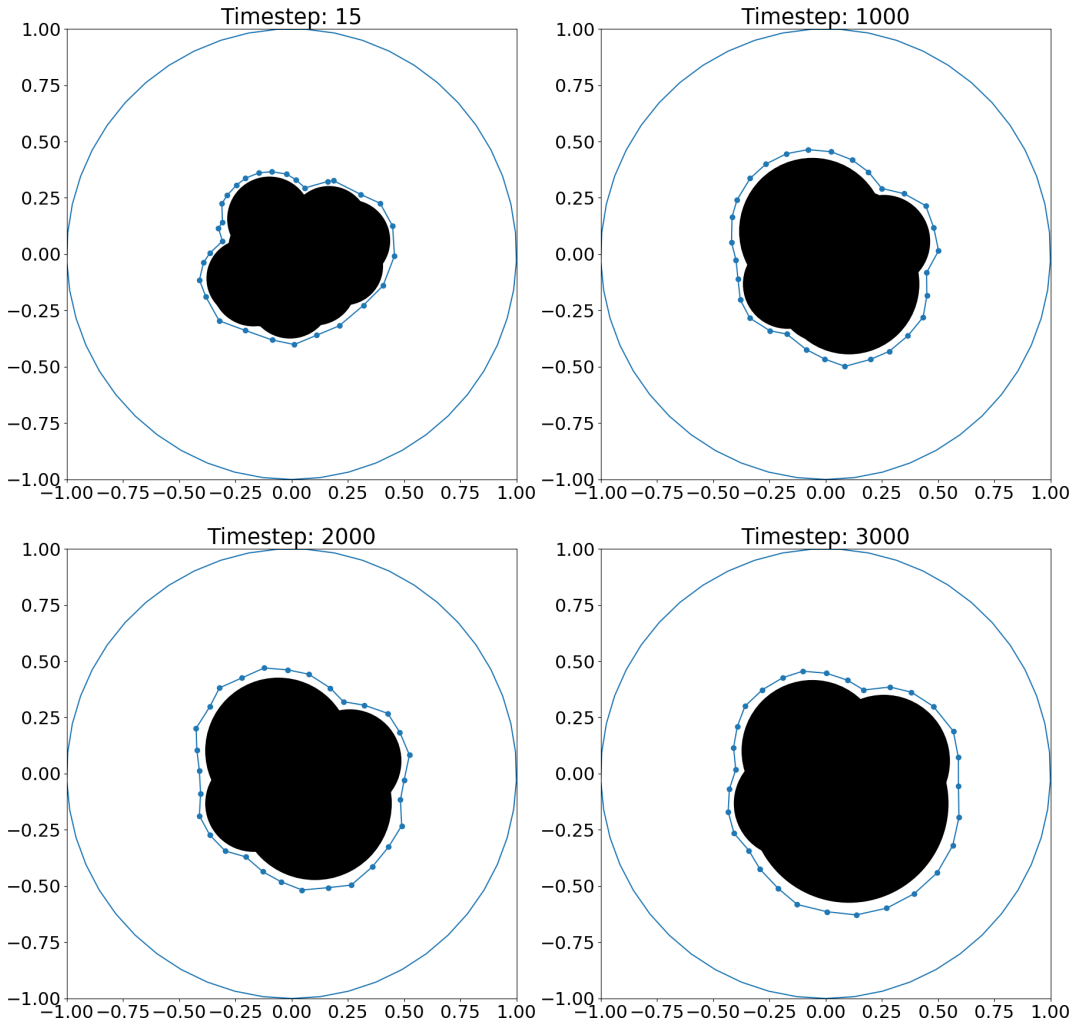


Figure 5.20: A swarm of size $N = 30$ with $r = 0.2$ surrounding an *OOI* which morphs over time. Above are various phases of the simulation, showing the swarms' positioning around the *OOI*.

then be expanded slowly to a 2-dimensional setup. The spreading behaviour can then help with the spatial task of surrounding objects of interest as well. Here aspects of the design decisions shall be discussed, as well as the applications that, if pursued, this work can be applied to.

5.7.1 Boundary detection

The swarm described in Sec. 5.5 requires an explicit weighting function to handle the boundary detection, as this is only detected once compared to the potentially numerous neighbours on the other side of the agent. The parameter, α had to be chosen beforehand; if set too high then the force of the boundary would compact the agents, and if set too low the agents would be pushed onto the boundary itself.

One possibility is to improve the sensing, and have the agent use the full boundary detected. On a robotic platform this could be a LIDAR sensor, for example, which can then use a summation and apply it to the closest detected point of the boundary. This, however, will need to be used with caution as complicated environments with walls curving away would perhaps reflect the appropriate direction in which the robot should go if simply added to the overall force. Experiments would need to be conducted on how to best utilise the additional information that any additional sensor would bring.

Another potential approach would be to incorporate a form of learning for the parameter α. Given the number of neighbours an agent sees, they should learn how to associate the average number of neighbours with the setting of α; if on average they see more neighbours, they should then set α to a lower value, and vice versa.

5.7.2 Concavity in objects of interest

Fig. 5.19 shows objects of interest with some concavity, though in this particular instance the agents are able to avoid getting stuck. This is due to the large range of the communication allowing agents on the other side of the object to act as a force to push the agent out of the concave part of the object. Similarly, if there was a substantial amount of agents in the swarm then as they approach the object of interest and begin to surround it, the presence of additional neighbours would also help push the agent out of the concave. This would allow the swarm to continue to surround the object. This assumes, however, that prior to the task it was known how many agents should be deployed to achieve the goal. In an unknown setting, a change in strategy might have to be employed. One such potential strategy would be to increase the sensor range of

the agents with regards to detecting the object, and increase the repulsive force from the object of interest. This would initially create a loose-fitting formation around the object of interest. Once the agents have settled, i.e. all agents are equidistant from their neighbours (which can be observed locally if changes in energy are minimal), then the object's repulsive force can slowly be reduced, thus creating a 'form-fitting' swarm.

5.7.3 Varying the communication range

The swarms' communication range played a significant role in covering the environment. The communication range, however, can be altered to have a virtual communication range, r', as well. This could introduce some interesting behaviours, as a virtual range can lead to changes to the swarm coverage in real time. Let's say the relative position of the neighbour was x then the distance, s could be replaced with $s = r' - x$, i.e. the neighbour is in range but is further than the virtual communication range. This would reverse the force direction causing the agents to be attracted to each other instead.

Chen et al. (2012) investigated the idea of using a virtual size in a swarm of robots. Robots with different sizes would then be segregated, as the robots with smaller virtual size would squeeze through those with larger. Similarly, agents on the edge of the swarm can pass information if they should move or not by adapting their r'. For example, an agent on the edge of the swarm can promote exploration by increasing r'. This drives the agent away from the swarm, and therefore 'pulls' its neighbours. Conversely if the agent reduced r', this will 'push' the swarm.

