Coordination and Control Mechanisms for Embedded Swarm-like Agents

Robert J. Mullen

Submitted in partial fulfillment of the requirements
of Kingston University for the degree of
Doctor of Philosophy

May 2011

Abstract

Observations of the mechanisms of natural systems have given us a wide-range of problem solving tools that can be applied to computational and technology related challenges. This thesis explores the use of swarm intelligence mechanisms to facilitate group level cooperative coordination and control of swarm-like agents that are embedded in 2D or 3D environments, and explores how distributed dynamic behaviours can be integrated in to the self-organisation process. Specifically a number of algorithms are developed to facilitate adaptive pattern formation and manipulation for two distinctly different problems.

Firstly, large-scale pattern formation is considered using an embedded swarm of software agents. The agents are considered as virtual entities which are embedded into digital images at the pixel level, such that the intensity map of the image corresponds to a landscape within which the swarm of agents move. The agent-agent and agent-environment interactions are then studied in the context of emergent pattern formation, from which a number of ant-algorithms are developed to achieve a range of image and video processing solutions by inducing swarm self-organisation in response to user specified image features. Artificial pheromones are used to reinforce features of interest in the image landscape, and after the swarm has self-organised, the resultant pheromone map reveals the pattern feature to be extracted. The algorithm can be adapted for different types of image features with relative ease, and simultaneous self-organisation of multiple swarms in the same image environment is implemented to achieve distributed multi-feature extraction. The dynamic nature of the self-organising process is exploited to extend the functionality of the algorithm to feature tracking in real-time imagery, where the swarms effectively track features of interest from frame to frame. An adaptive threshold method is developed which exploits the distributed nature of the swarm approach, by allowing individual ant agents to adapt their own feature threshold parameters in response to their local environment. This is both an interesting study with regards to artificial swarm pattern formation, and also provides practical image and video processing solutions which do not require a full image scan or any filtering operations, unlike many traditional

methods. The novel adaptive threshold method eliminates the requirement for a user-set threshold, and allows for distributed, multi-level thresholding across image environments, as well as adaptive capabilities for dynamic imagery.

The second problem focuses on pattern formation and manipulation of a small swarm of hardware agents in a swarm robotics problem setting. Transferring from software agents to hardware agents introduces several difficulties to overcome in order to fully realise the distributed nature of the swarm intelligence approach to multi-robot formation control. The second part of this thesis focuses on designing a control architecture that enables cooperative coordination and control of multiple robots, leading to group level adaptive pattern formation and manipulation, using a fully distributed algorithm that requires no inter-robot communication and retains robot anonymity. This is achieved using a distributed variation of the virtual forces approach. The use of a genetic algorithm for problem specific parameter optimisation is investigated to improve performance with respect to pattern formation for area coverage. A multi-behavioural approach is investigated for the problem scenario of locating and monitoring multiple target areas within a partially observable environment, where the self-organising pattern formation behaviours are exploited to provide distributed coverage. A new mechanism called Virtual Robot Nodes (VRNs) is introduced which improves swarm-level cohesion and allows for more complex formation and pattern management. The VRN method allows individual robots to self-manage their experienced virtual forces in response to their perception of their local environment and neighbouring robots, allowing for distributed dynamic adaptation. Verification of the proposed algorithms is carried out through a range of experiments in 2D simulation, physics and sensor based simulation, and embedded simulation on real robots in a laboratory environment, for a range of test scenarios. The application of different nature inspired control architectures for small to large sized swarms, and from software entities to hardware entities, promotes a focal point for discussion on the wide-ranging potential for harnessing the knowledge of nature in solving computational problems.

Acknowledgements

I would like to thank my supervisors; Ndedi Monekosso, Paolo Remagnino and Sarah Barman, for the help and support they have provided throughout my work for this thesis. I would also like to thank Paul Wilkin and RBG KEW for their assistance with the leaf image data-set collection. I am grateful to Kingston University Faculty of Computing, Information Systems and Mathematics for funding my PhD. Finally, I would like to acknowledge the endless support of my family and friends, as well as past supervisors and tutors who have undoubtedly helped me get to where I am now.

List of Publications

Journal Papers

1. R. J. Mullen, S. Barman, P. Remagnino and D. Monekosso - **Towards autonomous robot swarms for multi-target localisation and monitoring with applications to counter IED operations** in *International Journal of Intelligent Defence Support Systems*, 4(1) Inderscience Publishers, January, pp. 87-107, (2011).

2. R. J. Mullen, D. Monekosso, S. Barman and P. Remagnino - **A review of ant algorithms** in *Expert Systems with Applications*, 36(6) Elsevier, August, pp. 9608-9617, (2009).

Conference Papers

1. R. J. Mullen, D. N. Monekosso, S. A. Barman and P. Remagnino - **Reactive coordination and adaptive lattice formation in mobile robotic surveillance swarms** in *10th International Symposium on Distributed Autonomous Robotics Systems*, 1 Nov - 3 Nov 2010, Lausanne, Switzerland, (2010).

2. R. J. Mullen, D. N. Monekosso, S. A. Barman and P. Remagnino - **Autonomous control laws for mobile robotic surveillance swarms** in *IEEE International Symposium on Computational Intelligence for Security and Defense Applications*, 8 July - 10 July 2009, Ottawa, Canada, pp. 104-109, (2009).

3. R. J. Mullen, D. Monekosso, S. Barman, P. Remagnino and P. Wilkin - Chapter **Artificial ants to extract leaf outlines and primary venation patterns** in *Ant Colony Optimization and Swarm Intelligence*, Lecture Notes in Computer Science, Edited by Dorigo, M.; Birattari, M.; Blum, C.; Clerc, M.; Stutzle, T.; Winfield, A. F. T., Springer, pp. 251-258. (2008).

Contents

Chapter 1

Introduction

Over the past decade there has been a great deal of interest from computer scientists, engineers and physicists into how natural swarm behaviours occur, and how we can harness such simple yet sophisticated behaviours to solve real-world science, engineering and industrial problems. Indeed, through the study of the mechanisms of natural swarm systems a wide range of nature inspired computational problem solving methods have been developed, for a wide range of different problems and applications across science and industry, and the term *swarm intelligence* has been established as describing a field of research that encompasses many different areas of science; from biology, to physics, to computer science.

Many of the early well-established computational swarm intelligence algorithms were inspired by the behaviours observed in social insects, and have proved to be particularly successful at solving Combinatorial Optimisation (CO) problems. These CO problems are however somewhat restrictive in terms of the movement in the environment in which the swarm agents are embedded to solve the problem, in comparison to the type of environments in which natural insect swarms operate in. The study of Self-Organisation (SO) and emergent behaviour in swarm intelligence has lead researchers to simulate swarm behaviours in more open, unrestrictive environments with examples ranging from discretised digital image media to 3D virtual environments. The operational environment becomes even more unrestrictive and closer to the natural insect swarms environment when we consider swarm intelligence applied to multi-robot systems (MRS). This latter

area of research, now commonly known as *swarm robotics*, encompasses a whole new set of challenges due to the hardware considerations of the robotic agents, as well as the real-world environment issues.

At the centre of swarm intelligence is self-organised collective movement and emergent pattern formation, be it similar to the foraging patterns of ants, the flocking patterns of birds, or the honeycomb structure of a bee's nest. The mechanisms behind these behaviours all vary, but also share common properties. One important property is that these 'systems' are all decentralised, in that there is no central command or decision making centre, rather these complex, cooperative behaviours are the result of many local, relatively simple interactions. This is arguably the pinnacle of swarm intelligence, and exploiting such properties in computational problem solving can, and has, proved to be particularly rewarding. However, designing systems based on the principles of swarm intelligence is not so simple in practice.

This thesis considers swarm-like agents which are embedded in a 2D or 3D environment (where the environment could be virtual or real-world). By this it is meant that the agents move and sense within a world that has structure in two or three dimensions, and the agents themselves are considered as physical entities within that world. This distinction is important, as the research in this thesis considers on an abstract level that the agents, be they virtual or physical, act in some way as if they were material agents in a material environment. In other words, although much of the work of this thesis considers virtual agents and environments, and simulated real-world agents and environments, the developed behaviours are compared on an abstract level to real world 'material' agents and environments. On one level this work thus deals with a simulated 'virtual swarm,' and on another level, it deals with utilising the virtual swarm to solve a number of specific problems by exploiting self-organised collective movement and emergent pattern formation within the swarm.

In order to achieve self-organised collective movement and emergent pattern formation in a swarm of 'simple' homogeneous embedded agents, one has to design a control algorithm common to all agents, with appropriate local interaction rules to provide the desired distributed collective behaviour(s). This is a challenge in itself, and, when considering a less-constrained environment landscape, one has to consider potential variations in structure and other characteristics across the

environment, as well as dynamic elements to the environment, which means the control algorithms have to be robust to environmental changes. Again, given the distributed nature of the swarm approach, this is a non-trivial problem.

Each agent needs to reason about its surroundings and status in relation to the desired global behaviour, with only limited local knowledge. The control mechanisms must ensure that individual agents with only a local, limited and partial view of their surroundings, can estimate their performance in relation to the entire swarm in order to achieve cohesive swarm-level coordination and control.

The merits of swarm intelligence have been well demonstrated for a vast range of problems and applications, yet there are still many open problems in creating distributed adaptive swarm behaviours. This thesis addresses a number of these problems in two very distinct application areas, which respectively provide their own challenges to the problems at hand, yet also demonstrate the diverse applicability of the swarm intelligence approach.

1.1 Main Aims and Objectives

It has been suggested [1] that biologists overemphasise the role of evolution, and that many of the phenomena associated with swarm SO, and in particular pattern formation, can be explained by applying simple physical or mathematical rules. Studying and designing such rules is what this thesis entails. The particular challenges of focus are:

- The use of embedded agents in 2D or 3D environments, with the first part of the thesis concerning virtual agents and environments, and the second concerning real-world[1] agents and environments;
- Maintaining a distributed system (specifically with agent anonymity and only indirect communication);
- Achieving swarm-level distributed adaptation.

[1]The term 'real-world' as used here covers both simulated real-world, as well as actual physical real-world.

The main aims and objectives of this thesis are summarised below:

- To develop a new set of swarm coordination and control algorithms for cooperative collective movement and pattern formation, exploiting distributed adaptation.
- To better understand the role of 'self-organisation' within autonomous cooperative coordination and pattern formation, in terms of how multiple relatively simple swarm agents interact at the local level, to produce complex behaviours at the global level, in 2D and 3D environments.
- To investigate two main questions:
  - Is simple, indirect, communication at the local-level adequate to facilitate complex coherent pattern formation behaviour at the global-level, for swarms of anonymous agents in complex unconstrained environments?
    - **Hypothesis 1.** Relative complex behaviour at the global level can be achieved with little direct communication, as long as the agents can sense their local environment with an adequate degree of accuracy relevant to the particular problem.
  - Can individual agent adaptation converge to swarm adaptation through the process of self-organisation?
    - **Hypothesis 2.** Through simple homogeneous adaptation rules applied to individual agents, the swarm as a whole can adapt in a distributed fashion to optimise across a varying environment and improve overall performance in terms of cohesive pattern formation and adaptation.

Hypothesis 1 relates to the fundamentals of SO, and the first part of this hypothesis; *"relative complex behaviour at the global level can be achieved with little direct communication,"* has been well studied from both a biological and computational perspective. A large amount of work in this area has assumed agents to have perfect, or near perfect, sensing capabilities required to action the SO behaviour. This is especially the case in the above mentioned CO type problems, as well as other 'virtual agent' type scenarios. Given the focus of research in this

thesis being on embedded agents in environments with 2D or 3D structure, the latter part of hypothesis 1; *"..as long as the agents can sense their local environment with an adequate degree of accuracy relevant to the particular problem,"* essentially adds an important caveat to the well cited former statement. With increasing capabilities to realise swarm robotics systems to be deployed outside of laboratory environments, this caveat becomes increasingly important, to challenge the sometimes assumed 'robustness' characteristic of the swarm intelligence approach.

The tern 'robustness' as used in this thesis refers principally to the ability of the swarm to cope with sensor noise, in terms of the abilities of individual agents to sense their local environment. The term 'robustness' is also used in reference to 'robustness to robot failure;' which refers to the ability of a robot swarm to function when members of that swarm have suffered failures, where these 'failures' could be hardware failures or software failures of varying degrees.

Hypothesis 2 relates to exploitation of the SO mechanism to invoke distributed adaptation. An example from nature of such a process is the way in which a school of fish rapidly adapts its global shape and direction to avoid predator attacks. This adaptation at the global level is the result of each individual fish reacting to its local surroundings, resulting in a distributed adaptation of the entire school. The process of SO itself allows for some level of adaptation, but this hypothesis challenges that the level of adaptation can be increased to specifically optimise pattern formation and adaptation, by explicitly encoding additional homogeneous adaptation rules across all members of the swarm.

1.2 Main Contributions and Publications

The next chapter details related work and background relevant to this thesis. Chapters 3, 4, 5 and 6 contain the research carried out for this thesis, with concluding remarks and discussion given in chapter 7, and appendices in chapter 8.

Chapter 3 focuses on the application of an ant-algorithm for feature extraction in digital image media. The research is two-fold; investigating the role of SO in pattern formation and adaptation with an artificial swarm of ant-like agents em-

bedded in digital images; and secondly providing a qualitative and quantitative analysis of the ant-algorithm approach to image feature extraction. The main contribution from this chapter is in providing advancements in the development and application of ant-algorithms for use in image analysis. A modified version of the *AntSystem* algorithm is developed for image feature extraction, where virtual pheromone trails are used for stigmergy as well as to represent the final output solution. Varying heuristic information and thresholding techniques are explored and validated. A novel application for ant-algorithms is explored as a case-study for the proposed feature extraction algorithm, with work carried out in collaboration with RBG KEW. The algorithm is applied to a dataset of digitally scanned leaf images, to perform image feature extraction on leaf outlines and primary venation patterns. A rigorous analysis of the algorithm performance with respect to varying image quality is carried out, and specifically, the algorithm is evaluated qualitatively and quantitatively against ground truth data and traditional existing image processing techniques. The quantitative analysis advances on previous work in this area, where mostly only simple qualitative visual inspection of output results has been discussed previously. A novel addition to the proposed algorithm is developed which allows multiple swarms to extract multiple different features from the image, simultaneously self-organising within the same digital image environment. This extension to the algorithm is evaluated qualitatively for a number of different features and image types. Chapter 3 contains work from the following two publications:

- R. J. Mullen, D. Monekosso, S. Barman and P. Remagnino - **A review of ant algorithms** in *Expert Systems with Applications*, 36(6) Elsevier, August, pp. 9608-9617, (2009).
- R. J. Mullen, D. Monekosso, S. Barman, P. Remagnino and P. Wilkin - Chapter **Artificial ants to extract leaf outlines and primary venation patterns** in *Ant Colony Optimization and Swarm Intelligence*, Lecture Notes in Computer Science, Edited by Dorigo, M.; Birattari, M.; Blum, C.; Clerc, M.; Stutzle, T.; Winfield, A. F. T., Springer, pp. 251-258. (2008).

Chapter 4 extends on the work from chapter 3, focusing on swarm adaptation in digital image environments. There are two main contributions from this chapter.

Firstly, a new adaptive threshold method is proposed for use with the image feature extraction ant-algorithm presented in chapter 3. This allows for distributed threshold adaptation, with each individual ant-agent managing its own threshold, thus allowing the swarm to dynamically adapt to different regions of the image, as the swarm self-organises. This adaptive algorithm is shown to successfully converge to near-optimum thresholds for a range of different image types, often outperforming a non-optimum static, global threshold. Furthermore, the adaptive threshold eliminates the requirement for a user-set threshold which can require time consuming trial and error and persistent changing for different image types and datasets. The second main contribution from chapter 4 is the application of the proposed image feature extraction algorithm to dynamic imagery, including real-time video. A qualitative and quantitative analysis of the swarms ability to adapt to changing image patterns is carried out. The quantitative analysis furthers the state-of-the-art by providing a statistical measure of adaptive SO in the context of image pattern formation and adaptation, and provides a quantitative analysis in terms of image processing related performance. The swarms ability to continuously self-organise to track image features and patterns in real-time imagery is analysed qualitatively in the context of image feature tracking, with a number of case-study examples furthering the state-of-the-art in terms of analysing the capabilities and potential of this approach for video processing applications. The inherently adaptive nature of the swarm intelligence SO approach is shown to work well for basic image feature tracking. Furthermore, the novel adaptive threshold method is applied to real-time imagery and distributed real-time parameter adaptation is demonstrated through a number of case-study examples.

In chapter 5 the focus of the research shifts from 'software agents' in 'artificial environment' so 'hardware agents' in 'real-world environments.' This chapter focuses on pattern formation and control of a swarm of relatively simple homogeneous robots. Based on the balancing of local internal and external environmental virtual forces, a number of reactive coordination and control laws are developed to autonomously coordinate a group of nonholonomic homogeneous robots to facilitate self-organised pattern formation and manipulation. The control laws are adapted to create a number of elementary behaviours such as dispersion and target tracking and a novel prototypical swarm robotics system is developed for

the problem of target acquisition and monitoring in partially observable environments. A multi-behaviour strategy is developed and implemented, based on simple reactive, local decision making, incorporated into a finite-state-machine architecture. The proposed algorithms are evaluated through a number of case study problem scenarios in simple 2D simulation, physics and sensor based simulation, and embedded simulation on real robots in a controlled laboratory environment. The experimentation on real-robots provides valuable additional results on how well swarm-related behaviours work on real hardware, and in particular how well these behaviours and methods cope with real-world sensor and actuator noise. Chapter 5 contains research from the following publications:

- R. J. Mullen, S. Barman, P. Remagnino and D. Monekosso – **Towards autonomous robot swarms for multi-target localisation and monitoring with applications to counter IED operations** in *International Journal of Intelligent Defence Support Systems*, 4(1) Inderscience Publishers, January, pp. 87–107, (2011).
- R. J. Mullen, D. N. Monekosso, S. A. Barman and P. Remagnino – **Autonomous control laws for mobile robotic surveillance swarms** in *IEEE International Symposium on Computational Intelligence for Security and Defense Applications*, 8 July – 10 July 2009, Ottawa, Canada, pp. 104–109, (2009).

Chapter 6 extends on the work from chapter 5, developing further the control algorithms from chapter 5 to include novel distributed parameter adaptation and sensor driven dynamic collective movement. The main contribution from this chapter is the development of a new formation control and adaptation method called the Virtual Robot Node (VRN) mechanism. The VRN mechanism acts as a distributed dynamic formation management system, and is shown to increase swarm cohesion and reduce the 'clustering effect' which is often prominent in swarm robotics systems using virtual potential fields for SO. The VRN mechanism is also shown to provide flocking capability. A novel method of sensor driven dynamic collective movement is proposed, using on-board range sensors to allow the swarm to dynamically adapt to changing geospatial characteristics while traversing a structured environment. The VRN mechanism is again shown to increase swarm cohesion for this purpose, and is evaluated through a number

of case-study experiments. This method advances the state-of-the-art by allowing the robot swarm to adapt the formation to changing environment structures, while maintaining swarm cohesion, without relying on inter-robot communication or individual agent identification. Chapter 6 contains work from the following publication:

- R. J. Mullen, D. N. Monekosso, S. A. Barman and P. Remagnino - **Reactive coordination and adaptive lattice formation in mobile robotic surveillance swarms** in *10th International Symposium on Distributed Autonomous Robotics Systems*, 1 Nov - 3 Nov 2010, Lausanne, Switzerland, (2010).

Other developments, minor contributions and future work are discussed in chapter 7. Each research chapter also contains a short introduction and related work section relevant to the specific research contained within that chapter.

Chapter 2

Related Work

2.1 Introduction

This chapter covers related work and background knowledge relevant to the topics investigated and discussed throughout this thesis. Firstly an overview of the nature inspired elements of the work of this thesis and much of the related work is given. The focus is then given to two main areas of swarm intelligence research which are the focus of the research in this thesis, namely *Ant Algorithms* and *Swarm Robotics.*

2.2 Nature Inspired

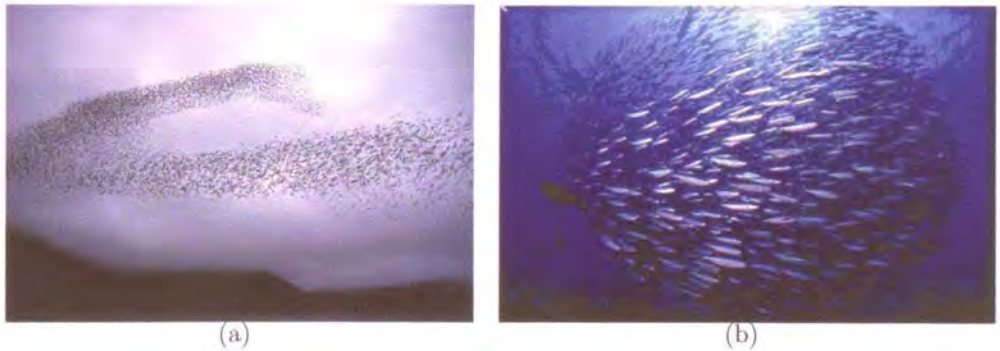
Much of the research in this thesis is inspired by observations of swarms in nature, be it social insect swarms, or microscopic particle swarms, and in particular the pattern formations and collective movement that results at the macroscopic, or 'swarm level,' due to the many local interactions at the microscopic or individual level.

The field of swarm intelligence [2] encompasses the study of how biological swarms in nature achieve collective behaviour and distributed problem solving. Moreover, swarm intelligence considers how such distributed problem solving techniques observed in nature can be applied to real-world problems in science and engineering,

and in particular, how these distributed behaviours can be represented and implemented in computational form [3].

The mechanisms behind such impressive displays of collective movement such as birds flocking (Figure 2.1(a)) and fish schooling (Figure 2.1(b)) have been widely studied from both a biological and computational perspective. Such behaviours have been imitated in computer code for applications in computer animation [4], for example to simulate the collective movement of large collections of virtual 'entities' on screen in popular movies (for example battle scenes in the Lord of the Rings trilogy and the bats in the cave in Batman Begins). Similar algorithms have been used to simulate human crowd movements in order to optimise planning for ticket halls and gates in public transport stations, or evacuation procedures [5][6]. Flocking-based algorithms have also been used as a means to display complex real-time multidimensional data [7] in an effort to increase user readability. In recent years much research has been dedicated to applying these behaviours to robotics, to create control mechanisms for collective movement of multiple robots, which has great scope for a wide range of applications and scenarios. This particular area of research, now commonly referred to as *swarm robotics*, is discussed in more depth towards the end of this chapter.

The behaviours exhibited by social insects are particularly interesting, and ac-



(a) (b)

Figure 2.1: Examples of swarming behaviours in nature. (a) An image of a flock of birds exhibiting collective movement (image taken from *http://en.wikipedia.org/wiki/Flocking_%28behavior%29*, accessed 11th February 2011); (b) an image of a school of fish exhibiting collective movement (image taken from *http://www.borneo.com.au/diving/information/layang*, accessed 11th February 2011.)

count for a large proportion of inspiration for many computational methods in swarm intelligence. Social insects that live in colonies such as ants, bees and termites, show impressive levels of cooperative, collective problem solving. Yet as individuals these insects are relatively simple, and incapable of achieving much on their own. In many cases there does not appear to be a hierarchical decision making process, with no one individual in charge, yet the colonies appear very organised and capable of achieving an impressive range of tasks; simultaneously, robustly and efficiently. The specialisation, or division of labour, can in part be attributed to anatomical differences in individuals in the colony, for example in polymorphic species of ants, although many aspects of the collective problem solving can be attributed to Self-Organisation (SO) [2]. In terms of social insects SO refers to the phenomena of complex collective behaviour at the global, or swarm scale, emerging from interactions among individuals at the local scale, that exhibit only simple behaviour. Behaviours that exhibit SO are inherently decentralised, and it is now well known that group level decisions in social insect colonies are often the result of local interactions between individual members of the colony with one another and the environment [1].

There exists another interesting analogy between observations of swarming behaviours in nature with social insects and animals, and with statistical particle physics. Indeed computational swarm intelligence methods such as Particle Swarm Optimisation (PSO) [8], which is a population based stochastic optimisation method inspired by observations of collective movement and social behaviour of biological entities in the wild, has been used to simulate, analyse and optimise statistical physics systems [9].

The types of behaviours observed in swarms in nature, and in particular SO properties, lend themselves well to solving a multitude of real-world computational problems. The next two sections give an overview of two different areas of computational swarm intelligence specifically relevant to this thesis.

2.3 Ant Algorithms

Optimisation problems can be found in many areas of industry, as well as in the scientific community. An example of an industry based optimisation problem is

that of logistical traffic routing, where the goal may be simply to determine the most efficient route for a set of delivery vehicles to take. A more specialised and scientific based application exists in the computational biology community, where predicting the structure of proteins formed from their linear sequences is an important problem. In short, optimisation problems are wide ranging and plentiful; and as such, methods for solving these problems have been, and continue to be, a widely researched topic.

There exists a particular sub-set of optimisation problems, known as Combinatorial Optimisation (CO) problems [10], in which a positive cost value is assigned to each object in the search space by an objective function, and the goal is to find an object of minimal cost.

Algorithms that guarantee the optimal solution to CO problems exist, but often at great computational cost, especially when dealing with NP-hard CO problems. For NP-hard CO problems, the computational time required to compute the optimal solution is often too high for any practical implementation. With this, approximation algorithms have received much attention, in order to compute accurate solutions in significantly less time.

Ant algorithms are one of the most recent approximate optimisation methods to be developed. These algorithms are inspired by the bahaviour of real ants in the wild [11], and more specifically, by the indirect communication between ants within the colony via the secretion of chemical pheromones. Within the Artificial Intelligence (AI) community, ant algorithms are considered under the category of swarm intelligence [2]. As previously explained, swarm intelligence encompasses the implementation of intelligent multi-agent systems that are based on the behaviour of real world insect swarms, as a problem solving tool. Other such algorithms that have been developed are based on, for example, the behaviour of swarms of wasps and bees. Continuing from the success of the original ant algorithm proposed by Dorigo et al. in [11], further development lead to a more general purpose optimisation technique known as Ant Colony Optimisation (ACO), which was later formalised into a metaheuristic[1] in [12, 13]. Examples of other metaheuristics include simulated annealing [14], tabu search [15, 16] and local search [17].

[1] A *metaheuristic* is a set of guide lines for algorithmic development that can be applied to differing optimisation problems with little change necessary.

2.3.1 Ants in the Wild

The original inspired optimisation idea behind ant algorithms came from observations of the foraging behaviour of ants in the wild, and moreover, the phenomena known as *stigmergy*. A term introduced in 1959 by Pierre-Paul Grasse [18], stigmergy refers to the indirect communication amongst a self-organising emergent system via individuals modifying their local environment. Experimental studies carried out by Deneubourg et al. [19] explore the stigmergic nature of ant colonies, in which ants communicate indirectly by laying down pheromone trails, which ants then tend to follow. Experiments in [20] show the convergence of ant trails as a result of the tendency of ants to follow a trail that contains a higher concentration of pheromone deposit.

The Analogy

There exists a well known experiment called the "shortest bridge" experiment [20], which is often used to describe the process of stigmergy used in ant algorithms, and the analogy to the behaviour of real ants in nature. Figure 2.2 shows a diagram representing two possible paths between a nest and a food source, with the upper path shorter in length than that of the lower. Initially, as the ants leave the nest foraging for food, approximately half of the ants will follow the upper path and half the lower (Figure 2.2(a)). Due to the upper path being shorter than the lower, the ants following the upper path will reach the food source sooner (Figure 2.2(b)). Once an ant has collected food it returns to the nest by following the pheromone trail laid down on the way to the food source, again laying pheromone on the way, thus further reinforcing the pheromone trail along this path (Figure 2.2(c)). These ants will return to the nest first, reinforcing the pheromone trail along the whole of the upper path. Any ants now leaving the nest will be more likely to follow the upper path from the nest to the food due to the higher pheromone concentration along this route, likewise for any ants returning to the nest from the food source. In this way eventually most, if not all, of the ants end up following the upper path and convergence to the shortest path is achieved (Figure 2.2(d)). Essentially what we see with the foraging behaviour of ants in the wild can be described in a more computational way as an intelligent multi-agent system solution to a shortest-route optimisation problem,

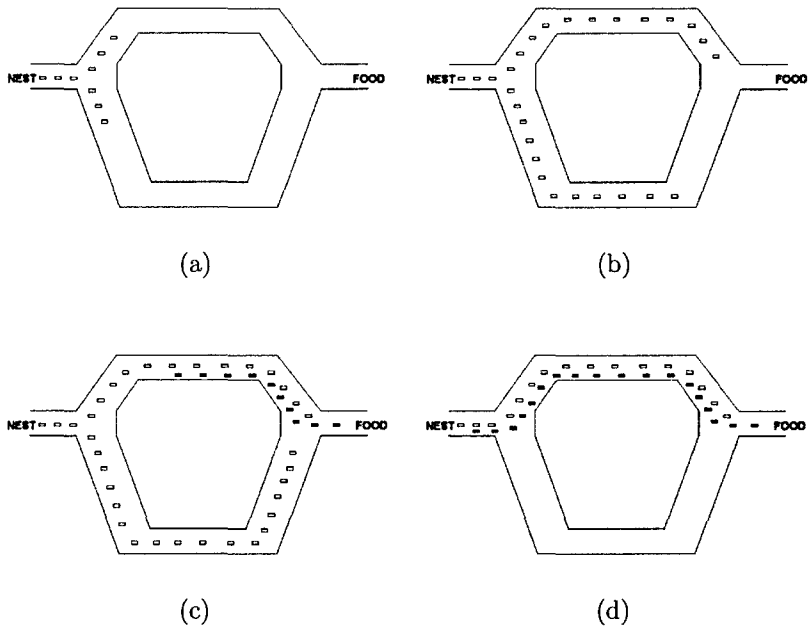


Figure 2.2: A schematic illustration of the "shortest bridge" experiment [21]

where in this case the agents in the system are ants. It is these observations that inspired the first ant algorithm [11], which was applied originally to the well known traveling salesman "benchmark" problem.

2.3.2 From Nature to Computers

The transition from the natural to artificial ant colony involves the use of simple computational agents that work cooperatively, communicating through artificial pheromone trails. In an iterative fashion[2], each ant moves from state S_i to state S_j guided by two main factors:

1. *Heuristic information:* A measure of the heuristic preference (which is application based) for moving from state S_i to state S_j. This information is known *a priori* to the algorithm run, and is not modified during.

[2]Unlike natural ants, artificial ants usually exist in a discretised world; that is, they move from discrete state to discrete state.

2. *Artificial pheromone trail(s):* A measurement of the pheromone deposition from ants previous transitions from state S_i to state S_j. In other words a measurement of the "so far" learned preference. This information is modified during the algorithm run by the artificial ants.

2.3.3 Properties of the Artificial Ant

The following details the main properties associated with the artificial ant [21]:

- Each artificial ant has an internal memory which is used to store the path followed by the ant (i.e. the previously visited states);
- Starting in an initial state $S_{initial}$ each ant tries to build a feasible solution to the given problem, moving in an iterative fashion through its search space / environment;
- The guidance factors involved in an ants movement take the form of a transition rule which is applied before every move from state S_i to state S_j. The transition rule may also include additional problem specific constraints and may utilise the ants internal memory;
- The amount of pheromone each ant deposits is governed by a problem specific pheromone update rule;
- Ants may deposit pheromones associated with states, or alternatively, with state transitions;
- Pheromone deposition may occur at every state transition during the solution construction. This is known as *online step-by-step pheromone trial update*;
- Alternatively ants may retrace their paths once a solution has been constructed and only then deposit pheromone, all along their individual paths. This is known as *online delayed pheromone update.*

2.3.4 Additional Characteristics

In addition to the above properties, artificial ants may also have characteristics to improve their performance that do not have a natural counterpart. Examples of widely used additional characteristics include *local search* [17, 22] and *candidate list* [12, 22].

Depending on the problem to be solved, daemon actions may be introduced into the algorithm. Daemon actions influence the guidance of the ants during algorithm runtime and can be used in order to speed up convergence. An example of a daemon action may be adding extra pheromone to the best solution trail so far, at the end of each iteration.

2.3.5 The Original Ant System

The first ant algorithm, named "Ant System" (AS), was developed in the nineties by Dorigo et al. [11]. As a test-bed for this algorithm the well known benchmark Traveling Salesman Problem (TSP) was used.

For a set of M towns, the TSP problem involves finding the shortest length closed tour visiting each town only once. In other words finding the shortest route to visit each town once, ending up back at the starting town. In the case of the Euclidean TSP, the path length between any given towns i and j is considered to be the Euclidean distance between i and j, such that the path length $d_{ij} = \left[(x_i - x_j)^2 + (y_i - y_j)^2\right]^{1/2}$

Each ant in the system has the following characteristics:

- An ant decides which town to go to using a transition rule that is a function of the distance to the town and the amount of pheromone present along the connecting path;
- Transitions to already visited towns are added to a *tabu list* and are not allowed;
- Once a tour is complete, the ant lays a pheromone trail along each path visited in the tour.

An *iteration* is defined here as the interval in $(t, t+1)$ where each of the N ants moves once. An *epoch* is then defined to be every n iterations, when each ant has completed a tour. After each epoch the pheromone intensity trails are updated according to the following formula:

$$\tau_{ij}(t+n) = (1-\rho) \cdot \tau_{ij}(t) + \sum_{k=1}^{N} \Delta\tau_{ij}^{k} \tag{2.1}$$

where $\rho \in (0, 1]$ is the evaporation rate and $\Delta\tau_{ij}^{k}$ is the quantity of pheromone laid on path (i, j) by the kth ant between time t and $t+n$, and is given by:

$$\Delta\tau_{ij}^{k} = \begin{cases} Q/L_k & \text{if ant } k \text{ used edge } (i,j) \text{ in its tour,} \\ 0 & \text{otherwise} \end{cases} \tag{2.2}$$

where Q is a constant and L_k is the tour length of the kth ant.

The heuristic information in this case is called the *visibility*, η_{ij}, and is defined as the quantity $1/d_{ij}$. As opposed to the pheromone trail, this quantity is not modified during the algorithm run. A *tabu list* is implemented as a growing vector containing the list of the previous and current visited towns. $tabu_k(s)$ gives the sth town visited by the kth ant in the current tour.

The probability of the kth ant making the transition from town i to town j is given by:

$$p_{ij}^{k}(t) = \begin{cases} \dfrac{[\tau_{ij}(t)]^{\alpha} \cdot [\eta_{ij}]^{\beta}}{\sum\limits_{k \in allowed_k} [\tau_{ik}(t)]^{\alpha} \cdot [\eta_{ik}]^{\beta}} & \text{if } j \in allowed_k \\ 0 & \text{otherwise} \end{cases} \tag{2.3}$$

where $allowed_k = \{M - tabu_k\}$ and α and β control the relative importance of the pheromone trail and visibility respectively.

2.3.6 Ant Algorithms as Computational Optimisation Techniques

The success of AS as applied to the classic TSP test-bed led to the development of the ACO metaheuristic [12, 13]. The ACO metaheuristic was developed to describe, in a more general way, the overall method of solving combinatorial problems by approximate solutions based on the generic behaviour of natural ants. In this way, all specific applications of the ACO metaheuristic may be described as ant algorithms, however, ant algorithms are not confined to ACO.

The ACO Metaheuristic

ACO is structured into three main functions (Algorithm 2.1). *AntSolutionsConstruct()* performs the solution construction process as described previously. Artificial ants move through adjacent states of a problem according to a transition rule, iteratively building solutions. *PheromoneUpdate()* performs pheromone trail updates. This may involve updating the pheromone trails once complete solutions have been built, or updating after each iteration.

In addition to pheromone trail reinforcement, ACO also includes pheromone trail evaporation. Evaporation of the pheromone trails is included to help ants 'forget' bad solutions that were learned early on in the algorithm run. Implementation could be as simple as reducing all pheromone trails by a set amount after each epoch. *DeamonActions()* is an optional step in the algorithm which involves applying additional updates from a global perspective (there exists no natural counterpart). An example could be applying additional pheromone reinforcement to the best solution generated (known as *offline pheromone trail update*).

Algorithm 2.1: Pseudo-code for the Ant Colony Optimisation Metaheuristic.

```
ParameterInitialisation
while Termination conditions not met do
   for ScheduleActivities do
      AntSolutionsConstruct()
      PheromoneUpdate()
      DeamonActions() {optional}
   end for ScheduleActivities
end while
```

2.3.7 Past and Present Applications

Ant algorithms have been successfully applied to many different problems in many different areas of scientific research and development, and are being used increasingly in industry, including successful implementation into 'real-world' deployed systems. This section details some of the more 'typical' applications to which ant algorithms and the ACO metaheuristic have been applied.

As previously stated, ant algorithms are particularly well suited to NP-hard combinatorial optimisation problems, and indeed the first problem to which the original AS algorithm was applied was the classic TSP problem, which itself is NP-hard. The TSP problem also has the characteristic of being a constrained shortest path problem, which is another important type of problem to which ant algorithms lend themselves particularly well to.

After the initial test-bed that was the TSP problem, ant algorithms were applied to other NP-hard problems such as the Quadratic Assignment problem (QAP) [11, 23] and the Job-Shop Scheduling (JSP) problem [11, 24]. The original AS was applied to the QAP and JSP to show its robustness, and shortly thereafter, improved ant algorithms were developed to solve these problems specifically. One of the next major applications for which ant algorithms were developed was the dynamic problem of data network routing [25, 26], a shortest path problem where properties of the system such as node availability vary over time. The introduction of dynamic properties to the problem can significantly increase the difficulty of achieving the desired result, and although algorithms that employ SO have often proved to be particularly good at solving dynamic problems, explicit ways of dealing with adaptive problems are often required, and this remains an open problem.

Other main applications of ant algorithms include the vehicle routing problem [27, 28], graph colouring [29] and set covering [30]. The vehicle routing problem in particular has strong links with industry; companies have developed tools built on ant algorithms that have been successfully deployed in real world working applications for various vehicle routing scenarios. Two specific example applications here include DyvOil and AntRoute [31, 32].

DyvOil is an application which supports planning the sales and distribution process of fuel oil. An *off-line* module of DyvOil, which solves static vehicle routing

problems by using ACO algorithms, is used every evening to plan vehicle tours for the next day. This off-line planning module has been tested in a real world setting with a leading Swiss fuel oil distribution company called Pina Petroli. An increase of 20% upto 30% of vehicle routing performance has been observed using this technique over human generated plans [32]. Efficiency has also been shown when implementing another ant algorithm called the Ant Colony System (ACS) [22, 33], for dynamic vehicle routing problems in a similar scenario.

AntRoute is a similar application which has been designed for use in the supply chain of the Switzerland based supermarket known as Migros. An algorithm based on MACS-VRPTW [28] is used to compute the tours of the distribution vehicles to supermarkets across Switzerland, and has again shown to outperform human planners.

The scope of applications in more recent years has increased significantly, with many more novel implementations of ant algorithms being developed. Aside from the highly specialised algorithms there are a number of broader areas of study to which ant algorithms are being applied. Examples of broad application areas typically seen in more recent trends include continuous optimisation [34, 35] and parallel processing implementations [36, 37]. Continuous optimisation is a particularly interesting development for ant algorithms, since they were originally developed for discrete optimisation problems [13], and indeed it is this class of problem for which the ACO metaheuristic was developed.

A large proportion of developed ant algorithms and their applications (including those mentioned so far) are somewhat abstract from actual ants in nature, especially for example in typically route optimisation problems, where the ant agents do not occupy physical open environments, instead they are confined to traversing a set number of point to point routes.

In recent years a number of problems in the area of image processing have been tackled using ant algorithms. Some of these problems allow for a more open approach to the ant colony representation, in terms of freedom of movement. For example, ant algorithms have been developed for image boundary detection [38, 39, 40], and for visual saliency detection [41], which involve a colony of virtual ant agents being embedded in a digital image environment, where the ant agents are entities which move around in a virtual world landscape which is discretised

into pixel locations.

These types of applications allow for a different perspective on computational SO, and arguably a closer comparison the natural counterpart. Indeed other applications focus explicitly on studying SO of computational, or virtual swarms. For example in [42, 43] the authors simulate virtual swarms to investigate how swarms of insect-like agents form patterns and build cognitive maps by creating a network of trails, and in [44] a virtual environment is developed for testing ant-like models.

These more 'open environment' problems are central to the research of this thesis, and will be discussed in more detail throughout.

2.3.8 Ant Algorithms as an Alternative to Other Machine Learning Techniques

Certain forms of ant algorithms, and in particular ACO algorithms, show similarities with other machine learning techniques such as evolutionary computing and neural networks [13, 21]. Below is a brief summary of some of the observed similarities and differences, highlighting where ant algorithms may be used in place of other well established machine learning techniques.

Both evolutionary computing and the ACO metaheuristic employ a population of individuals to incrementally build more suitable solutions to a given problem, by building on the solutions of previous populations. The main difference between the two is that with evolutionary computing, the knowledge of the problem is contained only within the current population, whereas the ACO metaheuristic continually uses information from a number of previous generation populations as its knowledge base, in the form of pheromone trails. There does exist however, specific forms of evolutionary computing algorithms that are more in line with the characteristics of the ACO metaheuristic. Further comparison with these specific algorithms can be found in [13] and [21].

Artificial neural networks [45], like ant algorithms, are biologically inspired learning systems. Problems solved by both ant algorithms and neural networks can often be represented graphically by a set of connecting nodes. What are considered as 'states' in an ant algorithm are analogous with 'neurons' in a neural

network. The local neighbourhood structure around a given state is then equivalent to the set of synaptic-like links exiting the specific neuron [13]. The ants are then associated with the input signals that propagate through the neural network. As connecting paths in ant algorithms receive more pheromone the more they are used, the more a synapse is used, the more its strength is increased. Also as with ant algorithms, synapses used to create better solutions are reinforced more than others.

The similarities seen in such algorithms makes it perhaps unsurprising that they are often used to solve similar types of problems. The differences however, mean that the performances of these algorithms differ from each other for different specific tasks. In many works such as the original AS [11] and ACS [22], ant algorithms have been tested on benchmark problems such as the TSP, and have yielded better solutions (i.e. closer to the true optimum) over other heuristic approaches. Also, in tests for rate of convergence towards the optimum solution, in many cases ant algorithms have proved to converge faster than other heuristic approaches. In other cases [46, 47] it has been shown that by introducing certain elements particular to ant algorithms, such as the stigmergic effect via synthetic pheromones, to other machine learning techniques such a *quality learning* (Q-learning), this has increased performance over the original algorithm design (see Section 2.3.9).

2.3.9 Ant Algorithms as a Counterpart to Other Machine Learning Techniques

Two major machine learning techniques that have been used in conjunction with ant algorithms are Q-learning [48] and *genetic algorithms* [49, 50, 51].

Ant-algorithms and Q-learning

Q-learning falls within the category of reinforcement learning, which is a subset of machine learning to which one could also relate the concept of ant algorithms to. Reinforcement learning involves agents learning by trial and error which actions are best to take in their current environment in order to achieve their goals [45]. In a training phase, each time an agent performs an action in its

environment, it may receive a reward or penalty reflecting the desirability of the outcome of the action performed. The goal of the agent is then to choose sequences of actions that maximise the cumulative reward. More specifically, Q-learning involves learning an action-value function (where the action values are known as Q-values), which measures the utility (or 'goodness') of taking a given action in a given state within the environment. At each time step, t, an agent in state s_i takes an action a which takes it to a new state s_{i+1}. The agent then receives a reward[3] r depending on the new state. The Q-values for each state-action pair are updated at each time step until convergence between successive Q-values approaches zero, using the following equation:

$$\begin{aligned} Q_n(s_t, a) \leftarrow \; & (1-\alpha_n) Q_{n-1}(s_t, a) \\ & +\alpha_n \left[r_t + \gamma \max_{a'} Q_{n-1}(s_{t+1}, a')\right] \end{aligned} \tag{2.4}$$

and

$$\alpha_n = \frac{1}{1 + \text{visits}_n(s_t, a)} \tag{2.5}$$

where γ is the discount factor, a' is the action that maximises Q, and visits(s_t, a) is the total number of times the given state-action pair have previously been visited.

An algorithm inspired by the original AS, called *Ant-Q*, was developed by Dorigo and Gambardella [52, 53]. This algorithm has many similarities with the Q-learning algorithm, but also a few key differences; mainly that Ant-Q, unlike typical Q-learning algorithms, involves using multiple agents. These agents communicate, exchanging information in the form of AQ-values (which is the analogue of Q-values in Q-learning). As with AS, the Ant-Q algorithm was developed originally for the classic benchmark problem TSP. For TSP, $AQ(r, s)$ is the Ant-Q value associated with the path (r, s) between cities. $HE(r, s)$ is a heuristic value associated to path (r, s), which for TSP is the inverse of distance. k is an agent whose task it is to complete a closed tour of all cities, and associated with each agent k there is a list, $J_k(r)$, of all cities still to be visited, where r is the current city. This list acts as a kind of memory, and is another important

[3]The reward is usually discounted into the future.

difference between Ant-Q and Q-learning. The state transition rule for an agent k in city r is as follows:

$$s = \begin{cases} argmax_{u \in J_k(r)} \left\{ [AQ(r,u)]^{\delta} \cdot [HE(r,u)]^{\beta} \right\} & \text{if } q \leq q_0 \\ S & \text{otherwise} \end{cases} \tag{2.6}$$

where α and β are parameters which weigh the relative importance of the learned AQ-values and the heuristic values, q is a uniform probability randomly chosen value in $[0,1]$, q_0 $(0 \leq q_0 \leq 1)$ is a parameter such that the higher q_0 the smaller the probability to make a random choice, and S is a random variable selected according to a probability distribution given by the function of the $AQ(r,u)$'s and $HE(r,u)$'s, with $u \in J_r(r)$.

The update rule for the AQ-values is as follows:

$$\begin{aligned} AQ(r,s) \leftarrow \; & (1-\alpha) \cdot AQ(r,s) \\ & +\alpha \left(\Delta AQ(r,s) + \gamma \cdot Max_{z \in J_k(s)} AQ(s,z) \right) \end{aligned} \tag{2.7}$$

where α and γ are the learning step and discount factor respectively. This update rule is the same as in Q-learning except that the set of available actions in state s, i.e. the set $J_k(s)$, is a function of the previous history of agent k.

This approach adapts the idea of ant-algorithms to that of Q-learning, and it is important to note that Ant-Q does not make use of artificial pheromones. A different approach has been developed in [46, 47], which adapts the idea of Q-learning to that of ant-algorithms, by introducing the use of artificial pheromones into multi-agent Q-learning.

The pheromone Q-learning (Phe-Q) algorithm uses the same Q-value update function as in Equation 2.4, but with an additional factor to be maximised, called the belief factor. The belief factor is a function of the synthetic pheromone concentration on the trial and reflects the extent to which an agent will take into account the information laid down by other agents from the same cooperating group. The belief factor is the ratio between the sum of actual pheromone concentrations in the current plus surrounding states, and the sum of the maximum possible pheromone concentration in the current plus surrounding states, and is given by:

$$B(s,a) = \frac{\sum_{s \in N_a} \Phi(s)}{\sum_{s \in N_a} \Phi \max(\sigma)} \tag{2.8}$$

where $\Phi(s)$ is the pheromone concentration at state s in the environment, and N_a is the set of surrounding states for a chosen action a.

With the addition of the belief factor the Q-learning update function then becomes:

$$\begin{aligned} Q_n(s_t, a) \leftarrow \quad & (1 - \alpha_n) Q_{n-1}(s_t, a) \\ & + \alpha_n \{ r_t + \gamma \max_{a'} [Q_{n-1}(s_{t+1}, a') + \xi B(s_{t+1}, a')] \} \end{aligned} \tag{2.9}$$

where the parameter ξ is a sigmoid function of time epochs ≥ 0, such that it increases with the number of agents who successfully complete the given task.

In this example, where a key feature of the ant algorithms has been coupled with another established machine learning technique, improvements in the performance compared to the same algorithm without the additional ant algorithm feature have been shown [47]. This is a clear example of how 'hybrid' algorithms, bringing elements of different machine learning techniques together, can produce superior performing algorithms.

Ant Algorithms with Genetic Algorithms

Genetic algorithms are a branch of the wider field of study known as evolutionary computing, which is another subset of machine learning. The study of evolutionary computing, which began in the 1950s and 1960s, is inspired by biological evolution, where systems are developed that evolve a population of candidate solutions to a given problem, using operators based on natural selection and natural genetic variation [54]. With the biological / nature inspired properties, genetic algorithms share similarities with many swarm intelligence algorithms.

The basic genetic algorithm involves iteratively updating a population of hypotheses. At each iteration, all individual hypothesis are evaluated against a problem specific fitness function. A new population is then created, first by probabilistically selecting a proportion of the most fit individuals. The remaining proportion of the new population is then created by probabilistically selecting

pairs of the most fit individuals, and creating new offspring hypothesis by applying genetic crossover operators. In addition to this, genetic mutation operators may also be applied to random members of the new population before carrying it to the next iteration.

A genetic algorithm is used in [50] and [51] to automatically adapt the control parameters of ant algorithms. Choosing appropriate control parameters is vital to the success of the algorithm for the given application or problem, making sure an appropriate balance is met between exploration of unexplored areas of the search space / environment, and exploitation of previously learned preferences for states to visit, within the context of the problem. The use of genetic algorithms as described below could provide a useful automated alternative to the often used lengthly trial and error process of selecting appropriate control parameters, as well as potentially providing an adaptive on-line balance between exploration and exploitation within the search space / environment.

In [51], the control parameters of the ant algorithm ACS are genetically adapted for the TSP as follows. Each ant is encoded with the algorithm control parameters in the form of bit strings (chromosomes). All chromosomes are initially randomised. At each iteration four ants are chosen via a tournament selection method. All four selected ants then construct their TSP tours. The algorithm then checks to see if a new tour has been found. The global update of the pheromone trail is carried out by the ant that produced the best overall tour, using that ants encoded ρ parameter. The fitness of each ant is calculated as the length of the tour found by that ant. The two best selected ants are crossed over to produce two offspring, which are then mutated. The worst two selected ants are then replaced by the two offspring. The idea here is to put pressure on the population to improve its performance, while keeping the population size constant.

Although there are clear benefits to be gained by using genetic algorithms with ant algorithms, their introduction can also bring about some problems. In [51] it was shown that due to the high variability amongst the population of ants, many ants became trapped in local minima, with pheromone trails from good solutions being erased by worse performing ants, with no guarantee of finding the optimal solution. Conversely however, the introduction of the genetic algorithm was shown to increase the speed of convergence towards good solutions; a desirable

characteristic for large problems where it is not desired, or indeed may not be possible to find the optimal solution.

As previously stated, choosing appropriate control parameters for a given ant algorithm is an important factor in its success with a given problem. In problems where the characteristics may change in some way over time, or problems that present themselves in different ways under different circumstances, it would be very advantageous to have some degree of automation for the fine tuning of the control parameters to suit the changing problem characteristics. The above presented method of using a genetic algorithm is one way of achieving this, and further research into similar methods could prove valuable in many applications of ant algorithms, especially those of a dynamic nature.

One potential limitation of this approach however is the time taken to optimise the parameters, resulting in this approach often being used as an 'off-line' optimisation method, where the parameters are optimised prior to a live experiment run. Given this limitation, there is significant motivation to pursue 'on-line' methods of autonomous parameter optimisation, where the optimisation occurs in real-time, as the experiment is running, enabling rapid adaptation to dynamic changing problem characteristics. Such on-line methods are explored later in this thesis (specifically in chapters 4 and 6).

2.3.10 Other Ant Algorithm Approaches

The ACO metaheuristic, and in particular the AS algorithm have been described above, as they are particularly relevant to the first two research chapters of this thesis. It is important to mention however, that the behaviour of ants in the wild has been studied computationally using different approaches. As mentioned in Section 2.3.7, the authors in [42, 43] simulate virtual swarms to investigate how swarms of insect-like agents form patterns and build cognitive maps by creating a network of trails.

Similar to the ACO and AS, and consistent with observations from nature, their approach involves the ant agents following pheromone trial gradients with stochastic functions. A major difference is the use of a weighting factor to the movement decision making process, which takes into account the orientation and motion of the ant agents. A rule base is defined such that very sharp turns are

much less likely than turns through small angles, resulting in a probabilistic bias in the forward direction.

Given the more open, unrestricted movement allowed, the direction bias effectively replaces the problem specific state heuristic information of the ACO metaheuristic. This interesting study of elementary SO concluded that only simple pheromone trail following is required to produce the evolution of complex patterns of organised flow of social insect traffic. This approach has also been used for studying pattern formation in digital image media [38], which is of particular interest to the work of this thesis, and is discussed in greater detail in chapter 3.

2.4 Swarm Robotics

Swarm robotics is a relatively new area of research and development that has emerged from the swarm intelligence paradigm [2]. Much of the inspiration behind swarm intelligence based systems comes from observations of biological swarms in nature, and in particular the self-organising, emergent properties they exhibit, for example the foraging behaviours of a colony of ants [22][18], as discussed above.

There are many similarities between swarm robotics algorithms and other swarm intelligence algorithms such as the above described ant algorithms. Swarm robotics algorithms are however tailored for controlling real physical agents, operating in real world environments. With these considerations comes many additional challenges, which can restrict potential solutions that might otherwise be available when considering virtual swarms in virtual, or abstract environments. Overcoming these additional challenges provides much focus and motivation for further research in this area of swarm intelligence, which shall be discussed in more detail in chapters 5 and 6.

Swarm robotics largely considers systems of multiple, relatively simple, homogeneous robots, with local and limited sensing and communication abilities, that work collectively to achieve some unified goal [55]. The use of multiple robots necessitates the system to be scalable in nature, and the local and limited sensing and communication abilities provide a focus on a decentralised approach. A decentralised system can greatly improve robustness to robot failure, and the use

of relatively simple robots offers the potential for the robots to be relatively small and inexpensive, thus increasing robot portability and expendability.

Three main properties typically stated in relation to swarm robotics (which again come from observations of natural swarms) are: *robustness*, *flexibility* and *scalability* [56][57]. Robustness relates to the swarms ability to still function in the presence of disturbances from the environment and/or failed robots (including full and partial hardware failures). Flexibility relates to the ability to perform under varying circumstances, such as different environment conditions, or for multiple scenarios and tasks. Scalability relates to the ability to operate from small to large sized swarms without significant performance degradation. These are desired properties and appear in varying degrees for different swarm robotics methods. Recent work in particular has challenged the assumption that these properties are inherent in swarm robotics methods [58][59].

Given that typically the robots will be relatively simple, it follows that the task or tasks will generally be difficult, if not impossible, to be carried out by a single robot, and that the use of multiple robots will provide a substantial increase in performance in achieving the given task or tasks.

Research in the area of swarm robotics, and indeed multi-robot systems in general, remains very active. Many different methods are being developed to achieve autonomous multi-robot control, for a wide range of applications and robotic platforms. Advancements in robotics hardware are providing more and more options for smaller scale robots than were previously available, which provides more possibilities for swarm robotics systems to be developed to a deployment ready level. The attributes associated with swarm robotics systems, as summarised above, have advantages across a range of applications, including, but not limited to; security and defence, environmental monitoring, search and rescue, structure inspection and cleaning. Robotic platforms used ranges from sub-aqua to ground based, aerial and even space-borne, ranging in size from micro-bots to satellites, and numbers from tens to hundreds and upwards.

2.4.1 Coordination and Control Mechanisms

There are numerous different approaches to swarm robotics coordination and control, taking inspiration from a multitude of different areas, predominantly

from biological and natural observed phenomena.

Stigmergy in Swarm Robotics

From the swarm intelligence paradigm, the concept of stigmergy [18] has been used, for example in [60][61], to achieve multi-robot coordination and control by using artificial pheromones to guide the robots. Figure 2.3 shows a photo of three robots pursuing a luminous trail (from [61]). The use of tangible pheromones for stigmergy in swarm robotics has been studied for example in [62][63] where the authors have developed a set of ant-like robots that can deposit and sense ethanol for use as pheromones, to create self-organisational cooperative behaviours. Despite promising experimentation results, the current implementation has limited use in real-world scenarios due to deployability issues.

Another strategy of employing stigmergy in swarm robotics systems is the use of individual robots in the swarm as a physical stigmergic medium [64]. This type of stigmergy has been explored in, for example [65], where virtual pheromones are implemented as simple local communications between physical robots, including the encoding of diffusion gradients and decay. For example, virtual pheromones are transmitted with a known intensity, and their signal strength decreases linearly with distance.

Virtual Forces and Physics Laws

Another approach is the well established potential fields method [66]. This method is widely used for robot control/coordination, and may be considered as a form of stigmergy as, similar to to that of using virtual pheromones, it involves computing virtual force vector-fields, resultant from the environment and/or fellow robots, which are then used to guide the robots movements.

The use of virtual pheromones and potential/force vector fields mimics to varying extents the use of real, tangible pheromones for indirect communication, but without the practical implementation issues.

Another similar method uses physics based laws to compute local artificial forces [67][68] to guide the robots movements. This type of approach draws on another equally interesting analogy with nature in the form of particle physics

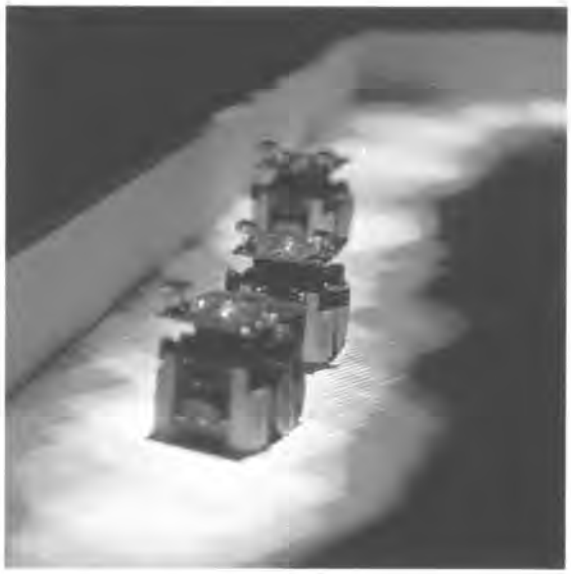


Figure 2.3: A photo of three robots using stigmergy, pursuing a luminous trail. Photo from [61].

swarms. In its most simple form, at an abstract level, this type of system acts as a molecular dynamics ($F = ma$) simulation, treating the robots as physical particles. Algorithm 2.2 gives simplified pseudo-code of a typical implementation of the particle physics approach to robot swarm control. Typical requirements are that each individual robot can measure the relative range and bearing to neighbouring robots, an advantage being that this method does not require direct inter-robot communications. The range and bearing measurements are then used in conjunction with a physics based law, a simple example being Newtons Law of Gravitation; $F = Gm_1m_2/r^2$, where G is the gravitational constant, m is the point mass (typically equal to 1 for swarm robotics implementations), and r is the measured distance to the neighbouring robot. Various conditions can then be imposed (depending on which particular Physics Law is used) to manipulate the resultant movements of the robots, for example balancing attractive and repulsive forces in order to produce specific formations and collective coordination behaviours, and ultimately produce a process of swarm SO.

An advantage of the latter, physics-based method is that it does not require the computation of a global vector field of artificial forces, rather each robot computes the forces it experiences locally, thus offering a more distributed approach. The property of not requiring inter-robot communication, along with the strong

Algorithm 2.2: Pseudo-code for the particle physics approach to robot swarm control.

```
for all neighbouring robots within range do
    Measure the range/bearing to neighbouring robot
    Calculate the force vector due to neighbouring robot
end for
Calculate the total force vector from all neighbouring robots
Convert to displacement vector
Calculate required actuator velocities
```

theoretical background of the physics which is used in these systems, makes for great motivation in pursuing this form of swarm SO, and indeed this is covered in much greater detail in chapters 5 and 6.

Flocking

The concept of flocking was realised in computational form in [4] where a computerised model of coordinated animal motion such as bird flocks and fish schools was created. These computerised agents, known as 'boids,' are controlled in a basic flocking model consisting of three simple steering behaviours:

1. Separation - to avoid agents clustering and collisions.
2. Alignment - to maintain a common heading with flockmates.
3. Cohesion - To move towards the average position of local flockmates.

Figure 2.4 shows schematic representations of these three behaviours.

This well established nature inspired technique has been used in many variations to achieve multi-robot coordination and control (for example [69][70][71]), and

(a) (b) (c)

Figure 2.4: A schematic representation of the boids flocking behaviours; (a) separation; (b) alignment and (c) cohesion. Images from *http://www.red3d.com/cwr/boids/*, last accessed 8th Febraury 2011.

often also uses virtual forces to achieve the desired behaviours. One of the potential limitations of the typical flocking approach is that it requires inter-robot communication, for example to share heading information [70] or velocity [71] between the robots.

Other Methods

A more engineering based method is reported in [72], which uses a kinematics control approach and merges a number of elementary behaviours into one final behaviour to facilitate the entrapment and escorting of a target by multiple robots.

The latter mentioned approach, along with the methods using physics based laws to compute virtual forces, tend more towards a deterministic nature, as opposed to the stochastic nature of the ant-algorithm approaches of [60][61]. The stochastic nature of many swarm intelligence methods makes them particularly good at solving unpredictable and dynamic problems (in [73] stochasticity is introduced in a Lyapunov-based flocking controller and shown to improve performance). This can also however make it difficult to fully predict the behaviour of the solution itself [74], which, in particular safety critical applications, might be an undesirable feature.

2.4.2 System Architectures

Swarm robotics systems are designed in many different ways, for different reasons. The choice of design architecture might be related to the particular problem being tackled, or the application to which the system will be applied, or might simply be a consequence of methods being employed or the focus of the research.

Many works focus on developing a single swarm behaviour, either to solve a specific task, or to improve a particular existing behaviour or propose a new one. This does not necessarily mean such behaviours must be used alone, and in fact in order to solve more complicated problems it is often the case that several different behaviours will be used. Subsumption architectures have been used in mobile robotics for some time [75], allowing modular-like layered multi-behaviour systems. A number of behaviours were applied to a group of mobile

robots for studying swarm-like interactions using a subsumption-like architecture in [76]. Finite State Machines (FSM), or Finite State Automata, (FSA), are also commonly used for multi-behaviour swarm robotics. This type of system architecture allows simple rule-based transitions between different behaviours, represented as states. State transitions are often triggered by external input such as sensory information. Probabilistic Finite State Machines (PFSM) are FSMs where the transitions between states are triggered with certain probabilities. An adaptive foraging method is presented in [77] which uses a FSM as the system architecture.

System architectures will also depend on the choice, or restrictions, of the hardware being considered. For example, different methods of communication might require alternative system architectures. Bearing in mind the nature inspired characteristics of swarm robotics, and the focus on relatively simple robotics hardware, it is not surprising that often (although certainly not exclusively) swarm robotics system architectures themselves are relatively simple in nature.

2.4.3 Learning and Adaptation

Swarm robotics systems tend to inherently exhibit a level of adaptation due to the very nature of SO and the nature inspired characteristics of swarm robotics. It can in many cases be advantageous to explicitly implement additional methods of adaption and/or learning into swarm robotics behaviours and systems. Methods of adaptation tend to be more *online* in that the adaptation occurs in real-time, as the system is running, and the swarm is 'deployed.' In [77] a number of simple adaptation rules are introduced to a swarm robotics system for the task of foraging. The rules are based on local sensing and communications and allow the system to dynamically change the number of foraging agents, which results in the swarm being more energy efficient.

There are a multitude of learning methods commonly used to optimise behaviours, for example learning optimal parameter settings, or for evolving the behaviours themselves. Methods of learning range from traditional reinforcement learning methods such as the previously mentioned Q-learning, to evolutionary methods such as neural networks and the previously mentioned GAs, to population based methods such as PSO and more. PSO has been used as a tool to optimise a

sequence of controls in order to achieve a desired formation within the context of multi-robot coordination [78][79], and used to model the characteristics of a multi-robot search process [80]. A PSO algorithm is physically embedded into a small number of robots in [81], where each robot represents a particle in the PSO algorithm, and the robots search for a target within a given search environment. In [82] a physically embedded GA is used for parallel learning in a 2-robot system where the robots regularly communicate genetics strings and fitness to one another. In [83] the authors use reinforcement learning to learn control policies for the specific behaviours of corridor following and obstacle avoidance. Learning by imitation is another popular approach, where the learning strategies mimic the ways in which humans learn many basic tasks. In [84] a method of learning by imitation is used to teach a MRS to play robot football, by observing games of other teams and learning new states and actions.

