UNIVERSITY OF SOUTHAMPTON

FACULTY OF PHYSICAL AND APPLIED SCIENCES

Electronics and Computer Science

Hardware Variation in Robotic Swarm and Behavioural Sorting with Swarm Chromatography

by

Beining SHANG

Thesis for the degree of Doctor of Philosophy

June 23, 2017

UNIVERSITY OF SOUTHAMPTON

ABSTRACT

FACULTY OF PHYSICAL AND APPLIED SCIENCES
Electronics and Computer Science

Doctor of Philosophy

HARDWARE VARIATION IN ROBOTIC SWARM AND BEHAVIOURAL SORTING WITH SWARM CHROMATOGRAPHY

by Beining SHANG

Social insects can achieve remarkable outcomes, various examples can be found in ants, bees, etc. Inspired by social insects, swarm robotic research considers coordinating a group of relatively simple and autonomous robots to finish tasks collaboratively based on direct or indirect interactions. Such systems can offer advantages of robustness, flexibility and scalability, just like social insects.

For many years, various researchers have endeavoured to design intelligent artificial swarms and many hardware-based swarm robots have been implemented. One assumption that made by a majority of swarm robotic researchers, particularly in software simulation is that a robotic swarm is a group of identical robots, there is no difference between any two of them. However, differences among hardware robots are unavoidable, which exist in robotic sensors, actuators, etc. These hardware differences, albeit small, can affect the robots' response to the environment. Moreover, hardware differences can provoke robots' heterogeneity which then profoundly influence swarm performance due to the non-linearity in the controller and uncertainty in the environment. Nevertheless, questions about how hardware differences influence swarm performance and how to make use of them remain a research challenge.

In this work, the issue of hardware variation in swarm robots is investigated. Specifically swarm robots with hardware variations are modelled and simulated in a line following scenario. It is found that even small hardware variations can result in behavioural heterogeneity. Although the variations can be compensated by the software controller in training, the hardware variations and resulting differences in training are amplified in the interactions between the robot and the environment.

To know how exactly hardware variation influence robotic behaviours, a novel approach, inspired by the chromatography method in chemistry, is proposed to sort swarm robots according to their hardware circumstances. This method is based on a large number of interactions between robots and the environment. Individual robot's unique hardware circumstance determines its unique decision making and reaction during each robotic controlling step, and these unique

microscopic reactions accumulate and contribute to the robot's macroscopic behaviour. The behavioural sorting results show that the behaviour of an individual robot is not determined by a single parameter but by the combination of multiple hardware factors. Different combinations of hardware parameters can help robots achieve similar behaviours.

The efficiency of the behavioural sorting method is investigated, particularly the influence of the robot's controller and environmental factor. By simulating various combinations of robots with different integration lengths of the controller and arenas with different pattern densities, it is discovered that if the robots' ability to memorise previous events is coupled with the density of the sorting arena, better sorting results can be achieved.

This work is regarded as an initial investigation into the issue of unavoidable hardware differences between swarm robots. Given the research outcome and that real swarms will necessarily show hardware variations, it is therefore necessary to contemplate current swarm algorithms in the context of diverse robot populations. In addition, a new research field of swarm chromatography for sorting robotic behaviours to improve swarm efficiency is initiated.

Contents

List of Figures

List of Tables

List of Symbols

K_i The integral gain of the robot PI controller. 31, 32, 39, 48, 85

K_p The proportional gain of the robot PI controller. 31, 32, 39, 48, 85

O_l IR Sensor lateral offset: pointing biased towards left or right. 30, 31, 34, 46

O_s IR Sensor saggital offset: pointing biased towards front or back. 30, 31, 34, 46

S IR sensor response to single reflective point/pixel. 28, 29

V_0 The constant voltage used to drive robot forward. 32

$V_{IR\_L}(t)$ The voltage output of left IR sensor at time t. 31, 32

$V_{IR\_R}(t)$ The voltage output of right IR sensor at time t. 31, 32

V_{IR} IR Sensor output voltage. 29, 52

$V_{PI}(t)$ The output voltage of the robot PI controller at time t. 31, 32, 85

$V_{ml}(t)$ The voltage fed to robot left motor drive. 32, 33

$V_{mr}(t)$ The voltage fed to robot right motor drive. 32, 33

α IR sensor sensitivity. 28, 29, 31, 34, 52

β IR Sensor output offset. 29, 31, 52

$\delta(t)$ The voltage difference between left and right IR sensor at time t. 31, 32, 85

$\dot{\phi}$ Changes in robot orientation. 33

$\dot{x}$ Changes in robot x coordinate. 33

$\dot{y}$ Changes in robot y coordinate. 33

$\measuredangle v$ IR sensor viewing angle. 28, 29, 31, 34, 52

$\omega_L(t)$ The angular speed of robot left wheel at timestep t. 33

$\omega_R(t)$ The angular speed of robot right wheel at timestep t. 33

θ Incidence angle of the reflective light. 28, 52

b The distance between robot's two wheels. 33, 34

h IR Sensor height, the vertical distance between IR sensor and the ground. 30, 31, 34

m_L Left motor drive gain. 33, 34

m_R Right motor drive gain. 33, 34

ml The integration length: the number of errors to be integrated in the robot's controller. 84, 85

r_L Radius of robot left wheel. 33, 34

r_R Radius of robot right wheel. 33, 34

x Distance between IR sensor and the reflective point. 28, 52

Declaration of Authorship

I, Beining SHANG , declare that the thesis entitled *Hardware Variation in Robotic Swarm and Behavioural Sorting with Swarm Chromatography* and the work presented in the thesis are both my own, and have been generated by me as the result of my own original research. I confirm that:

- this work was done wholly or mainly while in candidature for a research degree at this University;
- where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated;
- where I have consulted the published work of others, this is always clearly attributed;
- where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work;
- I have acknowledged all main sources of help;
- where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself;
- parts of this work have been published and listed in Section 1.6.

Signed:.....

Date:.....

Acknowledgements

I would like to thank all of the people who encouraged and supported me during the undertaking of my research.

First and foremost, no words can fully express my gratitude to my two supervisors. I would like to thank my supervisor Dr. Richard Crowder for providing me this opportunity to work with him on this amazing topic. I would never forget the moments of detailed discussions, corrections of my writings and the significant amount of support he provided throughout my Ph.D. I am sincerely grateful to my second supervisor Dr. Klaus-Peter Zauner, who has offered me not only constant support, encouragement on research but also the guidance in everyday life. Without him, it would be impossible to finish this research work. The life experiences and advice he shared opened my mind and will continuously inspire me to explore further, try harder. For this, I am truly grateful.

Furthermore, I would like to thank all of the members of the Agents, Interaction and Complexity group at the University of Southampton who were of great support throughout my Ph.D. It has been a pleasure to work and socialise with such a great team. Special thanks to Shaofei Chen, Dengji Zhao, Chetan Mehra and Evangelos Tolias, without whom, these Ph.D. years would not be so fascinating.

Finally, I dedicate this thesis to my parents and my brother, who believed in me, offered guidance and supported my decisions no matter what. Those shared moments of both happiness and sadness will never be forgotten. The encouragement all along the way will continuously shine light on my road to the future.

Chapter 1

Introduction

Within the natural world, insects can achieve remarkable results, for example, termites can build large and complex mounds (Lüscher, 1961), army ants organise impressive foraging raids (Rettenmeyer, 1963), or honey bees build a series of parallel combs. These insects can collectively accomplish complex tasks beyond individual capabilities. Inspirations are drawn to swarm robot research which considers how to design a group of robots which can work collectively to finish specific tasks. Adopting a group of relatively simple, collectively working robots may offer many advantages in efficiency, fault-tolerance and cost per system (Bonabeau et al., 1999). Because of these advantages, numerous attempts have been seen (Valdastri et al., 2006; Hauert et al., 2008a; Spears et al., 2009; Zhang et al., 2013; Hilder et al., 2016; Patil et al., 2016; Novischi and Florea, 2016) to develop robotic swarms.

One of the important concepts in designing swarm robot is the heterogeneity and homogeneity of the swarm. In nature, heterogeneity is discussed by Beshers and Fewell (2001):

- *Each worker specialises in a subset of the complete repertoire of tasks performed by the colony*
- *This subset varies across individual workers in the colony.*

Many examples can be found in nature: in a bee colony, some bees specialise various specific tasks, like food foraging, building, attending to offspring, and so on (Bonabeau et al., 1999). This is often because that specialists consume less time and energy and are therefore more proficient in specific tasks comparing with generalists, thus improving the performance of the whole colony. Such examples inspire robotic research, especially robotic swarms, because both social insects and swarm robots are controlled in a decentralised manor and individual decisions are made based on local perceptions. In robotic swarms, different robots' allocating themselves in different tasks can result in benefits like increasing energy efficiency, higher parallelism and reduction of interference between workers (Şahin, 2005).

In robotic swarms, heterogeneity describes the scenario that difference or diversity can be found between any two robots in a swarm (Potter et al., 2001). Homogeneity means robots in the swarm are identical. To study this systematically, current research of collaborative robots is firstly categorised in Fig 1.1. Based on whether robots are made intentionally different, collaborative robots can be separated into robot ecosystems[1] and swarms. In a robot ecosystem, robots which usually are different in their size, functions, etc. are adopted. Successful implementations have been reported in Parker (1994) and Dorigo et al. (2013).

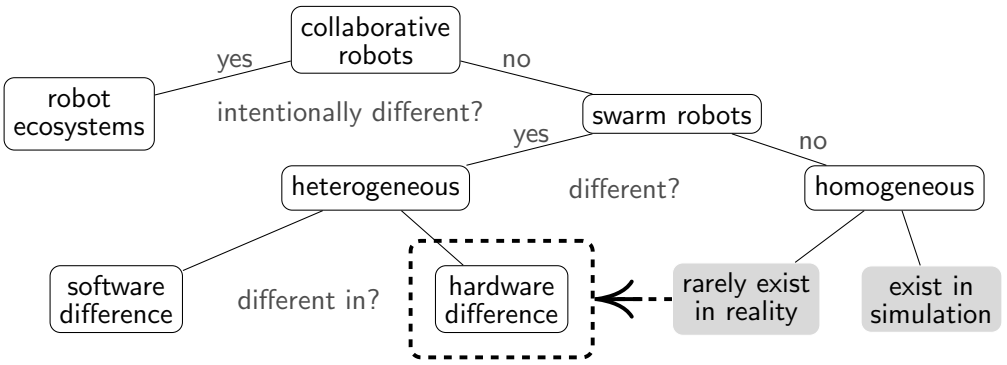


Figure 1.1: Collaborative Robots Categorization: Based on whether robots are made intentionally different, collaborative robots can be separated into robot ecosystems (intentionally different) and swarms (intentionally similar). For swarm robots, heterogeneous robots are the ones which are different in either software (for instance, different control or software strategies or just different parameters in the software controller) or hardware (variation in their hardware. For example, two robots' motors have different driving characteristics.). If all robots in a swarm are identical, they are homogeneous. In simulations a large group of homogeneous swarm robots can be created easily. However, homogeneous robots are hardly found in reality because hardware differences are unavoidable.

In typical swarm systems, robots are usually made to the same design and similar in their shapes, functions and abilities. To be specific, robots in a swarm are usually made of same type of modules including sensors, actuators, batteries, platforms, etc. According to whether differences exist between any two robots in the swarm, swarm robots can be separated into heterogeneous and homogeneous swarms. For instance, robots can have different software procedures comparing with others (Bongard, 2007; Pugh and Martinoli, 2007). On the other hand, they can use the same software procedure, but values of the parameters in the software are different.

In addition, there is another factor which can also differentiate robots in a swarm: the hardware differences. The hardware differences refer to components variations, uncertainty generated from the assembly process and different wear and tear conditions during using. In practice it is impossible to avoid these differences. Therefore except in simulations where identical robots can be duplicated, it is very difficult to find homogeneous robots in reality because of the hardware differences like component variations, assembly uncertainty, wear and tear, etc.

[1] Although in some research, this type of robots with different shapes and functions is called swarm robot. However it is more accurate to categorise them as robot ecosystem.

1.1 Shortcomings of Existing Swarm Robot Research

In the current research of robotic swarms, one assumption made by majority of the researchers is that individuals in swarm robotic system are identical and the existence of hardware difference has been neglected. To be specific, in simulation-based research, simulated robots in a swarm has no difference comparing with their kins in terms of the hardware. In addition, in hardware-based research, robots in a swarm are used with the assumption that their hardware are exactly the same.

The assumption is contradictory to the problem that researchers face in practice. In the field of general robots, hardware differences such as sensors' different sensitivities, actuators' different driving characteristics, etc. cause problems (Koestler and Bräunl, 2004; Malheiros et al., 2009). To ensure robotic system running smoothly, efforts of compensating the hardware differences have to be made (Roth et al., 1987; Lobo and Dias, 2007; Alici and Shirinzadeh, 2006). Therefore in swarm robotic swarms, it can not be denied that hardware differences will influence robotic behaviours.

On the other hand, one should admitted that if the software controller of the swarm robots is robust enough to allow robots to cope with the types of hardware difference, swarm robots with homogeneous behaviour can still be created. However, it is rather difficult and takes time and efforts to design such software controller or algorithm (Elliott and Shadbolt, 2003). Thus, compensating hardware difference using software approach is not a viable solution in robotic swarms.

Additionally, hardware variation is intrinsic and within current manufacture technology, it is almost impossible to get rid of it completely, while compensating hardware difference would cost an unwarranted amount of effort and time. Thus for a period of time, it will continuously influence the design, implementation and practical use of robot swarms.

1.2 Motivations and Challenges

The highlighted shortcoming of the current swarm robotic research acts as a motivation for this research. It is important to investigate the issue of hardware difference in the course of swarm robots and understand more of the role which hardware difference plays in terms of robotic behaviours. This is beneficial to the current swarm robotic research as hardware difference commonly exists. If hardware difference indeed influences the behaviour of robot to a level that the hardware difference can not be ignored, it would be necessary to improve current swarm algorithm, as it is assumed that all robots in a swarm are identical.

Additionally, if the behaviour of robot is diverse due to hardware difference, resulting heterogeneous behaviours of the robot, algorithms can be designed to make use of such diverse behaviour by allocating task accordingly, which will further improve the task efficiency of the swarm.

This work is an initial investigation into the issue of unavoidable hardware difference between swarm robots, which will provide not only a better understanding of the issue itself, but also a possible starting point of a new research field.

1.3 Research Scope and Aims

According to the previous discussion, hardware differences can be categorised into three types in Figure 1.2. This work will only focus on the first two parts, thus hardware variation. There are two reasons for neglecting the damage and deterioration part: firstly hardware variation is the first problem encountered after production, therefore it should be solved first. Secondly damage and deterioration can be considered as hardware variation which arises when the robot is used, and once the first two problems are solved, the problem of hardware difference caused by damage and variation could be solved using the same approach.

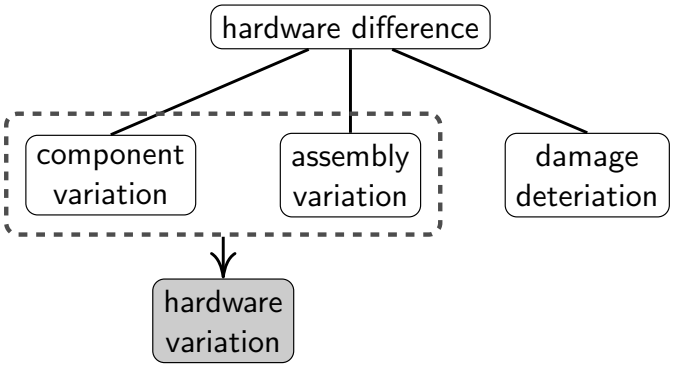


Figure 1.2: Categorization of Hardware Difference: Hardware difference generally originates: when components are manufactured (components variation), when robots are assembled (assembly variation) and when robots are used (damage and deterioration). This work will focus on the first two phases (the dashed area), namely hardware variation.

Therefore this work mainly focuses on heterogeneous swarm robots with identical hardware modules, among which some hardware variation could be found between different swarm members.

The aim of this research is:

1. **Investigate how much the behaviours of swarm robots are influenced by hardware variation.**
2. **Understand how hardware variation influences robotic behaviours.**
3. **Design a technique which sorts robots according to their unique behaviours caused by hardware variation.**

1.4 Research Contributions

Against the research aims identified above, this work makes the following contributions:

- **Hardware variations and its influence**: Hardware variations do exist in hardware-based robotic swarms, which is however ignored by most of swarm robotic researchers. This is the first time that the issue of hardware variation is addressed and it is found out that hardware variation of swarm robots can influence robotic behaviours. It is also the first time that the idea of making use of hardware variation to select favourable robotic behaviours in order to improve robots' performance is brought.
- **Simulation of hardware variations**: By utilising the approach of simulating a typical swarm robot in a line-following task, it is demonstrated that these intrinsic hardware variations did influence behaviours of an individual robot in the swarm. It is also investigated that how robotic behaviours were influenced under different magnitudes of hardware variation, what type of effect could be caused by the variation of an individual component, and the relationship between robotic hardware circumstance and its behaviours.
- **Swarm chromatography**: A novel behavioural sorting technique called swarm chromatography was proposed. The method of differentiating the robots through the accumulated effect of numerous interactions with the environment is analogous to separating chemical mixtures by chromatography. This method is robust that the sorting of the robots does not depend on other parameters but only on the hardware characteristics of individual robots. In addition, the sorting efficiency was investigated in related with the configuration of robot's memory and sorting arena.

1.5 Thesis Structure

The rest of the thesis is organised as follows:

The current literature of swarm robotic research was reviewed in Chapter 2 with particular focus on three important aspects: the systematic modelling, behavioural design and robotic interactions. In addition, the concept of heterogeneity of swarm robots was addressed and major features which cause swarm robots heterogeneous were identified as well as how researchers make use of this phenomenon. Based on the review of previous research, the gap in the current research was identified that most of the researchers concentrate on the software procedures which cause robotic swarm heterogeneous, however the intrinsic hardware variation can not be ignored.

Based on the findings, the existence of hardware variation in swarm robots was firstly consolidated in Chapter 3, and the argument was then laid out that although hardware variations are

small, they can still influence robotic behaviours. In order to test the argument, the methodology was proposed which utilises a typical swarm robot with varied parameters for modelling the hardware variation to accomplish a simple line-following task. The detailed modelling of the robotic system, task configuration and how the simulation is organised were also presented.

In order to find out if hardware difference influence robotic behaviours, robots with difference circumstances of hardware variation was simulated in a line-following scenario and robots' trajectories were compared in Chapter 4. Results showed that hardware variation indeed influences robotic behaviours. A number of scenarios with decreased magnitude of hardware variation were also tested, it was found that tiny hardware variation could still make an impact in terms of trajectories generated by the robots.

To understand how exactly hardware variation influence robotic behaviours, the relationship between robotic behaviours and its hardware circumstance was investigated in Chapter 5 with the proposed swarm chromatography technique. The efficiency of the technique was further investigated in relation to the configuration of robot's controller and the sorting arena in Chapter 6. The conclusion was drawn in Chapter 7.

1.6 Publications

The contributions of the research lead to the following publications:

- Beining Shang, Richard Crowder and Klaus-Peter Zauner. (2013). Simulation of Hardware Variations in Swarm Robots. In *IEEE International Conference on Systems, Man, and Cybernetics*, pages 4066-4071, Manchester, UK
- Beining Shang, Richard Crowder and Klaus-Peter Zauner. (2014). Swarm Behavioral Sorting based on Robotic Hardware Variation. In *The 5th International Conference on Simulation and Modeling Methodologies, Technologies and Application*, pages 631-636, Wien, Austria.
- Beining Shang, Richard, Crowder and Klaus-Peter Zauner. (2016). An Approach to Sorting Swarm Robots to Optimize Performance. In *Proceedings of ASME 2016 International Design Engineering Technical Conferences*, pages 1-8, Charlotte, North Carolina.

Chapter 2

Related Work

In this chapter, related work for swarm robots is discussed. To avoid misunderstanding of the main topic, collaborative robots is firstly categorized and the area of this research is restricted within the field of swarm robot (Section 2.1). The current research of swarm robot is then described (Section 2.2) and topics covered here are three essential aspects: systematic modelling, robotic behavioural design and the robotic interactions, which serves as the foundations of this research.

In Section 2.3, heterogeneous swarm robots are discussed. By identifying the origins of behavioural heterogeneity, it is found out that heterogeneity for swarm robots generally emerge due to either software or hardware. Limited studies concentrate on the hardware aspects which trigger behavioural heterogeneity. Finally, Section 2.4 summarizes.

2.1 Collaborative Robots and Research Scope

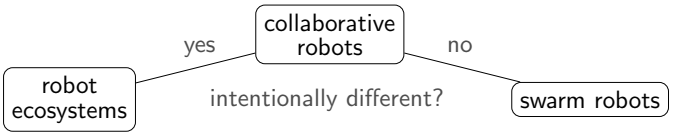
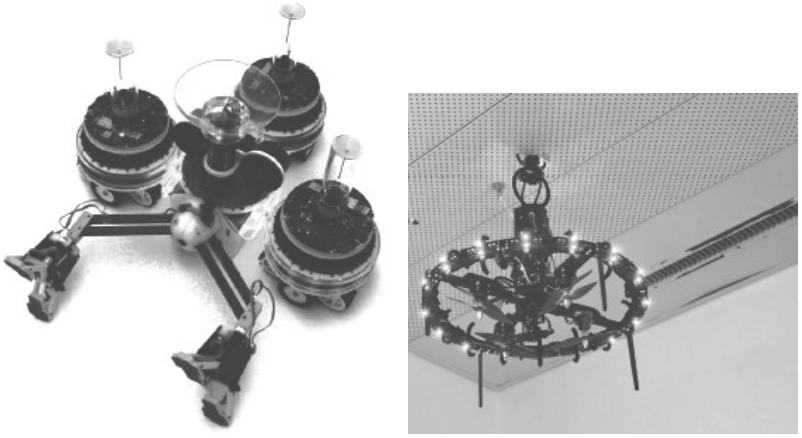


Figure 2.1: Collaborative Robots Categorization: Based on that whether robots are made intentionally different, collaborative robots can be separated into robot ecosystems and swarms, where the former are made intentionally different in the size, functionalities, etc.; the latter are made intentionally similar and made to the same design.

The categorization of current collaborative robotic research is illustrated in Figure 2.1. Based on that whether robots are made intentionally different, collaborative robots can be separated into robot ecosystems and swarms. In a robot ecosystem, robots which usually are different in their size, functions are used. For instance, Dorigo et al. (2013) adopted three types of robot illustrated in Fig 2.2 to create a system called the *swarmanoid* to explore the physical and behavioural interactions between different robot types. These collaborative robots can take advantage of

different functionalities to accomplish complex tasks. In Dorigo et al. (2013)'s case, mapping of the environment is easily done by the *eye-bot*, a smaller size quad-copter, which can fly above the ground. Anther example would be that Parker (1994) used two types of mobile robots different in their mechanical structure, sensors and actuators, illustrated in Fig 2.3 to achieve fault tolerant cooperation.



(a) Three footbots and One Handbot (b) One Eyebot

Figure 2.2: Dorigo's Heterogeneous Robots (Dorigo et al., 2013): In (a), a *hand-bot* in the middle is surrounded by three *foot-bots*. The *hand-bot* is equipped with two arms, and there is a clamp at each end, which makes it capable of holding objects. Lacking in mobility, the *hand-bot* can only move with help from three *footbots*. In (b), an *eye-bot* is attached to the ceiling. It is equipped with a camera pointing below, which makes it capable of retrieving information from the ground.

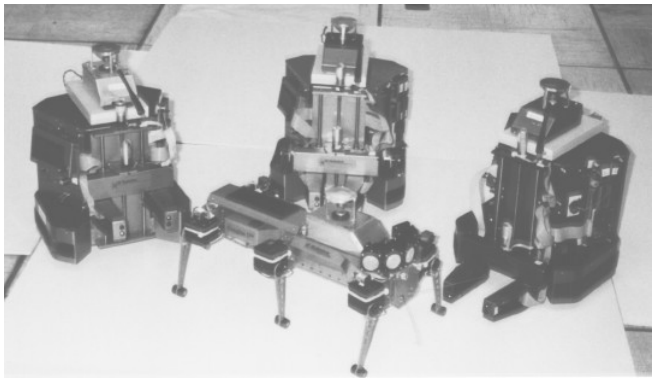


Figure 2.3: Parker's Heterogeneous Robots (Parker, 1994): There are two types of robots, three R-2 robots at the rear and one Genghis-II in the front.

Different from the robotic ecosystem, swarm robots are similar as they are made to the same design. As robotic swarms are usually mass-produced for cost consideration, it is normal that the number of robots in a swarm can reach up to hundreds or thousands, showing Figure 2.4. Such system offers many advantages in task efficiency, fault tolerance, cost per system (Bonabeau et al., 1999).

In this work, we will concentrate on the swarm robotic systems. In such systems, robots are usually similar regarding their shapes, functions and abilities. To be specific, all robots in the

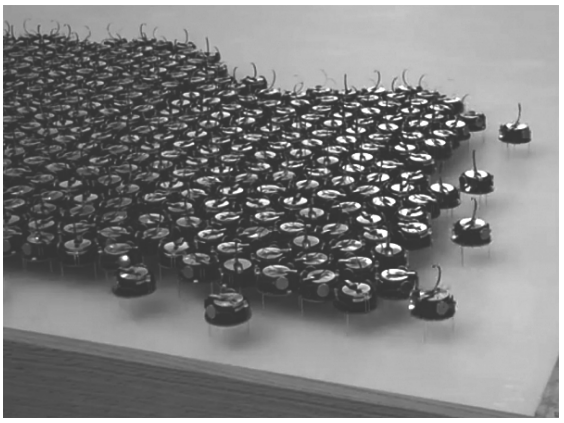


Figure 2.4: The Robotic Swarm Consisting Hundreds of Individuals (Rubenstein et al., 2014)

same swarm are usually made of same types of modules including sensors, actuators, batteries, platforms, etc.

2.2 Current Research Picture in Swarm Robots

The research for swarm robots is about designing a decentralised robotic system consisting a number of similar individuals which can accomplish task collectively. As it is decentralised controlled, the functioning of the system relies largely on the interactions between individuals and the environment. Therefore, the following review of currently swarm robotic research will be conducted in three aspects: systematic modelling, robotic behavioural design and interaction methods.

2.2.1 Modelling Swarm Robotic Systems

Modelling helps the researchers to gain a better understanding of the system and simulation is just another word for modelling. There are two advantages for using simulation in swarm robot research:

- Different behaviours can be applied or tested on simulated swarm robots easily and result can be obtained quickly.
- Simulated swarm robots can be created and managed easily without worrying about the hardware issues (device malfunction, battery recharge, etc.) which can become very timing-consuming in hardware-based swarm robotic research if a large number of robots are used.

However it can not be ignored that there are differences between simulation and actual experiments which are caused by random or systematic differences that exist in hardware-based experiments, robot entities are abstractly modelled in simulation etc. (Jacobi, 1997).

Depending on how detailed a robotic system is modelled, modelling approach in swarm robotic research can be categorized into the following three types: sensor-based, microscopic and macroscopic modelling, which will be discussed in the following. Figure 2.5 summarizes these three types of simulation methods.

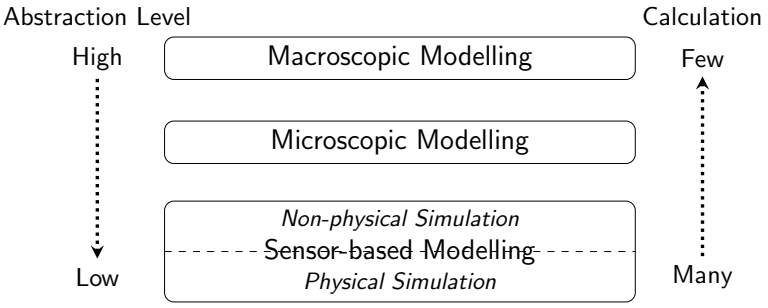


Figure 2.5: Summary of Modelling Methods: Macroscopic model models robotic swarm at high level and physical simulation uses the lowest-level model. Calculations of macroscopic model are very few and physical simulation needs many calculations.

Sensor-based Modelling

Sensor-based modelling is a modelling method, in which sensors and actuators of each robot, usually as well as objects in the simulated environment are all modelled. The sensor's output is generated according to the environment which robot is currently in. Generated output data is then sent to controller which then outputs commands to be executed by actuators. Thus, interactions are modelled. Depending on whether physical properties of objects are described or not, simulations based on this kind of modelling method can be categorized in to physical and non-physical simulations.