The agents in the system do not differentiate greatly with regards to their roles during simulation. On a superficial level, the agents on the outside could be observed as acting 'scouts', and while exploring/pushing outward subsequently bringing the inner agents out with them. This behaviour though is capped when the agents encounters boundaries or have minimised their energy, i.e. fully spread out as a swarm. With regards to the boundary the parameter α in Eq. 5.18 is what will play an important role with respect to the 'scouting'. That said, a stronger emergence of explicit roles could occur if, say, an evolutionary scheme was used with this swarm with the virtual communication range, mentioned above, being the evolving component. This could then give rise to a heterogeneous swarm with various virtual communication ranges amongst the agents, and self-organise the distribution of roles described above (attracting/repulsing).

Overall the additions described above would introduce a guided motion to the swarm as a whole. In this Chapter a lite version of Boids was used as a baseline to to compare with. If, however, guided motion was introduced to the swarm then a complete version of Boids, including cohesion and alignment as well as separation, would be very interesting to compare with. This would include not only a comparison of general behaviour, such as flocking, but more specifically, how many agents are 'lost' from the collective while the swarm is in motion. An example task could be to explore the environment and then to push a heavy object back, which would require exploration but also enough agents to remain within the swarm to be able to collectively bring back the object. Alternatively, to prevent a fluid from spreading, would require sufficient agents to adequately surround and contain the fluid in an effective amount of time. There has also been recent work regarding the emergence of superorganisms if the size of the Boids' swarm is significantly large (Maruyama et al., 2019), in which a division of labour and new types of inter-agent behaviours occurs. This would be a fascinating model to have as a comparison, with respect to the emergent properties and whether guidance for task-dependent role distribution can be introduced for the swarm to effectively explore and exploit the environment

5.7.4 Physical robot

The next appropriate step for this work is to test the system on a swarm of physical robots. A suitable candidate would be the E-puck (Mondada et al., 2009) as this robot can detect the direction and distance of received messages.

Another likely candidate is the Robotarium (Pickem et al., 2017). This allows outside users to send their software to be tested on a robot swarm, consisting of custom built robots called GRITSBots. This platform also has the required capabilities to acquire the information needed for the robots to incorporate the system proposed in this Chapter. The additional advantage is that the Robotarium's testbed can generate virtual environments for the robots, which will allow testing of the morphing *OOI* to be conducted.

5.7.5 Multi-swarm applications

There has been considerable work into the efficient clean-up of oil spills with robots (Shah et al., 2018; Vasilijevic et al., 2015), but there remains the challenge of detecting and navigating to and around the oil spill. To surround the oil spill we

could use the Turing swarm described in Chapter 4. This may be achieved by having the water robots induce a pattern (see Sec. 4.5) by detecting the oil, and the parameters can be defined to prefer the formation of a stripe pattern, thus forming a ring around the oil spill. The challenge, however, is that the rippling of the water makes it difficult for the sensors to be accurate.

It is proposed here that a multi-swarm approach may be adopted using a swarm of airborne drones in conjunction with the water-based Turing swarm. The airborne drones would follow the same method as described in Sec. 5.6.1 to reach and circle the irregularly shaped spill. To reach the oil spill, the drone swarm would simply require an approximate location for the centre of the oil spill, which may be acquired from an external source. An example of finding this centre could be from the work presented by Gil and Alacid (2018) in which a plane with a side-looking airbourne radar (SLAR) was used to locate the oil. The airborne drones could then have local sensors to detect the oil and begin their encirclement. Finally, the airborne drones could act as the POIs for inducing the striped pattern in the water-based Turing swarm given that the airborne drones will be easier to communicate with than detecting the oil with their onboard sensors.

This approach may also be applied to other scenarios, such as fighting fires (Innocente and Grasso, 2018; Ghamry et al., 2017). The drones can encircle the fire with their top down view and act as POIs for the ground robots, which can surround and suffocate the fire.

5.8 Conclusion

This Chapter tackled the issue of maximising a swarm spread while maintaining contact between agents to allow interactions to still occur. An equidistant configuration of agents is required to investigate this problem. Starting with an energy function defined by the distance between the midpoint of the two neighbouring agents, a 1-dimensional system was created where the swarm was evenly spread out in a 1-dimensional environment. It was seen that as the number of agents in the swarm increased, the time for this configuration to be attained increased exponentially. The system was then slowly adapted to include fixed boundaries and the agents given limited communication abilities. This significantly reduced the time for swarms to complete the same task.

The system was then adapted to become 2-dimensional, and the distances between neighbours was used to calculate the forces between them. Comparing this to a swarm

which only uses the separation rule of Boids, dubbed Boids lite, the swarm using the energy function performs better, with regards to time, if the communication range for the swarms is large enough that the swarms can cover the whole environment. If the communication range was smaller than this, though, the Boids lite and the swarm using the energy function would perform similarly at the start. The swarm with the energy function would stop spreading, though, significantly earlier than the Boids lite. This is due to there being no opposing force slowing the agents in the Boids lite swarm. This also means, however, that the Boids lite swarm becomes disconnected and less evenly spread, though it covers more of the space.

The spreading principle was also shown to be a useful way of surrounding objects of interest. This was shown to work not only on circular objects but also on objects of arbitrary shape, both static and fluctuating. Finally, future work needs to include testing using physical robots as well as investigating the concept of a virtual range to generate different behaviours within the swarm. It is also suggested that this may be part of a robotic multi-swarm system in which the swarm using the energy function may be airborne, and act as points of interest for a Turing swarm, with the example of oil spills to highlight this. This approach is described in Sec. 5.7.5 for the potential cleanup.

Chapter 6

Conclusions

The main aim of this Thesis was to investigate the potential of simple systems to be realised in the behaviour of robotic swarms. One of the key aspects was the communication between the individual agents, and the resulting behaviours. In this Chapter, the key outcomes and conclusions arising from this Thesis will be summarised.

6.1 Information dynamics in an evolving swarm

In Chapter 3, an evolutionary swarm was tasked with acquiring energy, which it achieves by entering an energy source area. These sources contained unlimited energy and supplied more energy the closer the agent approached the centre of the source. The agents were able to detect the sources at all times using distance sensors, and in combination with signals from neighbouring agents, the information about the sources was used as the input to an Elman network. Through evolutionary learning, the swarm would approach any source, so maximising the energy obtained. If additional, and equally attractive (with regards to position and energy output) sources were present, these caused the swarm to deteriorate in its performance. Specifically, the agents learned to cycle between sources instead of staying at one. When signalling was turned off, the swarm's performance improved. If a swarm was initially trained with signalling on, and then the signalling was switched off, the swarm would initially perform significantly worse. Subsequently, however, the swarm not only recovered but improved its performance. This indicated that the agents' controllers were not able to process conflicting information arising in the case of multiple sources and the signals instead acted as noise which perturbed the agents. What was interesting to note was the swarm's general behaviour in environments which differed to the environment they

trained on, i.e. how many sources the swarm was exposed to during training. Considering that the agent had four distance sensors, this indicated that the swarm preferred to be trained in environments in which about this number of sources were present. Simulations were conducted in which physical aspects of the agents evolved, i.e. the range of their distance sensors. The results showed that the swarm only wants distance sensing if there are few sources available, as having distance sensing helps identify the best way of navigating to these limited sources. As the number of sources increased, the average agent quickly evolved to set distance sensing range very close to zero, and the swarm's overall performance matched that of the swarm trained on a single source. This supports the view that the swarm was experiencing information overload, and was indecisive about the choices to make. Through limiting these choices and the information available the swarm regained good performance.