2.4.4 Prominent Problems Tackled

As previously stated, swarm robotics has been applied to numerous application areas, within which there are a number of specific problems to which swarm robotics is typically applied (and correspondingly, is particularly well suited to). These problems share similarities to those observed in nature, where natural swarms can be seen to solve such problems with sophisticated simplicity. In some cases these problems may be solved with relative ease in multi-robot systems employing such properties as global direct communications, however when considering relatively simple robotic agents the problems become much more challenging. Below is an overview of some of the more prominent problems typically addressed with swarm robotics to create corresponding behaviours.

Cooperative Movement

Cooperative movement encompasses a number of sub problems prominent in swarm robotics, and in one form or another is at the centre of all swarm robotics problems. Broadly speaking, it involves the collective, coordinated movement of the swarm; attributes commonly associated with the 'flocking' behaviour. Cooperative movement is used in such tasks as foraging, cooperative searching and cooperative transport.

Dispersion

Dispersion involves the spreading out of the swarm into, or across a given area or environment. This could be in the form of, for example, autonomous deployment; to provide optimal distributed sensor coverage of a given environment; to search a given environment; or to facilitate a distributed communications network.

In [85] the authors propose an algorithm for distributed coverage, where the environment is decomposed in to cells represented using an adjacency graph, which is incremently constructed and shared amongst the robots. This approach does however assume unrestricted communications between the robots. In [86] a bio-inspired approach to the deployment of a homogeneous robot swarm to multiple sites in the environment is proposed. A quorum based stochastic control policy is proposed that enables the swarm to distribute among multiple sites in a specific ratio. The algorithm is decentralised and requires no explicit inter-robot communication. The authors in [87] use wireless signal intensity as a rough approximation of inter-robot distance to assist a large swarm of small robots in dispersion, and in [88] the authors demonstrate an algorithm that uses only laser range data and range limited communications to attempt to maximise coverage while minimising loss of connectivity during deployment. A swarm of fifty-six real robots were used in [89] to demonstrate a multi-behaviour dispersion algorithm. A photo of the swarm is given in Figure 2.5.

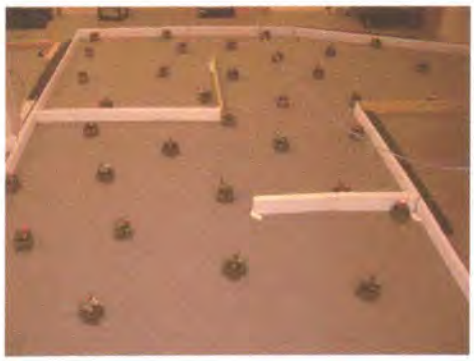


Figure 2.5: An example of a robot swarm dispersing into an environment (image taken from [89]).

Aggregation

Aggregation is essentially the opposite of dispersion, where the aim is for the swarm to gather at a common location. It is a behaviour which is useful in other behaviours such as collective movement and self-assembly, with many examples seen in nature.

In [90] the authors investigate the problem of gathering a swarm of robotic agents on the plane using a crude range limited sensing capability that can only classify neighbouring agents as either near or far. In [91] a number of swarm robotics aggregation methods are presented, including evolutionary methods and multi-behaviour approaches. In [92] a potential fields approach to swarm aggregation is proposed and demonstrated for multiple kinematic robots.

Self-Assembly

Self-assembly is where multiple simple robotic agents connect together to create a more complex structure which is then able to perform some additional task or tasks that the individuals would not be able to perform apart. An example from nature is where ants join together and use their connected bodies to create a bridge structure over a gap in the environment to allow fellow ants to cross. Replicating such behaviours in swarm robotics would be useful in scenarios such as search and rescue, where a swarm of robots could assemble into a structure to be able to climb over obstacles and disassemble into smaller components to move through small gaps, in a collapsed building for example.

These types of behaviours have been studied in swarm robotics in, for example [93][94][95]. Figure 2.6 shows an example of how a robot swarm might form a 'multi-robot organism' to collectively pass a barrier in the environment (image taken from [95]).

Formation Control

In formation control the goal is for the robots to collectively manoeuvre into a formation pattern, which could be arbitrary or of specific geometry. The collective, cooperative movement of robots in a swarm is central to much of the work in this

Figure 2.6: An example of how a robot swarm might form a 'multi-robot organism' to collectively pass a barrier in the environment (image taken from [95]).

area of research. The ability to autonomously coordinate the spatial distribution of the robots for a specific task or scenario is an important function to many swarm robotics systems. The additional challenge of dynamically managing the formation structure of the swarm for changing environment or problem characteristics remains an open problem of particular interest, and solutions to these problems provide useful behaviours relevant to numerous applications, scenarios and related problems.

In [96] a multi-state algorithm is proposed which utilises artificial forces to facilitate the formation of triangular lattice structures based on the *Physicomimetics* framework presented in [67]. The algorithm was demonstrated on real robots as well as in 3D simulations. In [97] and [72] the authors present a framework for multi-robot formation control based on the decomposition of the problem into a number of elementary tasks which are combined in a singularity-robust task-priority inverse kinematics method. The framework is demonstrated in simulation and on real robots for formation control problems including escorting a target; where the robots create a formation around a given target, and maintain formation as the target moves through the environment. An example of the robots creating a formation around the 'target' is given in Figure 2.7. The demonstrated approach does however assume global knowledge of the targets position. In [69] a number of artificial force control laws are used to facilitate SO of a 'flock' of UAVs into specific formations to provide optimum ground coverage for specific scenarios. The control laws are demonstrated in simulation for a moving line formation and a circular formation.

Figure 2.7: An example of a robot swarm creating a formation around a 'target' (image taken from [72]).

Potential fields methods along with sliding mode techniques have been used by Gazi *et. al.* [98][99] to facilitate swarm formation control of non-holonomic agents, which forces the motion of the agents along the gradient field of the potential function generated based on the individual distance requirements of the swarm agents. In [100] the authors use wireless local communications between robots to facilitate swarm formation control, inspired by ancient Roman surveying methods to align the robots based on restricted indirect observatory communication.

2.5 Related Work Discussion

The related work described above has focused on two main areas of swarm intelligence: ant algorithms and swarm robotics. This background knowledge is aimed to provide a broader perspective of the research of this thesis. The research of this thesis focuses on embedded world swarms, as opposed to the traditionally more 'abstract' swarm implementations described above for the ant algorithms. This focus on swarm agents embedded into environment landscapes links the two strands of swarm intelligence, and shares the common research goal of better understanding SO in this context, as well as providing two distinctly different yet equally challenging scenarios with specific problems to solve. Additional more focused background information is given at the beginning of each chapter to provide more context specifically relevant to the research carried out therein.

Chapter 3

Swarm Self-Organisation and Pattern Formation in Digital Image Environments

3.1 Introduction

In this chapter the role of self-organisation in artificial swarm pattern formation is explored. In particular, a study is carried out on the use of ant-algorithms for facilitating emergent self-organisation and pattern formation at the swarm level, in response to local agent-agent and agent-environment interactions, by imposing simple decision making rules on simple artificial agents. The self-organisation process studied in this chapter, and indeed throughout this thesis, can be likened to the use of *templates*, where a template is a pattern used to construct another pattern [2]. An example in nature is where an ant colony builds a pattern of walls around a brood pile in the environment, which becomes the nest. Similar environment-based templates are used to stimulate self-organised pattern formation in this chapter.

Although such methods are not entirely new, the work of this chapter focuses on unconstrained digital environments, which provides new challenges and applications. Unlike combinatorial optimisation problems such as *shortest route* and *scheduling*, the work here looks more specifically at pattern formation and adaptation of embedded agents in open, environment landscapes. The description of

open environments refers here to an environment landscape in which the swarm agents are free to traverse in a similar way to ants moving around a patch of land in the real world. One significant difference to the nature counterpart is that this work considers a discretised computational world, in both time and space. Nevertheless, this work aims to draw a meaningful analogy between swarm behaviours observed in nature, and those that can be created in and utilise the digital world.

An investigation into the application of swarm intelligence to image feature extraction is carried out, and specifically, an ant-algorithm is developed for this purpose. An analysis is carried out to assess the performance of the algorithm for the specific task of image feature extraction, as well as to study the properties of self-organisation in a digitally embedded swarm.

3.2 Related Work

Digital image processing is well established amongst the scientific community and their exists many methods for performing various image processing tasks. Although certain aspects of machine learning and artificial intelligence have been utilised in image processing for some time, the use of ant algorithms to perform image processing tasks is a relatively new technique. Ant algorithms have been used for basic low level image segmentation via boundary detection methods [38, 39, 40] and via clustering methods [101, 102]. In [41] the authors present an ant-algorithm for visual saliency detection in images. The major differences between these approaches is the nature of the heuristic information used to guide the ant agents, however there is no clear advantage over one particular method. Although these works show promising results in terms of image segmentation, the respective analysis is mainly limited to that of a qualitative nature.

In [42, 43] the authors investigate how swarms of insect-like agents form patterns and build cognitive maps by creating a network of trails. The authors in [38] follow a similar approach to investigate the self-organising nature of a swarm of artificial ants in response to image edge features in a digital image habitat. These works introduce more quantitative analysis of the self-organising properties of the algorithms (but not the application of image processing).

The work of this chapter follows along similar lines, investigating self- organisation and pattern formation in swarms of ant-like agents, using digital imagery to form the agents environment. This chapter focuses on the use of a relatively simple ant-algorithm, as a key aim of this thesis is to show only simple decision making and communication is required to achieve complex pattern formation and adaptation in complex digital environments, in analogy to what we observe in natural swarms in real-world environments.

Previous work on applying ant-algorithms specifically to image processing tasks has only provided limited qualitative performance analysis, and has not explored the full potential of exploiting the self-organisational properties of the swarm intelligence approach. This chapter provides a quantitative analysis of an ant-algorithm approach to image feature extraction, and explores the properties of this approach and how the temporal self-organising and distributed nature of this method can be exploited.

Furthermore, the image processing application is used as a case-study to further explore the ways in which inspiration can be taken from nature to create interesting and useful swarming behaviours in the digital world, and how such techniques can be exploited to create complex pattern formation and adaption at the swarm level, based on local agent-agent and agent-environment interactions.

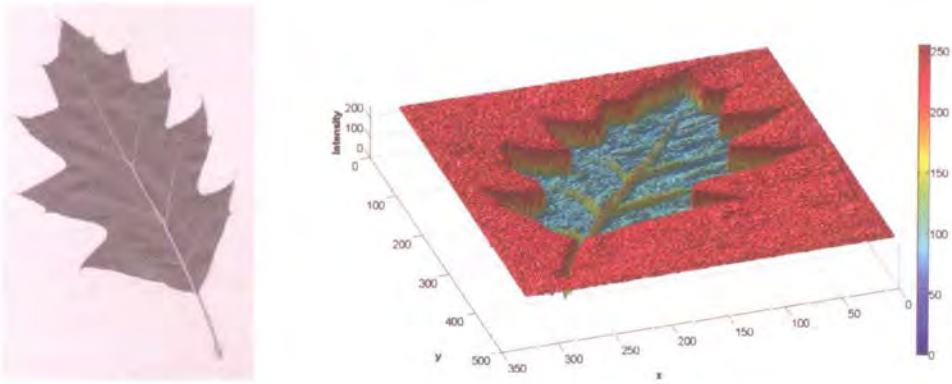
3.3 Digital Image Environments

Digital imagery is considered as an environment landscape for the agents to explore. The agents occupy pixels in the environment, and are able to move from pixel to pixel, 1 pixel at a time[1]. By plotting the intensity levels of a greyscale image (Figure 3.1(a)) we can visualise a complex 3D environment in which the agents are embedded (Figure 3.1(b)).

Using digital imagery as the environments affords the following advantages in the presented work: (a) Versatility and scope for a wide range of different environment structures; (b) ease of visualisation for qualitative analysis; (c) application

[1]Although this research considers the agents to move 1 pixel at a time, it should be noted that this does not have to be the case. An extension to this work could include allowing different agents to move by different amounts, to carry out multiple scale-space analysis. This is however beyond the scope of this thesis, and is discussed briefly as future work in chapter 7.

specific algorithms for image processing tasks.



(a) (b)

Figure 3.1: (a) An example greyscale image (of a digitally scanned image of a leaf); and (b) the corresponding 3D plot of the pixel intensity levels.

3.4 An Ant Algorithm for Image Feature Extraction

The basic framework of this algorithm is based around the workings of the original Ant System (AS) [22], with a modified, application specific pheromone update rule and heuristic information.

The algorithm employs artificial ants as simple computational agents. The algorithm is initialised with N ants occupying 'random' pixels within the image, where *pixels* in the *image* are equivalent to *states* in the search *environment*. The aim of the ants is to locate and map out the boundaries within the image. This is achieved by introducing heuristic information that weighs higher the probability of an ant moving from its current location to the allowed surrounding pixels that have the greatest boundary characteristics (greatest change in image gradient for example). Each ant deposits an amount of pheromone with each move to a new pixel, where the amount deposited may also be a function of, e.g. change in image gradient, and pheromone evaporation occurs at a fixed rate per iteration. The transition rule is then a function of the heuristic information and the pheromone map.

Formally, at each time step t, each of the N ants moves a distance of 1 pixel to one of the eight surrounding pixels (again it should be noted that working in the scale-space where agents are able to move varying distances is possible, but beyond the scope of research of this thesis). Each ants transition from state to state is guided by two main factors: *heuristic information*, and *artificial pheromone trails.*

The heuristic information is defined here as the *visibility*, η, which is the local image gradient around the ants current pixel location, measured with respect to the direction of travel from the ants previous location. In the context of this work the term *heuristic information* is used to describe any information the agents use from the local environment, to influence their movement. At each time step each ant calculates the visibility, η, associated with each possible move to the eight surrounding pixels, such that for an ant at pixel location (x, y),

$$\eta_{x-1,y-1} = |(I(x-1,y) - I(x,y-1))| \tag{3.1}$$

$$\eta_{x-1,y} = |(I(x-1,y-1) - I(x-1,y+1))| \tag{3.2}$$

$$\eta_{x-1,y+1} = |(I(x-1,y) - I(x,y+1))| \tag{3.3}$$

$$\eta_{x,y-1} = |(I(x+1,y-1) - I(x-1,y-1))| \tag{3.4}$$

$$\eta_{x,y+1} = |(I(x-1,y+1) - I(x+1,y+1))| \tag{3.5}$$

$$\eta_{x+1,y-1} = |(I(x,y-1) - I(x+1,y))| \tag{3.6}$$

$$\eta_{x+1,y} = |(I(x+1,y-1) - I(x+1,y+1))| \tag{3.7}$$

$$\eta_{x+1,y+1} = |(I(x+1,y) - I(x,y+1))| \tag{3.8}$$

where $I(x, y)$ gives the image intensity value at pixel location (x, y), and η_{ij} is equivalent to $\eta_{\pm x \pm y}$ depending on which direction the ant moves in.

This problem specific heuristic information aims to guide the ant agents to follow along edges and high contrast boundary regions within the image.

The pheromone concentration at any given pixel is given by τ_{ij}. Pheromone deposition by the ants happens at the end of each time step, along with a constant evaporation of the entire pheromone field. These processes are governed by the following pheromone update rule:

$$\tau_{ij}(t+1) = (1-\rho) \cdot \tau_{ij}(t) + \sum_{k=1}^{N} \Delta\tau_{ij}^{k} \tag{3.9}$$

where $\rho \in (0, 1]$ is the evaporation rate and $\Delta\tau_{ij}^{k}$ is the quantity of pheromone deposited at pixel location (i, j) by the kth ant and is given by:

$$\Delta\tau_{ij}^{k} = \begin{cases} \eta_{ij}/255 & \text{if ant } k \text{ in pixel } (i,j) \text{ and } \eta_{ij} > T \\ 0 & \text{otherwise} \end{cases} \tag{3.10}$$

where T is a user defined threshold value that can be set to only allow pheromone deposition by ants following edges or boundaries above a certain 'strength.' In addition there is also a *daemon action* implemented that terminates any ant agent with $\eta_{ij} < T$ for more than Z consecutive time steps. This terminated agent is immediately replaced by a new ant agent at a new 'random' location. This step is implemented to reduce the amount of ant agents 'lost' searching large background areas of the image and to speed up the rate of convergence of the agents onto the desired regions of the image search space.

Each ant then chooses its next pixel location by applying a probabilistic state transition rule, such that the probability of the kth ant moving to pixel location (i, j), at time step t, is given by

$$p_{ij}^{k}(t) = \begin{cases} \dfrac{[\tau_{ij}(t)]^{\alpha} \cdot [\eta_{ij}]^{\beta}}{\sum\limits_{k \in allowed_k} [\tau_{i_k j_k}(t)]^{\alpha} \cdot [\eta_{i_k j_k}]^{\beta}} & \text{if } I(i,j) \in allowed_k \\ 0 & \text{otherwise} \end{cases} \tag{3.11}$$

where $allowed_k$ is the eight pixels surrounding the kth ant, excluding any pixels in $tabu_k$, and α and β control the relative importance of the pheromone trail and visibility respectively. $tabu_k$ is a list containing the last n pixel locations visited by the kth ant, where $n = tabu_{max}$ gives the number of time steps into the past for which ants cannot re-visit previously visited pixel locations.

The input to the algorithm is a greyscale image, and the output is the emerged pheromone field, $\tau(t_{final})$. This is then converted to a binary image, τ_{BW}, to serve as the output feature image. Since the pheromone update rule (Equation 3.10) includes the threshold, all pixels containing pheromone are set as feature pixels when creating the binary feature map, such that,

$$\tau_{BW_{ij}} = \begin{cases} 1 & \text{if } \tau_{ij}(t_{final}) > 0 \\ 0 & \text{otherwise} \end{cases} \tag{3.12}$$

3.4.1 Examples

Figure 3.2 shows three example images of varying complexity. Also shown are the final positions of the ant-agents after the algorithm has run for $t = 200$ time-steps, along with the emerged pheromone map. From visual inspection we observe a higher concentration of ant-agents along the stronger edge features in the images, showing the effective self-organisation of the swarm in response to the structure of the environment. Correspondingly we see a build-up of pheromone concentration along the strong edge features, with the emerged pheromone field effectively mapping out the image edge features.

3.5 A Case Study in Leaf Pattern Extraction

Leaf shapes and venation patterns offer a wide ranging array of naturally occurring patterns for use in this study, and furthermore they offer an interesting and challenging image feature extraction problem.

Identification of leaf types from their venation pattern and outline is common practice for botanists, however traditional plant taxonomy often involves hand drawing such features and making subjective identification by eye [103]. With

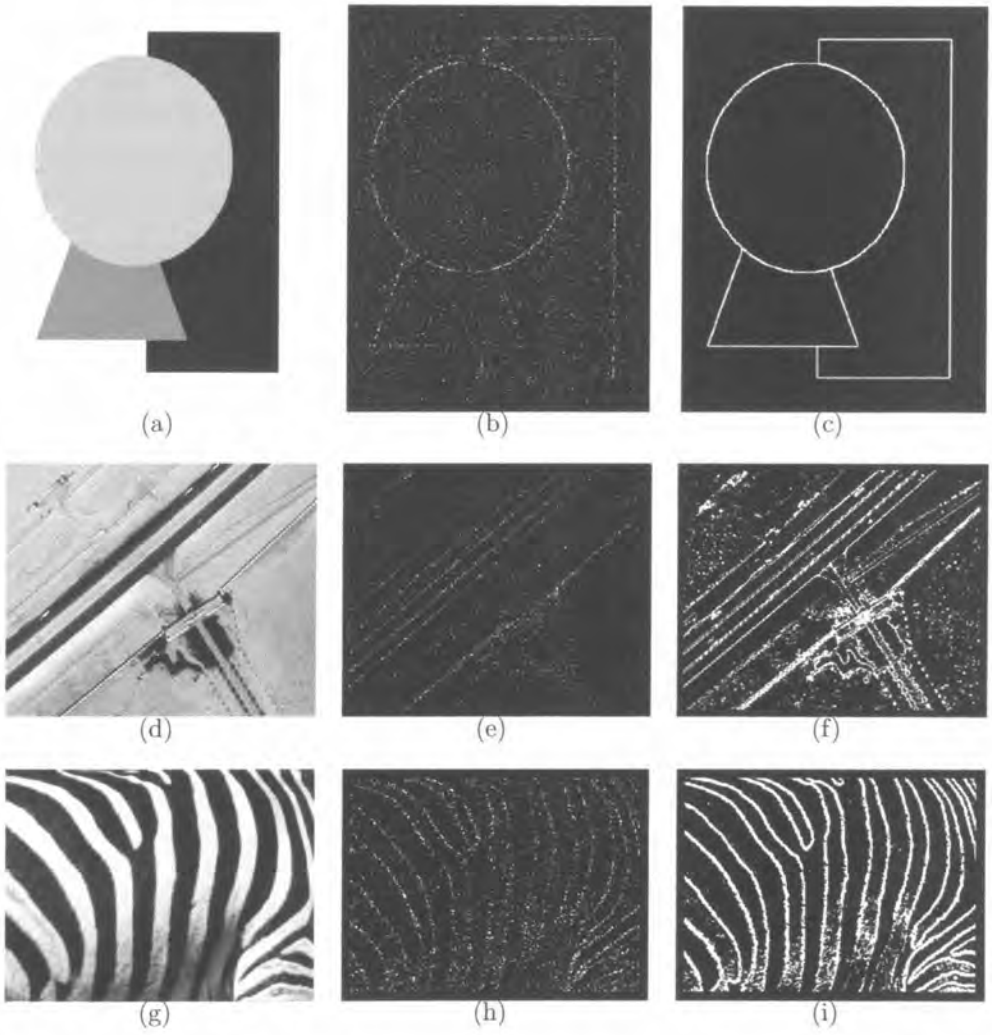


Figure 3.2: Example images including basic geometric shapes (a,bc), a satellite image (d,e,f) and a cropped image of a zebra (g,h,i). The left column shows the original images, the centre column shows the swarm agents positions (white pixels), and the right column shows the evolved pheromone maps.

vastly increasing data sets of digitised herbarium specimen, the scope for data mining such sources of data has wide reaching importance in applications ranging from vegetation inventory to medicinal plant use to evolutionary links with environmental change. Quantification of leaf features such as venation patterns and outline structures would add a whole new dimension to such data sets, extending the comparative search possibilities of plant characteristics. Due to the very large numbers of specimens involved in these data sets, a robust automated method is required that can efficiently quantify specimen characteristics from a wide range of images. Since specimens are in varying conditions when digitised, robustness of any method used is of key importance.

For this case study, a dataset of live leaf specimen from the *Quercus* family were collected from KEW Gardens, and captured into digital form using a high resolution flatbed scanner at RGB KEW[2].

3.5.1 Ground Truth Images

Ground truth images for the real leaf images were created by manually tracing the leaf outlines and primary venation patterns via a touch-screen tablet PC device. The on-screen pencil size was 3px by 3px so there is a ± 2 pixel error margin on the human perceived edges. However this may also be viewed as a 1 pixel buffer to the ambiguity of visually defining a 'true' edge pixel. In other words there is a 1 pixel thick buffer around the subjectively defined 'true' edge pixel at each instance. Since there is no definitive way to determine where a true edge actually lies, this approach was chosen as opposed to subjectively defining single pixel width edges.

Although it is known that the venation pattern can continue out to the leaf's outer edge, in many of the images in the collected dataset, due to limitations of image quality (resolution and noise artifacts), the venation is difficult to observe in these regions even with the human eye. Accordingly, in the ground truth process, for those regions where it was known that there would be venation present, but even to the human eye this was barely visible (a cut-off of $< 2px$ was used), those areas

[2]Royal Botanic Gardens, KEW, is a scientific institution which carries out research in systematics, biological interactions, economic botany, conservation and horticulture [104]. KEW Gardens is an expansive landscape of living plants next to the RBG KEW institution.

were not marked as venation. Through conversations with an expert botanist at RBG KEW [105] it was decided that although such missing venation could cause problems if the focus was on a rigorous pattern classification algorithm, extracting the *primary* venation pattern, and leaf outline was still sufficient for meaningful comparison and further analysis. Since the work in this thesis is not concerned with pattern classification, this was deemed acceptable for the purpose of this case-study in image feature extraction for swarm pattern formation analysis.

In order to assess the accuracy of this manual ground truth method, the process was carried out on the artificial leaf images, to which the true boundary locations are known. The number of true positive (TP), true negative (TN), false positive (FP) and false negative (FN) pixel classifications were calculated by performing a pixel wise comparison of the manual ground truth images with the true binary boundary images of the five artificial images.

Table 3.1 gives the *sensitivity*, *specificity* and *accuracy* calculated via the following equations, along with the *Mean Square Error* (MSE)

$$Sensitivity = TP/TP + FN \tag{3.13}$$

$$Specificity = TN/TN + FP \tag{3.14}$$

$$Accuracy = TP + TN/TP + FP + FN + TN \tag{3.15}$$

The sensitivity (or recall rate) is a measure of the proportion of actual positives identified as such. The specificity is a measure of the proportion of negatives which are correctly identified. The accuracy is a measure of the proportion of all true results (both TP and TN).

The results in Table 3.1 show us that the errors inherent in this method of ground truthing are minimal and are also at a level of consistency over a range of images so as not to have a significant effect on the results presented.

For each image, unless stated otherwise, the algorithm was run for $t = 500$ time steps, with the following parameters: $N = 5000$, $\alpha = 1.0$, $\beta = 7.0$, $\rho = 0.0001$, $T = 20$, $Z = 5$, and $tabu_{max} = 500$.

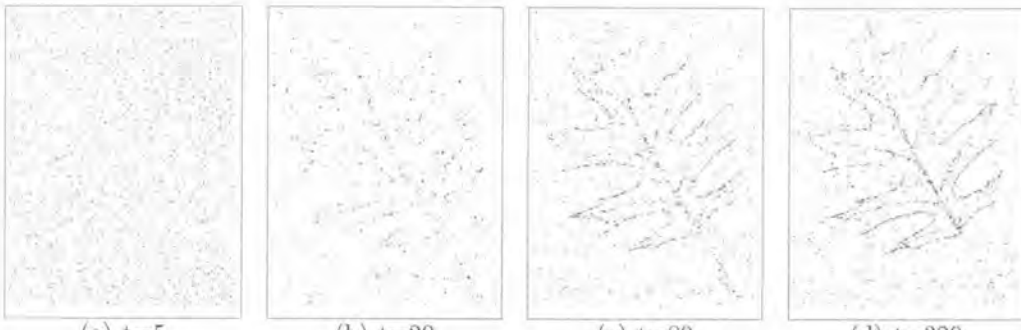
Table 3.1: Results of the errors incured when manually ground truthing the images by hand. Results here are computed by manually ground truthing a set of artificial leaf images, where the true segmentation is known. Statistics computed are: sensitivity, specificity, accuracy and mean square error (MSE).

Image	Sensitivity	Specificity	Accuracy	MSE
1	0.941	0.968	0.966	0.031
2	0.993	0.966	0.968	0.032
3	0.990	0.974	0.975	0.025
4	0.996	0.969	0.970	0.029
5	0.997	0.958	0.959	0.040
Averages	**0.983**	**0.967**	**0.968**	**0.031**

3.5.2 Qualitative Analysis

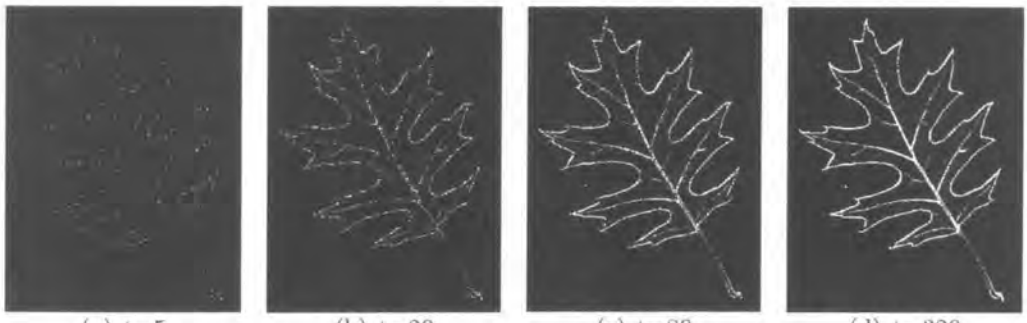
As the algorithm runs, the ant agents converge onto the boundary regions of the image, as illustrated in Figure 3.3, which shows the ant agents positions within the image space as black pixels on a white background. The corresponding emergent pheromone field can be seen in Figure 3.4, where the brighter pixels correspond to higher pheromone intensity at that pixel location within the image search space. Figure 3.5 shows example results of the final pheromone field next to the corresponding ground truth image and original image for both a real and artificial leaf image. As we can see, the resultant pheromone field maps out the boundaries within the image, showing clearly the leaf outline and primary venation pattern, where large amounts of pheromone have built up.

Closer visual inspection reveals some inevitable limitations. The primary venations in Figure 3.5(a) branching off the main vertical spine actually continue all the way to the edge of the leaf, however they become very thin as they get closer to the leaf edge and this makes them much less discernible even to the human eye. The effects of noise and non-uniform lighting at this level have a much greater affect, and as can be seen in Figure 3.5(c), these 'weak' low contrast edges are not represented in the algorithm output. Such missing information could cause problems for leaf type classification, as closely related venation patterns might not be picked up, and unrelated patterns might be wrongly grouped together [105].



(a) t=5 (b) t=20 (c) t=80 (d) t=320

Figure 3.3: Ant agent positions (black pixels) within a digital leaf image at different time-steps in the algorithm run.



(a) t=5 (b) t=20 (c) t=80 (d) t=320

Figure 3.4: Emerging pheromone map at different time-steps in the algorithm run. Brighter pixels equal higher pheromone concentration at that point.

The quality of the specimen can vary significantly between samples, with some specimen exhibiting much more defined venation patterns than others, and aside from the effects of noise, as well as lighting issues inherent in the scanning process, the problem is worsened by the fact that all the images have been downsized significantly from their original scanned size.

By lowering the threshold value T it is possible to allow pheromone to be deposited at these lower contrast edges, however this is at the cost of increasing the effects of noise in the final pheromone field. Also, and again due to the low contrast of these edges, the pheromone trails in these regions often become disjointed, as the variation in intensity levels due to noise here is comparable to the variation due to the edges of the venation pattern.

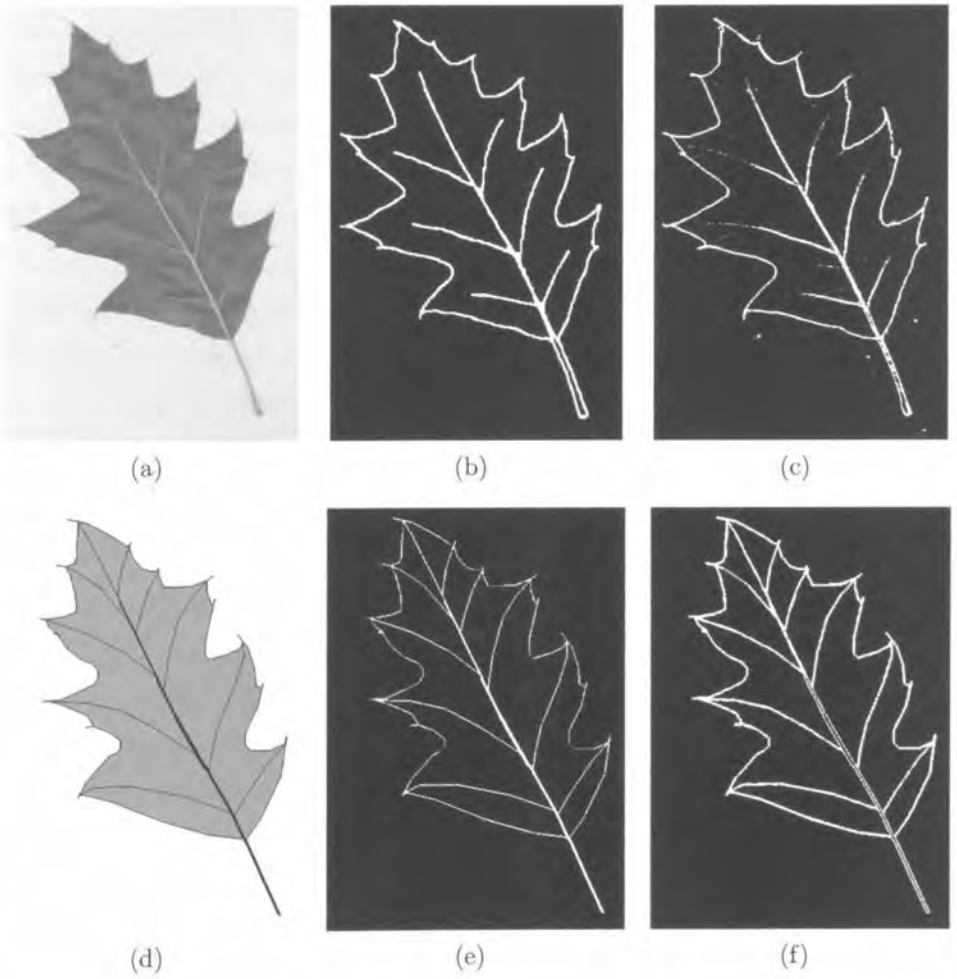


Figure 3.5: Example results: original images, (a,d), corresponding ground truth images, (b,e), and final pheromone fields, (c,f). Original image (a) is a real leaf image, and original image (d) is an artificial leaf image.

The images shown in Figures 3.3, 3.4 and 3.5 are typical of the results seen in all the leaf images used in this study.

3.5.3 Quantitative Analysis

The quantitative performance analysis of the algorithm is carried out by computing the sensitivity, specificity and accuracy of the algorithm output when compared to the ground truth images, in a similar fashion as described in section 3.5.1. When analysing the performance of an image feature extraction algorithm, traditionally the results would focus on the end output of the algorithm (for example a binary pixel classification). Although the performance of the ant-algorithm approach for the specific purpose of image feature extraction is indeed of interest to this work, of particular interest is the examination of the self-organisational properties of this approach. One of the key properties to characterise when a given pheromone is self-organised is the creation of spatiotemporal structures in an initially homogeneous medium [2]. As such, it would be more instructive to look at the statistical results over time, throughout the self-organising process.

The results in Figure 3.6 are averages from the results of twenty real leaf images (see Appendix A for thumbnails of the leaf images). The results show how the accuracy, sensitivity and specificity vary as the algorithm runs. As can be seen, the overall accuracy increases over time, as the ant agents converge on the edges and the pheromone concentration here increases such that more and more pixels are detected as edge pixels. This results in an increase in the number of TP classifications and a decrease in FN, which is also reflected in the increase in sensitivity (Figure 3.6(b)). The specificity (Figure 3.6(c)) decreases over time as the number of TN counts decreases and the FP increases, as pheromone concentration builds up across the entire pheromone field, including areas outside of the 'true' edge regions. The specificity does not however decrease by any large amount and remains at a high value due to the fact that the TN count is always much greater than the other counts because the majority of the pixels within all of the images are in fact not edge or boundary pixels (i.e. they are background pixels).

Figure 3.7 compares the results obtained from a real leaf image with high contrast venation pattern, to those obtained from an artificial leaf image. The accuracy,

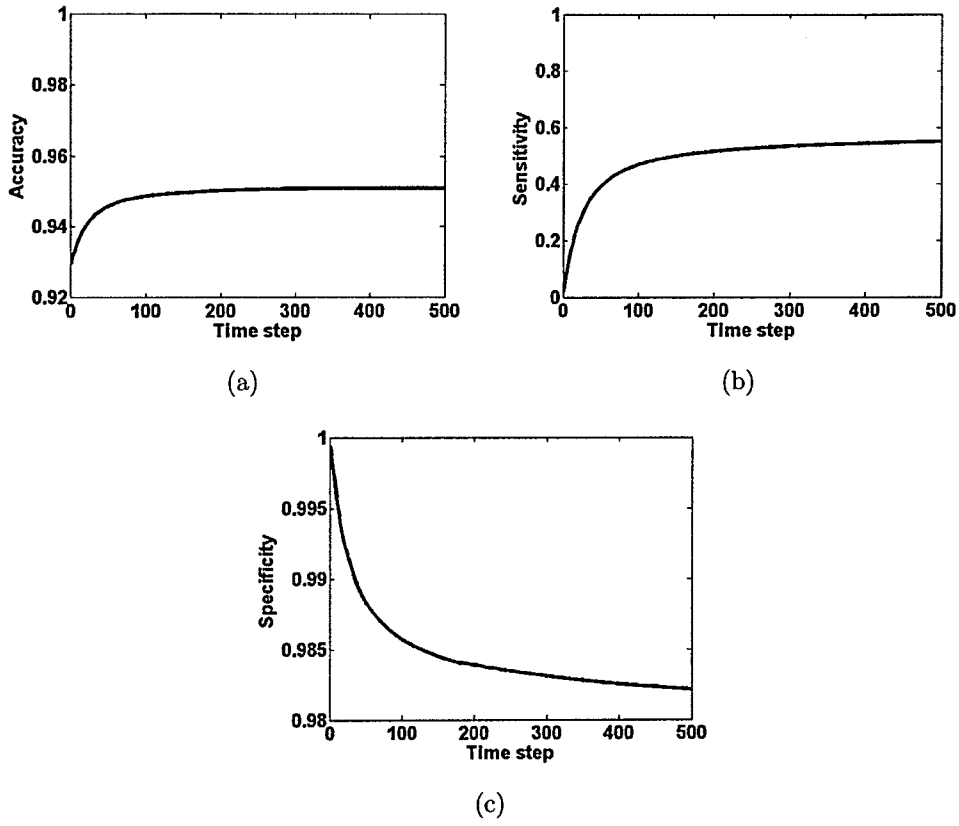


Figure 3.6: Plots of average results from 20 real leaf images. Plots show average accuracy, (a), sensitivity, (b), and specificity, (c), over 500 time-steps, as measured from comparing the algorithm output with the ground truth images. Note: Scales are not the same for each image

sensitivity and specificity are all higher for the artificial image (this is true in general for all the artificial leaf images). This is perhaps not surprising since the artificial images are noise free, such that the backgrounds, both inside and outside of the leaf, are of uniform grey level intensity. This means that the only change in image gradient occurs at the leaf outline edge and the venation pattern edges, so the scope for error here is minimal. There will however still be a small error count, and this is due to the fact that pixels directly next to 'true' edge pixels may also receive a large amount of pheromone, hence this is why the accuracy is not at 100 percent for the artificial image in Figure 3.7(a). From an image processing perspective this might initially be thought of as a disadvantage due to the reduction in accuracy. However, there is an alternative viewpoint in that the

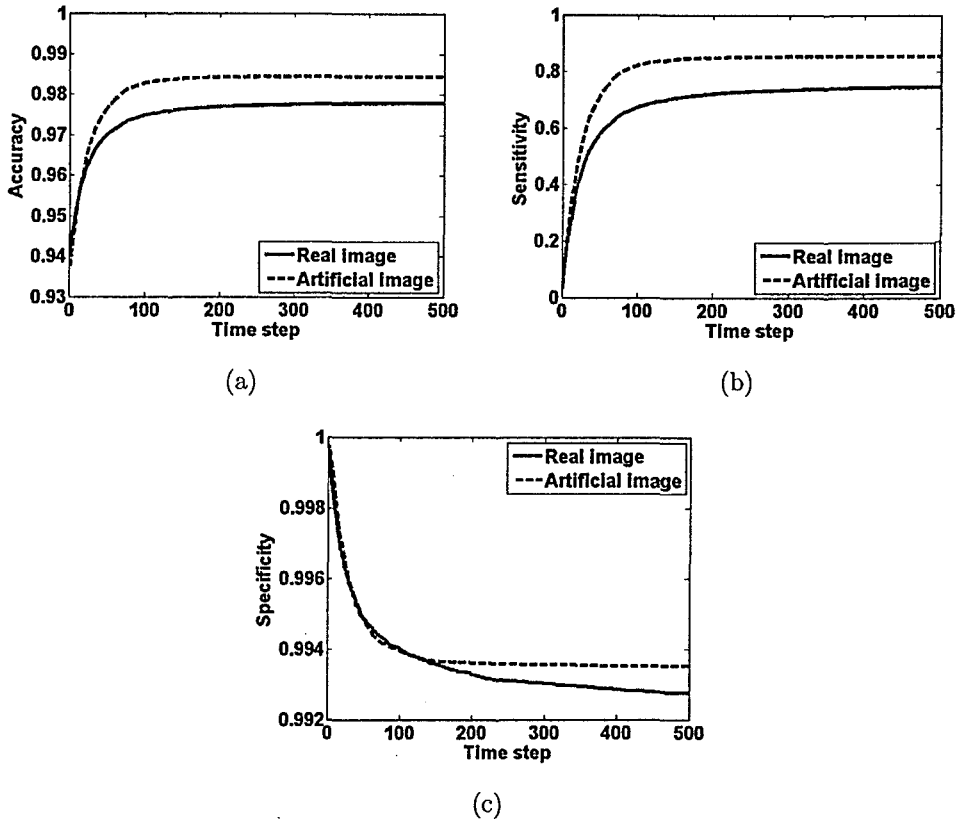


Figure 3.7: Plots comparing results from an artificial leaf image and from a real leaf image with high contrast venation pattern. Plots show accuracy, (a), sensitivity, (b), and specificity, (c), over 500 time-steps, as measured from comparing the algorithm output with the ground truth images. Note: Scales are not the same for each plot.

'blurring' effect the cumulative pheromone effect can have over the image edge features, can in some way provide a buffer for the subjective definition of where in fact the true edge feature lies.

The above analysis was carried out using an empirically determined threshold value. Figure 3.8 shows a plot of the Receiver Operator Characteristic (ROC) curve of the ant-algorithm applied to a typical leaf image, with varying threshold T values. The results show a ROC curve approaching the desired $(0, 1)$ point of perfect classification. Further analysis of the effects of varying parameters follow in the forthcoming sections.

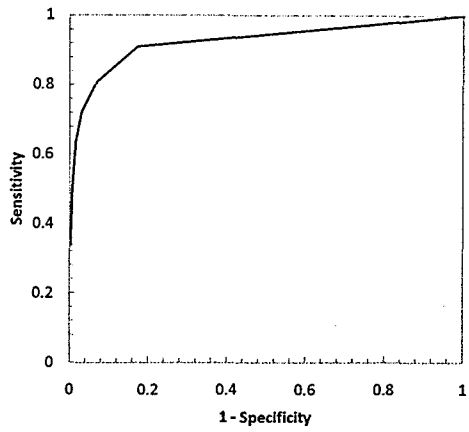


Figure 3.8: Receiver Operator Characteristic (ROC) plots showing the ROC curve for the ant-algorithm with varying threshold T.

3.5.4 Comparison to Existing Methods

Although the focus of this thesis is on the mechanisms of swarm self-organisation and adaptation, since the application of image feature extraction is being used as a case study, and thus the performance of the algorithm for this purpose is a measure of the effectiveness of the self-organisation process, it would be interesting and instructive to assess how well the ant-algorithm approach works in comparison to traditional image processing methods for feature extraction.

Here, the proposed algorithm is compared with existing methods of image edge feature extraction. The Sobel and Canny edge detector algorithms are applied separately to the same test data-set of real leaf images and the average accuracy is compared to that obtained with the ant-algorithm approach. This comparison is included to show how the ant-algorithm performs in comparison to existing methods for the specific problem of image edge feature extraction. Image processing application aside, since the goal of the ants is to self organise in response to the image edge features (thus providing the pheromone map for feature extraction), this comparison does give an indication of the extent to which the swarm has managed to self organise, by comparing to two well established methods for dealing with image edge feature extraction.

Table 3.2: A comparison of the average accuracy obtained using the proposed ant algorithm approach, the Sobel edge detector and the Canny edge detector. Experiments are carried out on the leaf data-set, as described in section 3.5.3.

Algorithm	Average accuracy
Ant	0.945
Sobel	0.940
Canny	0.897

From Table 3.2 we see a comparable performance between the three algorithms when comparing the average accuracy (taken as an average over the leaf image dataset described previously).

Figure 3.9 shows a plot of *sensitivity* versus $1 - specificity$, showing results in ROC space of the ant-algorithm, Canny and Sobel edge detectors, applied to the dataset of 20 leaf images from RBG KEW. We see the ant-algorithm results clustering closer to the $(0, 1)$ point of perfect classification, with the Canny and Sobel performing comparatively in this test.

Looking at Figure 3.10 we can make a qualitative comparison between the three methods. From visual inspection we notice how the Canny method has picked up a greater amount of detail in the image gradients, which can account for the slightly lower average accuracy (due to over segmentation with regards to the ground truth data). One positive feature of the Canny method is the low level of discontinuities along the major contours. To an extent the same can be said for the Sobel method, when comparing to the ant algorithm method, which suffers more from discontinuities along the contours. The Canny and Sobel methods (being edge detectors) both show edge responses on each side of the larger venations. The ant algorithm on the other hand shows larger areas of detection, covering whole venations, as opposed to just the edges. This is due to the reinforcement nature of the pheromone map, whereby a build-up of pheromone occurs on and around these strong image gradients, representing the venation structure and leaf outline. This offers greater flexibility for feature extraction.

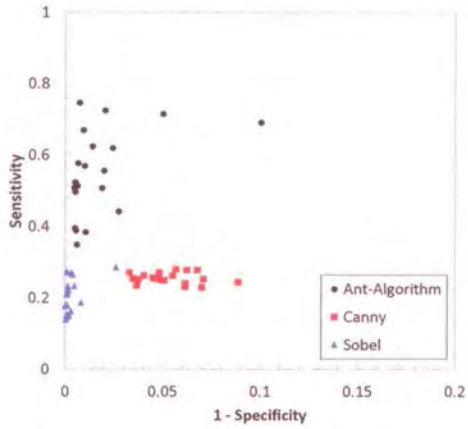


Figure 3.9: Receiver Operator Characteristic (ROC) plot showing a comparison in ROC space of the results from the ant-algorithm, Canny, and Sobel edge detectors applied to the dataset of 20 leaf images from RBG KEW. Note the different scales of the x and y axis.

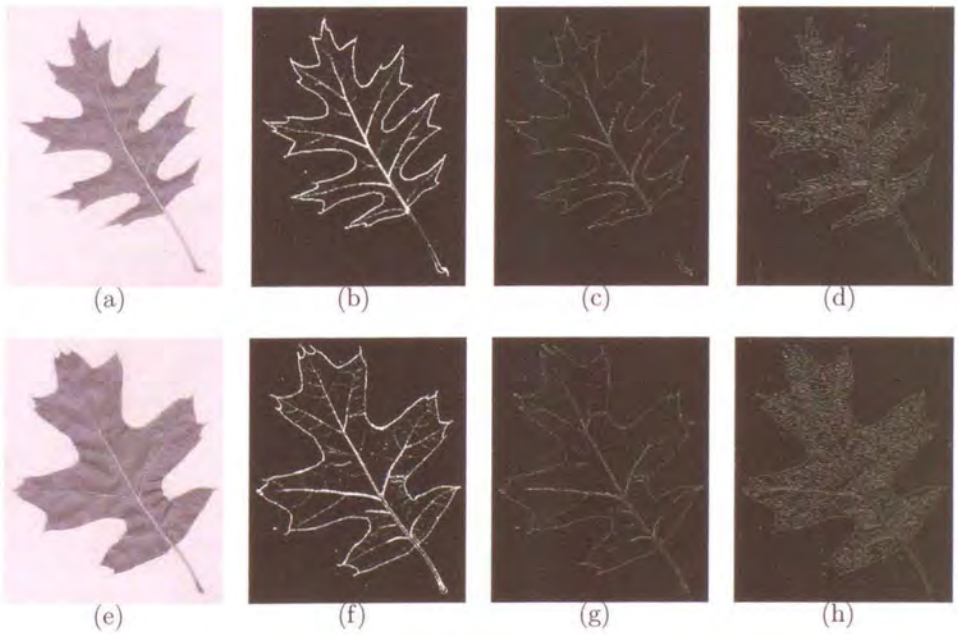


Figure 3.10: Example results comparing methods: original images, (a,e), ant algorithm output, (b,f), Sobel output, (c,g), Canny output (d,h).

3.6 Parameter Exploration

There are a number of key parameters that alter the way in which the swarm behaves. In the previous case-study for leaf pattern extraction a process of trial and error was employed to find the most appropriate parameter configuration. This section will explore the key parameters and properties of the algorithm. Although significant efforts have been made in the past to explore and understand the key parameters of ant-algorithms (and indeed other swarm intelligence algorithms), this has not been done for the application of image processing, and moreover, in unconstrained open environments in general.

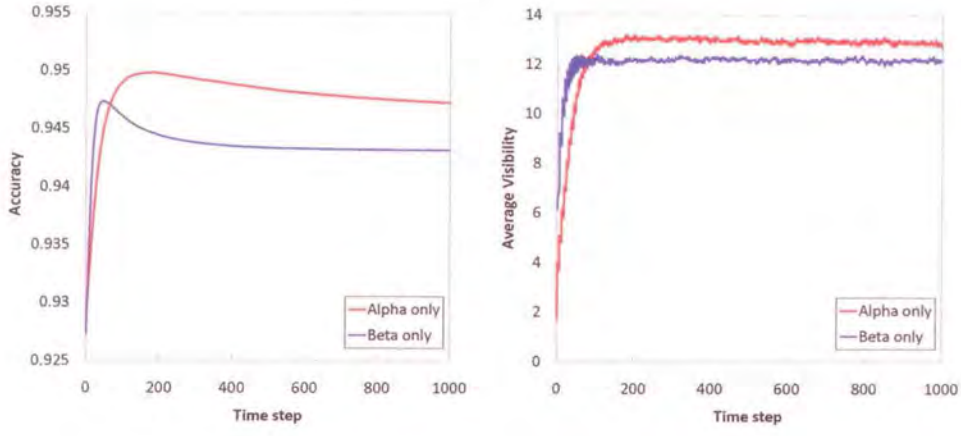
3.6.1 Pheromone Versus Heuristic Information

As has been discussed previously, one of the key components to successful self-organisation, particularly in the case of ant algorithms, is the use of stigmergy. For ant algorithms this is commonly achieved by using artificial pheromones to effectively allow individual ants to indirectly share information.

In the case of the image feature extraction algorithm there is an interesting two-fold relevance to the use of artificial pheromones. Not only do the pheromones allow the ant agents to communicate indirectly, but the pheromone trails are also the output of the algorithm; with the final evolved pheromone map depicting the features of the image to be extracted.

With regards to this study, of particular importance are the control parameters α and β. Choosing appropriate values here is necessary to ensure a productive trade-off between exploration and exploitation in the search environment. If an effective trade-off is met, then the positive and negative feedback of the system ensures that the stronger edges in the image are well represented in the output, while the weaker edges, such as those due to noise in the image, are evaporated away.

Figure 3.11 compares the statistical accuracy (as before, measured by comparison to ground truth images) and average visibility (the heuristic visibility averaged over the entire swarm) results from running the algorithm twice on a sample of 20 real leaf images, with parameters as before. One run is set with only pheromone information guiding the agents (with $\alpha = 1$), and the other with only heuristic



(a) (b)

Figure 3.11: Plots of average results from 20 real leaf images, running the algorithm separately with only pheromone information (α control parameter), and then with only heuristic information (β control parameter). (a) A plot showing the accuracy versus time-steps, and (b), a plot showing the average visibility versus time-steps.

information (with $\beta = 1$). The differences in the results are comparable to those between a real and an artificial image, as in Figure 3.7.

It is interesting to note the difference in accuracy (Figure 3.11(a)) at the early time-steps. The more rapid increase in accuracy for the 'beta only' run can be attributed to the fact that the agents in this case will be forced always to explore the search environment. In the early stages of the run this will result in new desired regions being found quickly, however without the ability to sense the artificial pheromone, there is no form of inter-agent communication and stigmergic behaviour will not occur. Conversely, when agents are only influenced by the pheromone trails, then the rate of convergence will be slower as agents will become trapped in local minima early in the algorithm run, especially when noise is present in the image and the pheromone field may contain features not representing 'true' image edges.

In Figure 3.11(b) we see a similar trend, with the α-only results showing slower initial convergence but finishing with a higher level of average visibility than the β-only results.

Figure 3.12 shows results obtained with varying α, β parameter settings, using a

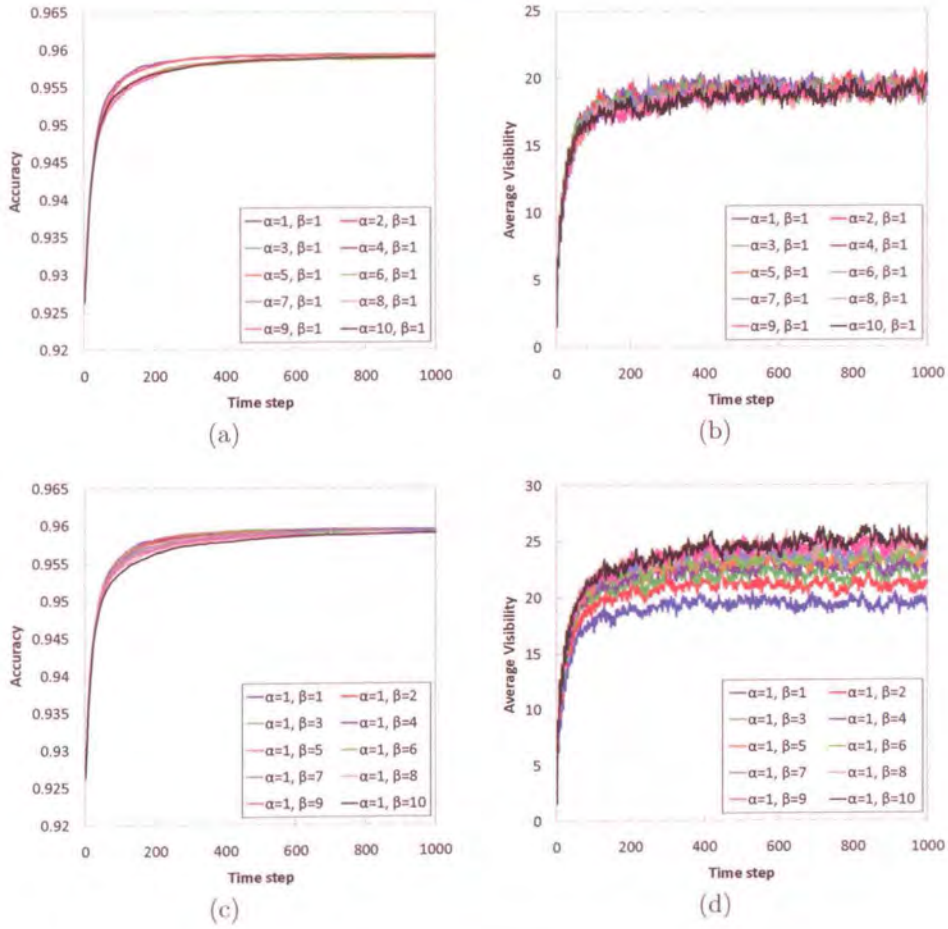


Figure 3.12: Results showing the variation in accuracy (a,c) and average visibility (b,d) when using different α and β parameter settings.

typical real leaf image. The results show that with a constant $\beta = 1$, increasing the α parameter has little effect on the accuracy (Figure 3.12(a)) and average visibility (Figure 3.12(b)). For a constant $\alpha = 1$, we see a significant increase in average visibility with increasing β (Figure 3.12(d)). Interestingly, although over a less significant scale, we observe a decrease in accuracy with increasing β (Figure 3.12(c)).

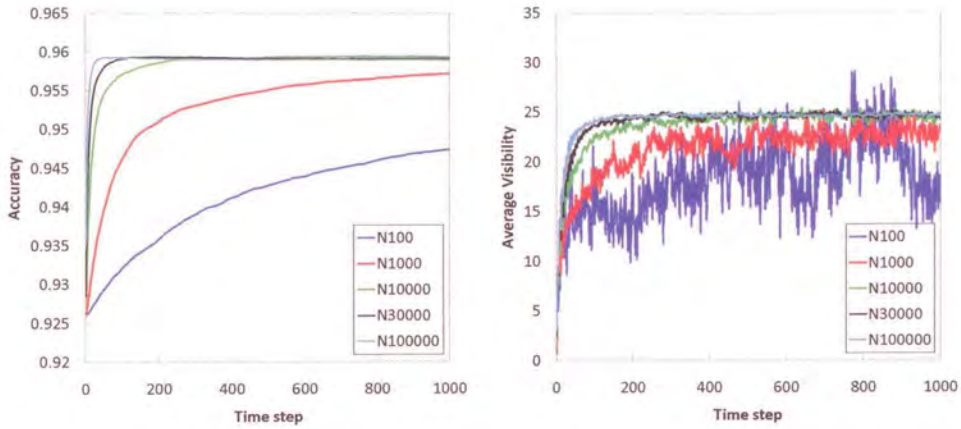
3.6.2 Varying the Swarm Size

The size of the swarm, that is, the number of ant agents, N, inherently effects the output of the algorithm as the pheromone map is produced by the collective swarm.

As Figure 3.13(a) shows, with increasing N we see an decrease in convergence time for the measured accuracy. This is perhaps not surprising if we consider that the more ant agents we have embedded in the image, the more pixels will be occupied by agents, and consequently, more pixels will simultaneously have pheromone deposited in accordance with the pheromone update rule. As Figure 3.14 shows, eventually the accuracy for a swarm of size $N = 100$ reaches a comparable level to that for a swarm of size $N = 30000$. It simply takes significantly longer for the pheromone map to build up, assuming a relatively low evaporation rate ($\rho = 0.0001$).

Figure 3.13(b) shows comparatively little difference in convergence time of average visibility for varying the swarm size between $N = 100$ and $N = 100000$. Since the accuracy is measured against ground truth data, this performance metric is relevant to the specific task of pattern extraction, and is affected by the threshold values. The average visibility on the other hand, gives us an idea of the extent to which the swarm is converging on the desired pattern, measured with respect to swarms raw perception of the environment in terms of the heuristic information. It follows then that the two metrics are not intrinsically linked, rather they provide a complimentary analysis of the swarms behaviour.

We also observe a decrease in variation in average visibility with increasing N, but this is as expected given this measure is averaged over the entire swarm. Figure 3.13 shows that for both the accuracy and average visibility, there is only a small difference in convergence time between swarm sizes of $N = 30000$ and



(a) (b)

Figure 3.13: Results for varying the size of the swarm, applied to a typical real leaf image. Plots show the accuracy, (a), and average visibility, (b), versus time-steps.

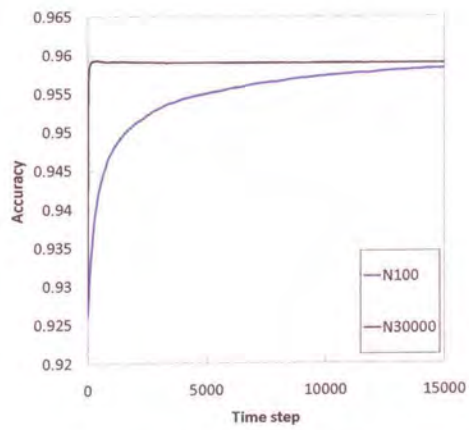


Figure 3.14: Average accuracy comparing $N = 100$ and $N = 30000$ over a longer period of time.

$N = 100000$ (and hence not showing results for larger N). Increasing N further has a negligible affect on the results, as agents are free to move over one another, and the pheromone evaporation rule ensures that the pheromone field does not become saturated.

3.6.3 Varying the Threshold

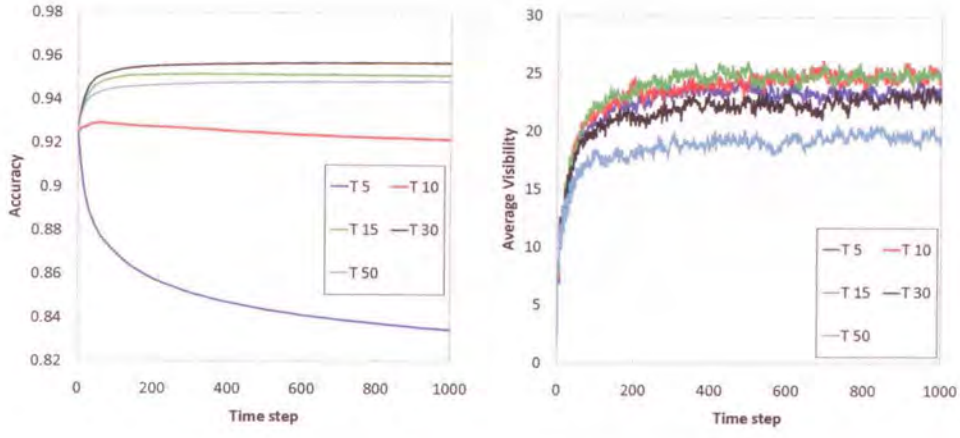
The threshold value T serves two purposes. (i) To only allow ants to deposit pheromone if they are following an edge of gradient greater than T. This affects the self-organisation process by eliminating pheromone information for particularly weak edges (below T), and moreover, effects the algorithm output (i.e. the pheromone map) directly. (ii) To define the daemon action condition, replacing ants 'lost' searching areas of the image with visibility below T. This acts as a catalyst to the self-organisation process.

It follows that varying T will have a significant affect on the performance of the algorithm, both in terms of the output specific to the problem of feature extraction, and the general self-organisation process.

As expected, in Figure 3.15(a) we observe a significant variation in Accuracy with varying T. Initially, as T increases, so does the resultant accuracy. This is then followed by a decrease in accuracy with further increase in T from $T = 30$ to $T = 50$. Again this is not unexpected given the mechanics of the algorithm, and the specific problem of edge feature extraction, giving rise to an optimum value of T, which will indeed be different depending on the characteristics of the specific image. It is interesting to note that while there is a significant variation in the resultant accuracy, the variation in T does not significantly effect the convergence rate.

In Figure 3.15(b) we see an inverse relationship between T and average visibility, with increasing T resulting in lower average visibility for large T. This can be accounted for by considering the fact that the visibility is not directly affected by the threshold T. The self-organisation process is however affected by the daemon action, as well as the pheromone map, both of which are directly related to T. Increasing T to a large value that leaves only a small proportion of the image space above the threshold will have a detrimental effect on the self-organisation process, as the daemon action will be continuously replacing the majority of ant agents to new 'random' locations. With the majority of ant agents not converging on areas of high visibility, the average visibility will be lower.

Due to the threshold imposed on pheromone deposition in Equation 3.10, and the conversion to a binary feature map in Equation 3.12, a low threshold produces a low accuracy due to the small amounts of pheromone being deposited in the vast

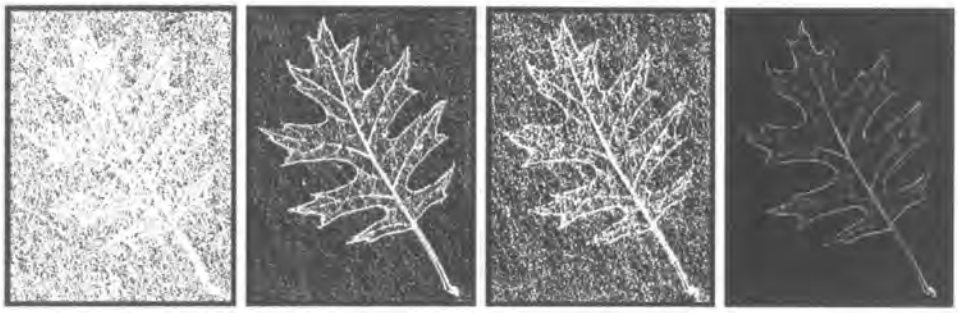


(a) (b)

Figure 3.15: Results for varying the threshold value T, applied to a typical real leaf image. Plots show the accuracy, (a), and average visibility, (b), versus time-steps.

regions of the image where there are only very weak edges present. These very weak pheromone trails will still be treated as image features, thus producing a binary feature map such as the example given in Figure 3.16(a). A low or non-existent threshold will also mean that the daemon action will not function, thus leaving more ant agents searching the featureless regions, which increases the accuracy degradation further still. If we study τ, the pheromone map prior to conversion to binary (Figure 3.16(b)), we can see that the pheromone trails along the prominent edge features are more pronounced. However in order to create a useful feature map, we would have to impose a specific threshold when converting to binary. This illustrates the requirement for the threshold T, in either way of implementation. Figure 3.16(d) shows the final pheromone map, τ, again with $T = 1$ but with a higher evaporation rate, with $\rho = 0.02$. From visual inspection we see a much cleaner extraction of the edge features, with little noise present. If we inspect the corresponding τ_{BW} (Figure 3.16(c)) we see there is still in fact a large amount of pheromone present in the background areas of the image, resulting again in a highly over segmented binary image.