- **Physical Simulation**
  Physical simulation using sensor-based models models the interactions of the robots and the environment based on physical rules of our actual world by assigning physical properties to the objects including the mass and the motor torque required to move the robots. It is obvious that this simulation process is much complex, which often requires high computation capacity (Bahçeci and Şahin, 2005; Soysal and Şahin, 2005). It is worth noting that parallel simulation methods are used over multiple computers which are connected by local area network (LAN) to overcome the complexity of simulations in this kind (Trianni and Dorigo, 2005; Trianni et al., 2005).

- **Non-physical Simulation**
  Non-physical simulation uses a rather simple model for interactions. Since physical properties are not assigned to objects in this simulation, the dynamics of the robots and the

objects in the environment are ignored and they are considered as objects without physical properties, for instance, adding just some logic values to eliminate collisions. Examples can be found in Balch and Hybinette (2000); Howard et al. (2002a,b); Hayes and Dormiani Tabatabaei (2002); Trianni et al. (2002). Comparing with physical simulation, high computation capacity is not required.

Microscopic Modelling

Microscopic modelling method models each robot and their interactions mathematically. In this kind, robots are defined to have different states. Robot's state can be changed to another, according to both internal and external events inside the robots and in the environment. Probabilities are assigned to transitions of their states. Thus, the system behaviour and the noise in the environment are easily integrated into these probabilistic models.

At each simulation step, the probabilities of the state transitions are calculated. If a generated random numbers between 0 and 1 are lower than calculated probability, this transition is likely to occur at this time and the state of robot changes. However extra attention is needed while designing probabilities of the state transition. A well-designed set of probabilities indicates better behaviours of each robot and better behaviours of the whole swarm system. Examples can be seen in Jeanson et al. (2005); Martinoli and Easton (2002); Ijspeert et al. (2001); Martinoli et al. (2004).

Macroscopic Modelling

In macroscopic model, system behaviour is defined with differential equations. The average number of robots in a particular state at a certain time step is represented by variables of the differential equation.

Comparing with the microscopic model which models each robot, macroscopic models the whole behaviour of the system directly. Therefore system behaviour is obtained once after the model is solved. However system behaviour can not be obtained until all robots' states are obtained in microscopic model.

In Martinoli et al. (2004); Lerman et al. (2004), probabilities are applied to the system state transitions which can handle noise in a simple way which is very similar to that in microscopic models.

2.2.2 Behavioural Design for Swarm Robots

Adaptation is any change in the structure of an object or its function in order to survive more effectively in the environment (Bayindir and Şahin, 2007) which is one of important characteristics for robots behaviour design. In the research of swarm robots, adaptation not only refers to

behaviour of individual robot but also refers to behaviour of the whole swarm robots for accomplishing a task collectively. Based on robots' adaptation ability and time scale, behaviours can be categorized into the following three kinds: non-adaptive (manual), learning and evolution.

Both learning and evolution behaviours have adaptation ability, manual does not. The difference between learning and evolution is length of period which the whole swarm has used to adapt itself to different circumstances. This is the same in biology. Evolution often takes generations, however learning based behaviours can be fine-tuned as soon as new knowledge is obtained.

Non-adaptive Behaviour

This category of robot's behaviour is rather simple, which usually features following characteristics:

- Robot's states are finite and predefined.
- Either one or both of "state changing sequences" and "state changing conditions" are predefined or fixed.

Most non-adaptive behaviours can be implemented using the following three architectures including subsumption, probabilistic finite state automata and distributed potential field approach, which will be discussed in the following.

- **Subsumption**
  Brooks (1985) firstly published the subsumption approach, actions of robot is defined into different layers from lowest to the highest. All layers can access to its sensor data and generate commands for actuators. Under such circumstances, low layer actions always have high priorities than high layer actions. In other words the low layer outputs are always been executed first and high layer outputs are inhibited if generated commands are contradictory with low layer outputs. This architecture ensures that overall goal can be achieved while low layer action can still function timely and correctly.

  Examples can be found in Hristoskova et al. (2011), in the scenarios of modern computer game, player can lead a squad consisting of several unintelligent robots which are just capable of navigation and shooting. Complex squad commands (overall goal) can be decomposed into several commands (output from each layer), of which one and only one command is selected by an 'Arbitrator' to executed by actuators (layer priorities).

- **Probabilistic Finite State Automata**
  In this method, robot's behaviours are defined as several discrete states. Different state have different probabilities for robot to switch into. State transition is triggered by either internal/external events or randomly. At each unit of time, a random number is generated and compared with this probabilities. If generated number is smaller than certain probability, this state is the one that robot needs to switch into.

These probabilities can change based on robot task finishing situation. For instance, if robot is instructed to perform searching-then-homing tasks. Initially probability for robot to return home is very low. If robot can not find specified target, this probability of homing will increase as time goes by, which means there might be no target nearby and it is unnecessary to search more and consume powers. Then at beginning of next unit of time, if homing probability is larger than the generated random number, robot will return home (Labella et al., 2004). Applications using similar approach can also be found in Liu et al. (2007).

- **Distributed Potential Field**
  This approach is mainly about adopting virtual forces into a group of robot functioning as a swarm. The area in which such forces are effective is called potential field. Each robot has its own fields, namely distributed potential field. An attraction force is generated to an object if it is far and a repulsion force is generated if it is too near to the robots. The ultimate goal of the swarm also generates an attraction force which applies to each of the swarm members. Output of each robot will be obtained through the calculation of these forces.

  In Hashimoto et al. (2008), the swarm consisting several robots can follow, surround a human and maintain stability while the human moves in obstacle environment. Each robot calculates its own virtual force from attraction and repulsion from its neighbouring robots. In addition a surrounding force is generated based on distance to the human to be maintained, which can be regarded as the ultimate goal of the swarm.

Behaviour with Learning Ability

Based on whether an external supervisor exists or not, learning algorithms can be categorized into supervised learning and unsupervised learning. In the research of swarm robots, it is difficult to have a human as a supervisor while robots are out in the field to finish tasks. Therefore most of learning algorithm in swarm robotic research are unsupervised learning algorithms.

On the other hand, there are two type of learning signals: local reinforcement signal and global reinforcement signal. The former signal is only effective within the robot itself while the latter is usually applied to the whole swarm. Both Matarič (1997) and Li et al. (2004) explicitly studied how these two types of signals influence the swarm performance. Furthermore, there are also variations like combining reinforcement learning method with neural network controller (Kuremoto and M. Obayashi, 2009; Conforth and Meng, 2010).

Evolutionary Behaviour

In this approach, suitable controller of swarm robots are selected through a process which is similar to that how genes have been selected in biology. Such process is often implemented as

genetic algorithm, in which least-fit solutions/robot controllers are eliminated through cycles of crossover and mutation operation, leaving only the best-fit ones. Genetic algorithm consists of several steps (Jakobi et al., 1995), as illustrated in Fig. 2.6:

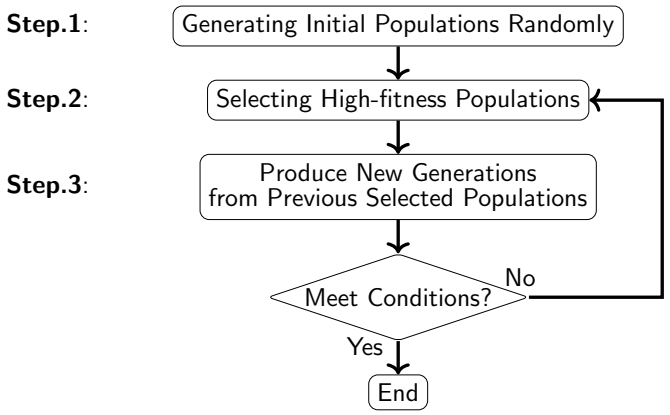


Figure 2.6: Genetic Algorithm Process: Typical Process consists of three steps. Firstly, the first generation of population is randomly generated. Those which have high fitness scores will be selected at Step.2. New generations will be produced at Step.3 from these selected populations by the means of genetic operations such as crossover or mutation. Conditions (for example, number of iterations, fitness etc.) will be checked to terminate the process. If not, the process will move to Step.2, populations will be selected again for high-fitness ones. In the end, only individuals with highest fitness scores will survive.

In swarm robotic research, initially a large amount of number sequences are randomly generated which can be considered as an abstract expression of robots' controller. A fitness function is designed to be able to select expressions of best-fit controllers. These expressions are then crossed over and mutated to generate new generations which often share characteristics of their parent controllers. These steps are repeated until best-fit expressions are found. While implementing, the following points are critical:

- Transformations between controller and abstracted expressions.
- How fitness functions which select good-fit controller are designed.
- Qualities of mutation and crossover operations

One of the problems that evolutionary algorithm has is that at initial stage, a large amount of data is often needed to train the algorithm. And normally, this process is often computation-intensive (Jakobi, 1998), in which CPUs of individual swarm robots can not handle. This requires that evolutionary algorithm are usually trained well on devices with higher processing speed such as a personal computer or mainframe before downloading to swarm robots. Another problem is that since training process usually uses simulations which is not as accurate as physical experiments, trained evolutionary algorithm may behave differently when downloaded to hardware-based robots.

2.2.3 Interactions of Swarm Robots

One of common tasks for swarm robot is to finish activities collaboratively, in which interactions have to happen and information has to be exchanged. The interaction at this point is only limited to that happens between one robot and another.

Based on their information exchange medium and robot's intention for information exchange, such interactions of robotic swarm can be classified into the following three kinds listed in the following:

- Interaction via communication (type 1)
- Interaction via sensing (type 2)
- Interaction via the environment (type 3)

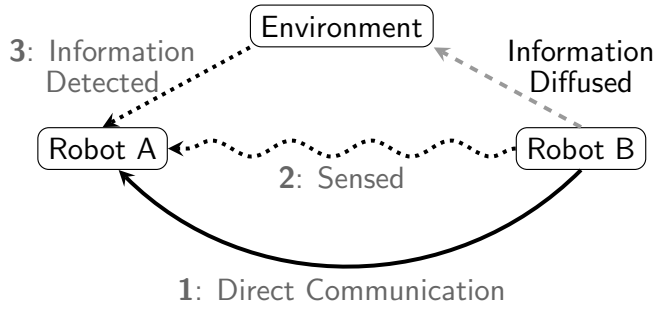


Figure 2.7: Three Interaction Approaches. The solid line means direct communication, usually implemented using radio broadcasting. The dotted lines refer to information interception using sensors which is not capable of direct communication between robots. The red dashed line means information diffusion. Numbers 1, 2, 3 refer to the type of the interaction.

In Fig. 2.7, the scenario is that robot B's information should be transferred to robot A. Robot A can get these information though direct communication from B (approach 1: interaction via communication), or Robot A uses its sensors to obtain such information directly coming from B (approach 2: interaction via sensing), or this information are diffused to the environment and Robot A sensed this information from the environment (approach 3: interaction via environment).

Interaction via Communication (type 1)

In the scenarios of type 1, as numbered in Fig. 2.7, robot shows its information positively to others in order to form a collaboratively-working group. It is different from both "interaction via sensing, type 2" and "interaction via environment, type 3", since robot B positively gives

out information to A by direct communication. Robot B does have the intention to send its information out, where in both approaches 2 and 3, robot does not.

Comparing with "interaction via sensing" and "interaction via environment", this type of information spreading approach is more straight-forward and efficient since information is given out positively to allowed other robot to act accordingly.

This approach is usually implemented using broadcasting or one-to-one communication mechanism. However the latter one requires an unique ID code for each of robots in the swarm, therefore one-to-one communication is not efficient which is almost abandoned due to its reducing the scalability and flexibility of the system. Thus the approach of one-to-one communication will not be discussed.

Broadcasting mechanism can be implemented in different ways. The following are typical:

- Short-range wireless transmission: Hauert et al. (2008b) used radio to maintain distance and communication between nearby micro air vehicles
- Infra-red broadcasting communication Rubenstein et al. (2012) used infra-red light to broadcast information to nearby robots.
- LED light on the robot for indicating information: Mondada et al. (2003) used 24 LEDs to express the state of the robot.

Interaction via Sensing (type 2)

Using approach of type 2, robot's information (like speed, path, distance etc.) is sensed by other robots. In Fig. 2.7, robot A is able to measure through its own sensors information regarding Robot B. At no time does Robot A directly communicate with Robot B. For example, robots which are equipped with sonar sensors are able to detect relative speed and direction of others without direct communication. Once Robot B's information is sensed, A will act accordingly. A typical example is sonar.

In this approach, it is vital that robot must have the ability to discriminate environment and robot, which is also called as kin recognition (Hamilton, 1963).It is an important feature of animals in nature. Animal can act either the same with or different from the behaviour of their kins. This natural process enables the perfect cooperative behaviours. A flock of sardines are chased by shark under the sea. Each one turns to the direction almost the same with that of its nearby kins to hide itself in the flock, resulting in decreasing the probability of being caught (Ward and Hart, 2003). Examples based on kin recognition in swarm robotics are Spears et al. (2004); Turgut et al. (2008); Li et al. (2004); Hayes and Dormiani Tabatabaei (2002); Trianni et al. (2003).

Interaction via the Environment (type 3)

Approach of Type 3 includes the robot swarm, in which direct communication or sensing does not exist between any two robots. Instead, environment acts as the transmitting medium in the information spreading process. Another important factor is added into this standard that the environment must hold or have the ability to maintain the information for a period of time. For example, the environment "memorises" the passage of robot B, while robot A is able to interpret the environment and gain knowledge of the presence of B. A typical example is pheromone-based swarm robots.

The factor of memorization emphasizes the effect of environment during the information spreading process. This category excludes the studies in which information is transmitted instantly by the means of the environment (like infra-red reflection etc.). Since effect of infra-red light emitting is more straight-forward and the environment does not retain the transmitted information for some time, it is more accurate to put these studies into either of the previous two categories.

This category is important because that it is widely used in the nature like pheromone-based ants etc. Pheromones are left by previous ant after finding a food source which serves as the guideline for following ants leading to the food source, known as stigmergy (Deneubourg et al., 1989). Although the communication approach is rather simple, it is difficult to create such an environment in which allows this communication.

However some studies using this approach do exits. In Hauert et al. (2008a), pheromone-based position information is maintained using MAVs (Micro Air Vehicles) via wireless communications. Such virtual pheromone information helps the rest of MAV to maintain a stable communication tunnel between two points. Similar virtual pheromone approach can also be found in Payton et al. (2001).

2.3 Heterogeneous Swarm Robots

As discussed in Chapter 1, heterogeneity in swarm robots can emerge due to either robots' software controller or hardware characteristics. The following review of literatures will be separated into two categories: software-based heterogeneity and hardware-based heterogeneity, particularly it will be focused on the emergence of heterogeneity and its influence to robotic behaviours.

2.3.1 Software-based heterogeneity

Most of researchers have been working on swarm robots with heterogeneous behaviours which are triggered by software approaches. We have categorized current literature relating to swarm robotic behavioural heterogeneity in Fig. 2.8.

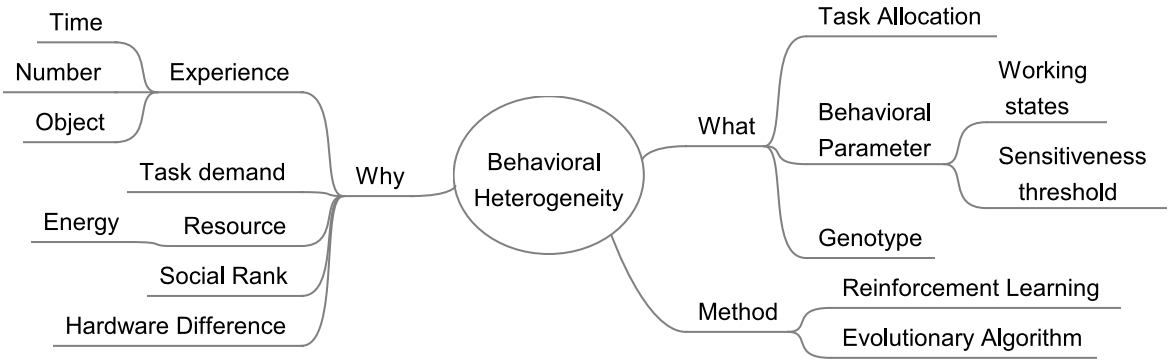


Figure 2.8: Previous research of swarm robotic behavioural heterogeneity: "Why" refers to the reasons which cause robots specialized. They can be their experiences in terms of goal achieving time, number of goal achieved times or object which they have encountered. It can also be the task demands, resources, their social rankings or hardware differences. "What" refers to how robots are specialized. For example, robots specialized in certain task will insist on choosing that type of task. Robots insist in particular working state or robot have different sensitivity or threshold. 'Method' refers to the approaches to achieve behavioural heterogeneity in robotic swarms.

Heterogeneity Emergence

Li et al. (2002, 2004) examines the emergence of heterogeneity and studies the relationship between heterogeneity and swarm efficiency in the stick pulling experiment shown in Fig. 2.9. It is shown that heterogeneity can achieve similar or better performance. A robot pulls an unoccu-

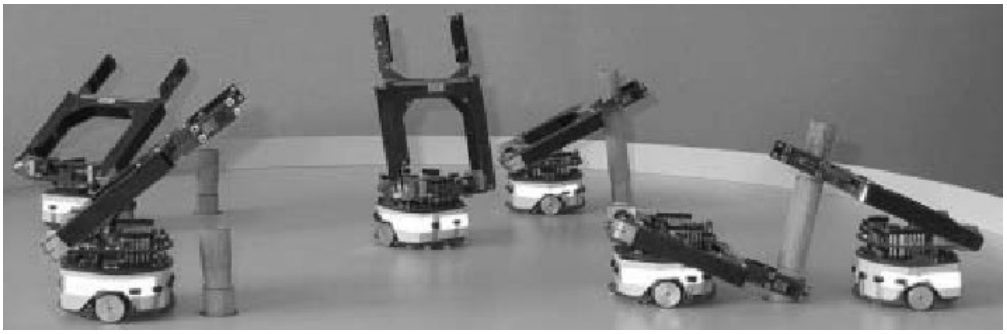


Figure 2.9: Stick Pulling Experiment (Li et al., 2004): Each robot is equipped with a gripper. A number of sticks are placed in the holes randomly located in the arena. The sticks are selected to be long enough that two robots have to work collaboratively to pull it out.

pied stick and waits for a certain period which is defined as the gripping time parameter (GTP). Either a successful collaboration could be achieved for that another robot engages before GTP is over; or GTP times out and the first robot quits pulling and continues to search for new sticks. Therefore GTP is the most important parameter which is strongly related to the stick pulling rate (number of sticks that has been successfully pulled out during a specific period.) GTP is initially set to a predefined number for all robots and is subject to change afterwards. In the first round, each robot randomly choose to either decrease or increase its GTP parameter value. If it

is a successful collaboration in this round, the robot further decrease or increase its maximum waiting time depending on how that robot's GTP changed previously.

Simulation results show that at end of the simulation, robots forms into two cluster: some with large GTP and others have small GTP. Robots in the first cluster specialize to keep holding the stick for a long time and others are rather impatient and keep shifting around, with which the group performance is improved.

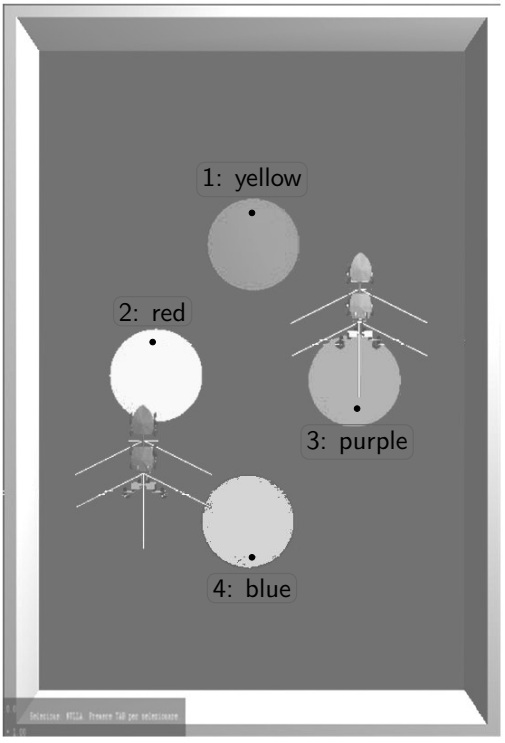


Figure 2.10: Arena of Colour Sensing Heterogeneity (Arena et al., 2012): Number 1, 2, 3, 4 indicate the order of the targets which robots visit. In this scenario, a blue target will appear first. Only after it has been approached by any of the two robots, a yellow target appears in the arena and simultaneously the blue target disappears.

In Arena et al. (2012), heterogeneity can also occurs owing to robots' previous experiences. Robots are set to visit targets in different colours one by one. Whenever a target is visited and no matter what colour the target is, a global signal is broadcast to all members in the swarm, resulting a reward for each robot, which biases robots' reactions upon its previous visited colours. Since robots have visited different colours, robots specialized in finding different colours. This work is done through simulation in an arena in Fig. 2.10.

Task Allocation and Partition

Murciano and Millán (1996); Murciano et al. (1997) present an approach to select optimal distribution of assigning robots to different tasks, illustrated in Fig. 2.11. Robots need to gather a

number of different types of objects in order to assemble complete pieces. Robots adaptively adjust its preference in gathering certain type of objects according to team/individual performance in the past trails. In order to maximize number of complete set of gathered objects which consists one of each type, distribution of robots' preference should be the same or similar to the distribution of different types of objects in the environment. In Zhang et al. (2007), a dynamically

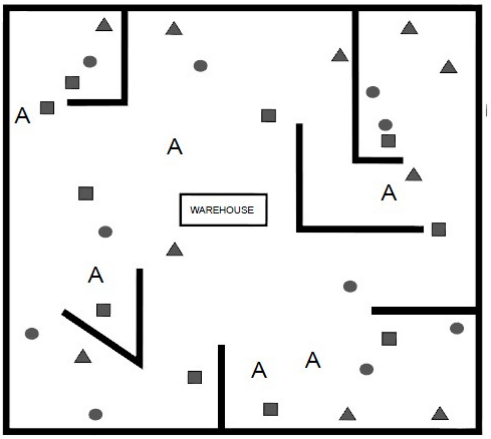


Figure 2.11: Objects Gathering Task (Murciano et al., 1997): Three types of objects, represented by square, circle and triangle, are to be collected and transported to the warehouse located in the middle of the arena. Robots are indicated by 'A'. Black lines in the middle represent obstacles.

adjusted threshold is adopted to evolve specialists for different tasks illustrated in Fig. 2.12. When a particular task is demanded, the threshold for this task decreases, and robots' preferences differentiate. According to task efficiency information on the local blackboard among robots which are doing the same task , the threshold is then re-evaluated: if efficiency is low, some robots will be moved out by increasing task-threshold.

Brutschy et al. (2011) studied the cost and benefit of behavioural heterogeneity. His assumption is that a robot working repeatedly on the same type of task improves its task performance due to learning. Robots prefer to improve more by repeating the same task and probability of choosing this task becomes higher. Since longer distance might be used to search for the same task, robot forgets and efficiency in that task decreases, and probability of choosing that task becomes lower. The robot's state machine for the work is shown in Fig. 2.13. In Brutschy et al. (2011)'s scenario, there are two types of tasks for a group of non-communication robots with this characteristics. Simulation results show that selective strategy helps to achieve better swarm performance for most of the times. By varying the portions of two types of tasks, a decrease is found in swarm performance. It is concluded that "specialization is not a good choice in highly dynamic environments, as specialists may not be able to adapt to changes fast enough".

In Pini et al. (2011), non-communication robots are required to move sufficient amount of objects from source to nest illustrated in Fig. 2.15. Between source area and nest area, there is a cache, in which objects can be dropped by robots on the source side. Objects dropped in the cache can be taken by other robots from nest side and moved to nest. This cache is designed

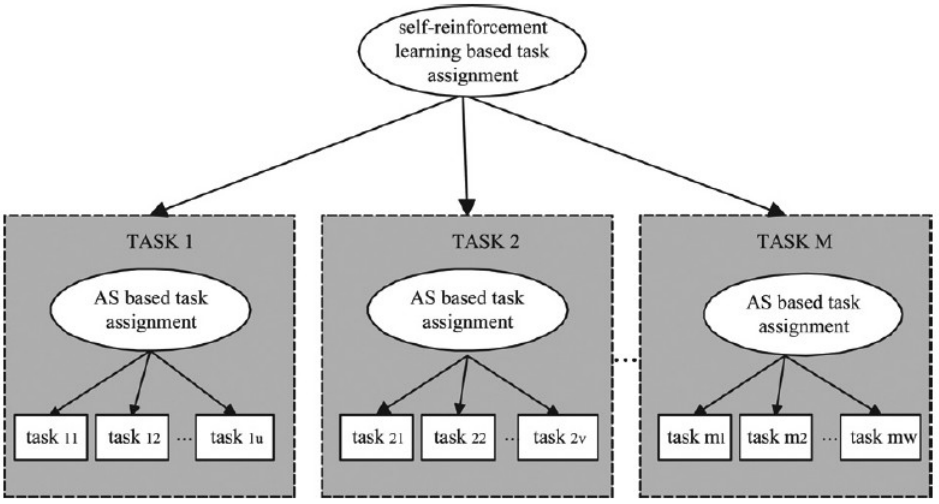


Figure 2.12: Objects Organization Zhang et al. (2007), robots are separated into different groups according to the tasks they have been assigned (TASK 1, TASK 2 and TASK 3). Each task consists of several sub-tasks, for example, TASK 1 can be separated in to task 11, task 12 ... task 1u. A local blackboard system is adopted, on which robots that has been assigned to the same task can exchange their information. 'AS' refers to ant system algorithm.

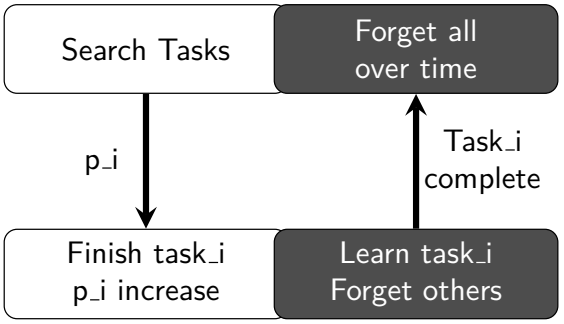


Figure 2.13: State Transition (Brutschy et al., 2011): p_i is robot's probability of choosing $task\_i$. The white rectangles represent actions executed by the robot, dark rectangles show the effect of learning and forgetting on the robot.

that robot can not pass through. There is a corridor which links the source side and nest side for robots to pass through, however extra time must be spent. Each robot's way-selecting process is probabilistic based on time which it previously recorded. Specific timing information is only updated once after the robot finishes the un-partitioned task or sub-task.

This concept is validated through simulation. Performance of the swarm is improved comparing with results of that robots adopt either 'always-partition' or 'never-partition' strategies. By comparing results from experiments using different environment conditions, the author concludes that the swarm robots using this strategy is able to respond to both environment and swarms size changes.

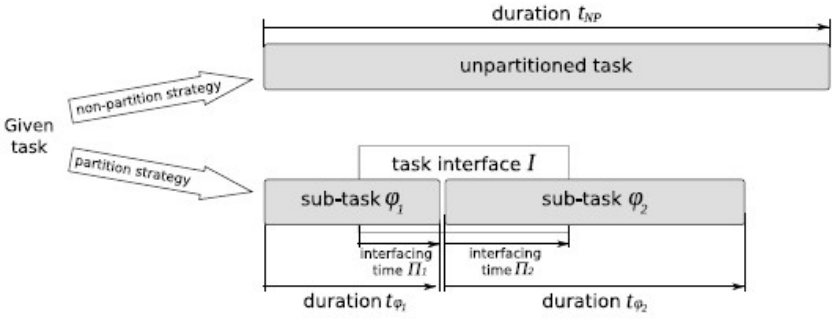


Figure 2.14: Representation of the Sequential Task Partition Problem (Pini et al., 2011)

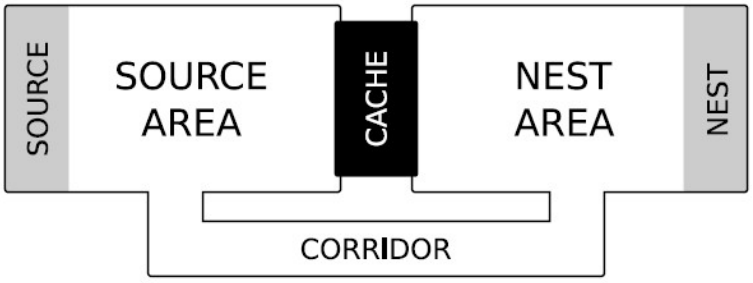


Figure 2.15: Testing Scenario of Task Partition Problem (Pini et al., 2011)

2.3.2 Hardware-based Heterogeneity

Although most of researchers concentrate on the software approaches which make swarm robots behave heterogeneously, some people do realize that hardware variation exist in swarm robots, which can also trigger diverse behaviours. However the number of literatures are limited.