A strong trait for success in systems is generality, or the ability to apply the same technique to a different setup. The swarm only achieved this generality when the distance sensors were not always triggered, and the controller had to learn how to process the entire range of values that the collection of sensors would receive. This allowed the swarms that trained on three or four sources to be the best performing when placed in different environments. If it is not entirely clear at the outset what the environments are going to entail, then it might be difficult to train a robotic swarm for a general setting with the robots having a fixed setup of sensing and actuators. Evolutionary strategies are an effective tool for learning, and many aspects of a system can be part of the evolutionary process. Specifically, the physical aspects of the agents (the distance sensing range) were allowed to evolve with the controller. This allows the system to adapt to its environment as it changes. In robots, however, it is difficult to evolve a physical attribute, and particularly if the robotic system is required to operate entirely autonomously, including physical evolution.

6.2 Turing patterns for robotic swarms

Reaction-diffusion (RD) systems can serve as models of the spatial and temporal changes of different interacting species. The concentration of a species at each point in space depends upon the relationship this species has with the other species (reaction) and the rate at which it travels across space (diffusion). In a two-species RD system there tends to be an activator species and an inhibitor species, where the concentration of the inhibitor species will lower that of the activator. RD systems are able

to generate periodic spatial patterns, such as honey-comb spots and stripes, known as Turing patterns. Chapter 4 investigated a swarm attempting to recreate such patterns by each agent representing a point in space and using communication of on board concentration values to solve the RD equations. The FitzHugh-Nagumo model was used for the reaction equations. The swarm of agents initially had no sensors, and could not discern the direction of received messages, but with only second order derivatives considered in the RD system, directionality is not an issue. A static, compact swarm was considered first. This was able to generate Turing patterns, and the type of pattern was influenced by the parameter settings. The RD swarm was then spatially randomly distributed in a bound environment, with each agent attempting to locate an agent with a maximum activator value. The agents still had no sensors, instead used Brownian motion to explore with the RD system as an influence. The RD swarm created Turing patterns through their messaging, with each agent only aware of what their explicit concentration values were, and they were then able to replicate the concentrations of the virtual values in the physical space through clustering. The RD swarm was shown to form spotted patterns in the physical space, although striped patterns proved to be more challenging, since the space of the RD system was now represented by the moving agents meaning that the space itself was also changing. If the individual agents in the RD swarm were consistently near each other, achieved by reducing the size of the environment, striped patterns were easier to produce. In a similar vein, increasing the number of robots/agents would also encourage the striped patterns to form as there is a greater chance of having multiple neighbours to better enforce patterns. A binary sensor was then added to the agents allowing them to detect points of interest. If an agent detected a point of interest it would influence the concentrations on that agent. This modification allowed patterns to be induced, and by having multiple points of interest a striped pattern was formed. The RD system was then transferred to a swarm of simulated Kilobots, which had the additional limitation of lossy communication. Following the same procedure it was shown that for a static evenly spaced setup of Kilobots differing Turing patterns were formed. Furthermore, they too were able to replicate the patterns in physical space, and were able to use an initially generated Turing pattern to influence their behaviour, as seen by ring formations.

The approach of starting with a self-organised robust system, and slowly integrating agent behaviour was able to show how effective the RD system was on its own. This allowed sensory information to be included as we understood how the swarm behaved, and thus how to inject guidance. Furthermore, the parameter selection changed

the preference for pattern formation, so we could generate different patterns in real time through changing the parameters, while keeping the updated rules the same. This now forms the basis for understanding how to incorporate more information from the environment to produce appropriate behaviour for a specific task, while still maintaining the underlying strengths of self-organisation present in RD systems.

6.3 Maximising an evenly spread swarm in an unknown environment

Chapter 5 tackled the issue of maximising a swarm spread but maintaining contact to allow interactions between agents to still occur. An equidistant configuration proved to be optimal for tackling this problem. The Chapter began by considering a 1-dimensional setup in which the agents in the swarm had perfect knowledge, i.e. full information about neighbours was available at all times, and so the agents had to minimise an energy function to be able to find the midpoints between neighbours. From here the setup was modified to include fixed boundaries and the agents were given limited communication range to their neighbours, and if no neighbours were detected on either side then they would have a virtual neighbour at the extreme range. It was observed that by having the system fixed onto the perimeter of a circle, the swarm not only spread around its local environment, but in a global sense surrounded the circle. The size of the swarm and the communication range played a significant role as to whether the swarm was able to cover the whole environment. It was found that a swarm with a smaller communication range performed in a shorter time, and this was due to the swarm experiencing less counter forces by detecting fewer neighbours. From here the system was adapted to a 2-dimensional setting in which the Euclidean distances to neighbours were used to calculate the forces between them. This updated the velocity rule as there would always be fixed virtual neighbours. Since there can now be a large number of neighbours in any direction, this could overpower the force from a boundary detection as the boundary was currently only detected once, i.e. only counted as one neighbour. A weighting function was applied depending on the number of neighbours, but its parameter needed to be set beforehand. Comparing this to a swarm only using the separation rule from Boids, termed Boids lite, it would outperform Boids lite in terms of speed if the communication range of the swarms was sufficient for the agents was sufficient for the swarm to be able to cover the whole environment. The Boids lite,

however, covered significantly more of the environment if the communication range was small. Upon closer inspection, the two swarms were comparable in initial speed and coverage, but there was no force opposing the Boids lite, resulting in the swarm being less connected than the swarm using the energy function. Similar to the earlier observation of the swarm spreading on a perimeter of a circle, the swarm was able to approach an object of interest which can have arbitrary shape, and with the spreading behaviour encircle it.

The approach used to develop this swarm was based on that in Chapter 4 for the RD swarm. Starting with a simple energy function the dynamics of a simple 1-dimensional system were studied, and then slowly developed towards a swarm setup. It was observed that the spreading behaviour does not apply simply to coverage, but can also be used for smooth encircling of objects, i.e. we can build systems towards displaying different behaviours having the same underlying behaviour. This shows the strengths of principled behavioural systems, as they can be combined with other functions and systems for specific problems facing robotic systems, such as oil spills and forest fires, as discussed in Sec. 5.7.5.