The choice of threshold parameters is often critical in algorithm performance, and this case is no exception. Although it has been shown that the swarm can effectively evolve a pheromone map representing the pattern we wish to extract,



(a) (b) (c) (d)

Figure 3.16: A comparison showing the evolved pheromone map with $T = 1$, (a) the standard binary feature map τ_{BW}, (b) the final pheromone map τ, prior to binary conversion, (c) τ_{BW} with a greater evaporation rate of $\rho = 0.02$, and (d) τ with $\rho = 0.02$.

a threshold is still required in order to remove the unwanted background information. In the next chapter, attention will be given to the concept of adaptive thresholding, where in the case of the developed ant algorithm, each individual agent will be able to adjust their own local thresholds as they move around the image.

3.6.4 Varying $tabu_{max}$

The $tabu_{max}$ parameter gives the number of previously visited pixels into the past which ant agents are not allowed to re-visit. The function of this mechanism is to prevent the ant agents getting trapped in local minima (which is a widely encountered problem in heuristic based search algorithms).

Figure 3.17 shows the agent positions and evolved pheromone maps after 1000 time-steps, comparing results with $tabu_{max} = 1$ and $tabu_{max} = 500$. Figure 3.17(a),3.17(b) gives an example of the issue of local minima. With $tabu_{max} = 1$, once the ant agents have found a strong edge, they become trapped, moving backwards and forwards over the same few pixels, and thus do not trace along the edge features, as can be seen by the resultant pheromone map, which contains many breaks in the pattern edges. In comparison to Figure 3.17(c),3.17(d), where $tabu_{max} = 500$, we can see the importance of this functionality.

Figure 3.18 shows the corresponding accuracy and average visibility, covering a

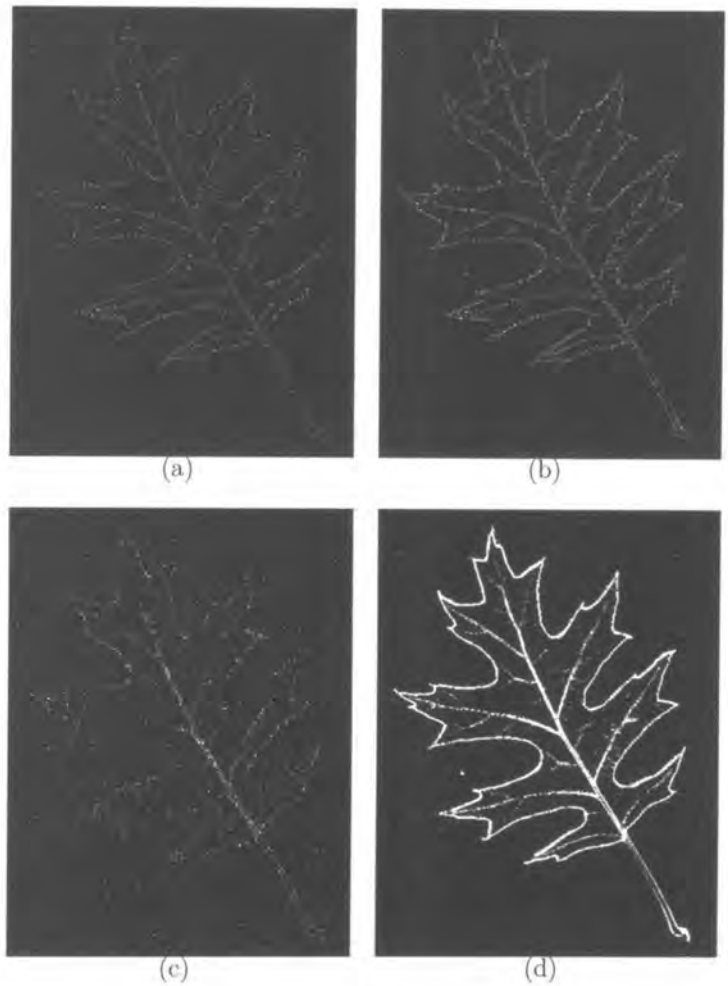
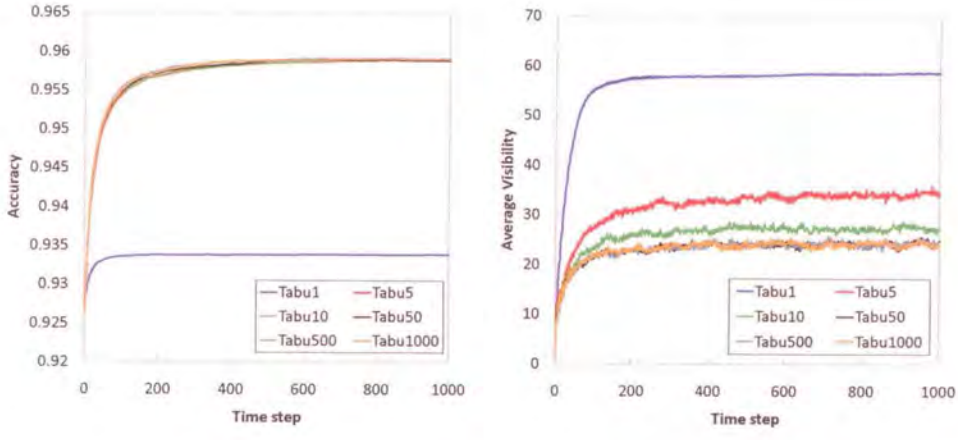


Figure 3.17: Results showing final agent positions and pheromone maps with $tabu_{max} = 1$ (a,b) and $tabu_{max} = 500$ (c,d), applied to a typical real leaf image.

large range of $tabu_{max}$. We see a substantially lower accuracy for $tabu_{max} = 1$, which corresponds to the local minima issue we see in Figure 3.17, preventing the pheromone map from fully evolving. Conversely, we see a larger average visibility for smaller $tabu_{max}$. This is because as the daemon action allows the swarm to converge onto the pattern (through repetitions of the daemon action), once the agents are present on the pattern, they do not move from the pattern, thus the vast majority of agents have a high visibility. The problem is however, that the agents do not move much at all, thus the swarm stagnates, and the full pattern is not achieved in the pheromone map. If fewer agents were used (smaller N) in



(a) (b)

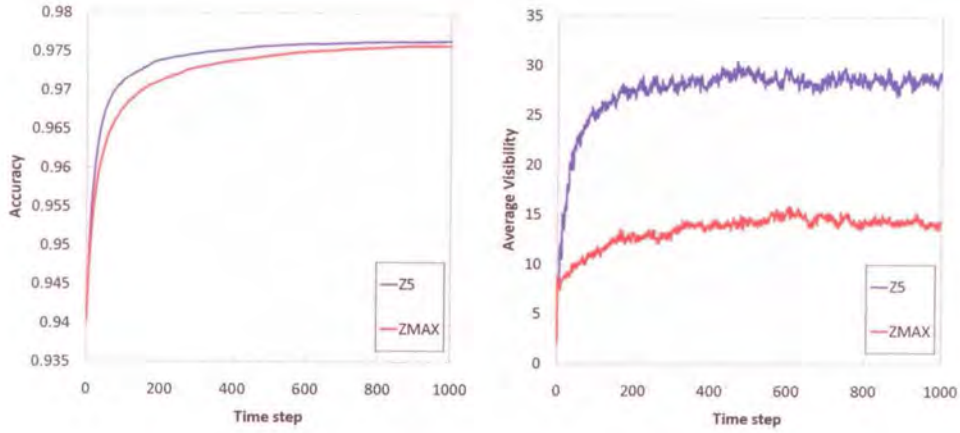
Figure 3.18: Results for varying $tabu_{max}$, applied to a typical real leaf image. Plots show the accuracy, (a), and average visibility, (b), versus time-steps.

this case of a small $tabu_{max}$, there would be even less pheromone map.

The use of a large $tabu_{max}$ thus promotes exploration within the swarm, and although this is at the expense of allowing agents to 'wander' away from the desired image features, ultimately the exploration yields greater coverage of the pattern, thus allowing the pheromone map the evolve with less breaks, and allowing the swarm to traverse along the pattern without getting trapped in local minima. As Figure 3.18 shows, there is little difference in the resultant accuracy curves for $tabu_{max} = 10 - 1000$, and decreasing difference in average visibility with increasing $tabu_{max}$. This shows the importance of utilising this technique, but that there is little difference in performance with wide varying parameter changes with increasing $tabu_{max}$.

3.6.5 The Daemon Action

The daemon action is influenced by the two parameters, T and Z. Its purpose is to act as a catalyst to speed up the self-organisation process by replacing ant agents that are 'lost' searching areas of the image not containing features of interest. Figure 3.19 compares results obtained from a typical leaf image, running the algorithm with $Z = 5$, and then with $Z = Z_{max}$, i.e. the maximum possible Z resulting in no daemon actions occurring, where Z is equal to the maximum



(a) (b)

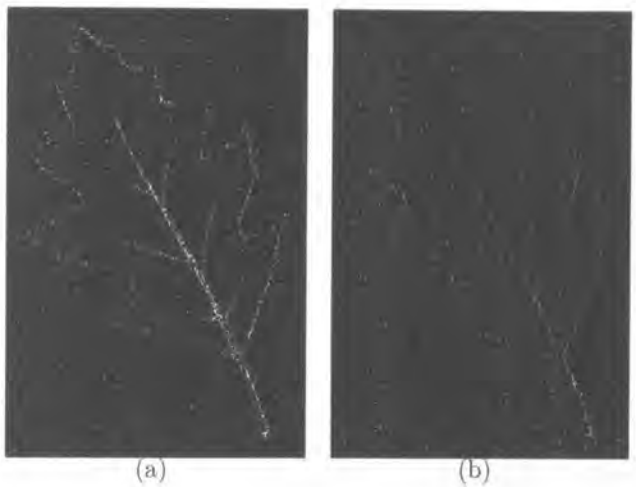
Figure 3.19: Results showing with ($Z = 5$) and without the daemon action ($Z = Z_{max}$), applied to a typical real leaf image. Plots show the accuracy, (a), and average visibility, (b), versus time-steps.

number of timesteps. We see that the rate of increase, and the converged value, of the average visibility is significantly greater when using the daemon action. This is also reflected in the resultant accuracy, although less pronounced. Concurrently in Figure 3.20 from visual inspection we observe that the final positions of the swarm when using the daemon action show better convergence onto the leaf outline and venation pattern, when compared to the non-daemon action counterpart. This is indeed in agreement with Figure 3.19, and shows the positive 'catalyst' effect of this daemon action on the self-organisation process.

3.7 Multiple Swarms for Multiple Features

If we again turn our attention to nature, there are numerous examples of ant colonies being made up of different types of specialist ants to carry out different specific tasks [106, 107, 108]. In this same way, the proposed algorithm can be modified to employ multiple swarms to simultaneously search the image environment, with each swarm programmed differently (for example with different parameter settings, or alternative heuristic information).

An issue identified in the earlier case-study on venation feature extraction was the



(a) (b)

Figure 3.20: Results showing final agent positions with and without the the use of the daemon action, applied to a typical real leaf image. (a), $Z = 5$, (b), $Z = max$.

difficulty in discerning the smaller, tertiary venations in the extraction process. This problem was largely due to limiting factors in image quality and resolution. When setting the threshold T low enough to extract tertiary venation, this was detrimental to the extraction of the primary venation pattern due to the resulting evolved pheromone map including many broken segments where the tertiary venation was not well defined, and where background noise in the image had been extracted as features.

By employing two swarms, each using a different threshold, it is possible to allow them to simultaneously self-organise and evolve two separate pheromone maps; one for the primary venation pattern and leaf outline, and one for tertiary patterns. Figure 3.21 gives two examples using leaves from the RBG KEW dataset. For these examples we have two swarms, with $N = 3000$ for each. Swarm A, which has a pheromone map depicted in red (Figure 3.21(c),3.21(g)), has a threshold of $T = 25$. Swarm B has a dual threshold of $T_{upper} = 25, T_{lower} = 15$, with a pheromone map depicted in green (Figure 3.21(d),3.21(h)). The upper and lower thresholds limit Swarm B to edge features within a given range. The combined pheromone maps (Figure 3.21(b),3.21(f)) show the leaf outline, primary and tertiary venation pattern, as extracted by the two swarms.

From a swarm intelligence perspective this is an interesting visual example of si-

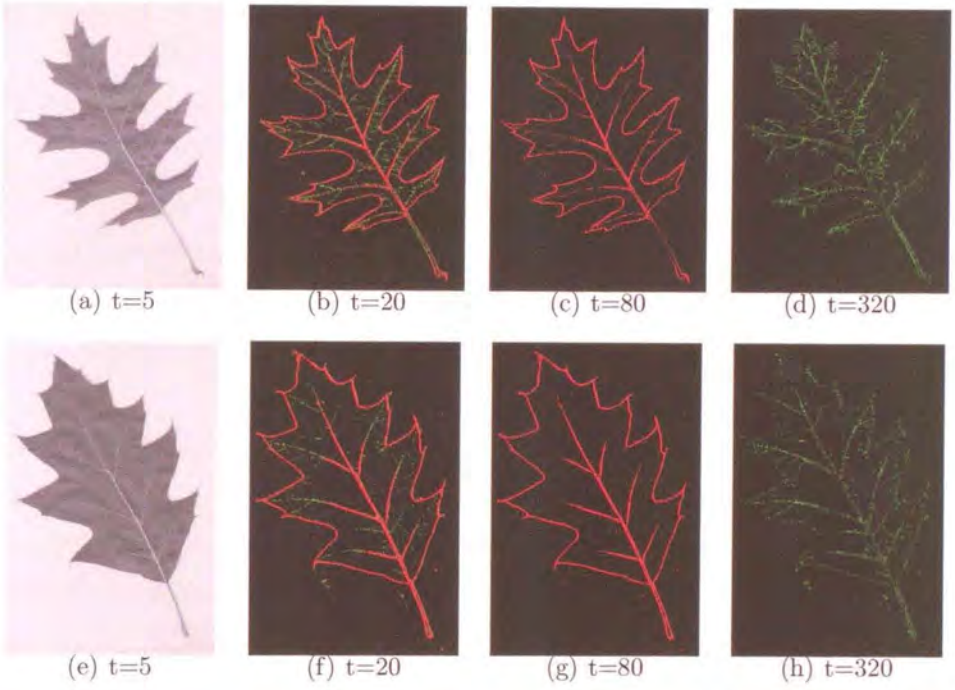


Figure 3.21: Examples of using dual swarms with alternative thresholds to simultaneously evolve pheromone maps of different features. (a,e) Original images; (b,f) combined pheromone maps; (c,g) pheromone map from swarm with $T = 20$; (d,f) pheromone map from swarm with dual threshold $T_{upper} = 25, T_{lower} = 15$.

multaneous self-organisation of two specialist swarms acting independently from one another. In employing this nature inspired technique it has been demonstrated visually how a simple extension to the implementation of the ant-algorithm to allow multiple swarms to run independently has facilitated dual feature extraction from the digital image environments, and furthermore, has enabled an alternative solution to a typical image processing problem.

3.7.1 Alternative Image Features

Thus far only image edge gradient has been considered as the heuristic information to guide the ant agents. It would be interesting to study the effects on the self-organisation and pattern formation using different heuristic information, to guide the agents towards alternative image features.

RGB Feature Extraction

The images used thus far have been greyscale. Using RGB colour images, the heuristic information can be changed to guide the ants towards areas of the image with a prominent intensity of a particular colour.

For example, if the heuristic information is defined as,

$$\eta_{ij} = (R_{ij} - B_{ij}) + (R_{ij} - G_{ij}) \tag{3.16}$$

where R, G and B are the *Red*, *Green* and *Blue* channels of the image respectively, then the visibility for the ants becomes a weighted average of the red channel in the RGB image, such that the ants will converge on regions of the image with a prominent red intensity, with $T = 30$.

Figure 3.22 shows an example of using the above red weighted RGB heuristic information to allow the swarm to self-organise in response to red features in the image.

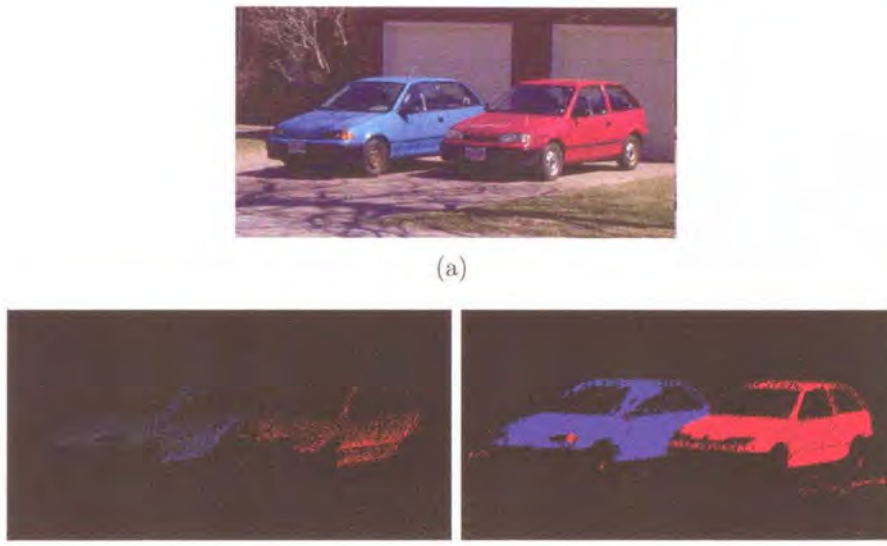
Figure 3.23 shows the results of two swarms searching the same image, with one swarm using RGB red weighted heuristic information, as defined above, and the other using RGB blue weighted heuristic information, i.e.

$$\eta_{ij} = (B_{ij} - R_{ij}) + (B_{ij} - G_{ij}) \tag{3.17}$$

In Figure 3.23(b) the red and blue pixels represent the positions of the swarm

(a) (b) (c)

Figure 3.22: An example of using RGB heuristic information for colour feature extraction, setting the visibility as a weighted average of the RGB red channel. (a) Original image; (b) agent positions; (c) pheromone map.



(a) (b) (c)

Figure 3.23: An example of using two swarms simultaneously to extract different features. One swarm has visibility as a weighted average of the RGB red channel, and the other with the blue channel. (a) Original image; (b) agent positions; (c) pheromone map.

programmed with RGB red and blue weighted heuristic information, respectively. Likewise, in Figure 3.23(c) the red and blue show the evolved pheromone maps from the two respective swarms. We can see how each swarm colony has converged onto the red and blue features within the image (in this case two parked cars), thus effectively resulting in a dual segmentation image represented by the evolved pheromone maps. In such a case, the two swarms are not interacting with one another, and are free to occupy the same pixels and move across one anothers paths.

3.7.2 Swarm COG Daemon Action

In Section 3.4 the daemon action was defined such that any ant agents with $\eta_{ij} < T$ for more than Z consecutive time steps would be terminated and replaced at a new 'random' location. For cases where the desired features are expected to be present throughout the whole image, as is often the case with image edge features (as used for the leaf pattern extraction), this set-up works well to expedite the

process of self-organisation of the swarm towards the locations of the desired features.

For the application of feature extraction there are instances where we wish the ant colony to converge onto a single specific area of the image, i.e. a feature in the image that appears only once, in a single area of the image. In such a case, a change to the daemon action can act as a better catalyst to the self-organisation process.

After each agent has moved, the Centre Of Gravity (COG) of the swarm within the image is calculated as,

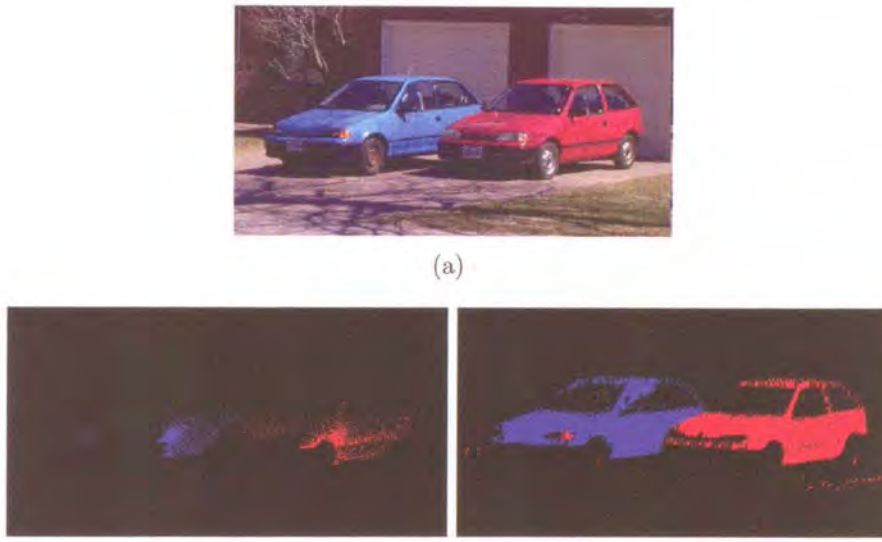
$$x_{COG} = \sum_n x_n/N,\ y_{COG} = \sum_n y_n/N \tag{3.18}$$

where x_n and y_n are the x and y coordinates of the nth ant.

A condition is then set such that as long as $\eta_{x_{COG},y_{COG}} \geq T$ then the terminated agent is immediately replaced at the new location of (x_{COG}, y_{COG}), else, the terminated agent is immediately replaced at a new 'random' location, as before.

Figure 3.24 shows results for the same set-up as for Figure 3.23, except this time using the COG daemon action. We observe better convergence of the two swarms on to the red and blue cars, respectively, with less 'outlier' agents searching undesired areas of the image. This behaviour is as expected, and highlights how such modifications to the algorithm can alter the global behaviour of the swarms. It should be noted however that there is little difference between the two output pheromone maps, as even with the many outlier agents in Figure 3.23(b), the swarm is large enough that there are enough agents over the desired regions to produce the desired pheromone map.

In a similar way, Figure 3.25 shows an example of two swarms self-organising to different feature types in an artificial image comprising of a grid pattern and a solid red square. One swarm is using RGB red weighted heuristic information, and the other using greyscale gradient information (as was used for the leaf venation extraction), also using the COG daemon action. Again we see how the red swarm has converged on to the red area of the image, with the green swarm mapping the prominent edge features. A similar example, this time using a real image, is given in Figure 3.26, showing similar characteristics. The use of the



(a) (b) (c)

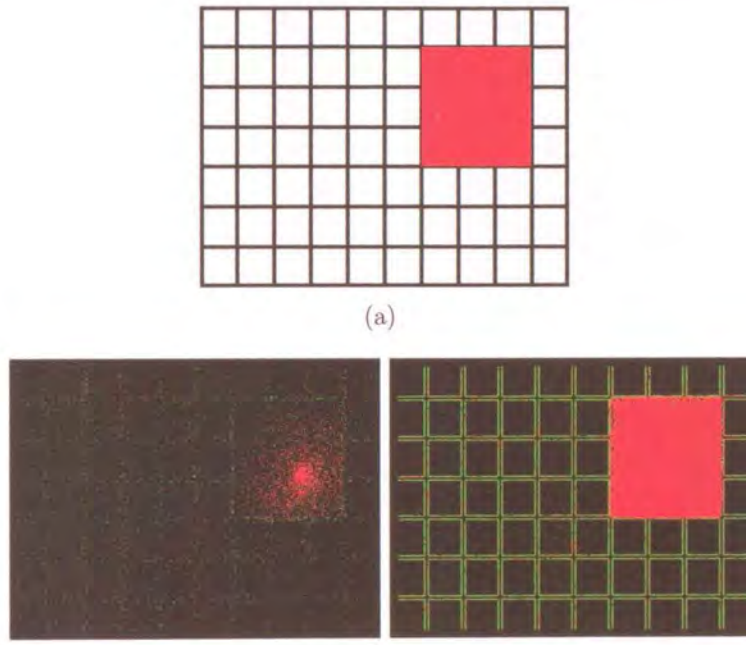
Figure 3.24: An example of using two swarms simultaneously to extract different features. One swarm has visibility as a weighted average of the RGB red channel, and the other with the blue channel. This time using the COG daemon action. (a) Original image; (b) agent positions; (c) pheromone map.

swarms COG will be explored more thoroughly in the next chapter, when feature tracking in dynamic imagery is considered.

3.8 Conclusions and Discussion

This chapter has investigated the self-organising properties of an ant-algorithm (based on the Ant System approach [22]) when placed in a digital, discretised environment, in the form of a digital image landscape. A new implementation of the Ant System algorithm has been developed, specifically tailored for image feature extraction.

As a novel application for ant-algorithms, a case-study in collaboration with RBG KEW for image feature extraction was carried out, where the algorithm was specifically set-up for the autonomous extraction of leaf outline and venation pattern, from digitally scanned images of live Quercus leaves. This case-study provided an application focus and example data-set in order to assess the performance of the algorithm for a specific feature extraction problem, and to study,



(a)

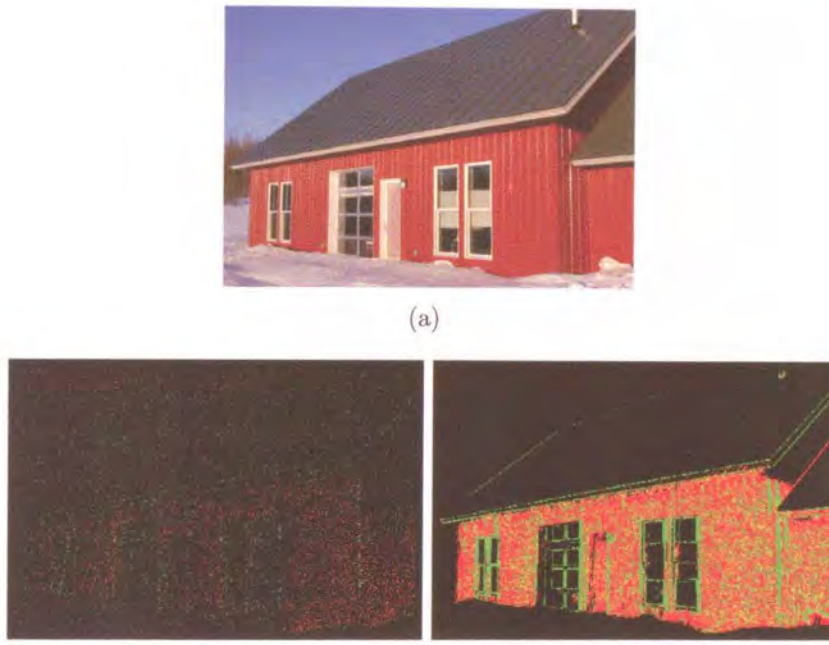
(b) (c)

Figure 3.25: An example of using two swarms simultaneously to extract different features from an artificial image, showing multiple feature self-organisation. One swarm has visibility as a weighted average of the RGB red channel, and the other with the image gradient heuristic information. (a) Original image; (b) agent positions; (c) pheromone map.

in more general terms, the way in which a swarm of embedded software agents can form complex patterns in response to local environment stimuli.

The performance of the algorithm has been evaluated (in the context of the case-study feature extraction problem) by a qualitative analysis of the output images as well as a quantitative analysis by statistical measures against ground truth images. The analysis showed the algorithm to work well, producing acceptable statistical accuracy (measured against ground truth data), and visually accurate representations of the desired features. The quantitative analysis has improved on previous work in the area of ant-algorithms for image processing, by providing statistical results to assess the performance of this approach, as well as investigating the performance in relation to a novel real-world application.

A comparison to well known existing methods for edge feature extraction yielded comparative results in terms of average accuracy achieved over a sample dataset,



(a)

(b) (c)

Figure 3.26: An example of using two swarms simultaneously to extract different features. One swarm has visibility as a weighted average of the RGB red channel, and the other with the image gradient heuristic information. (a) Original image; (b) agent positions; (c) pheromone map.

and a systematic exploration of the various algorithm parameters was carried out, with expected results, showing relative predictability of the algorithm. The comparison to existing methods and quantitative exploration of the algorithm parameters has provided further contributions to knowledge in the research area of ant-algorithms for image processing.

This analysis has pointed out some limitations such as sensitivity to noise. This however is not specific to the proposed approach, and is in fact a typical problem amongst edge detection and feature extraction algorithms. An optimised threshold value, or trade-off between exploration and exploitation, and positive and negative feedback within the system might help improve the robustness of the algorithm to noisy data. Specifically, it would be interesting to investigate if the swarm was able to self-adapt the exploitation/exploration ratio as the self-organisation process evolved, or the threshold value T (i.e. in a dynamic fashion), to achieve greater accuracy.

One of the main motivations behind the use of artificial swarms in the area of image processing is to improve on robustness and automation. Many traditional edge detection methods, such as the previously mentioned Canny edge detector [109], operate on the entire image in a linear fashion, and require a number of parameters to be tuned for a given image in order to produce satisfactory results. The ant algorithm approach works in a different way, relying on the phenomenon known as *stigmergy* [2] to produce self emergent behaviour amongst the artificial swarm in response to the given environment, which in this case is a digital image. When programmed with a search preference towards high image gradient change, the swarm converges onto the stronger edges of the image and the resulting pheromone field produced by the swarm maps out these edges. This convergence is reflected in Figure 3.6(a) with the rapid increase in accuracy over time during the early time-steps as the swarm is converging onto the image edges. After approximately 400 time-steps the gradient of the accuracy curve has almost dropped off to zero, as convergence is achieved. Through a systematic exploration of the main parameters however, a number of dependencies of the algorithm performance on parameter settings have been identified, which will be addressed further in the next chapter.

A potential advantage of the swarm intelligence approach is the potential to remove the requirement of a sensitivity threshold. In many traditional methods choosing an appropriate threshold value is of most importance in obtaining the best quality results, and indeed it has been shown that this is still the case with the ant-algorithm approach in its current state. The next chapter explores the potential of exploiting the multi-agent nature of the ant-algorithm approach, by investigating the use of distributed dynamic thresholds, whereby each individual ant agent has the ability to adjust its own threshold value as it moves through the image, thus removing the requirement for a user set, global threshold. A careful choice of all other parameters to optimise for image edge detection in general (i.e. not just leaf images) could provide a robust enough platform so as to not require any threshold value, and moreover, no parameter tuning at all, for different images.

The use of multiple swarms for multiple features is an interesting concept which deserves some discussion. Extracting multiple features in image processing is not a new concept, however using swarms in this context is. This method ex-

ploits the distributed temporal nature of the swarm approach to simultaneously evolve multiple feature maps and patterns from the image data by means of self-organisation of multiple swarms. This is similar to what has been observed in social insect colonies in nature, where instead of performing all the required tasks, a worker usually specialises in a set of tasks. This division of labour, whereby different tasks are performed simultaneously by groups of specialised individuals, is believed to be more efficient than if tasks were performed sequentially by unspecialised individuals [2]. By having multiple swarms using different heuristic information we are able to utilise this notion and observe simultaneous self-organisation in response to different features in a common digital image environment.

The next chapter focuses attention on developing the concept of parameter self-optimisation by allowing individual members of the swarm to dynamically adjust key parameters, with a view to increasing the swarms ability to adapt to different image environments without having to change user-set parameters, and thus increase the overall accuracy in terms of feature extraction. The research also concentrate on further exploiting the spatiotemporal evolutionary nature of the swarm approach, by applying the ant-algorithm method described in this chapter to real-time imagery, to facilitate real-time feature tracking.

Chapter 4

Swarm Adaptation, Dynamic Pattern Formation and Feature Tracking

4.1 Introduction

Within social insect societies in nature, different tasks are often performed simultaneously by specialised individuals; a phenomenon known as division of labour [2]. Using specialised workers for simultaneous task solving is believed to be more efficient, avoiding task switching, which costs energy and time. There are a number of methods of division of labour observed in social insects, such as *temporal polyethism*; where individuals of the same age tend to perform identical tasks, and *worker polymorphism*; where workers having different morphologies perform different tasks [2].

The distributed nature of swarm intelligence methods such as the ant- algorithm presented in the previous chapter allow us to further utilise such nature inspired approaches in order to further increase efficiency in the self-organising process and consequently improve the problem solving performance. Inspired by the distributed and adaptive specialisation observed in social insects, and in response to limitations made apparent in the previous chapter (discussed in section 3.8) with regards to the proposed ant-algorithm, this chapter explores parameter adaptation, exploiting the distributed nature of the embedded swarm by allowing

individual agents within the swarm to adapt their individual parameters for a given environment. This approach is likened to the specialisation of individual worker agents as observed in the wild, and as was implemented in the previous chapter for multiple swarms tracking multiple features in digital imagery (section 3.7).

A number of studies suggests that individual experience shapes behavioural ontogeny, and that thresholds for responding to given stimuli may be dynamic, rather than static (see for example [110] and a short survey can be found in [2]). In this chapter, focus is placed on adaptive specialisation through dynamic thresholds, giving individual agents the ability to change their behaviours in response to perceived changes in the environment.

Additionally, this chapter builds upon the work of the previous chapter, introducing dynamic environments such that the swarm has the additional challenge of adapting to changing environment structures, involving adaptive self-organisation and pattern formation. Furthermore, the self-organisation is exploited to facilitate feature tracking within dynamic image environments.

4.2 Related Work

As previously discussed, extensive work has been carried out to study self-organisation in swarm-based systems, and in many cases this is synonymous with adaptation in terms of swarm behaviours, as self-organisation often implies some form of adaptation, for example in terms of the changing, or emerging structure of the swarm in pattern formation. An example of such adaptation in nature is the way in which bee's use dancing as a means of recruitment in foraging tasks. In order to focus foraging towards the best food sources, as well as refocus foraging in response to variations in available forage, honeybees adjust both the duration and vigor of their dancing as a function of profitability of their current source [111]. A forager uses an internal gauge to assess the profitability of their source. The bee's nervous system has a threshold calibrated into it, which the bee uses to weigh variables when deciding whether a patch is worth foraging, and if so, whether it is worth advertising to fellow workers [111]. There is an analogy to be drawn here with the application of the ant swarm to digital imagery. The internal

gauge of each individual ant is the threshold T, which governs whether or not an ant should deposit pheromone (thus advertising to fellow agents), depending on the local visibility, η, (i.e. the profitability) associated with a given move. This chapter examines a method of adaptation of the parameter T in response to variation in the visibility, akin to how honeybee's vary their dancing in response to variation in available forage.

The previous chapter examined swarm self-organisation in digital image environments, and also discussed a number of related works in this area. Despite this, there is little evidence in the literature of studies involving swarm-based methods applied to dynamic imagery. In [38] the authors study the self-organising properties of a swarm embedded in a digital image environment, including the swarms ability to re-adapt as the image environment is swapped with another one. In [39] the authors examine the ability of the swarms to self-regulate their population size for given image environments, and show that regulating the population size can result in faster convergence times of the swarm to adapt to a particular image structure. The same authors extend their research in [112] to investigate the swarms ability to search for peaks and valleys in dynamic 3D landscapes represented by mathematical functions.

In this chapter the concept of swarms adapting in dynamic 3D digital environments is further investigated. Research in this area is extended by providing a quantitative analysis of the adaptive process by statistical evaluation of the adaptive process in terms of pattern adaptation in response to changing image environments. Furthermore, the ant algorithm presented in the previous chapter is applied to real time imagery to study the self-organisation process in a highly dynamic environment, as well as to evaluate the effectiveness of this approach to image and video feature tracking. By using real-time imagery the environment becomes highly dynamic, offering wide ranging characteristics in terms of, for example, structure, features and rate of change. As mentioned previously there is also the practical consideration of potential application of this research to image and video processing, which shall be discussed in more detail throughout the chapter.

4.3 Adaptive Parameters

The varied effects of the different parameters for the proposed ant-algorithm were explored in the previous chapter. Some parameters, and in particular the threshold parameter T[1], had a greater effect than others on the swarm self-organisation process, and particularly it was clear that certain parameters needed fine-tuning for optimal performance with respect to a particular image environment.

In nature, the requirement for information transfer in swarms is well studied, and it is now well known that decision making in insect societies is decentralised. Actions by different group members need to be carefully tuned in order to achieve adaptive behaviours at the group level [111]. In the computerised world the same holds, and, as was studied in the previous chapter, algorithm parameter counterparts need to be well tuned in order to fully exploit the self-organisation process. For situations where the environment is dynamic, a specific set of tuned parameters might not hold as appropriate for the varying characteristics of that environment. This section explores methods of automated parameter adaptation focusing on the threshold value T.

4.3.1 Adaptive Threshold

The role of the threshold was explored in the previous chapter, and it was shown that using different values of T can significantly affect the performance of the algorithm, with a particular value of T being the optimum. Since finding the optimum, or even a satisfactory value of T can be very time consuming, attention is focused here on the use of an automated adaptive threshold. The idea of the adaptive threshold is to remove the requirement of a user-set threshold, which would otherwise need to be manually selected for different datasets and/or specific images.

The approach taken here focuses on exploiting the distributed nature of the ant-algorithm. Instead of employing a global threshold (where all ant agents use the same value of T), each of the $n \in N$ individual agents are allowed to maintain their own threshold value T_n. This approach is useful for dealing

[1] T is a user defined threshold value that can be set to only allow pheromone deposition by ants following edges or boundaries above a certain 'strength.'

with image environments where different areas of the image have features of interest requiring different threshold values (for example a scanned image with varying light intensity levels, or the case in the previous chapter where primary and secondary leaf venations in scanned leaf images required different threshold values). Adapting individual agent's parameters that affects the overall global behaviour of the swarm is analogous to adaptation techniques used by swarms in nature, such as the previously mentioned honeybee's adapting their individual methods of communication to change group level foraging [111].

Since the ants are (at least initially) distributed throughout the image environment, each individual agent sets its initial T_n value according to its local neighbourhood. The adaptive threshold is initialised such that, for agent n, at timestep $t = 1$,

$$T(1)_n = \sum_{S=1}^{8} \eta_S / 8 \tag{4.1}$$

where η_S here is the visibility of the 8 surrounding pixel locations.

This allows an automated initial 'first guess' at the threshold value. For subsequent time-steps the adaptive threshold is defined by the following update rule:

$$T(t)_n = \begin{cases} T(t-1)_n + 1 & \text{if } \eta_n(x,y) > T(t-1)_n \\ T(t-1)_n - 1 & \text{if } \eta_n(x,y) < T(t-1)_n \\ \bar{T} & \text{if } T(t-1)_n < \bar{T} \end{cases} \tag{4.2}$$

where $\eta_n(x,y)$ is the current visibility of agent n at pixel location (x,y), and $\bar{T}$ is the average threshold over all agents.

This novel update rule allows individual ant agents to increase or decrease their individual threshold values depending on their individual visibility perception where they are in the environment, while maintaining a threshold that is above the level of background noise, as calculated from the global perception of the entire swarm.

Figure 4.1(a) compares results averaged over the 20 leaf image test dataset used in the previous chapter. The plot shows the accuracy versus timesteps for the previously used global $T = 20$ (determined by trial and error to give good results

across the leaf dataset), and for the adaptive threshold method, $T_{adaptive}$. The plot also shows how the average threshold value ($\sum T_n/N$) varies over time. The accuracy here is the statistical accuracy calculated by comparison to ground truth images (as defined in the previous chapter, section 3.5). We see that the $T_{adaptive}$ method has resulted in less accuracy than for the static $T = 20$ threshold, which was chosen by systematic trial and error. This suggests that the adaptive method is not finding the optimum value of T. However, from qualitative visual inspection, the results obtained are still of a good quality (with a statistical accuracy of approximately 0.009 less than with $T = 20$). We see that on average, the average threshold converges to a value of approximately $\bar{T} = 23$, which is close to the value obtained via systematic trial and error.

Figure 4.1(b) shows a similar plot for the image shown in Figure 4.2(a).[2] We notice how the accuracy for the adaptive threshold and the static threshold of $T = 20$ have converged to approximately the same value in this particular case. It is interesting to note that the average visibility threshold has converged close to $T = 20$, which would account for the similar performance in this case. Figure 4.1(c) shows results for the image in Figure 4.2(d), this time showing the adaptive threshold method to yield higher accuracy. If we draw our attention to the corresponding evolved pheromone maps (Figure 4.2(e),4.2(f), for $T = 20$ and $T_{adaptive}$, respectively), we observe over segmentation for the case of $T = 20$, with $T_{adaptive}$ suffering less from this problem. Figure 4.1(c) shows the average visibility threshold to converge close to $T = 40$ in this case, which accounts for the difference in accuracy from the different instances, and the fact that the $T = 20$ instance resulted in over segmentation. We see similar, less pronounced results in Figure 4.1(d) and Figure 4.2(g),4.2(h),4.2(i), again with the $T = 20$ instance showing greater over segmentation.

It should be noted that in Figure 4.1(b),4.1(c),4.1(d), the accuracy for the $T = 20$ instances decrease with time (in contrast to Figure 4.1(a)). This is because in the ground-truth data for these images the edge features are defined as 1 pixel width boundaries (as opposed to the 3 pixel width used in the leaf image ground truth data). Along with the already high true negative count, this results in a particularly high accuracy to start with. For this comparison, of most

[2]The images in Figure 4.2 are from a dataset which includes manually created human ground truth edge maps similar to those shown in chapter 3 for the leaf images. Further details can be found in [113].

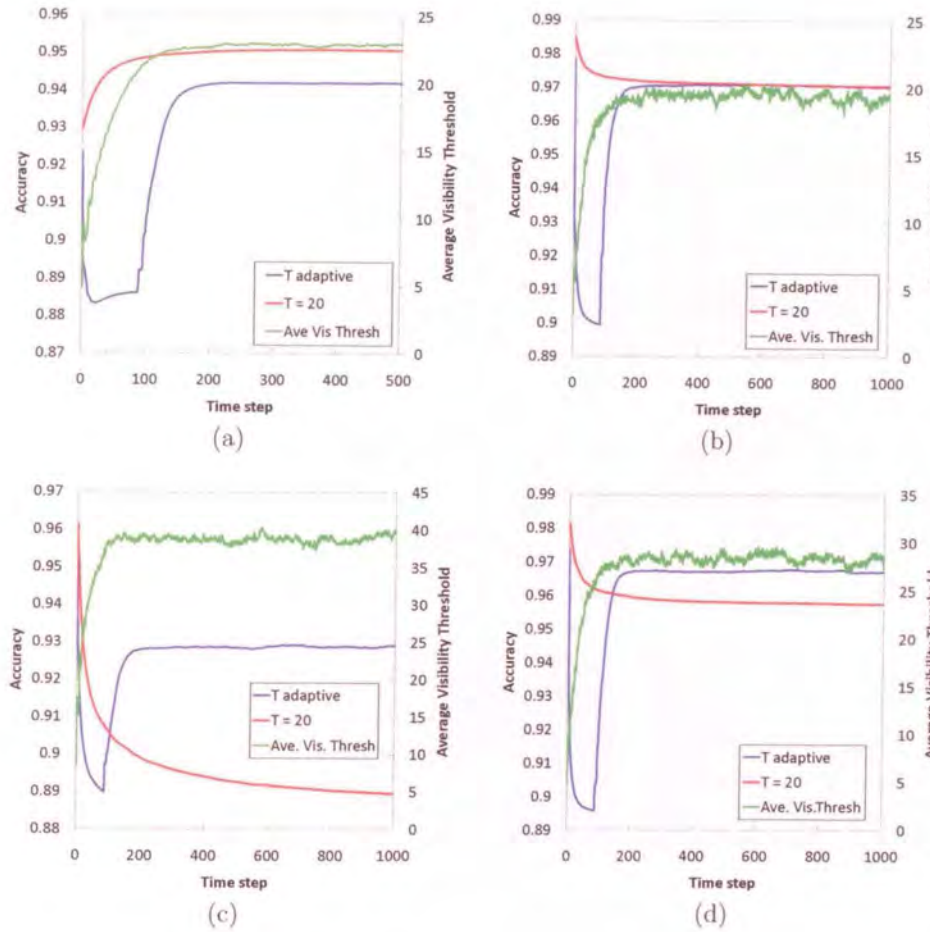


Figure 4.1: A comparison of the accuracy versus time-steps, using the distributed adaptive threshold, and global static $T = 20$ separately. (a) Averaged over the 20 images from the RBG KEW leaf data-set used previously; (b) for the image in Figure 4.2(a); (c) for the image in Figure 4.2(d); (d) for the image in Figure 4.2(g). The plots also include the average visibility threshold of all the agents (when using the distributed adaptive threshold), showing how the threshold adapted over time.

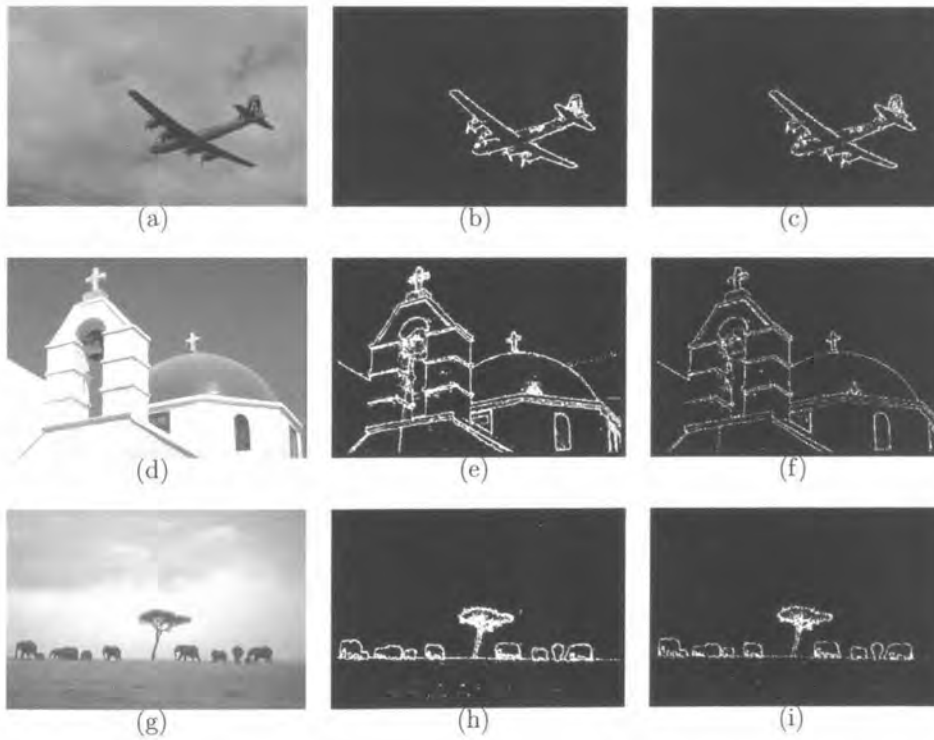


Figure 4.2: Example images (a,d,g) showing the evolved pheromone map using the distributed adaptive threshold (c,f,i), comparing the the evolved pheromone map with global static $T = 20$ (as used previously for the leaf image data-set) (b,e,h).

interest however is the final accuracy, once the swarm has converged, as a direct comparison of performance. The nature of the curve is of interest with respect to analysing the self-organising behaviour of the swarm (i.e. the convergence rate). The reader is referred to section 3.5.1 in the previous chapter for a more detailed discussion on the ground truth process.

Analysing the curves for the adaptive threshold accuracy, we can assess the different self-organisation process that is occurring with this method. The initial drop in accuracy seen with this method results from the fact that in the early time-steps the majority of the agents will have a low threshold (as can be seen with the corresponding average visibility threshold curves), due to many of the agents starting in 'background' regions of the image. This results in a rapid build-up of pheromone in the background regions of the image, thus reducing the statistical accuracy. As the swarm begins to converge onto the edge regions, simultaneously the pheromone in the background regions will begin to evaporate, and this is where we observe a rapid increase in the accuracy until the swarm converges and the accuracy curve levels out.

Figure 4.3 shows additional example comparison images of the evolved pheromone map obtained using $T = 20$ and $T_{adaptive}$, for a range of example images. Qualitatively, from visual inspection we see that in these cases, the threshold of $T = 20$ produces an over evolved pheromone map (or over segmented image), with the adaptive threshold producing more visually concise results, qualitatively better representing the pattern of edge features in the images. These are examples of where, in the manual case, one would have to go through the often lengthy process of trial and error again, in order to find the optimum value of T for these different images. Using $T_{adaptive}$ however does not require this process, and although it may not find the optimum threshold, it does produce near optimum results.

Figure 4.4 shows the distribution of the individual agents threshold values at the beginning and end of an algorithm run for a typical leaf image, using $T_{adaptive}$. We can see from the two histograms that the distribution has effectively shifted from a majority population with a threshold value of $T = 1$ at the beginning, to a majority population with a threshold value of $T = 22$ at the end. This is consistent with the majority of the swarm starting on background areas of the image, thus from Equation 4.1 the threshold will be relatively small owing to

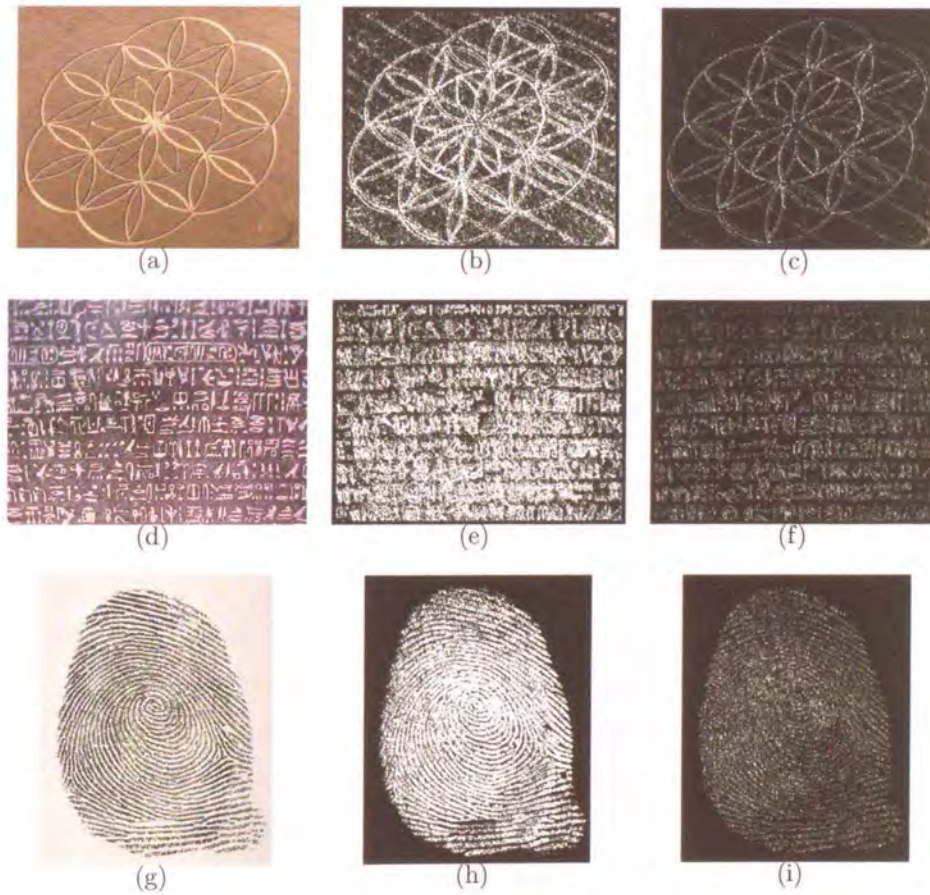


Figure 4.3: Example images (a,d,g) showing the evolved pheromone map using the distributed adaptive threshold (c,f,i), comparing the the evolved pheromone map with global static $T = 20$ (as used previously for the leaf image data-set) (b,e,h).

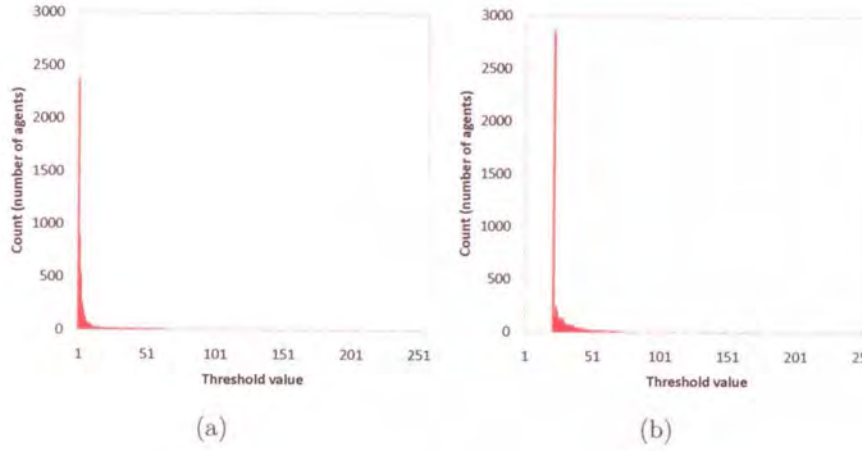


Figure 4.4: Histogram plots showing the distribution of individual threshold values in the swarm population, (a) at the beginning of the algorithm run ($t = 1$), and (b) at the end of the algorithm run ($t = 1000$), for a typical leaf image.

only minor changes in gradient on the background areas of the image. As the swarm self-organises onto the edge regions of the image, the agents adjust their threshold values accordingly, giving rise to the shift in threshold population we see in Figure 4.4(b).

This work with the adaptive threshold goes some way to consolidating Hypothesis 2 (stated in chapter 1), showing convergence of $T_{adaptive}$ through a novel homogeneous adaption rule, allowing the distributed swarm to adapt across a varying environment to optimise for specific regions of the environment, or for specific environment characteristics.

It is clear that different images will inevitably require different thresholds in order to produce the desirable segmentation / edge map. Inspired by swarm adaptation in the wild, the adaptive method presented here, although it does not guarantee the optimum results, it does provide acceptable results, without the requirement for any threshold. This is achieved through exploiting the distributed nature of the ant-algorithm approach, and the self-organising nature of the embedded swarm. The parameter adaptation here is effectively allowing the swarm to learn appropriate threshold values for a given image environment. The adaptation process occurs simultaneously with the pattern self-organisation of the swarm, complimenting the temporal nature of the ant-algorithm, and this approach to the specific image processing problem setting.

4.4 Pattern Adaptation

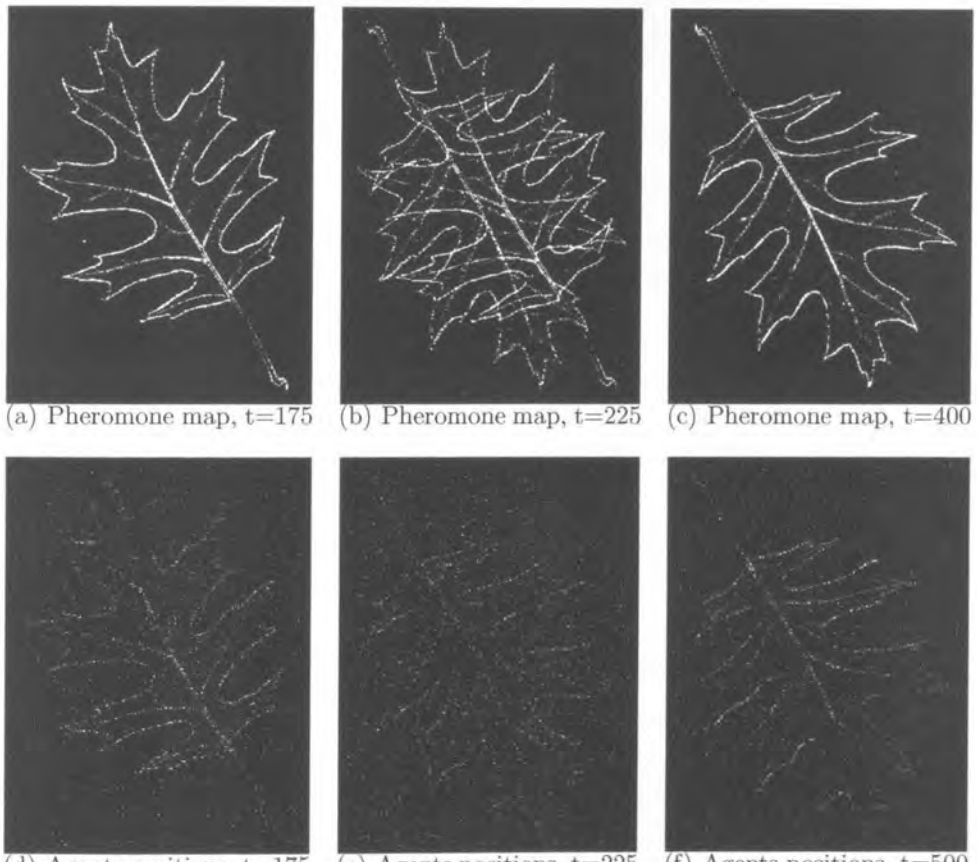
The self-organisation methods of the ant-algorithm approach, and indeed of the swarm intelligence paradigm in general, are inherently adaptive. This section focuses on further exploiting such adaptive nature for the purpose of pattern adaptation, again employing the ant-algorithm approach in digital image environments.

To study the swarms ability to adapt to changing environment structures we can consider the problem of allowing the swarm to self-organise to a particular static image, and then change the environment to another image, and observe the adaptation process. Figure 4.5 shows an example of the swarm adapting to a changing image environment. From $t = 1$ to $t = 199$ the image environment is that of Figure 4.6(a), then at $t = 200$ the image environment is switched to that of Figure 4.6(b). The algorithm run is the same as that defined in chapter 3.4.

In Figure 4.5 the adaptation process is visualised by showing the changing pheromone map and agents positions. Figures 4.5(a) and 4.5(d) show the evolved pheromone map and agents positions respectively, at $t = 175$, as the swarm has adapted to the original image (Figure 4.6(a)). Figures 4.5(b) and 4.5(e) show the same at $t = 225$, shortly after the image environment has been rotated 180 degrees clockwise (Figure 4.6(b)), and Figures 4.5(c) and 4.5(f) show the same at $t = 400$, after the swarm has re-adapted to the new image environment. From visual inspection we can see how the swarm has adapted to the change in environment, re-adapting to the new image pattern. Figure 4.5(b) shows the new pattern emerging in the pheromone map, while the old pattern is evaporating away.

Figure 4.7 shows a plot of the accuracy versus time-steps for the adaptive experiment. We observe an increase in accuracy to a plateau at approximately $t = 100$ as the swarm is adapting to the original pattern. At $t = 200$, as the image is rotated, we see a sharp drop in the accuracy, as the evolved pheromone map no longer represents the pattern of the image. We then observe an increase in accuracy to approximately $t = 250$, as the swarm adapts to the new image pattern, and the old pheromone map evaporates, with the new pheromone map emerging. Again, a plateau is reached as the swarm has self-organised to the new pattern.

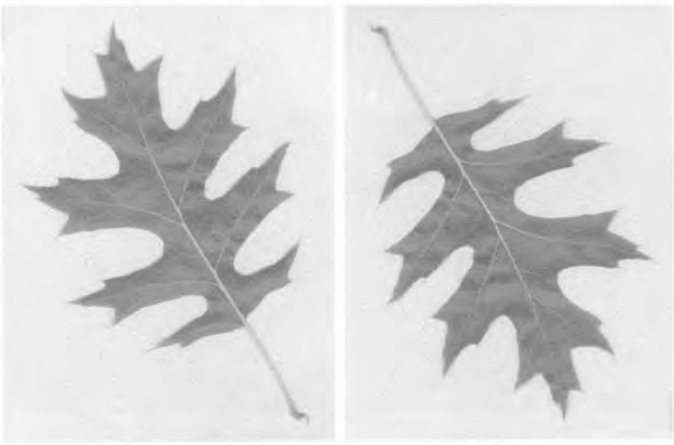
Figures 4.8 and 4.9 give further examples of pattern adaptation. From $t = 1$ to



(a) Pheromone map, t=175 (b) Pheromone map, t=225 (c) Pheromone map, t=400

(d) Agents positions, t=175 (e) Agents positions, t=225 (f) Agents positions, t=500

Figure 4.5: An example of the swarm adapting to a change in environment structure, showing the evolving pheromone map (a-c) and agents positions (d-f), at different time-steps. At $t = 200$ the image environment is rotated 180 degrees clockwise.



(a) (b)

Figure 4.6: (a) Initial and (b) rotated image environment.

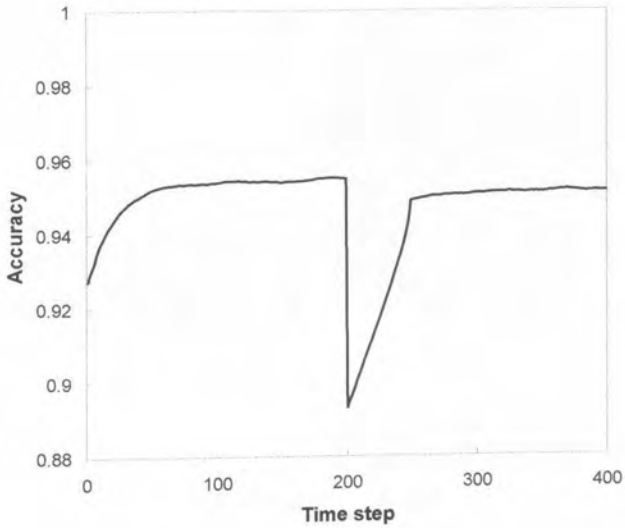


Figure 4.7: Accuracy versus time-steps for a changing image environment (corresponding to Figures 4.5 and 4.6).

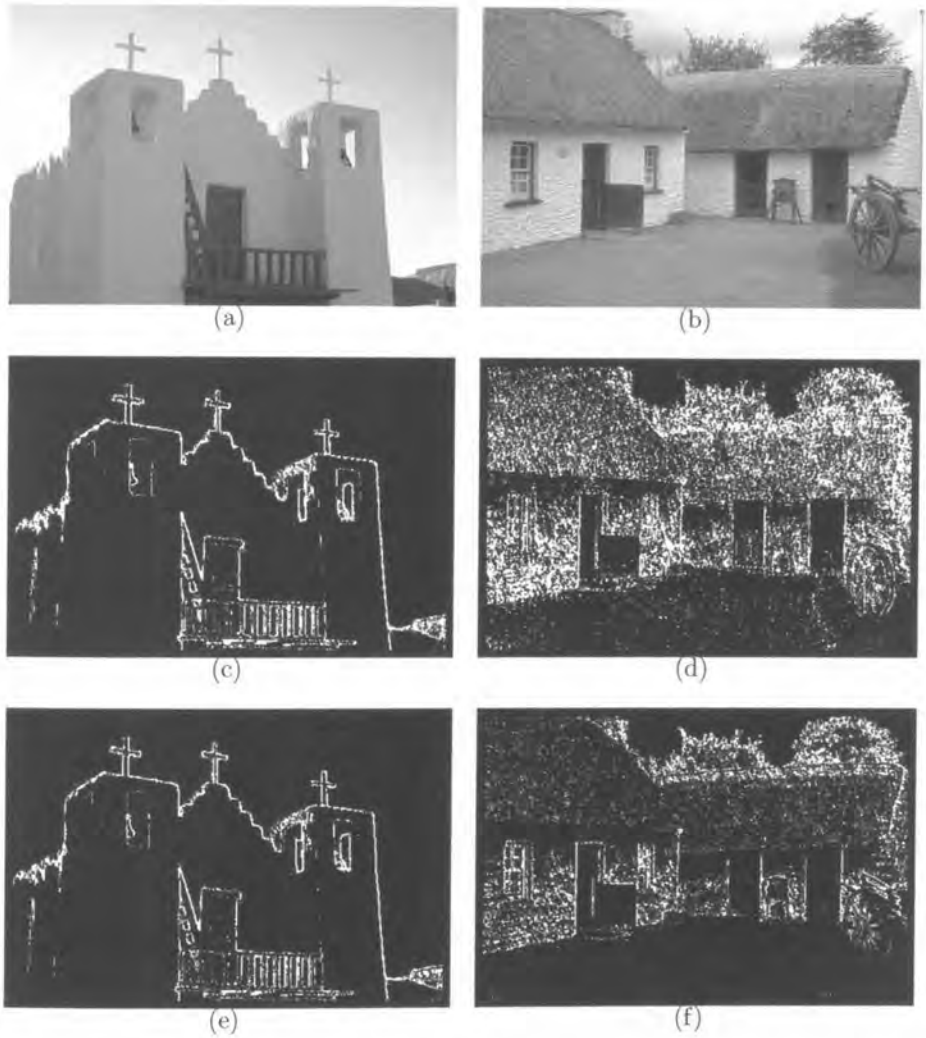


Figure 4.8: An example of the swarm adapting to a change in environment structure, comparing the algorithm with global static $T = 20$ ((c) at $t = 495$, (d) at $t = 1000$) and $T_{adaptive}$ ((e) at $t = 495$, (f) at $t = 1000$). The image is swapped from (a) to (b) at $t = 500$.