Pugh and Martinoli (2007) study the impact of hardware variation to robotic learning process when using different software controller. In their work, the hardware variation is limited to sensor offsets and scaling factors. It is found in their simulation that in the case of evolving obstacle avoidance, both genetic algorithms and particle swarm optimization are able to withstand small variations in sensor offsets and large variations in sensor scaling factors, while showing poor performance with high offset variations. By observing population diversity throughout evolution, it was discovered that PSO (Particle Swarm Optimization) maintains much higher diversity. The diversity is not caused by the variations of the hardware, it is the intrinsic property of the algorithm.

Elliott and Shadbolt (2003) argue from developmental robotics' point of view that like humans, none of two robots are the same, either due to inter-individual variation or that no two robots experience the same environmental inputs. It is argued that behavioural homogeneous robots can possibly be made which hardware needs to be fine-tuned to reduce the variation to a level which it can be ignored, however this is infeasible in practice and might bring undesirable results.

2.4 Summary

In this chapter, related work in the field of swarm robots was discussed. The current literatures was reviewed through three aspects including system modelling, robotic behavioural design and robotic interactions. Then heterogeneous swarm robots were discussed. By identifying the origins of behavioural heterogeneity, it was found out that heterogeneity for swarm robots generally emerge due to either software or hardware.

The research gap was then identified: the majority of the researchers concentrated on the heterogeneity which is caused by the software and limited research addressed the hardware issue. Therefore in the following chapters, the issue of hardware variation in swarm robots will be systematically investigated, particularly how the hardware variation influences the robotic behaviours.

Chapter 3

Methodology

In this chapter, the model of a typical mobile robot suitable for a swarm undertaking a range of simple tasks is described. The task selected for this research is proposed for investigating the relationships between the robot's hardware variations and the robot's behaviour.

In Section 3.1, hardware variations found in various components at difference stages are identified. It is hypothesized that these hardware variations can affect the behaviour of robots. The model of the robot is described in Section 3.2. Following this, the method of simulating hardware variations on the proposed robotic model is explained in Section 3.3. The line-following task is proposed in Section 3.4 and the arguments for using this type of task are also provided. Finally, simulations challenges for this research and how they are solved are described in Section 3.5.

3.1 Problem Description

In swarms implemented in real-world applications, physical robots are built either by hand or through an automated mass production process. Although they are built to the same design and are often regarded as identical in practice, they are not truly identical because hardware differences exists. Figure 3.1 illustrates some of the reasons which cause variations at hardware level in these robotic swarms.

Differences among swarm robots can emerge when components are manufactured, when robots are assembled and when they are used. An example of components variation would be that the same type of sensors from two robots have different sensitivity (Pugh and Martinoli, 2007). Actuators and batteries can have individual characteristics. For mobile robots, the tyres for the wheels are often made from rubber to improve traction, which makes it rather difficult to manufacture with exactly the same diameter. Furthermore asymmetric load distribution will make tyre compress differently, resulting different wheel diameters in practice (Roth et al., 1987).

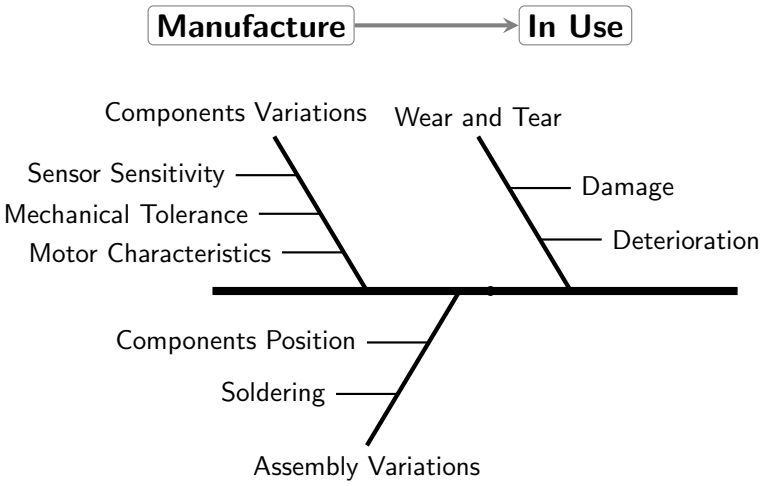


Figure 3.1: The causes of hardware differences: reasons which can cause robots different in their hardware are listed. They are categorized based on robot's life span: when components are manufactured, when the robot is assembled and when it is used. During manufacture, variations exist in sensor sensitivity, motor driving ability, tolerance of mechanical parts, etc. 'Mechanical Tolerance' refers to the permissible limits in a measured value of a manufactured item, such as the length of the axle for robot's wheel. The mechanical tolerance is typically expressed as $X = \pm Y$, where X is the nomial value and Y is the allowable deviation. During assembly, components' placement and soldering parameters varies. When using swarm robots, different damage and deterioration situations are encountered by robots. All of these circumstances are applied to robots' hardware, which differentiate them. This list is not exhausted.

In the assembly phase, positions of components and soldering parameters vary. For instance, sensors can be placed with slightly different orientations during soldering, or the quality of soldered joint may inference the peak current from the drive, and hence limit the motor's output torque. In addition, motor parameters may vary significantly due to temperature, supply frequency and magnetic saturation (Toliyat et al., 2003).

In use, the robots experience different circumstances of wear and tear, such as sensor ageing, battery draining, mechanic deterioration, or even damage. In summary, hardware robots which are manufactured to the same design are not identical in many aspects. The hardware variations are unavoidable.

Of all these differences, variations in the sensors and actuators stand at the centre of the process in terms of influencing robot behaviours (Fig 3.2). Variations in robotic sensors can cause a robot to perceive different information which is then sent to the controller. Depending on the design, the controller of a robot can be linear or non-linear. A linear controller, which usualy consists of an amplifier, amplifies the difference of the sensory information. If the controller is non-linear, such as a controller with learning ability, the control strategy may vary as the learning process progresses, the difference in the sensory information can be further amplified and reflects on the actuation command. As a consequence of the actuation, the sensory input changes lead to another cycle through the interaction loops between robot and environment.

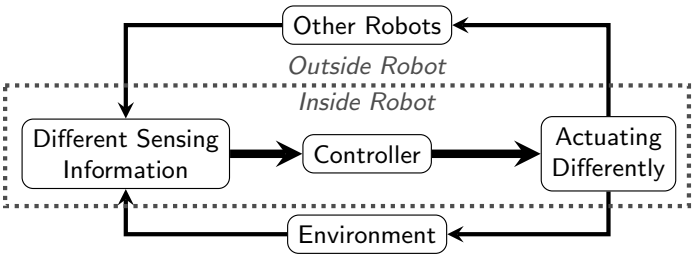


Figure 3.2: Sensors and actuators variations influence robotic behaviour: The thin dash-dotted line distinguishes between actions which happen inside and outside the robot. Variations in the robotic sensors can cause the robot to perceive different informations. Based on the varied sensory information, the robotic controller outputs different actuation commands to the actuators. Since the robot's actuators are also different, takes the robot to different environments and may even influence other robots in the environment. Then again, different sensory information is perceived from the environment. Thus variations on the sensors and actuators may influence the behaviour of a robot.

Sensors are the only source of gathering information, based on which the controller acts. Actuators are the components in a robotic system which act according to the output of robot's controller. Comparing with other components, differences generated by actuators are typically larger. Thus, variations of both the robot's sensors and actuators may directly influence its behaviours. In order to simplify the problem, it is assumed in this research that other parts of the hardware of the robots are identical and hardware variations only refer to those which can be found in their sensors and actuators.

Although it is difficult to find identical robots in terms of hardware in a swarm, identical behaviour of hardware robots in a swarm can still be achieved with costly means. This was reported by Elliott and Shadbolt (2003), where particular software needs to be fine-tuned to compensate the inherent hardware differences. This approach is very difficult and not cost-effective. This is because compensating and accurate calibration needs extra equipments and measuring process. It is very difficult to get accurate values for each parameter within a complex robotic system. In addition, such measuring process has to be taken on each of the robots within the swarm. Furthermore, due to the constant wear and tear, such process have to be repeated at regular intervals. Therefore compensating them with software is not a viable solution and the issue of hardware variations should not be ignored.

3.2 The Model of the Swarm Robot

The model of a typical swarm robot with minimal mechanical, electronic and computing elements used for this research is discussed in this section. It is assumed that the robot can follow a highly reflective track, which requires the robot being fitted with two IR photoelectric sensors. The basic features of the robot being simulated are shown in Fig 3.3, here the robot is based on a conventional, differentially steered, two wheeled robot fitted with a caster in the front and rear.

This robotic system is not modelled in every detail, only the essential parts of the robot are considered. For instance, the physical dynamics of the system is not modelled, such as the rolling friction of the wheels. Although modelling such details only involves a further set of parameters, detailed modelling would add one more layer of complexity to the problem, which makes it difficult to draw the conclusion of the relationship between hardware variations and robotic behaviours.

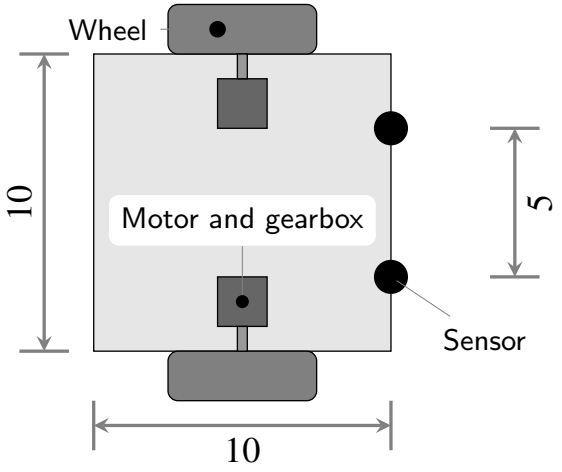


Figure 3.3: Plan view of the robot, showing the two motors used for the differential steering and the two IR sensors which are fitted in the front of the robot and point to the ground. Dimensions are in arbitrary unit. The dimensions shown are fixed and identical in all robots, but subject to manufacture variance.

3.2.1 IR Sensors

The two downward-pointing IR sensors are located at the front of the robot. To model the sensor's response, the reflective line is considered to be multiple consecutive points, which can reflect light (Fig 3.4). The magnitude of the IR sensor output can therefore be obtained by summing the response of individual reflective point within sensor's viewing range (Benet et al., 2002).

The sensor's response to an individual reflective point is modelled using Equ. 3.1,

$$S(x,\theta) = \frac{\alpha}{x^2}\cos(\theta) \qquad \text{Within } \measuredangle v \tag{3.1}$$

where

$S(x)$ is the IR sensor's response to an individual reflective point.

θ is the incidence angle of the reflective light (Fig 3.5(b)).

x is the distance between the sensor and the reflecting point on the ground (Fig. 3.5(b)).

$\measuredangle v$ is the IR sensor's viewing angle (Fig 3.5(b)).

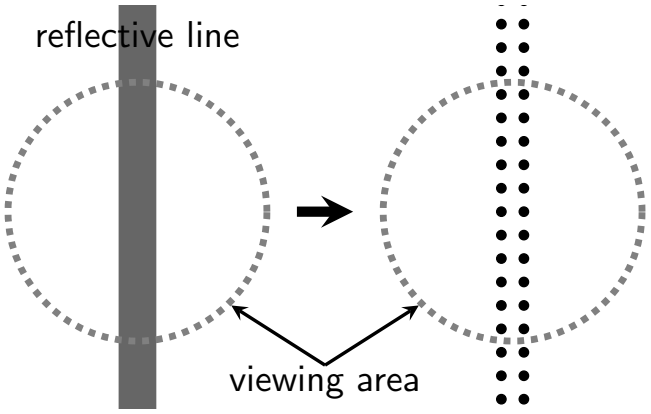
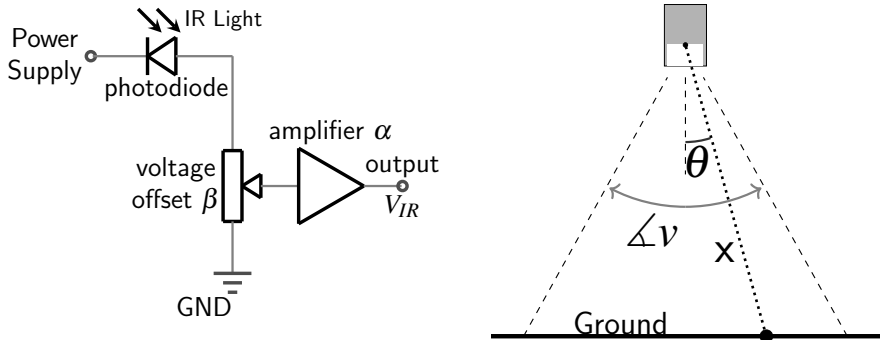


Figure 3.4: Top View of the Reflective line from the IR sensor: The dashed circle is the viewing area on the ground projected by the sensor's viewing angle. The reflective line (the wide grey line on the left) is considered as multiple consecutive reflective points (black dots on the right). The output voltage of the IR sensor is considered as the sum of sensor's response to individual dot within its viewing area.

α is the gain of the amplifier and determines the sensitivity of the sensor (Fig 3.5(a)).



(a) Parameters for IR Sensor's Electrical Characteristics

(b) This is a side view of the sensor. x is the distance from the reflective point on the ground to the sensor.

Figure 3.5: Modelling of IR Sensor

The output voltage of the IR sensor V_{IR} can be considered as Eq. 3.2.

$$V_{IR} = \sum_{1}^{n} S(x_n, \theta_n) + \beta \qquad \text{Within } \measuredangle v \tag{3.2}$$

where

n is the number of dots within IR sensor's viewing angle (Fig 3.4)

β models the sensor's output offset and the effect of ambient light (Fig 3.5).

The IR sensors are installed in the front of the robot with a certain height above the ground. The IR sensor arrangement (the direction which the sensor points to) is also modelled, showing

Fig 3.6. The parameters lateral offset angle O_l and saggital offset angle O_s are illustrated in Fig 3.7 and Fig 3.8.

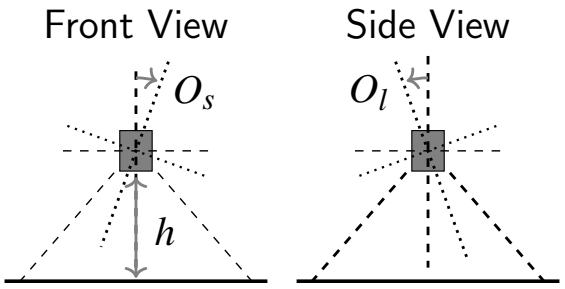
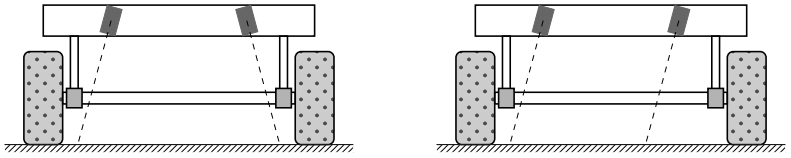
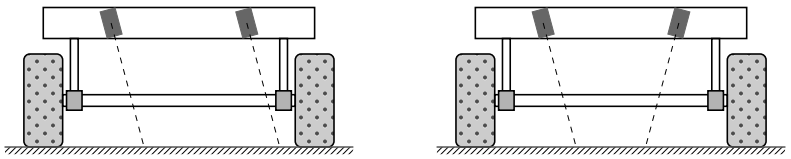


Figure 3.6: Sensor Arrangement. h represents the vertical distance from the IR to the ground. The direction which the IR sensor points to is modelling with O_l and O_s. To be specific, the sensors can be individually angled toward either the front or rear of the robot (parameter O_s), and either left or right of the robot (parameter O_l).



(a) As both values of lateral offset for the two sensor are positive, both sensors point away from the centre of the robot.

(b) As the value of the lateral offset for the right sensor is positive, right sensor points away from the centre of the robot; as the value of the lateral offset for the left sensor is negative, the left sensor's orientation is offset to the centre of the robot.

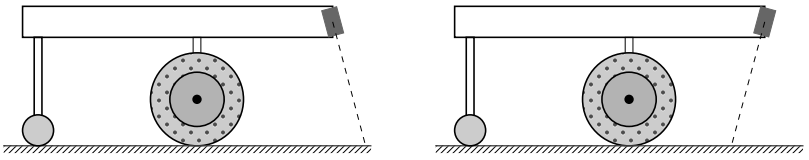


(c) As the value of the lateral offset for the right sensor is negative, the orientation of the right sensor is offset to the centre of the robot; As the value of the lateral offset for the left sensor is positive, the left sensor orientates away from the centre of the robot.

(d) As both values of the left and right sensors are negative, the orientations of both sensors are offset to the centre of the robot.

Figure 3.7: This is the front view of the robot. There are two IR sensors on the robot. Each sensor have a lateral offset angle parameter. The parameter lateral offset angle O_l determines the direction of the IR sensor to be near to or away from the centre of the robot only on the left and right basic. Depending on the values of the parameter, the arrangement of the IR sensors can have different combinations.

The parameters of IR sensor which are used to model both component and assembly variation are summarized in Tab 3.1.



(a) As both values of saggital offset for the two sensor are positive, both sensors point forward away from the centre of the robot.

(b) As both values of saggital offset for the two sensor are negative, both sensors point backwards to the centre of the robot.

Figure 3.8: This is the right view of the robot. The parameter saggital offset angle O_s determines the directions which the IR sensor points to, either the front or the rear of the robot.

Table 3.1: Hardware Variation Types for IR Sensor Parameters

Component		Assembly	
sensitivity	α	h	height
viewing angle	$\measuredangle v$	O_s	saggital offset
		O_l	lateral offset

For component variation, α will be varied to model the variation of sensor sensitivity; $\measuredangle v$ will also be varied to model variation of the viewing angle. During assembly, the sensor alignment can be different. For instance, IR sensor can be slightly placed either pointing to the left or right, to the front or back of the direction to which the sensor should point to. In addition, the position of IR sensor can be slightly higher or lower, resulting different sensor heights.

Hardware variation can also be found on the parameter sensor voltage offset β. However this type of variation can be eliminated if the output voltage of the IR sensor is subtracted with the output voltage when the IR sensor is completely blocked. Normally this process has to be done when using with IR sensors as only the change of the voltage reflects features of the environment. In addition, sensor voltage offset in rare circumstances will deteriorate unlike wheel radius or motor gain. Therefore sensor voltage offset will not be considered in the rest of this research.

3.2.2 Controller

In a line-following scenario, the robot's controller will try to keep the output of left and right IR sensors identical, if not the robot will change its relative position to the reflective line. The controller used for the robot is a PI controller and the control system of the robot is shown in Fig 3.9, where the difference of the IR sensor output (Equ 3.3) is fed to a PI (proportional, integral) amplifier (Equ 3.4), obtaining V_{PI}.

$$\delta(t) = V_{IR\_L}(t) - V_{IR\_R}(t) \tag{3.3}$$

$$V_{PI}(t) = K_p\delta(t) + K_i \int_{-\infty}^{t} \delta(\tau)d\tau \tag{3.4}$$

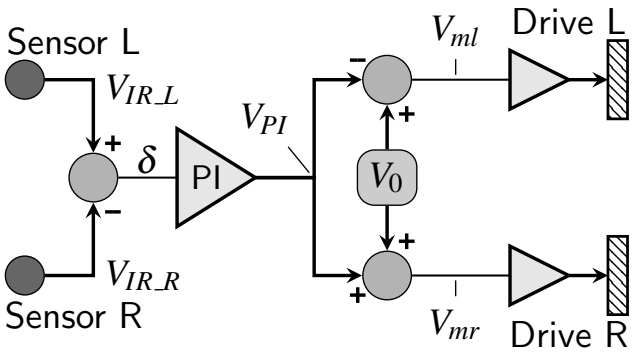


Figure 3.9: Controller of the Robot

where

$V_{IR\_L}(t)$ is the left IR sensor output voltage at time t,
$V_{IR\_R}(t)$ is the right IR sensor output voltage at time t,
$\delta(t)$ is the voltage difference between left and right IR sensor at time t,
K_p is the proportional coefficient of the PI controller,
K_i is the integral coefficient of the PI controller,
$V_{PI}(t)$ is the output of the PI controller at time t.

Generally the integral term accumulates all errors in the past and gives the accumulated offset that should have been corrected previously. In practice, PI controller used in robotic systems normally accumlates errors of a limited number of control steps. In this research, the errors of previous 300 control steps are integrated.

The controller output $V_{PI}(t)$ is then used to differentiate the rotation of the two motor drives. A constant voltage V_0 is added to both motor drives to keep the robot moving forward at all times. Otherwise, if the output voltages of the two IR sensors remain the same for a period of time, $V_{PI}(t)$ would be zero and the supply voltage to the motors is zero, thus robot will not move.

$$\begin{aligned} V_{ml}(t) &= V_0 - V_{PI}(t) \\ V_{mr}(t) &= V_0 + V_{PI}(t) \end{aligned} \tag{3.5}$$

where

$V_{ml}(t)$ is the supply voltage to the left drive train,
$V_{mr}(t)$ is the supply voltage to the right drive train.

3.2.3 Motor Drives

In this model of the robot, a conventional, differential-steering approach was used, where the two drive wheels are powered by brushed D.C. (direct current) motors and gearbox. The model of a D.C. motor can be expressed with Equ 3.6.

$$V_m = k_A \, \phi \, \omega + I_a \, R_a + L_a \frac{dI_a}{dt} \tag{3.6}$$

where

V_m is the voltage supplied to the motor terminal,
k_A is the geometry constant of the motor,
ϕ is the flux per pole,
ω is the speed of rotation,
I_a is the current through the armature of the motor,
R_a is the resistance of the armature,
L_a is the inductance of the armature.

In this research a number of assumptions are made (*i*) the motor drive is considered to be linear, and (*ii*) the torque requirements are effectively constant, hence to a first approximation the motor can be modelled as a pure gain. The gearbox between D.C. motor and the wheel is also considered as a pure gain. Effectively the motor drive of the robot refers to any modules after the output of the controller and before the wheel and it is modelled with Equ 3.7. The parameters m_R and m_L is used to describe the gain of the motor and any gearbox connected.

$$\begin{aligned} \omega_R(t) &= V_{mr}(t)\, m_R \\ \omega_L(t) &= V_{ml}(t)\, m_L \end{aligned} \tag{3.7}$$

where

$\omega_R(t)$ is the speed of rotation for the right wheel,
$\omega_L(t)$ is the speed of rotation for the left wheel,
m_R is the gain of the right motor drive,
m_L is the gain of the left motor drive.

Based on the speed of rotation for individual wheel, the linear velocity of the robot can determined using the radius of the individual wheels r_R and r_L, hence the robot linear ($\dot{x}$ and $\dot{y}$) and turning $\dot{\phi}$ speeds can be calculated:

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\phi} \end{bmatrix} = \begin{bmatrix} -\frac{r_L sin\dot{\phi}}{2} & -\frac{r_R sin\dot{\phi}}{2} \\ \frac{r_L cos\dot{\phi}}{2} & \frac{r_R cos\dot{\phi}}{2} \\ -\frac{r_L}{b} & \frac{r_R}{b} \end{bmatrix} \begin{bmatrix} \omega_L(t) \\ \omega_R(t) \end{bmatrix} \tag{3.8}$$

where

b is the distance between the two wheels, knows as wheel separation,
$\dot{x}$ is the change of x coordinate for the robot,
$\dot{y}$ is the change of y coordinate for the robot,
$\dot{\phi}$ is the change of orientation for the robot,
r_L is the radius of the left wheel,
r_R is the radius of the right wheel,

Typical variations found on a motor drive is the gain (m_R and m_L). In addition, variation can also be found on the wheel radius (r_L and r_R) and the wheel separation b. For instance, rubber tyres

often used in mobile robots are difficult to manufacture to exactly the same diameters. Asymmetric load distribution can also lead to different diameters due to different compression(Borenstein, 1996).

3.3 Hardware Variations Design

A number of parameters in the model are used to describe the hardware characteristics of the robot. The hardware variation of the robot is modelled by altering the values of these parameters.

3.3.1 Parameters for Hardware Variation

The objective of this research is to investigate how individual robot's behaviours are influenced by hardware variations originating from either components or assembly. The prototype of a conventional line-following robot is created with a number of parameters, which can be altered accordingly to model such variations.

- *Component Variations:* IR sensitivity α, IR viewing angle $\measuredangle v$, drive train gain m_R m_L, the wheel radius r_R r_L.
- *Assembly Variations:* IR height h, IR lateral offset O_l, IR saggital offset O_s, wheel seperation b.

The robot is equipped with two IR sensors, two motor drives and two wheels. Hardware difference can be found on all of the components. All the parameters on the robot which are used to model hardware variations are showing in Fig 3.10 and summarized in Table 3.2.

Table 3.2: The Parameters to Model Hardware Variations: Nine parameters are used to model the hardware variations of the robot's sensors (six variables) and actuators (three variables). In the *Type* column, "A" means variations emerge during assembly and "C" means components variations (discussed in Fig 3.1). In the *Existence* column, "LR" means this parameter exists both on the specific right and left components on the robot, and "1" means the number of this parameter which the robot has is only one. All the parameters to be varied are illustrated in Figure 3.10.

Components	Symbol	Description	Type	Existence
IR Sensor	α	sensitivity	C	LR
	$\measuredangle v$	field of view	C	LR
	h	height	A	LR
	O_l	lateral offset	A	LR
	O_s	sagittal offset	A	LR
Actuator	m	motor drive gain	C	LR
	r	wheel radius	C	LR
	b	wheel separation	A	1

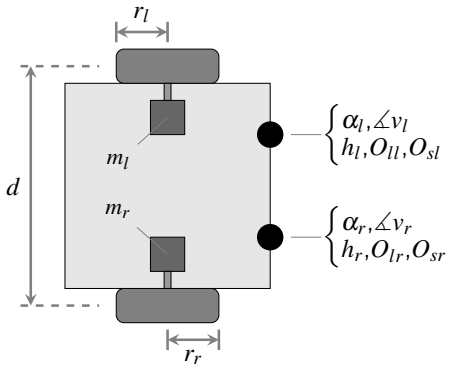


Figure 3.10: All Parameters Used to Model Hardware Variation

3.3.2 Generating Robots with Hardware Variation

To model a group of robots with hardware variations, a control robot is firstly designed with values of all its parameters being set. These values are obtained from a realistic line-following robot with similar hardware settings. Other robots within the group are then derived from this control robot with its parameter values being varied model hardware variation among the group. This process is illustrated in Table 3.3.

Table 3.3: Generating robots with hardware variations: Firstly, the control robot R_0 is generated and the values of its parameters are obtained from a realistic robots with similar hardware settings. To model hardware variations in the swarm, other robots R_1, R_2, ... R_n are then generated with parameters which are varied from those of the control robots R_0 by multiplying with $1 + r_{n,m}$. $r_{n,m}$ is a randomly generated number whose value is between $-0.2 \leqslant r_{n,m} \leqslant +0.2$. For the sensor orientation offset parameters (sensor lateral and saggital offset), a maximum angle of 3° is used.