Bibliography

Adam, J. A. (2006). *Mathematics in nature: Modeling patterns in the natural world*. Princeton University Press.

Adami, C. (1998). *Introduction to Artificial Life*. TELOS Springer-Verlag, Santa Clara.

Alon, U., Surette, M. G., Barkai, N., and Leibler, S. (1999). Robustness in bacterial chemotaxis. *Nature*, 397(6715):168–171.

Ashby, W. R. (1947a). The nervous system as physical machine: With special reference to the origin of adaptive behavior. *Mind*, 56(221):44–59.

Ashby, W. R. (1947b). Principles of the self-organizing dynamic system. *Journal of General Psychology*, 37(2):125–128.

Bahat, A. and Eisenbach, M. (2006). Sperm thermotaxis. *Molecular and Cellular Endocrinology*, 252(1-2):115–119.

Bak, P., Tang, C., and Wiesenfeld, K. (1988). Self-organized criticality. *Physical review A*, 38(1):364.

Balch, T. (1999). Reward and diversity in multirobot foraging. In *IJCAI-99 Workshop on Agents Learning*.

Barnes, L. E., Fields, M. A., and Valavanis, K. P. (2009). Swarm formation control utilizing elliptical surfaces and limiting functions. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 39(6):1434–1445.

Barrio, R. A. and Varea, C. (2006). Non-linear systems. *Physica A: Statistical Mechanics and its Applications*, 372(2):210 – 223.

Bays, C. (2010). Introduction to cellular automata and conway's game of life. In *Game of Life Cellular Automata*, pages 1–7. Springer.

Beckers, R., Holland, O. E., and Deneubourg, J. (2000). Fom local actions to global tasks: Stigmergy and collective robotics. In *Prerational Intelligence: Adaptive Behavior and Intelligent Systems Without Symbols and Logic, Volume 1, Volume 2 Prerational Intelligence: Interdisciplinary Perspectives on the Behavior of Natural and Artificial Systems, Volume 3*, pages 1008–1022. Springer.

Benzinger, T. H. (1969). Heat regulation: homeostasis of central temperature in man. *Physiological Reviews*, 49(4):671–759.

Berg, K. and Aronoff, M. (2017). Self-organization in the spelling of english suffixes: The emergence of culture out of anarchy. *Language*, 93(1):37–64.

Birattari, M., Delhaisse, B., Francesca, G., and Kerdoncuff, Y. (2016). Observing the effects of overdesign in the automatic design of control software for robot swarms. In *International Conference on Swarm Intelligence*, pages 149–160. Springer.

Bojinov, H., Casal, A., and Hogg, T. (2000). Emergent structures in modular self-reconfigurable robots. In *Proceedings 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation.*, volume 2.

Bonabeau, E. and Dorigo, M.and Theraulaz, G. (1999). *Swarm intelligence: from natural to artificial systems*. Oxford university press.

Braitenberg, V. (1986). *Vehicles: Experiments in synthetic psychology*. MIT press.

Bratton, D. and Kennedy, J. (2007). Defining a standard for particle swarm optimization. In *2007 IEEE swarm intelligence symposium*, pages 120–127. IEEE.

Buhl, J., Sumpter, D. J. T., Couzin, I. D., Hale, J. J., Despland, E., Miller, E. R., and Simpson, S. J. (2006). From disorder to order in marching locusts. *Science*, 312(5778):1402–1406.

Cannon, W. B. (1926). Physiological regulation of normal states: some tentative postulates concerning biological homeostatics. *Jubilee Volume for Charles Richet*.

Carrillo-Zapata, D., Sharpe, J., Winfield, A. F. T., Giuggioli, L., and Hauert, S. (2019). Toward controllable morphogenesis in large robot swarms. *IEEE Robotics and Automation Letters*, 4(4):3386–3393.

Chakraborty, A. and Kar, A. K. (2017). Swarm intelligence: A review of algorithms. In *Nature-Inspired Computing and Optimization*, pages 475–494. Springer.

Chaudhuri, P. P., Chowdhury, D. R., Nandi, S., and Chattopadhyay, S. (1997). *Additive cellular automata: theory and applications*, volume 43. John Wiley & Sons.

Chen, J., Gauci, M., Price, M. J., and Groß, R. (2012). Segregation in swarms of e-puck robots based on the brazil nut effect. In *Proceedings of the 11th International Conference on Autonomous Agents and Multiagent Systems-Volume 1*, pages 163–170.

Clerc, M. and Kennedy, J. (2002). The particle swarm-explosion, stability, and convergence in a multidimensional complex space. *IEEE Transactions on Evolutionary Computation*, 6(1):58–73.

Corning, P. A. (2002). The re-emergence of "emergence": A venerable concept in search of a theory. *Complexity*, 7(6):18–30.

Correll, N. and Martinoli, A. (2007). Robust distributed coverage using a swarm of miniature robots. In *Proceedings 2007 IEEE International Conference on Robotics and Automation*, pages 379–384. IEEE.

Correll, N. and Martinoli, A. (2011). Modeling and designing self-organized aggregation in a swarm of miniature robots. *Int. J. Robotics Research*, 30(5):615–626.

Cortes, J., Martinez, S., Karatas, T., and Bullo, F. (2004). Coverage control for mobile sensing networks. *IEEE Transactions on Robotics and Automation*, 20(2):243–255.

Couzin, I. D., Ioannou, C. C., Demirel, G., Gross, T., Torney, C. J., Hartnett, A., Conradt, L., Levin, S. A., and Leonard, N. E. (2011). Uninformed individuals promote democratic consensus in animal groups. *Science*, 334(6062):1578–1580.

Couzin, I. D., Krause, J., Franks, N. R., and Levin, S. A. (2005). Effective leadership and decision-making in animal groups on the move. *Nature*, 433(7025):513–516.

Crandall, J. W., Anderson, N., Ashcraft, C., Grosh, J., Henderson, J., McClellan, J., Neupane, A., and Goodrich, M. A. (2017). Human-swarm interaction as shared control: Achieving flexible fault-tolerant systems. In *International Conference on Engineering Psychology and Cognitive Ergonomics*, pages 266–284. Springer.

Crosato, E., Jiang, L., Lecheval, V., Lizier, J. T., Wang, X. R., Tichit, P., Theraulaz, G., and Prokopenko, M. (2018). Informative and misinformative interactions in a school of fish. *Swarm Intelligence*, 12(4):283–305.

Crowther, W. J. (2004). Rule-based guidance for flight vehicle flocking. *Proceedings of the Institution of Mechanical Engineers, Part G: Journal of Aerospace Engineering*, 218(2):111–124.

Crutchfield, J. P. (1994). The calculi of emergence: computation, dynamics and induction. *Physica D: Nonlinear Phenomena*, 75(1-3):11–54.