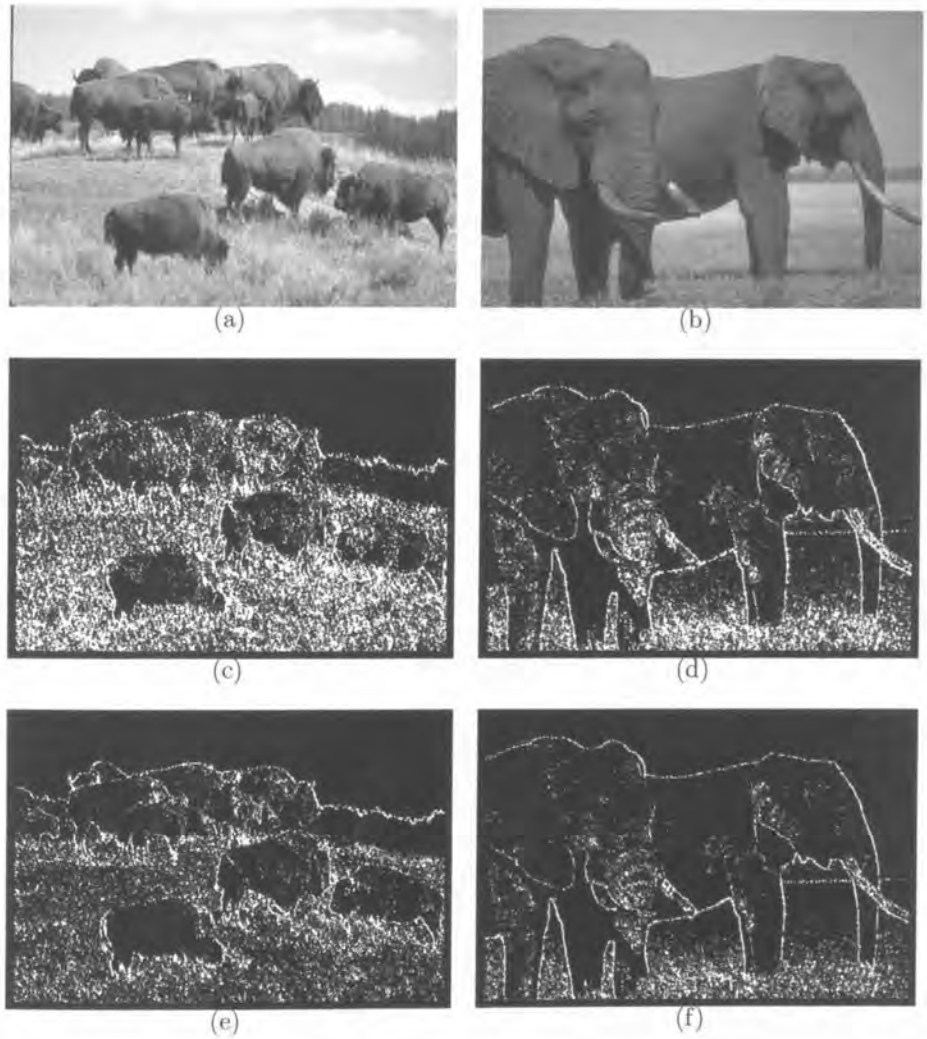


Figure 4.9: An example of the swarm adapting to a change in environment structure, comparing the algorithm with static $T = 20$ ((c) at $t = 495$, (d) at $t = 1000$) and $T_{adaptive}$ ((e) at $t = 495$, (f) at $t = 1000$). The image is swapped from (a) to (b) at $t = 500$.

$t = 499$ the algorithm runs with images (a), then swaps to images (b) for $t = 500$ to $t = 1000$. Images (c) and (e) give the evolved pheromone maps at $t = 495$ for the algorithm run with a static $T = 20$ and $T_{adaptive}$, respectively. Images (d) and (f) show the same at time $t = 1000$. Corresponding plots of accuracy and average visibility threshold versus time-steps are given in Figure 4.10.

Again in these two examples we observe that the pheromone maps have evolved to the edge feature patterns of the initial image, and then adapted to the second images, with the previous pattern being fully evaporated over time. From the accuracy plots in Figure 4.10 we can see that for both examples, and both methods, the adaptation process takes approximately 200 time-steps. For the adaptive threshold method, it is interesting to analyse the change in average visibility threshold of the swarm, during the adaptation process. The change in image for the example in Figure 4.8 results in an increase in average visibility threshold, while for the example in Figure 4.9 there is a decrease. In both cases the resultant accuracy for the adaptive threshold method is greater. The difference in converged accuracy between $T_{adaptive}$ and $T = 20$ in Figure 4.10(a) for the initial image is small, as the average visibility threshold is close to 20, then as the image changes, and the average visibility threshold increases to approxi-

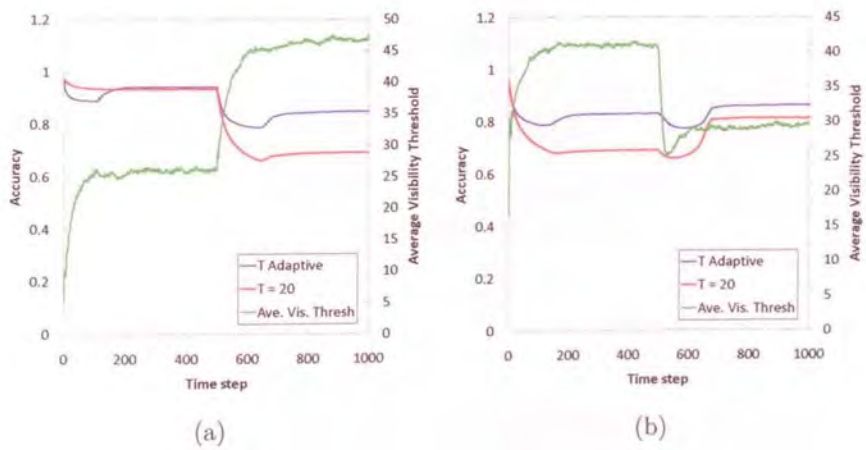


Figure 4.10: A comparison of the accuracy versus time-steps, using the distributed adaptive threshold, and global static $T = 20$ separately. (a) Plot corresponding to Figure 4.8, and (b) corresponding to Figure 4.9. The plots also include the average visibility threshold of all the agents (when using the distributed adaptive threshold), showing how the threshold adapted over time.

mately 47, the difference in accuracy also increases. For both methods there is a decrease in accuracy in the first example and an increase in the second, when the image is swapped. This is reflected in the visually over-segmented images in the emerged pheromone maps in Figure 4.8(d),4.8(f) and Figure 4.9(c),4.9(e).

The adaptive threshold is shown here to increase the adaptable capabilities of the swarm, by allowing the swarm to autonomously change a key parameter in response to a change in the perceived environment and adapt accordingly. This results in adaptation on two levels: (i) the swarm physically adapts its structure and movements by self-organising to the environment features, exhibiting *group level* adaptive pattern formation; (ii) individual agents adapt their threshold values in response to the change in image environment, thus adapting their response to the environment in terms of *individual* movement and pheromone deposition.

The physical adaptation is fundamental to the ant-algorithm design. The individual parameter adaptation augments the adaptive capabilities of the swarm allowing for autonomous, *in-situ* distributed parameter self-optimisation.

4.5 Real-Time Feature Tracking

The adaptive nature of the ant-algorithm approach can be exploited by applying the algorithm to real-time imagery; such that the image environment is continuously changing, as new image frames are captured by the imaging device. In this way, the swarm will effectively track the image features from frame to frame, continuously self-organising to the changing environment.

4.5.1 Edge Tracking

Figure 4.11 shows example snap-shots of the swarm tracking image edge features; using the ant-algorithm described in chapter 3, section 3.4. The input images are from a camera on-board a small mobile robot (a Surveyor SRV-1), running at a resolution of 320 by 256 pixels at approximately 8 frames per second. In the given example the robot was made to rotate anti-clockwise on its axis in the ground plane, and snap-shots were taken at frames 100, 300, 500 and 700.

From visual inspection we can see how the swarm adapts to the continuously

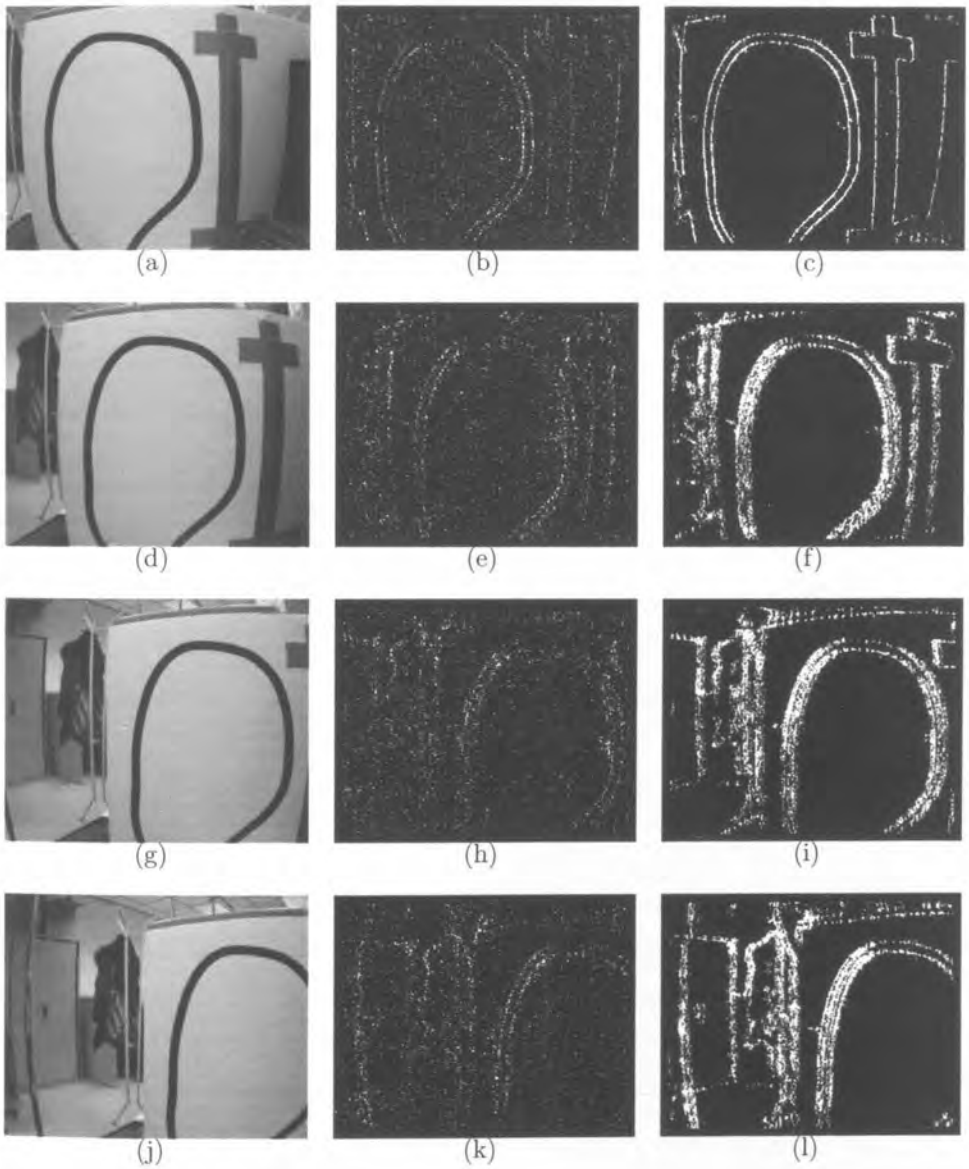


Figure 4.11: Example frames showing real-time edge tracking. Left column: original greyscale image; centre column: agent positions; right column: pheromone map. (a,b,c) frame 100; (d,e,f) frame 300; (g,h,i) frame 500; (j,k,l) frame 700.

changing environment structure, with the pheromone map showing clearly the prominent edge features within the image frames, and the agent positions reflecting the self-organisation of the swarm onto the prominent edge features across the changing image frames.

The algorithm used is the same as defined in the previous chapter, but is implemented to run with no time-limit, with the image environment being continuously updated as new image frames are received from the video camera. The following parameter settings were used: $N = 5000$, $tabu_{max} = 8$, $\alpha = 1$, $\beta = 5$, $\rho = 0.08$, $T = 18$. These parameters were again chosen through systematic trial and error. $tabu_{max}$ is set lower for the real-time edge tracking to impose less restrictions on the agents movements, given the continuously changing environment. Likewise, ρ is set higher, to enable the pheromone map to quickly evaporate edge features from previous image frames. Another difference is that the daemon action is not used in this implementation, as for the example given, the edge features exist throughout the entire image (as opposed to specific areas as was the case for the leaf images), and given that the edge features will often only be moving small distances from frame to frame, it would be detrimental to continuously replace agents considered 'lost,' as invariably they would in fact not be 'lost,' but instead re-adapting to the new image frame.

As with the pattern adaptation in section 4.4, it is possible to apply the adaptive threshold for real-time adaptation to augment the swarm self-organisation. Figure 4.12 shows example snap-shots from a static USB webcam (at 320 by 240 pixels resolution) capturing images at approximately 7.5 frames per second. In this example the algorithm is run with the adaptive threshold applied, and the exposure of the camera is increased then decreased over a number of frames. Figure 4.13 shows a plot of the average visibility threshold, plotted with the exposure level of the camera, versus time-steps (corresponding to the example image frames given in Figure 4.12).

As the exposure is increased, so does the number and intensity of the edge regions in the image. With the adaptive threshold method this results in an increase in the average visibility threshold of the swarm, allowing the swarm to converge onto the emerging edges, and resulting in an emerging pheromone map that represents the emerging edge features. A similar behaviour is observed when reducing the exposure, as can be seen in Figure 4.13.

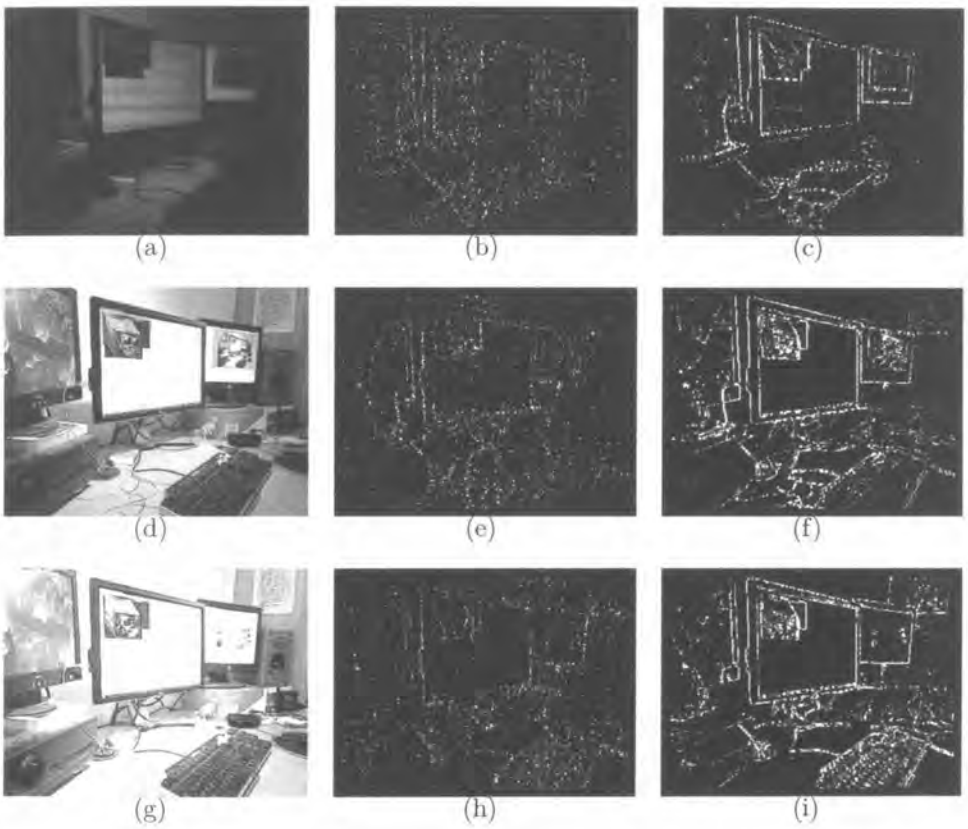


Figure 4.12: Example frames showing real-time edge tracking with the adaptive threshold. Left column: original greyscale images; centre column: agent positions; right column: pheromone maps. (a,b,c) frame 100; (d,e,f) frame 300; (g,h,i) frame 500. The input image exposure is increased from top row to bottom.

Again this is of particular advantage for the application of image and video processing; allowing the algorithm to autonomously adapt the threshold value for varying image properties such as different lighting conditions. From the swarm perspective, the distributed temporal properties of this approach are being further exploited, allowing individual members of the swarm to adapt over time, in response to local perception of the environment, resulting in a global, swarm level adaptation.

Although the adaptive threshold is based on the visibility of the agents (which in this case is itself based on image gradient information), where there is a low light level (or low exposure), this results in less prominent edge features, and hence the adaptive threshold is effected by the changing exposure levels.

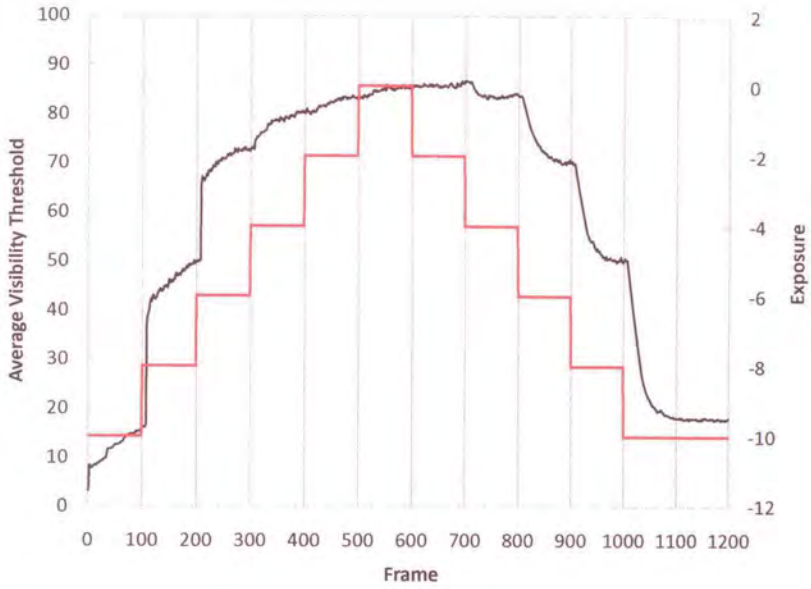


Figure 4.13: Showing the average visibility threshold of the swarm (black line) and the exposure setting of the camera (red line) versus image frames. This plot corresponds to the images given in Figure 4.12.

Figure 4.14 gives a further example, where the web camera is slowly tilted upwards, from facing downwards at an empty desk surface to view a computer keyboard and screen. Figure 4.15 gives the corresponding plot, showing again the average visibility threshold versus time-steps, and also the input image variance. This example illustrates how the adaptive threshold is changing in accordance to increasing levels of features in the image (characterised by the increasing variance, as more objects appear in the camera Field Of View (FOV)), showing the average visibility threshold of the swarm increasing in response to the increasing image variance.

The individual agents change their T parameters in response to changes in the environment as they perceive it locally. The goal of the agents is to track prominent edge features in the image environment. By changing T based on local sensing of the environment, an individual agent is able to 'self-manage' its own responsiveness to the edge features in its vicinity. Additionally, this changes the pheromone deposition from the individual agent, which in turn affects other agents in the vicinity, thus affecting the global behaviour of the swarm.

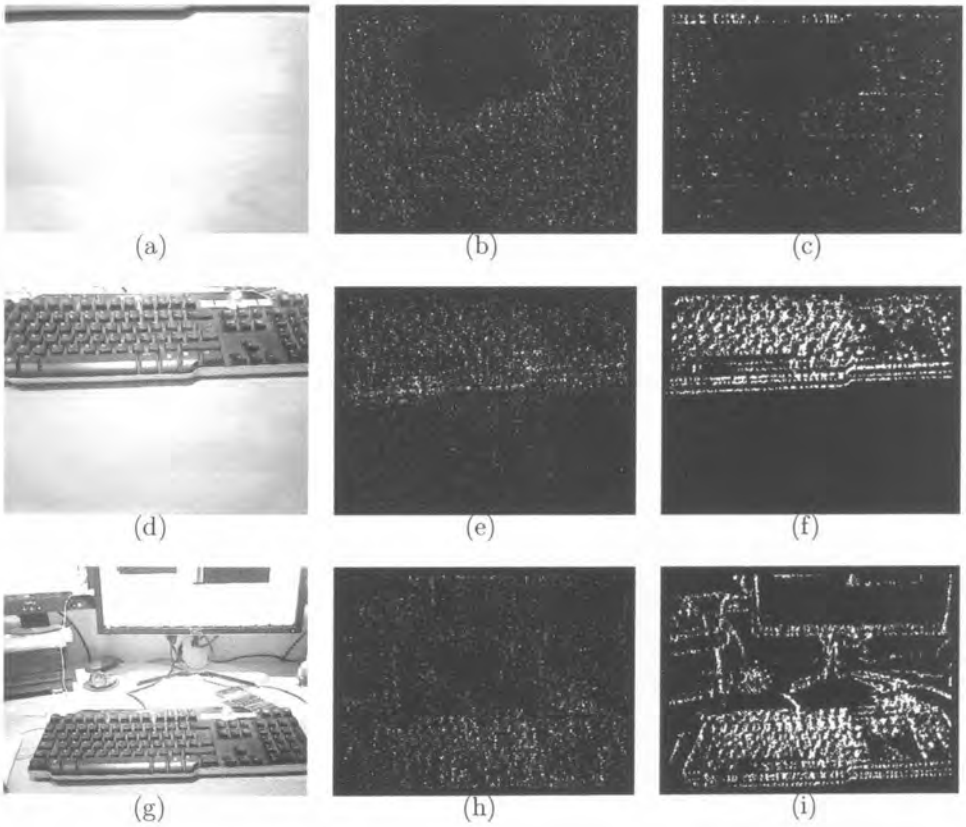


Figure 4.14: Example frames showing real-time edge tracking with the adaptive threshold. Left column: original greyscale images; centre column: agent positions; right column: pheromone maps. (a,b,c) frame 100; (d,e,f) frame 500; (g,h,i) frame 1000.

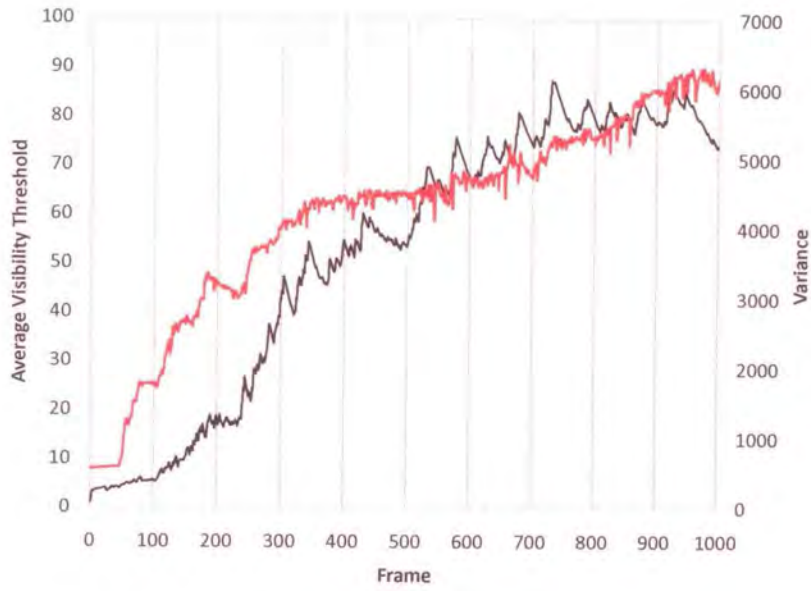


Figure 4.15: Showing the average visibility threshold of the swarm (black line) and the variance of input image (red line) versus time-steps. Plot corresponds to the images given in Figure 4.14.

Case Study in Robot Vision Corridor Following

The real-time edge tracking implementation of the ant-algorithm is assessed by applying it to a simple robot vision task of corridor following. The swarms ability to continuously track the edges in the robots forward FOV (via an on-board video camera) will determine the robots ability to follow along the centre of the corridor, by estimating the current direction of the robot with respect to the corridor walls, and sending appropriate commands to the wheel motors.

The ant-algorithm applied to the video feed from the on-board camera is the same as described in the previous section on real time edge tracking (based on the ant-algorithm defined in chapter 3). After each agent has moved the Centre Of Gravity (COG) of the pheromone map is calculated and the position of the COG is used to direct the robot's movement. From detecting the edges along the walls of the corridor the robot is programmed to track along the centre of the corridor, by employing a simple rule-set to determine whether to robot should turn left, move forward, or turn right, depending on where the edges of the corridor environment lie in the robot's FOV.

Figure 4.16 shows example snap-shots from the robot's on-board camera with

the robot placed in the corridor environment. The snap-shots show the original image with pheromone map and COG superimposed, for a left turn, forward, and right turn situation.

The COG is calculated from the pheromone in the lower third of the image, since the walls of the corridor will appear in this region of the image when in close proximity to the robot. The robot movement is then determined by the following rule-set:

$$Movement = \begin{cases} Left & \text{if } COG_x > Centre + deadzone \\ Right & \text{if } COG_x < Centre - deadzone \\ Forwards & \text{otherwise} \end{cases} \tag{4.3}$$

where *deadzone*, is a sensitivity parameter defining the distance, plus or minus, from the centre of the image, for in which if COG_x lies there is no left or right movement required. A value of $deadzone = 20$ is used here, determined empirically.

For this case study experimentation is carried out using a single e-puck robot [114]. The e-puck is a small-scale differential drive laboratory robot approximately 7cm in diameter. It has a number of on-board sensors including: a forward facing camera, 8 IR proximity sensors, accelerometer and 3 microphones. It is also equipped with a speaker, an LED ring, and supports Bluetooth communication. For this experiment we only use the motor functionality of the robot, and attach a higher resolution (320 by 240 pixels) USB webcam to the front of the robot, to allow for a higher framerate (approximately 8 frames per second). The ant-algorithm runs continuously on an external computer, with the image environment updating with each new image frame received from the on-board camera. With each loop through the algorithm, the COG is calculated and the desired movement direction determined from Equation 4.3. The corresponding motor commands are then sent to the robot via bluetooth communication. The experiment is carried out in a purpose build robot enclosure measuring approximately $120cm$ by $120cm$, with a 'U' shaped corridor in which the robot is placed at one end. The purpose built enclosure includes an overhead vision based tracking system (see Appendix B for details) to record the robots (x, y) positions in the ground plane within the enclosure, to $\pm 1.4cm$. The following parameter set-

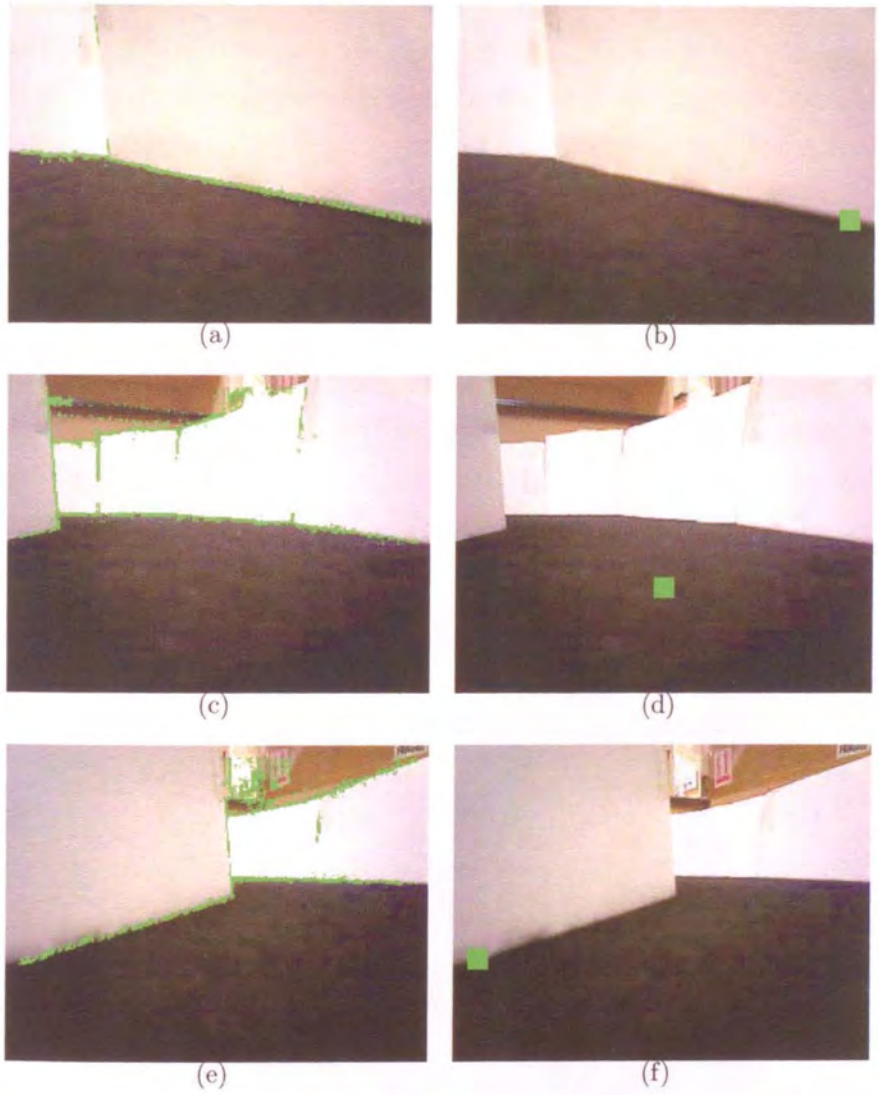


Figure 4.16: Example frames showing real-time corridor tracking. Left column: original RGB image with pheromone superimposed (bright green pixels); right column: original RGB image with pheromone COG superimposed (bright green square). (a,b) Left turn; (c,d) forward; (e,f) right turn.

tings were used: $N = 5000$, $\rho = 0.08$, $T = 20$, $\alpha = 1.0$, $\beta = 5.0$, $tabu_{max} = 8$ and $Z = 30$.

Figure 4.17 shows the robots trajectory (in the 2D ground plane) with respect to the corridor walls in the environment. From visual inspection we see that the robot successfully traversed the corridor path, following the curvature. From the ant-algorithm tracking the edges of the corridor meeting the floor, the robot was able to continuously adjust its direction in order to traverse a route along a quasi-centreline of the corridor, thus tracking the desired route and avoiding collisions with the corridor walls. The slight fluctuations in the recorded route are due to a number of reasons. Firstly, the algorithm is using only a simple rule-set to command the robots motors, with no gradual change in wheel speed between the *left*, *right* and *forwards* motions. Second, the robot used suffers greatly from wheel-slip in the environment used for the experiment. Finally, there are noise imperfections in the system used for capturing the robots pose (details given in Appendix B).

The heuristic information used to guide the swarm agents in this chapter has been based on local image gradient, allowing the swarm to self-organise in response to the edge patterns within a greyscale digital image environment. For additional experiments using RGB heuristic information, see Appendix C.

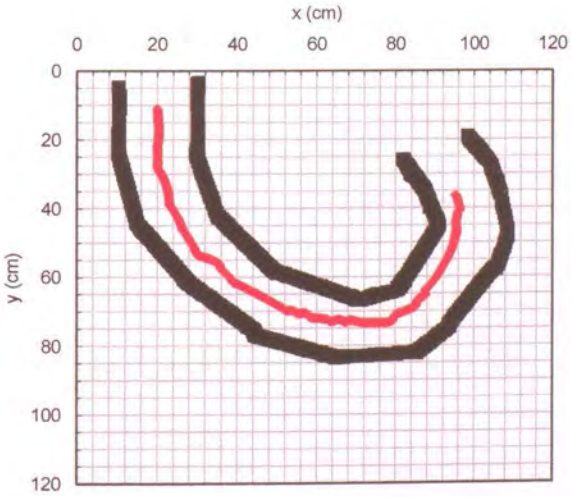


Figure 4.17: A plot of the robot trajectory for the robot-corridor following experiment. A red line shows the robot trajectory, with a black line representing the corridor walls.

4.6 Conclusions and Discussion

This chapter has extended the work of the previous chapter by focusing on adaptation in the self-organisation process. Using the ant algorithm proposed in chapter 3 for the core swarm control, the swarms abilities have been augmented by allowing individual ant agents to control their own threshold parameter settings. The way in which the agents adapt their parameters has been purposefully limited to simple rule-based logic, to maintain a level of simplicity akin to the local decision making processes observed in real-world swarms. Experimentation for parameter adaptation has focused on adapting the threshold value T, which was shown in the previous chapter to be of particular importance in determining the performance of the algorithm for the purpose of image feature extraction, and which is likened to the internal behavioural thresholds believed to be present in social insects in nature.

Despite these simple rules, it has been shown that while the swarm may not collectively converge on the optimum parameter setting for a given image, by allowing individuals to continually adjust their threshold T values, the average T value of the swarm does converge as the swarm converges, with the resultant evolved pheromone map showing the edge features as desired, over a range of different images with varying threshold values. From an image segmentation application point of view, eliminating the requirement for setting a threshold and still producing good quality results is a particular advantage. Setting an appropriate threshold requires prior knowledge of the image or images to be processed, and an often lengthly process is required to determine an appropriate threshold for each image set, or even each individual image, to be processed. The use of the ant-algorithm swarm approach with the proposed adaptive threshold provides a novel image and video processing solution which exploits the distributed nature of the swarm self-organising approach, to deliver distributed adaptive feature extraction and tracking in digital image media.

From a swarm intelligence theoretical standpoint, this reinforces the concept of nature inspired distributed simple decision making as a powerful optimisation and problem solving tool in the digital, computerised world. Giving individual swarm agents the facility to dynamically adjust their threshold parameter in response to the changing environment increases the self-organisation capability

of the swarm by increasing adaptability. This allows the swarm to successfully self-organise into patterns representing the image feature structure over a wider range of image characteristics (for example ambient lighting, entropy, noise).

The self-organising pattern formation behaviour of the swarm has been further examined by considering dynamic imagery. In such a case, there is an additional challenge for the swarm; once a particular pattern formation has been reached, the swarm must be able to 'forget' this pattern formation and adapt, through continuous self-organisation, to a new pattern, or indeed a continuously changing pattern. Through a number of visual examples and case study experiments, the highly adaptive nature of the proposed swarm ant algorithm for tracking features within digital imagery has been shown, and moreover, it has been demonstrated how rapid adaptive continuous self-organisation can be achieved to facilitate quasi-real-time pattern adaptation and tracking. The state-of-the-art in using ant-algorithms for image and video processing has been advanced by the implementation of the proposed ant-algorithm for dynamic image feature tracking, and the quantitative analysis carried out in this chapter has provided statistical measures of performance both with respect to the image processing application problem, as well as the self-organisation performance.

The use of parameter adaptation with dynamic imagery has shown to yield promising results, with the swarm changing the threshold value in accordance with changing images, resulting in increased accuracy over an otherwise static threshold value. Real-time dynamic imagery means that the environment is continuously changing, and the swarm has to cope not only with the changing patterns of image features, but also the changing characteristics of the image landscape, affecting the required threshold value (such as lighting levels, frequency of features). These real-time challenges are more similar to what is found in nature, where swarms have to deal with a constantly changing world. Methods of self-organisation, distributed decision making and adaptation allow swarms in nature to cope with these dynamic challenges in relatively simple, yet sophisticated ways. Understanding the ways in which such behaviours are achieved, and porting these behaviours to the world of computational intelligence, is a challenge in itself.

This chapter has attempted to examine a number of adaptive self-organisational methods, in terms of ant-algorithms for image processing and computer vision

applications. A level of success has been achieved in harnessing the power of distributed adaptation for the purpose of implementing an adaptive threshold technique for the ant-algorithm presented in the previous chapter. Increasing the dynamic nature of the environment has introduced new challenges, prompting further developments to the implementation of the ant-algorithm approach, and providing additional discussion of the inspiration and knowledge we can gain from swarms in nature, as well as the analogy between natural and computarised swarms.

The next chapter continues to study nature inspired self-organisation methods applied to pattern formation, adaptation and manipulation. Focus is shifted however to an alternative approach to the ant-algorithm, and methods of *stigmergy* are employed in a different way. The research also moves away from the digital, discretised world of image data, to consider instead real world continuous environments, where the swarm agents are embodied in mobile robots.

Chapter 5

Self-Organisation and Pattern Formation with Autonomous Robot Swarms

5.1 Introduction

In this chapter the study setting changes from software agents in digital image environments to 'hardware agents' in real-world environments. For the purpose of this work 'hardware agents' are mobile mechanical entities such as robots, rovers, Unmanned Aerial Vehicles (UAVs) and Unmanned Ground Vehicles (UGVs). Due to experimentation limitations (such as available hardware, testing facilities and safety issues) the 'real-world' is approximated to varying degrees for much of the work to follow, by use of various simulation methods and packages (including embedded simulation on real-robots). Since the focus of research here is on the theory and development of the swarm behaviour mechanics, and not the detailed engineering problems associated with the hardware, these approximation methods are deemed acceptable to conduct meaningful research into self-organisation and pattern formation with swarms of hardware agents, for a number of behaviours and real-world scenarios. In fact, the transition from software agents, to simple 2D simulation, to physics and sensor based simulation, to embedded simulation on real robots; provides an interesting and informative point of discussion on how far we can utilise the inspiration of nature in designing

and implementing artificial swarms.

When considering swarm agents as robots, there is immediately a number of limitations imposed on what we can expect our agents to do (and, equally important, what the robot agents actually do irrespective of what they are supposed to do). The robot's actuators will determine the ways in which the agent can move around in, and interact with, the environment. The available sensors will determine how much, and what information the agent can get from the environment. There will be other limiting factors such as communications and power (both computational and energy). These factors mean that certain traits observed in swarms in nature simply cannot be replicated in hardware (at least not with presently available technology). Consequently, related algorithms must be designed with such considerations in mind. This does not however mean that the core components of the algorithms have to be designed with specific hardware configurations in mind, in fact this research tries to avoid such restrictive methodology.

The algorithms and methods presented in this chapter share similar biological inspiration to the ant-algorithm variations presented in previous chapters. Those presented in this chapter are tailored to self-organisation and pattern formation with a swarm of robotic entities, moving in a real-world environment, for real-world applications and scenarios.

The previous chapter studied pattern self-organisation and pattern formation of a swarm in response to a given environment landscape, where the swarm formed patterns based on features in the environment. This chapter looks at self-organisation and pattern formation based not on environment characteristics, but instead based on agent-agent spatial relationships. Furthermore, this chapter studies utilising self-organisation methods to create dynamic group behaviours and applying such behaviours to a number of scenarios.

As previously described, a swarm intelligence approach implies that at some stage in the design process, there has been inspiration taken from observations of systems in nature. This work is particularly focused on the use of indirect communication, inspired by *stigmergy* [18], as observed in ant colonies in the wild. This type of indirect communication has been used extensively in computational problems for a multitude of applications [115], with particular success in combi-

natorial optimisation problems [10] after being developed into algorithmic form in [11]. Stigmergy is used to drive the emergent behaviour and self-organisation that produces the desired collective behaviour from the swarm of robots. The way in which stigmergy is implemented in this chapter and the next is quite different from the ant algorithm approach of the previous two chapters, and this is largely due to required hardware considerations and feasibility when dealing with real-world hardware agents environments.

The basic idea is that instead of communicating information directly between robots, which would create inter-robot dependency, the system is designed such that each robot can exploit local sensory information. For example the locations of neighbouring robots, together with other internal factors, are used to calculate the robot's new trajectory, in a reactive cooperative manner. This work does assume that on-board sensors are available to facilitate indirect observatory communication between robots, such as calculating the range and bearing to neighbouring robots, and also detecting obstacles or local environment structure. There exist various types of sensor that could provide such information, for example infra-red sensors (often used in small-scale laboratory experiments), laser range finders, sonar and omnidirectional vision. The research of this chapter (as well as the next) focuses on the swarm control mechanisms, and it is assumed that this local information is available. Although the use of specific sensors is beyond the scope of the work of this thesis, some consideration is given to the effects that sensor noise would have on the proposed swarm control mechanisms.

The internal factors depend on the specific task being performed, and again inspiration is taken from nature to facilitate these tasks in simple yet effective ways (as detailed in section 5.5.3).

5.2 Related Work

Research in the area of multi-robot systems and swarm robotics remains very active, with no single approach taking prominence as the accepted method for achieving autonomy to a standard deemed acceptable for operational use. There are many different methods being developed within the scientific community to achieve autonomous multi-robot control, for a wide variety of applications. A

brief overview is given here of a selection of related methods specific to multi-robot pattern formation, control and collective movement, and in particular swarm robotics methods.

Particle Swarm Optimisation (PSO) [8] has been used to tackle a variety of problems in multi-robot systems. PSO is a population based stochastic optimisation method inspired by observations of collective movement and social behaviour of biological entities in the wild, such as bird flocking or fish schooling. PSO has been used as a tool to optimise a sequence of controls in order to achieve a desired formation within the context of multi-robot coordination [78][79], and used to model the characteristics of a multi-robot search process [80]. A PSO algorithm is physically embedded into a small number of robots in [81], where each robot represents a particle in the PSO algorithm, and the robots search for a target within a given search environment.

The well established nature inspired technique of flocking [4] has been used in many variations to achieve multi-robot coordination and control (for example [69][70][71]), and often uses virtual forces to achieve the desired behaviours. Flocking methods provide mechanisms for collective swarm level movement, which can provide the ability of the swarm to maintain a particular formation type while manoeuvring through a given environment (for example [116]). One of the potential limitations of the typical flocking approach is that it requires inter-robot communication, for example to share heading information [70] or velocity [71] between the robots.

From the swarm intelligence paradigm, the concept of stigmergy [18] has been used, for example in [60][61], to achieve multi-robot coordination and control by using artificial pheromones to guide the robots. In [62][63] the authors have developed a set of ant-like robots that can deposit and sense ethanol for use as artificial pheromones to create self-organisational cooperative behaviours. Similarly, the well established potential fields method [66] is still widely used for robot control/coordination. Both of these methods involve sensing/computing virtual force vector-fields which guide the robots movement. Another similar method uses physics based laws to compute local artificial forces [67][68] to guide the robots movements. The latter method does not require the computation of a global vector field of artificial forces, rather each robot computes the forces it experiences locally, thus offering a more distributed approach.

A more engineering based method is reported in [72], which uses a kinematics control approach and merges a number of elementary behaviours into one final behaviour to facilitate the entrapment and escorting of a target by multiple robots.

The latter mentioned approach, along with the methods using physics based laws to compute virtual forces, tend more towards a deterministic nature, as opposed to the stochastic nature of the ant-algorithm approaches of [60][61]. The stochastic nature of many swarm intelligence methods makes them particularly good at solving unpredictable and dynamic problems (in [73] stochasticity is introduced in a Lyapunov-based flocking controller and shown to improve performance). This can also however make it difficult to fully predict the behaviour of the solution itself [74], which, in particular safety critical applications, might be an undesirable feature.

Another important potential property that can be maintained using the physics based swarm methods is that of robot agent anonymity. The robot swarm is homogeneous, and each individual robot has no identity, which is in part due to not requiring direct inter-robot communication. An important consequence of this anonymity is that it reduces inter-robot dependency, thus increasing overall robustness of the swarm to robot failure.

The work in this chapter aims to further investigate the swarm intelligence approach to robotics, with particular consideration to the observations in the wild of complex behaviour resulting from relatively simple entities acting on stimulus in their local environment [18]. This work follows a different approach to stigmergy than the previous two chapters, which used artificial pheromones for indirect communication. Indirect communication is used here in the form of relative positional sensing, whereby an individual robotic agent can sense the relative range and bearing to local neighbouring robots. This observatory *indirect* communication (as opposed to individual agents *directly* communicating their whereabouts to one another) forms the basis on which the robots perform pattern self-organisation and formation control. The use of physics based laws for virtual force manipulation to induce self-organised behaviour is employed (inspired by the work of [67][68], and the analogy with particle physics swarms [9]), as it is believed that there is great potential to extend and improve the use of this approach for swarm robotics formation control and collective movement,

exploiting the fully distributed potential in this method.

This approach to distributed autonomous robot coordination and control is examined through a number of case-study experiments, and research is furthered in this area, utilising such methods and extending the basic formation control to achieve a number of elementary behaviours, which are then merged together to create a multi-behaviour autonomous system.

5.3 Physics-based Autonomous Control Laws

A simple yet sophisticated approach to robotic swarm self-organisation and pattern formation is proposed to facilitate cohesive, collective movement. The methods presented in this chapter are based on the physics potential fields approach [67]. Although there has been successful implementations of ant-algorithm based methods in swarm robotics (for example [60][61][62][63]), it was decided to focus on a different method to what was the focus of the previous two chapters. An analogy remains to be drawn however with the potential fields methods of focus in this chapter, and the ant algorithm methods from the previous chapters, and this is discussed in more detail later. The primary reasons for choosing physics-based methods for the development and study of new swarm robotics behaviours are the following:

1. The strong theoretical background of the related physics laws can lead to greater predictability and better means of analysis, while still offering self-organisation properties.
2. Ease of implementation with simple robots (reducing issues with available hardware.)
3. No direct inter-robot communication is required, robot anonymity maintained.

Point (1.) raises an interesting question on the differences between deterministic and non-deterministic methods. In [67] the authors provide a mathematical analysis from a physics perspective, providing useful knowledge for optimising parameter settings and predicting behaviour. Conversely, in [73] the authors

show an increase in performance of their flocking robot controller by introducing stochasticy. Although deterministic methods can still be unpredictable due to, for example, sensor and actuator noise, using control methods that are explicitly non-deterministic could prove particularly problematic in terms of safety critical issues, for example if the robots were to be deployed in a human populated area. Point (2.) relates to the previously discussed issues regarding hardware limitations and constraints. Firstly, with the hardware available for this project it would not be possible to carry out any experiments on real robots if methods were employed using tangible pheromones. Secondly, although the study of ant-algorithms used for swarm robotics is both interesting and worthwhile, a conscious shift was made of the focus of the study to focus on methods more tractable for implementation in real-world scenarios, and to study the self-organising properties of a method that is quite different, yet holds similar grounding, to the stigmergic methods such as ant-algorithms. Point (3.) highlights another conscious decision, to tackle a challenging open problem in swarm robotics systems.

As previously stated, in order to achieve the desired result of configuring the swarm into a regular lattice formation inspiration is taken from physics based laws, and firstly, in this case study the use specifically of Newton's law of gravitation. Newton's law of gravitation is relatively simple in terms of its structure, and consequently, is relatively simple to implement. The use of this law for swarm pattern formation has been studied previously from a physics perspective in [67], with relative success in particle-based simulations and limited testing on robotic platforms for simple swarm behaviours. In this chapter the analysis is carried out with the analogy to nature in mind, and work is advanced in developing the concept into a multi-behavioural task-driven system.

Considering agents as particles of matter, we can consider a scenario where agents exert virtual forces on one another such that the force, F, experienced by agent i, due to agent j, is given by

$$F = \frac{Gm_i m_j}{r^2} \tag{5.1}$$

where m_i and m_j denote the mass of agent i and j respectively, r is the distance between the two agents and G is a constant (equivalent to the gravitational constant in Newton's law of gravitation). For the purpose of this study it is

assumed that $m_i = m_j = 1$.

Under this framework there is no need to calculate a global force map, rather each agent calculates the local forces it experiences thus allowing for a distributed approach. One assumption made here is that each robotic agent is capable of calculating the range and bearing of neighbouring agents. A discrete time approximation is used such that at each time-step each agent within the system calculates a total force vector experienced due to neighbouring agents. Each agent then calculates a new velocity vector from their total force vector as follows. Using a discrete-time approximation with $\Delta t = 1$ and a friction coefficient, $\sigma = 0.8$, the x and y components of the velocity at time-step n are given by

$$\nu_x(n) = (1 - \sigma).\nu_x(n-1) + F.cos(\theta) \tag{5.2}$$

$$\nu_y(n) = (1 - \sigma).\nu_y(n-1) + F.sin(\theta) \tag{5.3}$$

where θ is the bearing of the total force vector F.

The friction coefficient is included to add stability to the system, by limiting the momentum build-up of the agents. The value of $\sigma = 0.8$ is chosen, by trial and error, to give the best performance to this effect. An additional constraint is applied to limit the allowed overall force, such that $F = \epsilon$ if $F > \epsilon$ and $F = -\epsilon$ if $F < -\epsilon$, where ϵ is the maximum allowed force.

In order for the agents to maneuver simultaneously into a regular lattice-like pattern the following conditions are imposed on the force law calculations:

$$\text{for} \quad r > R, \quad F \rightarrow -ve \tag{5.4}$$

$$\text{for} \quad r \leq R, \quad F \rightarrow +ve \tag{5.5}$$

This forces agents to maneuver into formations where they are all separated by a specified distance R, thus forming a regular lattice formation of agents.

5.3.1 Point-mass Simulations

For this first study in the realm of swarm robotics a simple simulation model is maintained, treating the robotic agents as point-mass particles, moving in a fixed horizontal plane. These simplifications make the movements of the robotic agents similar to the ant-agents from the previous chapters. Such simplifications are deemed acceptable since the focus of this work is not concerned with high fidelity models, rather the focus is on designing and analysing the underlying swarm behaviours. That said, the importance of evaluating the feasibility of such behaviours being implemented on actual hardware must be acknowledged, and thus additional work in the forthcoming sections does go some way to exploring performance and implementation issues in more realistic simulations of real-world systems, including testing with real laboratory robots.

The point mass simulation is carried out using the *Player/Stage* project [117]. Simulation is performed in 2D of the spatial distribution (with continuous (x, y) coordinations as opposed to the discretised pixel coordinates from the previous chapters) of identical robot-agents which have the ability to control their own velocity and to sense the range and bearing of neighbouring agents. These simulations do not account for collisions, and the robots can move freely across one another.

5.3.2 Performance Metrics

A number of different metrics are used in order to assess the performance of the swarm in terms of cohesion, specifically looking at the spatial distribution of the swarm over time. One of the key goals is for the swarm to be able to construct and maintain a lattice formation. Maximum cohesion would correspond to a repeating, regular lattice formation within equal spacing between neighbouring robots. One of the main performance metrics used is the *average minimum agents separation range*,

$$P_{\overline{min}} = \frac{\sum_{n=1}^{N} r_{nmin}}{N} \tag{5.6}$$

where r_{nmin} is the minimum distance to a neighbouring robot from agent n.

Similarly, the *average mean agent separation range* is given by,

$$P_{\overline{ave}} = \frac{\sum_{n=1}^{N} r_{nave}}{N} \tag{5.7}$$

where r_{nave} is the average distance to a neighbouring robot from agent n.

Another metric used is the *Variance of mean agent separation range*, given by,

$$P_{Var(ave)} = Var\left(\sum_{n=1}^{N} r_{nave}\right) \tag{5.8}$$

The $P_{Var(ave)}$ metric is designed to show how stable the swarm is through self-organisation and adaptation.

In the approach used, a user-set desired inter-robot separation distance R is defined, and thus performance can also be measured in terms of the error between the desired R and that actually achieved by the swarm. There are also several other performance metrics used which are specific to certain applications, problems and swarm behaviours which are used throughout this and the next chapter. All additional performance metrics are defined in the relevant sections.

5.3.3 Examples

Figure 5.1 shows snapshots of the swarm running the above Newtonian formation control law using the Player/Stage point mass simulator. The images show the self-organised formations after 5000 time-steps, for increasing values of N, with $R = 25$, $\sigma = 0.8$, $G = 100$ and $\epsilon = 1.0$. The agents start in a pseudo-random cluster at the centre of the image before executing the control law and expanding outwards. We note that with increasing N, although the swarm maintains the ability to successfully expand, the formation becomes less regular. With increasing N we observe that quasi-equal separation distance is achieved around the edges of the swarm formation, however this becomes less cohesive towards the centre. The observation that for $N = 25$ we do not achieve an ideal lattice formation suggests that this problem is not an issue of scaling with increasing N. In fact, with the larger N examples, if the experiment was left to run for longer than 5000 time-steps, we would observe larger regions around the edges of the

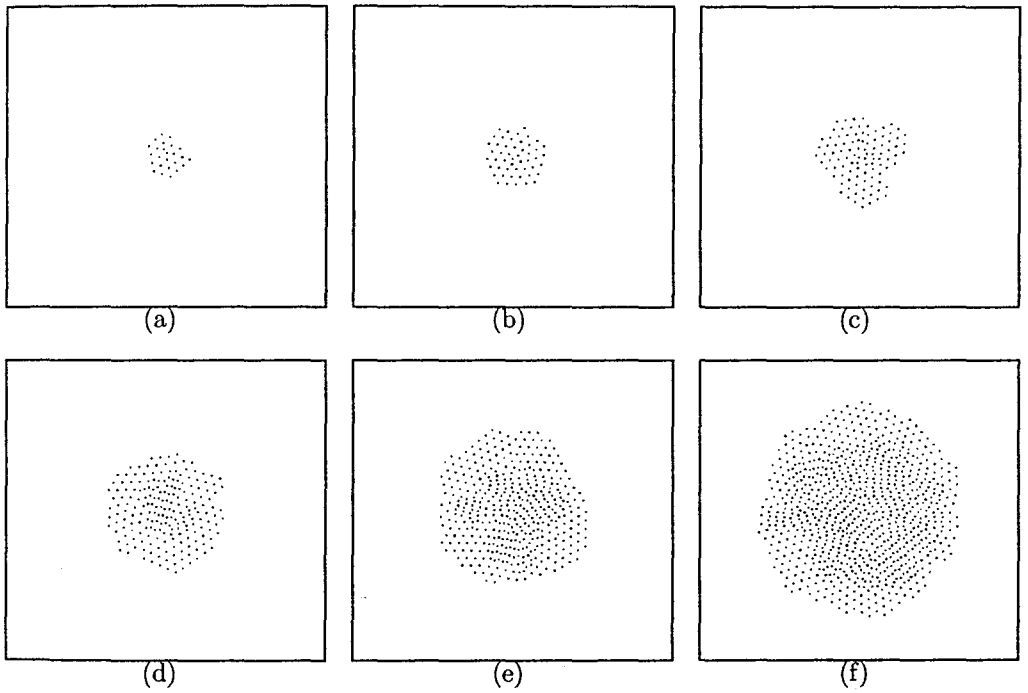


Figure 5.1: Point mass simulation of the Newtonian force law for varying N. (a) $N = 25$; (b) $N = 50$; (c) $N = 100$; (d) $N = 200$; (e) $N = 400$; (f) $N = 800$.

formation with quasi-equal separation distances, however full convergence is not guaranteed.

Figure 5.2 shows the corresponding average minimum agent separation range for the experiments of Figure 5.1. With $R = 25$, for a lattice formation with quasi-equal separation distance, we would expect that the minimum inter-robot separation distance to converge to near 25. For the experiments with $N = 25, 50$, we indeed observe this to be the case. For the other experiments, where an ideal lattice formation is not achieved, the error in average minimum separation distance increases with increasing N.

Figure 5.3 shows two examples formations with 7 agents, again using the *Player/Stage* simulation. The example on the left is run with $\epsilon = 1.0$ and the example on the right, exhibiting the clustering problem, is run with $\epsilon = 0.1$. These examples were run for 200 time-steps, with $R = 25$, $\sigma = 0.8$, $G = 275$. On the left is the 'desired' formation, representing a repeating lattice of equilateral triangles (with quasi-equal inter-agent separation distance). On the right is an example of

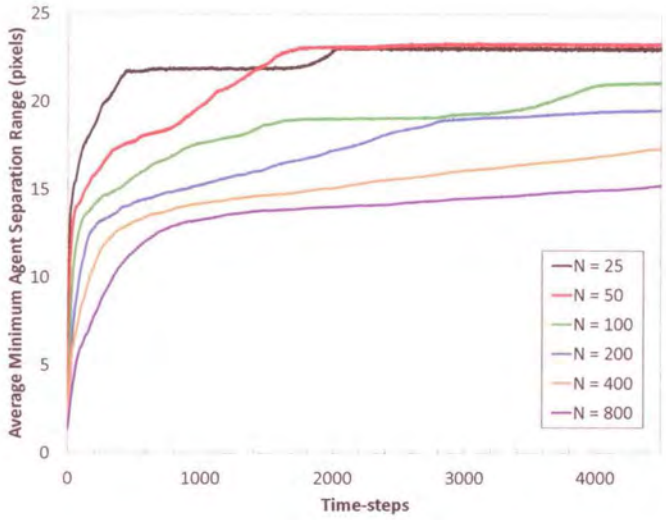


Figure 5.2: A plot showing $P_{\overline{min}}$, the average minimum separation distance between agents vs. time, for varying N.

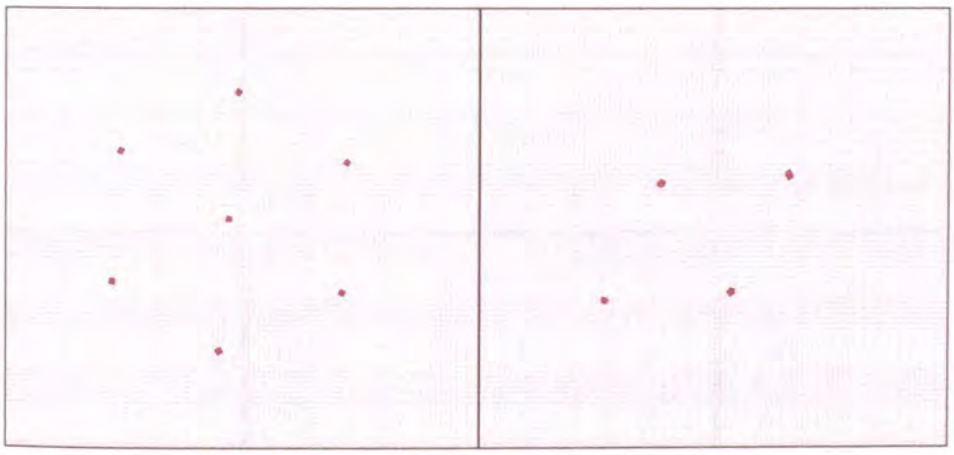


Figure 5.3: An example ideal formation (left), with $\epsilon = 1.0$, and an example of the clustering problem (right), with $\epsilon = 0.1$, with 7 robots, in the *Player/Stage* 2D simulation environment.

what is known as the 'clustering' problem [67][96], where we observe that three of the nodes in the formation have 2 robots 'clustered' together. This 'clustering' phenomenon can also be seen in Figure 5.1(a), resulting in a non-uniform lattice.

The clustering problem is where groups of robots remain clustered together, with $r < R$. This phenomenon is known to be caused by local minima of the inter-robot forces, for example when a robot is surrounded by a number of robots such

that the resultant force is not sufficient for it to escape, and that robot's influence on the surrounding robots is not sufficient to cause them to move out of the way. It is thus not surprising that for such a low maximum allowed force, ϵ, that the clustering problem occurs.

Figure 5.4 gives further examples of the clustering problem, with $N = 500$ agents, and for (a) $G = 100$ and (b) $G = 200$. With $G = 100$ the swarm has successfully expanded, however the resultant formation is not quasi-regular except for around the edges, again due to the clustering problem. For $G = 200$ we observe clustering more similar to that of Figure 5.3, right, where there are multiple agents clustered around many of the nodes in the formation, such that the swarm has not fully expanded. Counting each cluster as a node, the swarm has managed to self-organise into a quasi-regular lattice formation however.

It is clear that the clustering phenomenon has multiple characteristics and cannot be solved by simple parameter adjustment, such as increasing ϵ. This is a known problem when using virtual forces for self-organisation, and one which is tackled in the next chapter. Despite the clustering problem, this approach to self-organisation remains a powerful tool for autonomous multi-robot pattern formation and manipulation, which is investigated in the following sections with a number of case-study experiments.

Figures 5.5, 5.6 and 5.7 give further examples, this time varying the G parameter, with $R = 25$, $\sigma = 0.8$ and $\epsilon = 1.0$. In Figure 5.5 we see how different formations can result from different values of G. Here we have two examples of a final formation of a repeating equilateral triangle lattice, but with different topologies. We see little variation in average agents average range for different values of G (Figure 5.6), and similar groupings of the plots for the variance of average agents average range, with $G = 200, 235$ converging to a variance approximately $2cm$ above the others.

In the next section the parameters are explored further, in the context of a case-study coverage problem scenario.

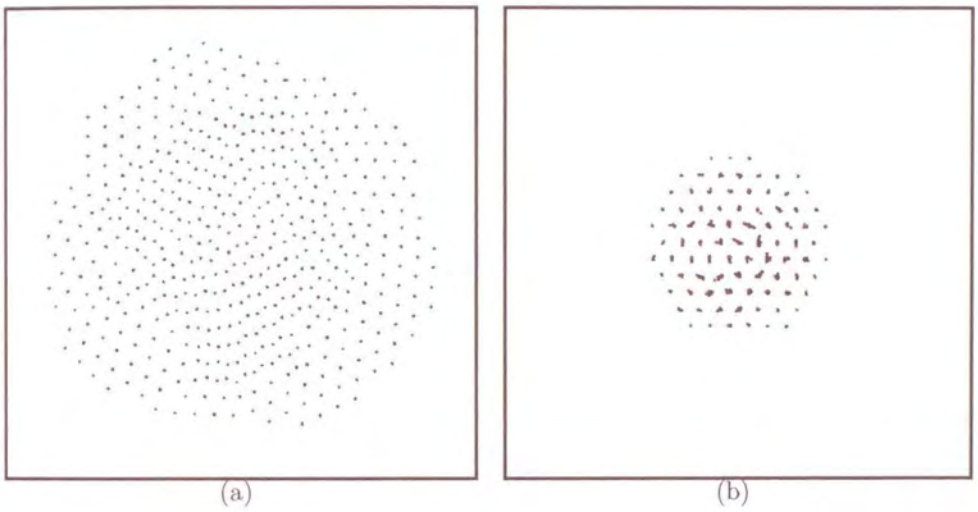


Figure 5.4: Point mass simulation of the Newtonian force law with $N = 500$ agents, showing the affect of G on the clustering phenomenon. (a) $G = 100$; (b) $G = 200$.

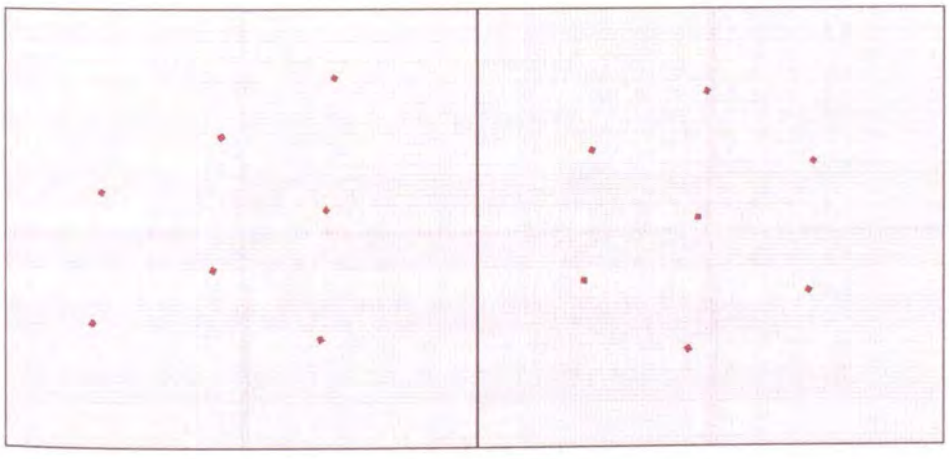


Figure 5.5: An example formation with $G = 200$ (left), and with $G = 275$ (right), with 7 robots, in the *Player/Stage* 2D simulation environment.