Device	Parameter	R_0	R_1	R_2	...	R_n
left IR	sensitivity	α_l	$\alpha_l(1+r_{1,1})$	$\alpha_l(1+r_{2,1})$	...	$\alpha_l(1+r_{n,1})$
	field of view	$\measuredangle v_l$	$\measuredangle v_l(1+r_{1,3})$	$\measuredangle v_l(1+r_{2,3})$	...	$\measuredangle v_l(1+r_{n,3})$
	height	h_l	$h_l(1+r_{1,4})$	$h_l(1+r_{2,4})$	...	$h_l(1+r_{n,4})$
	lateral offset	$o_{ll}=0$	$3^\circ \times r_{1,5}$	$3^\circ \times r_{2,5}$	...	$3^\circ \times r_{n,5}$
	sagittal offset	$o_{sl}=0$	$3^\circ \times r_{1,6}$	$3^\circ \times r_{2,6}$	...	$3^\circ \times r_{n,6}$
right IR	sensitivity	α_r	$\alpha_r(1+r_{1,7})$	$\alpha_r(1+r_{2,7})$	...	$\alpha_r(1+r_{n,7})$
	field of view	$\measuredangle v_r$	$\measuredangle v_r(1+r_{1,9})$	$\measuredangle v_r(1+r_{2,9})$	...	$\measuredangle v_r(1+r_{n,9})$
	height	h_r	$h_r(1+r_{1,10})$	$h_r(1+r_{2,10})$	...	$h_r(1+r_{n,10})$
	lateral offset	$o_{lr}=0$	$3^\circ \times r_{1,11}$	$3^\circ \times r_{2,11}$	...	$3^\circ \times r_{n,11}$
	saggital offset	$o_{sr}=0$	$3^\circ \times r_{1,12}$	$3^\circ \times r_{2,12}$	...	$3^\circ \times r_{n,12}$
left actuator	motor gain	m_l	$m_l(1+r_{1,13})$	$m_l(1+r_{2,13})$	...	$m_l(1+r_{n,13})$
	wheel radius	r_l	$r_l(1+r_{1,14})$	$r_l(1+r_{2,14})$	...	$r_l(1+r_{n,14})$
right actuator	motor gain	m_r	$m_r(1+r_{1,15})$	$m_r(1+r_{2,15})$	...	$m_r(1+r_{n,15})$
	wheel radius	r_r	$r_r(1+r_{1,16})$	$r_r(1+r_{2,16})$	...	$r_r(1+r_{n,16})$
	wheel distance	d	$d(1+r_{1,17})$	$d(1+r_{2,17})$	...	$d(1+r_{n,17})$

For instance, the height of left sensor of the control robot R_0 is 5. When deriving R_1, the generated random number for left sensor height $r_{1,4}$ is -0.01, the height of its left sensor is calculated as

$$\begin{aligned} h_{l,R0} &= 5 \\ r_{1,4} &= -0.01 \\ h_{l,R1} &= h_{l,R0} \times (1 + r_{1,4}) \\ &= 5 \times (1 - 0.01) \\ &= 4.75 \end{aligned}$$

Therefore, the height of left sensor of the varied robot R_1 is 4.75.

In practice, the hardware variations of each parameter typically follow a Gaussian distribution within a large number of similar robots (Lyon, 2013). As the rest of robots in a group is generated based on the control robot, therefore $\{r_{1,1}, r_{2,1}, ..., r_{n,1}\}$, $\{r_{1,2}, r_{2,2}, ..., r_{n,2}\}$, $\{r_{1,3}, r_{2,3}, ..., r_{n,3}\}$, ..., $\{r_{1,17}, r_{2,17}, ..., r_{n,17}\}$ should follow Gaussian distribution and the mean of the distribution is considered as 0 for simplified the model of hardware variation. Effectively the modelling of the hardware variation for the group of robots is to use series of Gaussian-distributed random numbers to vary the parameters of the control robot.

Each type of components have their own working principle and their own manufacture process, the values of their parameters generally follows different Gaussian distributions. To be specific, different groups of $r_{n,m}$ have different variances. For instance, variance of the sensor parameter values are generally small; variance of motor parameter values are generally large, illustrated in Table 3.4.

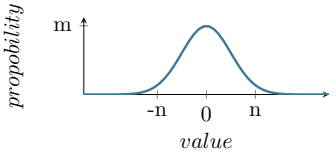
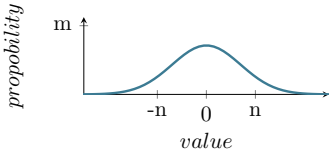
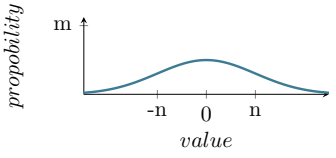
3.4 Experimental Design

To test the behaviour of the hardware-varied robot, the typical line-following task is used. The testing process involves two stages: the selection of the controller parameters and testing of the robot with the line-following tasks.

3.4.1 Controller Parameter Selection

The parameter selecting process is to select the PI controller parameters proportional coefficient and integral coefficient, with which robot can follow a trajectory with minimal difference comparing with the target reflective line, known as parameter selecting line. The black line illustrated in Fig 3.11 is the reflective target line to be followed and consists of a single period of sinusoid with two straight elements at both beginning and end.

Table 3.4: Different parameters are varied by adding Gaussian-distributed random numbers with different variances. Due to strict quality control, sensor embedded parameters including sensitivity, view angle are usually within a small range. Therefore when simulating hardware variation with these three parameters, Guassian-distributed random numbers with small variance are used. When sensors are being installed on the robot, their orientation and position (related with height, lateral and saggital offset parameters) can be different from the ones installed on other robots. In this case random numbers with medium variance are used. Parameters with the robotic actuators (motors/wheels) usually have large variances.

	Parameters	Variance	Gaussian Distribution
sensor	sensitivity field of view	small	
sensor	height lateral offset saggital offset	medium	
actuator	motor drive gain wheel radius wheel distance	large	

This particular pattern of the target line is chosen for several reasons: firstly, the sinusoid contains both right and left curves so that robot's ability of steering to both directions can be tuned; secondly, the straight lines at the beginning and end help to guide the robot correctly to the sinusoid part. Otherwise, robot with large magnitude of hardware difference from certain components might not be able to reach the most important part of parameter selecting route, the sinusoid part specifically.

Another type of parameter selecting arena (Appendix Figure B.1) was tried during the experiments, results did not show much difference. Therefore the simple parameter selecting arena (Figure 3.11) was used throughout the rest of the research.

During parameter selecting process, robots start from the starting point on the left of the arena with the orientation to to the right. The robot is required to reach the end point on the right of the arena by following the reflective target. All robots, even if they have different hardware variation settings, will start the parameter selecting process with exactly same initial condition including the position and orientation at the starting point to satisfy the requirement of controlled experiments. The target of the parameter selecting process is to find a set of parameters which helps robot to follow the line as accurately as possible.

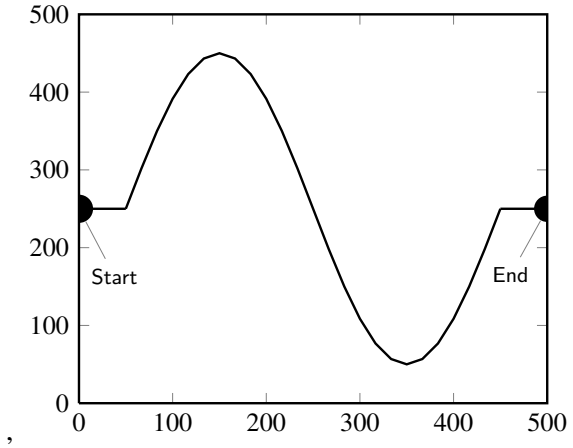


Figure 3.11: The controller parameters selecting arena's dimension is 500 × 500 in arbitrary units, comparing with the size of the robot in Figure 3.3. The line to be followed is a sinusoid with two straight elements. The robot's start and end points are shown.

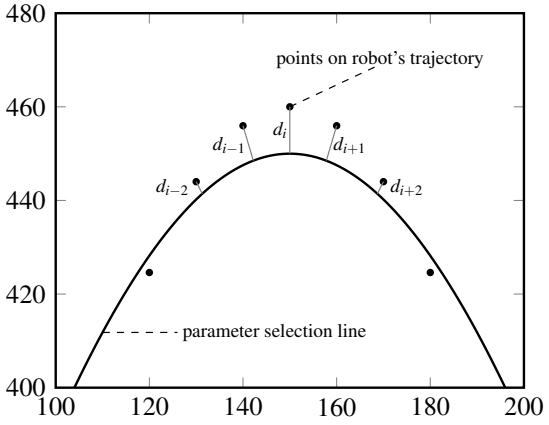


Figure 3.12: Analysing Robot's Trajectory for Controller Parameters Selection: The curved line is the target line to be followed. The dots represent locations of the robot at each simulation step which form into the trajectory of the robot. Trajectory of the robot is analysed for each step by calculating the minimum distance d_i between the robot and the target line.

To measure the accuracy of the robot's following the target line, a variable of average position error (E_{avg}) is defined and calculated as the following. The positional error (e_i) at each simulation step (i) is defined to be the minimum distance (d_i) from robot's current location to the target line (Figure 3.12). Upon completion of the line following, the positional error at every simulation step are accumulated and averaged by the total number of steps (n) for the robot to finish the whole target line, thus the average positional error (E_{avg}) is obtained (Equ 3.9).

$$\begin{aligned} e_i &= d_i \\ E_{avg} &= \frac{1}{n}\sum_{0}^{i=n} e_i \end{aligned} \tag{3.9}$$

Effectively the average position error (E_{avg}) is the average distance between the location of the robot and the target line over all steps taken by the robot when following the target line.

In order to obtain a satisfactory set of PI controller parameters, the simulation used in this research searches exhaustedly in the parameter space. The parameter space is two dimensional space as there are only two parameters to be selected: the proportional coefficient K_p and the integral coefficient K_i. The searchable region for is $0 \leq K_p \leq 150$ and $0 \leq K_i \leq 100$, equivalently an area of 150×100. At the beginning of this research, a much larger search space was defined, however in practice, it was found that all of the selected controller parameters falls within this smaller region.

A Sobol sequence is used to select the initial values for the controller parameters within the parameter search space. Sobol sequence, firstly introduced by Sobol (1976), is an example of quasi-random low-discrepancy number generator. In other words, numbers from Sobol sequence not only have uniform distribution over the search space, but also cover the search space more evenly comparing with other randomly generated numbers (Sobol and Levitan, 1976). As for the case of selecting controller parameters of the robot, the set of parameter can be found with relatively fewer number of trials. In this research, the number of trials to select the controller parameter was set to be 10^5 for each robot, which is adequate for the parameter space of 150×100.

The Matlab code for generating required sobol number is showing in the following.

```
p=sobolset(2,'Skip',1e3,'Leap',1e5)
```

What the code does is generating a 2-D Sobol sequence, skip the first 1000 values, and then retain every 101st point. The numbers in the sequence are all within $(0,1)$. Therefore in the nth trial, the controller parameter can be obtained as:

$$\begin{aligned} k_p &= \texttt{p(1,n)} \times 150 \\ k_i &= \texttt{p(2,n)} \times 100 \end{aligned} \tag{3.10}$$

During each trial, a set of controller parameters which is selected by the Sobol number from the controller parameter space is assigned to the parameters of the robot's controller. The robot is then tested in the parameter selecting arena (Fig 3.11) to follow the target line. Upon completing the simulation, trajectory of the robot is analysed according to Equ 3.9, thus the average position error (E_{avg}) is obtained. With large number of trials, the set of parameters with which the robot achieves the smallest average position error is selected as the controller parameters for the robot.

It is admitted that the set of parameters selected may not be the optimal parameter for the robot, however finding the optimal set will cost an unreasonable amount of time and effort. In fact, a robot is normally used after adequate training in reality. In addition, an experiment was conducted in which the controller parameter were searched excessively (The number of trials was 10^7). It was found that the set of parameters obtained in the excessive parameter search did not reduce the average positional error (E_{avg}) by a significant amount.

3.4.2 Testing

After parameter selecting process, robots were then tested in the testing arenas, from which trajectories of the tested robots will be analysed to evaluate their behavioural characteristics. Various types of testing arenas will be used in this research. One example of the testing arenas are illustrated in Fig 3.13.

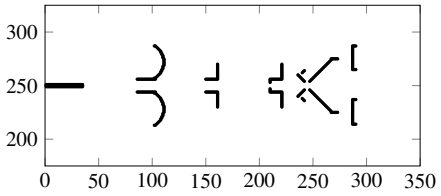


Figure 3.13: Arena used to test the behaviours of robots. The black lines are the reflective lines that robots have to follow. The arena is symmetrical around $y = 250$. The robots start at $(0, 250)$ with the orientation to the right. After sufficient period of time for the simulation, trajectories of the robots are analysed.

The aim of this research is to investigate how hardware variation influences the behaviour of swarm robots. Therefore it is important to find out the suitable comparison metrics which can be used to evaluate characteristics of the robotic behaviours.

If multiple robots with minor difference in their hardware parameters are tested in the same line-following arena with exactly the same initial condition, the difference in their trajectories is only cause by the difference in their hardware parameters. Thus one can discover the relationship between hardware parameters and the trajectory generated, and possibly which type of hardware leads to which trajectory. In other word, the behaviour characteristics of a robot caused by the hardware variation can be identified.

Therefore in this research, trajectories generated by the robots will be metrics to evaluate characteristics of the robotic behaviours.

3.5 Approach to Simulations

This research is conducted through simulation. The investigation of hardware variation in swarm robots requires a massive number of computation-extensive simulations. The challenges encountered when conducting the simulations are described as well as the the techniques used to tackle them.

3.5.1 Simulations Challenges

As discussed in the previous chapter, a number of robots which are all derived from the control robot, will be initiated at the beginning of the simulation with variations added to their

parameters to model hardware variations. These robots need to be tuned for optimal controller parameter individually in the parameter selecting arena. Then the robots are tested in the testing arena to get the behaviours. During this process, a number of difficulties are encountered:

- The number of parameter settings of the robots are very large. Firstly, there are 15 parameters in the model and difference(s) can be added to the individual parameter or multiple parameters at the same time. Secondly, the difference is an quantified percentage value and different parameters of a particular robot can have different percentages. How to manage the parameter setting for each simulation and its result remain a challenge.
- Computational requirement for the simulation is very high due to the method of exhausted search for selecting the controller parameters. It takes some time for one robot to following the parameter selection line and additional time for evaluating the trajectory. The line following and trajectory evaluation have to be repeat for 10^5 times with different controllers parameters from the parameter search space. This process has to be conducted for every robot in the group.
- Scientific evaluation of the simulation results is very difficult because the number of simulations is large and each simulation have different parameter settings.

In order to solve the challenges, a number of techniques have been applied including: version control system Git, use of Iridis supercomputer for parallel computing, and automatic simulation report generation.

3.5.2 Simulation File Management and Git

As a large number of simulations will be undertaken, it is necessary to keep a recording of everything for reference. To do this, a unique name is give to each simulation, and the name starts to date as one of the specifiers. A snapshot of experiments are shown in the following.

```
dir:data ..................................... Directory contains all simulations
├── ......
├── dir:20140527-Chromato4-LongMemory-Arena1
├── dir:......
├── dir:20141025_ml450-MemABS_Xcos-orient_testRt0.7_R
├── dir:20141026_ml450-MemABS_Xcos-bDoub-orient_testRt0.7
└── ......
```

In each simulation, there are several basic elements: configuration of the experiment, arena used, robots participated in the simulation, simulation results, post-simulation data processing scripts and most importantly version file stores the version number of the codes for experiment configuration, generating arenas, robots and running simulations. Particularly, 'configuration'

includes robot's hardware parameter setting, basic simulation parameters like initial condition, simulation time etc. which are the required parameters to start the simulation. A snapshot of the directory containing one simulation is shown.

```
dir:20140527-Chromato4-LongMemory-Arena1 ..... experiment name
 ├─ file:config ..... configuration file
 ├─ dir:arena
 ├─ dir:robot
 ├─ dir:results ..... simulation results
 ├─ dir:script ..... for post-simulation data processing
 └─ file:VERSION ..... version number for codes, configuration, arena
```

A common way to keep the track of files would be saving a hardware copy of the files in the directory for each simulation. However as the number of simulation is large, the stored hard copy may be modified by mistake, it is safe to use version control software such as *Git* which is widely used in industries to maintain project files. To be specific, git repositories are created both for 'configuration' and 'codes'. Before simulation starts, modifications which have been made to each files will be committed, which are then saved to each repository. By committing the changes, an unique version number will be obtained. For instance,

```
config version: 10.0-4-g1b288
codes version: 4.0-29-g79e5b
```

This version number is used as the index to track the 'configuration' or 'codes'. During simulation, both the version numbers for currently 'configuration' and 'data' will be stored automatically in the VERSION file for every simulation .

Git version control system does not record the related files directly but keep tracking of what changes have been made to the files, which circumvents unnecessary disk occupancy and more importantly this is particularly useful for the simulation 'codes' since a clearly path of modifications can be perceived and this prevents mistakes from happening during data analysis phase. With help of the aforesaid version number, 'configuration' and 'codes' can be easily traced and reverted to that version easily for conducting further simulation.

3.5.3 Parallel Computing and Iridis

The computational requirement for the simulation is large. In each simulation, there are several robots and each robot has to be tested in the parameter selecting arena extensively. Specifically, during parameter selecting process, it takes 0.36 second to test and evaluate one set of controller parameters for one robot. The selection of the controller parameters requires at least 10^5 samples in the 2-dimensional parameter space. To select the controller parameters for one robot, it

requires

$$0.36 \times 100000 \div 60 \div 60 \approx 10 \text{ hours}$$

In one experiment, a swarm consists at lease 32 individuals. In total, the tuning of the swarm needs

$$32 \times 10 = 320 \text{ hours}$$

For a swarm consisting of 32 individual robots, it requires at least 320 hours just for choose the parameters of the robots in the swarm. In addition, a swarm of 32 robots is a relatively small group. In this work, the swarm have more than 200 robots. Therefore if the simulation is conducted on a conventional PC platform, it will take weeks to finish one experiment.

To address this problem, the parallel computing approach was used. It utilizes multiple processors to run the simulation simultaneously to decrease the simulation time. The simulation is separated in a way that the computation of each robot is separated as individual process. One process is allocated to one processor core. With multiple processor cores, several robots can be simulated simultaneously.

For this research the supercomputer Iridis at University of Southampton was used. According to Wolton (2012), a maximum number of 384 processor cores can be utilized at the same time. In other words, a maximum number of 384 robots can be prepared within 10-hour time.

With the parallel computing technique, the time spent on each simulation is dramatically reduced compared with the conventional technique with a PC platform.

A script is included in Appendix C which illustrates how simulation jobs are submitted to the Iridis 3 supercomputer.

3.5.4 Automatic Report Generation

As the number of simulations is large, it is convenient to have a summary of the simulation results for data analyse. To do this, an automatic report generation script is design, which can generate a pdf file containing crucial information from simulation such as 'configuration' (including its version number), 'codes' version number, parameter selecting results, testing graph etc.

Firstly, a latex template for simulation report is created, in which some blanks is to be filled with informations from the simulation. A Bash script is used to fetch required data from each simulation folder to replace the specified keywords. After this, the latex file is compiled and pdf file is generated. An example of the automatic generated report can be seen in Appendix D.

3.6 Summary

In this chapter, the hardware variation problems in swarm robotics, especially the sources of hardware variation were firstly identified. It was found that hardware variations could emerge from the component manufacture process, robot assembly process and the use of the robots. Of all types of hardware variations on the robotic components, sensors and actuators can influence the behaviours most as it controls the input and output of the robotic system. It was hypothesized that hardware variations, which are though small, might still influence robotic behaviours since the amplification impact of the software controller and the environment.

In order to prove the hypothesis, the model of a typical line-following robot and the methods of generating hardware variations for the robotic sensors, actuators and mechanical structure were presented. The approach of using Gaussian-distributed random numbers to model hardware variations in a swarm consisting a large number of robots were also described, as well as the method of selecting parameters for the robot PI controller.

Following this, an example of the testing arena used to analyse robot's behaviour was presented. It was argued that difference in the hardware is the only cause for difference in the trajectories if the robots are tested in the same environment with exactly the same initial condition. Therefore robot's trajectories can be used to investigate the relationship between hardware variation and robot's behaviour caused by hardware variations.

In the end, the techniques used in this work to solve the challenges encountered during the simulations was described.

In the next chapter, hardware-varied robots and the control robot will be simulated and their trajectories will be compared to see if any difference can be found in their trajectories.

Chapter 4

The Effect of Hardware Variation on Robots' Trajectories

The model of a typical swarm robot and the method of selection the controller parameters were described in the last chapter for investigating the issue of hardware variation. In this chapter the effect of hardware variation on robots' trajectories will be investigated. Specifically, a number of robot with minor hardware difference will be tested to see if different trajectories can be generated as the result of the hardware difference.

Section 4.1 explains the effects of robotic components, software controller and environment during the amplification process for hardware variations. The design of the experiment is discussed in Section 4.2. The simulation results are presented in Section 4.3 and 4.4. Finally Section 4.5 summarized the investigation of this chapter.

4.1 Methodology

As discussed in Section 3.1, hardware variation can be found in the construction of swarm robots. To investigate if hardware variation can influence robotic behaviours which, to be specific, refers to the robots' trajectories in the line-following task, a group of robots with minor difference in their hardware parameters are simulated to see if different trajectories are taken by the robots.

To ensure that the experiment is well constructed, the only difference between individual robots is the values of their hardware parameters. To be specific, the same method for selecting the controller parameter will be used for all robots in the group. In addition, the robots will be tested in a line-following arena with exactly the same initial conditions (starting position and orientation). In this case, if robots takes different trajectories during the line-following task, this is only triggered by the hardware difference of the robots.

With this control experiment, one can find out if hardware difference influence the trajectories or the behaviours of robots in the line-following task. By gradually reducing the magnitude of hardware difference, one can further identify how small the hardware difference is so that robotic trajectories will no longer be influenced.

The control robot described in Section 3.3 was used to create a group of robot with minor hardware difference by varying values the parameters. In the experiment of this chapter, rather than creating a group of robots with every parameters being varied, it is better to use the robots with one parameter is different from the control robot. And different parameters are varied in different robots. (In this case, the number of robots in the group will be equal to the number of parameters which is chosen to model hardware variation.) Thus one can find out what type of trajectory the robot takes if certain parameter is varied.

In addition, fixed magnitude of hardware variation will be applied to the robots. This implies that the hardware varied robots will be generated by applying a fixed percentage value to the parameters of the control robot.

4.2 Experimental Design

The experiment consists of three steps: preparing robots, selecting controller parameters for the robots and testing them using line-following task.

4.2.1 Preparing the Robots

As discussed in Section 4.1, for each of the robots in the group, only one parameter will be different from the control robot and the rest parameters of the robot will remain the same. All parameters used to model hardware variation are listed in Table 3.2 and Fig 3.10. As there are 15 parameters to model hardware variation, the group of robots used in this experiment consists of 16 individuals. One of them is the control robot and 15 of them are the robots whose parameters are varied from the control one. The naming scheme of the 16 robots is described in Fig 4.1. And explanations of the identifier for individual robot can be found in the last column of Table 4.1.

For the 15 robots whose hardware parameters are to be varied, the same magnitude of hardware difference will be applied. To be specific, the selected parameter value of the individual robot in the group will be decreased by a reasonable and determined value of 1%.

There are exemptions for the parameters including saggital offset O_s, lateral offset O_l. As values of these two parameters of the control robot is 0 and 1% less is still zero. Therefore a maximum value of 3° for the offset angle is given to these two parameters. 1% means the offset of the sensor alignment angle is $1\% \times 3°$. The direction of sensor alignment is referred to Fig 3.8 and 3.7. The above process is summarized in Table 4.1.

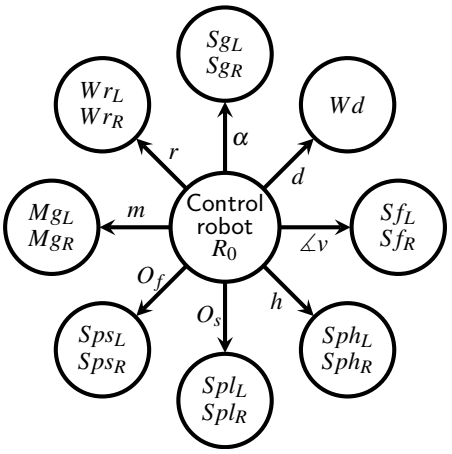


Figure 4.1: The Naming Scheme for Robots Used in this Experiment: The control robot is R_0. The rest robots in the group is generated from the control one. And only one parameter is different for each robot. The descriptions of the parameters can be found in Table 3.2. 15 robots are generated from the control one. If the parameters are related with the IR sensor, the identifier of the robot starts with *S*, *M* refers to motor drive and *W* refers to wheel.

Table 4.1: Robots and Their Hardware Differences Comparing with the Controlled One: Only the parameter identified is varied and the rest of the parameters of the individual robot remain the same with the control robot.

Robot Identifier	Hardware Difference	Explanation
R_0	N/A	control robot
Sg_L	$\alpha_{Sg_L} = \alpha_{l,R0} \times (1-0.01)$	left sensor gain
Sg_R	$\alpha_{Sg_R} = \alpha_{r,R0} \times (1-0.01)$	right sensor gain
Sf_L	$\measuredangle v_{Sf_L} = \measuredangle v_{l,R0} \times (1-0.01)$	left field of view (viewing angle)
Sf_R	$\measuredangle v_{Sf_R} = \measuredangle v_{r,R0} \times (1-0.01)$	right field of view (viewing angle)
Sph_L	$h_{Sph_L} = h_{l,R0} \times (1-0.01)$	left sensor height
Sph_R	$h_{Sph_R} = h_{r,R0} \times (1-0.01)$	right sensor height
Spl_L	$o_{Spl_L} = 3^\circ \times (-0.01)$	left sensor lateral offset
Spl_R	$o_{Spl_R} = 3^\circ \times (-0.01)$	right sensor lateral offset
Sps_L	$o_{Sps_L} = 3^\circ \times (-0.01)$	left sensor saggital offset
Sps_R	$o_{Sps_R} = 3^\circ \times (-0.01)$	right sensor saggital offset
Mg_L	$m_{Mg_L} = m_{l,R0} \times (1-0.01)$	left motor gain
Mg_R	$m_{Mg_R} = m_{r,R0} \times (1-0.01)$	right motor gain
Wr_L	$r_{Wr_L} = r_{l,R0} \times (1-0.01)$	left wheel radius
Wr_R	$r_{Wr_R} = r_{r,R0} \times (1-0.01)$	right wheel radius
Wd	$d_{Wd} = d_{R0} \times (1-0.01)$	wheel separation

4.2.2 Robotic Controller Parameter Selection

After the group of 16 robots are prepared, the controller parameters for each robot are to be selected using the method explained in Section 3.4.1. The controller parameters to be selected are listed Table 4.2.

Table 4.2: Controller Parameters to be Selected

Parameter	Explanation
K_p	proportional coefficient
K_i	integral coefficient

As described in Section 3.4.1, during controller parameter selection, each of the 16 robots in the group is required to follow the parameter selection line (Fig 3.11) with different sets of controller parameters which is picked by the Sobol sequence within the parameter space. 10^5 sets of controller parameters are tested. The set which helps the robot achieve the smallest average position error E_{avg} (Equ. 3.9) in terms of following the target line is selected as the controller parameter for the robot.

After the sets of parameter are determined for all robots in the group, the average positional errors E_{avg} of all 16 robots in the group are shown in Figure 4.2. The average positional error E_{avg} describes how accurate a robot follows the parameter selection line with the selected controller parameters. To be specific, the average position error E_{avg} is a measurement of the average distance between the location of a robot and the parameter selection line over all the steps throughout the parameter selection line. Comparing with the size of robot in Fig 3.3, the distance between robots' trajectories and the target line is small and all robots in the group followed the parameter selection line closely. While the robots have different hardware parameter values, they all took similar trajectories with the selected controller parameters in the parameter selecting arena.

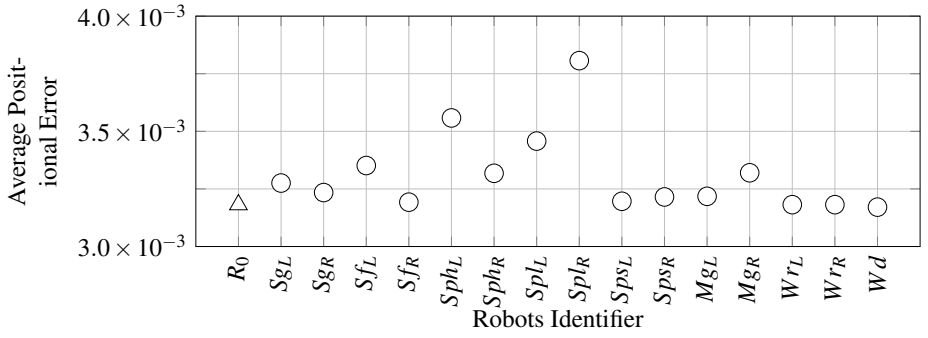


Figure 4.2: The Average Positional Error of the Robots with 1% Hardware Variations: The average position error for all robots are within a small range $\in [0.003, 0.004]$. Comparing with the size of the robot (10×10 Fig 3.3), the trajectories taken by all robots are almost the same with the parameter selection line.