Curtis, S. A., Mica, J., Nuth, J., Marr, G., Rilee, M., and Bhat, M. (2000). Ants(autonomous nano technology swarm)- an artificial intelligence approach to asteroid belt resource exploration. In *IAF, International Astronautical Congress, 51 st, Rio de Janeiro, Brazil*.

Demange, G., Zapolsky, H., Patte, R., and Brunel, M. (2017). A phase field model for snow crystal growth in three dimensions. *NPJ Computational Materials*, 3(1):1–7.

Deneubourg, J., Goss, S., Franks, N., Sendova-Franks, A., Detrain, C., and Chrétien, L. (1991). The dynamics of collective sorting robot-like ants and ant-like robots. In *Proceedings of the first international conference on simulation of adaptive behavior on from animals to animats*, pages 356–363.

Der, R., Steinmetz, U., and Pasemann, F. (1999). A new principle to back up evolution with learning. *Computational Intelligence for Modelling, Control, and Automation; IOS Press: Amsterdam, Demark*, pages 43–47.

Dimidov, C., Oriolo, G., and Trianni, V. (2016). Random walks in swarm robotics: An experiment with Kilobots. In *Int. Conf. Swarm Intelligence*, pages 185–196.

Dorigo, M. and Di Caro, G. (1999). Ant colony optimization: a new meta-heuristic. In *Proceedings of the 1999 congress on evolutionary computation-CEC99 (Cat. No. 99TH8406)*, volume 2, pages 1470–1477. IEEE.

Dorigo, M. and Gambardella, L. M. (1997). Ant colony system: a cooperative learning approach to the traveling salesman problem. *IEEE Transactions on Evolutionary Computation*, 1(1):53–66.

Doursat, R., Sayama, H., and Michel, O. (2013). A review of morphogenetic engineering. *Natural Computing*, 12(4):517–535.

Ducatelle, F., Di Caro, G. A., Pinciroli, C., and Gambardella, L. M. (2011). Self-organized cooperation between robotic swarms. *Swarm Intelligence*, 5(2):73–96.

Duncan, S. (2019). Taking stigmergy out of the lab and into the field: student research abstract. In *Proceedings of the 34th ACM/SIGAPP Symposium on Applied Computing*, pages 961–964.

Economou, A. D., Ohazama, A., Porntaveetus, T., Sharpe, P. T., Kondo, S., Basson, M. A., Gritli-Linde, A., Cobourne, M. T., and Green, J. B. (2012). Periodic stripe formation by a Turing mechanism operating at growth zones in the mammalian palate. *Nature Genetics*, 44(3):348.

Edwards, S. J. (2000). Swarming on the battlefield: past, present, and future.

Ermentrout, B. (1991). Stripes or spots? Nonlinear effects in bifurcation of reaction-diffusion equations on the square. *Proceedings of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*, 434(1891):413–417.

Erskine, A. and Herrmann, J. M. (2015). Crips: Critical particle swarm optimisation. In *Artificial Life Conference Proceedings 13*, pages 207–214. MIT Press.

Eugene, F. (1991). Markets efficient capital II. *The Journal of Finance*, 46(5).

Evjemo, L. D., Gjerstad, T., Grøtli, E. I., and Sziebig, G. (2020). Trends in smart manufacturing: Role of humans and industrial robots in smart factories. *Current Robotics Reports*, 1(2):35–41.

Fernández, N., Maldonado, C., and Gershenson, C. (2014). Information measures of complexity, emergence, self-organization, homeostasis, and autopoiesis. In *Guided self-organization: Inception*, pages 19–51. Springer.

Ferrer, E. (2018). A wearable general-purpose solution for human-swarm interaction. In *Proceedings of the Future Technologies Conference*, pages 1059–1076. Springer.

FitzHugh, R. (1961). Impulses and physiological states in theoretical models of nerve membrane. *Biophysical Journal*, 1(6):445–466.

Fujisawa, R., Dobata, S., Sugawara, K., and Matsuno, F. (2014). Designing pheromone communication in swarm robotics: Group foraging behavior mediated by chemical substance. *Swarm Intelligence*, 8(3):227–246.

Garcia-Gonzalo, E. and Fernandez-Martinez, J. L. (2012). A brief historical review of particle swarm optimization (pso). *Journal of Bioinformatics and Intelligent Control*, 1(1):3–16.

Gardner, M. (1970). Mathematical games: The fantastic combinations of john conway's new solitaire game "life". *Scientific American*, 223(4):120–123.

Gauci, M., Ortiz, M. E., Rubenstein, M., and Nagpal, R. (2017). Error cascades in collective behavior: A case study of the gradient algorithm on 1000 physical agents. In *Proc.*, AAMAS '17, pages 1404–1412.

Gershenson, C. (2012). Guiding the self-organization of random boolean networks. *Theory in Biosciences*, 131(3):181–191.

Gershenson, C. and Heylighen, F. (2003). When can we call a system self-organizing? In *European Conference on Artificial Life*, pages 606–614. Springer.

Ghamry, K. A., Kamel, M. A., and Zhang, Y. (2017). Multiple uavs in forest fire fighting mission using particle swarm optimization. In *2017 International Conference on Unmanned Aircraft Systems (ICUAS)*, pages 1404–1409. IEEE.

Ghosh, N., Banerjee, I., and Sherratt, R. S. (2019). On-demand fuzzy clustering and ant-colony optimisation based mobile data collection in wireless sensor network. *Wireless Networks*, 25(4):1829–1845.

Gil, P. and Alacid, B. (2018). Oil spill detection in terma-side-looking airborne radar images using image features and region segmentation. *Sensors*, 18(1):151.

Google (2013). *LiquidFun*. `https://google.github.io/liquidfun/`.

Grassé, P. (1959). La reconstruction du nid et les coordinations interindividuelles chezbellicositermes natalensis etcubitermes sp. la théorie de la stigmergie: Essai d'interprétation du comportement des termites constructeurs. *Insectes Sociaux*, 6(1):41–80.

Gray, P. and Scott, S. K. (1983). Autocatalytic reactions in the isothermal, continuous stirred tank reactor: isolas and other forms of multistability. *Chemical Engineering Science*, 38(1):29–43.

Groß, R., Bonani, M., Mondada, F., and Dorigo, M. (2006). Autonomous self-assembly in swarm-bots. *IEEE Transactions on Robotics*, 22(6):1115–1130.

Gross, R. and Dorigo, M. (2009). Towards group transport by swarms of robots. *International Journal of Bio-Inspired Computation*, 1(1-2):1–13.

Guo, H., Meng, Y., and Jin, Y. (2009). A cellular mechanism for multi-robot construction via evolutionary multi-objective optimization of a gene regulatory network. *BioSystems*, 98(3):193–203.

Haken, H. (1977). Synergetics. *Physics Bulletin*, 28(9):412–414.