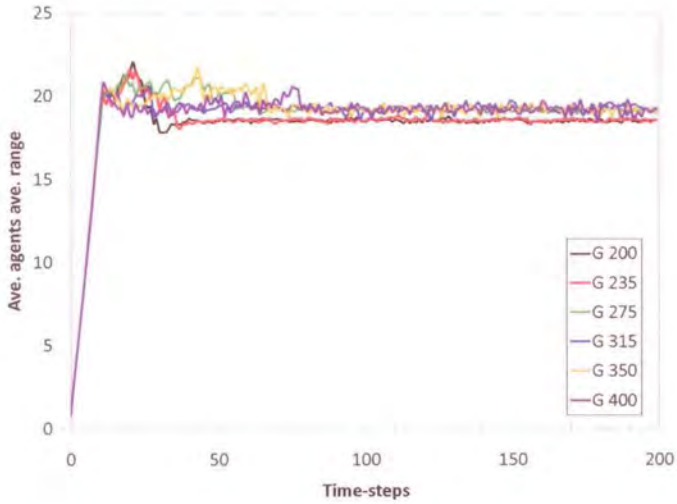


Figure 5.6: A plot showing $P_{\overline{ave}}$, the mean average separation distance between agents for a formation experiment with 7 agents, vs. time, for varying G.

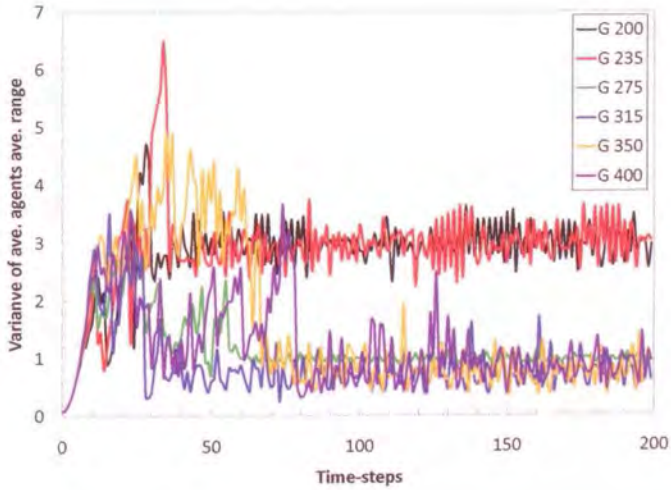


Figure 5.7: A plot showing $P_{Var(ave)}$, the variance of average separation distance between agents for a formation experiment with 7 agents, vs. time, for varying G.

5.4 A Case Study in the UAV Coverage Problem

When considering applications in surveillance there are obvious benefits of having more than one mobile robot sensor, for example if there is a large area to cover, or a complex area with many occlusions that would require many different viewpoints, or if there were multiple targets to track.

The application of swarm robotics in security and defence ranges across platforms from sub-aqua to ground based, aerial and even space-borne, ranging in size from micro-bots to satellites, and numbers from tens to hundreds (and in theory at least, into the thousands). Such particulars would of course depend on the specifics of the application, for example micro-bots may be used for covert building security, wheeled robots/UGVs for warehouses and public places, aerial robots/UAVs for regional/large outdoor areas, and satellite or space formations for planetary coverage.

With the security threats of the present day, both in everyday civilian life as well as on the battlefield, increasing surveillance capabilities is an important issue and a very active area of research. Although UAVs have been utilised in the military for some time now, their use in the public domain remains relatively sparse [118]. The deployment of UAVs in an urban setting would necessitate a small platform requiring minimal take-off and landing space.

Small scale Vertical Take-Off and Landing (VTOL) UAVs are becoming more readily available however and could prove extremely useful for rapid response situations within the emergency services [119]. Given the portability of currently available platforms they could quite feasibly be carried in mobile police and fire units, ready to be rapidly deployed in situations where airborne surveillance is necessary. Clear advantages here are speed of deployment without having to wait for manned air support to arrive, as well as the potential ability to reach areas inaccessible or too dangerous for manned air support.

Such devices are readily available to be used via remote user control (for example [120][121]), but this would require skilled operators as well as mobile control stations. This case study looks at utilising a physics-based swarm approach to the development of autonomous coordination and control of swarms of such de-

vices, in theory enabling rapid deployment of an airborne sensor grid providing multiple Field Of View (FOV) ground coverage. Programmable small-scale airborne robotic agents are beginning to become available (for example see [122]), so effective methods to fully exploit such technology is of particular interest.

Typical scenarios where such technology could provide critical information would be spontaneous public order incidents, tracking suspects over large areas and monitoring of critical incident sites. Focus is thus given to a coverage problem via a sensor network facilitated by a swarm of mobile VTOL UAVs. Since the primary focus of this thesis is with the development of methods for self-organising pattern formation, the goal here is to create a system for autonomous self-organisation of a swarm of robotic entities into a formation that provides optimum distributed coverage of a given area. Again a high fidelity model is beyond the scope of research of this thesis, and the focus remains on the swarm-level control.

The swarm intelligence approach lends itself well to multi-robot systems, with self-organisation and cooperative behaviour central to this paradigm. Despite successful methods based on swarm intelligence (see [123][56] for detailed reviews), there is a lack of mathematical modelling and validation of these approaches, due largely to the challenges involved in modelling such dynamic behaviours [74]. By developing control algorithms with an underpinning framework based on physics based laws there is scope to gain a better understanding of the self-organisation mechanics being employing. This chapter focuses on this underpinning framework, with an overall aim to develop a set of hybrid control laws building in methods more akin to the swarm intelligence approach, specifically the use of indirect communication based on stigmergy, in the form of physics-based virtual forces.

The coverage problem in the context of this work involves a swarm of identical robotic agents working cooperatively to form a sensor lattice that provides optimum coverage of a given ground area. For the purpose of this study the robots are assumed to be VTOL UAV platforms, all moving at the same altitude in a fixed plane parallel to the ground (simplified for point mass simulations, using the Player/Stage project, Figure 5.8, left).

Each robot is equipped with a downwards facing sensor which is approximated to have a ground level circular sensing area centred on the robots projected

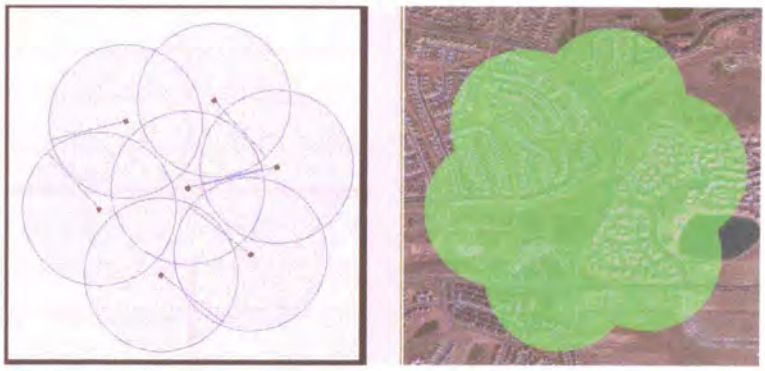


Figure 5.8: Left: Player-stage 2D simulation, with red squares representing agents, and blue circles their corresponding sensor areas. Right: Projected coverage onto a ground level ROI.

ground coordinates (at *nadir*), with a fixed radius r_s. In simulation terms the area of coverage is projected onto a global ground map as circles centred on the robots (x, y) coordinates (Figure 5.8, right) and the percent coverage is calculated against the set Region Of Interest (ROI).

The aim for the swarm is to move from a tight cluster (as if all the agents have been deployed from the same location) into a stable formation configuration which maximises the percentage coverage of a specific area by all agents collectively. This requires a balance between the correct separation distance R between all agents within the swarm and the total area that is desired to be covered. The initial positions of the agents are taken as random coordinates within a small grid space centred over the middle of the ROI to be covered (to represent a common deployment location). The initial velocities of the agents are set to zero, however the first time-step in the algorithm will calculate their velocities based on their relevant initial positions to one another (see section 5.3).

The self-organising process in this case is the autonomous self-configuration of the robots into a regular spatial formation. Unlike the ant-algorithm from the previous two chapters, in this chapter the agents will be self-organising primarily in response to agent-agent interactions, and not agent-environment interactions. Despite the different methods used here, the focus is still on the common function of pattern formation via self-organisation. This case study was chosen as a test-bed analysis study, to provide context and background to the well known benchmark coverage problem, in moving from digital swarms in a digital envi-

ronment, to hardware swarms in a real world environment.

5.4.1 Pattern Formation and Coverage

Within the proposed framework (as defined in section 5.3), the swarm naturally converges to a regular lattice formation of repeating equilateral triangles of separation R (Figure 5.9). For a circular sensor area, as chosen for experimental analysis in this case study, we achieve zero sensor overlap, while maintaining minimum gaps in the sensor lattice, by setting $R = R_t = 2r_s$. By eliminating the sensor overlap and thus maximising the overall coverage achieved from all the sensors collectively, we do however introduce gaps in the sensor lattice at the centre point of each equilateral triangle formed by the agents (Figure 5.9), where no coverage is achieved. There is therefore a trade-off between minimising sensor redundancy, and the introduction of gaps in the sensor lattice. In the initial 2D set-up, when $R = R_t$ the circular sensor areas are tangential to one another and the area of the gap between the circles can be calculated as the area of the equilateral triangle minus half of the area of the circular sensing area:

$$A_{gap} = \frac{R_t^2\sqrt{3}}{4} - \frac{\pi r_s^2}{2} \tag{5.9}$$

In this study a sensing area of $r_s = 21m$ is used (this number does not represent any specific characteristics), and so for the situation where three circular sensor areas are tangential to each other, the area of the gap in between, given by Equation 5.9, is $A_{gap} = 88.9m^2$. This is equivalent to 2.1% of the total area of the three circular sensing areas surrounding.

If we wish to reduce the size of the gap to zero, creating uniform coverage within the lattice at the expense of sensor overlap, we need to reduce the agents separation R to the point where the perimeters of the three adjacent sensor circles meet at a common point at the centre of the equilateral triangle that they form. This separation distance is given by:

$$R_o = 2r_s cos30 \tag{5.10}$$

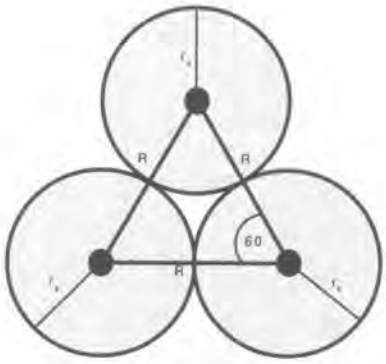


Figure 5.9: Equilateral triangle lattice formation for agents (solid black dots) separated by distance R, with sensor range r_s.

For the experimental set-up used in this case study we obtain the value of $R_o = 36.4m$.

If minimising the sensor redundancy is a priority however, the resultant hexagonal lattice formed by repeating equilateral triangle formations with the sensor circles tangential to one another, has been proved (*Carl Friedrich Gauss*) to yield the highest density for a circular packing arrangement in 2D Euclidean space, given by $\eta_h = \pi/\sqrt{12} \approx 0.9069$. Having $R = R_t$ will not necessarily result in the maximum achievable coverage however, as this would depend on the size of the ROI, with respect to r_s.

The following experimentation investigates the effects of various parameters within the control law framework on the behaviour of the swarm with respect to the coverage problem described above, and specifically investigates how varying the separation distance R, along with the gravitational constant G, affects the overall coverage achieved for given ROIs.

Increasing the desired separation between agents decreases sensor overlap and thus increases the overall area covered up to the point at which the area of sensor overlap reaches zero. It is therefore desirable to find the correct separation distance R to maximise overall coverage. If R is too small this may result in too much sensor overlap, however if R is too large this may result in agents covering areas outside of the ROI.

In addition to finding the best R value, also of interest is the value of the constant G (Equation 5.1). The value of G affects the rate at which the swarm converges to a stable configuration and can determine the spatial characteristics

of the particular final configuration (as shown in Figure 5.5). Although there are other parameters to consider, focus is given to the values for R and G, as initial experimentation has shown these two parameters to be of particular influence on the overall behaviour of the swarm.

Figure 5.10 shows the percentage coverage achieved, after 500 time-steps, for 64 different combination of R and G, for a swarm of 7 agents starting in a tight pseudo random cluster in the middle of 5 different sized square ROIs (as if the agents had been deployed from the same place). As expected, overall we see an increase in percent coverage with increasing separation distance R up until a point. For ROI sizes $50m^2$ and $100m^2$ we then see a drop in overall percentage coverage (for $R > 40$), as agents begin to move outside the ROI. For ROI size $50m^2$, we observe a decrease in percentage coverage with increasing G for $R = 6, 10$. This is due to the swarms inability to converge to a stable configuration for small R and large G. In such a situation the agents continue moving within a small cluster, with the cluster as a whole drifting slowly. The decrease observed for ROI size $50m^2$ in Figure 5.10 is due to the cluster drifting

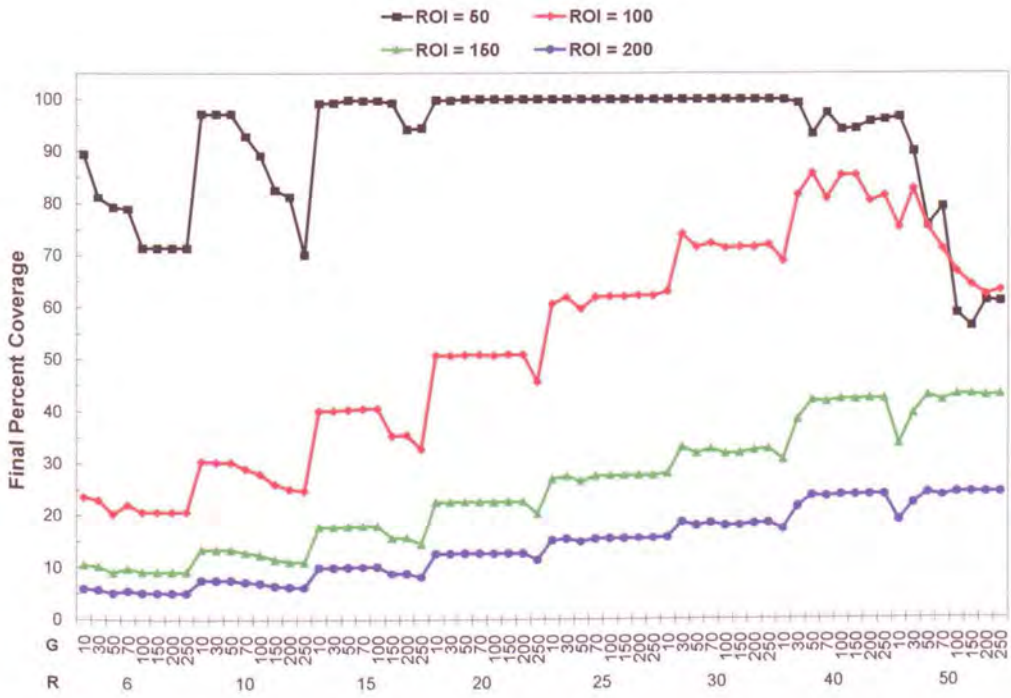


Figure 5.10: A plot of the final percent coverage for different R and G combinations for different sized ROIs (in sq m).

outside the ROI under these conditions.

Figure 5.11 further shows the affect that varying the R parameter has on the overall percent coverage, again showing the outcome for 4 different sized ROIs. Again these experiments involve the swarm starting in a tight cluster in the centre of the ROI, and so it follows that for the smaller sized ($50m^2$) ROI complete coverage is possible, and indeed is achieved over a range of R values. For $R > 35$ however, the percent coverage drops below 100 percent, as the swarm expands to the point where agents are no longer within the ROI. As the size of ROI is increased to $100m^2$ complete coverage becomes impossible given the number of agents, however we still see a narrow range for optimum R value. As we increase the size of ROI further, the range of R values that provide optimum coverage increases, as the swarm separates to a point where there is no sensor overlap, and because of the large ROI all agents stay within this area. Of course if R was increased further still, there would reach a point where agents move outside the ROI as before.

The optimum value of R for a given ROI size is of course dependent on the agents sensors FOV, which are assumed here to be constant.

Varying parameter G has a less prominent affect on the final coverage once the agents have converged, although can still be seen to bring about a change of upto

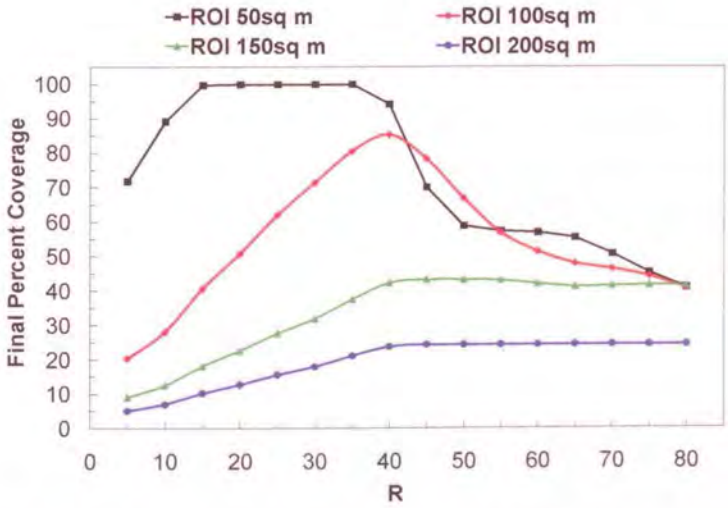


Figure 5.11: A plot of the final percent coverage for different values of R, with $G = 100$, for various sized ROIs.

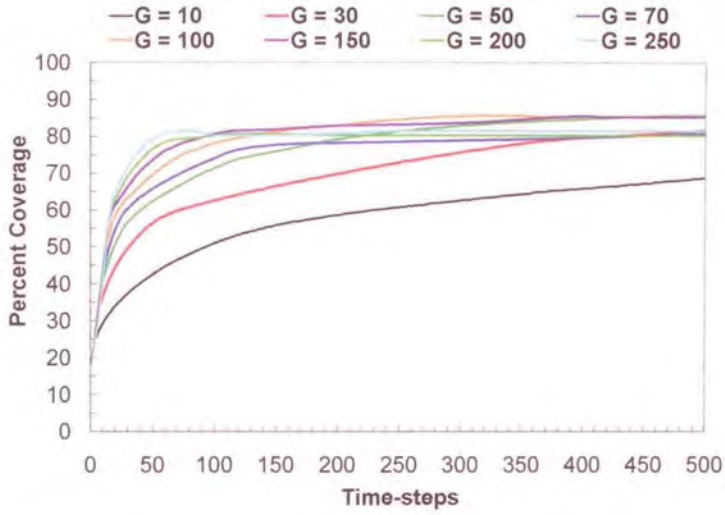


Figure 5.12: A plot of percent coverage vs. time-steps for different values of G, with $R = 40$.

8 percent-coverage under the same R value. Figure 5.12 illustrates how parameter G affects the rate of convergence toward the final stable configuration, showing percent coverage vs. time-steps for various values of G with $R = 40$ for the same scenario as previously described (7 agents with a 100m by 100m ROI). As the rate of change in percent coverage reaches zero, the swarm has converged into a stable configuration, and this gives the final percent coverage achieved. As we have seen in Figure 5.10, the value of G does not dramatically affect the final percent coverage achieved for the given scenario, however, as Figure 5.12 shows, parameter G does have a significant impact on the rate of convergence; a factor of importance when considering time sensitive situations which would require fast convergence, or alternatively requiring a slow convergence pattern if the desired behaviour was for the swarm to perform a sweeping search of a large area over time.

5.4.2 Parameter Optimisation by a Genetic Algorithm

Chapter 3 explored the effects of different parameters for the ant-algorithm swarm in digital image environments, much like the work here for the physics-based approach to self-organisation of robotic entities for the coverage problem. The work of chapter 4 developed a distributed adaptive parameter method and showed

how this could improve self-organisation performance. The adaptive parameter method could be considered *on-line* in that the parameters were adapted as the algorithm was running, in-situ. In this section an *off-line* method is studied, where parameter optimisation is carried out as a separate stage, with a evolutionary learning method.

From an operational planning point of view, a genetic algorithm [54] is implemented to learn the best R and G combination for maximum coverage in a stable converged formation, prior to deployment. The aim here is to realise the potential of learning with regards to off-line planning for specific scenarios, within the framework presented within this chapter. This is also a first step towards developing on-line learning techniques for real-time adaption to changing scenarios whilst on deployment, similar to the adaptive threshold presented in the previous chapter.

Genetic algorithms are commonly used when an optimum solution is required from a large search space, and are used to tackle a wide range of optimisation problems, including those relating to robotics, for example [82][124], and [125] where control parameters were obtained with a genetic algorithm for a similar lattice formation problem.

The genetic algorithm evolves a population of candidate solutions, in this case combinations of R and G, using single-point crossover and single-bit mutation, with a fitness function represented by the total final coverage of the $100m^2$ ROI. For each candidate solution, the experiment is run using the control laws defined above (see section 5.3).

Figure 5.13 shows the average over 20 runs of the average fitness of the gene population over 15 evolutionary steps. We can see an overall increase in average fitness with increasing evolutionary steps, showing a convergence toward better performing R, G combinations within the population of candidate solutions. Although the increase appears small, this is due to the limited population size, such that good solutions are in some cases found almost immediately (hence the average fitness being approximately 0.52 in the first generation).

Figure 5.14 details the average over 20 runs of the number of times each possible R, G combination occurred during the 15 generations of the genetic algorithm, along with the known percentage coverage (from Figure 5.10). We see that the

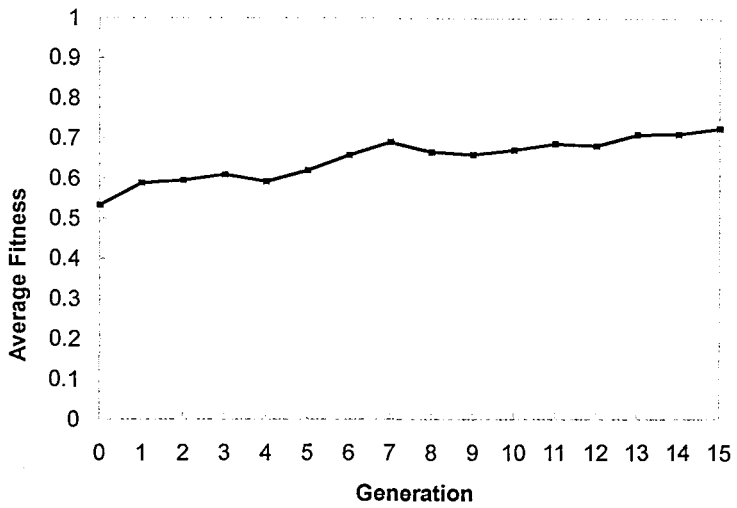


Figure 5.13: A plot of average fitness vs. evolutionary steps (averaged over 20 runs).

frequencies of the different R, G combination occurrences follow the general trend of increasing total percent coverage, with the higher frequency R, G combination occurrences corresponding to the higher achieved total percentage coverage.

Although the search space used is relatively small (hence the small gradient in Figure 5.13), we have seen that this approach is a viable option for learning parameter settings within the control-law framework and application/problem setting considered in this case study.

5.5 A Case Study in Target Localisation

This next case study stays with the subject of surveillance and monitoring, but encompasses additional challenges beyond just the coverage problem. This case study acts as a test-bed for a proof-of-concept multi-behaviour extension to the previously presented Newtonian control law, utilising additional self-organising distributed behaviours to tackle a more complex scenario.

As a more complex case-study, a target localisation problem is considered, in the form of an Improvised Explosives Device (IED) route clearance scenario. Countering IEDs is a major ongoing problem receiving a vast amount of research and development across a broad range of disciplines. A large sub-set of countering

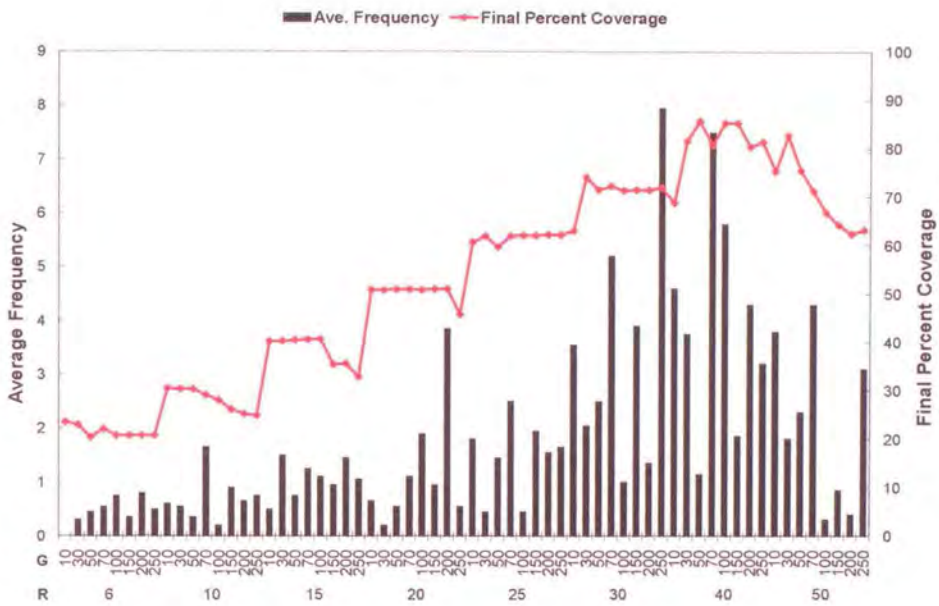


Figure 5.14: A plot showing the frequency of different R, G pairs after 15 generations of the genetic algorithm (left axis, averaged over 20 runs), along with the known final percent coverage (right axis)

IEDs is the problem of physically detecting them. Traditional detection methods, both in the military domain as well as in humanitarian demining, largely involve manual detection, which is a slow and hazardous method [126]. In recent years a multitude of sensor technologies have been developed to tackle the problem of physically detecting the vast array of different types of IEDs being used today [127]. These technologies include visible light, infra-red and thermal imaging systems, ground-penetrating radar, acoustic sensors, magnetic resonance and chemical detection such as the method reported in [128]. Methods of deployment for these sensors include airborne, attached to vehicles, carried by people, and attached on-board robots. In cases where closer inspection is required, or indeed when ground level searching would provide higher accuracy in detection, robotic solutions offer the great benefit of distancing human contact from potential IEDs. Existing deployed robots such as the Talon [129] are remote controlled by a human operator, and are used mostly for explosive ordinance disposal (EOD), and to inspect specific locations where a suspected IED has been located.

For the purpose of wide-area searching, future multi-robot systems have the po-

tential to autonomously search large areas at close range, increasing efficiency in terms of time and reliability of detection (due to the use of multiple robots simultaneously), while distancing human contact from close contact with potential IEDs. Such autonomous outdoor small-scale robots are being developed (for example see [130]) although are not currently widely operational.

This section describes the elementary functionality of a swarm robotics approach to this problem and provides proof of concept laboratory experiments to highlight the main properties of the proposed system.

5.5.1 Problem Definition

The overall goal of the robot swarm is to provide the coordinates of any IEDs present in a specified search area. For the purpose of the laboratory experiments presented here it is assumed that the robots are capable of detecting, to a given level of accuracy, an IED that is in close proximity (see Section 5.5.2). The sub-goals of the robot swarm are to disperse from a common deployment location, search the environment for IEDs, and upon detection, monitor the IED location and inform the operator of a positive detection. Informing the operator could be achieved by means of visual or audible communication/broadcast of some form, or simple one-way communication to a central command post or nearby support vehicle. With the additional ability to track the robots locations, there would be potential to create a map of detected potential IED locations across a wide area. Such implementation issues are however outside the scope of study of this thesis, as the focus remains on the underlying self-organising pattern formation behaviours.

5.5.2 The Proposed System

The proposed approach is swarm intelligence driven, and in particular agrees with the general accepted criteria for swarm robotics systems [55]. Perhaps the most relevant and important consequence of adopting swarm intelligence based methods is that the design will focus on a decentralised approach. A decentralised approach reduces the levels of inter-robot dependency, and in turn increases the scalability of the system, as well as providing potential for high levels of fault tol-

erance in terms of robot redundancy. For the particular application presented in this case-study, an increase in scalability is useful when considering the flexibility of the system, for example being able to increase the number of deployed robots to search a larger area, with no additional computational cost. High tolerance to robot failure is of obvious benefit to this application given the dangerous nature of the task being carried out by the robots. This approach also aims to minimise the sensory and communication requirements of the robots, again to increase scalability and robustness, as well as to reduce the cost and potential size of the robots, thus also increasing the expendability and portability of individual robots, which is again of particular interest to this chosen application.

5.5.3 Coordination Control Laws

As described in section 5.3, the coordination control laws are central to the collective behaviour of the swarm, computing the trajectories of individual robots based on their local environment.

The aforementioned indirect communication is implemented here as virtual forces which each robot experiences as a result of its external neighbourhood and internal local state. All virtual forces are calculated from range and bearing measurements to neighbouring robots, targets and virtual beacons, as measured by the individual robot. The range and bearing measurements are input into physics-based force laws, namely a split-Newtonian potential inspired law (based on Newton's law of gravitation, as above), and a Lennard-Jones potential inspired law, to compute the virtual forces which are then weighted to give a total force vector acing upon the individual robot, which is in turn used to compute a new displacement vector.

Figure 5.15 shows a comparison plot of the split Newtonian and Lennard-Jones potentials. There are two different characteristics of particular relevance. The Lennard-Jones potential provides a more gradual change between attractive and repulsive forces, in contrast to the split Newtonian which has a sharp jump between positive and negative F. The split Newtonian becomes smaller with increasing r for both positive and negative F, whereas for the Lennard-Jones potential increasing r gives increasing negative F. These different characteristics are used for different purposes in the swarm behaviours described below.

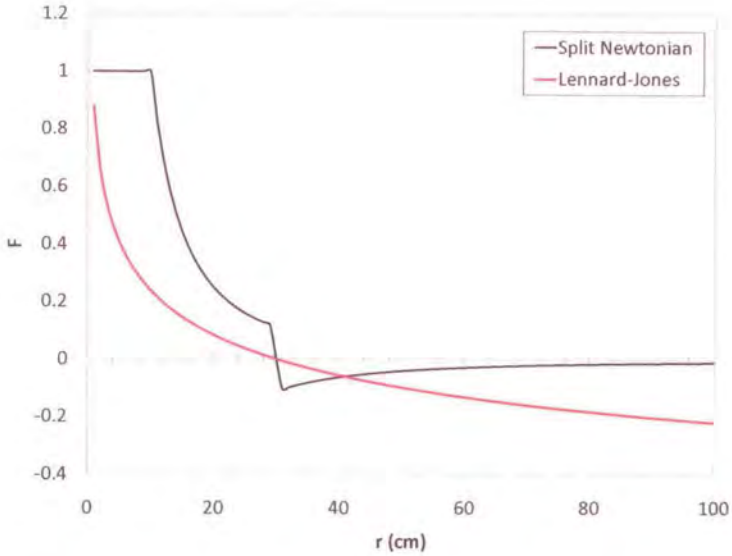


Figure 5.15: A plot comparing the split-Newtonian and Lennard-Jones potential.

The work in the previous case-study is extended here by developing different virtual force methods into a number of specific behaviours, all based on self-organising collective movement, which are then incorporated into a multi behaviour system for the given case study.

This method is particularly useful for self-organising entities into regular patterns [67][68][131] and can be modified with relative ease to facilitate a number of tasks relevant to the problem application posed in this case-study. This method of coordinated control is fully distributed, with each individual robot computing its own virtual forces experienced, with no need to compute a global force map, and no inter-robot communication required. The adopted approach is a versatile and powerful tool for implementing distributed, self-organised behaviour in a swarm of mobile robots.

Formation Control

The basic control law yields the self-organisation of the swarm into a regular spaced repeating lattice formation. This is achieved by each individual robot measuring the range, r, and bearing, θ, to any neighbouring robots $n \in N$, within a given 'visible range', r_{vis}. The visible range is user-set, however there may also be physical constraints imposed depending on the sensor method used to

determine the range to neighbouring robots. For formation control the Lennard-Jones based potential is used here, as it provides a more gradual change between attractive and repulsive forces, which reduces the potential robot oscillation effect around the desired robot locations, especially when considering differential-drive robots. The inter-robot force, F_R experienced due to neighbouring robot n is given by:

$$F_{Rn} = \begin{cases} 4\epsilon \left[\left(\frac{R}{r}\right)^{\sigma} - \left(\frac{R}{r}\right)^{\tau} \right] & \text{if } r \leq r_{vis} \\ 0 & \text{otherwise} \end{cases} \tag{5.11}$$

where ϵ is the maximum allowed attractive force, R is the desired separation distance between neighbouring robots, and σ and τ are control parameters. An additional constraint is applied to limit the allowed overall force, such that $F_{Rn} = \epsilon$ if $F_{Rn} > \epsilon$ and $F_{Rn} = -\epsilon$ if $F_{Rn} < -\epsilon$. The x and y components are given by:

$$F_{Rnx} = F_{Rn} cos(\theta_n) \tag{5.12}$$

$$F_{Rny} = F_{Rn} sin(\theta_n) \tag{5.13}$$

The total x and y components of the force experienced are then given by:

$$F_{Rtotalx} = \sum_{n=1}^{N} F_{Rnx} \tag{5.14}$$

$$F_{Rtotaly} = \sum_{n=1}^{N} F_{Rny} \tag{5.15}$$

A discrete-time approximation is employed, such that the resultant x and y components of the velocity are calculated as:

$$v_x = F_{Rtotalx} \Delta t \tag{5.16}$$

$$v_y = F_{Rtotaly} \Delta t \tag{5.17}$$

From v_x and v_y the new displacement vectors are calculated as:

$$\Delta x = v_x \Delta t \tag{5.18}$$

$$\Delta y = v_y \Delta t \tag{5.19}$$

The calculated Δx and Δy, together with the robot's own bearing ϕ, are then input into a low-level motor controller to actuate the robot's movement towards the new desired location (see Appendix D for details on the low-level motor controller).

Distributed Searching

The distributed search behaviour employs the split-Newtonian based potential, and includes an additional internal factor as a force acting upon the robot. This behaviour aims to distribute the swarm from a common deployment location to disperse and wander the search space. The split-Newtonian based potential is implemented with a more abrupt change in attractive and repulsive forces, to achieve a higher distribution response amongst the robots. The force experienced due to neighbouring robot n, when executing the distributed search behaviour, is given by:

$$F_{RSn} = \begin{cases} G/r^2 & \text{if } r \leq R \text{ and } r \leq r_{vis} \\ -G/r^2 & \text{if } r > R \text{ and } r \leq r_{vis} \\ 0 & \text{otherwise} \end{cases} \tag{5.20}$$

where G is the gravitational constant, which effects the rate of change in force with distance. The total x and y components are computed in the same way as described above, and again an additional constraint is applied to limit the allowed overall force, such that $F_{RSn} = \epsilon$ if $F_{RSn} > \epsilon$ and $F_{RSn} = -\epsilon$ if $F_{RSn} < -\epsilon$.

The inclusion of the additional force is nature inspired by the foraging behaviour of ants in the wild. The basic concept is that foraging ants are more likely to continue moving in their current heading and are less likely to make turns at large angles to their current heading [42]. To mimic this behaviour an attractive

force is included directly infront of the robot in addition to the forces due to neighbouring robots. The internal force is calculated in a similar way to the inter-robot forces, assuming an attractive virtual beacon at distance S directly infront of the robot, the force is calculated as:

$$F_I = -G/S^2 \tag{5.21}$$

with the x and y components given by:

$$F_{Ix} = F_I cos(\phi) \tag{5.22}$$

$$F_{Iy} = F_I sin(\phi) \tag{5.23}$$

The x and y components of the total force experienced under the distributed search control law, F_{Dtotal}, are given by a weighted combination of $F_{RStotal}$ and F_I:

$$F_{Dtotalx} = (\alpha F_{RStotalx}) + (\rho F_{Ix}) \tag{5.24}$$

$$F_{Dtotaly} = (\alpha F_{RStotaly}) + (\rho F_{Iy}) \tag{5.25}$$

where α is the inter-robot force weight and ρ is the search force weight.

Surrounding Targets

The control law to facilitate the behaviour of surrounding a target is again a simple extension to the basic formation control, similarly using the Lennard-Jones based potential. The inter-robot force, F_R, is again calculated from Equation 5.11. Upon detecting the target it is assumed that the robot ascertains the location of the target relative to its own position, so as to be able to calculate the target range, r_T, and bearing, θ_T. The robot then experiences a force due to the detected target, given by:

$$F_T = 4\epsilon \left[\left(\frac{R_T}{r_T} \right)^{\sigma} - \left(\frac{R_T}{r_T} \right)^{\tau} \right] \tag{5.26}$$

where R_T is the desired distance of the robot from the target. As previously, the x and y components are given by:

$$F_{T_x} = F_T cos(\theta_T) \tag{5.27}$$

$$F_{T_y} = F_T sin(\theta_T) \tag{5.28}$$

As with the distributed search behaviour, the x and y components of the total force experienced under the target surround control law, F_{Stotal}, are given by a weighted combination of the total inter-robot force, F_{Rtotal}, as calculated by Equations 5.11-5.15, and in this case, the force due to the detected target, F_T:

$$F_{Stotalx} = (\alpha F_{Rtotalx}) + (\mu F_{Tx}) \tag{5.29}$$

$$F_{Stotaly} = (\alpha F_{Rtotaly}) + (\mu F_{Ty}) \tag{5.30}$$

where μ is the target force weight.

5.5.4 Robot System Architecture

The system is designed not to be limited to a specific robotic platform. The above control laws are the central components to the collective behaviour of the system at the swarm level[1], and are intended to be applicable to most types of mobile robotic platforms.

The scenario is scaled down to enable proof of concept testing and analysis in a laboratory setting, using the e-puck [114] robot as the UGV platform. The e-puck is a small-scale differential drive laboratory robot approximately 7cm in diameter.

[1] *Swarm level* refers to the properties of the system when considering all of the robots collectively as a single swarm entity, rather than the properties of individual robots.

It has a number of on-board sensors including: a forward facing camera, 8 IR proximity sensors, accelerometer and 3 microphones. It is also equipped with a speaker, an LED ring, and supports Bluetooth communication.

With the transition from 2D point mass simulation to physics and sensor based simulation (see below), it is now important to take into account the possibility of collisions. As a safety critical fail-safe, a reactive obstacle avoidance behaviour is implemented separate from the swarm-level behaviours, as a individual behaviour with an overriding priority. A generic reactive obstacle avoidance method is used (see Appendix E for details) utilising the e-puck's on-board proximity sensors (of course this would be implemented accordingly given the specifics of the particular robotic platform being used). The use of the physics control laws for dealing with structures in the environment is dealt with in the next chapter.

The different behavioural controls are implemented in the form of a Finite State Machine (FSM) as shown in Figure 5.16. The state transitions in the FSM are triggered by conditions based on sensory information received from the various on-board sensors of the robot. An overview of the individual robot architecture is given in Figure 5.17. Whilst in the distributed search state the robots only experience forces from their internal search beacons and any neighbouring robots within their local visible range, r_{vis}, executing the control law as defined by the equations in section 5.5.3. A given robot does not experience any force due to any target until that robot has physically detected the specific target via its on-board camera. Once a specific target has been located, its location is assumed known to the detecting robot, and that robot then enters the target surround behavioural state, at which point it now experiences a force due to the detected target, as given by Equation 5.26, and executes the control law as defined by the equations in section 5.5.3.

Again, the coordination control laws output displacement vectors, which, for the chosen robots, need to be converted into left and right wheel velocities. A motor control law derived from the one reported in [132] has been developed for this purpose, which provides smooth closed-loop steering for differential-drive robots towards the desired location (see Appendix D for details).

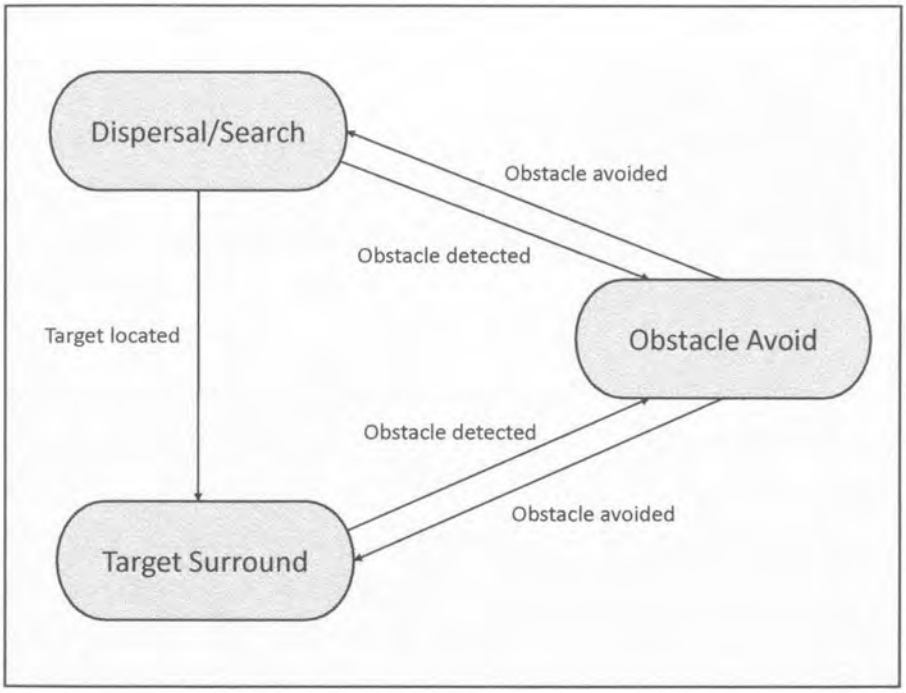


Figure 5.16: A schematic diagram of the Finite State Machine.

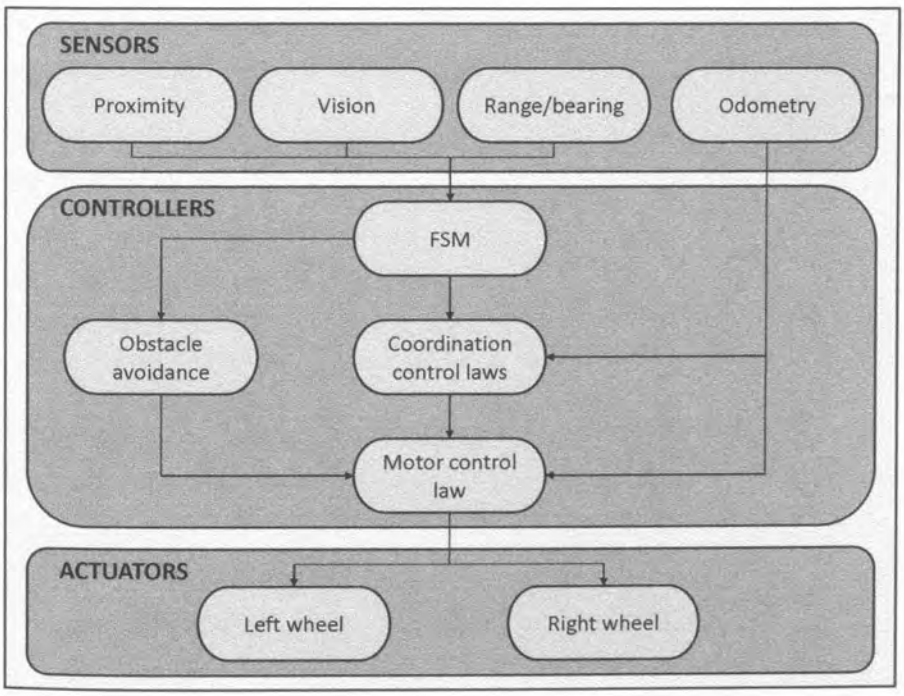


Figure 5.17: A schematic diagram of the robot system architecture.

5.5.5 Sensor and Physics-based Simulations

Experiments are carried out in a sensor and physics-based simulation environment using the Webots professional mobile robot simulation package [133], which includes an accurate model of the e-puck robot, with simulated on-board sensors. Figure 5.18 shows a photograph of 3 e-puck robots in a triangular formation in the laboratory environment, on the left, with the corresponding Webots simulation on the right.

In Figure 5.17, for the chosen robotic platform, the proximity sensors are the e-puck's on-board IR sensors. The vision is the e-puck's on-board camera, however this could be any equivalent sensor with the ability to detect the presence of the specific targets, with the ability to provide an estimate of the target's position relative to the robot. For the results reported here the range/bearing sensors are simulated by filtering absolute known positions of all the robots in the simulation. Similar experiments have been carried out simulating an on-board IR-based range/bearing system similar to the one reported in [134], yielding comparative results to using filtered absolute positioning. The odometry requirements are such that each robot knows its own global bearing, knowledge which is assumed the robots can obtain (for example with an on-board compass).

Since the problem of the actual detection of the IEDs is beyond the scope of the work of this thesis, this process is simplified for the purpose of the experiments, and the IEDs are represented as coloured markers within the environment. Self-developed image processing algorithms are used to facilitate vision based detection of the IEDs via the robots' on-board cameras (see Appendix F for details).

Figure 5.19 shows an example experiment set-up. The image on the bottom shows a view of the Webots simulation environment at time $t = 0$, and the image on the top details a 2D map of the same. The environment shown in Figure 5.19 is of a bounded rectangular shape, chosen to bear resemblance to a stretch of road in a real-world scenario. Solid obstacles are included to simulate any generic real-world obstacle that may be encountered (such as rocks, debris), and in the case of Figure 5.19 there are 3 targets to locate.

Since the focus of this research is not concerned with the design of individual robots and their performance, concern is not given to such external issues as

Figure 5.18: A photograph of 3 e-puck robots (left), and the corresponding physics and sensor based simulation in Webots (right).

terrain type, gradient, atmospheric effects and so forth. Although it is appreciated that these are important factors when considering operational deployment of a developed system, for the research in this thesis it is assumed that equivalent deployable robotic platforms to the ones used in the laboratory experiments would be available (for example [130]). The sensor and physics-based simulation environments thus use a simple zero gradient, smooth surface as the ground.

Wheel slippage is also ignored in the work presented in this case-study (such effects are investigated later), and a kinematics based model is used to simulate the robots' movements. All objects within the environment are simulated with bounding boxes to enable realistic simulation of any occurring collisions, including those between robots.

5.5.6 Laboratory Experiments

This section presents and analyses results from a number of laboratory experiments designed to show the self-organising behaviour obtained from the developed control laws. Furthermore this section reports results from experiments to show the scalability, adaptability and robustness of the system. Finally a case study experiment is shown, designed to simulate the specific task of target localisation and monitoring, with reference to counter IED operations.

Figure 5.20 shows plots of the robot locations ($N = 10$) and past trajectories at different points in time during an experiment showing the self-organisation of the robot swarm from a compact deployment configuration with a common bearing $\phi = 0$, to a quasi regular spaced lattice configuration. This experiment

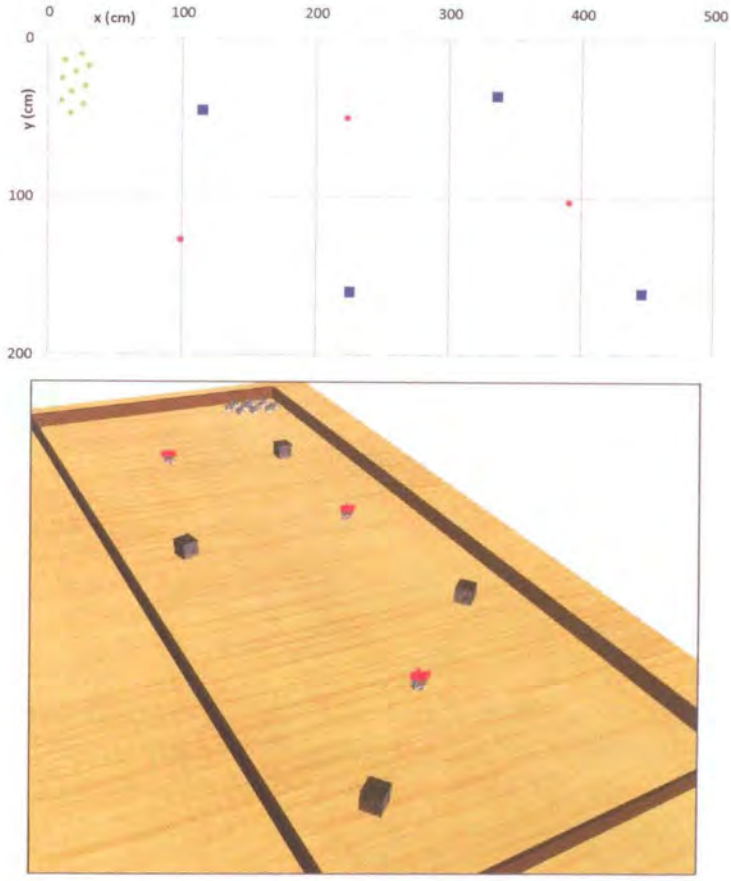


Figure 5.19: A screen-shot of the Webots simulation environment (bottom) with the corresponding 2D map (top). In the 2D map, solid green circles represent robot locations, solid red circles indicate target locations, and solid blue squares indicate obstacle locations.

was carried out using only the formation control law described in section 5.5.3, with $\epsilon = 1.0$, $R = 30cm$, $r_{vis} = 150cm$, $\sigma = 0.1$ and $\tau = 0.05$. From Figure 5.17 the robots use here only the range/bearing and odometry sensors, which provide information of the range and bearing to neighbouring robots every 100ms for the formation coordination control law. The formation coordination and control law then calculates a new displacement vector which the motor control law converts into left and right wheel speeds to drive the robot towards the desired location. From time $t = 1s$ each robot manoeuvres in response to the virtual forces experienced due to the relative positions of the neighbouring robots, with the design of the control law ensuring that each robot attempts to maintain a specified

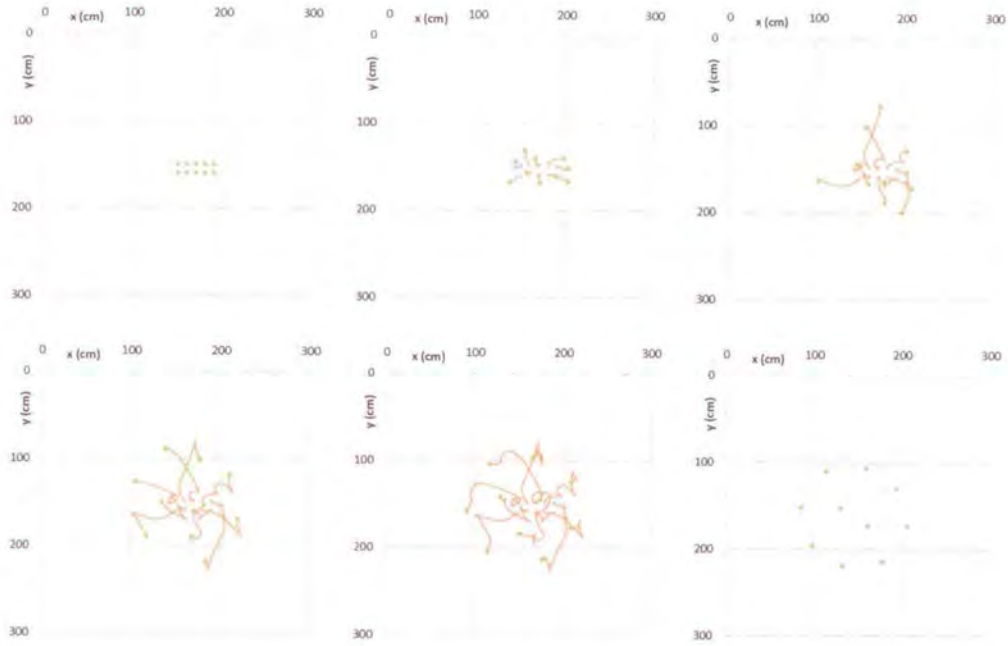


Figure 5.20: Plots of the robot locations and trajectories at different time-steps during a formation experiment. Top-left: $t = 1s$; top-middle: $t = 10s$; top-right: $t = 20s$; bottom-left: $t = 30s$; bottom-middle: $t = 100s$; bottom-right: $t = 600s$. Solid green circles indicate robot locations, orange lines indicate the trajectory history.

separation distance R from neighbouring robots. This self-organising behaviour provides a fully distributed method of creating and maintaining a regular formation pattern of mobile robots. This behaviour is a useful method for use in a range of applications, such as providing mobile sensor lattices (as shown in the previous case study), mobile communication lattices and autonomous payload distribution.

Figure 5.21 shows the average separation distance between robots versus time for the formation experiment, carried out with 3, 5, 10, 20 and 30 robots separately. As the gradients of the plots approach zero, the swarm has reached a stable configuration. We see from Figure 5.21 that the time taken to reach a stable configuration does not vary significantly with increasing numbers of robots between 3 and 30. This shows a degree of scalability of the proposed system for small sized swarms (in the range considered as plausible for the given application). This scalability can in part be attributed to the distributed nature of the

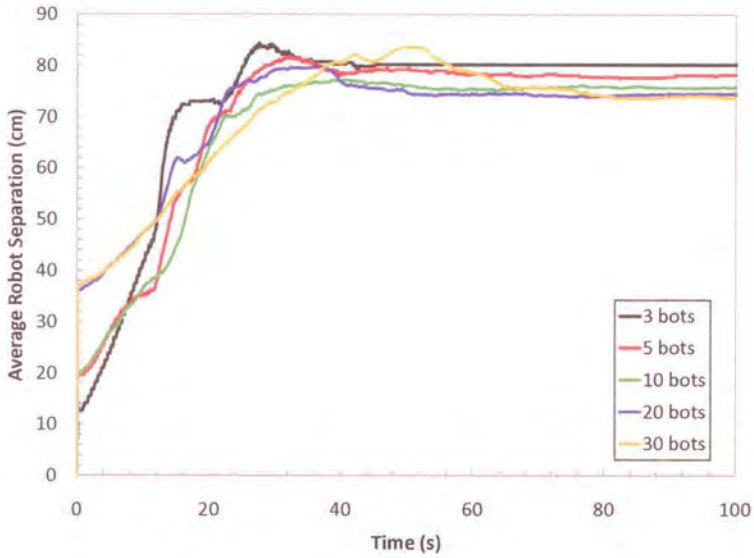


Figure 5.21: A plot showing $P_{\overline{ave}}$, the average separation distance between robots vs. time, for a formation experiment.

swarm robotics approach, which greatly improves the flexibility of the system. The key factor to the distributed nature of the system is the non-requirement for explicit inter-robot communication. The small increase in convergence time in Figure 5.21 is simply due to the fact that with an increasing number of robots, some of the robots may have to travel further before the swarm achieves stability.

Section 5.5.3 detailed the extension of the formation coordination and control law to facilitate a target surround behaviour. The target surround behaviour is presented in an experiment where 6 robots have the goal of surrounding a target of known location. Also shown is the robustness and adaptability of the behaviour by removing 2 of the robots (to simulate total robot failure / destruction) and moving the target, at separate occasions during the experiment. The target surround behaviour was run with the following settings: $\epsilon = 1.0$, $R = 30cm$, $R_T = 15cm$, $r_{vis} = 50cm$, $\sigma = 0.1$, $\tau = 0.05$, $\alpha = 1.0$ and $\mu = 2.0$. Figure 5.22 shows plots of the robot locations and past trajectories, along with the target location, at different points in time during the experiment. At time $t = 1s$ the robots start in a line formation approximately $90cm$ North of the target. Guided by the target surround coordination and control law the robots approach the target then surround the target, forming a unified perimeter at specified distance R_T. After the robots have surrounded the target, at time $t = 120s$

two robots are removed, forcing the swarm to reconfigure in order to maintain a uniform perimeter. Once the swarm has reconfigured around the target, at time $t = 180s$ the target is moved approximately $60cm$ North of the robots. The robots then manoeuvre to the target's new location and achieve a uniform surrounding perimeter by time $t = 300s$.

Figure 5.23 shows the average inter-robot distance and the average distance between the robots and the target, versus time, for the above experiment. Both the inter-robot and robot-target plots reach approximately zero gradient by approximately $30s$. We see a slight deviation from zero gradient in both the inter-robot and robot-target separation at time $t = 120s$ where two of the robots are removed from the experiment. At time $t = 180s$ we see a large spike in the robot-target separation distance, as the target is moved, with a rapid convergence back to zero gradient at the previous separation distance, showing the fast response of the swarm to the target displacement. The inter-robot separation distance only suffers a minor deviation from zero gradient when the target is moved, as the robots remain in close proximity to one another as they move towards the new target location.

The final experiment for this case-study utilises the full system of the FSM (Figure 5.16) and robot architecture given in Figure 5.17. This experiment simulates the problem of searching for targets of known appearance, and once located, surrounding the targets to provide visual coverage and a physical perimeter. Again this simulation can be likened to a real-world scenario of searching for and localising IEDs in a given area.

The experiment takes place in a bounded environment of a size 500cm by 200cm, with 4 obstacles and 3 targets, as shown in Figure 5.19. The robots deploy from a common corner of the environment, in a pseudo-random cluster, with random orientations ϕ, and execute the FSM, starting in the distributed search state. The target surround behaviour is executed with the same parameters as above. The distributed search parameters are set-up as with the target surround experiment, above, with the additional distributed search parameters taking the values: $G = 500$, $S = 10cm$, $\rho = 1.0$ and, $r_{vis} = 250cm$ when executing the distributed search behaviour.

Figure 5.24 gives 2D plots of the environment showing the robot positions ($N =$

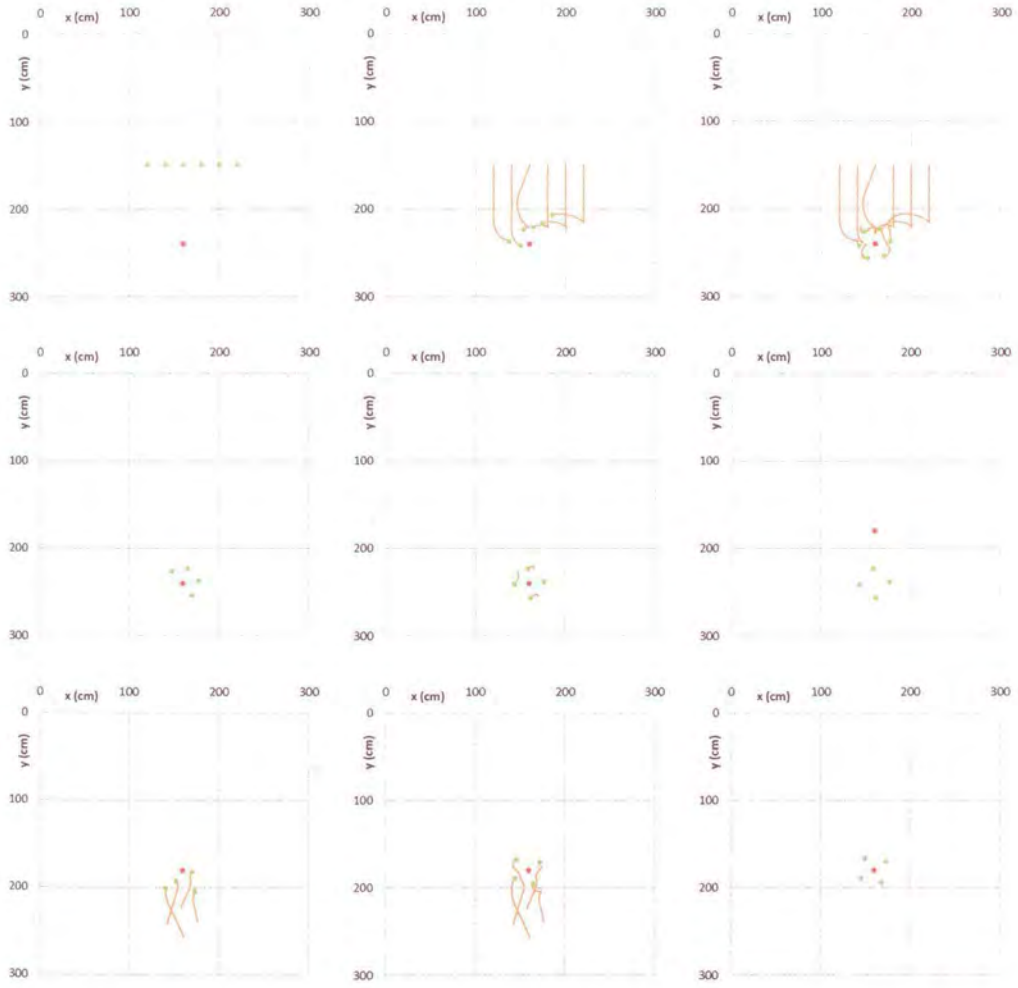


Figure 5.22: Plots of the target location and robot locations and trajectories at different time-steps, showing adaptability and robustness to robot failure and target movement. Top-left: $t = 1s$; top-centre: $t = 10s$; top-right: $t = 100s$; middle-left: $t = 120s$; middle-centre: $t = 125s$; middle-right: $t = 180s$; bottom-left: $t = 185s$; bottom-centre: $t = 200s$; bottom-right: $t = 300s$. Solid green circles indicate robot locations, orange lines indicate robot trajectory history, a solid red circle indicates the target location.

10) at different points in time. From time $t = 1s$ the robots execute the distributed search behaviour, distributing from the deployment location to search the environment for targets. As the robots detect targets they change to the target surround state and manoeuvre to within a 15cm range of the target. As more robots detect the same target they manoeuvre into a regular perimeter

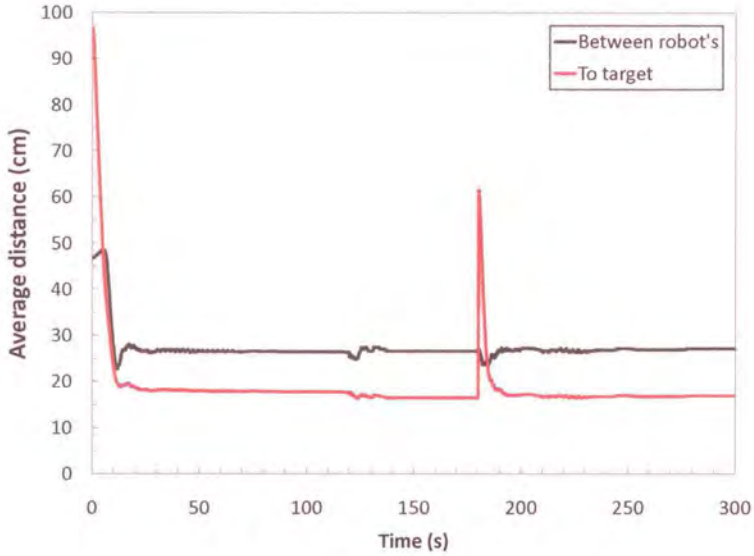


Figure 5.23: A plot showing the average separation distance between robots and between all robots and the target, vs. time, for a target surround experiment.

around the target, providing multiple FOV coverage of the target and physically enclosing the target as best as possible given the number of robots. From Figure 5.24 we see that by time $t = 100s$ all three targets have been located, with 2 robots monitoring one target and 1 robot monitoring the other two targets, respectively. By time $t = 900s$ all of the robots have detected a target, with 4 robots monitoring two of the targets respectively and 2 robots monitoring the other target.

Figure 5.25 shows plots of the average inter-robot separation, averaged over 5 experiment runs, versus time for different numbers of robots. As the gradients converge to zero this corresponds to more robots surrounding targets, thus reducing the degree of change in the inter-robot separation. We see similar convergence times for 3 to 15 robots, showing a degree of scalability of the system for the given task scenario of searching and surrounding targets, using relatively small groups of robots. We note that the lower average separation for 3 robots is due to the fact that only two targets were discovered when running the experiment with only 3 robots. The target furthest from the deployment location (Figure 5.19) was never located, and thus the range of the average separation only spanned between the two targets nearest the deployment location, once convergence on the targets had occurred.