As this is a rather simple parameter selecting method and the same method is applied to every robot in the group, it is not expected that robots will be tuned to a state that they all take exactly the same trajectory, without showing any average position error at all. However with this parameter selecting method, all robots were able to follow the parameter selecting line accurately and follow almost the same trajectory. In other words, the robots' hardware differences were partially compensated by the controller parameters selected.

On the other hand, although the selected controller parameters help a robot to achieve the best accuracy for following the parameter selecting line, difference between the trajectories of the robots and the parameter selecting line still exist. By evaluating such difference, one can perceive that how well the robot is tuned, how diverse the group of robots behave, which can be regarded as a preliminary review of how hardware variation influences robot's behaviours.

4.2.3 Testing

After selecting the controller parameters, all the robots were tested in the testing arena showing in Figure 4.3. In the arena, all the robots start from the location $(0, 250)$, with the same orientation to the right. The robots are tested individually and not as a swarm, so the robots were not required to interact with anything, except the lines.

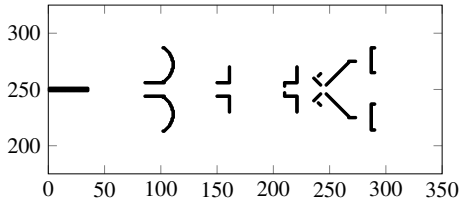


Figure 4.3: The Testing Arenas: The black lines are the lines to be followed. They are symmetrical about y=250 and these lines have gaps. Robots start at the coordinate (0,250) with the orientation to the right.

The testing arena provides a structured environment with three key features.

- The arena is symmetrical around $y = 250$. The control robot whose right and left components are identical will always follow a straight trajectory as long as the lines are symmetric, regardless of how other sections of the lines are placed. However this is not the case for robots with variations on its either left or right component, who can be easily distracted by other sections of the lines and generate different trajectories. In addition, straight trajectory from the control robot is a good reference for comparing with trajectories from other robots.

- The lines are not consecutive. Due to the design of the robot PI controller, if no reflective lines are perceived by the IR sensors, the robot will maintain its orientation and continue to move forward (due to the constant voltage applied to the controller, which was explained in Section 3.2.2). In this case, if the orientation of the robot is not strictly to the right, robot will slowly move away from the symmetric line y=250, thus generate different trajectories against the control robot.

- Several branching sections are created with the reflective lines which are placed along and near y=250. The lines can trigger different robots to move away from the symmetrical line so that one can figure out the power of influence to the trajectories which are caused by different types of hardware variation.

The environment of this type can show not only the different behaviours of the robots, but also how the varied parameter influence robotic behaviours and the influential power of different types of hardware difference.

4.3 Results and Discussion

The simulated robots used in this experiment are all variants of the control robot, with a single parameter being reduced by a fixed and small percentage (1%) for each robot, as shown in Table 4.1. Even though these difference are very small, it was shown that robots still took different trajectories in the testing arena (Figure 4.4).

The trajectories of the robots with varied hardware parameters are different comparing with the control robot R_0. R_0 is distracted neither to the left nor to the right and takes a trajectories which is exactly the symmetrical line y=250. This is because there is no difference between its left and right sensor or actuators. Other robots except Robot Wd go to either the upper or lower side of the arena.

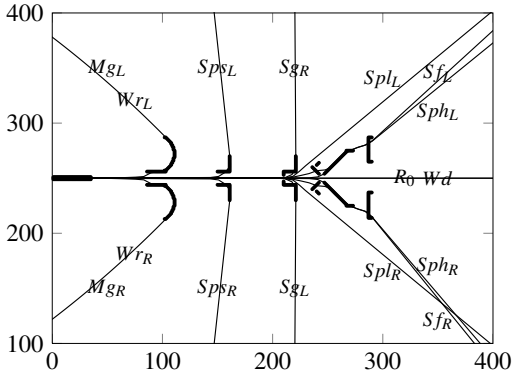


Figure 4.4: Robots' Trajectories in the Testing Arena: The thick lines are the reflective lines to be followed. The thin lines are robots' trajectories which are labelled with robots' identifiers (Table 4.1). All robots start at the coordinate (0,250) with the orientation to the right. The direction of travel for all robots is from left to right. All robots reached the boundary of the arena.

The effect of hardware difference applied on individual parameter of the robot is discussed in the following.

4.3.1 Wheel Distance

The difference between the robot Wd and the control robot R_0 is that Wd's wheel separation is 1% smaller than that of R_0.

According to Equ 3.8, the parameter wheel separation d is only related to the orientation change of the robot. To be specific, the orientation change can be expressed with Equ 4.1.

$$d\theta = \frac{1}{d}\left(\omega_R r_R - \omega_L r_L\right) dt \tag{4.1}$$

where

$d\theta$ is the change of orientation for the robot,
ω_L is the speed of rotation for the left wheel,
ω_R is the speed of rotation for the right wheel,
r_R is the radius of the right wheel,
r_L is the radius of the left wheel,
d is the separation of the two wheels.

When d decreases by 1%, $d\theta$ increases, meaning that the robot can change its orientation more quickly. In other words, the robot has a smaller turning radius. However as robot Wd does not have any difference between its left parameters of either IR sensor, motor drive or wheels and the corresponding parameters on the right, it was not distracted from the symmetrical line y=250, therefore robot Wd did not have to change it orientations and robots Wd and R_0 generate the same trajectory and behave similarly in this testing arena.

4.3.2 Motor Gain and Wheel Radius

A variable speed of the wheel *Velocity* is defined as the velocity of the point on the wheel which directly contacts with the ground. For either left and right wheel, the speed of the wheel can be expressed in Equ 4.2.

$$Velocity = \omega \times r \tag{4.2}$$

where

Velocity is the wheel speed defined,
ω is the speed of rotation for the wheel,
r is the radius of the wheel.

As discussed in Section 3.2.3, both left and right robotic motor drives are modelled as two individual pure gains which cover any module exists after output of PI controller and before the wheel. Therefore according to Equ 3.7, the wheel velocity can be expressed as Equ 4.3

$$Velocity = V_m \times m \times r \tag{4.3}$$

where

V_m is the voltage supplied to the motor terminal,
m is the gain of motor drive.

Therefore any change made to the gain of the motor drive have the same effect when the same change are made to the wheel radius. These two types of variations founded on the actuator

gain and wheel radius is considered as the same type. This explains that robots Mg_L and Wr_L generated the same trajectory, and robots Mg_R and Wr_R followed the same path.

4.3.3 Sensor Viewing Angle and Height

The robots with these two types of variation took similar path in this testing arena: Sf_L and Sph_L almost have the same trajectory until they pass x=300 where they slowly diverge; and the same trend can be found in the trajectories of robots Sf_R and Sph_R.

This finding can be explained that lowering sensor height has almost the same effect of reducing the field of view of the sensor, showing Figure 4.5. White reducing sensor height, the perception area of the sensor is also reduced. However both actions are not exactly the same, lowing the height of the sensor also reduce the distance from the reflective point on the ground to the sensor. This explains that why their trajectories are not exactly the same.

The modelling of the IR sensor mentioned in Section 3.2.1 repeated in Equ 4.4.

$$V_{IR} = \sum_{1}^{n} \frac{\alpha}{x^2} \cos(\theta) + \beta \qquad \text{Within } \measuredangle v \tag{4.4}$$

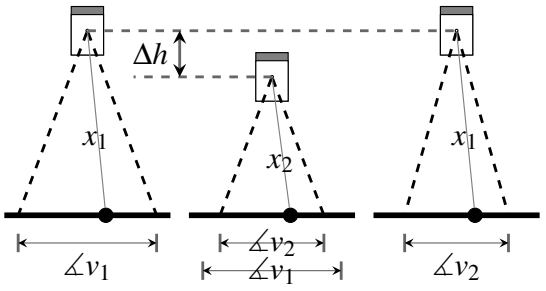


Figure 4.5: Lowering Sensor Height and Reducing Sensor Field of View: An IR sensor with the viewing angle of $\measuredangle v_1$ and normal height is shown on the left. Supposing there is an reflective dot on the ground and the distance from the dot to the sensor is x_1. If the height of the sensor is reduced (showing in the middle), its viewing angle would be reduced to $\measuredangle v_2$, the number of reflective dots within the IR sensor's viewing arena would decrease, and distance which the reflective light travels would also be reduced to x_2. According to Equ 4.4, reduced reflective light travel distance will counteract the effect of reduced viewing angle. For comparison, the third figure shows the IR sensor with normal installing height but reduced the viewing angle $\measuredangle v_2$. The distance which the reflective light travels remain the same as x_1.

.

4.3.4 Sensor Offset and Sensor Gain

Spl_L, Spl_R are the robots whose lateral offset angle for the IR sensor is different from the control robot. This parameter is described in Figure 3.7, which determines if the IR sensor points away from the centre of the Robot (either left or right).

Sps_L, Sps_R are the robots whose saggital offset angle for the IR sensor is different from the control robot. This parameter is described in Figure 3.8, which determines if the IR sensor points to either the front or the back of the robot.

Sg_L, Sg_R are the robots whose sensor gain parameter is different from the control robot. This parameter is described in Figure 3.5(a).

The trajectories which is taken by these six robots are different from the one taken by the control robot.

4.3.5 Summary

In this experiment, the starting point of all robots are on the symmetric line y=250, and the orientation of all robots are aligned with the symmetric line. For any robot, as long as there is no difference between any of the parameters on the left and the corresponding parameter on the right, it was not distracted from the path taken by the control robot. Examples can be found on the control robot R_0 and W_d.

Secondly, the testing arena is designed to distract the robots from the symmetric line y=250. For the robots which were distracted early, the type of hardware difference found on the robot have large influence on the path it took. On the other hand, for robots which were later distracted away from the symmetric line, the type of hardware difference found on the robot have small influence on the path it took. In other words, the influence of the parameters (whose values are varied with a fixed percentage) to the trajectory in the testing arena can be ranked from large to small as:

1. gain of motor drive and wheel radius
2. IR sensor saggital offset angle
3. IR sensor gain
4. IR sensor lateral offset angle
5. IR sensor viewing angle
6. IR sensor height

The parameter of wheel separation is not in the list, as its influence to the path of the robot can not be explored by this type of testing arena.

4.4 Further Experimental Results and Discussion

It is discovered from the previous experiment that when the magnitude of the hardware difference was 1%, robots took different trajectories comparing with the control robot. However it is not known how small the magnitude is, so that the robot's trajectory will no longer be influenced. Therefore several additional experiments are carried out to discover the smallest magnitude of hardware variation which can influence trajectories taken by the robots in the testing arena.

4.4.1 Robots

Five additional experiments are simulated, in which different magnitudes (in percentage) of hardware difference are applied to the robots, which are summarized in Table 4.3. In each experiment, there are 16 robots and one of them is the control robot. For each of the rest 15 robots, only one particular parameter is varied by a certain percent comparing with the control robot. Different parameters are varied among the rest 15 robots.

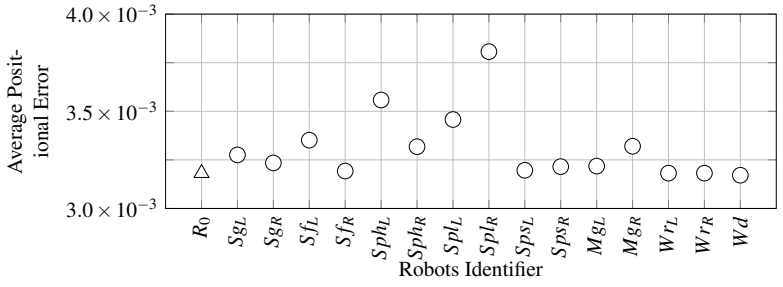
Table 4.3: Different Magnitudes of Hardware Difference: In each simulation, hardware variations of the robot have different magnitudes. For instance in No.1, the value of each parameters of the robot is 1% smaller than that of the control robot. And robots in experiment No.6 are more similar to the control robot since there is only $1e^{-5}\%$ difference in terms of the values of the selected parameters.

Simulation No.	1	2	3	4	5	6
Magnitude (%)	1	0.1	0.01	$1e^{-3}$	$1e^{-4}$	$1e^{-5}$

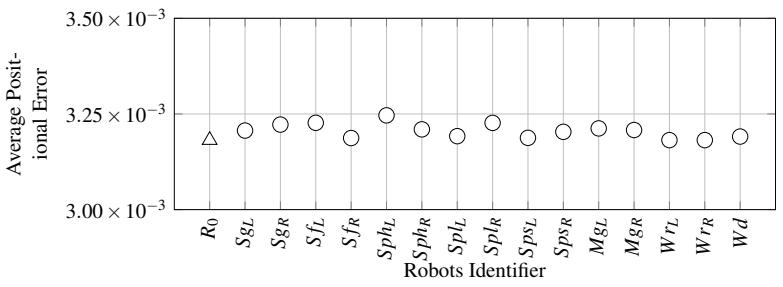
4.4.2 Controller Parameter Selecting

During each experiment, after the robots are prepared, the controller parameters of the robots are selected and applied. The average positional error E_{avg} (defined in Section 3.4.1) during the parameter selecting process for the robots in all experiments are shown in Figure 4.6.

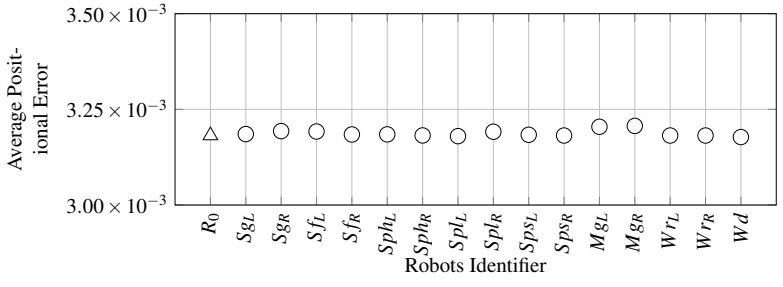
When the magnitude of the hardware difference is 1%, the points in Figure 4.6(a) which refer to robots' positional error are scattered in the region $y \in (0.003, 0.004)$. As the variation magnitude decreases in the rest experiments (Figure 4.6(a) - 4.6(f)), these points slowly converge to the same horizontal position where the point of the control robot locates. This can be interpreted as that the trajectories taken by the robots in the controller parameter selection arena became more similar to that of the control robot. In other words, the behaviours of robots in the parameter selection arena are almost homogeneous. The variance of the average position errors E_{avg} for the robots in each experiment is calculated and shown in Figure 4.7.



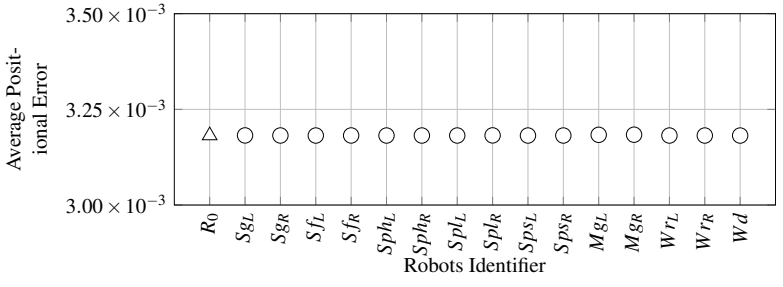
(a) Average Position Error for Robots with 1% Hardware Difference



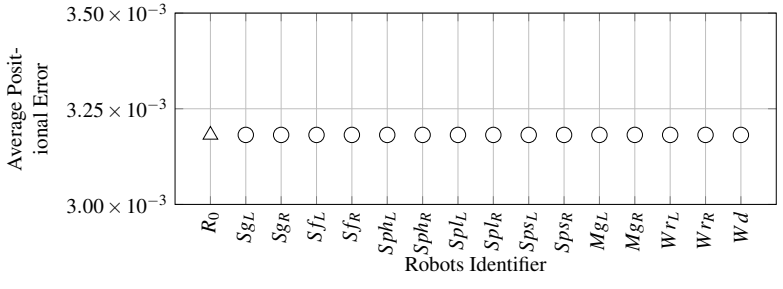
(b) Average Position Error for Robots with 0.1% Hardware Difference



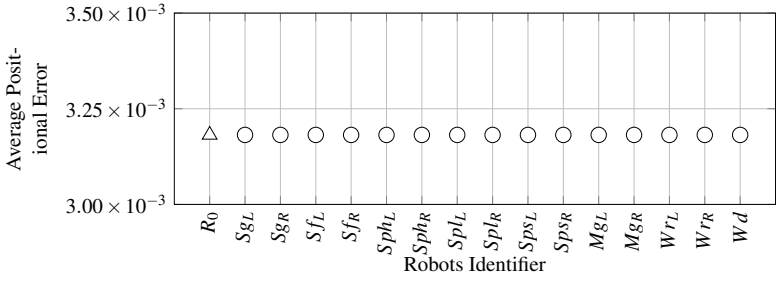
(c) Average Position Error for Robots with 0.01% Hardware Difference



(d) Average Position Error for Robots with $1e^{-3}$% Hardware Difference



(e) Average Position Error for Robots with $1e^{-4}$% Hardware Difference



(f) Average Position Error for Robots with $1e^{-5}$% Hardware Difference

Figure 4.6: Average Position Error of Robotic Groups with Different Magnitude of Hardware Variation

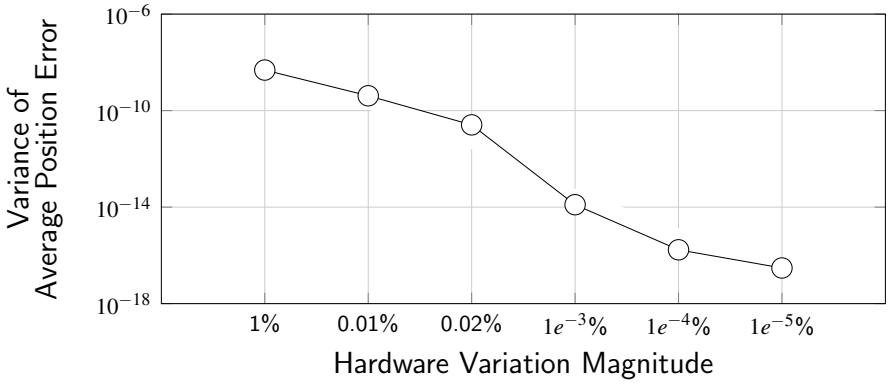


Figure 4.7: The variances of the tuning errors of robots in each group decrease when the magnitude of hardware variation decreases.

It is discovered that When the magnitude of hardware variation decreases, the trajectories taken by the robots become more and more similar to that taken by the control robot.

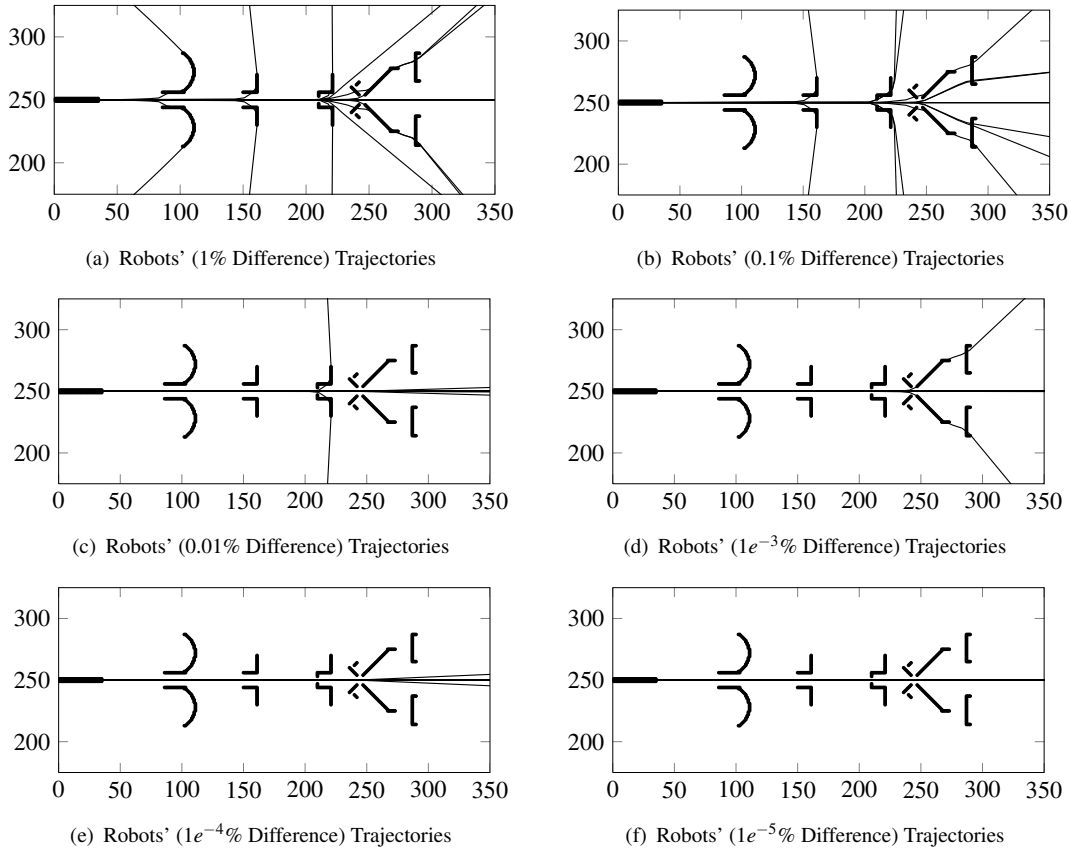
4.4.3 Results and Discussions

After the controller parameters for each robot were selected, the robots were then simulated in the testing arena (Figure 4.3). The trajectories of all robots are in Figure 4.8, which shows an consistency with the change of the average position errors. Robots with different magnitudes of hardware variations generate different trajectory patterns in the parameter selecting arena and as the magnitude of hardware difference decreases, robotic trajectories become more convergent to that of the control robot.

For instance, when the magnitude of the hardware variation is 1%, 16 robots, including the control one, end up in 12 different locations when they reached the border of the testing arena, showing in Figure 4.8(a). When the hardware variations decrease from 0.1%, 0.01%, $1e^{-3}$%, $1e^{-4}$% to $1e^{-5}$% gradually, the number of aforesaid locations also decreased, from 12, 5, 3, 3 to 1. The divergence of the trajectories is closely related with the magnitude of hardware variation. The larger the magnitude is, the more diverse the trajectories are and vice versa.

Secondly, the divergence of the robots' trajectories can still emerge even when the magnitude of hardware variation is as small as $1e^{-4}$%. Although the majority of the robots in the group generate the same trajectory with the control robot, some of them can still diverge at the last bifurcation point, showing in Figure 4.8(e). This consolidates the hypothesis that even though hardware variation is small, it can still influence robotic trajectories.

Lastly, it is found that the strength of behavioural influence is different when same magnitude of hardware variation is applied to different parameters. In other words, some parameters can withstand a comparably high magnitude of variation without showing difference in the trajectories; while for others, even if the hardware variation are very small, the robot can still take different trajectories.



(a) Robots' (1% Difference) Trajectories

(b) Robots' (0.1% Difference) Trajectories

(c) Robots' (0.01% Difference) Trajectories

(d) Robots' ($1e^{-3}$% Difference) Trajectories

(e) Robots' ($1e^{-4}$% Difference) Trajectories

(f) Robots' ($1e^{-5}$% Difference) Trajectories

Figure 4.8: Trajectories of the Robots with Different Magnitudes of Hardware Variations: For all experiment, all robots started at the same coordinate (0,250) with the orientation to the right. The direction of travel is to the right. When 1% of hardware difference is applied, the robots took different trajectories. When the magnitude of the variation decreases, less difference is found in their trajectories. Even $1e^{-4}$% difference is added to the robotic hardware parameters, difference in their trajectories can still be observed in Figure 4.8(e). Until the hardware difference is further reduced to $1e^{-5}$%, the robots took almost the same trajectories showing in Figure 4.8(f).

4.5 Conclusion

In this chapter, a group of robots with minor hardware difference were tested in a line-following task. Robots are all derived from the control robot by applying a fixed percentage value to a particular hardware parameter for each of them. The trajectories taken by the robots are compared with that of the control robot.

All robots were firstly required to follow a continuous reflective line to select the parameters for the PI controller which can help the robot achieve minimal average positional error when following the controller parameter line. It is found that when each of the hardware parameter values was reduced by 1% in turn, the robot could still take a trajectories which is very similar to the target line and the average position error is very small comparing with the size of the robot.

As all robots with the selected controller parameters were able to follow the parameter selecting line accurately and follow almost the same trajectory, the robots hardware differences were partially compensated by the controller parameters selected and showed almost homogeneous behaviours in the parameter selection arena.

After the controller parameters were selected, all robots were tested in the testing arena. The testing arena is designed to be symmetric in order to differentiate the trajectories of the robots. It is discovered that as long as difference exists between left parameter of a robot and the corresponding parameter on the right, the robot can be distracted from the symmetric line which is the trajectory taken by the control robot.

To be specific, this testing arena requires the symmetric values for all left and right parameters. Therefore any difference between the left parameter can corresponding parameters on the right will cause the robot takes a path which is different from the control one. If the robot was distracted from the centre early, the hardware difference of the robot has large influence to the trajectory in this arena.

Therefore it is discovered that when the parameters of the robot are individually varied with a fixed percentage value, different parameters influence the trajectories of the robot differently. Some parameters such as gain of motor drive, wheel radius, IR sensor saggital offset angle, etc. have large influence over the trajectory taken in this testing arena. Others have a smaller amount of influence.

Additional experiments were conducted in which the fix percentage of hardware difference was reduced gradually for the varied robots. It is revealed that the divergence of the trajectories of the robots in the group was lessened as the magnitude of hardware difference was decreased. Robots' trajectories was different comparing with the control robot, even when the magnitude of hardware variation is as small as $1e^{-4}\%$.

Chapter 5

The Mechanism of Robot Chromatography

In the previous chapter, a group of robots with minor hardware difference were tested using a line-following task. It was found that minute hardware difference could result in robots taking different trajectories comparing that of the control robot and showing different behaviours in the testing arenas. The investigation in the previous chapter was conducted on the robots that each robot has only one type of hardware differences.

In practice, a robot used in a swarm may be different from others in the group in multiple parameters. Therefore in this chapter, a group of robots, varing hardware parameters determined randomly to model realistic scenario of hardware variations, will be used.

A method of sorting robots according to their behaviours, which is adapted from the chromatography experiment in chemistry, is used to investigate how hardware variation influences behaviour of a swarm.

The structure of this chapter is: chemistry chromatography is introduced in Section 5.1 as well as the reason for using this approach. The experimental design is described in Section 5.2 and 5.3. The results are presented and discussed in Section 5.4, with the conclusions in Section 5.5.

5.1 Methodology

5.1.1 Chromatography in Chemistry

Chromatography is a general term of the chemistry techniques used to separate a mixture of substances for both preparative and analytical purposes. One example is the column chromatography, in which the separation of a mixture of substances happens in a vertical column. Usually the mixture of substances is dissolved in a liquid solvent such as ether, hexane, etc. The solvent

containing the mixture is called mobile phase. The mobile phase will flow through the stationary phase under gravity. The most common stationary phase for column chromatography is silica gel, alumina, etc. As the mixture of substances in the mobile phase have different travelling speeds in the stationary phase due to different characteristics of the substances in the mixture, for instance molecule size, thus the mixture is separated over time (Ettre, 1993).

Figure 5.1 illustrates the process of a column chromatography experiment. Figure 5.2 are series of photos taken at different times during a column chromatography experiment (Zlatich, 2013). In column chromatography, the pressure applied to the mobile phase is effectively caused by gravity which continuously forces the mobile phase to move downwards through the stationary phase.