Hale, M. F., Angus, M., Buchanan, E., Li, W., Woolley, R., Le Goff, L. K., De Carlo, M., Timmis, J., Winfield, A. F., Hart, E., Eiben, A. E., and Tyrrell, A. M. (2020). Hardware design for autonomous robot evolution. In *2020 IEEE Symposium Series on Computational Intelligence (SSCI)*, pages 2140–2147. IEEE.

Halley, J. D. and Winkler, D. A. (2008). Classification of emergence and its relation to self-organization. *Complexity*, 13(5).

Hartman, C. and Benes, B. (2006). Autonomous boids. *Computer Animation and Virtual Worlds*, 17(3-4):199–206.

Hashimoto, T. and Ikegami, T. (1996). Emergence of net-grammar in communicating agents. *BioSystems*, 38(1).

Hauert, S., Leven, S., Varga, M., Ruini, F., Cangelosi, A., Zufferey, J., and Floreano, D. (2011). Reynolds flocking in reality with fixed-wing robots: communication range vs. maximum turning rate. In *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 5015–5020. IEEE.

Hauert, S., Zufferey, J., and Floreano, D. (2009). Evolved swarming without positioning information: an application in aerial communication relay. *Autonomous Robots*, 26(1):21–32.

Herrmann, J. M. (2001). Dynamical systems for predictive control of autonomous robots. *Theory in Biosciences*, 120(3-4):241–252.

Hoff, N., Wood, R., and Nagpal, R. (2013). Distributed colony-level algorithm switching for robot swarm foraging. In *Distributed Autonomous Robotic Systems*, pages 417–430. Springer.

Hölldobler, B. and Wilson, E. O. (1990). *The Ants*. Harvard University Press.

Imrie, C. and Herrmann, J. M. (2017). Self-organisation of spatial behaviour in a kilobot swarm. In *Conference on Biomimetic and Biohybrid Systems*, pages 551–561. Springer.

Imrie, C. C., Herrmann, J. M., and Witkowski, O. (2021). The paradox of choice in evolving swarms: information overload leads to limited sensing. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO) 2021*.

Innocente, M. S. and Grasso, P. (2018). Swarm of autonomous drones self-organised to fight the spread of wildfires. In *RSFF*, pages 30–39.

International Federation of Robotics (2020). Executive summary world robotics 2020 industrial robots.

Jensen, H. J. (1998). *Self-organized criticality: emergent complex behavior in physical and biological systems*, volume 10. Cambridge university press.

Jin, J. and Tang, L. (2010). Optimal coverage path planning for arable farming on 2D surfaces. *Transactions of the ASABE*, 53(1).

Jin, Y. and Meng, Y. (2011). Morphogenetic robotics: An emerging new field in developmental robotics. *IEEE Transact. Systems, Man, Cybern. C*, 41(2):145–160.

Jones, J. C., Myerscough, M. R., Graham, S., and Oldroyd, B. P. (2004). Honey bee nest thermoregulation: diversity promotes stability. *Science*, 305(5682):402–404.

Kenney, W. L. and Munce, T. A. (2003). Invited review: aging and human temperature regulation. *Journal of Applied Physiology*, 95(6):2598–2603.

Kiyatkin, E. A. (2010). Brain temperature homeostasis: physiological fluctuations and pathological shifts. *Frontiers in Bioscience*, 15:73.

Klausmeier, C. A. (1999). Regular and irregular patterns in semiarid vegetation. *Science*, 284(5421):1826–1828.

Klyubin, A. S., Polani, D., and Nehaniv, C. L. (2005). Empowerment: A universal agent-centric measure of control. In *2005 IEEE Congress on Evolutionary Computation*. IEEE.

Kondo, S. (2009). How animals get their skin patterns: fish pigment pattern as a live turing wave. In *Systems Biology*, pages 37–46. Springer.

Kong, L., Zhao, M., Liu, X., Lu, J., Liu, Y., Wu, M., and Shu, W. (2013). Surface coverage in sensor networks. *IEEE Transactions on Parallel and Distributed Systems*, 25(1).

Krieger, M. J. B., Billeter, J., and Keller, L. (2000). Ant-like task allocation and recruitment in cooperative robots. *Nature*, 406(6799):992–995.

Kriegman, S., Blackiston, D., Levin, M., and Bongard, J. (2020). A scalable pipeline for designing reconfigurable organisms. *Proceedings of the National Academy of Sciences of the U.S.A.*, 117(4):1853–1859.

Kumar, A. S., Manikutty, G., Bhavani, R. R., and Couceiro, M. S. (2017). Search and rescue operations using robotic darwinian particle swarm optimization. In *2017 International Conference on Advances in Computing, Communications and Informatics (ICACCI)*, pages 1839–1843. IEEE.

Legenstein, R. and Maass, W. (2007). Edge of chaos and prediction of computational performance for neural circuit models. *Neural Networks*, 20(3):323–334.

Li, W., Gauci, M., and Groß, R. (2016). Turing learning: a metric-free approach to inferring behavior and its application to swarms. *Swarm Intelligence*, 10(3):211–243.

Liu, L. and Ma, H. (2011). On coverage of wireless sensor networks for rolling terrains. *IEEE Transactions on Parallel and Distributed Systems*, 23(1):118–125.

Liu, Y., Gong, D., Sun, J., and Jin, Y. (2017). A many-objective evolutionary algorithm using a one-by-one selection strategy. *IEEE Transactions on Cybernetics*, 47(9):2689–2702.

Lizier, J. T., Prokopenko, M., and Zomaya, A. Y. (2012). Local measures of information storage in complex distributed computation. *Information Sciences*, 208:39–54.

Lyons, B. I. and Herrmann, J. M. (2020). Reflexive reinforcement learning: Methods for self-referential autonomous learning. In *12th International Joint Conference on Computational Intelligence (IJCCI 2020)*.

Magnenat, S., Philippsen, R., and Mondada, F. (2012). Autonomous construction using scarce resources in unknown environments. *Autonomous Robots*, 33(4):467–485.

Marjovi, A., Nunes, J. G., Marques, L., and de Almeida, A. (2009). Multi-robot exploration and fire searching. In *2009 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1929–1934. IEEE.

Marocco, D. and Nolfi, S. (2006). Self-organization of communication in evolving robots. In *Proceedings of the Conference on Artificial Life (ALIFE)*.

Martínez-García, R., Calabrese, J. M., Mueller, T., Olson, K. A., and López, C. (2013). Optimizing the search for resources by sharing information: Mongolian gazelles as a case study. *Physical Review Letters*, 110(24):248106.

Martius, G. (2010). *Goal-oriented control of self-organizing behavior in autonomous robots*. PhD thesis, PhD thesis, Göttingen University.

Martius, G. and Herrmann, J. M. (2010). Taming the beast: Guided self-organization of behavior in autonomous robots. In *From Animals to Animats 11*, pages 50–61. Springer.