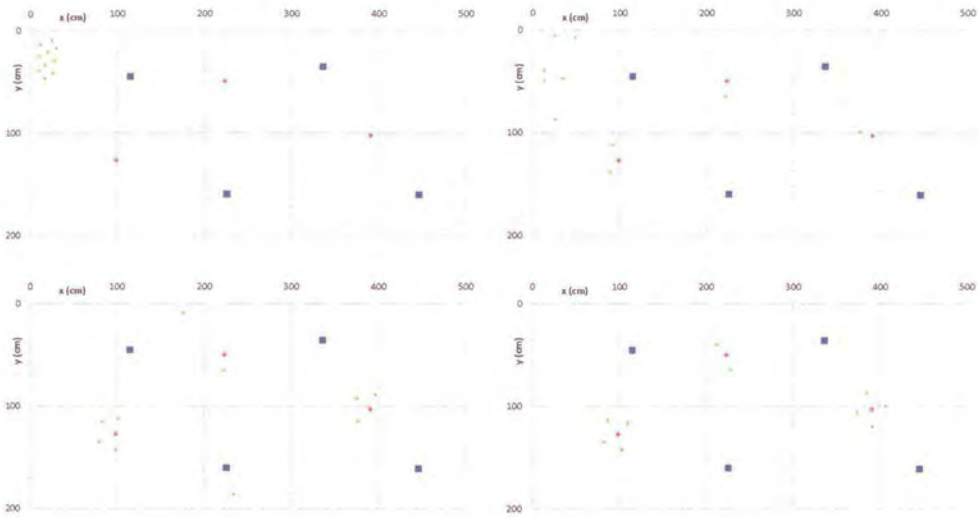


Figure 5.24: Plots of the robot locations at different time-steps during a search and surround experiment. Top-left: $t = 1s$; top-right: $t = 100s$; bottom-left: $t = 300s$; bottom-right: $t = 900s$. Solid green circles indicate robot locations, solid red circles indicate target locations, solid blue squares indicate obstacle locations.

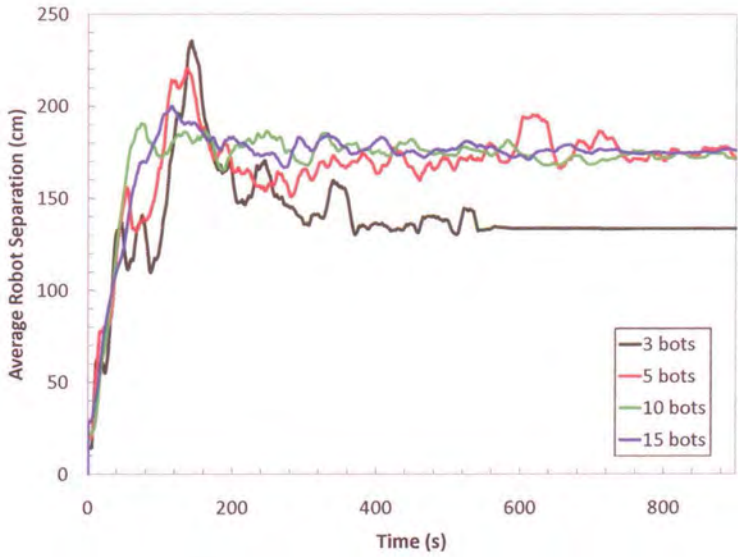


Figure 5.25: A plot showing $P_{\overline{ave}}$, the average separation distance between robots vs. time, for different numbers of robots performing a search and surround experiment.

Table 5.1: A table showing the times taken to locate targets, and distribution scores, for the target search and surround experiment performed with different numbers of robots. Results are averaged over 5 experiment runs.

No. Robots	Time taken to locate (s)			Distribution score, Ψ
	1 Target	2 Targets	3 Targets	
3	60.7	194.5	—	0.67
5	79.8	220.6	472.0	0.68
10	57.8	104.8	211.7	0.78
15	36.2	75.9	114.2	0.84

Table 5.1 shows the average times taken for 1, 2 and 3 targets to be located for different numbers of robots performing the target search and surround experiment. Also shown is a distribution score (ranging from 0 to 1, with 1 being optimum), which is derived from the number of robots distributed around each detected target, with respect to the number of available robots (specifically, a normalised ratio of the number of robots around each target to the robot/target ratio, averaged over the total number of targets).

The distribution score, Ψ is given by:

$$\Psi = \frac{\sum_{t=1}^{T} \left[\frac{\chi_t}{N/T} \right]}{T} \tag{5.31}$$

where N is the number of robots, T is the number of targets, and χ_t is the scaled number of robots around target t, such that:

$$\chi_t = \begin{cases} \sum n \in t & \text{if } \sum n \in t \leq N/T \\ N/T & \text{otherwise} \end{cases} \tag{5.32}$$

With 15 robots all targets are located in the fastest time, with the highest distribution score, which is perhaps not surprising given the robot to target ratio. Interestingly the results for 5 robots yield the slowest detection times, which suggests that increasing N will not necessarily increase performance.

5.6 Implementation with Real Robots

In this section a number of experiments are carried out, designed to assess various elements of robustness of the proposed swarm formation control laws. Although the experiments are carried out using real robots, this is only considered as 'embedded' simulation; a particular scenario is being simulated, and the swarm agents are being embedded as real robots, in a real world. The experiments are carried out in a controlled laboratory environment, using small scale robotic platforms, with a number of limiting factors as detailed below. As such this cannot be considered in any way 'field testing,' although using simple hardware in a controlled environment does allow for investigation into the basic robustness of the system to noisy sensor and actuator performance. These experiments also provide insight into the level of robustness to robot failure of this approach to self-organising pattern formation and control.

5.6.1 Experimental Set-Up

Experiments are carried out both in a sensor and physics-based simulation environment, using the Webots professional mobile robot simulation package [133]; as well as on real robots in a purpose build robot enclosure (see Figure 5.26, and Appendix B for details). Again the e-puck robot [114] is used as the UGV platform.

Although it is possible to implement on-board range/bearing and odometry measurements[2], due to hardware limitations with the laboratory experiments, for the work presented here these processes are simulated by filtering information from an overhead robot pose tracking system. The tracking system consists of an overhead camera suspended above the robot enclosure, which is connected to a PC running a number of image processing algorithms which track coloured markers placed ontop of the robots and targets within the environment. With the current set-up positions are calculated to approximately $\pm 1.4cm$ and orientations to approximately $\pm 0.05rads$ at a sampling rate of approximately $0.125s$. Each robot is controlled remotely via a Bluetooth link to a PC. Each robot receives filtered

[2]Relative range/bearing estimation can be achieved using for example the IR-sensors, or vision; odometry can be achieved via wheel-based and accelerometer, or visual, odometry.

Figure 5.26: A snapshot of 5 robots surrounding a 'target' in the purpose built robot enclosure.

range and bearing measurements to all neighbouring robots in range, as well as it's own bearing, as measured by the overhead vision-based tracking system. See Appendix B for further information on the overhead tracking system.

This hybrid system utilises a centralised sensor system to simulate distributed sensing, and thus it should be noted that although it would be possible to implement on-board vision based relative range / bearing measurements, it is likely that the resultant sensor noise would in each case be qualitatively different. Such differences might occur due to for example on-board range / bearing measurements being measured slightly differently from on-board each individual robot, whereas in this experimental set-up, each robot receives the same inter-robot distances as measured centrally by the overhead vision tracking system.

Each robot has its own separate controller, as if it were running on-board the robot itself. In contrast to the real robot set-up, the Webots simulation experiments are carried out under 'ideal conditions,' which in practical terms means there is no sensor noise (e.g. lighting fluctuations affecting IR readings and vision) and no actuator noise (e.g. wheel-slip).

Each time-step in the experiment is equivalent to approximately $0.1s$ real-time. During each time-step, each robot calculates the range and bearing to all neigh-

bouring robots, and the target, if present and detected. Each robot then executes one of the control laws, which will in turn output a displacement vector based on the virtual forces acting upon the robot. To convert the displacement vectors into left and right wheel velocities the same motor control law as mentioned previously is used (see Appendix D for details).

For the following experiments the following parameter settings are used: $N = 5$, $r_{vis} = 50cm$, $\epsilon = 1.0$, $\sigma = 0.1$, $\tau = 0.05$, $\alpha = 1.0$ and $\mu = 2.0$, determined by trial and error to give the best performance.

5.6.2 Comparing Real-Robot Results to Physics and Sensor Based Simulations

The first experiment involves 5 robots starting in a line within the robot enclosure. Each robot then executes the *formation control* behaviour, with $R = 30cm$. Figure 5.27 shows four snap-shot plots of the robot positions and past trajectories at various stages in the self-organisation process for the formation experiment with the real robot set-up. Figure 5.28 shows the error in average inter-robot separation distance R, versus time-steps. The plot shows results obtained from a typical experiment run with the real robot set-up, as well as results obtained from a physics and sensor based simulation under ideal conditions. There is a clear difference in stability when comparing the real robots experiment with the 'ideal' simulation experiment. The simulation experiment produces a much smoother curve that, once converged to near zero error ($0.7cm$), follows a zero gradient as the robots maintain their optimum positions. The real robots experiment produces a curve with many more fluctuations. Despite this observation, both curves reach near zero error in approximately equal time (700 time-steps). The real robots curve does not however stabilise to a zero gradient, but instead continues to fluctuate around zero error. This means the robots continue to move small distances when the swarm has converged to the ideal formation. The swarm as a whole does not move, rather each robot constantly adjusts its own position in order to maintain the ideal formation. These fluctuations can be attributed to the noisy data inherent in the real robot set-up. The robots are relying on information from the overhead tracking system in order to compute their range/bearing measurements. This set-up relies on vision based tracking, which

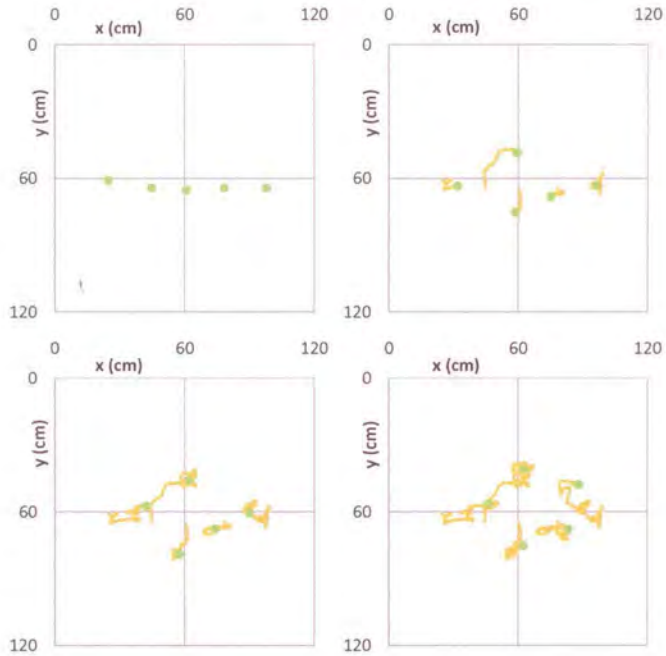


Figure 5.27: Plots showing the positions (green circles) and past trajectories (orange lines) of the robots at various time-steps during the formation experiment with the real robot set-up. Top left: time-step = 1; top right: time-step = 200; bottom left: time-step = 600; bottom right: time-step = 1000.

inherently produces noisy data (for example due to fluctuating ambient lighting). This noisy data is not unlike what would occur if on-board range/bearing sensors were being used, and thus it is encouraging to see that the swarm manages to converge to near zero error and maintain the ideal formation in the presence of such noise, showing robustness to noisy data.

The second experiment involves 5 robots starting in a line, with the goal of collectively moving towards and surrounding a nearby target of known location. Figure 5.29 shows four snap-shot plots of the robot positions and past trajectories at various stages in the self-organisation process for the target surround experiment with the real robot set-up. Figure 5.30 shows typical results of average robot to target range and average robot separation range versus time-steps, obtained from the real robot set-up (corresponding to Figure 5.29) and physics and sensor based simulation, with $R = 30cm$ and $R_T = 15cm$. Comparing the real robots data to the simulation data we see similar characteristics to the first experiment. For both the average robot to target range and average robot separation

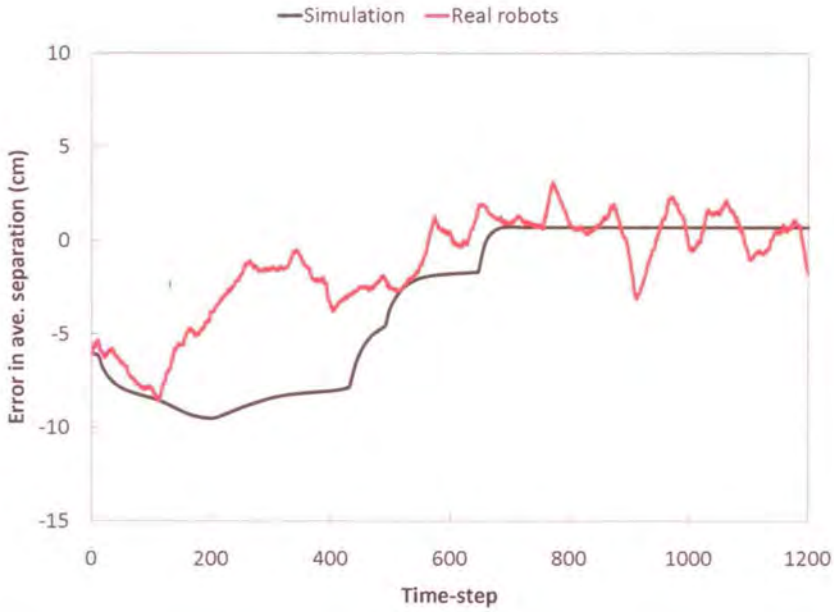


Figure 5.28: Showing the error in average robot separation distance vs. time-steps for a formation control experiment. The plot shows results obtained from real robots as well as through physics and sensor based simulation in ideal settings.

range, the real robots curve follows the general trend of the simulation curve, but exhibits a higher level of fluctuations, attributed to noisy data. By approximately 220 time-steps the robots have reached an optimal range from the target of approximately $17cm$ (at which point the average robot to target range curve approaches zero gradient). Within approximately the same amount of time the average robot separation range also converges to an optimum of approximately $27cm$. We note that with the target surround behaviour, the robot swarm is required to maintain two user specified desired range values (R and R_T), and in the swarm self-organising process there is a trade-off between the two, which depends on the number of robots within the swarm. In this experiment the errors are however relatively low, with approximately $2cm$ for the converged average robot to target range, and approximately $3cm$ for the converged average robot separation range.

Finally, two experiments aimed to test robustness to robot failure are presented. These experiments take the same form as the previous *target surround* experiment, carried out on real robots, with the exception of one of the robots being a 'failed' robot. For these experiments, 'failed' simply means that the robot is

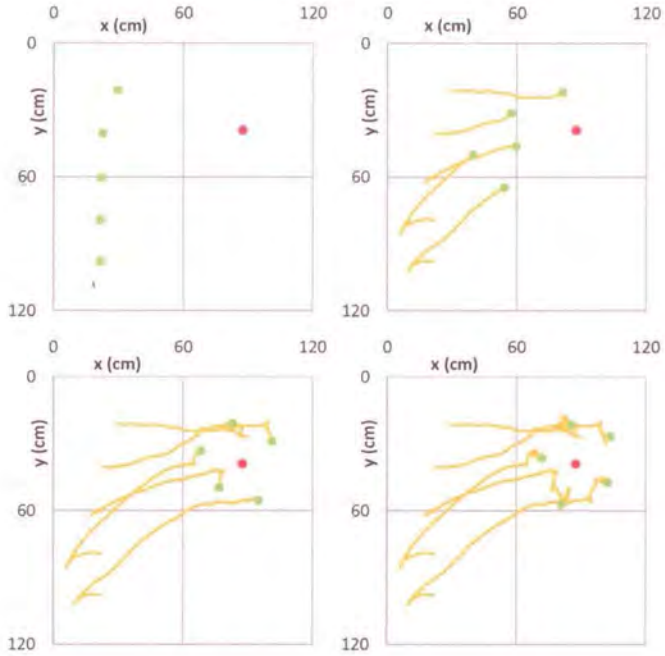


Figure 5.29: Plots showing the positions (green circles) and past trajectories (orange lines) of the robots at various time-steps during the target surround experiment with the real robot set-up. Top left: time-step = 1; top right: time-step = 200; bottom left: time-step = 600; bottom right: time-step = 1000. The target position is given by a red circle.

unable to move correctly, due to e.g. lack of power or damaged motors/wheels.

Figure 5.31 top row, shows plots of the robot positions and past trajectories at the beginning ($timestep = 1$) and end ($timestep = 600$) of the self-organisation process for the target surround experiment with the real robot set-up and one failed robot from $timestep = 1$, with a loose wheel, resulting in the failed robot being able to move only small distances from its starting location. Figure 5.31 bottom row, shows similar plots of the same experiment, with one robot becoming failed (this time powered off) on close approach to the target ($r = 12cm, timestep = 70$). Figure 5.32 shows the average robot to target range and average robot separation range versus time-steps, for the same two failed robot experiments, as well as the same experiment carried out without a failed robot. The results from the failed robot experiments do not include the failed robot in the average calculations once the robot has become 'failed,' to allow for a fair comparison, as the analysis is focussed on the ability of the remaining swarm to function.

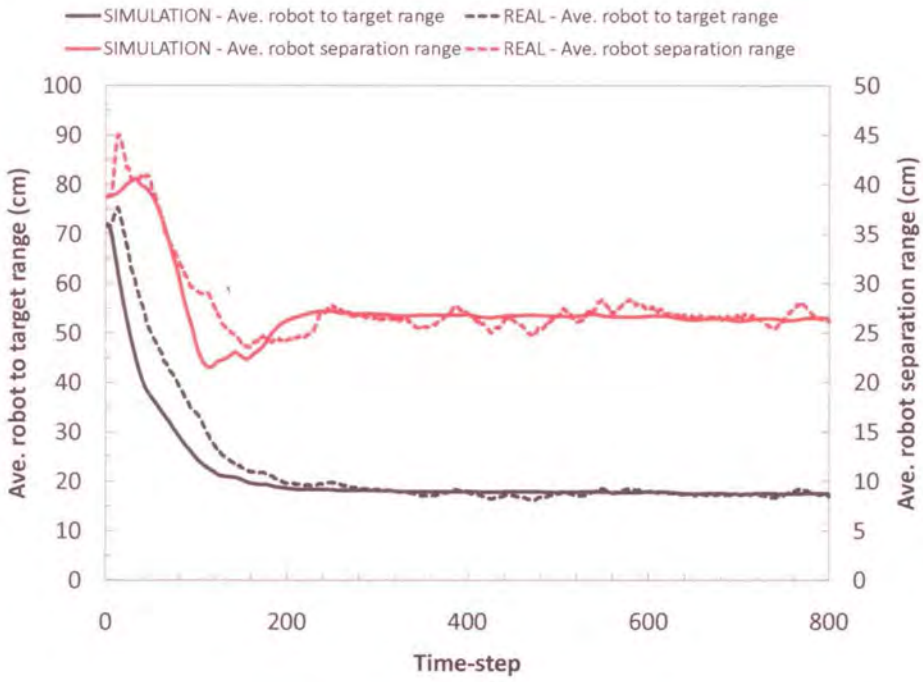


Figure 5.30: Showing the average robot separation distance and average robot to target range, vs. time-steps, for a target surround experiment. The plot shows results obtained from real robots as well as through physics and sensor based simulation in ideal settings.

Immediately we see that the results obtained from with and without a failed robot are comparable, with both sets of results with the failed robot converging to a similar average robot to target range (approximately $17cm$), and average robot separation range (approximately $28cm$), as the results obtained with no failed robot.

For the first of the two failed robot experiments (top row, Figure 5.31) we see that the robot swarm is able to autonomously reconfigure in order to compensate for the absence of the failed robot in the vicinity of the target, providing the optimum surrounding formation with the 4 working robots as opposed to the otherwise 5. By approximately time-step = 400, none of the 4 working robots are within the visible range r_{vis} of the failed robot, and thus are entirely unaffected by its presence from this point onwards. For $timestep < 400$ the 4 working robots are still relatively unaffected by the almost stationary failed robot. This can in part be attributed to the fact that the target force, F_T, is weighted greater than the inter-robot force F_R, and thus the robots are drawn towards to target location

with greater force.

For the second instance of the failed robot experiment (bottom row, Figure 5.31), the failed robot does not become 'failed' until it is close to the target (at range $r = 12cm$). In this case the all of the other robots have to self-organise into formation around the target with the failed robot within visible range, thus being detected as a neighbouring robot. We see from Figure 5.31, bottom right, that the robot starting second from bottom of the plot has to manoeuvre around the failed robot in order to achieve the desired position. This effect can also be seen in Figure 5.32, with the average robot separation range taking longer to converge for the experiment with the failed robot close to the target, due to the failed robot obstructing the path of other robots thus increasing the time taken to self organise.

We note that the difference in starting values for the initial failed robot experiment are due to the fact that the failed robot is the closest to the target at the beginning of the experiment.

5.7 Conclusions and Discussion

This chapter has studied a swarm intelligence inspired approach to pattern formation and control of a swarm of mobile robotic agents. A number of variations of hybrid physics-swarm control laws have been investigated through a number of case study experiments and scenarios.

The use of a relatively simple force law has been investigated, inspired by classical physics and the work presented in [67], for use in low-level automated coordination of a swarm of robotic agents, taking on the form of VTOL UAVs in a simple 2D simulation. It has been demonstrated how such control laws can be used to create a lattice of sensors, to tackle the coverage problem, which, in a case study scenario, was presented in the form of an airborne surveillance application. Performance in relation to the coverage problem has been assessed with respect to the key parameters (R and G) in the presented basic control law using a variation on Newton's law of Gravitation to balance inter-robot virtual forces.

This work demonstrates the possibilities of how methodologies from swarm intelligence can work well in conjunction with classical physics theory. It is envisaged

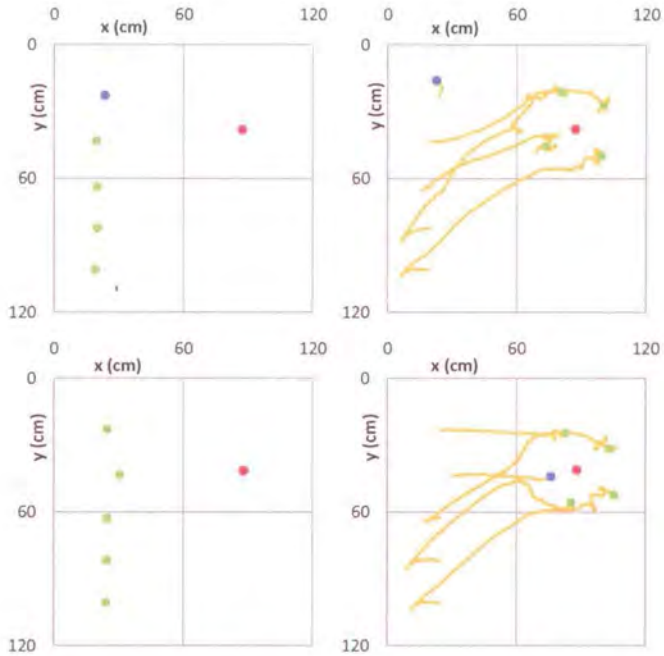


Figure 5.31: Plots showing the positions (green circles) and past trajectories (orange lines) of the robots at various time-steps during two instances of the target surround experiment with one 'failed' robot (blue circle), with the real robot set-up. Top row: 1 robot 'failed' from starting position, left: time-step = 1; right: time-step = 600. Bottom row: 1 robot 'failed' upon close approach to the target, left: time-step = 1; right: time-step = 600. The target position is given by a red circle.

that this hybrid approach will increase the reliability of such a system, allowing for more robust analysis of the performance and workings based on physics laws, while adding the advantages of the swarm intelligence approach in terms of adaptability, distributed control and minimal requirements of agent capabilities [2].

Although success was found in the use of a genetic algorithm for optimising the control law, this was for a relatively simple scenario with a limited search space. This approach to off-line parameter optimisation may be useful for scenarios with specific prior knowledge available, such that the parameters can be optimised for a specific scenario and problem setting. Such off-line parameter optimisation does not however accommodate situations where the scenario/problem setting is changeable mid deployment. The next chapter focuses more on adaptive control laws, developing a method of on-line parameter adaption for distributed pattern

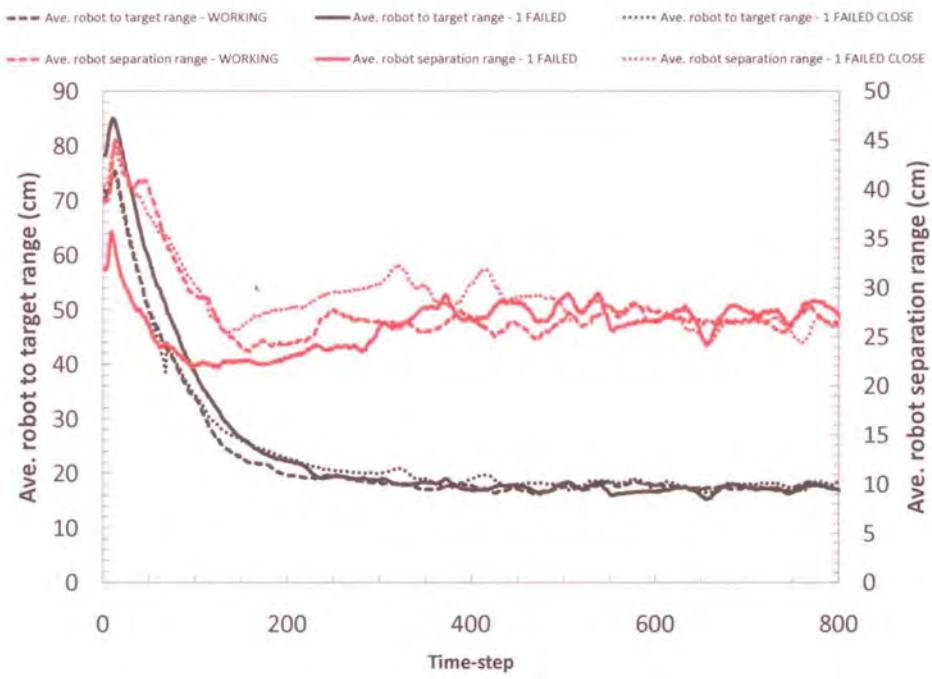


Figure 5.32: Showing the average robot separation distance and average robot to target range, vs. time-steps, for a target surround experiment. The plot shows results obtained from real robots in three instances: firstly with all robots functioning correctly, secondly with 1 failed robot from the start, and thirdly with one failed robot close to the target location.

formation control, where the robots can change their individual parameters in response to changing environment characteristics, similar to the adaptive threshold method developed in the previous chapter, and more akin to observations of social swarms in the wild.

It has been shown how simple indirect communication between individual robots via relative range/bearing sensing can yield distributed self-organising behaviour that can be adapted to facilitate a number of useful behaviours at the swarm level, and furthermore, that a number of similar behaviours can be merged to form a simple yet effective system for tackling a number of problems in the security and defence domain.

Although the presented behaviours could apply to a range of scenarios, reference is made to the counter IED operations as this is undoubtedly an area of high priority with regards to developing new technologies with potential to reduce the

levels of required direct human contact in these processes. The robustness to robot failure and potential expendability of the robots of the system presented in this chapter may be particularly advantageous for tackling the problem of countering IEDs, given the destructive nature of these devices.

Through a number of experiments in physics and sensor based simulation and on real robots, these behaviours have been demonstrated and shown a degree of robustness to the noisy sensor and actuator performance inherent in real robots/hardware. Furthermore, experimentation has demonstrated a level of robustness to robot failure, which is an important property and a key advantage of this approach, and indeed often of the swarm robotics approach in general. The experiments on real robots in particular will help to advance the state-of-the-art in swarm robotics research, providing additional evidence of porting swarm behaviours onto real-world hardware, which is an important factor in advancing this area of research.

There are a few additional points to be made regarding the failed robot experiments. The failed robot in the first instance presented here was not completely failed in that it still had much of its functionality, it was simply unable to move effectively. With the experimental set-up used for this work, if the robot was completely shut-down, this would not have made a significant difference to the results. If however on-board IR range/bearing sensing was being employed, if the robot was completely failed then the other robots would be unable to detect it as a neighbouring robot, rather it would simply become an obstacle to avoid. These different levels of robot failure can have different effects on the overall behaviour of swarm robotics systems [58][59], which is subject to further ongoing investigation.

In transitioning from simple 2D point mass simulation, to physics and sensor based simulation, to embedded simulation on real robots, there has been a number of additional factors to contend with alongside the self-organisational control (such as low-level motor controllers, obstacle/collision avoidance, sensor and actuator noise). Despite these additional complications, the core swarm self-organisation behaviours have required little to no change in implementation. In fact, to varying degrees the proposed methods naturally encompass mechanisms to deal with a number of these complications (for example robustness to sensor and actuator noise, collision avoidance). One of the major challenges in swarm

robotics is being able to exploit these mechanisms given the hardware limitations, and in particular the challenge of maintaining agent anonymity and not using direct communications. A new method to help overcome some of the problems associated with these challenges is the focus of the research carried out in the next chapter.

Chapter 6

Adaptive Algorithms for Cooperative Multi-Robot Coordination and Control

6.1 Introduction

In the previous chapter only fixed parameters were used for the control laws. Although the presented methods exhibited a level of adaptability resultant from the self-organised approach (for example formation self-repair, collective target tracking), the behaviours were demonstrated only for relatively open and unconstrained environments. In this chapter the focus is on adaptive control laws to facilitate improved swarm cohesion in terms of formation creation, maintenance, coordination and control.

In a similar fashion to the adaptive threshold method proposed for the ant-algorithm presented in chapter 4, the focus in this chapter is again on facilitating distributed parameter self-adaptation at the individual agent level, in response to changes in the perceived environment, which have a positive collective effect at the swarm level (cf. Hypothesis 2). The simplistic homogeneous robot swarm is maintained from the previous chapter, where the individual robot agents remain anonymous to one another and cannot communicate directly. The major challenge is thus to create control mechanisms that are the same for each individual robot, but that are capable of adapting for each individual robot's current sta-

tus, based on that robot's perception of it's local surrounding environment, and that collectively produce increased cohesion at the swarm level, without explicit information sharing amongst the swarm.

In this chapter the Virtual Robot Node (VRN) mechanism is proposed, which acts as a type of distributed virtual force management system for the individual robots. The VRN mechanism is used to tackle the 'clustering' problem reported in the previous chapter, when considering multi-robot formation control. In addition, the use of the VRN mechanism is demonstrated for *in*-formation collective movement, similar to what is often referred to as 'flocking' behaviour. Finally, a method of sensor-based dynamic formation adaptation is investigated, for swarms moving through structured environments of changing spatial and geometrical characteristics. Similar to the ant algorithm developed in chapters 3 and 4, the sensor-based dynamic adaptation algorithm uses environment templates to stimulate self-organised pattern adaptation. The use of templates, which is essentially a pattern, or structure, used to construct another pattern, is another nature inspired technique [2].

6.2 Related Work

The use of virtual forces for distributed multi-robot coordination and control and swarm robotics has proved to be a very successful and versatile approach, and consequently much effort has gone into, and continues to go into, developing virtual force based methods for particular tasks, scenarios and functions. This chapter is specifically interested in exploiting the dynamic abilities of such methods as well as facilitating sensor-based self-adaption. Here a brief overview is given of related virtual force/potential field methods which focus on similar adaptive approaches to formation control.

The authors in [135] present a distributed potential fields based method for formation control and maintenance when navigating to a goal location. The method employs several behaviours represented as motor schemas, including a 'maintain-formation' schema which utilises a method based on 'attachment sites,' where robots have around them a number of attachment sites at specific angles, depending on the desired formation, and attractive potentials are generated towards the

nearest site. In [71] a dynamic flocking method is proposed using artificial forces and a decentralised adaptive control mechanism. Through sensing structures in the environment the control mechanism allows the robots to adapt their parameters to shrink and grow the network formation in accordance with the changing environment (the authors give an example of the network tracking a moving target and shrinking to pass through a narrowing in the environment). The proposed method was shown, through computer simulation, to yield promising results. One limitation however is that the proposed method relies on inter-robot communication, thus increasing robot requirements and potentially affecting the scalability of the method. The authors in [136] present an algorithm for achieving segregation in a swarm of mobile robots, based on the Brazil nut effect, which requires the swarm of robots to self-organise into a spatial arrangement in which the robots are ranked by size. The method is verified by physics-based computer simulations and shown to be robust to noise. The initial algorithm does however rely on communication, and although a simplified version is presented which does not rely on communication, this increases the error of the method.

The VRN mechanism developed in this chapter aims to facilitate distributed adaptive formation control without requiring any inter-robot communication, instead using only local sensor-based information. By not resorting to inter-robot communications to achieve cohesive adaptation, the simplistic homogeneous robot swarm requirements are maintained, and moreover, this increases the potential for a more scalable and robust system. Using only indirect communication is also more analogous to swarms in nature, and thus may provide a better platform for harnessing the collective, cooperative behaviours observed in the wild.

6.3 Basic Control Law

In this chapter, the *basic* control law refers to one of the variation on the physics-based control laws explored in the previous chapter. To recap, the basic control law yields the self-organisation of the swarm into a quasi-regular spaced repeating lattice formation, based on the balancing of virtual forces governed by physics laws. This is achieved by each individual robot measuring the range, r, and bearing, θ, to any neighbouring robots $n \in N$, within a given 'visible range', r_{vis}. The

visible range is user-set, however there may also be physical constraints imposed depending on the sensor method used to determine the range to neighbouring robots. The inter-robot force, F_R, experienced due to neighbouring robot, n, is given by:

$$F_{Rn} = \begin{cases} 4\epsilon\left[\left(\frac{R}{r}\right)^{\sigma} - \left(\frac{R}{r}\right)^{\tau}\right] & \text{if } r \leq r_{vis} \\ 0 & \text{otherwise} \end{cases} \tag{6.1}$$

where ϵ is the maximum allowed force, R is the desired separation distance between neighbouring robots, and σ and τ are control parameters. The x and y components are given by: $F_{Rnx} = F_{Rn}cos(\theta_n)$ and $F_{Rny} = F_{Rn}sin(\theta_n)$. The total x and y components of the force experienced are then given by: $F_{Rtotalx} = \sum_{n=1}^{N} F_{Rnx}$ and $F_{Rtotaly} = \sum_{n=1}^{N} F_{Rny}$. A discrete-time approximation is employed such that the resultant x and y components of the velocity are calculated as: $v_x = F_{Rtotalx}\Delta t$ and $v_y = F_{Rtotaly}\Delta t$. From v_x and v_y the new displacement vectors are calculated as: $\Delta x = v_x\Delta t$ and $\Delta y = v_y\Delta t$. The calculated Δx and Δy values give the desired robot displacement at time t, due to the virtual forces acting upon it.

6.4 Virtual Robot Nodes

The basic control law described above (which is explored in more detail in chapter 5) is limited in being only capable of creating certain lattice formations, which are based on a repeating lattice of equilateral triangles. The desired result is however not always achieved, due to, for example the 'clustering' problem [67][96], and the formation topology can be unpredictable. This also makes formation maintenance problematic, especially when dealing with dynamic environments.

A major aim of this chapter is to remove this limitation by introducing a novel method of adaptive formation control, which is achieved by extending the notion of virtual forces to include VRNs. VRNs can be used not only to create specific user defined formations, but can also be used to facilitate self-adapting formation control, as well as sensor directed collective movement.

Any robot can place a VRN within its neighbourhood at a given r and θ, which will then result in the same virtual force being calculated as if there had been

a real robot detected at that range and bearing. VRNs are only visible to the robot that placed them; maintaining the distributed property of the system. The idea is to allow individual robots to manage their own spatial requirements in a reactive, dynamic manner.

6.4.1 VRN Experiments

Experiments are carried out in a physics and sensor based simulation environment, using the Webots professional mobile robot simulation package [133]. Again the e-puck [114] is used as the robotic platform, which is a small-scale differential drive laboratory robot approximately 7cm in diameter.

Each time-step in the experiment is equivalent to approximately $0.1s$ real-time. During each time-step, each robot calculates the range and bearing to all neighbouring robots. Each robot then executes one of the control laws, which will in turn output a displacement vector based on the virtual forces acting upon the robot. The displacement vector then needs to be converted into left and right wheel velocities in order to drive the e-puck robot towards the desired location. The same motor control law used in the previous chapter is used here for this purpose (described in Appendix D).

For a number of experiments the e-puck robot is modified in simulation to include 16 long-range (3m) IR distance sensors around the circumference of the robot (see Figure 6.1), to allow the robot to detect the boundaries of the environment

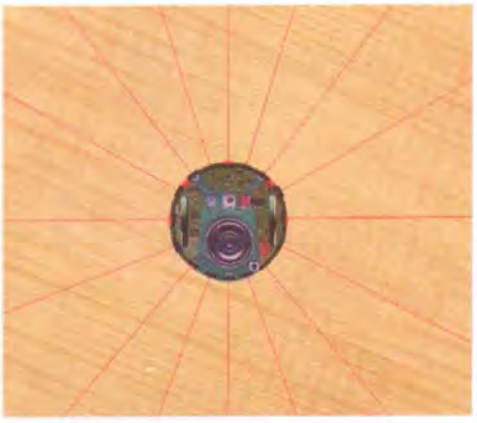


Figure 6.1: A top-down view of the e-puck robot in Webots, modified with 16 long-range distance sensors around the robot's circumference

(similar to the e-puck IR extension board [134][137] and Khepera IR extension board [138]). These distance sensors are positioned above the height of the robot so as to avoid neighbouring robots being detected as part of the environment.

For all experiments reported in this chapter, unless stated otherwise, the control parameters for the basic control law are set with: $\sigma = 0.1$ and $\tau = 0.05$.

The following sections present a number of experiments designed to show the effectiveness of the VRN mechanism and other extensions to the basic control law for a number of multi-robot cooperative coordination and control scenarios. In particular the aim is to show a number of advancements to the existing method of utilising artificial forces to govern robot formation control, and to show that similar adaptive self-organisation to the adaptive threshold of chapter 4 can be achieved in physical robot swarms.

6.4.2 Formation Management

By setting particular conditions and constraints on when and where an individual robot places a VRN it is possible to achieve and maintain a more cohesively structured formation when compared to the methods presented in the previous chapter. Additionally this approach can be used to induce self-organised flocking behaviours. The following section concentrates on the application of the VRN mechanism to the control law detailed in the previous chapter, in order to create and maintain a more cohesive formation, and to alleviate some of the limitations of the original control law (specifically the clustering problem).

The robots neighbourhood is divided into J segments, $\Psi_j \in J$, with each segment defined by a radius of r_{seg} and two bounding bearings ψ_a and ψ_b (See Figure 6.2, left). When the robot obtains the information regarding the range, r, and bearing, θ, of any neighbouring robots, the robot checks which segments contain neighbouring robots. Those segments which do not contain any neighbouring robots receive a VRN at range r_{VRN} and bearing $\theta_{VRN} = (\psi_a + \psi_b)/2$, such that:

$$VRN(j) = \begin{cases} FALSE & \text{if any } \theta_{n \in N} \in \Psi_j \\ TRUE & \text{otherwise} \end{cases} \tag{6.2}$$

In order to achieve a cohesive repeating equilateral triangle lattice the robots

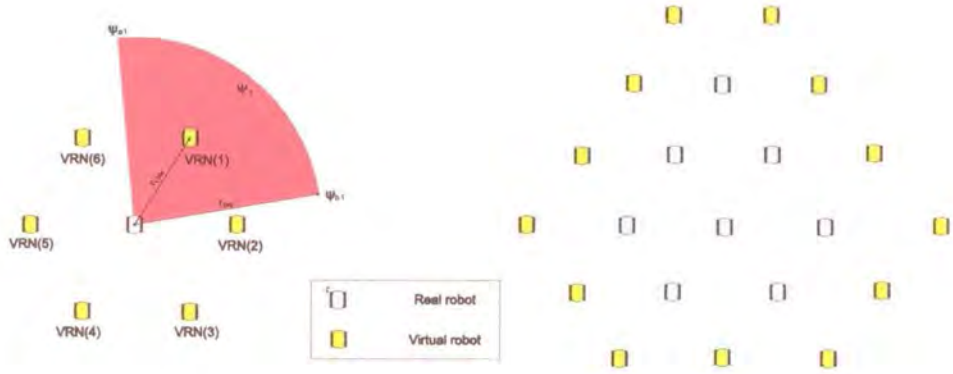


Figure 6.2: Left: A schematic diagram of the VRN positions and segments; right: a schematic diagram of the real robot and VRN positions for a double wedge formation

neighbourhood is divided into $J = 6$ segments, with $\theta_{VRNj} = j\pi/6$, $\psi_{aj} = (j\pi/3 - 4\pi/9)$ and $\psi_{bj} = (j\pi/3 - \pi/18)$.

Robots on the periphery of the lattice thus create a VRN boundary around the outer region of the current formation, which effects the overall shape of the formation, in this case, with 8 robots, resulting in a double wedge formation as depicted in Figure 6.2, right.

To demonstrate this functionality, attention is drawn to a limitation of the basic control law described in the previous chapter. Given 7 robots the resultant formation would be a well formed hexagonal lattice with a robot at the centre, maintaining the repeating lattice of equilateral triangles. If there were 8 robots however, the resultant formation would be less cohesive, with one robot 'trapped,' resulting in a less uniform lattice, not resembling a repeating lattice of equilateral triangles. This occurs due to the 'clustering' problem, as reported in [67][96]. The clustering problem is where groups of robots remain clustered together, with $r < R$. This phenomenon is known to be caused by local minima of the inter-robot forces, for example when a robot is surrounded by a number of robots such that the resultant force is not sufficient for it to escape, and that robot's influence on the surrounding robots is not sufficient to cause them to move out of the way.

An experiment is carried out in physics and sensor based simulation with 8 robots starting in a pseudo-random cluster (Figure 6.3(a)), running the basic formation control law described above, with $R = 50cm$, $\epsilon = 20.0$ and $r_{vis} = 3R/2$. The

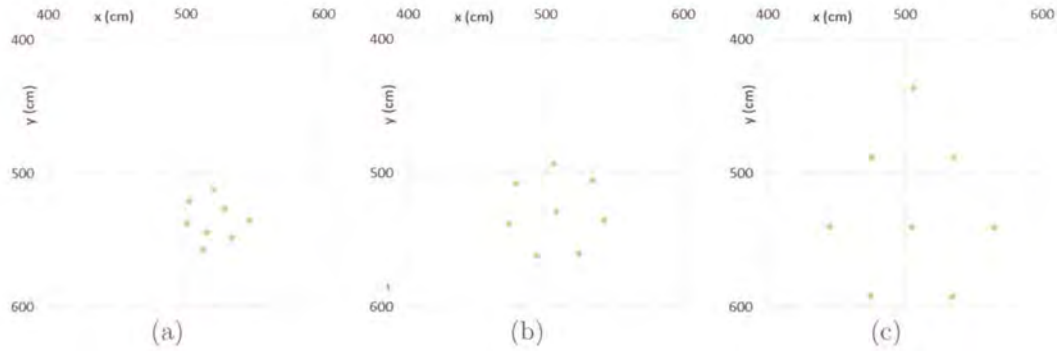
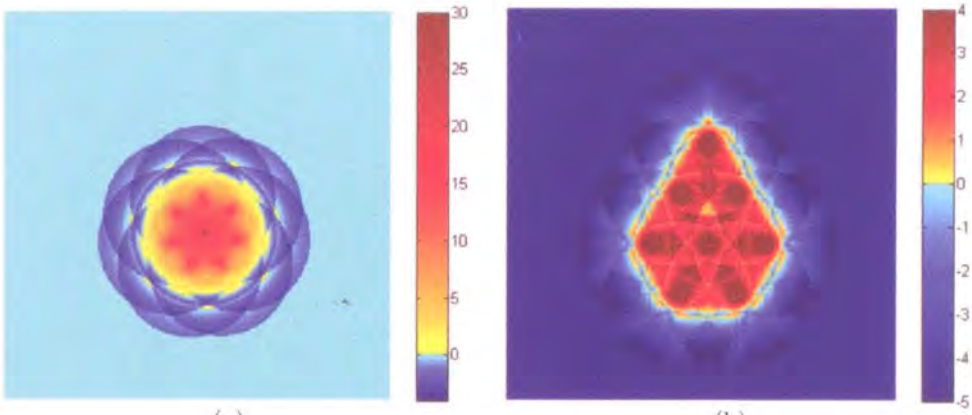


Figure 6.3: Plots showing the positions (solid green circles) of the robots in (a), starting positions; (b), the final positions from running the basic formation control law; (c), the final positions from running the basic formation control law with the addition of the VRN mechanism.

resultant lattice formation can be seen in Figure 6.3(b). Although the robots have manoeuvred from their initial positions into a formation, there is one robot 'trapped' in the centre of the formation, preventing the swarm from achieving the desired repeating equilateral triangle lattice.

The use of the VRN mechanism enables each of the robots on the periphery of the lattice to behave as if there were another layer of robots beyond them, causing the formation to expand into the desired repeating equilateral triangle lattice, making a double wedge formation. Repeating the same experiment with the VRN enabled control law with $r_{VRN} = 60cm$, we observe a final lattice formation with greater geometric cohesion, showing near uniform inter-robot separation (Figure 6.3(c)). The VRN mechanism is shown here to overcome the clustering problem in a single merged behaviour, by introducing additional attractive forces expanding the robots on the periphery of the formation outwards, overcoming the cumulative attractive inwards force towards the centre of the cluster. Figure 6.4 compares the potential fields around the robots corresponding to the formations in Figure 6.3(b) for the basic control law and Figure 6.3(c) for the VRN control law. We can see the presence of the additional attractive forces (negative potential) around the formation with the VRN control law, allowing the formation to expand to achieve quasi-regular inter-robot spacing.

Figure 6.5 shows a plot of the error in average minimum inter-robot distance (calculated against the desired inter-robot distance of 50cm) for both the forma-



(a) (b)

Figure 6.4: Showing the potential fields around 8 robots after self-organisation for (a) the basic control law, and (b) the VRN control law (corresponding to the formations in Figure 6.3(b) and 6.3(c) for the basic and VRN control laws respectively).

tion experiments. As expected, we indeed see the VRN control law yielding a smaller error as the formation converges, in comparison to the basic control law. The VRN method does however introduce an error of its own, due to the static r_{VRN} causing a constant attractive force on the periphery robots expanding the formation beyond its equilibrium point. This results in the resulting formation having an average $r > R$. As can be seen in Figure 6.5, this error is however less than the error due to the clustering problem, and the resultant formation is much more cohesive in terms of formation structure.

Figure 6.6 gives a comparison between the VRN and basic control laws for formation control with varying numbers of robots, showing the average Standard Deviation (SD) of minimum inter-robot distance (left) and the average error in average minimum inter-robot separation distance (right). The experiments are run with $R = 50cm$, for $N = 4, 8$ and 16 robots. For each method and for each set number of robots, the experiments are run 10 times, changing the pseudo-random starting positions each time.

Figure 6.7 shows corresponding plots of the robot positions post self-organisation, showing typical converged lattice formations. By visual inspection we notice that for each instance the robots have created a lattice formation with quasi-regular inter-robot separation distance. We can immediately notice the difference in

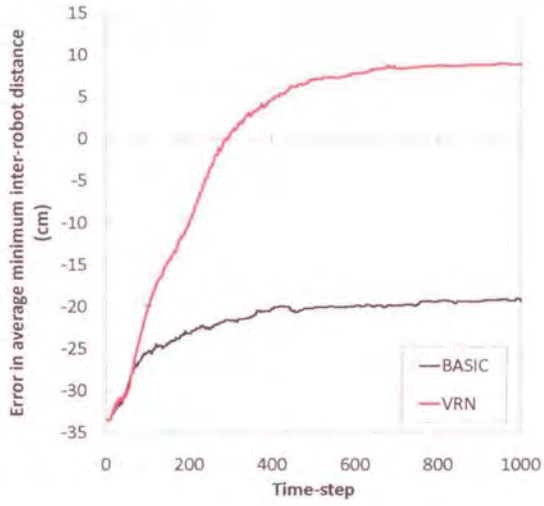


Figure 6.5: Results for the VRN formation comparison experiment, showing a typical plot of the error of the average minimum inter-robot distance versus time-steps, for both the basic and VRN enabled control laws in physics and sensor based simulation. The error is calculated against the desired inter-robot distance of 50cm. The corresponding positions plots are given in Figure 6.3.

the separation distance achieved between the VRN and basic method, which reflects the average results shown in Figure 6.6. We can also see a more defined structure to the VRN lattice formations, with Figure 6.7(b,c) clearly exhibiting the repeating equilateral triangle lattice that one might intuitively expect. On the contrary, we can clearly see the 'clustering' phenomenon occurring in Figure 6.7(e,f) with the basic control law.

For each N value we see that the VRN method has yielded lower errors in comparison to the basic method. Again the errors present in the basic method are mainly due to the 'clustering' problem, as mentioned previously, with the VRN method overcoming this problem. Figure 6.6 also shows the positive error present with the VRN method to be less than the error due to the clustering problem, over the range of N.

It should also be noted that the difference in error for increasing N with the VRN method is relatively small, showing a level of scalability for small groups of robots. The basic method however shows a significant increase in error with increasing N (approximately $+10.7cm$ between $N = 4$ and $N = 8$ and approximately $+6.1cm$ between $N = 8$ and $N = 16$).

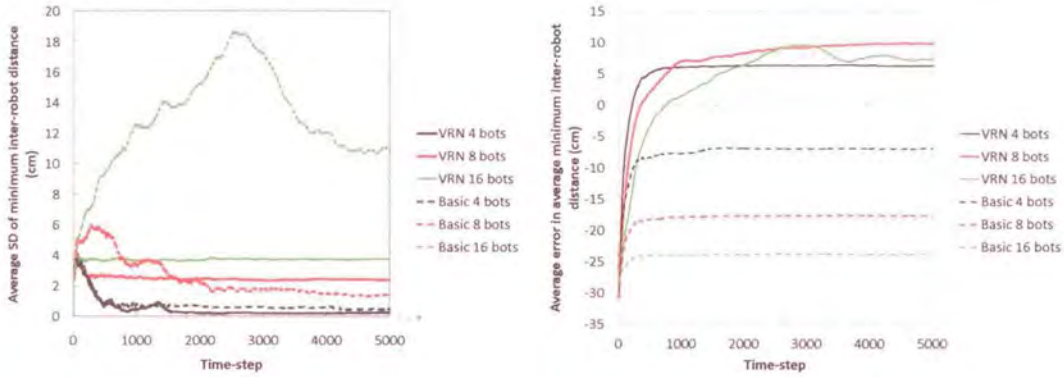


Figure 6.6: A comparison between the VRN and basic control laws for formation control with varying numbers of robots. Right; showing the average error in average minimum inter-robot separation distance, and left; showing the average S.D. of inter-robot separation distance. Results averaged over 10 experiments.

The scalability is investigated further, considering a swarm size up to $N = 800$, in simple 2D point mass simulation, using the Player/Stage simulation software. Figure 6.8 gives examples of the formation achieved after 20000 time-steps, for $N = 200$, $N = 400$ and $N = 800$, comparing the basic method with the VRN method, with $R = 30cm$. From visual inspection it is immediately clear that the basic control law has suffered greatly from the 'clustering' problem at these larger swarm sizes, and has failed to expand into the desired formation. The VRN control law (running with $r_{VRN} = 40$) does manage to expand the swarm, and from visual inspection there appears to be reasonable levels of cohesion. In Figure 6.8(a), for $N = 200$, we observe that the swarm has achieved quasi-regular spacing in most areas of the formation, but not uniformly across the entire formation. With increasing N we see that this observation holds.

In Figure 6.9 the corresponding error in average minimum inter-robot separation distance is shown. In correspondence to the observations of Figure 6.8 we see a large negative error for the basic control law (which, in similar fashion to Figure 6.6, increases with increasing N). An important observation of the VRN control method is that the curves have not reached a plateau even after 20000 time-steps, which suggests that the formation has still not fully converged. So although the VRN method does significantly reduce the clustering effect, for large N the scalability is affected by convergence time.

Although in this chapter the 'basic control law' refers to the Lennard-Jones imple-

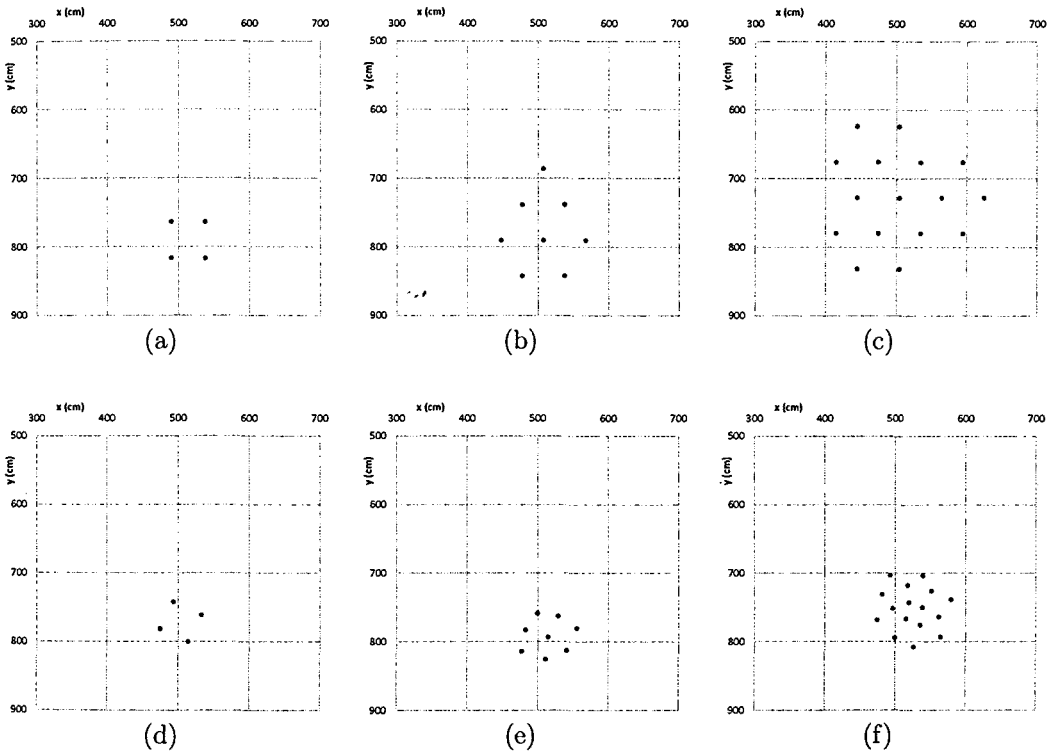


Figure 6.7: Plots showing final robot positions for a comparison between the VRN and basic control laws for formation control with varying numbers of robots. (a), $N = 4$ VRN control law; (b), $N = 8$ VRN control law; (c), $N = 16$ VRN control law; (d), $N = 4$ basic control law; (e), $N = 8$ basic control law; (f), $N = 16$ basic control law.

mentation as defined above, for comparison purposes Figure 6.10 shows formation results using the split-Newtonian implementation of the basic control law (from chapter 5). As a reminder, the split-Newtonian control law gives,

$$F = \frac{Gm_i m_j}{r^2} \tag{6.3}$$

with the following conditions,

$$\text{for} \quad r > R, \quad F \rightarrow -ve \tag{6.4}$$

$$\text{for} \quad r \leq R, \quad F \rightarrow +ve \tag{6.5}$$

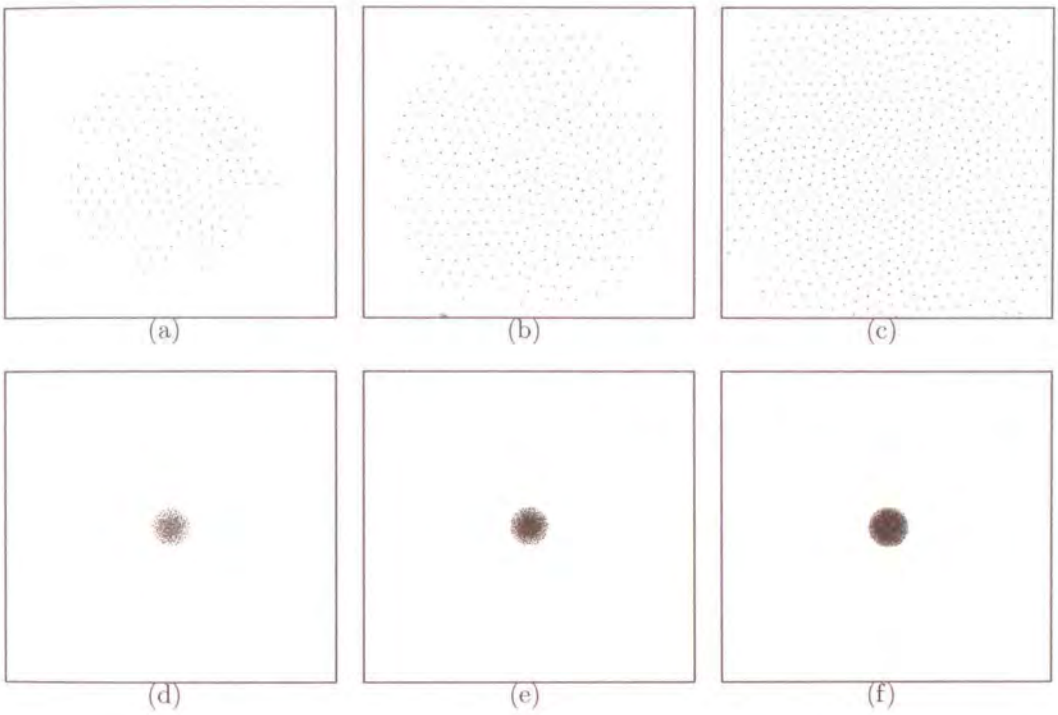


Figure 6.8: Images showing final robot positions for a comparison between the VRN and basic control laws for formation control with varying numbers of robots. (a), $N = 200$ VRN control law; (b), $N = 400$ VRN control law; (c), $N = 800$ VRN control law; (d), $N = 200$ basic control law; (e), $N = 400$ basic control law; (f), $N = 800$ basic control law. Note that the scales are the same for each sub-figure.

where r is the measured inter-robot separation distance, G is the gravitational constant and m is the mass.

For the experiments shown in Figure 6.10 the same parameters were used as for the Lennard-Jones and VRN methods above, additionally with $G = 200$ and $m = 1$.

Like with the Lennard-Jones basic control law, and consistent to what was seen in the previous chapter, Figure 6.10 shows the split-Newtonian method to also suffer from the common clustering problem.

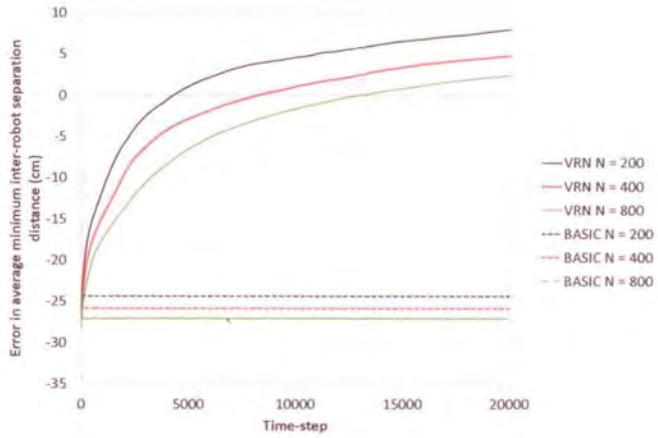


Figure 6.9: A comparison between the VRN and basic control laws for formation control with varying large numbers of robots, showing the error in average minimum inter-robot separation distance.

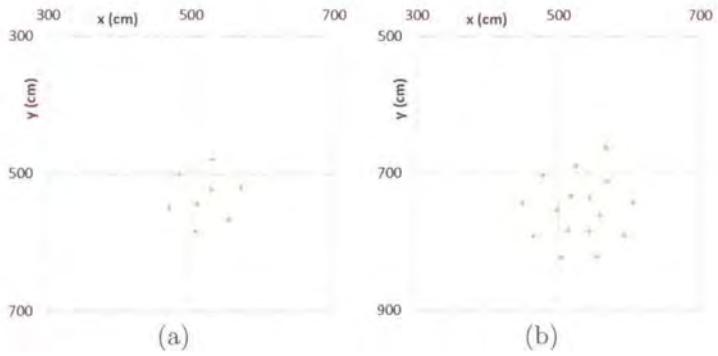


Figure 6.10: Plots showing the positions (solid green circles) of the robots in the formation created with the split-Newtonian method for (a), 8 robots and (b), 16 robots.

6.4.3 Flocking

Using the VRN mechanism, only a simple additional constraint is required on the formation management in order to induce directed movement of the robots *in*-formation, yielding a simple flocking behaviour.

The constraint is such that any robot or VRN contributing to the movement force (Equation 6.1) which lies at a bearing greater than $\pm\pi/2$ *rads* to the desired flocking direction contributes a reduced force. Formally this can be written:

$$F_{Rn} = \begin{cases} F_{Rn} & \text{if } \theta_f - \pi/2 \leq \theta \leq \theta_f + \pi/2 \\ F_{Rn} \cdot \gamma & \text{otherwise} \end{cases} \tag{6.6}$$

where θ_f is the desired flocking bearing, and γ is the reduction factor.

Flocking experiments are carried out as a dynamic extension to formation control, where in addition to creating a lattice formation, the robots also have to move as a collective, or 'flock.' This creates an significant additional challenge to the individual agents, of being able to maintain a cohesive formation while moving as a collective swarm entity, with only limited local knowledge of the swarm, and no direct communication between agents.

To analyse the VRN flocking behaviour a number of experiments are carried out implementing the constraint of Equation 6.6 into the formation management (again in physics and sensor based simulation). Figure 6.11 reports plots of the robot positions and past trajectories for two similar flocking experiments, with $N = 5$ and $N = 20$ robots respectively.

For each experiment there is implemented a given θ_f, identical for each robot, which is changed at specified points in the experiment, to induce a number of direction changes. As with previous experiments, the robot starting positions take the form of a tight pseudo random cluster.

From visual inspection of Figure 6.11 we observe that the swarm effectively moves as a collective, with a common heading and in a formation with quasi-regular inter-robot separation distance. These observations hold true for both instances of the experiment, with $N = 5$ and $N = 20$. The corresponding plots of the potential fields around to robots (with left, right and upwards flocking orientation bias) are given in Figure 6.12. The constraint of Equation 6.6 effectively changes the potential field to give an attractive bias in the desired flocking direction, while maintaining formation structure.