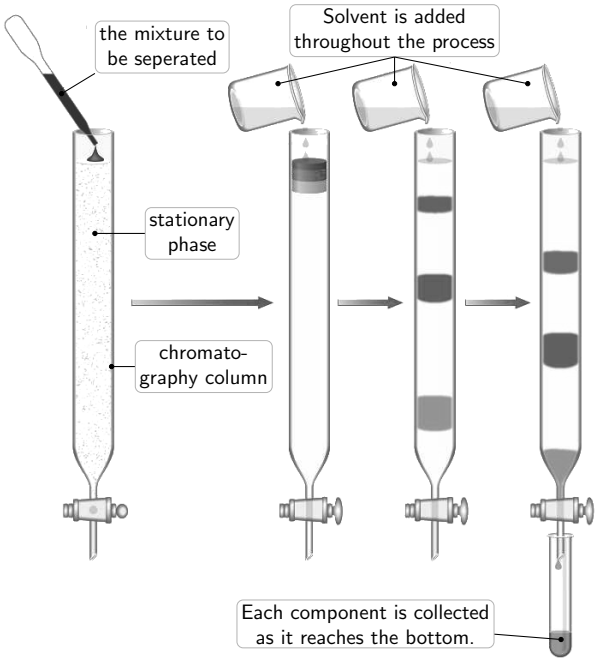


Figure 5.1: Column Chromatography Experiment in Chemistry (adapted from http://www.m2c3.com/chemistry/VLI/M4_Topic2/la_16_07.jpg): The mobile phase contains the mixture of substances to be separated. The mobile phase is then added to the vertically-placed chromatography column which contains the stationary phase. Due to gravity, the substances in the mixture travel downwards. As different substances in the mixture have different travelling speeds, different substances in the mixture reach the end of the column at different time, allowing separation.

The separation is based on differential partitioning between the mobile and the stationary phases (Harwood and Moody, 1989). For instance, substances in the mixture have different molecular sizes. Compound with small molecular sizes can go through the substance in the stationary phase very fast while large ones have difficulties or can not go through at all; in some cases the non-covalent force such as hydrogen bond between molecules in the mixture and the substances in the stationary phase slows down the travelling speed, and the mixture is separated over time (Snyder and Dolan, 2010).

Figure 5.2: Examples of a Column Chromatography Experiment (Zlatich, 2013): These five photos are taken at different times of a column chromatography experiments (starting from left to right). Substances in the mixture have different colours. They are poured into the column from the top at the beginning. After some time, the substances slowly separate into different layers in the column.

Apart from the difference in the interaction between the mobile phase and the stationary phase, successful separation also relies on the pressure (in the column chromatography case, pressure is caused by gravity) which consistently pushes the mixture to go downwards through the stationary phase (Miller, 2009). Without the pressure, the mixture may just stay at the top of the column and interactions would never happen. Sometimes, in order to get higher resolution for the separation, additional pressure with the help of a pump, is applied to the mobile phase, known as 'High Performance Liquid Chromatography' (Dong, 2009). The additional pressure helps the mobile phase interact at higher rate with the stationary phase, resulting higher resolution of separation. The pressure which forces the mobile phase to interact with the stationary phase is essential to the separation.

Figure 5.3 summarizes the important factors of chemistry chromatography: unique chemical behavioural characteristics, pressure and the required time and space. The separation of the mixture in the chromatography experiment relies on the interaction between the substances in the mixture and the stationary phase. Given enough time, space and pressure, subtle difference of the interactions between substances is accumulated, hence the substances separate.

Figure 5.3: The Process in Chromatography in Chemistry

5.1.2 Chromatography for Swarm Robots

In chemistry, only when the substances in the mixtures have different characteristics or behaviours when interacting with the stationary phase, their minor behavioural difference can be accumulated and shown at the global level. If there is no behavioural difference or the behavioural difference can not be explored by the stationary phase, no separation will occur. Robots in a swarm are subject to hardware variations, and each of them have a unique hardware

circumstance which triggers unique behavioural characteristic. Given this, robots are quite similar to the substances in the mixture which have different behaviours when interacting with the stationary phase.

In addition, it is discovered in Chapter 4 that robots with minor hardware difference took different trajectories in the line-following scenario and their unique behaviours were observed. If the unique interaction between the robots and the reflective lines can be accumulated and shown on the global level, robots can be separated according their behaviours. Therefore a special arena needs to be designed so that not only robotic behaviours can be explored but also the interaction can be accumulated for a large quantity.

Pressure is also important to the process as it pushes the mobile phase through the stationary phase and makes the interaction between the mixture and stationary phase happen. Without the pressure, no interaction will happen. In order to separate the robots in a swarm, some form of pressure should be implemented in order to force the robots to interact with the arena.

To ensure satisfactory separation, the simulation time and the size of the arena should be as large as required. In this case, the arena need to be big enough and the simulation time should be long.

In general, to separate robots in a swarm, the experiment for swarm robots should have similar factors and the same process that the column chromatography experiment in chemistry has, showing Figure 5.4 and Table 5.1.

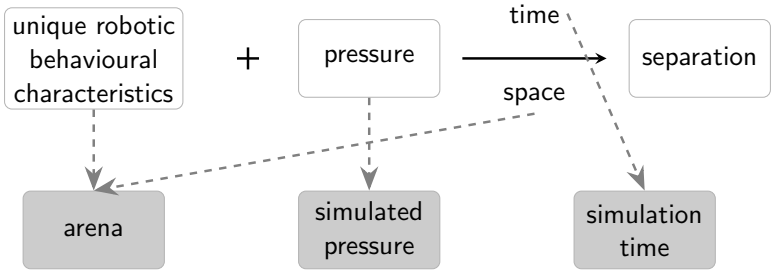


Figure 5.4: From Chemistry Chromatography to Robotic Chromatography: There are four essential factors for chromatography in Chemistry. In order to implement chromatography for robots, these four essential factors are realized. A special arena is designed to explore robots' unique behavioural characteristics and the arena will be big enough to accommodate the lengthy separation process during simulation. The effect of pressure will be simulated in order to push the robots to go through the arena. The time needed to accumulate the minor behavioural characteristic during each interaction will realized by the lengthy simulation time.

If the conditions of time and space are fulfilled, the robot's minor behavioural characteristic during the interaction with the environment can be accumulated, shown on a global level, which possibly leads to a separation. Therefore it is believed that the idea of chemistry chromatography techniques can be adopted to implement a method which can sort or separate robots in a swarm according to their behaviours. In the next section, the implementation of the four factors are discussed.

Factors	Chemistry Chromatography	Chromatography for Swarm Robots
1	Unique physical characteristics of the chemicals	Unique behavioural characteristics of the robots
2	Pressure (caused by gravity or pressured gas)	Simulated pressure
3	Space	Arena
4	Time	Simulation time

Table 5.1: Comparison between Chromatography Experiment in Chemistry and Swarm Robots

5.2 Experimental Design for Robotic Chromatography

To implement chromatography experiment for swarm robot, the experimental design consists of the arena design, and the design of the applied pressure.

5.2.1 Arena Design

A special type of arena was designed to fulfil the following requirements:

1. The arena can explore robots' unique behavioural characteristics.
2. The arena should encourage many interactions for the robots, so that the behavioural differences can be accumulated.
3. The arena should be large enough to accommodate the lengthy accumulation process.

A portion of the arena is shown in Figure 5.5. It is covered with reflective lines with fixed length and random orientation. The middle point of the lines are aligned to the grid of the arena.

The line pattern was chosen as the pattern of the arena. When robots encounter a reflective line, some robot will follow the line and others will be directed to other directions depending on the hardware circumstance as well as the position and orientation of the robot. During the interaction, the robot changes its speed and direction of movement based on information perceived by the sensors. Most importantly the behavioural characteristics of the robot can be explored.

The grid used to determine the location of the line is illustrated in Figure 5.6. Firstly the arena is divided into many identical hexagon-shape cells which are adjacent to each other. The hexagon cell is chosen because hexagon shape utilizes the arena area fully without any uncovered space. Secondly, the centre point of individual line is aligned with the centre of the hexagon cell which offers a better spread of the line all over individual cell. The length of the line is proportional to the size of the hexagon cell. Lastly, the orientation of the line is randomly determined by a uniformly distributed random number generator. In other words, the arena is fully utilized by

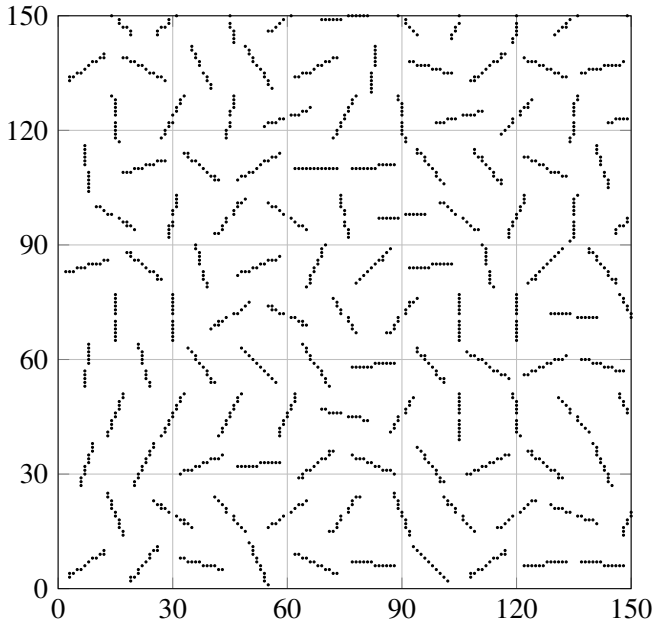


Figure 5.5: A Portion of the Arena for Robotic Chromatography Experiment: Comparing with the size of robot (10×10) in arbitrary size, the size of the area shown is 150×150 also in arbitrary unit. The arena is covered with the reflective lines. The lines have a fixed length of 10. The centre point of the lines are located on the grid of the arena. The orientations of the lines are randomly determined.

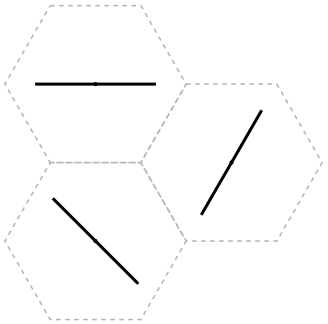


Figure 5.6: Arena Grid and Location of the Reflective Lines: The arena grid is determined by the hexagon cells which fully cover the whole arena. In each cell, the centre points of the reflective line is aligned with the centre point of the cell. The length of the line shown is 10 in arbitrary unit, comparing to the size of the robot 10×10. The orientation of the line is determined by a uniformly distributed, randomly generated numbers. Only the reflective lines can be seen by the robot.

hexagon-shape cells with a randomly orientated reflective line at the centre of each cell, thus the arena is fully utilized to maximize the number of interactions between the robots and the environment.

Only the reflective lines can be seen by the robot and the hexgon cells are used just for placing the lines correctly.

The last requirement for the arena design is that the arena should be large enough to accommodate the behavioural accumulation process. Due to the fact that MAT file in Matlab which is used to store the arena information has a limited size, it is impossible to create an arena with very large size. An approach of shuffling a number of smaller arenas to form a much bigger arena is adopted. These smaller arenas are used in a loop: once all arenas have been used, the first one is reused and so on. This process is illustrated in Figure 5.7. Effectively, the arena which robots run in is a seamless combination of a number of smaller arenas.

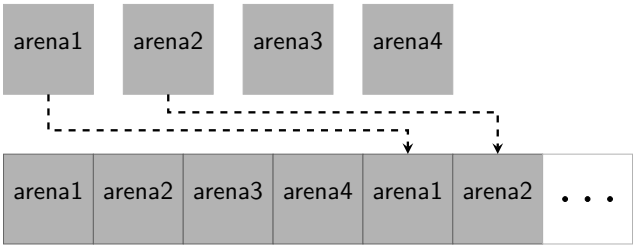


Figure 5.7: Constructing the Arena with a Large x Axis: Firstly a number of smaller arenas (arena1, arena2, arena3 and arena4) are created. During the simulation, once a robot reaches the right boundary of arena1, then it automatically enter arena2 from the left side. Effectively, the arena which robots run in is a seamless combination of a number of smaller arenas.

To ensure that the arena, effectively a significant larger y axis, can be modelled, a mechanism illustrated in Figure 5.8 was designed for the simulation. Once a robot reaches the bottom or top boundary of the arena, it automatically reappears on the other side of the arena with maintained orientation.

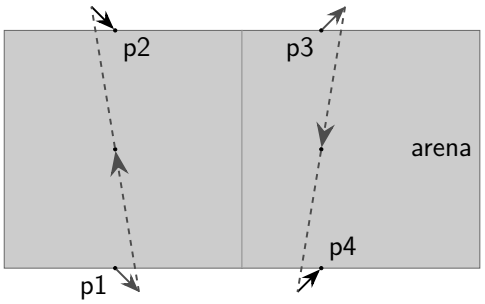


Figure 5.8: Top and Bottom Boundary Re-entering Mechanism: Once a robot reaches the top or bottom boundary of the arena (for instance from the coordinate p1 or p3), the robot will reappear on the other side of the arena (from the coordinate p2 or p4) with the same orientation.

To have a better understanding of the top and bottom boundary re-entering mechanism, it can be considered that the arena is wrapped on the surface of a pipe, then the top and bottom boundary of the arena is connected to each other. As a result, the robot runs on the surface of the pipe, equivalently the robot is able to re-enter the top boundary of the arena with the same orientation which it has when reaching the bottom boundary of the arena.

In this case, an arena with both large x and y axis is obtained. In addition because of the arena top and bottom boundary re-entering mechanism, only the x coordinate of the robot is of more importance comparing with the y coordinate.

5.2.2 Simulation of the Chromatography Pressure

In chemical chromatography, the pressure is used in order to force the substance to pass through the stationary phase and encourage the interactions. As in chemical chromatography the simulated pressure will be applied along the direction of travel, hence the x axis.

As the orientation of the robot will also change due to the robot's interaction with the environment using the two IR sensors which are located in the front of the robot, if the direction of the pressure is aligned with the direction of the IR sensors, the number of interaction can be increased.

While orientating the robot to the right is the necessary to the separation of the robots, the influence of the pressure affecting the robot's orientation should be applied in a way that the interaction between robot and reflective line pattern will not be significantly influenced. Otherwise, the dominant factor is the pressure and the behavioural characteristics of robots will not explored and accumulated. Therefore any change of the robot's orientation should be relative to the current orientation of the robot.

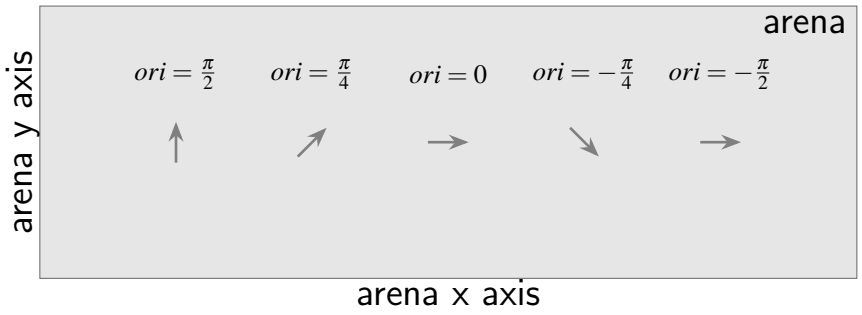


Figure 5.9: Robot's Orientation and Corresponding Direction in the Arena: The arrows refer to the direction of the arena which the robot is facing to. if the robot faces to the right of the arena, the orientation of the robot is 0. If the robot faces the positive y axis of the arena, its orientation is $\pi/2$. If the robot faces the negative y axis of the arena, its orientation is $-\pi/2$.

The effect of the simulated pressure applied to the robots is expressed in Equ 5.1 and 5.2. The robot's orientation and the corresponding direction in the arena are illustrated in Figure: 5.9

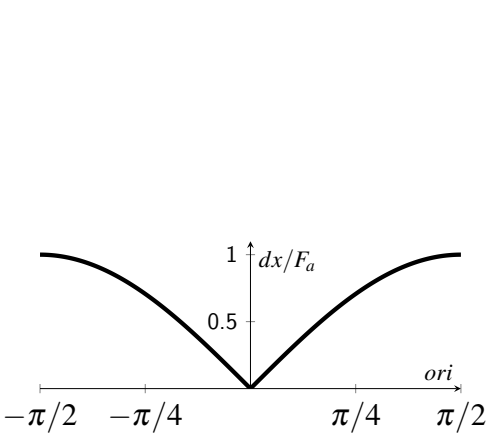
$$\begin{aligned} dx &= F_a \cdot \left|\sin(ori^{t-1})\right| \\ d(ori) &= -F_b \cdot \tan(ori^{t-1}) \end{aligned} \tag{5.1}$$

$$\begin{aligned} x^t &= x^{t-1} + dx \\ ori_j^t &= ori_j^{t-1} + d(ori) \end{aligned} \tag{5.2}$$

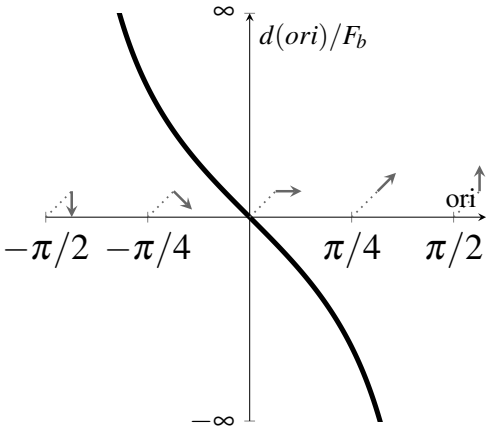
where

dx is the change of x coordinate which should be applied to the robot at simulation time t
$d(ori)$ is the change of orientation which should be made to the robot at simulation time t
x^t is the x coordinate of the robot at simulation time t,
x^{t-1} is the x coordinate of the robot at simulation time $t-1$,
ori^t and $ori^{t-1}$is the orientation of the robot at simulation time t and $t-1$,
F_a is the simulated pressure influencing the robot's x coordinate,
F_b is the simulated pressure affecting the orientation of the robot.

From the equation, dx is always a positive number showing in Figure 5.10(a). $d(ori)$ can be either positive or negative depending on the orientation of the robot, showing in Figure 5.10(b).



(a) If the difference between the robot's orientation and the right direction of the arena exists, the x coordinate of the robot will be increased since F_a is a positive constant. If the difference is large, the increment is large, and vise versa.



(b) The direction which the robot is facing in the arena is illustrated with arrows when its orientation is equal to the value of the point on the x axis of this figure. For instance, when the robot's orientation is 0, it is facing the right direction of the arena. If the orientation is $\pi/4$, the robot is facing to the right upwards of the arena.
Since F_b is a positive constant, if the orientation of the robot is positive, meaning the robot is facing upwards to the positive y axis of the arena, its orientation will be reduced by a number proportional to the absolute value of the tangent of the orientation. If the orientation is negative, the orientation will be increased.

Figure 5.10: Simulated Pressure Impacts Robot's x Coordinate and Orientation: Orientation of the robot in relation to its direction in the arena is explained in the figure below.

For the change of robot's x coordinate, the equation can be interpreted as that as long as the robot's orientation and direction of the simulated pressure is different, a positive number is added to the x coordinate of the robot. In other words, the speed of each swarm robot at each simulation step is the vector addition of the swarm robot's speed and a subcomponent of speed due to the simulated pressure.

For the change of robot's orientation, the equation can be interpreted as that as long as the robot's orientation and direction of the simulated pressure is different, an angular speed is added to the swarm robot which force the swarm robot to orientate to the right.

The implementation of the simulated pressure which is similar to the pressure applied in chemistry chromatography experiment was described. The simulated pressure will not only increase the swarm robot's x coordinate but also ensures that the swarm robots continue to face the right-hand side of the arena by altering the orientation of the robot.

5.3 Design of the Robotic Swarm

During the experiments in Chapter 4, each of the varied robots has only one type of hardware variation. Specifically, comparing with the control robot, one of the parameters on each of the varied robot is different from the control one. Whereas in reality, multiple hardware parameters of one robot are different from those of others and hardware variation across all robots in a swarm are different. Hence for robots used in this experiment, all parameters of individual robot will be randomly varied.

The deviations for individual parameters of all robots in a swarm can be different in practice. Sensor gain and viewing angle usually have small sigma value due to more strict quality control during the manufacture. Sensor assembly variation (sensor height, lateral offset and sagittal offset) are comparably small comparing with variations on the motor and gearbox gain, and wheel separation. Thus the random numbers used to vary individual parameter of all robots will follow Gaussian distributions with different deviation, equivalently the sigma values for the distributions will be different. As listed in Table 5.2, three sigma values (large deviation $\sigma = 0.05$, medium deviation $\sigma = 0.03$, small deviation $\sigma = 0.01$) are used to generate random number sequences with different distributions.

Table 5.2: There are 13 parameters to be varied on each robot (two IR sensors and two wheels). The parameters of the robot were varied with Gaussian-distributed random numbers which have different deviations, equivalently different sigma values.

Component	Parameter	Description	Sigma
IR sensor	α	gain	$\sigma = 0.01$
	$\measuredangle v$	view angle	$\sigma = 0.01$
	h	height	$\sigma = 0.03$
	O_l	lateral	$\sigma = 0.03$
	O_s	sagittal	$\sigma = 0.03$
Motor drive	m	gain	$\sigma = 0.05$
Wheel	d	separation	$\sigma = 0.05$

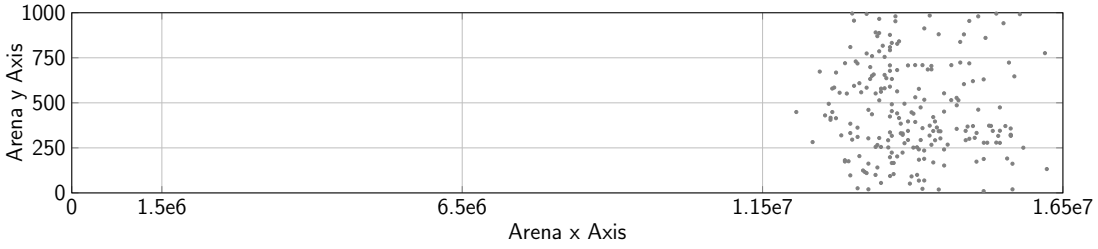
With the method described, 210 robots were generated for this experiment. Individual parameter of all robots were varied with a sequence of Gaussian-distributed random numbers with specific sigma value. The parameters α, $\measuredangle v$ used sequences of random numbers with $\sigma = 0.01$. The

parameters h, O_l, O_s used sequences of random numbers with $\sigma = 0.03$. And the parameters m and d used sequences of random numbers with $\sigma = 0.05$.

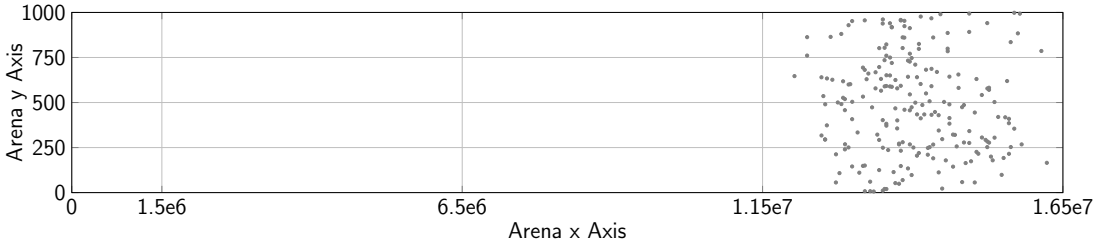
The controller parameters of all robots were selected using the method described in Section 3.4.1.

5.4 Results and Discussion

The group of the robots were simulated in two arenas (Arena1 and Arena2) which were generated using the method described in Section 5.2.1. The location of all robots are shown in Figure 5.11. All robots started at the coordination (0, 500) individually. As discussed previously, no interaction between robots were modelled at this stage of the research. After they travelled from $x = 0$ to $x = 1.65e7$, their final positions were scattered in the range $1.15e7 \leq x \leq 1.65e7$.



(a) Robots' Location in Arena1



(b) These Robots' Location in Arena2

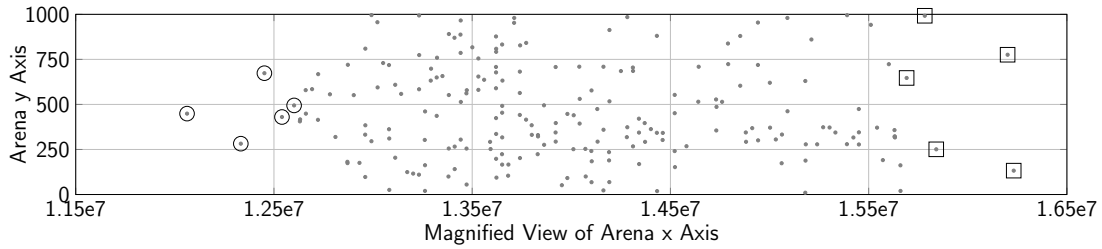
Figure 5.11: Robots' location in Arena 1 and 2 with full view of x axis. Dots in both figures denote the location of 210 robots after same period of simulation time. In both experiments, all robots set off from the same coordination $(0, 500)$ and they moved towards right. When the simulations ended, robots scattered within the range $1.15e7 < x < 1.65e7$.

5.4.1 Similar Orders of Robots in Two Arenas

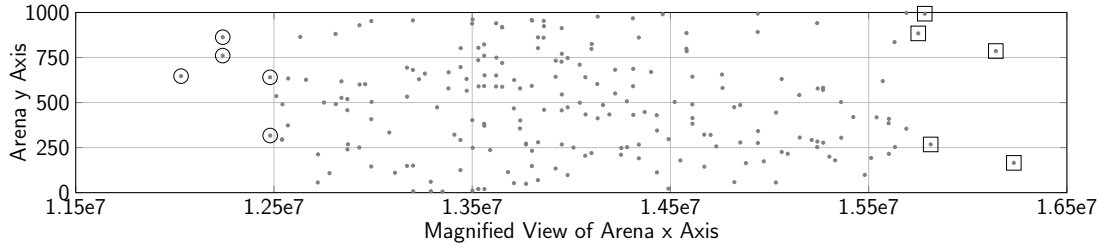
The difference between the two arenas is the orientations of reflective lines as discussed in Section 5.2.1. A sequence of uniformly distributed random numbers was used to determine the orientation of every lines in the arena. Due to different sequences used for the two arenas, the orientation of individual lines in these arenas are different.

Additionally, only the x coordinate matters when comparing locations of the robots. Thanks to the arena re-enter mechanism defined in Section 5.2.1, if the robot reaches either upper or bottom boundary of the arena, it will re-enter the arena from either bottom or upper boundary of the arena with the same x coordinates and orientation.

The locations of the robots in two experiment are shown in Figure 5.12 with magnified view of the x axis. The first five robots located on the right of the robot separation in Arena1 are analysed. It is found that the same five robot can also be found on the right of the robot separation in Arena2 (marked with squares in Figure 5.12). In other words, robots which have large x coordinates in Arena1 also have large x coordinates in Arena2. It is the same situation for the five robots tailed in the separation of the robots (marked with circles in Figure 5.12).



(a) Farthest and Nearest 5 Robots' Location in Arena1



(b) These Robots' Location in Arena2

Figure 5.12: Magnified View of the Two Arenas. Dots in both figures denote the locations of robots. Dots with square markers (in top figure) are the five robots with large x coordinates. These robots also ranked in the top five in Arena2 (square markers in bottom figure) in terms of their x coordinate. Robots with small x coordinates in Arena1 (marked with circle) also tailed in Arena2 (circle markers).

The order of all robots in the separation is analysed by comparing the x coordinates of all robots in Arena1 and Arena2, showing Figure 5.13. As the order of robots in terms of x coordinates of their locations after a fixed simulation time are almost consistent in both arenas (showing in Fig 5.13). Although there are some outliers which do not comply with this consistency, the order of all robots are generally similar in the two arenas. As a result this approach is able to separate hardware-varied robots with an almost the same order regardless of the orientations of reflective elements in the arenas.

Results from Arena1 will be used for the following investigation on the relationship between the x coordinate of robot's location and its hardware parameters.

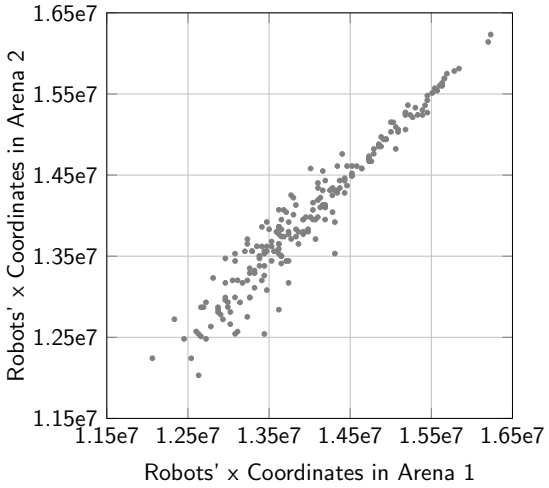


Figure 5.13: Comparison of Robots' x Coordinates in Arena1 and Arena2: The x coordinates (in arbitrary unit) of the robots' locations in Arena1 (showing in the x axis of this figure) are compared to those in Arena2 (on the y axis). The line patten shows the order of the robots in terms of their x coordinates almost the same between the two arenas. For instance, robots which were far away from the starting point (0,500) were also far away from the starting point in Arena2. There are exception for robots the middle and tail part of the separation. But the order of all robots are generally similar in the two arenas.