Martius, G., Herrmann, J. M., and Der, R. (2007). Guided self-organisation for autonomous robot development. In *European Conference on Artificial Life*, pages 766–775. Springer.

Maruyama, N., Saito, D., and Ikegami, T. (2019). Emergence of superorganisms in a large scale boids model. In *2019 IEEE Symposium Series on Computational Intelligence (SSCI)*. IEEE.

Merton, R. K. (1968). The Matthew effect in science. *Science*, 159(3810):56–63.

Michael, N., Shen, S., Mohta, K., Kumar, V., Nagatani, K., Okada, Y., Kiribayashi, S., Otake, K., Yoshida, K., and O., K. (2014). Collaborative mapping of an earthquake damaged building via ground and aerial robots. In *Field and service robotics*, pages 33–47. Springer.

Miller, J. M., Wang, X. R., Lizier, J. T., Prokopenko, M., and Rossi, L. F. (2014). Measuring information dynamics in swarms. In *Guided self-organization: Inception*, pages 343–364. Springer.

Molins, P., Stillman, N., and Hauert, S. (2019). Trail formation using large swarms of minimal robots. *Cybernetics and Systems*, 50(8):693–710.

Mondada, F., Bonani, M., Raemy, X., Pugh, J., Cianci, C., Klaptocz, A., Magnenat, S., Zufferey, J., Floreano, D., and Martinoli, A. (2009). The e-puck, a robot designed for education in engineering. In *Proceedings of the 9th conference on autonomous robot systems and competitions*, volume 1, pages 59–65. IPCB: Instituto Politécnico de Castelo Branco.

Munafo, R. P. (2014). Stable localized moving patterns in the 2-d gray-scott model. *arXiv preprint arXiv:1501.01990*.

Murray, J. D. (2007). *Mathematical biology: I. An introduction*, volume 17. Springer Science & Business Media.

Nagumo, J., Arimoto, S., and Yoshizawa, S. (1962). An active pulse transmission line simulating nerve axon. *Proceedings of the IRE*, 50(10):2061–2070.

Obute, S. O., Dogar, M. R., and Boyle, J. H. (2019). Chemotaxis based virtual fence for swarm robots in unbounded environments. In *Conference on Biomimetic and Biohybrid Systems*, pages 216–227. Springer.

Oh, H., Shirazi, A. R., Sun, C., and Jin, Y. (2017). Bio-inspired self-organising multi-robot pattern formation: A review. *Robotics and Autonomous Systems*, 91:83–100.

Olfati-Saber, R., Fax, J. A., and Murray, R. M. (2007). Consensus and cooperation in networked multi-agent systems. *Proceedings of the IEEE*, 95(1).

Özdemir, A., Gauci, M., Kolling, A., Hall, M. D., and Groß, R. (2019). Spatial coverage without computation. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 9674–9680. IEEE.

Penders, J., Alboul, L., Witkowski, U., Naghsh, A., Saez-Pons, J., Herbrechtsmeier, S., and El-Habbal, M. (2011). A robot swarm assisting a human fire-fighter. *Advanced Robotics*, 25(1-2):93–117.

Pickem, D., Glotfelter, P., Wang, L., Mote, M., Ames, A., Feron, E., and Egerstedt, M. (2017). The robotarium: A remotely accessible swarm robotics research testbed. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1699–1706. IEEE.

Pinciroli, C., Talamali, M. S., Reina, A., Marshall, J. A. R., and Trianni, V. (2018). Simulating kilobots within argos: models and experimental validation. In *International Conference on Swarm Intelligence*, pages 176–187. Springer.

Pinciroli, C., Trianni, V., O'Grady, R., Pini, G., Brutschy, A., Brambilla, M., Mathews, N., Ferrante, E., Di Caro, G., Ducatelle, F., Stirling, T., Gutiérrez, A., Gambardellai, L. M., and Dorigo, M. (2011). ARGoS: A modular, multi-engine simulator for heterogeneous swarm robotics. In *IEEE/RSJ Int. Conf. Intelligent Robots and Systems (IROS)*, pages 5027–5034.

Pitonakova, L., Crowder, R., and Bullock, S. (2016). Information flow principles for plasticity in foraging robot swarms. *Swarm Intelligence*, 10(1):33–63.

Popovici, A. and Popovici, D. (2002). Cellular automata in image processing. In *Fifteenth International Symposium on Mathematical Theory of Networks and Systems*, volume 1, pages 1–6. Citeseer.

Prokopenko, M. (2013). *Guided self-organization: Inception*, volume 9. Springer Science & Business Media.

Reynolds, C. W. (1987). Flocks, herds and schools: A distributed behavioral model. In *Proceedings of the 14th annual conference on Computer graphics and interactive techniques*, pages 25–34.

Rizzoli, A. E., Oliverio, F., Montemanni, R., and Gambardella, L. M. (2004). Ant colony optimisation for vehicle routing problems: from theory to applications. *Galleria Rassegna Bimestrale Di Cultura*, 9(1):1–50.

Roth, L. M. and Willis, E. R. (1952). A study of cockroach behavior. *The American Midland Naturalist*, 47(1):66–129.

Rubenstein, M., Ahler, C., and Nagpal, R. (2012). Kilobot: A low cost scalable robot system for collective behaviors. In *IEEE Int. Conf. Robotics and Automation (ICRA)*, pages 3293–3298.

Rubenstein, M., Cornejo, A., and Nagpal, R. (2014). Programmable self-assembly in a thousand-robot swarm. *Science*, 345(6198):795–799.

Saadeh, Y. and Vyas, D. (2014). Nanorobotic applications in medicine: current proposals and designs. *American Journal of Robotic Surgery*, 1(1).

Salge, C., Glackin, C., and Polani, D. (2014). Empowerment–an introduction. In *Guided Self-Organization: Inception*. Springer.

San Juan, V., Santos, M., and Andújar, J. M. (2018). Intelligent uav map generation and discrete path planning for search and rescue operations. *Complexity*, 2018.

Sandeep, D. N. and Kumar, V. (2017). Review on clustering, coverage and connectivity in underwater wireless sensor networks: A communication techniques perspective. *IEEE Access*, 5.

Sauter, J. A., Matthews, R., Van Dyke Parunak, H., and Brueckner, S. A. (2005). Performance of digital pheromones for swarming vehicle control. In *Proceedings of the fourth international joint conference on autonomous agents and multiagent systems*, pages 903–910.

Sayama, H. (2009). Swarm chemistry. *Artificial Life*, 15(1):105–114.

Scheffer, M., Bascompte, J., Brock, W. A., Brovkin, V., Carpenter, S. R., Dakos, V., Held, H., Van Nes, E. H., Rietkerk, M., and Sugihara, G. (2009). Early-warning signals for critical transitions. *Nature*, 461(7260):53–59.