The plots in Figure 6.13 give the error in average minimum inter-robot distance and SD of robot orientations corresponding to the experiments shown in Figure 6.11. For an effective flocking behaviour, it is desired that the swarm forms and maintains a cohesive formation, while moving as a collective in a unified direction. As the error in minimum inter-robot separation distance reaches a plateau (treating the small fluctuations as minor adjustments by the individual

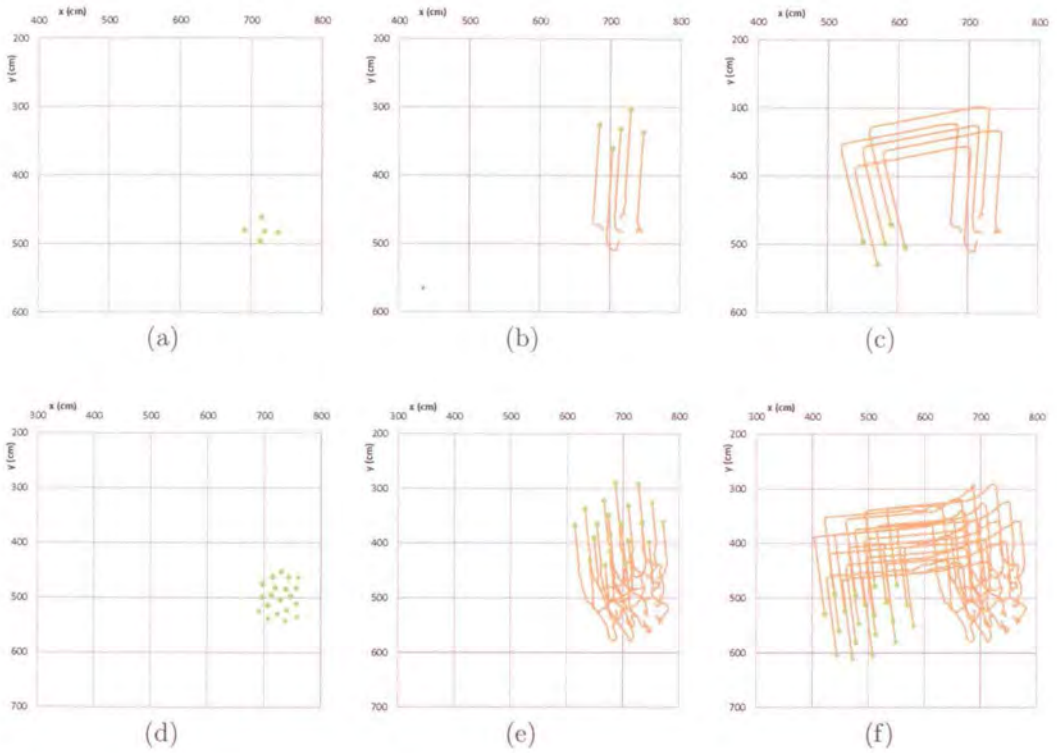
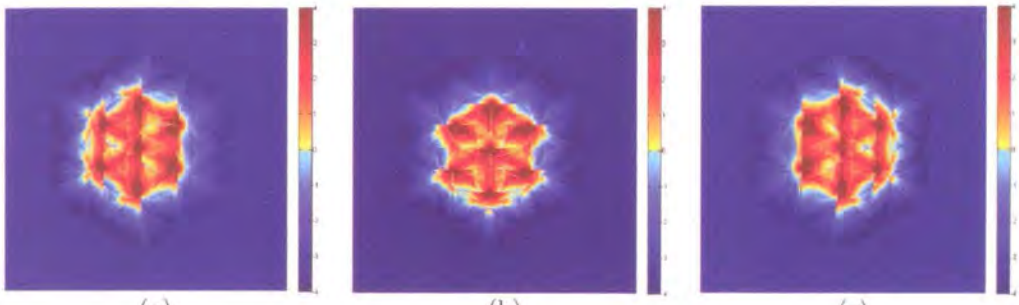


Figure 6.11: Plots showing robot positions (solid green circles) and past trajectories (orange lines) at different times, for robots executing the VRN flocking behaviour. (a) $N = 5, t = 1$; (b) $N = 5, t = 400$; (c) $N = 5, t = 1200$; (d) $N = 20, t = 1$; (e) $N = 20, t = 400$ and (f) $N = 20, t = 1200$.

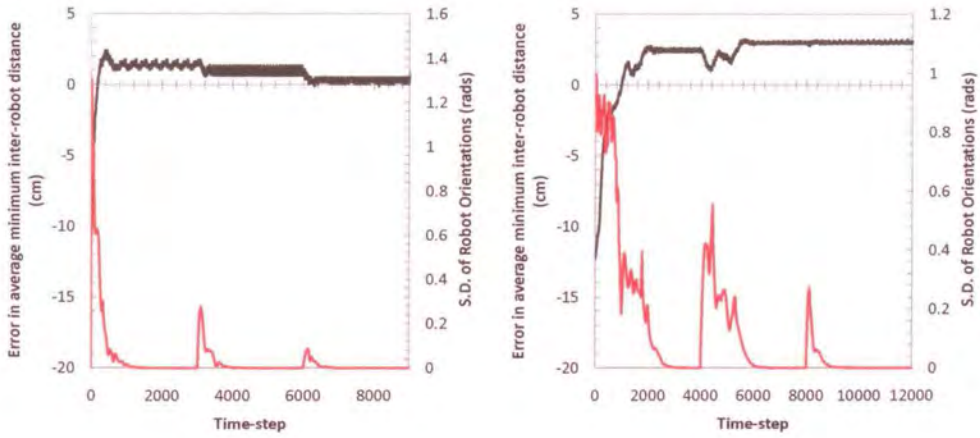
robots), this corresponds to to swarm reaching a cohesive formation. We see a similar trend here to the formation experiments (Figure 6.6), with a large gradient in the error curve as the swarm self-organises into formation from the initial pseudo-random starting positions. We also observe a sharp fluctuation at $t = 3000$ and $t = 6000$ where the orientation of the swarm is changed. We observe however that the error curve promptly returns to a plateau, showing rapid response to the change in direction.

For strong cohesion while moving, it is desired that all members of the swarm have the same orientation, with a SD of robot orientations close to zero. From Figure 6.13 we observe the SD of orientation to converge to zero from $t = 0$ as the swarm self-organises into formation and begins to move in the desired direction as a collective. We again see fluctuation at $t = 3000$ and $t = 6000$ as the direction



(a) (b) (c)

Figure 6.12: Plots showing the potential field around 7 robots using the VRN flocking control law, for (a) a right bias; (b) an upwards bias and (c) a left bias direction.



(a) (b)

Figure 6.13: Plots of the error in average minimum inter-robot distance (black curve, left axis) and S.D. of robot orientations (red curve, right axis) for two instances of a flocking experiments with (a) $N = 5$ robots and (b) $N = 20$ robots.

is changed, and the individual robots have to adjust to the new desired direction, while maintaining formation.

Figure 6.14 shows results from additional flocking experiments with larger N (carried out as point mass simulations in Player/Stage). The results are for similar experiments to those reported in Figure 6.13, with a change in direction at $t = 20000$ and $t = 40000$. A similar trend in the error in average minimum inter-robot distance is seen as the size of the swarm is increased, with only small fluctuations in error as the flocking direction is changed. The fluctuations

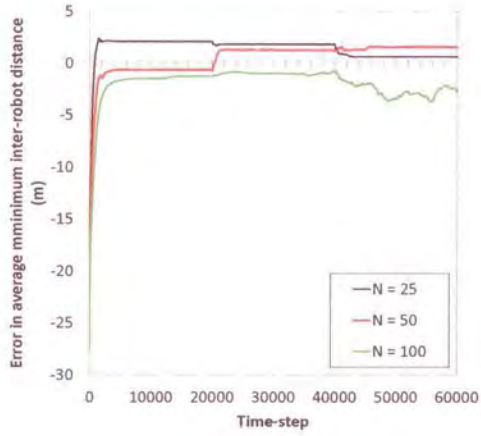


Figure 6.14: A plot of the error in average minimum inter-robot distance for a flocking experiment, with increasing N.

in error are slightly greater with the largest swarm size (of $N = 100$), as the self-organisation process takes longer with larger numbers of agents. This same phenomenon was observed with static formations (as shown in Figures 6.8 and 6.9). Although these are simple point mass simulations, these results show potential for scalability of the VRN approach to flocking behaviours. However again it should be noted that with the increase in robots to large N, although the desired flocking behaviour is still achieved, due to the larger number of robots, the initial formation takes longer to achieve from the initial pseudo-random starting positions, as many of the individual robot agents have further to move before a cohesive formation is achieved, and hence the larger fluctuations observed in error in average minimum inter-robot distance for large N.

This method of flocking is made possible by the presence of VRNs on the periphery of the formation, facilitating a virtual potential field localised around the swarm, which can be manipulated locally to create cohesive collective movement (with Equation 6.6). In comparison to the original flocking approach proposed by Reynolds [4], the VRN approach has the advantage that it does not require communication of velocity vectors between the individual agents. The velocity matching occurs as a consequence of resolving the inter-robot and robot-VRN virtual forces. One important point to make however is that in the flocking experiments above, it was assumed that a common bearing was known to all agents. In reality, this may not necessarily be achievable (unless for example each robot was receiving this information from a common external communications source).

For the VRN flocking method to work, a common heading is required, and this remains an open problem with the VRN approach as well, unless explicit robot communication is assumed, like with other existing flocking methods.

6.4.4 Dynamic Directed Movement Behaviour

This behaviour extends the basic control law to enable sensor driven dynamic, collective movement through a given environment. To facilitate this behaviour the e-puck robots are modified in the Webots simulation environment, for use in physics and sensor based simulation by providing each robot with 16 additional range finding sensors (equivalent to long-range IR, or laser), with approximate equal spacing around the circumference of the robots outer body, to provide 360 degree sensing of a walled environment. The sensors are placed above the height of the robots to avoid the main body of neighbouring robots being detected as the environment (see Figure 6.1).

Any robot that does not detect any real robots ahead of its current position (with respect to the forwards direction through the environment), considers itself a *leading* robot. This means that instead of attempting to maintain formation the robot attempts to move through the centre of the environment, guided by measurements taken from the on-board distance sensors. The robot will place a VRN with a forwards/left/right bias with respect to its current heading, to induce an attractive force in the desired direction. Any given robot will not know whether or not a neighbouring robot is a leading robot or not, maintaining the anonymous property of the system. Any robot not leading will dynamically adapt its desired inter-robot distance R with respect to measurements taken from the on-board range finders. A pseudocode overview of the dynamic directed movement process with the VRN mechanism is given in Algorithm 6.1.

In the most basic set-up, using only the 'basic' formation control law as described in section 6.3 and in chapter 5, the setting of $\xi(t) = range_{min}(t)$ is used, where $\xi(t)$ represents the robots perceived range to the wall at time t, and $range_{min}(t)$ is the minimum range measured from the on-board distance sensors at time t.

The following conditions are then set to adapt the robot's R value:

$$R(t) = \begin{cases} R(t-1) - 1 & \text{if } \xi(t) < (R(t-1) - (R(t-1)/\chi)) \\ R(t-1) + 1 & \text{if } \xi(t) > (R(t-1) + (R(t-1)/\chi)) \end{cases} \tag{6.7}$$

where χ is a sensitivity weighting parameter.

Adapting the R value in this way allows the swarm to dynamically expand and contract with changing geometric environment characteristics as observed by on-board sensors. The resultant movement is however somewhat non-cohesive, with no specific formation being maintained.

In order to achieve a more cohesive swarm movement, with the ability to achieve and maintain a specific formation, the VRN mechanism is implemented (as described above), and additional constraints are added on calculating the dynamic R value. From the VRN segment scanning, each robot estimates its current position with respect to the desired formation, and adjusts ξ accordingly, in order to maintain formation within the constraints of the environment. This is implemented with a simple rule-set, as follows:

$$\xi(t) = \begin{cases} range_{min}(t) & \text{if } VRN(1,2,3) = TRUE \\ range_{min}(t) & \text{if } VRN(4,5,6) = TRUE \\ range_{min}(t) - R(t) & \text{if } VRN(1,2,3,4,5,6) = FALSE \\ range_{min}(t) - (R(t)/2) & \text{otherwise} \end{cases} \tag{6.8}$$

where $VRN(n) = TRUE$ if a VRN is placed in segment n, and is $FALSE$ otherwise.

Algorithm 6.1: Pseudo-code for the *VRN Dynamic Directed Movement* behaviour.

```
Calculate neighbouring robot relative bearings
Decide if a leading robot
Replace any gaps in the immediate surrounding lattice with VRNs
Dynamically change R w.r.t range_min and current position in the formation
Set r_VRN = R + (R - r_min)
if leading robot then
    Ignore neighbouring robots and VRNs
    Calculate position w.r.t. centre of passageway, adjust left/right/forwards bias accordingly
    Add directional VRN
    Execute control law
else
    Calculate 6 nearest neighbours (including VRNs)
    Compute and resolve forces
    Execute control law
end if
```

These constraints allow for the offset of $range_{min}$, depending on where the robot is with respect to other robots in the formation, and with respect to the walls of the environment These simple rules essentially allow the robot to adapt R with respect to both the robots perception of the local environment structure, as well as the robots perception of the surrounding formation structure, with the aim of consequently increasing cohesion of the formation throughout the adaptive process.

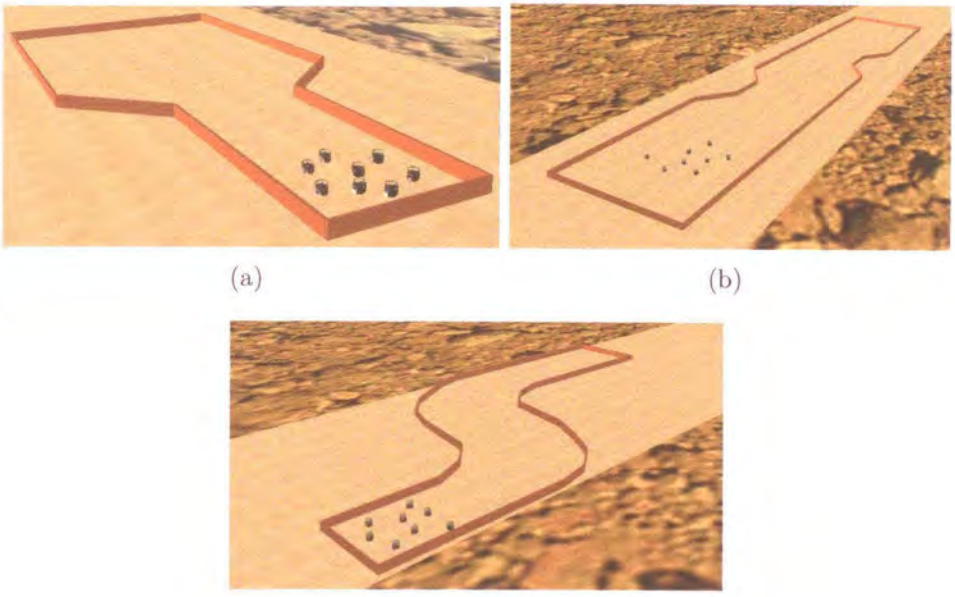
The VRN range is set as $r_{VRN} = R + (R - r_{min})$, where r_{min} is the minimum range to a neighbouring robot, to enable a consistent attractive force due to the VRNs throughout the adaptive process.

To further improve the swarm cohesion, inspiration is taken from recent observations in nature of the well established *flocking* phenomenon. In recent years it has been shown [139] that when birds in the wild exhibit flocking behaviours, each bird interacts on average with a fixed number of neighbours (six of seven), rather than with all neighbours within a fixed radius. A similar behavioural characteristic is implemented here by imposing each robot to choose the nearest 6 robots and/or VRNs to compute as neighbouring robots for inclusion in the control law.

The resultant behaviour allows the swarm lattice to self-organise into and maintain a cohesive formation and dynamically expand and contract with the changing geometric characteristics of the environment, while collectively moving along the passageway. The movement of the leading robot causes a chain reaction through the rest of the swarm as neighbouring robots are attracted to those moving away, along the passageway, with the VRNs maintaining a cohesive formation.

To test the dynamic directed movement extension to the proposed control law, along with the VRN mechanism, a number of experiments are considered where the challenge is for the robot swarm to collectively move through a walled passageway, distributing dynamically with regards to the changing width of the passageway (considering application scenarios such as structure inspection and route clearance), while maintaining maximum cohesion within the formation. A number of different environment scenarios are considered including; a clearing, a chicane and an 's-bend' corridor (see Figure 6.15).

First considered is the 'clearing' scenario, where the simulation environment in



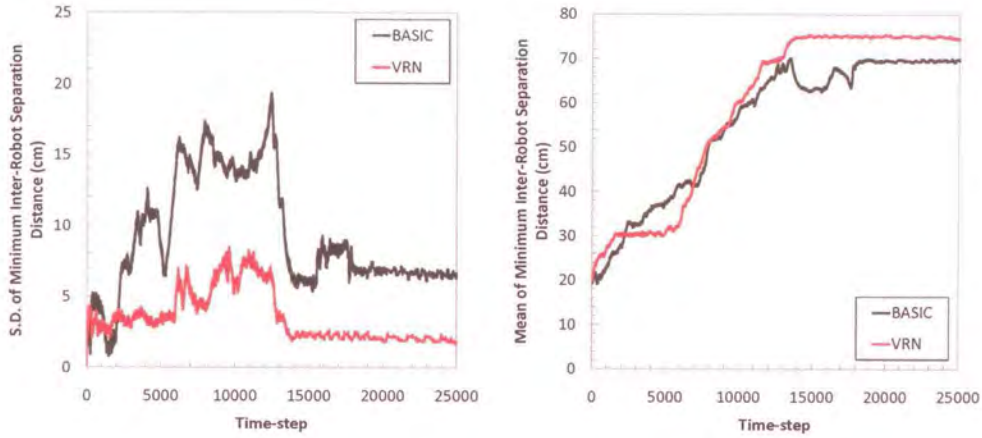
(a) (b)

(c)

Figure 6.15: Snapshots of the corridor environments in the Webots simulator, for (a) a clearing scenario; (b) a chicane scenario and (c) a s-bend scenario.

this experiment consists of a narrow walled passageway which opens into a wider passageway (see Figure 6.15(a)). The robots start in a pseudo-random cluster in the narrow section, and move into the wider section. The relevant parameters for this set of experiments are set with $\epsilon = 20.0$, $r_{vis} = 3R/2$ and $\chi = 3$, demonstrating different dynamic responses due to different environment characteristics, using the same key parameter settings.

Figures 6.16 and 6.17 shows example results from a typical experiment run for both the basic control law and VRN enabled dynamic directed movement algorithm, for the 'clearing' environment. In Figure 6.17(e-h), we see the robot positions and trajectories throughout the experiment with just the basic control law. We see that the robot swarm does effectively adapt to the increasing width of the environment, however the swarm does not appear to maintain any specific formation throughout. Figure 6.17(a-d), shows results of the same experiment with the VRN enabled control law. Again we see the swarm effectively adapting to the changing width of the environment, and we also see in this case that the swarm maintains a double wedge formation after forming this lattice from the initial pseudo-random starting positions and adapting to the increase



(a) (b)

Figure 6.16: Plots of (a) the SD of minimum inter-robot separation distance and (b) the mean of the minimum inter-robot separation distance, for the 'clearing' dynamic collective movement experiment.

in environment width.

Figure 6.16(a) shows a plot of the SD of the minimum inter-robot separation distance measured by each robot, versus time-steps, comparing the basic and VRN enabled control laws, for the 'clearing' experiment. We would expect the SD to be small when the robots are in a cohesive formation, with similar inter-robot separation distances (with an ideal formation having equal inter-robot distances and thus a SD of zero).

Firstly we observe that the VRN experiment exhibits a lower SD for the majority of the experiment. The area of relatively large peaks seen for both curves between approximately 6000 and 13000 time-steps is the region where the swarm is adapting to the increase in environment width. The experiment shows the basic control law to yield a larger increase in SD during this self-adapting process, suggesting a less cohesive collective behaviour, as observed in Figure 6.17.

Figure 6.16(b) shows, for the 'clearing' experiment, a plot of the mean of the minimum inter-robot separation distance measured by each robot, versus time-steps, comparing the basic and VRN control laws. For the VRN curve, the plateau between approximately 1500 and 6000 time-steps corresponds to where the swarm has self-organised into the double wedge formation (as can be seen in Figure 6.17(b)), and is moving along the narrow section of the environment

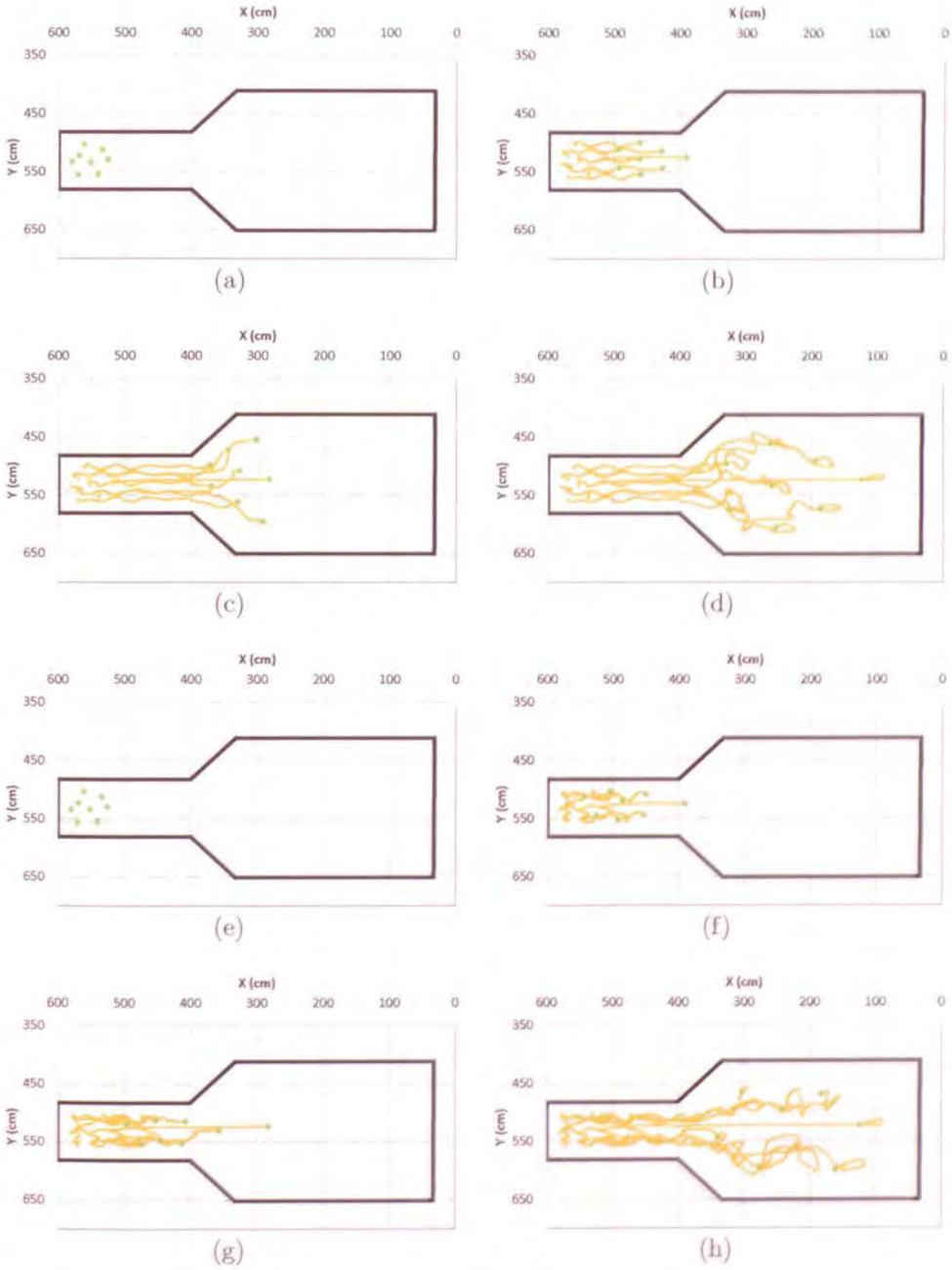


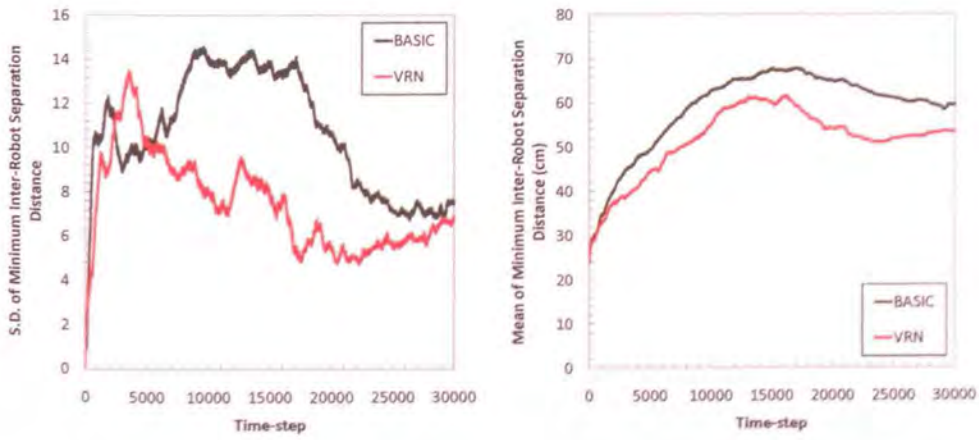
Figure 6.17: Plots showing robot positions at increasing timesteps for a comparison between the VRN and basic control laws for adaptive formation control, for the clearing experiment. (a) VRN at 1 timesteps; (b) VRN at 4000 timesteps; (c) VRN at 7000 timesteps; (d) VRN at 25000 timesteps; (e) basic at 1 timesteps; (f) basic at 4000 timesteps; (g) basic at 7000 timesteps; (h) basic at 25000 timesteps.

before the wider section. This correlates to the maintained low SD throughout this section as seen in Figure 6.16(a).

For the basic control law, the mean of the minimum inter-robot separation distance continues to increase from 0 to approximately 13000 times-steps, during which period the SD is seen the fluctuate to comparatively high values. This can be attributed to the observation that for the basic control law, the swarm does not achieve a cohesive formation from the initial starting positions. As can be seen in Figure 6.17(f), the robot positions show a non-cohesive formation expanding lengthways along the narrow section of the environment.

As the curves reach a plateau at 14000 and 18000 time-steps for the VRN and basic control laws, respectively, this corresponds to the final formations reached as the swarms have expanded into the larger section of the environment (Figure 6.17(d), (h)). We observe that the VRN control law has maintained a double wedge formation after the adaptive process, and achieved a final SD of approximately $0.69cm$. The swarm under the basic control law successfully manages to expand with the changing environment, but the final formation appears less cohesive from visual inspection, with a final SD of approximately $7.1cm$.

Figure 6.18 shows similar plots of SD and mean of minimum inter-robot separation distance for the 's-bend' experiment, with corresponding simulation snap-



(a) (b)

Figure 6.18: Plots of (a) the SD of minimum inter-robot separation distance and (b) the mean of the minimum inter-robot separation distance, for the 's-bend' dynamic collective movement experiment.

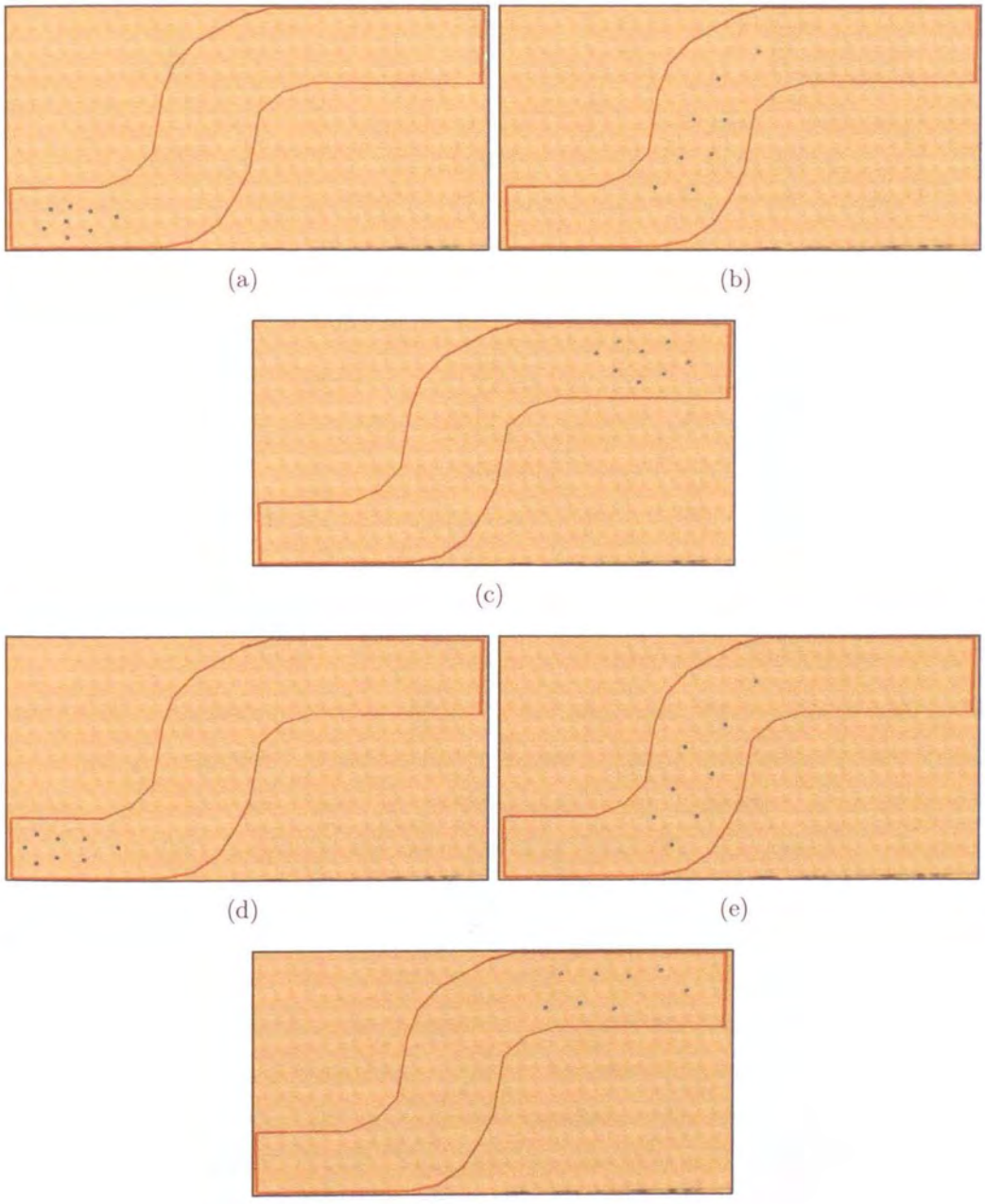


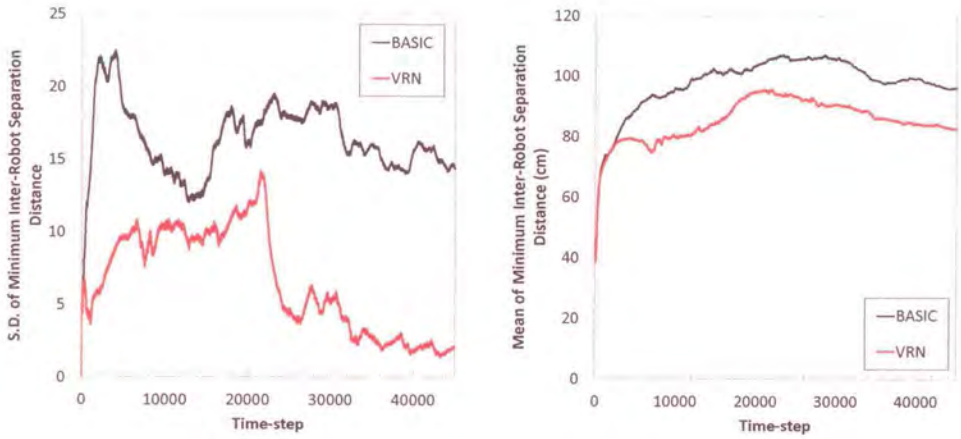
Figure 6.19: Simulation snapshots showing robot positions at increasing timesteps for a comparison between the VRN and basic control laws for adaptive control, for the s-bend experiment. (a) VRN at 1000 timesteps; (b) VRN at 10000 timesteps; (c) VRN at 30000 timesteps; (d) basic at 1000 timesteps; (e) basic at 10000 timesteps; (f) basic at 30000 timesteps.

shots given in Figure 6.19. We see a similar comparison between the VRN and basic control law implementations again, with on average a lower SD achieved with the VRN method throughout the experiment. The steady increase in mean minimum separation distance occurs as the swarm starts in a cluster and then 'stretches' into an elongated formation as it moves through the narrow s-bend. The mean is lower for the VRN approach as the formation maintains a more compact form, unlike the basic control law, which becomes more elongated through the corners of the s-bend. For the s-bend scenario a specific formation (such as the double wedge formation observed previously) is not created, and this is due to the narrow corners which the swarm has to navigate. Instead the formation for both control laws represents a chain-like formation of equilateral triangles, and visually, looking at the robot positions (Figure 6.19), there is less difference between the basic and VRN methods for the s-bend experiment. This is perhaps because for this experiment, there is no major change in size as such with the environment, rather the swarm simply has to navigate the corners in the corridor.

Figure 6.20 shows similar plots again, of SD and mean of minimum inter-robot separation distance for the 'chicane' experiment. These results share very similar characteristics to those from the s-bend experiment. From the plots in Figure 6.20, we do however again see a lower average SD and mean of minimum inter-robot separation distance achieved with the VRN method, and again this is attributed to the VRN method producing and maintaining a more compact, cohesive formation. In Figure 6.21, from visual inspection we see that the VRN method has achieved a cohesive formation (again representing a double wedge formation) from the initial starting positions, and maintains this formation through the chicane, ending in a cohesive chain formation. The basic method forms an elongated chain formation from the initial positions, which becomes less visually cohesive as the swarm moves through the chicane, which reflects the higher SD and mean minimum inter-robot separation distance shown in Figure 6.20.

6.4.5 Visbility Range Limitations

As previously discussed, restricting the visibility range of the robots to $r_{vis} = 3R/2$ helps in achieving a quasi-regular lattice formation, thus increasing formation cohesion. This assumes that the robots have sensing capabilities sufficient



(a) (b)

Figure 6.20: Plots of (a) the SD of minimum inter-robot separation distance and (b) the mean of the minimum inter-robot separation distance, for the 'chicane' dynamic collective movement experiment.

to satisfy this constraint. If for example a formation was required with a particularly large robot separation distance R, then the sensor range could become a limiting factor, and the constraint of $r_{vis} = 3R/2$ might not be possible.

Figure 6.22 shows an example of this situation for an instance with $N = 16$ robots, with $R = 50$, and r_{vis} limited to $r_{vis} = 55$ (shown for both the VRN and basic control laws). The experiments are run in the Webots physics and sensor based simulation package and also shown for comparison are the same experiments but with $r_{vis} = 3R/2$, as in Figure 6.7. Figure 6.23 shows the corresponding average minimum inter-robot separation distance for these experiments.

From visual inspection, for the VRN instance we see that with limited r_{vis} the resultant formation is not as cohesive as for $r_{vis} = 3R/2$ (in terms of regular inter-robot spacing and structure), however a quasi-regular separation is still achieved. This is reflected in the small difference between the curves for $r_{vis} = 3R/2$ and $r_{vis} = 55$ in Figure 6.23. It is interesting to note that the error is in fact reduced when using $r_{vis} = 55$, however this is at the expense of a less cohesive formation in terms of structure. This reduction in error can be attributed to the reduced effect of the attractive VRN force on the periphery of the formation which allows the formation to expand outwards to achieve better cohesion and reduce the clustering effect.

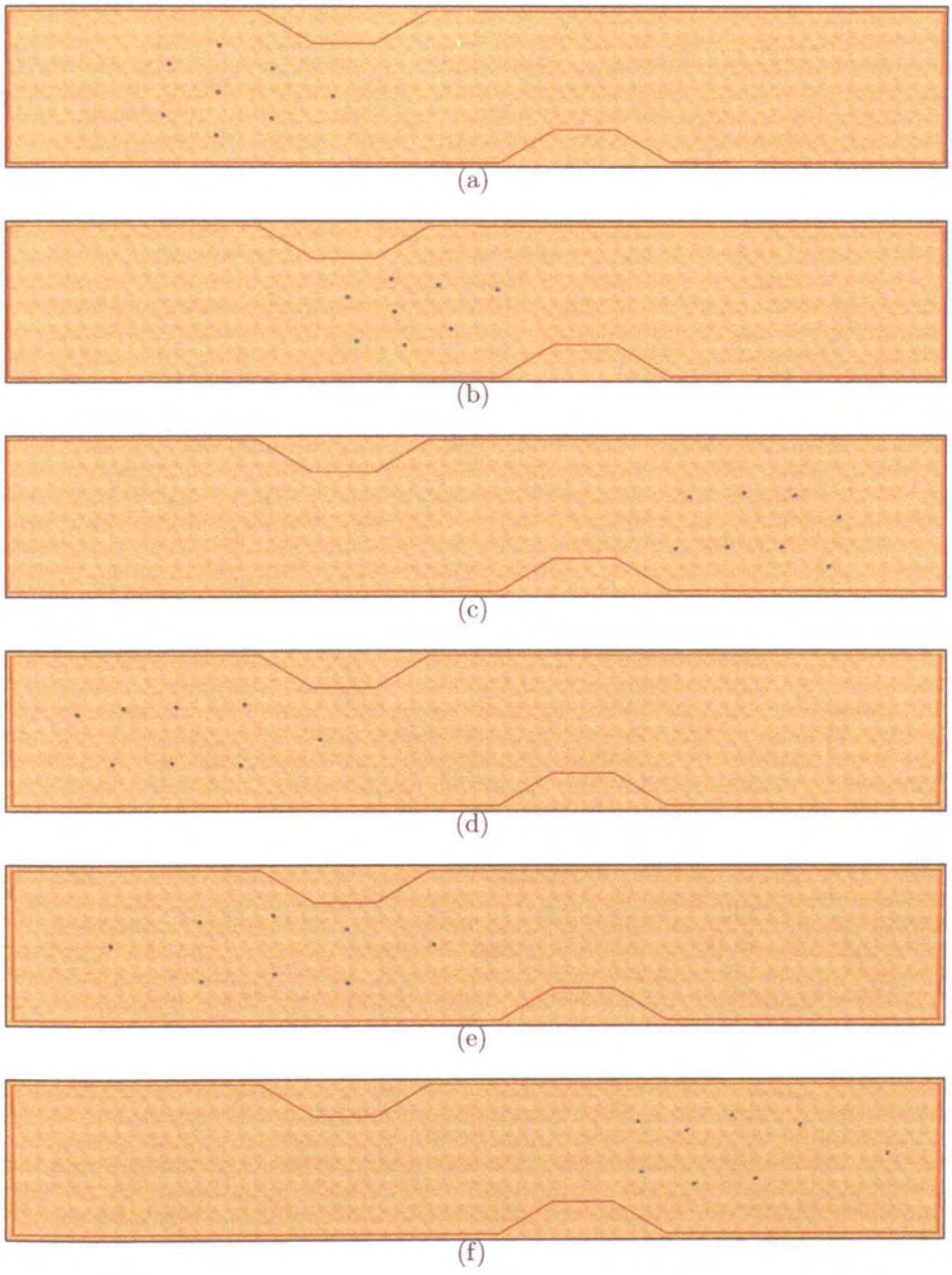


Figure 6.21: Simulation snapshots showing robot positions at increasing timesteps for a comparison between the VRN and basic control laws for adaptive formation control, for the chicane experiment. (a) VRN at 5000 timesteps; (b) VRN at 10000 timesteps; (c) VRN at 45000 timesteps; (d) basic at 5000 timesteps; (e) basic at 10000 timesteps; (f) basic at 45000 timesteps.

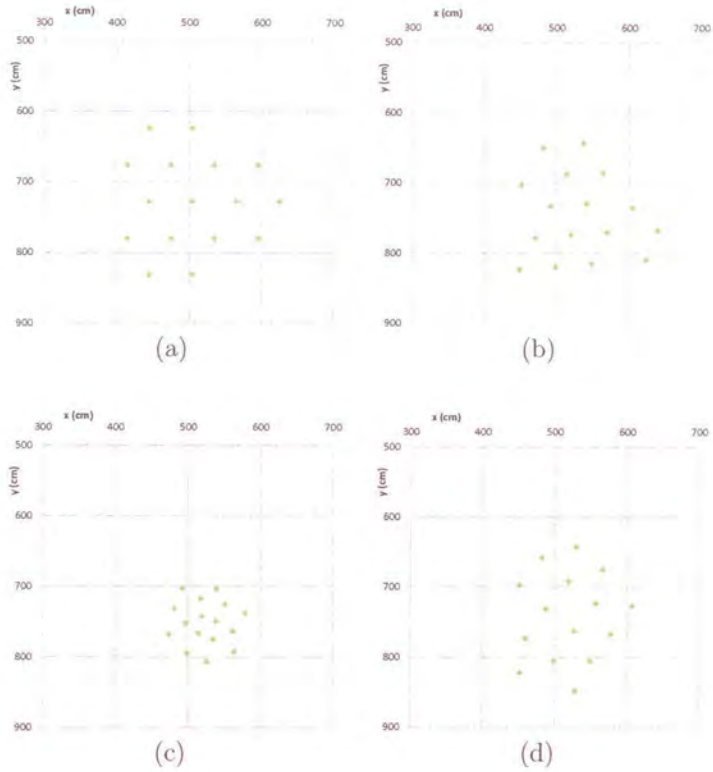


Figure 6.22: Plots showing final robot positions with $N = 16, R = 50$ for a comparison between the VRN and basic control laws for formation control with different r_{vis} values. (a), VRN control law with $r_{vis} = 3R/2$; (b), VRN control law with $r_{vis} = 55$; (c), basic control law with $r_{vis} = 3R/2$; and (d), VRN control law with $r_{vis} = 55$.

The difference from visual inspection of Figure 6.22 of the basic control instance is much more prominent (and this is reflected in the large difference in the error in average minimum inter-robot separation distance observed in Figure 6.23). With limited $r_{vis} = 55$ the formation has become much more expanded, although is still less cohesive in terms of regular structure. By reducing the r_{vis} range this has effectively reduced the clustering effect to a degree, by limiting the attractive component of the resultant force due to neighbouring robot agents, thus increasing the net repulsive force during the initial stages of self-organisation when the swarm is tightly packed. This effect then becomes adverse however, once the swarm has expanded and the robots can only 'see' the closest neighbouring robots.

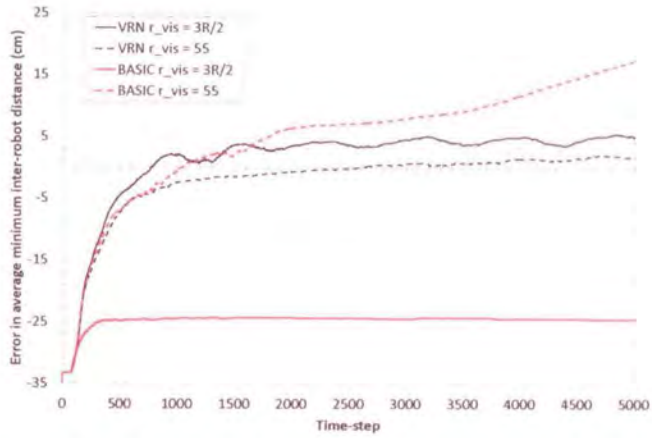


Figure 6.23: A comparison between the VRN and basic control laws for formation control with different r_{vis} values, showing the error in average minimum inter-robot separation distance. Experiment run with $N = 16, R = 50$.

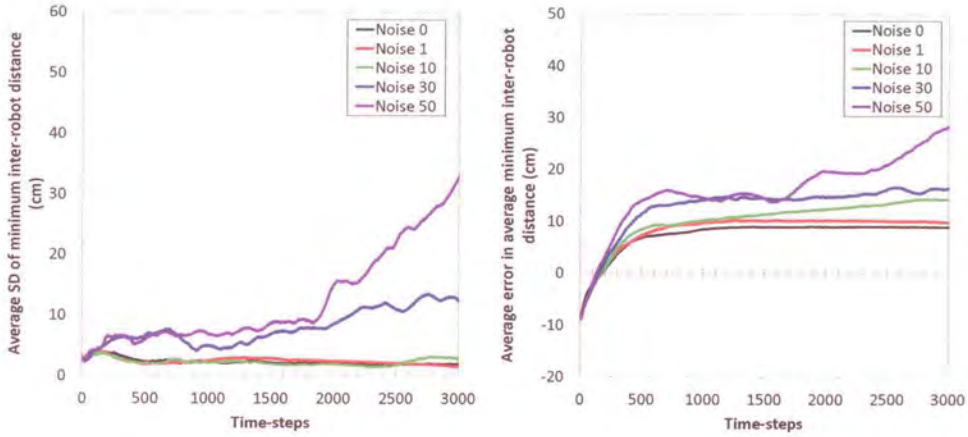
6.5 Robustness to Sensor Noise

The distributed nature of the proposed system, in particular the non-use of inter-robot communication, means that each individual robot is reliant entirely on its own sensors to gather the required information to execute the desired behaviours. It is therefore important to assess the robustness of the proposed system to the affects of sensor noise.

In this section a Webots simulation-based formation experiment is carried out similar to the one detailed in chapter 5 and above, with $N = 8$ robots, $R = 30$ and $r_{virtual} = 40$. The experiment is carried out for increasing levels of simulated sensor noise effecting the robots' θ, r and ϕ[1] measurements, for $0\%, 1\%, 10\%, 30\%$ and 50% levels of noise with respect to the measured value. For each level of noise the experiment is repeated 10 times and the results are averaged.

Figure 6.24 shows plots of the average SD of minimum inter-robot distance and average error in average minimum inter-robot distance, for the noise experiments. For the average SD (Figure 6.24(a)), the results for $0, 1\%$ and 10% noise follow similar trends, all lying within approximately $1.5 - 3cm$ for the majority of the time-steps. We then see a significant increase in average SD with increasing

[1] ϕ is the robot's measurement of its global orientation, using for example an on-board compass. This measurement is used in the low-level motor controls as described in Appendix D.



(a) (b)

Figure 6.24: Plots showing the effects of sensor noise on the VRN-based formation management. (a) Showing the average S.D. of inter-robot separation distance, and (b) showing the average error in average minimum inter-robot separation distance, for increasing levels of noise, as a percentage of the measured value.

noise to 30% and further still for 50%. The nature of the increase in average SD observed for the 30% and 50% levels of noise with increasing time-steps is due to the formations becoming unstable, with robots breaking away from the formation as the experiment progresses, thus reducing the overal cohesiveness of the swarm collective.

Further analysis is carried out by looking at the average error in average minimum inter-robot separation distance ((Figure 6.24(b)). Perhaps as expected, we see an increase in average error with increasing levels of noise. For 0% and 1% noise the swarm converges to a stable formation on average by approximately 1200 time-steps, with approximately a $0.8cm$ increase in average error. With 10% noise the swarm does not converge until on average 2700 time-steps, with an increase in error of approximately $5.8cm$. For a noise level of 30% the swarm converges at approximately 2850 time-steps with an increase in error of approximately $8.5cm$. With a noise level of 50% the swarm does not manage to maintain a stable formation, and by 3000 time-steps there is an increase in error of approximately $30cm$.

Although we see an increase in average SD and average error with a noise level of 30%, the swarm does maintain a relatively stable formation in that robots are not

seen to break away. The increase in SD and error is in large due to an observed increase in oscillation of individual robots about thier equilibrium positions, as the noise causes fluctuations in the sensor readings.

Given that in these experiments the levels of coherence remain relatively unchanged for levels of sensor noise up to 10%, and the overall formation management remains stable with levels of noise upto 30%, this provides example evidence of robustness of the system with respect to coping with noisy sensor data.

6.6 Conclusions and Discussion

This chapter has introduced a new formation control mechanism called VRNs, which can be used to overcome the 'clustering' problem inherent in the artificial force laws method, and moreover, offers more control over formation structure and has been shown to achieve higher levels of swarm cohesion over the basic control method, for a range of dynamic behaviours and tasks.

A dynamic directed movement extension to the control algorithms has been demonstrated, which allows distributed adaptation of the swarms structure, in response to changes in the geospatial characteristics of the environment, as sensed by the individual robots. This effectively extends the proposed control laws to achieve adaptive formation control for reactive collective movement through simple changing environments, while maintaining a fully distributed approach. The robots are adapting in response to the perceived 'template' of the local environment structure, and this is likened to the adaptive threshold of the ant algorithm developed in chapter 4. The adaptive formation control of this chapter successfully implements a distributed adaptive behaviour on a swarm of physical robots, providing great scope for a wide range of application specific behaviour developments.

The VRN mechanism has also been demonstrated as an effective means of achieving a directed flocking behaviour, without requiring direct inter-robot communications, as long as each individual robot knows a common heading. For the flocking behaviour as well as the static formation behaviour, experimental observations suggests a limitation of the scalability in terms of a potential increase in self-organisation time with increasing swarm size. However these results do

suggest that self-organisation, and thus the resultant desired behaviours, are still achievable with increasing swarm sizes, and that for swarm sizes up to at least $N = 100$ there is no significant performance degradation.

A key element to the dynamic directed movement behaviour is the distributed adaptation of the parameter R. This works in much the same way as the adaptive threshold method developed in chapter 4, where distributed adaptation of the threshold value T allows the swarm to adapt to variations across the image, and according to changes in the image environment, ultimately improving the performance. By self adaptation of the R parameter, the swarm effectively adapts its size in accordance with the geometric constraints of the environment, facilitating reactive autonomous formation manipulation. The VRN mechanism has been shown to increase cohesion in this adaptive process, by providing greater structure to the formation. In comparison, although still succeeding in adapting the spread of the formation, the basic method suffers greater loss of formation structure through the adaptive process.

By implementing the adaptation in a distributed way, there are potential gains in terms of lack of communication requirements, robustness to robot failure and scalability. As previously alluded to however, implementing such distributed behaviours on robotic agents operating in the real world introduces numerous challenges. In order to maintain robot anonymity and eliminate the requirement for inter-robot communication, the methods presented in this chapter (as well as the previous) require that the robots have the ability to sense neighbouring robots and objects in their local surroundings, and to also measure the range and bearing to such entities. The robots sensing abilities could be a limiting factor in terms of available performance. It has been shown that the sensor range to detect neighbouring robots can be a limiting factor, however with the VRN mechanism the resultant performance degradation is small for formation self-organisation, due in part to the limitless range of the VRNs.

Given the reliance on sensor readings, it was important to assess the ability of the developed algorithms to perform in the presence of sensor noise. Experiments involving simulated sensor noise have revealed a level of robustness of the VRN mechanism to sensor noise. Perhaps not surprisingly the performance is decreased with increasing noise, however the level of noise required to stop self-organisation appears to be relatively high. Again it must be stressed that this is an important

point given the reliance of the proposed methods on on-board sensor readings, and indeed further work in this area is required, specifically in testing with real robot / sensor hardware (as discussed further in the next chapter).

Another important point not yet discussed, again related to the chosen method of indirect communication via relative range/bearing sensing, is that the robots require line-of-sight in order to sense one another. Indeed in many applications this might not be a problem (such as the UAV scenario described in chapter 5), however for ground-based scenarios with environments which contain many obstacles, this could cause problems with the proposed method of range/bearing sensing. Further work would be required to ascertain the effects on the overall self-organisation of the swarm in the case of many occlusions.

Chapter 7

Conclusions and Discussion

The purpose of this chapter is to provide an overview and discussion of the conclusions reached from the work carried out in this thesis, particularly in relation to the main aims, objectives and hypotheses outlined in chapter 1. This chapter also recaps on the main contributions from the work of this thesis, in relation to the conclusions reached. Finally, a section of this chapter details the main areas of future work and potential further research related to the work of this thesis, with closing remarks given in the last section.

The overlaying focus of the research of this thesis has been on the study, development and experimentation of control algorithms for distributed swarms embedded in environment landscapes. More specifically, the focus has been on control algorithms for cooperative coordination and pattern formation, utilising the concept of self-organisation to facilitate complex cohesive formation behaviour at the global-level, using only simple, indirect communication as the local-level.

Referring back to chapter 1, the first hypothesis stated,

> *Relative complex behaviour at the global level can be achieved with little direct communication, as long as the agents can sense their local environment with an adequate degree of accuracy relevant to the particular problem.*

While it is acknowledged that other works have gone some way to showing this hypothesis to be true in the past, in this thesis the hypothesis is specifically

applied to the case of swarms embedded in environment landscapes, meaning that the swarm agents move and interact in 2D or 3D environments, be they virtual or real-world. The capabilities and requirements of the swarm become much more constrained when considering the agents as embedded entities, yet the world in which they are embedded and with which they interact, can be very open. These points are particularly pertinent when considering the 'real world' swarms of chapters 5 and 6, and the latter part of the hypothesis becomes more so important.

Both the image based ant-algorithms of chapters 3 and 4, and the swarm-robotics virtual force algorithms of chapters 5 and 6, demonstrate complex pattern formation behaviours using only local, indirect communication methods. Similar hypothesis are often associated with swarm intelligence methods, and in many cases have been demonstrated to be true. An important point of hypothesis 1 is the statement requiring that agents can sense their environment with a adequate degree of accuracy relevant to the particular problem. This is particularly important when considering swarm intelligence applied to robotics, as it is no longer plausible to assume that perfect sensing capabilities will be available, and, as such, methods of self-organisation relying on indirect communication via local sensing capabilities need to be designed with this in mind. The same does however hold for the virtual ant agents used in chapters 3 and 4; there is however far less restrictions on the capabilities of the individual agents to consider when designing such control algorithms.

The ant-algorithm of chapter 1 used the concept of stigmergy via synthetic pheromones to assist in the process of self-organised global pattern formation. It was assumed that the virtual agents could sense the pheromone trails without error, however, a probabilistic decision making process was implemented for decisions relating to the strength of pheromone detected. The virtual ant agents were assumed able to sense their local environment, and for the specific problem of leaf venation pattern extraction, the ants were programmed to 'see' the edge gradients in the image environment. The level of background noise in the image data has however been shown to have a serious affect on the agents abilities to detect the edges, and this in turn has a significant affect on the ability of the swarm to self-organise to the overall venation pattern and outline of the leaf patterns.

The swarm robot control laws of chapter 5 displayed global pattern formation and control while relying on only local indirect observatory communication. Despite the additional complications related to hardware considerations and limitations, the core swarm self organisation behaviours have required little to no change in implementation when moving from simple 2D point mass simulation to physics and sensor based simulation, which is in part due to the relatively simple requirements of the robot agents.

Embedded simulations on real robots showed the methods to work in the presence of sensor and actuator noise, however experiments in chapter 6 with simulated sensor noise showed a significant decrease in performance with large amounts of noise, to the point where self-organisation did not occur, and consequently, coherent pattern formation was not obtained. This is an important point to realise, as the term 'robustness' has been widely associated with swarm systems and behaviours, yet as has been shown in a number of different cases in this thesis, issues with robustness can severely reduce the success of self-organisation, and invalidate the first part of hypothesis 1, and in turn, necessitate the second part of hypothesis 1, in order to make this general hypothesis of self-organisation valid.

Similarly the term 'scalability' is often associated with swarm robotics systems. While indeed the general properties of swarm robotics systems provide a framework that has great potential to create scalable systems, the property cannot be assumed. Indeed experiments carried out in this thesis have suggested a limitation on the scalability in terms of an increase in self-organisation time for creating formations, with increasing numbers of robots. The experiments do however suggest that self-organisation does still occur with large numbers of robots, just with an increased convergence time. The experiments also showed that there is no significant decrease in performance for formation creation for swarms up to at least $N = 100$.

Hypothesis 2 stated,

> *Through simple homogeneous adaptation rules applied to individual agents, the swarm as a whole can adapt in a distributed fashion to optimise across a varying environment and improve overall performance in terms of cohesive pattern formation and adaptation.*

This hypothesis was tested in both the virtual and real-world settings considered in this thesis. In chapter 4 the ant algorithm for image pattern extraction was further developed to allow each individual agent to maintain and control an individual threshold value that controlled the sensitivity of the agents response to the given image feature of interest. Each agent was programmed with an identical adaptation rule, which was based on that agents local perception of its local surroundings, thus making for a homogeneous and distributed adaptation mechanism. As the swarm self-organised into a pattern formation representing a mapping of the desired image features, it was shown that the threshold of the entire swarm (averaged over all individual agents threshold values) converged to a near optimum value for the given image. This adaptation of the swarm to a given image environment improved the performance of the algorithm by facilitating autonomous parameter optimisation, which would otherwise have involved manual parameter fine tuning for the given image environment. This reinforces the concept of nature inspired distributed simple decision making as a powerful optimisation and problem solving tool in the digital, computerised world.

Research in chapter 4 also demonstrated how rapid adaptive continuous self-organisation can be achieved to facilitate quasi-real-time pattern adaptation and tracking. This did not involve explicit adaptation rules, but instead relied on the spatiotemporal qualities of the self-organisation methods employed with the ant-algorithm approach. This real-time adaptation to a continuously changing environment is more similar to what is found in nature, where swarms have to deal with a constantly changing world. Methods of self-organisation, distributed decision making and adaptation allow swarms in nature to cope with these dynamic challenges in relatively simple, yet sophisticated ways. The research of chapter 4 employed similar methods in computerised swarms, and exploited such qualities for use in digital image processing.

This method of distributed parameter adaptation was further exploited to facilitate dynamic autonomous parameter optimisation, where the distributed threshold values of the swarm were shown to adapt in real-time, to a continuously changing environment. As well as the globally converged threshold values, it was also shown that this distributed adaptation approach allowed for the agents of the swarm to adapt their threshold values depending on the specific properties of their local surroundings, such that different areas of the swarm in different

areas of the environment would converge to local optimum values thus allowing the swarm to dynamically adapt to environments with areas requiring multiple different threshold values.

By testing the claims of hypothesis 2 and exploiting the distributed nature of the ant algorithm approach with the adaptive threshold, research from this thesis has shown that the swarm intelligence approach to low-level image and video processing has potential for improvements over traditional filtering methods. Furthermore, considering the wider implications of hypothesis 2, this research has demonstrated further how the analogy to natural swarms can be exploited more widely in the computational world, utilising embedded swarms, more representative of the natural swarm counterpart.

The dynamic collective movement algorithm developed in chapter 6 was a test of hypothesis 2 in a 'real-world' embedded swarm application. Along with the homogeneous physical make-up of the robot swarm, the robots were also running identical control algorithms. Although it was relatively straight forward to implement the adaptive parameter tuning to the control algorithm, the major challenge here was maintaining a level of cohesiveness to the swarms formation and movement, as this required the individual robots to assess not only their current situation with respect to the local environment, but also with respect to the local swarm formation structure. This was ultimately achieved through the development of the VRN mechanism, which allows individual robots to estimate their position with respect to the periphery of the formation, and in turn self-manage local artificial forces affecting their position in the formation.

Through development and testing of the adaptive threshold ant algorithm and the VRN dynamic directed movement algorithms, the work of this thesis has shown, through a number of case studies, that individual agent adaptation can converge to swarm-level adaptation though the process of self-organisation. Moreover, this process has been exploited in two very different areas of application, which hold different challenges and characteristics. This latter point is important, as this work has further shown the level of applicability that the swarm intelligence approach has to problem solving in the computational world.

Although much of the work of this thesis has been inspired by swarms in nature, there are instances where particular elements of the natural swarm counterpart

cannot be replicated in the given computational problem, or where doing so would not benefit the solution to the particular problem. Such instances are particularly relevant in swarm robotics, where there are additional constraints to consider such as hardware limitations. For example although the use of actual tangible pheromones can be used (and indeed has been used in laboratory experiments [62][63]), it simply may not be practical to implement such a mechanism for use in operationally deployed robots. As such, alternative methods have to be created, such as the VRN mechanism developed in chapter 6, that take inspiration from the core concepts of the natural swarm behaviours (in this case indirect communication and elementary force laws) to produce the desired self-organising effect. This is similar to the concept of daemon actions (as used with the ant algorithms developed in chapters 3 and 4), where no specific natural counterpart exists, but the inclusion of the additional mechanism, working alongside the core swarm behaviour, increases the overall effectiveness of the solution.

7.1 Contributions Overview

The research of this thesis has made a number of incremental contributions to knowledge regarding the concept of swarm intelligence, specifically in the area of pattern formation and coordination and control. More substantial contributions have been made in the form of new algorithm developments, bringing new swarm intelligence inspired functionality to both image and video processing, and swarm robotics coordination and control. An overview highlighting specific important contributions from this thesis is given below:

- A number of new ant algorithms based on the Ant System algorithm have been developed for extracting and tracking various low-level features from digital images and videos.
- Specifically a novel multi-swarm multi-feature extraction / tracking algorithm has been developed, which allows the simultaneous extraction and/or tracking of multiple image features from either static or dynamic imagery. The algorithm is modified with ease to allow one or multiple swarm(s) to self-organise to alternative image features, in either static or dynamic image environments.

- A new distributed dynamic thresholding technique has been researched for the feature extraction ant algorithms, where each individual agent has control over its own feature threshold value, allowing distributed, adaptive, multi-level thresholding across the swarm. This eliminates the requirement for user-set thresholds and allows for changes in threshold requirements in dynamic imagery, as well as multi-level thresholding for imagery with varying characteristics.

- The feature extraction ant algorithms have been evaluated qualitatively and quantitatively against ground truth data and traditional existing image processing techniques. The quantitative analysis provides valuable statistical results to inform on the state-of-the-art of applying ant-algorithms and swarm self-organisation to image and video processing.

- A prototypical swarm robotics system has been developed for the problem of target acquisition and monitoring in partially observable environments. Based on the balancing of local internal and external environmental virtual forces, a number of reactive coordination and control laws have been developed to autonomously coordinate a group of nonholonomic homogeneous robots to distribute in a given environment and surround and monitor detected targets. A multi-behaviour strategy has been implemented, based on simple reactive, local decision making. Experiments carried out on real robots has helped advance the state-of-the-art in swarm robotics research, providing additional results on the robustness to real hardware sensor and actuator noise.

- A novel method of sensor driven dynamic collective movement for a swarm of mobile robots has been proposed, using on-board range sensors to allow the swarm to dynamically adapt to changing geospatial characteristics while traversing a structured environment.

- A new swarm robotics formation coordination and control algorithm has been developed called the VRN mechanism, which acts as a distributed dynamic formation management system. The VRN mechanism is shown to increase swarm cohesion in the process of pattern formation and reduce the 'clustering effect' which is often prominent in swarm robotics systems using virtual potential fields for self-organisation. In terms of dynamic collective

movement, the VRN mechanism is again shown to increase cohesiveness throughout the dynamic movement process. The VRN mechanism is further shown to provide a directed flocking capability, and to reduce problems due to range/bearing sensor dependency.

7.2 Future Work

While the research carried out in this thesis has brought new knowledge and capabilities to computational swarm intelligence, and has solved a number of problems in the related application areas, additional work can be done to further these advancements.

From an application point of view, further performance analysis of the ant algorithms of chapters 3 and 4 could be carried out in terms of comparing the developed feature extraction and tracking algorithms to other existing methods, for a more comprehensive comparison of the how well the ant algorithm methods perform in relation to existing methods not based on swarm intelligence. The capabilities of the image feature extraction algorithm could be developed further by considering the multi-swarm implementation to use multiple swarms for simultaneous multi-scale space analysis. The use of the proposed algorithms for extracting a larger range of image features would also be of interest, for example image entropy, or using bespoke feature vectors. Likewise, extending the use of the proposed algorithm for more complex image feature tracking is of interest.

It might also be possible to implement, in a similar way to the adaptive threshold method, a full parameter autonomous optimisation system. This could be implemented with a form of genetic algorithm, where each agent is encoded with a genetic string representing a given parameter set-up, and cross-over occurs as agents occupy the same pixels. The idea would be to evolve an optimum swarm for a given image data-set. In a similar approach to the adaptive threshold, it would be specifically interesting to investigate the possibility of adjusting the trade-off between exploration and exploitation to dynamically change as the self-organisation process evolved. This might help in reducing susceptibility to noise in the image data, by allowing the swarm to begin with a higher level of exploration in order to quickly find the features of interest, and then to exploit these

discovered locations and reduce the chance of agents wandering into noisy regions of the image environment.

A major area of future work from this thesis is validating the dynamic directed movement and VRN mechanism on real robots, and over a wider range of scenarios and environments. Although experiments were carried out to assess the affects of sensor noise, these experiments were limited, and testing on real hardware is an essential part of future work to assess the more practical implementation issues of the developed theoretical framework in dealing with real-world sensor noise, as well as actuator noise (such as wheel slippage). The sensors used in simulation for the dynamic directed movement algorithms were simple range finders (equivalent to a sparse laser range finder array around the robot). Future work should incorporate more sophisticated sensing capabilities, and in particular, it is of interest to assess the possible use of omnidirectional vision as a means of range/bearing sensing for the VRN architecture, where the vision sensor could also be used for sensing and monitoring purposes specific to given applications (in addition to being used for coordination and control). Future work might also consider different types of robot failure, and how they affect the proposed control laws and behaviours. Again this would benefit greatly from experimentation using real robots, where real sensor and actuator failures could be introduced.

Further work is required to assess the affect of occlusions in the environment with regards to required line-of-sight sensing for relative range/bearing measurements. The VRN mechanism might be well suited to tackling this problem, by replacing last known positions of neighbouring robots that move behind an obstacle with VRNs, and employing a predictive model to create a 'best guess' approach to where the robot is over time (with a decaying confidence level with increasing time that the robot does not re-appear in line-of-sight).

The flocking capabilities of the VRN mechanism should be further explored. In particular it would be interesting to research a method whereby the individual robots are able to estimate the heading of neighbouring robots, based on sensory information (which again could utilise omnidirectional vision). Robots on the periphery would also estimate the desired direction of travel with respect to the sensed environment structure, which would be balanced with the estimated direction of travel of the swarm to decide on the individuals desired flocking direction. This would eliminate the requirement for a 'leader' robot in the dynamic directed

movement algorithm, which could further increase the cohesiveness of the swarm movement, as robots on the leading edge of the swarm (with respect to the direction of travel) would also be influenced by the rest of the swarm. As previously mentioned, the potential benefit of employing the VRN flocking method is the elimination of the requirement to communicate information between robots, such as velocity and bearing. The VRN flocking methods should be directly compared to similar existing flocking methods, to highlight any differences in performance and potential improvements made by employing the VRN method.