5.4.2 The Effect of Motor Drive Gain

To investigate the effect of motor drive gain, a variable S_{mgain}, motor drive gain sum, is defined for measuring the left and right motor drive for the robot. For instance, the motor drive gain sum for robot R_m is defined in Equ 5.3.

$$S_{mgain,m} = r_{m,11} + r_{m,12} \tag{5.3}$$

where

$S_{mgain,m}$ is the motor drive gain sum for robot R_m

$r_{m,11}$ is the random number (in percentage) used to determine the value of the left motor drive gain parameter of robot R_m. The left motor drive gain is the 11th parameter of the robot.

$r_{m,12}$ is the random number (in percentage) used to determine the value of the left motor drive gain parameter of robot R_m. The right motor drive gain is the 12th parameter of the robot.

If this variable is large, the specific robot has the left and right motor drives with increased speed for a specific input values from the control system as discussed in Section 3.2.3.

It is evident that all robots which have travelled far away from the starting point had better drive train system on both the left and right, in Fig 5.14. For the robots which have powerful left and right motor drive, they are able to run fast on a straight line comparing with other robots with less powerful motor drive. During the experiment, all robots have the tendency of moving to the

right due to the simulated pressure. Therefore robots with powerful left and right motor drive have large x coordinates comparing with others and can be located on the right of the arena.

Although enhanced drive train system is the necessary condition for robots with large x coordinates, not all robots which have large gain on their drive train system travelled far.

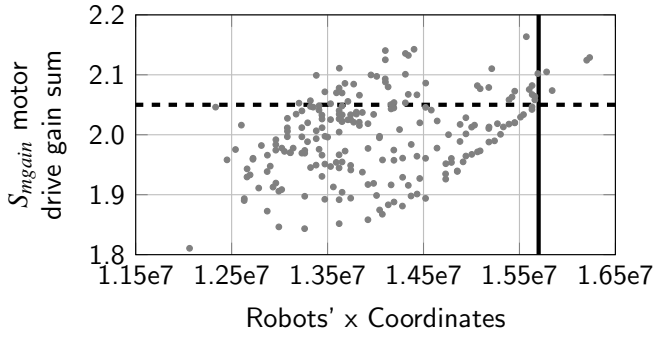


Figure 5.14: Robots' Locations and Their Drive Train: The x axis shows the x coordinates of robots' locations in the Arena1. The y axis shows the sum of left and right motor gain variation (in percentage) of individual robot, equivalently motor drive gain sum $S_{mgain,m}$. The robots which travelled far (which are on the right side of the solid line) have large gain on both left and right drive train system. Not all robots with large gain (which are above the dashed line) have large x coordinate.

After analysing the hardware parameters of all robots with high gain on both their left and right drive train system, there is no clear pattern on other robotic parameters. In other words, the distance between robotic end points and the starting point does not depend on one or two parameters of the robots, instead it may be determined by the combination of multiple parameters of each robot.

5.4.3 Robot Clusters

To identify if the robot's location is determined by a combination of multiple hardware parameters, the clustering approach is used. A variable d_{para}, namely parameter distance, is defined. For instance, the parameter distance between the robot R_m and R_n can be defined using Equ 5.4.

$$d_{para} = \sqrt{\sum_{i=1}^{13} (r_{m,i} - r_{n,i})^2} \tag{5.4}$$

where

r is the random number used to vary the parameter value of the control robot when generating hardware-varied robots

there are 13 parameters which need to be varied (defined in Table 5.2)

$r_{m,i}$ is the random number which are used to determine the ith parameter of robot R_m

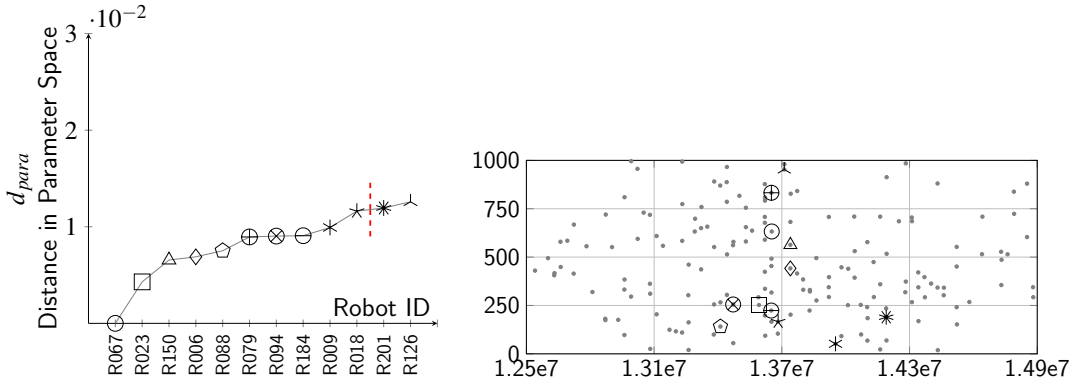
$r_{n,i}$ is the random number which are used to determine the ith parameter of robot R_n

Hence d_{para}, parameter distance, measures the distance between two robots in their parameter space, hence d_{para} tells how different one robot is from the other one.

There are 13 parameters which are varied for every robot in the group. Considering a 13-dimensional space, each parameter represents a dimension in that space. So each robot can be represented by a point in the multi-dimensional space. The variable parameter distance d_{para} calculates the Euclidean distance between two robots in the parameter space.

The clustering method is as follows. A robot is firstly selected, and the distance between this robot and the rest in the group are calculated and sorted from small to large. As a result, robots with similar hardware parameters are identified. The locations of these robots in the chromatography experiment are compared. As the Euclidean distances is used as the metric, therefore the cluster area will be an area with equal radius in all dimensions. The robot selected firstly lies at the centre of the cluster area.

Robot R067's Cluster



(a) The parameter distance d_{para} to the robot R067 of all robots in the group are calculated and sorted from small to large, showing on the y axis. The parameter distance for R067 and itself is zero.

(b) Locations of those robots in Arena1.

Figure 5.15: Location of robot R067 and similar ones and the parameter distance between them: The left figure shows the parameter distance between robot R067 and those robots which are similar to R067. The right figure shows their locations in the experiment. The same marker is used for the same robot in both figures. Starting from the most similar robot to the least similar robot of robot R067, R023, R150, etc. until R018 all have similar x coordinates with that of R067. However the distance on the x coordinate between R067 and R021 is very large.

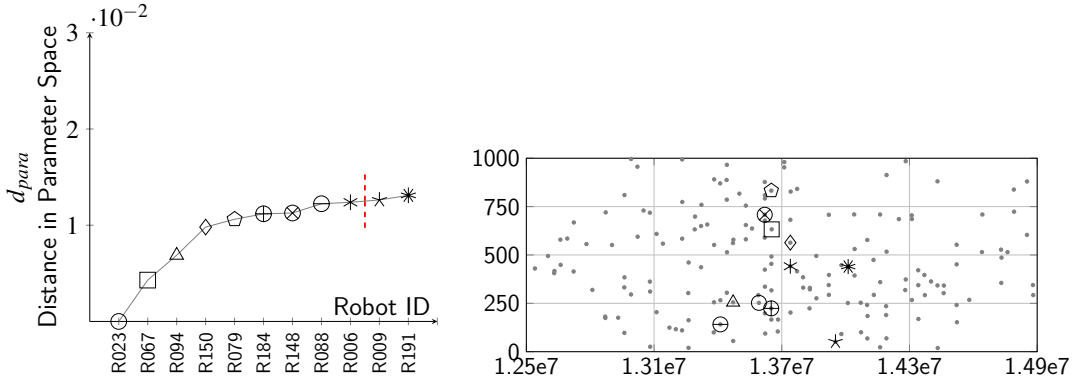
The parameter distance between R067 and the test robots in the group are calculated and sorted from small to large. Robots with similar hardware circumstances are listed on the x axis of Fig 5.15(a). R023 have the smallest parameter distance to R067 and is the most similar robot to R067 in terms of its parameters. R015 is the second most similar robot in the group. The location from the chromatography experiment of robot R067 and those which are similar to R067, showing in Figure 5.15(b).

As the parameter distance between robot R067 and each of other robots in the group are sorted, the selected robots showing on the x axis of Figure 5.15(a) all have similar hardware parameters. It is shown that the most similar ones (R023, R150, R006, R088, R079, R094, R184, R009, R018) have similar x coordinates in the experiment comparing with the x coordinate of R067.

Although R201 marked with 10-point star is very near to R067 in the parameter space comparing with the majority of robots in the group, R201's x coordinate in the chromatography experiment is much larger than those similar robots (to R067). Therefore R201 is not in the same cluster of R067. Since R201 does not belong the cluster, the robots afterwards (which are less similar to R067 in their hardware parameters) are also not in the same cluster. For instance although the robot R126 is just after R201 and it is x coordinate is very similar to the x coordinate of R067, it is still not within the same cluster of R067.

Robot R023's Cluster

From Figure 5.15, R023, which is the second most similar robot to R067, belongs to R067's cluster. If this clustering method is robust, R067 should belong to R023's cluster if it is analysed from R023's perspective. Hence the parameter distance between R023 and similar robots are calculated and sorted in Figure 5.16(a) and locations of these robots are shown in Figure 5.16(b).



(a) The parameter distance d_{para} to the robot R023 of all robots in the group are calculated and sorted from small to large, showing on the y axis. The parameter distance for R023 and itself is zero.

(b) Locations of those robots in Arena1.

Figure 5.16: Location of robot R023 and similar ones and the parameter distance between them: The left figure shows the parameter distance between robot R023 and those robots which are similar to R023. The right figure shows their locations in the experiment. The same marker is used for the same robot in both figures. The robots showing in the x axis of the left figure have similar x coordinates to R023 until R006.

In this case, the robots which are similar to R023 in terms of the hardware parameters are sorted according to their parameter distance to R023. However although R009 is similar to R023's hardware circumstance comparing with the majority of robots in the group, it does not belong to R023's cluster due to the large difference between R009 and R023's x coordinates in chromatography experiment. Thus R009 does not belong to R023's cluster.

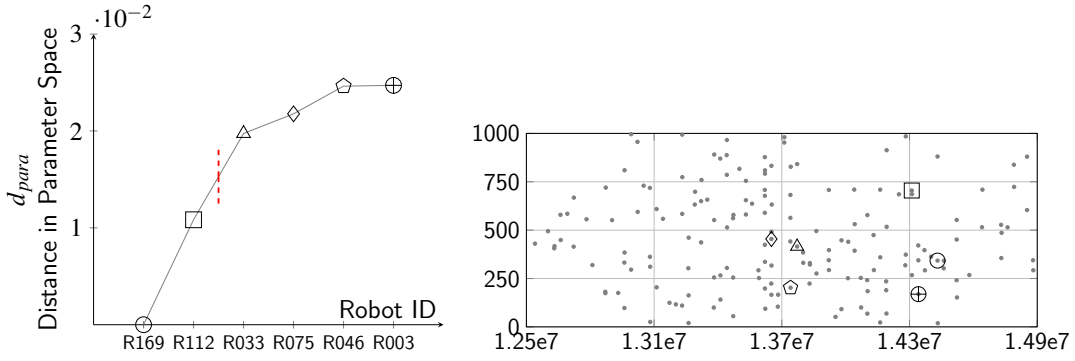
After all robots within R023's cluster are identified, it is confirmed that R067 belongs to R023's cluster. Therefore this is a robust clustering method.

In addition, it is found that robots in R067's cluster also appears in R023's cluster and vice versa. However some robots appeared in R067's cluster does not appear in R023's cluster, illustrated in Figure 5.17. It is evident that the two clusters are partially overlapping with each other in the multi-dimensional parameter space.

Figure 5.17: Partial Overlapping of R067 and R023's Clusters: The first line lists all robots in R067's cluster. The second line lists all robots in R023's cluster. The robots with grey background appear in both clusters. The robots identified with white background only appear in one cluster.

Robot R169's Cluster

The same clustering approach is used to analyse R169 and similar robots in terms of the hardware parameters, results are showing in Figure 5.18. Again the parameter distance between R169 and rest robots in the group were calculated and sorted from small to large. These robots are listed on the x axis of Figure 5.18(a). R112 is the most similar robot comparing with R169 in terms of their hardware circumstances. Both robots have similar x coordinates in the chromatography experiment. Therefore R169 and R112 belong to the same cluster.



(a) Parameter distance between the robot R169 and similar ones

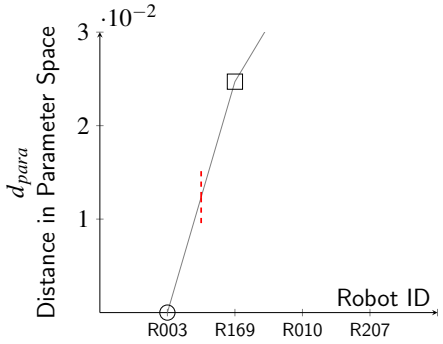
(b) Location of robot R169 and similar ones

Figure 5.18: Location of robot R169 and similar ones and the parameter distance between them: The parameter distance between R169 and rest robots in the group are calculated and sorted from small to large. Similar robots are listed on the x axis of the left figure. Locations of these robots are marked in the right figure with the same marker.

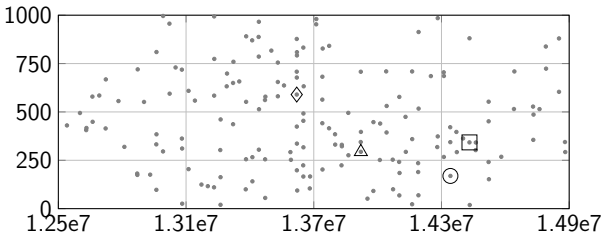
Although R033, R075 and R046 are very similar to R169 in the hardware parameters, they are not belong to the same cluster since the difference between the x coordinates of these robots and that of R169 is large. Although R003 has both similar hardware parameters and x coordinates in the experiment, R003 does not belong to the R169's cluster since the robots which are even closer (R033, R075, R046) to R169 are not within the cluster.

Robot R003's Cluster

Even though R003 does not belong to R169's cluster, the investigation of the similarity between R169 and R003 was conducted from R003's perspective, showing Figure 5.19.



(a) Parameter distance between the robot R003 and similar ones: As the limits of y axis is the same as the previous figures, some robots are beyong the maximum of y axis and not shown.



(b) Location of robot R003 and similar ones

Figure 5.19: Location of robot R003 and similar ones and the parameter distance between them: The most similar robot in the group to R003 is R169. R003 and R169 have similar x coordinate in the Arena1. However the parameter distance between these two robots is large.

The parameter distance between R003 and others in the group are calculated and sorted from small to large. The most similar robot to R003 in terms of the hardware parameters is R169. And their locations in the Arena1 in the experiment are near. However the parameter distance between R003 and R169 is large comparing with the parameter distance in the clusters which are analysed previously. Thus R003 and R169 do not belong to the same cluster. This result is coherent with the result in Figure 5.18.

To interpret this in another way, in the multi-dimensional parameter space, from the previous discussion, R169 is surrounded by R112, R033, R075, R046 and R003 (listed along the x axis of Figure 5.18(a)). If viewing from R003's perspective, R169 is the nearest one to R003 and some other robots are even further. Hence R003 is the only robot in its nearby space.

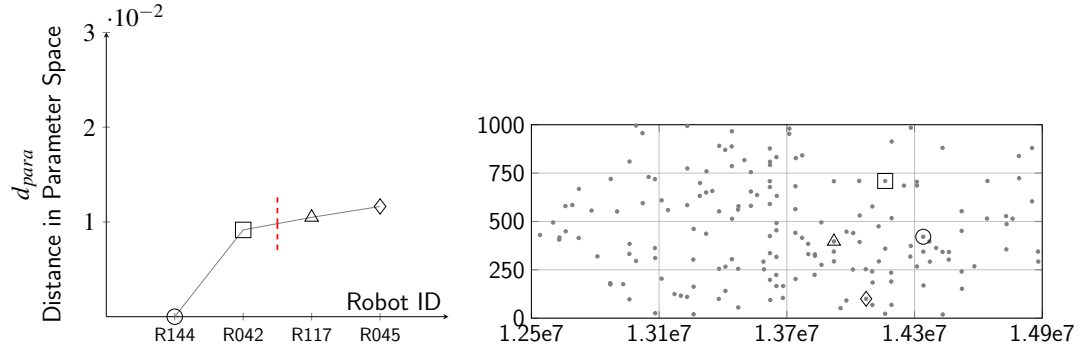
As the results of that R003 and R169 have similar performance in terms of their x coordinate in the chromatography experiment, they does not belong to the same cluster. In other words, robots located in different regions in the multi-dimensional parameter space can obtain similar

x coordinates in the chromatography experiment, thus have similar performance/behaviours in that particular task.

A similar example can be found with Robot R010 and R163, which can be found in Appendix G.

Robots R144 and R100's Clusters

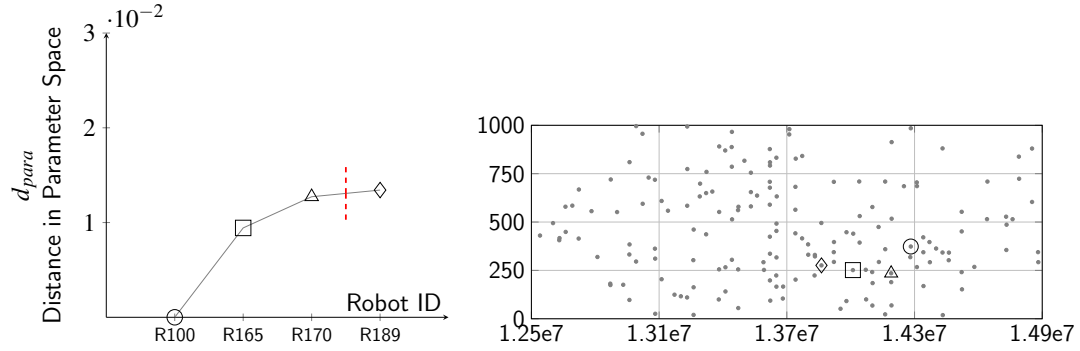
One might argue that although R169 and R003 do not belong to the same cluster, they still are quite near to each other in the parameter space comparing with other robots. Therefore two clusters are found in which robots have similar performance in the chromatography experiment, but they are not near to each other in the parameter space. These clusters are R144's cluster and R100's cluster, showing in Figure 5.20 and 5.21.



(a) Parameter distance between the robot R144 and similar ones

(b) Location of robot R144 and similar ones

Figure 5.20: Location of robot R144 and similar ones and the parameter distance between them



(a) Parameter distance between the robot R100 and similar ones

(b) Location of robot R100 and similar ones

Figure 5.21: Location of robot R100 and similar ones and the parameter distance between them

The robots R114 and R100 have quite similar x coordinates in the chromatography experiment. However they are comparably far from each other in the parameter space. The parameter distance between them is 0.033 which is large comparing with the parameter distance in previous clusters discussed.

Thus robots located in different regions in the multi-dimensional parameter space can still achieve similar behaviours in terms of their x coordinates in the chromatography experiment.

Clusters in Chromatography Experiment

According to the clustering method described above, all clusters among the group of robots located in the centre part of the sorting arena ($1.25e7 \leq x \leq 1.55e7$) were identified. The centre part of the arena is manually separated into five segments. Only clusters in which multiple robots can be found are marked and coloured, showing in Figure 5.22.

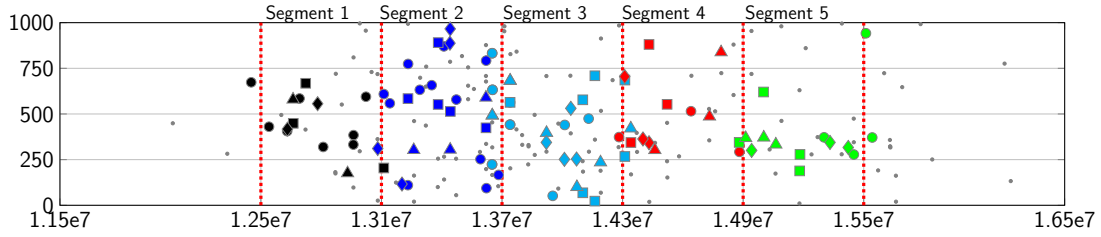


Figure 5.22: Clusters of Robots Located in the Middle of the Separation: The size of the arena is in arbitrary unit. The area ($1.25e7 \leq x \leq 1.55e7$) with the majority of robots is separated equally into five segments by the red vertical lines. In each segments, multiple clusters of robots were identified. Only the clusters with multiple robots are represented with markers. And the cluster which have only one robot are still illustrated with small dot. The same colour are given to the markers within the same segment. Robots in the same cluster have similar hardware parameters and achieve similar x coordinate in the chromatography experiment. Different clusters of robots can achieve similar x coordinates in the chromatography experiment.

5.5 Conclusion

In this chapter, a novel approach for sorting robot in a swarm according to their hardware variations was proposed. A set of robots is derived from a standard robot by adding minor variations in their parameters to model the intrinsic hardware difference that exists in real robotic swarms and they are then simulated in performing a line following task in the arenas covered with randomly oriented IR-reflective patterns.

Results show that this approach is able to sort the group of robots according to their hardware differences. The method of differentiating the robots through the accumulated effect of numerous interactions with the environment is analogous to separating chemical mixtures by chromatography.

To prove that both the sorting arena and the simulated pressure are indispensable to successful separation, two experiments was conducted. In Appendix E, robots were simulated in two arenas either fully covered by reflective materials (totally black arena) or blank arena which is not reflective at all(totally white arena). In the second experiment(Appendix F), robots were simulated in a normal arena without the pulling force.

This sorting method is robust that the sorting of the robots does not depend on the orientations of the reflective patterns in the arena, but on the hardware characteristics of individual robots. It is discovered that the robots which are located on the right of the separation have powerful motor drives both on the left and right, however not all robots with such hardware characteristic located in the same region of the arena.

Instead it is found out that the location of individual robot in the chromatography experiment is not determined by a single parameter but by the combinations of multiple hardware factors. Different combinations of hardware parameters can help robots achieve similar behaviours.

With the help of the newly-defined variable parameter distance and a robust clustering method, robots with similar hardware circumstances are identified, and their x coordinates in the chromatography experiment are compared. Results reveals that robots form into different clusters depending on their hardware parameters. Different clusters may contain different number of robots and some clusters may even overlap with each other. Most importantly it is found that different clusters of robots can achieve similar x coordinates in the experiment.

Further investigate will be undertook in the next chapter in order to improve the sorting efficiency of the swarm chromatography technique.

Chapter 6

Controller's Integration Length and Chromatography Arena Density

In the previous chapter, a novel method of sorting a group of swarm robots according to their unique behaviours caused by hardware variations was proposed. This chapter extends the previous research to improve the sorting efficiency of the swarm chromatography technique.

In this chapter several arenas which have different numbers of reflective lines per unit area are used in chromatography experiments for separating robots with different controller settings, particularly the length for integration. The methodology is discussed in Section 6.1 with the experimental design in Section 6.2 and 6.3. Results are discussed in Section 6.4, with the conclusions in Section 6.5.

6.1 Hypothesis and Methodology

As the robotic chromatography relies on a large number of interaction between the robot and the environment, it is hypothesized that increasing the number of interactions can result in a quicker separation of the robots. However if there are too many reflective lines in the arena, it is unlikely that the separation of robots will occur as expected. To find a good configuration of the chromatography experiment, a number of arenas in which the number of reflective lines per unit area is used for separating robots with different number of interactions.

In addition, the separation of robots with chromatography approach relies on the robots' unique behavioural characteristics. It is also hypothesized that prolonging robots' unique reaction in the arena may benefit the sorting efficiency. To be specific, each robot, due to its unique hardware circumstance, reacts differently when encountering a reflective line. If the robot can somehow continue to perform such reactions even after it has left the reflective line, the location of the robot would be different comparing with one whose reaction is not prolonged. Hence through

the accumulating process of the chromatography, observable difference in terms of robots' locations can happen earlier, resulting in a quicker separation.

The integral term of the robot's PI controller (described in Section 3.2.2) stores and integrates the difference in the voltages output by the two IR sensors from the past in order to correct accumulated errors. In other words, the robot is able to memorized the its experiences in the past which continuously affects instantaneous reaction of the robot. In this case, the robot's reaction is prolonged due to the integration of the PI controller. To investigate if increasing the number of errors to be integrated in the controller can benefit the separation of robots in the chromatography experiment, robots with different lengths for integration of the controller will be used.

Hence in this experiment, several groups of robots with different requirements for integration are used for behavioural sorting in the chromatography experiment with the arena which have different numbers of reflective lines per unit area.

The experimental design is similar to the design in the previous chapter. Minor difference can be found on the arena which is described as follows. The pressure in chemical chromatography was simulated with the same method described in Section 5.2.2.

6.2 Chromatography Arenas

Like the arenas used in the previous chapter, the ones used here were produced with the same method described in Section 5.2.1. Specifically the arena is divided into many hexagon cells which fully cover the arena. Reflective lines locate at the centre of the hexagon cell. The orientation of the line are determined randomly.

As there is one reflective line in each hexagon cell, the number of lines in a arena is equal to the number of hexagon cells. Instead of using a hexagon cell with fix size, different sizes of the hexagon cells are adopted for the experiments of this chapter in order to change the number of reflective lines per unit area.

Although the size of the hexagon cells changes, the length of the reflective line is 10 in arbitrary unit for all arenas used in the experiment of this chapter. (This length of the line can be comparing with the size of robot showing in Figure 3.3.)

In order to measuring the number of reflective lines per unit area, a variable arena pattern density d_p is define, showing Equ 6.1. The variables involved are illustrated in Figure 6.1.

$$d_p = \frac{l}{h} \tag{6.1}$$

where

d_p is the variable defined to measure the number of reflective lines per unit area in the arena

h is the distance between two opposite edges of a hexagon cell
l is the length of the reflective line and equals to 10 in arbitrary unit.

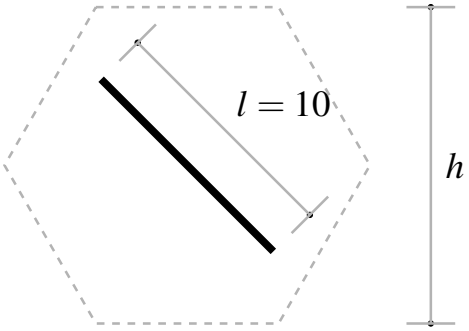


Figure 6.1: As the length of the reflective line is 10 in arbitrary unit, h refers to the distance between two opposite edges of the hexagon, which can be used to determine the size of the hexagon.

As a result, h, the distance between two opposite hexagon edges, can be calculated once the d_p is specified, hence the size of the hexagon can be determined. Figure 6.2 compares the size of hexagon with different d_p arena pattern densities. It is evident that the arena pattern density d_p also determines the gap around the reflective line.