Schmickl, T., Möslinger, C., and Crailsheim, K. (2006). Collective perception in a robot swarm. In *International Workshop on Swarm Robotics*, pages 144–157. Springer.

Schroeder, A., Ramakrishnan, S., Kumar, M., and Trease, B. (2017). Efficient spatial coverage by a robot swarm based on an ant foraging model and the lévy distribution. *Swarm Intelligence*, 11(1):39–69.

Shah, M., Shah, S. K., and Shah, M. (2018). Autonomous robotic vehicle for oil spills cleaning with nano particles. In *2018 International Conference on Manipulation, Automation and Robotics at Small Scales (MARSS)*, pages 1–6. IEEE.

Shew, W. L. and Plenz, D. (2013). The functional benefits of criticality in the cortex. *The neuroscientist*, 19(1):88–100.

Siero, E., Doelman, A., Eppinga, M. B., Rademacher, J. D. M., Rietkerk, M., and Siteur, K. (2015). Striped pattern selection by advective reaction-diffusion systems: Resilience of banded vegetation on slopes. *Chaos*, 25(3):036411.

Simaria, A. S. and Vilarinho, P. M. (2009). 2-antbal: An ant colony optimisation algorithm for balancing two-sided assembly lines. *Computers & Industrial Engineering*, 56(2):489–506.

Slavkov, I., Carrillo-Zapata, D., Carranza, N., Diego, X., Jansson, F., Kaandorp, J., Hauert, S., and Sharpe, J. (2018). Morphogenesis in robot swarms. *Science Robotics*, 3(25).

Smith, S. C. and Herrmann, J. M. (2011). Homeokinetic reinforcement learning. In *IAPR International Workshop on Partially Supervised Learning*, pages 82–91. Springer.

Soto, F., Wang, J., Ahmed, R., and Demirci, U. (2020). Medical micro/nanorobots in precision medicine. *Advanced Science*, 7(21).

Steil, J. J. and Maier, G. W. (2017). Robots in the digitalized workplace. *The Wiley Blackwell handbook of the psychology of the internet at work*, pages 403–422.

Stewart, R. L. and Russell, R. A. (2006). A distributed feedback mechanism to regulate wall construction by a robotic swarm. *Adaptive Behavior*, 14(1):21–51.

Stützle, T. and Hoos, H. H. (2000). MAX–MIN ant system. *Future Generation Computer Systems*, 16(8):889–914.

Suganuma, H., Kawai, Y., Park, J., and Asada, M. (2019). Maximization of transfer entropy leads to evolution of functional differentiation of swarms. In *The 2018 Conference on Artificial Life: A Hybrid of the European Conference on Artificial Life (ECAL) and the International Conference on the Synthesis and Simulation of Living Systems (ALIFE)*, pages 414–415. MIT Press.

Theraulaz, G. and Bonabeau, E. (1999). A brief history of stigmergy. *Artificial life*, 5(2):97–116.

Torney, C. J., Berdahl, A., and Couzin, I. D. (2011). Signalling and the evolution of cooperative foraging in dynamic environments. *PLoS Computational Biology*, 7(9).

Trautmann, H., Wagner, T., Naujoks, B., Preuss, M., and Mehnen, J. (2009). Statistical methods for convergence detection of multi-objective evolutionary algorithms. *Evolutionary Computation*, 17(4):493–509.

Trianni, V., Groß, R., Labella, T. H., Şahin, E., and Dorigo, M. (2003). Evolving aggregation behaviors in a swarm of robots. In *European Conference on Artificial Life*, pages 865–874. Springer.

Trianni, V., Nolfi, S., and Dorigo, M. (2006). Cooperative hole avoidance in a swarm-bot. *Robotics and Autonomous Systems*, 54(2):97–103.

Turing, A. M. (1952). The chemical basis of morphogenesis. *Philosophical Transactions of the Royal Society of London B*, 237(641):37–72.

Vaidis, M. and Otis, M. J. (2020). Toward a robot swarm protecting a group of migrants. *Intelligent Service Robotics*, pages 1–16.

Vardanega, P. J., Sooriyabandara, M., Giuliani, M., Winfield, A. F. T., and Hauert, S. (2020). Mutual shaping in swarm robotics: User studies in fire and rescue, storage organization, and bridge inspection. *Designing Self-Organization in the Physical Realm*.

Vargas, P. A., Sardinha, H., and Dragone, M. (2020). Combining lévy walks and flocking for cooperative surveillance using aerial swarms. In *17th European Conference on Multi-Agent Systems 2020*.

Vasilijevic, A., Calado, P., Lopez-Castejon, F., Hayes, D., Stilinovic, N., Nad, D., Mandic, F., Dias, P., Gomes, J., Molina, J. C., Guerrero, A., Gilabert, J., Miskovic, N., Vukic, Z., Sousa, J., and Georgiou, G. (2015). Heterogeneous robotic system for underwater oil spill survey. In *OCEANS 2015-Genova*, pages 1–7. IEEE.

Vassev, E., Sterritt, R., Rouff, C., and Hinchey, M. (2012). Swarm technology at nasa: building resilient systems. *IT Professional*, 14(2):36–42.

Vázquez-Otero, A., Faigl, J., and Muñuzuri, A. P. (2012). Path planning based on reaction-diffusion process. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 896–901. IEEE.

White, S. H., Del Rey, A. M., and Sánchez, G. R. (2007). Modeling epidemics using cellular automata. *Applied Mathematics and Computation*, 186(1):193–202.

Witkowski, O. and Ikegami, T. (2016). Emergence of swarming behavior: foraging agents evolve collective motion based on signaling. *PloS One*, 11(4):e0152756.

Wolfram, S. (1984). Computation theory of cellular automata. *Communications in Mathematical Physics*, 96(1):15–57.

Wong, D. Y., Rogers, G. J., Porretta, B., and Kundur, P. (1988). Eigenvalue analysis of very large power systems. *IEEE Transactions on Power Systems*, 3(2).

Yan, P., Bai, C., Zheng, H., and Guo, J. (2020). Flocking control of uav swarms with deep reinforcement leaming approach. In *2020 3rd International Conference on Unmanned Systems (ICUS)*, pages 592–599. IEEE.

Yanagisawa, H., Kawamata, O., and Ueda, K. (2019). Modeling emotions associated with novelty under uncertainty: A bayesian approach. *Frontiers in computational neuroscience*.

Zanchettin, A. M., Croft, E., Ding, H., and Li, M. (2018). Collaborative robots in the workplace [from the guest editors]. *IEEE Robotics Automation Magazine*, 25(2):16–17.

Zipser, D., Kehoe, B., Littlewort, G., and Fuster, J. (1993). A spiking network model of short-term active memory. *Journal of Neuroscience*, 13(8):3406–3420.

Zou, D. and Tan, P. (2012). Coslam: Collaborative visual slam in dynamic environments. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(2):354–366.