It would be interesting to research a learning strategy for the VRN mechanism, to allow the robots to learn when and where best to position VRNs given specific sensory input about the local environment. This could allow the system to evolve a number of cooperative movement strategies for behaviours such as searching, obstacle avoidance and structure inspection, where the behaviours are optimised for given environment types.

A direction of interest with regards to further application development is implementing the VRN mechanism with larger scale mobile robots to further develop a multi-robot system similar to the target acquisition and monitoring system developed in chapter 5, for use in search and rescue, and security and defence scenarios. The use of on-board omnidirectional vision for range/bearing sensing for the VRN mechanism could be used for more wider environment monitoring, to retrieve visual information from the environment, and collectively build a feature map of an unknown environment through passive fusing of visual information from the swarm of robots as they collectively move throughout the environment. The control laws of the robots in this scenario would be tailored to maximise collective information retrieval in minimal time, utilising adaptive dispersion and cooperative search behaviours, implemented using the VRN mechanism. In this context it would also be of interest to experiment with the use of a Probabilistic Finite State Machine (PFSM) architecture. Although the use of PFSMs are not new in swarm robotics, it would be instructive to assess whether this approach would improve overall system performance with respect to dealing with noisy sensor data leading to false positive target detections for example.

7.3 Closing Remarks

The research carried out in this thesis has been inspired by naturally occurring swarm behaviours observed in social insect swarms, animals, and fundamental physics. Understanding the mechanics of these swarming behaviours, and moreover, how to apply them to solve abstract computational problems, provides a wide scope of fascinating and challenging research, with often rewarding results. Indeed as the swarm intelligence research community has grown, the range and scope of applications and problems to which this relatively new area of computational science is applied, has expanded immeasurably. The inspiration from all types of swarms observed in nature continues to assist in the development of new methods for computational problem solving, inspiring both bespoke and widely applicable solutions. Both the large amount of successful methods previously developed, and the multitude of open problems still existing, provide great motivation for continuing research and development in the area of swarm intelligence. The work of this thesis has made a number of contributions to this most interesting of research areas, and at the same time has created a number of specific areas for future development in the specific areas of image processing and robotics applications. No doubt research in swarm intelligence will continue to mature, with an ever increasing knowledge base and understanding of naturally occurring swarm behaviours, and how to harness the sophisticated simplicity of such behaviours to efficiently solve problems in the computational world.

Chapter 8

Appendices

8.1 Appendix A: Leaf Image Dataset

Figure 8.1 shows thumbnails of the twenty real leaf images used in the image pattern feature extraction case-study in Chapter 3.

8.2 Appendix B: Real Robot Experiments Set-up

For the work in this thesis, a purpose built enclosure was built for carrying out embedded simulations on small-scale robots in a laboratory environment. The enclosure measures approximately $120cm$ by $120cm$ and includes an overhead vision-based robot pose tracking system.

The system tracks individual robot positions and orientation (x, y, θ), in the ground plane. Positions are calculated to approximately $\pm 1.4cm$ and orientations to approximately $\pm 0.05rads$ at a sampling rate of approximately $0.125s$. Tracking the robots pose serves two purposes: (a) the pose information is filtered and each individual robot is continuously sent readings of it's own bearing (to simulate an on-board compass) and also the range and bearing of neighbouring robots within the set visible range (to simulate the on-board relative range/bearing measurements); (b) the raw pose information is used to analyse

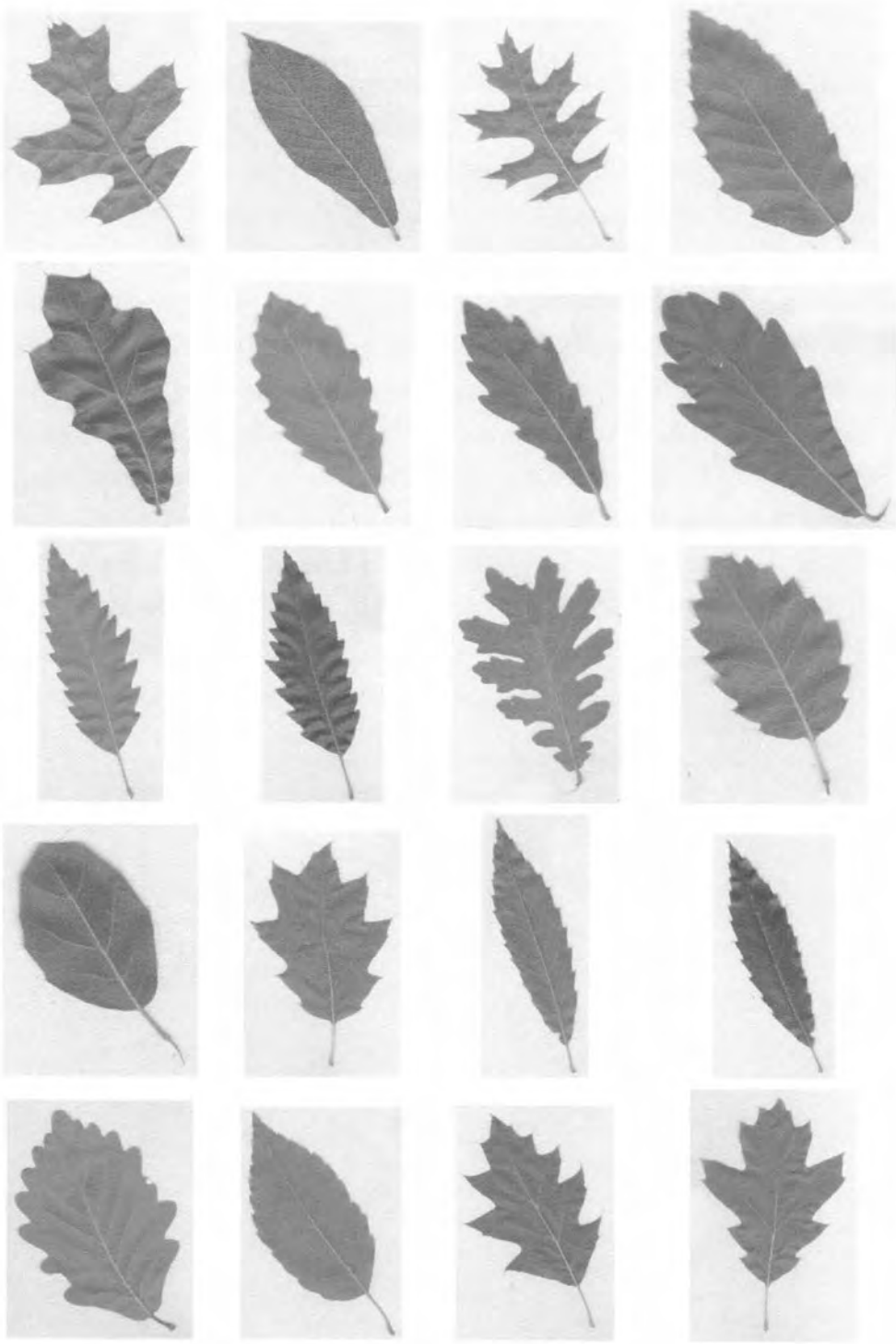


Figure 8.1: Thumbnail images of the RBG KEW leaf image dataset.

the performance of the swarm using various metrics.

Figure 8.2 shows a schematic representation of the laboratory set-up with the overhead tracking system. A camera is suspended above the robot enclosure, pointing downwards to give an overhead view. Each robot is given a marker which is placed ontop of the robot. The markers are a combination of colours uniquely identifying each robot and showing the orientation in the ground plane. These markers are then tracked by applying a number of RGB filters and morphological operators to the images captured from the overhead camera. The camera is attached to a PC where the image processing takes place to extract the (x, y, θ) pose information for each robot. The (x, y) information is initially extracted in the image frame (as pixel locations), which is then converted to the object plane (i.e. real world coordinates, in cms) through camera calibration (see below). The pose information is then passed to the individual controllers running separately for each robot, which are again running on the remote PC. The outputs from the robot controllers are left and right wheel velocities, which are then sent to the respective robots via Bluetooth communication.

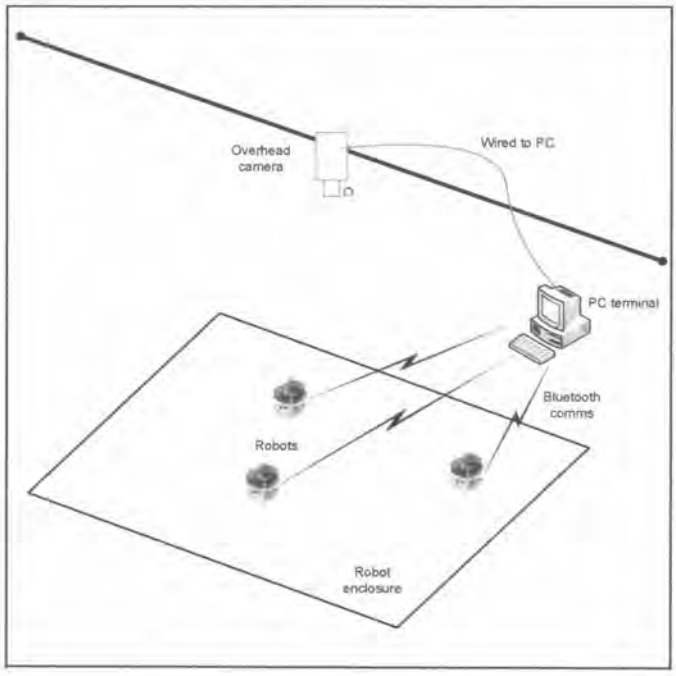


Figure 8.2: A schematic diagram of the overhead tracking system for embedded robot simulations.

8.2.1 Camera Calibration

The overhead vision-based tracking system tracks the robots positions in the image frame $\underline{p}(u, v)$. A transform $\underline{H}$ is required to convert the captured pose coordinates to the object plane, $\underline{P}(u, v)$, such that $\underline{p} = \underline{H}\ \underline{P}$. Referring to the pinhole camera model, a camera matrix is used to denote a projective mapping from the object frame to the image frame, denoting

$$\underline{A}.\underline{x} = \underline{b} \tag{8.1}$$

where

$$\underline{A} = \begin{pmatrix} x_0 y_0 1 & 000 \\ 000 & x_0 y_0 1 \\ \vdots & \vdots \\ x_N y_N 1 & 000 \\ 000 & x_N y_N 1 \end{pmatrix} \tag{8.2}$$

$$\underline{x} = \begin{pmatrix} a_{11} \\ a_{12} \\ a_{13} \\ a_{21} \\ a_{22} \\ a_{23} \end{pmatrix} \tag{8.3}$$

$$\underline{b} = \begin{pmatrix} u_0 \\ v_0 \\ \vdots \\ \vdots \\ u_N \\ v_N \end{pmatrix} \tag{8.4}$$

By measuring N points in the object plane and the same N points in the image plane, the camera parameters (the unknown 'a' values) can be calculated by means of a direct linear transformation, with,

$$\underline{x} = \left(\underline{A}^T \underline{A}\right)^{-1} \underline{A}^T \underline{b} \tag{8.5}$$

With the camera parameters known, from Equation 8.5, with,

$$u = a_{11}x + a_{12}y + a_{13} \tag{8.6}$$

$$v = a_{21}x + a_{22}y + a_{23} \tag{8.7}$$

Equations 8.10 and 8.11 can be rearranged to give,

$$x = \frac{\left(u - a_{12}\left(\frac{((a_{11}v)-(a_{11}a_{12}u)+(a_{21}a_{13})-(a_{11}a_{23}))}{((a_{11}a_{22})-(a_{21}a_{12}))}\right) - a_{13}\right)}{a_{11}} \tag{8.8}$$

$$y = \frac{((a_{11}v) - (a_{11}a_{21}u) + (a_{21}a_{13}) - (a_{11}a_{23}))}{((a_{11}a_{22}) - (a_{21}a_{12}))} \tag{8.9}$$

thus giving the pose coordinates in the object plane, from the pixel coordinates in the image frame, and the calibrated camera parameters.

8.3 Appendix C: RGB Feature Tracking

This appendix details additional experiments from chapter 4, carried out using RGB information to guide the ant-algorithm swarm to track specific coloured objects within an RGB image environment, in the same way as for the RGB feature extraction in chapter 3.

Figure 8.3 gives example snap-shots of an experiment showing the swarm tracking a red circle across image frames. The input to the algorithm here is from a USB webcam, at 320 by 240 pixels resolution and approximately 14 frames per second. A solid red circle (with RGB values $(255, 0, 0)$) of radius 50 pixels is then overlayed onto the image frames, and moved a distance of 1 pixel per frame around the image. The algorithm is the same as described in chapter 3, and implemented as in section 3.7.1, with the red weighted RGB heuristic information defined as:

$$\eta_{ij} = (R_{ij} - B_{ij}) + (R_{ij} - G_{ij}) \tag{8.10}$$

where R, G and B are the *red*, *green* and *blue* channels of the image respectively.

Figure 8.3 shows the swarm successfully tracking the red circle feature across multiple frames. Figure 8.4(a) shows a plot of the trajectories of the red circle shape along with the swarm COG against the image frames (in the z-axis). The COG shown here is the COG of the swarm agent positions, not the pheromone map. The swarm COG initially starts in the centre of the image as the swarm is initialised with agents at random locations throughout the image. The swarms quickly converges on the red feature and proceeds to track it as it moves around the image. The swarm COG can be seen to effectively track the trajectory of the red feature, although with a level of imperfection, as detailed in Figure 8.4(b), plotting the error distance between the swarm COG and the true centre of the red circle feature at each image frame. The error is maximum at the start before the swarm converges onto the feature (116 pixels) at which point to error drops to a minimum of 9 pixels, after which the error fluctuates at an average of 29 pixels throughout the 600 frames. Despite the calculated error levels, the COG manages to stay within the radius of the circle feature once the swarm converges onto the feature, showing good temporal adaptive self-organisation.

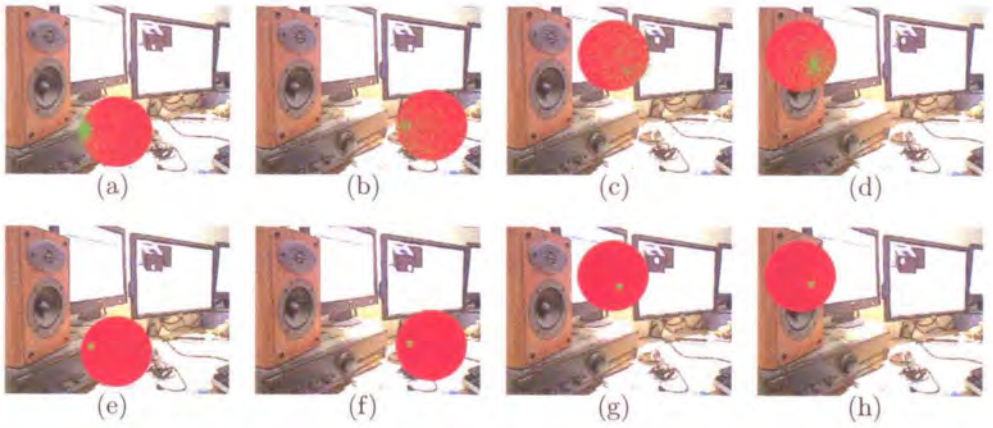
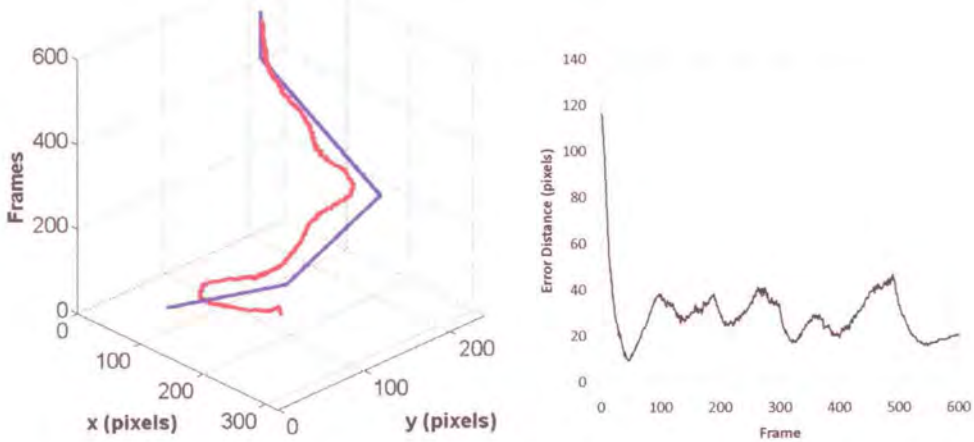


Figure 8.3: Example frames showing real-time RGB object tracking. Top row: original RGB image with agents positions superimposed (bright green pixels); bottom row: original RGB image with swarm COG superimposed (bright green square). (a,e) Frame 100; (b,f) frame 200; (c,g) frame 400; (d,h) frame 600.



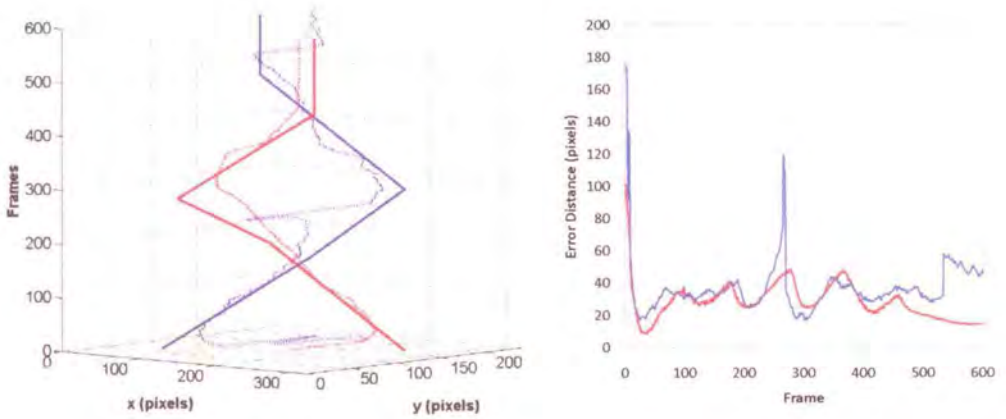
(a) (b)

Figure 8.4: (a) A 3D plot of trajectories of the swarm COG (red curve) and true centre of the shape (blue curve), and (b) a plot of the error between the swarm COG and true centre of the shape, versus image frames.

Another, similar example is depicted in Figure 8.5, this time utilising two swarms to simultaneously track two filled circle shapes; one red (with RGB values $(255, 0, 0)$) and one blue (with RGB values $(0, 0, 255)$). One of the swarms is programmed with red RGB heuristic information (Equation 8.10) and the other with blue RGB heuristic information, with:

$$\eta_{ij} = (B_{ij} - R_{ij}) + (B_{ij} - G_{ij}) \tag{8.11}$$

The shape features follow similar (non-overlapping) trajectories to that from the previous experiment. We see similar results with the dual swarm, dual feature tracking, with both swarms successfully tracking the respective features throughout the environment, with a mostly consistent level of error. The spike in the error for the blue swarm at approximately 265 frames, is where the COG of the swarm momentarily moved off the shape feature. This can happen due to, for example, changes in ambient lighting, causing members of the swarm to be suddenly attracted to other areas of the image. One point to note is the rapid self-healing observed, where the swarm quickly re-self-organises to the blue shape features and the COG moves back onto the shape. Such self-healing properties of



(a) (b)

Figure 8.5: (a) A 3D plot of trajectories of the dual swarm COG's (dotted red and blue curves) and true centre of the shapes (solid red and blue curve), and (b) a plot of the error between the swarm COG and true centre of the shape, for the red shape/swarm (red curve) and blue shape/swarm (blue curve), versus image frames.

swarm intelligence inspired algorithms will become a topic of further discussion in the next chapter.

The dynamic nature of the swarm approach shows potential for use in dynamic image processing, by exploiting the continuous self-organisation. It should be noted however, that these are relatively simple examples of image feature tracking, using simple RGB colour-space, and defining the objects to be tracked as definitive RGB colours.

8.3.1 Case Study in Robot Vision Object Tracking

As a further case study experiment the ant-algorithm of chapter 3, section 3.4, is applied to track a coloured object from frame to frame in real-time, in a robot vision application. The heuristic information component of the algorithm is modified to allow the swarm to track regions of the image with a prominent red intensity, using Equation 8.10. Although still using RGB colour space, this is a slightly more complex example, as the object to be tracked is a real-world object, and will thus not be of 'true red' colour, which was the case in the previous example.

The algorithm is then executed the same as above (and described in chapter 3, section 3.7.1). After each agent has moved the COG of the swarm is calculated within the image. Additionally the number of pixel locations containing pheromone are counted. Both of these measures are used to determine the robot's desired motor commands in order to track the object, by aiming to keep the object in the centre of the robot's FOV. The desired movement is calculated in a similar way to using Equation 4.3, but with the additional measurement of the amount of deposited pheromone being used to provide a crude estimate of the distance to the object. Depending on the estimated distance, the desired movement may include a forwards bias, or the robot may simple turn on the spot.

There is also a daemon action implemented that terminates any ant agent with $\eta_{ij} < T$ for more than Z consecutive time-steps, in a similar way to that of chapter 3, section 3.7.2. If the COG is currently located at a pixel such that $\eta_{i_{COG}j_{COG}} \geq T$ then the terminated agent is immediately replaced at the new location of (i_{COG}, j_{COG}), else, the terminated agent is immediately replaced at a new 'random' location within the image. At each time-step the most recent image frame received from the robot's on-board camera becomes the search image for the algorithm. Figure 8.6 shows example snap-shot frames of the swarm tracking an orange object moving across the image (from left to right), with the swarm agent positions and COG superimposed.

To test the performance of the algorithm with respect to object tracking, an experiment is carried out involving two e-puck [114] differential drive robots. One robot is given a bright orange 'jacket' and is driven manually around the previously mentioned purpose build robot enclosure. The other robot is controlled remotely via Bluetooth communication by a desktop computer running the above described algorithm, with the goal of following the other robot around the enclosure. The following parameter settings were used: $N = 2000$, $\rho = 0.05$, $T = 150$, $\alpha = 1.0$, $\beta = 5.0$, $tabu_{max} = 8$ and $Z = 1$.

Figure 8.7 shows an example snap-shot of two image frames taken from the robot's on-board camera running the algorithm with the RGB heuristic information. From visual inspection we can see that for the image frame including the object of interest, the swarm of ant-agents has converged onto the object, thus placing the COG over the object. This would then allow the robot to turn left to position

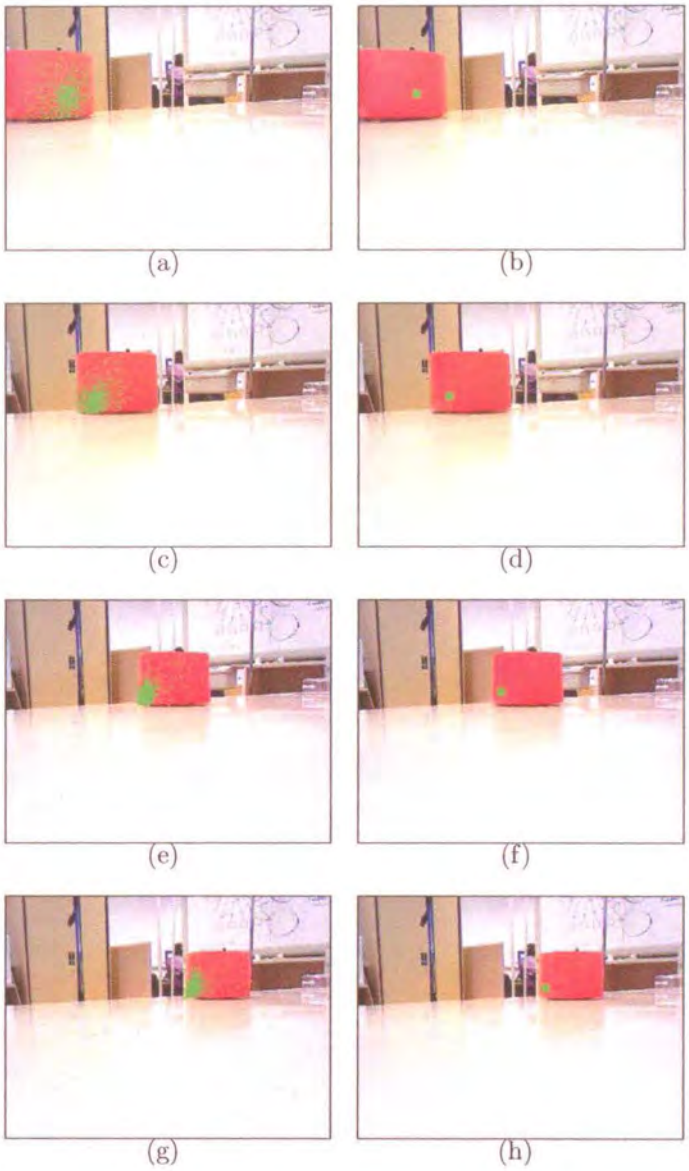


Figure 8.6: Example frames showing real-time RGB object tracking. Left column: original RGB image with agents positions superimposed (bright green pixels); right column: original RGB image with swarm COG superimposed (bright green square). (a,b) frame 200; (c,d) frame 400; (e,f) frame 600; (g,h) frame 800.

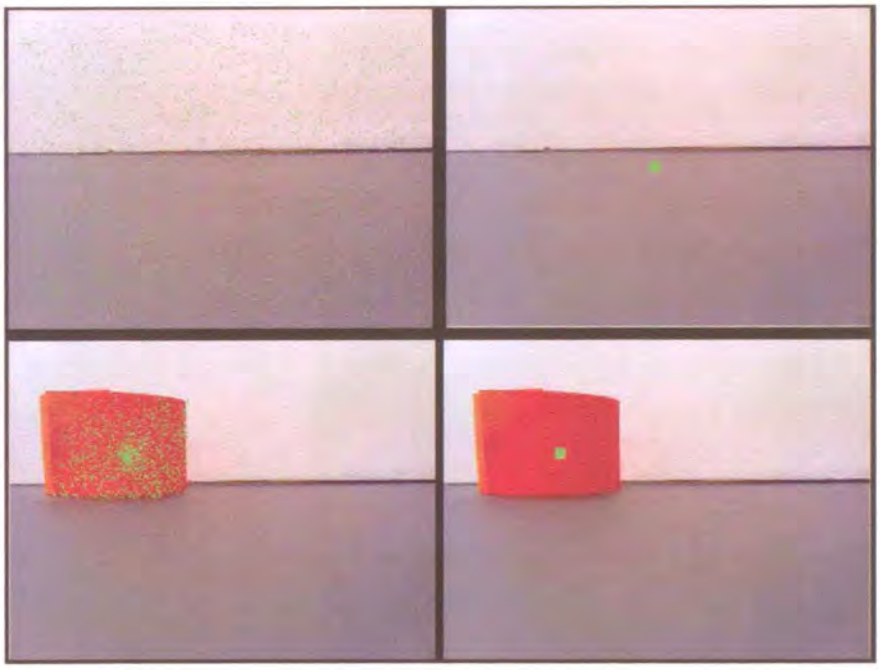


Figure 8.7: A snapshot image from the robot's on-board camera running the algorithm with the RGB heuristic information. The left column shows the original image with the ant-agents positions superimposed on-top. The right column shows the original image with the COG superimposed as a solid square. The top row is an example image frame where there is no object present. The bottom row is an example image frame where the target robot is in the follower robot's FOV.

the COG in the centre of its FOV.

Figure 8.8 shows the trajectories of the leader robot and follower robot for a typical run of the robot tracking experiment. From visual inspection we can see that the following-robot has successfully traversed a similar route to the object-robot, by using the proposed algorithm to track the robot using on-board vision. The maximum distance between the robots at any one time during the experiment is $28.1cm$, with an average separation distance of $19.1cm$ (the separation distance at the start is $22.2cm$). It should be noted that there was a substantial amount of wheel slip throughout the experiments, which in part caused an inconsistent distance between the two robots throughout the experiment, which can account for the minor differences in the two routes.

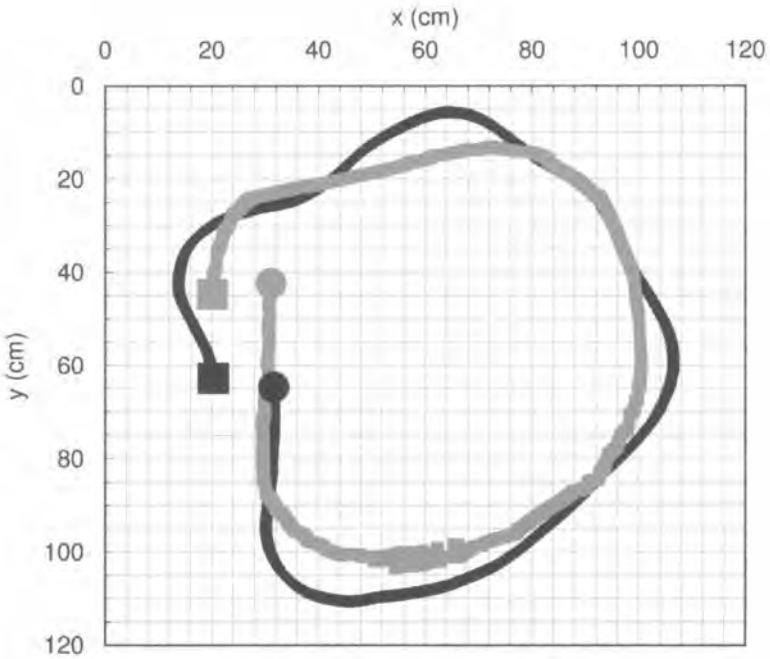


Figure 8.8: A plot of the robot trajectories for the robot-robot following experiment (using e-puck robots). A solid black line and solid grey line show the leader and follower trajectories respectively, with solid circle and solid square markers showing the start and end positions, respectively. The average velocity of both robots was approximately $1.7cm.s^{-1}$, with an average start to finish time of approximately two minutes.

8.4 Appendix D: Robot Low-level Motion Control

The swarm robotics control laws developed in this thesis all output desired Δx and Δy displacements at each time instance. In order for the robots to move to the desired location, the Δx and Δy displacements need to be converted into left and right wheel speeds for the differential drive robots used in this thesis.

A low-level motion control law based on the one reported in [132] is implemented in order to achieve this, which provides smooth steering of the differential-drive robots towards their desired locations.

A change of coordinates is required, defining the polar coordinate transformation as $\rho = \sqrt{x^2 + y^2}$, $\gamma = \arctan(y, x) - \phi + \pi$, $\delta = \gamma + \phi$, where x, y is the desired position in the robots coordinate frame, and ϕ is the robots global orientation.

The following control law can then be derived (see [132] for details) to give the driving velocity, ν, and steering velocity, ω:

$$\nu = k_1 \rho \cos(\gamma) \tag{8.12}$$

$$\omega = k_2 \gamma + k_1 \frac{\sin(\gamma)\cos(\gamma)}{\gamma}(\gamma + k_3 \delta) \tag{8.13}$$

where k_1, k_2 and k_3 are positive constants.

The left and right wheel speeds are then given by:

$$\omega_L = \frac{2\nu - d\omega}{2s} \tag{8.14}$$

$$\omega_R = \frac{2\nu}{s} - \frac{2\nu - d\omega}{2s} \tag{8.15}$$

where s is the wheel radius and d is the axle length of the homogeneous robots.

8.5 Appendix E: Generic Obstacle Avoidance

The robot controllers presented in this thesis encompass a mechanism that makes inter-robot collisions unlikely, due to the repulsive virtual forces that the robots would experience in close proximity to one another. This is however not guaranteed, and with the exception of the dynamic directed movement algorithm, the virtual force control laws do not take into account obstacles in the environment. It was therefore decided to include a generic 'reactive' obstacle avoidance behaviour common to all the proposed control laws. The obstacle avoidance behaviour uses the standard IR ring present on the e-puck robot, and acts as a 'fail-safe' mode, taking precedence over other behaviours whenever an obstacle is detected. Pseudo-code for the obstacle avoidance behaviour is given in Algorithm 8.1. This algorithm is executed at the beginning of each control law, for every time-step.

Depending on which sensor is deemed to have detected the object (or be closest to the detected object), the left and right wheels speeds are set to pre-defined

Algorithm 8.1: Pseudo-code for the generic obstacle avoidance behaviour.

```
Read all IR sensors
Determine maximum sensor reading
if Maximum sensor reading > obstacle threshold then
    Set left/right wheel speeds according to which sensor has the maximum reading
    Loop for several time-steps
else
    Continue executing control law
end if
```

values, to steer the robot away from the object. It is important to note that this very basic generic behaviour was included as a last-resort failsafe, and was rarely executed throughout the experiments carried out in this thesis.

8.6 Appendix F: Robot On-board Target Detection

For experiments in chapter 5, the e-puck's on-board camera was used to detect targets within an environment, to simulate robots detecting IEDs. This simplified simulation involved using bright red coloured markers placed in the environment. A simple image processing algorithm was then added to the robots control algorithm, which ran at the beginning of each time-step. Pseudo-code of the algorithm is given in Algorithm 8.2. The algorithm would simply output whether or not a target had been detected at that given time-step. If a target had been detected, the control law would switch to the 'target surround' behaviour.

Algorithm 8.2: Pseudo-code for the on-board target detection image processing.

```
Read input image
Apply RGB red filter to input image
Threshold filtered image
Apply blob filter for largest red feature
if Target blob > threshold then
    Target detected
else
    No target detected
end if
```

8.7 Appendix G: Publications

The following publications contain research carried out for this thesis.

- R. J. Mullen, S. Barman, P. Remagnino and D. Monekosso - **Towards autonomous robot swarms for multi-target localisation and monitoring with applications to counter IED operations** in *International Journal of Intelligent Defence Support Systems*, 4(1) Inderscience Publishers, January, pp. 87-107, (2011).

- R. J. Mullen, D. N. Monekosso, S. A. Barman and P. Remagnino - **Reactive coordination and adaptive lattice formation in mobile robotic surveillance swarms** in *10th International Symposium on Distributed Autonomous Robotics Systems*, 1 Nov - 3 Nov 2010, Lausanne, Switzerland, (2010).

- R. J. Mullen, D. N. Monekosso, S. A. Barman and P. Remagnino - **Autonomous control laws for mobile robotic surveillance swarms** in *IEEE International Symposium on Computational Intelligence for Security and Defense Applications*, 8 July - 10 July 2009, Ottawa, Canada, pp. 104-109, (2009).

- R. J. Mullen, D. Monekosso, S. Barman and P. Remagnino - **A review of ant algorithms** in *Expert Systems with Applications*, 36(6) Elsevier, August, pp. 9608-9617, (2009).

- R. J. Mullen, D. Monekosso, S. Barman, P. Remagnino and P. Wilkin - Chapter **Artificial ants to extract leaf outlines and primary venation patterns** in *Ant Colony Optimization and Swarm Intelligence*, Lecture Notes in Computer Science, Edited by Dorigo, M.; Birattari, M.; Blum, C.; Clerc, M.; Stutzle, T.; Winfield, A. F. T., Springer, pp. 251-258. (2008).

Bibliography

[1] Beekman, M., Sword, G.A., Simpson, S.J.: Biological foundations of swarm intelligence. In Blum, C., Merkle, D., eds.: Swarm Intelligence. Natural Computing Series. Springer Berlin Heidelberg (2008) 3–41

[2] Bonabeau, B., Dorigo, M., Theraulaz, G.: Swarm Intelligence: From Natural to Artificial Systems. Oxford University Press, New York, NY (1999)

[3] Engelbrecht, A.P.: Fundamentals of Computational Swarm Intelligence. Wiley (2005)

[4] Reynolds, C.: Flocks, herds and schools: A distributed behavioral model. In: Proceedings of the 14th Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH '87). (1987) 25–34

[5] Azahar, M., Sunar, M., Daman, D., Bade, A.: Survey on real-time crowds simulation. In Pan, Z., Zhang, X., El Rhalibi, A., Woo, W., Li, Y., eds.: Technologies for E-Learning and Digital Entertainment. Volume 5093 of Lecture Notes in Computer Science. Springer Berlin / Heidelberg (2008) 573–580

[6] Mao, T., Jiang, H., Xia, S., Wang, Z.: Recent advances on human crowd simulation. International Journal of Virtual Reality **9**(3) (2010) 27–32

[7] Vande Moere, A., Lau, A.: In-formation flocking: An approach to data visualization using multi-agent formation behavior. In Randall, M., Abbass, H., Wiles, J., eds.: Progress in Artificial Life. Volume 4828 of Lecture Notes in Computer Science. Springer Berlin / Heidelberg (2007) 292–304

[8] Kennedy, J., Eberhart, R.: Particle Swarm Optimization. In: Proceedings of the 1995 IEEE International Conference on Neural Networks. (1995) 1942–1948

[9] Bautu, A., Bautu, E.: Particle swarms in statistical physics. In Lazinica, A., ed.: Particle Swarm Optimisation. (2009)

[10] Papadimitriou, C., Steiglitz, K.: Combinatorial optimization – algorithms and complexity. New York: Dover (1982)

[11] Dorigo, M., Maniezzo, V., Colorni, A.: Ant System: Optimization by a colony of cooperating agents. IEEE Transactions on Systems, Man, and Cybernetics - Part B **26** (1996) 29–41

[12] Dorigo, M., Di Caro, G.: The Ant Colony Optimization metaheuristic. New Ideas in Optimization (1999) 11–32

[13] Dorigo, M., Di Caro, G., Gambardella, M.: Ant algorithms for discrete optimization. Artificial Life **5**(2) (1999) 137–172

[14] Kirkpatrick, S., Gelatt Jr, C.D., Vecchi, M.P.: Optimization by simulated annealing. Science **220** (1983) 671–680

[15] Glover, F.: Tabu search - part i. ORSA Journal on Computing **1**(3) (1989) 190–206

[16] Glover, F.: Tabu search - part ii. ORSA Journal on Computing **2**(1) (1990) 4–32

[17] Lourenco, H.R., Martin, O., Stuzle, T.: Iterated local search. Handbook of Metaheuristics **57** (2002) 321–353

[18] Grasse, P.P.: La reconstruction du nid et les coordinations interindividuelles chez bellicositermes natalensis et cubitermes sp. la theorie de la stigmergie: Essai dinterpretation du comportement des termites constructeurs. Insectes Sociaux **6** (1959) 41–81

[19] Deneubourg, J.L., Aron, S., Goss, S., Pasteels, J.M.: The self-organizing exploratory pattern of the argentine ant. Journal of Insect Behavior **3** (1990) 159

[20] Goss, S., Aron, S., Deneubourg, J.L., Pasteels, J.M.: Self-organized shortcuts in the argentine ant. Naturwissenschaften **76** (1989) 579–581

[21] Cordon, O., Herrera, F., Stutzle, T.: A review on the ant colony optimization metaheuristic: Basis, models and new trends. Mathware and Soft Computing **9**(2-3) (2002) 141–175

[22] Dorigo, M., Gambardella, L.M.: Ant Colony System: A cooperating learning approach to the travelling salesman problem. IEEE Transactions on Evolutionary Computation **1**(1) (1997) 53–66

[23] Maniezzo, V., Colorni, A., Dorigo, M.: The Ant System applied to the quadratic assignment problem. Technical Report IRIDIA/94-28 (1994)

[24] Colorni, A., Dorigo, M., Maniezzo, V., Trubian, M.: Ant System for job-shop scheduling. JORBEL - Belgian Journal of Operations Research, Statistics and Computer Science **34**(1) (1994) 39–53

[25] Schoonderwoerd, R., Holland, O., Bruten, J., Rothkrantz, L.: Ant-based load balancing in telecommunication networks. Adaptive Behavior **5**(2) (1996) 169–207

[26] Di Caro, G., Dorigo, M.: AntNet: Distributed stigmergetic control for communications networks. Journal of Artificial Intelligence Research **9** (1998) 317–365

[27] Bullnheimer, B., Hartl, R.F., Strauss, C.: An improved Ant System algorithm for the vehicle routing problem. Annals of Operations Research **89** (1999) 319–328

[28] Gambardella, L.M., Taillard, E.D., Agazzi, G.: MACS-VRPTW: A multiple ant colony system for vehicle routing problems with time windows. New Ideas in Optimization (1999) 63–76

[29] Costa, D., Hertz, A.: Ants can colour graphs. Journal of the Operational Research Society **48** (1997) 295–305

[30] Lessing, L., Dumitrescu, I., Stutzle, T.: A comparison between ACO algorithms for the set covering problem. In: ANTS'2004, Fourth Inter-

national Workshop on Ant Algorithms and Swarm Intelligence, Springer Verlag (2004) 1–12

[31] Gambardella, L.M., Rizzoli, A.E., Oliverio, F., Casagrande, N., Donati, A.V., Montemanni, R., Lucibello, E.: Ant colony optimization for vehicle routing in advanced logistic systems. In: MAS 2003 - International Workshop on Modelling and Applied Simulation, Bergeggi, Italy (2003) 3–9

[32] Rizzoli, A.E., Casagrande, N., Donati, A.V., Gambardella, L.M., Lepori, D., Montemanni, R., Pina, P., Zaffalon, M.: Planning and optimisation of vehicle routes for fuel oil distribution. In: MODSIM 2003 - Integrative Modelling of Biophysical, Social and Economic Systems for Resource Management Solutions, Townsville, Australia (2003) 2024–2029

[33] Gambardella, L.M., Dorigo, M.: Solving symmetric and asymmetric TSPs by ant colonies. In: Proceedings of the 1996 IEEE International Conference on Evolutionary Computation (ICEC'96), IEEE Press, Piscataway, NJ (1996) 622–627

[34] Socha, K.: ACO for continuous and mixed-variable optimization. In: Ant Colony Optimization and Swarm Intelligence, 4th International Workshop, ANTS 2004, Springer Verlag (2004) 25–36

[35] Socha, K., Blum, C.: An ant colony optimization algorithm for continuous optimization: application to feed-forward neural network training. Neural Computing and Applications **16**(3) (2007) 235–247

[36] Talbi, E.G., Roux, O., Fonlupt, C., Robillard, D.: Parallel ant colonies for combinatorial optimization problems. In: Parallel and Distributed Processing, 11 IPPS/SPDP'99 Workshops. Volume 1586. (1999) 239–247

[37] Manfrin, M., Birattari, M., Stutzle, T., Dorigo, M.: Parallel ant colony optimization for the travelling salesman problem. In: Proceedings of ANTS 2006, Springer Verlag (2006) 224–234

[38] Ramos, V., Almeida, F.: Artificial ant colonies in digital image habitats - a mass behaviour effect study on pattern recognition. In: Proceedings of ANTS'2000 - 2nd International Workshop on Ant Algorithms (From Ant Colonies to Artificial Ants). (2000) 113–116

[39] Fernandes, C., Ramos, V., Rosa, A.C.: Self-regulated artificial ant colonies on digital image habitats. Int. Journal of Lateral Computing **2**(1) (2005) 1–8

[40] Nezamabadi-pour, H., Saryazdi, S., Rashedi, E.: Edge detection using ant algorithms. Soft Computing **10** (2006) 623–628

[41] Ma, L., Tian, J., Yu, W.: Visual saliency detection in image using ant colony optimisation and local phase coherence. Electronic Letters **46** (2010) 1066–1068

[42] Chialvo, D., Millonas, M.M.: How swarms build cognitive maps. In Steels, L., ed.: The Biology and Technology of Intelligent Autonomous Agents. Springer-Verlag (1995) 439–450

[43] Rauch, E.M., Millonas, M.M., Chialvo, R.: Pattern formation and functionality in swarm models. Physics Letters A **207** (1995) 185–193

[44] Hamada, M.: Virtual environment for an ants-like model. The International Journal of Virtual Reality **9**(4) (2010) 61–66

[45] Mitchell, T.M.: Machine Learning. McGraw-Hill (1997)

[46] Monekosso, N.D., Remagnino, P.: Phe-Q: A pheromone based Q-learning. In: AI 2001: Advances in Artificial Intelligence: 14th International Joint Conference on Artificial Intelligence, Adelaide, Australia, Springer-Berlin (2001) 1611–3349

[47] Monekosso, N.D., Remagnino, P.: The analysis and performance evaluation of the pheromone-Q-learning algorithm. Expert Systems **21**(2) (2004) 80–91

[48] Watkins, C.J.C.H.: Learning with delayed rewards. PhD thesis, Psychology Department, University of Cambridge, England (2001)

[49] Holland, J.H.: Adaptation in natural and artificial systems. MIT Press, Cambridge, MA, USA (1992)

[50] White, T., Pagurek, B., Oppacher, F.: ASGA: Improving the ant system by integration with genetic algorithms. In: Programming 1998: Proceedings of the Third Annual Conference, University of Wisconsin, Madison, Wisconsin, USA, Morgan Kaufmann (1998) 610–617

[51] Pilat, M.L., White, T.: Using genetic algorithms to optimize ACS-TSP. In: ANTS '02: Proceedings of the Third International Workshop on Ant Algorithms, Springer-Verlag (2002) 282–287

[52] Gambardella, L.M., Dorigo, M.: Ant-Q: a reinforcement learning approach to the travelling salesman problem. In: Proceedings of ML-95, Twelfth International Conference on Machine Learning, Tahoe City, CA, Morgan Kaufman (1995) 252–260

[53] Dorigo, M., Gambardella, L.M.: A study of some properties of Ant-Q. In: PPSN IV: Proceedings of the 4th International Conference on Parallel Problem Solving from Nature, Springer-Verlag (1996) 656–665

[54] Mitchell, M.: An introduction to genetic algorithms. MIT Press (1996)

[55] Dorigo, M., Sahin, E.: Swarm robotics - special edition editorial. Autonomous Robots **17**(2-3) (2004) 111–113

[56] Bayindir, L., Sahin, E.: A review of studies in swarm robotics. Turk. J. Elec. Engin. **15** (2007) 115–147

[57] Sahin, E., Girgin, S., Bayindir, L., Turgut, A.E.: Swarm robotics. In Blum, C., Merkle, D., eds.: Swarm Intelligence. Natural Computing Series. Springer Berlin Heidelberg (2008) 87–100

[58] Winfield, A.F.T., Nembrini, J.: Safety in numbers: fault-tolerance in robot swarms. Int. J. Modelling, Identification and Control **1** (2005) 30–37

[59] Bjerknes, J.D., Winfield, A.F.T.: On fault tolerance and scalability of swarm robotic systems. In: Proceedings of the International Symposium on Distributed Autonomous Robotic Systems (DARS 2010). (2010)

[60] Sauter, J.A., Matthews, R., Parunak, H., Brueckner, S.A.: Performance of digital pheromones for swarming vehicle control. In: Proceedings of the

fourth international joint conference on Autonomous agents and multiagent systems. (2005) 903–910

[61] Garnier, S., Tache, F., Combe, M., Grimal, A., Theraulaz, G.: Alice in pheromone land: An experimental setup for the study of ant-like robots. In: Proceedings of the 2007 IEEE Swarm Intelligence Symposium (SIS 2007). (2007) 37–44

[62] Fujisawa, R., Dobata, S., Kubota, D., Imamura, I., Matsuno, F.: Dependency by concentration of pheromone trail for multiple robots. In: Proceedings of ANTS'2008 - 6th International Workshop on Ant Algorithms, Brussels, Belgium (2008) 283–290

[63] Fujisawa, R., Imamura, I., Matsuno, F.: Cooperative transportation by swarm robots using pheromone communication. In: Proceedings of DARS'2010 - 10th International Symposium on Distributed Robotics Systems, Lausanne, Switzerland (2010)

[64] Sahin, E.: Swarm robotics: From sources of inspiration to domains of application. In Sahin, E., Spears, W.M., eds.: Swarm Robotics. Volume 3342 of Lecture Notes in Computer Science. Springer Berlin / Heidelberg (2005) 10–20

[65] Payton, D., Estkowski, R., Howard, M.: Compound behaviors in pheromone robotics. Robotics and Autonomous Systems **44**(3-4) (2003) 229 – 240

[66] Khatib, O.: Real-time obstacle avoidance for manipulators and mobile robots. The International Journal of Robotics Research **5**(1) (1986) 90–98

[67] Spears, W.M., Spears, D.F., Hamann, J.C., Heil, R.: Distributed, physics-based control of swarms of vehicles. Autonomous Robots **17** (2004) 137–162

[68] Morgan, D.S., Schwartz, I.B.: Dynamic coordinated control laws in multiple agent models. Physics Letters A **340** (2005) 121–131

[69] Labonte, G.: Canadian Arctic Sovereignty: Local Intervention by Flocking UAVs. In: Proceedings of the 2009 IEEE Symposium on Computational Intelligence for Security and Defence Applications (CISDA 2009). (2009)

[70] Turgut, A.E., Celikkanat, H., Gokce, F., Sahin, E.: Self-organized flocking in mobile robot swarms. Swarm Intelligence **2** (2008) 97–120

[71] Manh, H., Sheng, W.: Adaptive flocking control for dynamic target tracking in mobile sensor networks. In: Proceedings of the 2009 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2009). (2009) 4843–4848

[72] Antonelli, G., Arrichiello, F., Chiaverini, S.: The entrapment/escorting mission - an experimental study using a multirobot system. IEEE Robotics and Automation Magazine (2008) 22–29

[73] Kumar, M., Milutinovic, D., Garg, D.P.: Role of stochasticity in self-organisation of robotic swarms. In: Proceedings of the 2008 American Control Conference. (2008)

[74] Winfield, A.F.T., Liu, W., Nembrini, J., Mertinoli, A.: Modelling a wireless connected swarm of mobile robots. Swarm Intelligence **2** (2008) 241–266

[75] Brooks, R.: A robust layered control system for a mobile robot. Robotics and Automation, IEEE Journal of **2**(1) (January 1986) 14–23

[76] Mataric, M.J.: Designing emergent behaviors: from local interactions to collective intelligence. In: Proceedings of the second international conference on From animals to animats 2 : simulation of adaptive behavior, Cambridge, MA, USA, MIT Press (1993) 432–441

[77] Liu, W., Winfield, A., Sa, J., Chen, J., Dou, L.: Strategies for energy optimisation in a swarm of foraging robots. In Sahin, E., Spears, W., Winfield, A., eds.: Swarm Robotics. Volume 4433 of Lecture Notes in Computer Science. Springer Berlin / Heidelberg (2007) 14–26

[78] Kwok, N.M., Ha, Q.P., Ngo, V.T., Hong, S.M.: Particle Swarm Optimization-based coordination of a group of construction vehicles. In: Proceedings of the 23th International Symposium on Automation and Robotics in Construction (ISARC 2006). (2006) 840–845

[79] Ngo, V.T., Kwok, N.M., Ha, Q.P.: Initialization of robotic formations using discrete particle swarm optimization. In: Proceedings of the 24th

International Symposium on Automation and Robotics in Construction (ISARC 2007). (2007) 355–360

[80] Pugh, J., Martinoli, A.: Inspiring and modeling multi-robot search with particle swarm optimization. In: Proceedings of the 4th IEEE Swarm Intelligence Symposium (SIS 2007). (2007) 332–339

[81] Hereford, J., Siebold, M.: Multi-robot search using a physically-embedded particle swarm optimization. International Journal of Computational Intelligence Research 4(2) (2008) 197–209

[82] Nehmzow, U.: Learning in multi-robot scenarios through physically embedded genetic algorithms. In: Proceedings of the seventh international conference on simulation of adaptive behavior on From animals to animats. ICSAB, Cambridge, MA, USA, MIT Press (2002) 391–392

[83] Smart, W., Pack Kaelbling, L.: Effective reinforcement learning for mobile robots. In: Robotics and Automation, 2002. Proceedings. ICRA '02. IEEE International Conference on. (2002)

[84] Barrios-Aranibar, D., Alsina, P.: Imitation learning: An application in a micro robot soccer game. In Nedjah, N., Coelho, L., Mourelle, L., eds.: Mobile Robots: The Evolutionary Approach. Volume 50 of Studies in Computational Intelligence. Springer Berlin / Heidelberg (2007) 201–219

[85] Kong, C.S., Peng, N.A., Rekletis, I.: Distributed coverage with multi-robot system. In: Proceedings of the 2006 IEEE International Conference on Robotics and Automation (ICRA 2006). (2006) 2423–2429

[86] Hsieh, M.A., Halasz, A., Berman, S., Kumar, V.: Biologically inspired redistribution of a swarm of robots among multiple sites. Swarm Intelligence 2 (2008) 121–141

[87] Ludwig, L., Gini, M.: Robotic swarm dispersion using wireless intensity signals. In Gini, M., Voyles, R., eds.: Distributed Autonomous Robotic Systems 7. Springer Japan (2006) 135–144

[88] Pakanati, A., Gini, M.: Swarm dispersion via potential fields, leader election, and counting hops. In Ando, N., Balakirsky, S., Hemker, T., Reggiani,

M., von Stryk, O., eds.: Simulation, Modeling, and Programming for Autonomous Robots. Volume 6472 of Lecture Notes in Computer Science. Springer Berlin / Heidelberg (2010) 485–496

[89] Mclurkin, J., Smith, J.: Distributed algorithms for dispersion in indoor environments using a swarm of autonomous mobile robots. In: 7th International Symposium on Distributed Autonomous Robotic Systems (DARS 2004). (2004)

[90] Gordon, N., Elor, Y., Bruckstein, A.: Gathering multiple robotic agents with crude distance sensing capabilities. In Dorigo, M., Birattari, M., Blum, C., Clerc, M., Sttzle, T., Winfield, A., eds.: Ant Colony Optimization and Swarm Intelligence. Volume 5217 of Lecture Notes in Computer Science. Springer Berlin / Heidelberg (2008) 72–83

[91] Soysal, O., Bahceci, E., Sahin, E.: Aggregation in swarm robotic systems: Evolution and probabilistic control. Turk. J. Elec. Engin. **15** (2007) 199–225

[92] Dimarogonas, D., Kyriakopoulos, K.: Connectedness preserving distributed swarm aggregation for multiple kinematic robots. Robotics, IEEE Transactions on **24**(5) (2008) 1213 –1223

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

[94] Nouyan, S., Dorigo, M.: Chain based path formation in swarms of robots. In Dorigo, M., Gambardella, L., Birattari, M., Martinoli, A., Poli, R., Sttzle, T., eds.: Ant Colony Optimization and Swarm Intelligence. Volume 4150 of Lecture Notes in Computer Science. Springer Berlin / Heidelberg (2006) 120–131

[95] Kernbach, S., Meister, E., Schlachter, F., Jebens, K., Szymanski, M., Liedke, J., Laneri, D., Winkler, L., Schmickl, T., Thenius, R., Corradi, P., Ricotti, L.: Symbiotic Robot Organisms: REPLICATOR and SYMBRION projects. In: Proc. of Performance Metrics for Intelligent Systems Workshop (PerMIS 08), Gaithersburg, USA (2008)

[96] Navarro, I., Pugh, J., Martinoli, A., Matia, F.: A distributed scalable approach to formation control in multi-robot systems. In: Proceedings of the International Symposium on Distributed Autonomous Robotic Systems (DARS 2008). (2008)

[97] Antonelli, G., Arrichiello, F., Chiaverini, S.: The entrapment/escorting mission for a multi-robot system: Theory and experiments. In: Advanced intelligent mechatronics, 2007 IEEE/ASME international conference on. (2007) 1 –6

[98] Gazi, V.: Swarm aggregations using artificial potentials and sliding-mode control. IEEE Transactions on Robotics **21** (2005) 1208–1214

[99] Gazi, V., Fidan, B., Hanay, Y., Koksal, M.: Aggregation, foraging, and formation control of swarms with non-holonomic agents using potential functions and sliding mode techniques. Turk. J. Elec. Engin. **15** (2007) 149–168

[100] Wessnitzer, J., Adamatzky, A., Melhuish, C.: Towards self-organising structure formations: A decentralized approach. In Kelemen, J., Sosk, P., eds.: Advances in Artificial Life. Volume 2159 of Lecture Notes in Computer Science. Springer Berlin / Heidelberg (2001) 573–581

[101] Ouadfel, S., Batouche, M.: Unsupervised image segmentation using a colony of cooperating ants. In: BMCV '02: Proceedings of the Second International Workshop on Biologically Motivated Computer Vision, Springer-Verlag (2002) 109–116

[102] Channa, A.H., Rajpoot, N.M., Rajpoot, K.M.: Texture segmentation using ant tree clustering. In: 2006 IEEE International Conference on Engineering of Intelligent Systems. (2006) 1–6

[103] Hickey, L.J.: Classification of the architecture of dicotyledonous leaves. American Journal of Botany **60**(1) (1973) 17–33

[104] : Royal Botanic Gardens, KEW. http://www.kew.org (Accessed January 2011)

[105] Wilkin, P. personal communication (February 2008) Royal Botanic Gardens, KEW, London, England.

[106] Chittka, L., Muller, H.: Learning, specialization, efficiency and task allocation in social insects. Communicative and Integrative Biology **2**(2) (2009) 151–154

[107] Arcaute, E., Christensen, K., Sendova-Franks, A., Dahl, T., Espinosa, A., Jensen, H.J.: Division of labour in ant colonies in terms of attractive fields. Ecological Complexity **6**(4) (2009) 396–402

[108] Reznikova, Z.I., Novgorodova, T.A.: The importance of individual and social experience for interaction between ants and symbiotic aphids. Doklady Biological Sciences **359** (1998) 173–175

[109] Canny, J.: A computational approach to edge detection. IEEE Trans. Pattern Analysis and Machine Intelligence **8** (1986) 679–698

[110] Calderone, N.W., Page, R.E.: Temporal polyethism and behavioural canalization in the honey bee, apis mellifera. Anim. Behav. **51** (1996) 631–643

[111] Rozenberg, G., Back, T., Blum, C., Eiben, A.E., Kok, J.N., Merkle, D., Spaink, H.P.: Swarm Intelligence: Introduction and Applications. Springer-Verlag, Berlin (1999)

[112] Fernandes, C., Ramos, V., Rosa, A.: Varying the population size of artificial foraging swarms on time varying landscapes. In Duch, W., Kacprzyk, J., Oja, E., Zadrozny, S., eds.: Artificial Neural Networks: Biological Inspirations - ICANN 2005. Volume 3696 of Lecture Notes in Computer Science. Springer Berlin / Heidelberg (2005) 311–316

[113] Martin, D., Fowlkes, C., Tal, D., Malik, J.: A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. In: Proc. 8th Int'l Conf. Computer Vision. Volume 2. (July 2001) 416–423

[114] Mondada, F., Bobani, M., Raemy, X., Pugh, J., Cianci, C., Klaptocz, A., Magnenat, S., Zufferey, J.C., Floreano, D., Martinoli, A.: The e-puck, a robot designed for education in engineering. In: Proceedings of the 9th Conference on Autonomous Robot Systems and Competitions. (2009) 59–65

[115] Mullen, R.J., Monekosso, D., Barman, S., Remagnino, P.: A review of ant algorithms. Expert Systems with Applications **36** (2009) 9608–9617

[116] Antonelli, G., Arrichiello, F., Chiaverini, S.: Flocking for multi-robot systems viathenull-space-based behavioral control. Swarm Intelligence **4** (2010) 37–56

[117] Gerkey, B., Vaughan, R.T., Howard, A.: The Player/Stage project: Tools for multi-robot and distributed sensor systems. In: Proceedings of the 11th International Conference on Advanced Robotics (ICAR 2003). (2003) 317–323

[118] Griffiths, S., Saunders, J., Curtis, A., McLain, T., Beard, R.: Obstacle and terrain avoidance for miniature aerial vehicles. IEEE Robotics and Automation Magazine **13**(3) (2006) 34–43

[119] Murphy, D., Cycon, J.: Applications for mini VTOL UAV for law enforcement. In: Proceedings of SPIE Sensors, C3I, Information, and Training Technologies for Law Enforcement. (1999) 35–43

[120] : Dragonflyer UAV. http://www.draganfly.com/uav-helicopter/draganflyer-x6/ (Accessed February 2011)

[121] : Microdrones VTOL UAV. http://www.mwpower.co.uk/uav/microdrones-vtol-uav.html (Accessed February 2011)

[122] : Skybotix Technologies. http://www.skybotix.com/ (Accessed February 2011)

[123] Lerman, K., Martinoli, A., Galstyan, A.: A review of probabilistic macroscopic models for swarm robotic systems. In: Swarm Robotics Workshop: State-of-the-art Survey. (2005) 143–152

[124] Stan, S., Radu, B., Maties, V.: Multi-objective design optimization of mini parallel robots using genetic algorithms. In: IEEE International Symposium on Industrial Electronics. (2007) 2173–2178

[125] Pinciroli, C., Birattari, M., Tuci, E., Dorigo, M., del Rey Zapatero, M., Vinko, T., Izzo, D.: Lattice formation in space for a swarm of pico satellites.

In Dorigo, M., Birattari, M., Blum, C., Clerc, M., Sttzle, T., Winfield, A., eds.: Ant Colony Optimization and Swarm Intelligence. Volume 5217 of Lecture Notes in Computer Science. Springer Berlin / Heidelberg (2008) 347–354

[126] U.S. Department of State: Hidden killers: the global demining crisis. Technical report, U.S. Dept. State, Washington, DC (1998)

[127] Zorpette, G.: Countering IEDs. IEEE Spectrum **45**(9) (2008) 23–32

[128] Cumming, C.J., Aker, C., Fisher, M., Fox, M., la Grone, M.J., Reust, D., Rockley, M.G., Swager, T.M., Towers, E., Williams, V.: Using novel fluorescent polymers as sensory materials for above-ground sensing of chemical signature compounds emanating from buried landmines. IEEE Transactions on Geoscience and Remote Sensing **39**(6) (2001) 1119–1128

[129] : QinetiQ Talon Robot. http://www.qinetiq-na.com/products-talon.htm (Accessed February 2011)

[130] : iRobot LANdroids Robot. http://www.irobot.com/gi/research/Advanced_Platforms/LANdroids_Robot (Accessed February 2011)

[131] Mullen, R.J., Monekosso, D., Barman, S., Remagnino, P.: Autonomous control laws for mobile robotic surveillance swarms. In: Proceedings of the 2009 IEEE Symposium on Computational Intelligence in Security and Defence Applications (CISDA 2009). (2009)

[132] Aicardi, M., Casalino, G., Bicchi, A., Balestrino, A.: Closed loop steering of unicycle-like vehicles via Lyapunov techniques. IEEE Robotics and Automation Magazine **2**(1) (1995) 27–35

[133] Michel, O.: Webots: Professional mobile robot simulation. Journal of Advanced Robotics Systems (2004)

[134] Gutierrez, A., Campo, A., Dorigo, M., Amor, D., Magdalena, L., Monasterio-Huelin, F.: An open localization and local communication embodied sensor. Sensors **8** (2008) 7545–7563

[135] Balch, T., Hybinette, M.: Social potentials for scalable multi-robot formations. In: Proceedings of the 2000 IEEE International Conference on Robotics and Automation (ICRA 2000). (2000)

[136] Gross, R., Magnenat, S., Mondada, F.: Segregation in swarms of mobile robots based on the brazil nut effect. In: Proceedings of the 2009 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2009). (2009)

[137] Gutierrez, A., Campo, A., Dorigo, M., Donate, J., Monasterio-Huelin, F., Magdalena, L.: Open e-puck range and bearing miniaturized board for local communications in swarm robotics. In: Proceedings of the 2009 IEEE International Conference on Robotics and Automation. (2009)

[138] Pugh, J., Raemy, X., Favre, C., Falconi, R., Martinoli, A.: A fast onboard relative positioning module for multirobot systems. IEEE/ASME Transactions on Mechatronics **14** (2009) 151–162

[139] Ballerini, M., Cabibbo, N., Candelier, R., Cavagna, A., Cisbani, E., Giardina, I., Lecomte, V., Orlandi, A., Parisi, G., Procaccini, A., Viale, M., Zdravkovic, V.: Interaction ruling animal collective behaviour depends on topological rather than metric distance: Evidence from a field study. Proc. Natl. Acad. Sci. USA (PNAS) **105**(4) (2008) 1232–1237