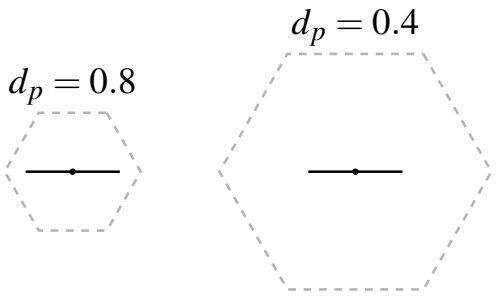
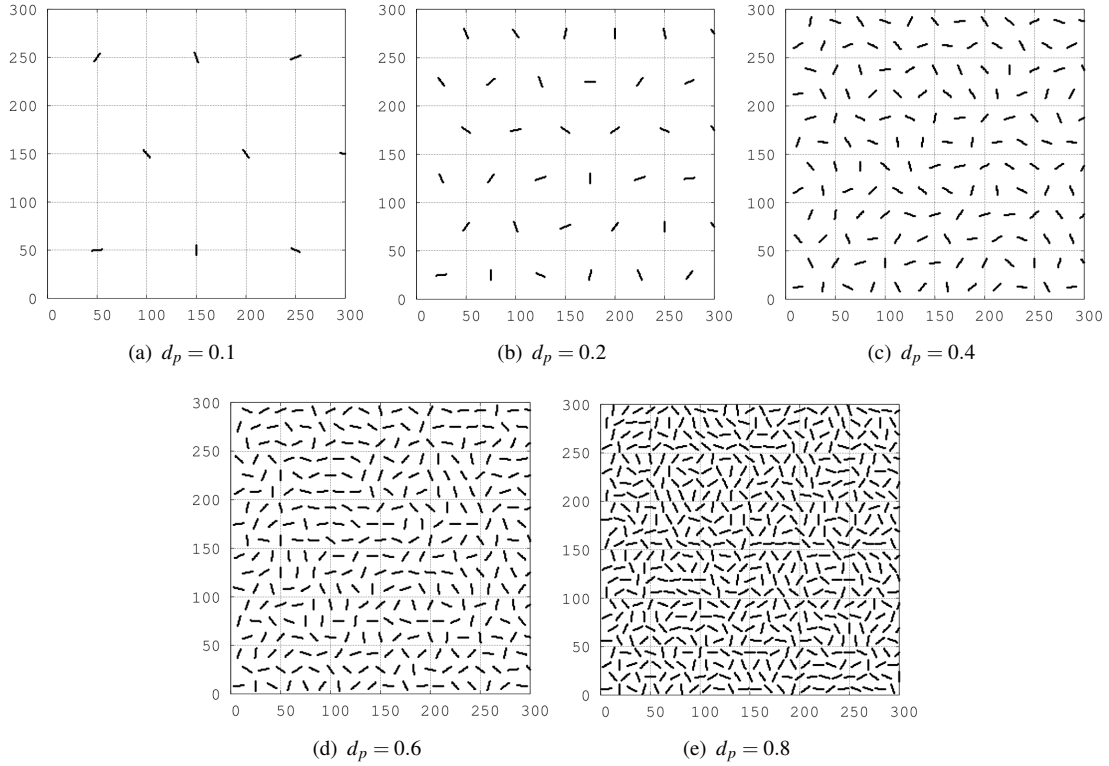


Figure 6.2: The length of the reflection lines in both hexagon cells is 10 in arbitrary unit. With different pattern densities d_p, the hexagon cells show different sizes. Since the length of the reflective line does not change, the gap between the reflective line and the hexagon edge changes with d_p.

In this experiment, nine arenas were used and their pattern densities d_p are 0.1, 0.2, ..., 0.9 specifically. Part of the arenas with pattern density of 0.1, 0.2, 0.4, 0.6 and 0.8 are shown in Figure 6.3. The rest arenas with pattern densities of 0.3, 0.5, 0.7 and 0.9 are shown in Figure H.1.

There were two mechanism used in Section 5.2.1 in order to construct a large arena to fulfil the space required by the chromatography experiment. Specifically the mechanism of reusing several arenas in a row ensures the robot never move out of the testing environment from the right boundary of the arena. With the help of the top and bottom boundary re-entering mechanism, the robot never move out of the testing environment from either top or bottom boundary of the arena. These two mechanisms were also used in this experiment.



(a) $d_p = 0.1$ (b) $d_p = 0.2$ (c) $d_p = 0.4$

(d) $d_p = 0.6$ (e) $d_p = 0.8$

Figure 6.3: Arenas with Different Pattern Densities. Only an area of 300x300 of the arenas is shown to illustrate the different densities of the reflective lines.

Additionally, it is designed that chromatography experiment separates robots along the x axis of the arena. Hence the only x coordinate of the robot is used for experimental result analysis.

6.3 Design of Swarm Robots

There are multiple groups of robots in this experiment. Each group consists of 32 individuals which were generated using the method described in Section 5.3. Specifically individual parameter of all 32 robots were varied with a sequence of Gaussian distributed random numbers with specific sigma values. The hardware parameters of the robots in a group are identical to the ones in the rest groups.

The controllers of all robots in a group have the same integration length. Different groups have different lengths for the integral term as discussed in Section 6.1. The design of the robot's controller is described as follows.

6.3.1 Robot's Controller

The controller of the robots used in this experiment is the PI controller described in Section 3.2.2. A variable *ml*, which denotes the integration length was incorporated into the controller. The

controller output is defined by Equ 6.2.

$$V_{PI}(t) = K_p\,\delta(t) + K_i \sum_{t-ml}^{\tau=t} \delta(\tau) \tag{6.2}$$

where

$V_{PI}(t)$ is the output of the PI controller at time t.

K_p is the proportional coefficient of the PI controller,

$\delta(t)$ is the error for PI controller which refers to the difference between the outputs of the two IR sensors at time step t.

K_i is the integral coefficient of the PI controller,

ml is the number of errors in the past which needs to be integrated.

The integral term of this controller only accumulates ml number of $\delta(t)$ in the past.

While $ml = 300$ for the robots used in the experiments of Chapter 4 and 5, ml was set to 0, 50, 100, ..., 900 specifically for the groups of robots. In total, there were 19 groups of robots.

The controller parameter K_p and K_i were selected with the method described in Section 3.4.1.

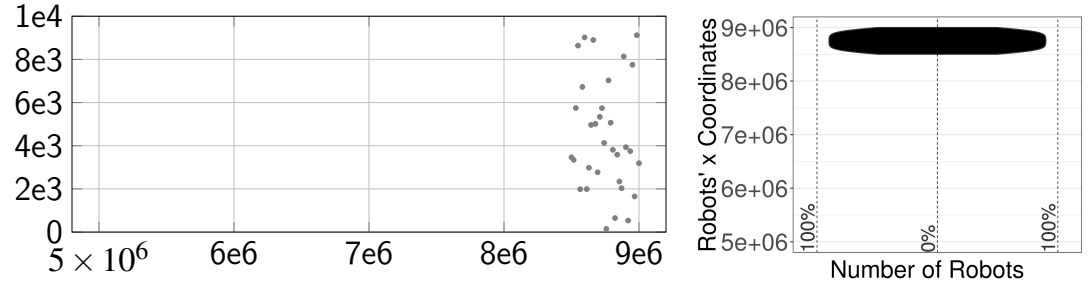
6.4 Results and Discussion

After the controller parameters for the robots were selected, they were tested in the chromatography experiment using the arenas described in Section 6.2. During this experiment, each group of 32 robots were simulated individually in each of the nine arenas which have different pattern densities d_p.

At the end of the simulation, the locations of 32 robots in a group were recorded. And the distribution of the robots' x coordinates in a group can be illustrated with a violin plot. The associated *R* script used to produce the violin plot based on the x coordinates of 32 robots is in Appendix I with an example of the 32 robots' x coordinates.

6.4.1 Violin Plot

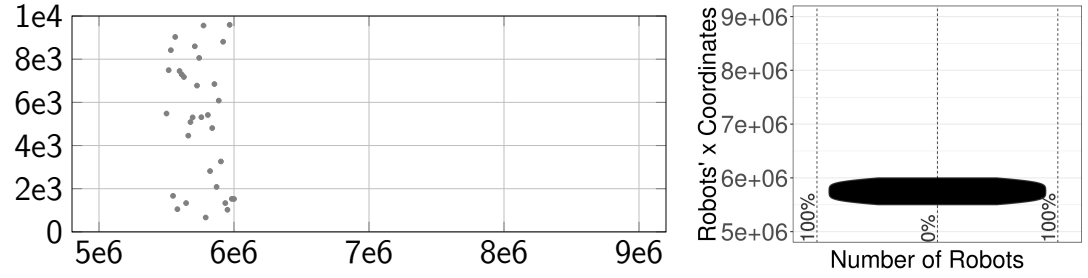
To understand how the violin plot illustrates the distribution of 32 robots' x coordinates from the chromatography experiment, four examples of 32 robots' x coordinates in different scenarios are provided, the robots' locations and the corresponding violin plots are showing in Figure 6.4, 6.5, 6.6 and 6.7.



(a) The black dots denote the location of 32 robots in the arena (in arbitrary unit). The robots' x coordinates are generated linearly between $5.5e6 \leq x \leq 6e6$. Therefore x coordinates of all robots are evenly scattered in the range $8.5e6 \leq x \leq 9e6$.

(b) The black area is the violin plot illustrating the distribution of the x coordinates of the robots. The y axis refers to the x coordinates of the robots. The x axis refers to the percentage of robots with specific x coordinates in all 32 robots.

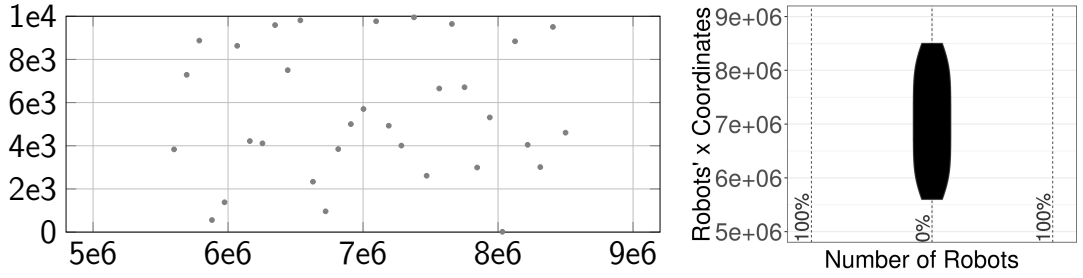
Figure 6.4: Example of Violin Plot and Robots' Locations: The left figure shows locations of 32 robots in the arena. The right figure shows the violin plot according to the distribution of the robots' x coordinates. As all 32 robots have similar x coordinates within a narrow range from the chromatography experiment, the violin shape has only one peak. Along the y axis, the violin shape covers the range of x coordinates of all robots.



(a) The black dots denote the location of 32 robots in the arena (in arbitrary unit). The robots' x coordinates are generated linearly between $5.5e6 \leq x \leq 6e6$. Therefore the x coordinates of all robots are evenly scattered in the range $5.5e6 \leq x \leq 6e6$.

(b) Violin plot for the distribution of the robots' x coordinates

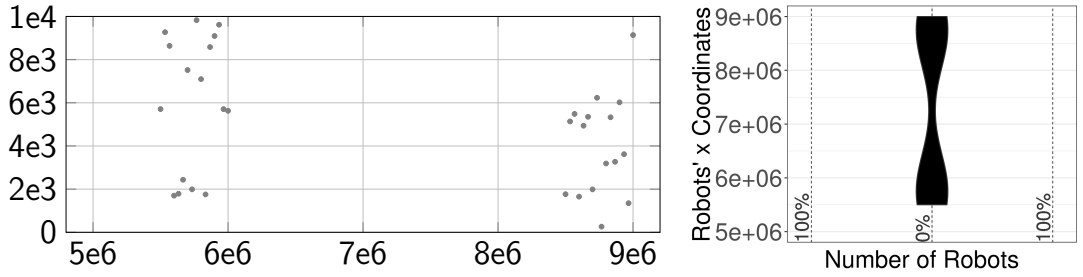
Figure 6.5: Example of Violin Plot and Robots' Locations: The left figure shows locations of 32 robots in the arena. The right figures shows the violin shape according to the distribution of the robots' x coordinates. The locations of all 32 robots in the arena are within the range $5.5e6 \leq x \leq 6e6$, hence the violin shape is covering the corresponding range along the y axis. Only the location of the violin shape is altered, the violin shape remains the same comparing with the example showing in Figure 6.4.



(a) The x coordinates of the 32 robots in the arena (in arbitrary unit) are evenly scattered in the range $5.5e6 \leq x \leq 8.5e6$.

(b) Violin plot for the distribution of the robots' x coordinates

Figure 6.6: Example of Violin Plot and Robots' Locations: The left figure shows locations of 32 robots in the arena. The right figures shows the violin shape according to the distribution of the robots' x coordinates. Due the uniform distribution of robots on the x axis of the arena, the violin shape has only one peak. Since the robots are located within a large range in the arena, along the y axis the shape covers a large range. Comparing with previous examples in Figure 6.4 and 6.5, the same number of robots are sparsely located in a much large range, therefore the violin shape is narrow.



(a) The locations of 32 robots in the arena (in arbitrary unit) are equally separated into two groups. The first 16 robots is located within the range of x coordinates $5.5e6 \leq x \leq 6e6$, the other 16 robots are located within the range $8.5e6 \leq x \leq 9e6$

(b) Violin plot for the distribution of the robots' x coordinates

Figure 6.7: Example of Violin Plot and Robots' Locations: The left figure shows locations of 32 robots in the arena. The right figures shows the violin shape according to the distribution of the robots' x coordinates. Since the 32 robots are located in two different areas in the arena, the violin shape has two peaks.

The violin plot is always symmetric on the left and right side. The y axis of the violin plot describes the x coordinates of the robots' locations in the chromatography experiment. If the violin plot at certain point of the y axis is wide, the number of robots is large with specific x coordinates from the chromatography experiment and vice versa. The peak of the violin plot shows the x coordinate of the chromatography arena where a majority of robots can be found.

The simulation results for the separation of the swarm robots are shown in Figure 6.8.

In general, the instantaneous movement of the robot is influenced by both the PI controller output and the simulated pressure. If the robot is on the reflective line, the robot will try to follow the line to the direction which the line points to. However the simulated pressure has an impact

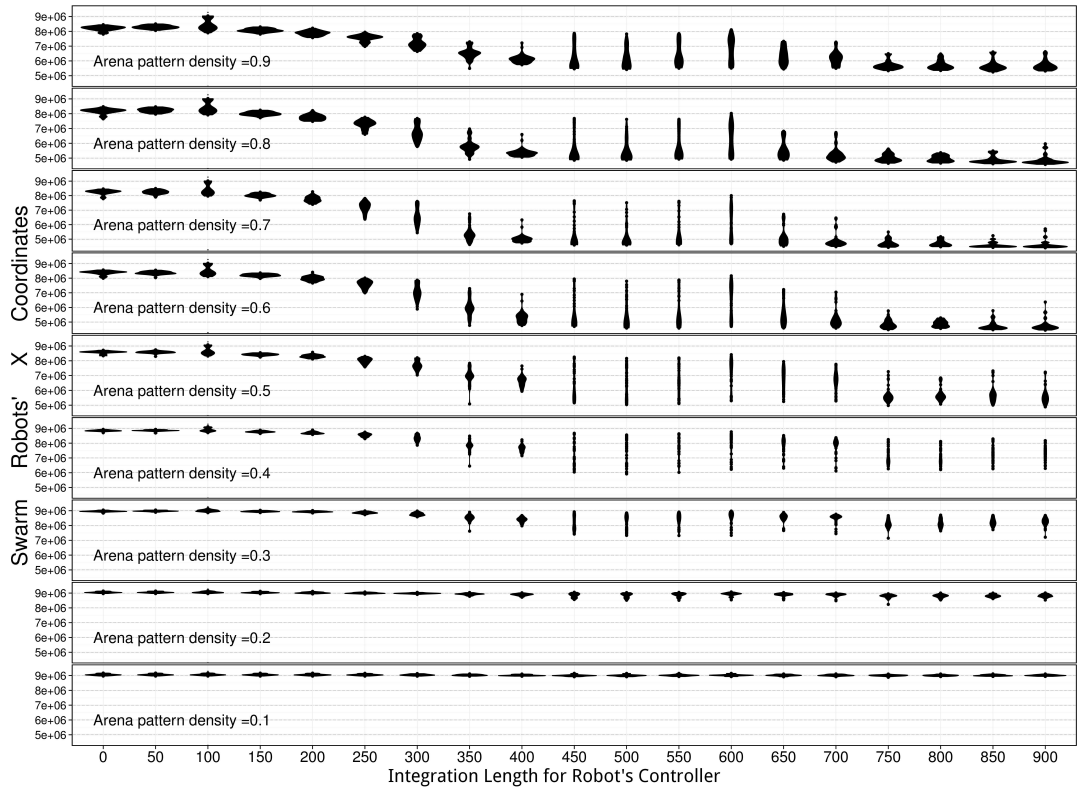


Figure 6.8: Violin plots illustrating the results of nineteen groups of robots with various integration lengths *ml* (along the x axis) for the PI controller performed chromatography experiments in 9 arenas with different pattern densities d_p (along the x axis). This figure was constructed by vertically concatenating 9 figures which shows the results of all groups in each arena: the figure on the bottom shows the results of all groups in the arena with pattern density $d_p = 0.1$. In each figure, there are nineteen groups of robots. The 32 robots in each group have the same integration length *ml*. The *ml* starts from 0 and increases by 50 steps per group until 900 steps. The y axis in each of the 9 figures have the same range, showing the x coordinates of robots upon completion of the experiment. The separation result for each group in each arena is illustrated with a violin plot. For each experiment, the width of the violin plot shows the number of robots converging at certain x coordinate. For instance, in the experiment $ml = 0$ and $d_p = 0.1$, the violin plot is wide and its centre is located at $y = 9e^6$, this means that the controller of the 32 robots do not have an integral term, the robots are located in the arena with their x coordinates around $9e^6$, thus they do not separate. However when $ml = 450$ and $d_p = 0.5$, the violin plot is narrow and long, it means that the 32 robots with 450 integration length separate almost evenly on the x coordinates of the $d_p = 0.5$ chromatography arena with the range of $(5e^6, 8.5e^6)$.

on the orientation and x coordinates of the robot. Depending on the current orientation of the robot, the additional change caused by the simulated pressure to the robot's x coordinates and orientation can be either small or large (illustrated in Figure 5.10). If the direction of the line is not largely different from the right direction of the arena, the impact caused of the simulated pressure can be counteracted by the robot, the robot would follow the line. However if the difference between the direction of the line and the right direction of the arena is large, the robot might be shifted away from the line before reaching the end of the line.

When the robot is located in the gap between the reflective lines and its IR sensors do not perceive any reflective line at all, due to the simulated pressure the robot would be orientated to the right direction of the arena. However according to Equ 6.2 the integral term would still have an impact over the controller output and alter the orientation of the robot to the direction other than the right as long as the integral term is not zero. The longer the integration length is, the longer the period is that the robot does not orientate to the right direction.

6.4.2 Robots' Separation and Controller Integration Length

The robots with zero integration length (first column of Figure 6.8) did not separated in any of the arenas. During simulation, when the robots were on the reflective lines, the swarm robots would try to follow the lines. The impact caused by the simulated pressure would be counteracted by the robot itself subject to the behaviour of the robot. When the robot were in the gap between the reflective lines, the robot would quickly change its orientation to the right without any counteraction as the integral term was zero and the controller output was also zero. When the arena density d_p were increased, the robot frequently encountered reflective lines, due to individual robot's unique hardware circumstance and controller parameters, the robot counteracted the impact of the simulated pressure differently, some robots appeared to be left behind the majority in the group. Hence when $ml = 0$ and $d_p \geq 0.5$, some robots appeared at the end of the separation.

As the integration length ml was increased, apart from the counteraction while the robot was on the lines, the stored difference from the two IR sensors' output in the integral term also made the robot counteract with the simulated pressure when the robot was not following any lines. As swarm robots in a groups have different hardware and software configurations, the swarm robots counteract against the simulated pressure differently: some swarm robots are able to maintain their previous orientation for sometime while some swarm robots can be easily influenced by the simulated pressure and conform to orientate to the right direction. The speed of swarm robots moving towards the right is slightly faster when the swarm robot faces straightly right than when it orientates to any other direction, causing the swarm robots begin to separate.

However when ml of the swarm robots further increase to 750, swarm robots in each group begin to converge. This is because that the integration length ml is too long, swarm robots always counteract to the simulated pressure when they are off the line. Given a lengthy period of time, swarm robots in a group have equal chances of orientating to any direction, thus the speed of every swarm robot have no large difference on average which makes the separation of swarm robots in a group with large ml not as good as that of swarm robots with smaller ml values.

6.4.3 Robots' Separation and Arena Density

Along the y axis of Figure 6.8, the arena pattern density d_p increases. When the integration length of the robots' controller was small, the group of robots did not separate well. Every robot in the group were located at the far end of the arena at the conclusion of a run. This is because that when the pattern density of the arena d_p was small, the robots did not encounter the reflective line very often. Evidently the robots' behavioural difference while they were on the line can not be accumulated for a large quantity. Even though some groups of robots have a large integration length, the integral term can not counteract with the simulated pressure since the IR output difference was zero when the robots are in the gaps between reflective lines. Hence, if the pattern density d_p is small, no separation happens for all groups of robots.

As the pattern density d_p of the arena increases until 750 steps, more interactions happened. In the case, the integral term of the controller which stores its previous difference between the two IR sensors began to counteract with simulated pressure which orientates the swarm robots to the right. The longer integration length was, for the more time it kept the swarm robots orientating to the direction other than the right, during which swarm robot's right-forward speed was reduced.

However when the arena density further increase to $d_p > 0.6$, a tendency of convergence for the swarm robots' end locations can be seen, especially when $ml > 750$. This is because that as the density of the arena patterns became large and all swarm robots constantly encountered the line; the frequent interactions between swarm robots and arena patterns made swarm robots' orientations change constantly and there was limited time left for swarm robots to to counteract the orientating effect differently. Hence the group of robots did not separated well.

6.5 Conclusion

In this chapter, the influence of swarm robot's integration length and the arena pattern density to the behaviour sorting results was investigated in the context of swarm chromatography. Nineteen groups of hardware-varied swarm robots with different integration lengths perform chromatography experiment in each of the nine arenas with different pattern densities. Results show that both controller integration length and arena pattern density are the keys to successful separation of the robots.

The chromatography experiment separates robots according to their ability of counteracting with the simulated pressure. Robots' behavioural differences can be explored either when they are on the line or when they are off the line which requires long integration length for the robot's controller. If the robot is on the line, robot's unique hardware and software circumstance leads to different ability of counteracting the simulated pressure (on-line behavioural difference). When the robot is off the line, the integral term which stores sensor output differences prolongs the robot's behaviour in terms of counteracting the simulated pressure (off-line behavioural difference).

The density of the arena patterns determines the number of interactions between swarm robots and the environment. If the number is low, robots' counteracting the simulated pressure do not happen very frequently. And for most of the time all robots orientate to the right, cause no large difference in right-forward speed, resulting unsuccessful separation. If the interaction is too often, the reflective lines consistently change the swarm robots' orientation and the robots can not show their differences in terms of the ability to counteract with the simulated pressure, thus sufficient separation can not occur. Hence the accumulation of the on-line behavioural difference the related with the density of the arena patterns.

The integration length of the robot's controller determines the number of the previous output differences between the two IR sensors. Particularly large integration length helps the robot to show its off-line behavioural difference. With short integration length, robot's orientation can be easily influenced by the simulated pressure when the robot is off the line. And the on-line behavioural difference is not enough for full separation of the robot even with a large number of interactions.

Both integration length and the arena density, determining the number of interactions with the reflective, line are the keys for separating robots with chromatography approach. If the integration length and the arena density matches with each other, robots can be separated with smaller distance along the x axis of the arena and less simulation time.

Furthermore, the integral term of the controller stores the differences between two IR sensors' output and affecting the instantaneous response of the robot. This is very similar to the learning ability implemented for some robots which enables individuals to learn from its past experience. Hence the result indicates that the difference of robots caused by hardware variation can be further amplified through a controller with the high-level capabilities.

Chapter 7

Conclusions

The work reported in this thesis investigated the issue of hardware variation in the context of swarm robots, particularly the existence of hardware variation, its influence on robotic behaviours. In addition, a novel behavioural sorting technique which can select robots according their behaviours caused by hardware variations.

Firstly, the literature on swarm robotic literature was reviewed. It was found that a majority of the researchers ignored the existence of hardware variation and assumed that individuals in a swarm are the same in both hardware-based and simulation-based swarm robotics. However this is not true as in practice robots in a swarm are different in term of the hardware. Not to mention the damage and deterioration caused when robot are used, the hardware variations occur during the component manufacture and robot assembly process can not be avoided. Following this the sources of hardware variations along the life span of a typical swarm robot was analysed, it was discovered that during component manufacturing the values of various parameters of sensors, actuators, mechanical subsystems can be different due to poor quality control, limitations of current manufacture technology. In addition, when components are assembled to construct a functional robot, more variations will occur such as sensor positioning, soldering quality etc.

Given the fact that hardware variation commonly exists in reality, the hypothesis was proposed that minute difference caused by hardware variation can influence components' performance and robot's reaction. For instance, robot's sensory capability depends on not only the sensitivity characteristics (component level) but also the positioning, alignment (assembly level). Any differences in sensor electrical characteristics or installation circumstances will result in different sensory ability for the robot. It is the same case for actuators. In addition, it was argued that if the robot was considered as an information processing system, sensors and actuators function as the input and output of such system, any difference occurred on the sensors and actuators would be amplified by the controller and the environment.

To test the hypothesis, a series of experiments was conducted involving a conventional swarm robot with standard hardware parameter values as the control robot and a number of robots

whose hardware parameters are individually varied from the control one. After the PI controller parameters were set with the exhausted search method, all robots were required to perform a line-following task. Results showed although the difference of the hardware parameters between the varied and the control robot are small, robots took different paths during the line-following task. Further experiments proved that difference in robots' trajectories can still be seen when the magnitude of the hardware variation is as low as 0.0001%. Thus it was demonstrated that although hardware variation is small, robotic behaviours in that particular task are still influenced as different trajectories were taken. Although the uniform tuning method for selecting the controller parameters helped robots achieve homogeneous behaviours in terms of following the parameter selecting line with robots' diverse hardware difference being compensated by the obtained controller parameters, the hardware variation and resulting differences in controller settings are amplified in the interactions between robot and environment. Thus the behaviour of the identically tuned robots in the same environment are subject to divergence.

To this point, it has been demonstrated that the commonly existed hardware variation, albeit small, can influence the behaviours of swarm robots and should not be ignored in swarm robotic research. However the question of how hardware variation influence the behaviour of individual robot remains. To answer this, the swarm robot chromatography technique was proposed to sort robots according to their behaviours caused by hardware variations, from which the relationship between behaviours and hardware can be drawn. The method of differentiating the robots through the accumulated effect of numerous interactions with the environment is analogous to separating chemical mixtures by chromatography. With this technique, robots' different response styles against the simulated pressure due to the unique hardware circumstance can be accumulated with the help of a special arena designed to encourage the number of interactions. Thus robots can be sorted according to distinct behaviours triggered by individual hardware circumstances. By comparing the sorting results to the hardware variation of individual robot, it was found that the behaviours of individual robot is not determined by a single parameter but by the combinations of multiple hardware factors. Different combinations of hardware parameters can help robots achieve similar behaviours.

Although the swarm chromatography technique gives robust behavioural sorting results, it does require a large arena and lengthy simulation time. To improve this, the influence of the integration length of the robot's controller and the arena pattern density to the sorting efficiency was investigated. Nineteen groups of hardware-varied swarm robots with different lengths for controller integral term perform chromatography experiments in each of the nine arenas with different pattern densities. It was found that if the arena pattern density matches with the memory length, robots can be sorted efficiently according to their behaviours. With the help of the controller integral term which stores the output difference of the two IR sensors, robot's response determined by its behavioural characteristics is prolonged while counteracting the simulated pressure which drives the robot towards right. However if the arena pattern is too dense, robot constantly encounters a line and there will be too much distractions constantly changing its orientation without letting the robot show its unique capability of counteracting with the simulated

pressure. Therefore integration length of the swarm robots and the arena pattern density has to match with each other in the swarm chromatography experiments for better sorting results, resulting a coupling effect of the integration length and the arena density.

This work can be viewed as an initial investigation into the issue of unavoidable hardware differences between swarm robots. Given the research outcome and that real swarms will necessarily show hardware variations, it is therefore necessary to contemplate current swarm algorithms in the context of diverse robot populations. In addition, a new research field of swarm chromatography for sorting robotic behaviours to improve swarm efficiency is initiated.

7.1 Future Work

Practical Implementation of Swarm Chromatography

The effectiveness of swarm chromatography technique to select robots with favoured behaviours is proved with simulation in this work. While long the domain of simulations, swarms of hundreds of robots are now becoming also feasible in hardware. It is, therefore, necessary to move this technique from scientific simulation to practical implementation. Specific apparatus and mechanism need to be created to have the same effect of the sorting arena as well as the simulated pressure.

To tackle the challenge of having a sorting arena of significant length, a belt conveyor system can be used. The belt will be covered with the patterns designed in Chapter 5. Driven by the pulley, the belt can move towards one direction with constant speed. In addition, to mimic the effect of the simulated pressure, a sail can be put on top of each robot with an identical angle. A fan is positioned on one side of the arena and constantly blowing wind towards to the belt. With this setup, robots could be sorted. Although the implementation proposed is feasible, its effectiveness still need to be proved with experiments.

Selectively Behavioural Sorting

It is demonstrated that the proposed swarm robot chromatography technique is able to sort the robot according to robots' behaviours. Based on the sorting results, robots which are likely to wonder around the environment and the ones which are reluctant to move and stay within a small area can be identified. These capabilities or preferences are robot's intrinsic behavioural characteristics triggered by robots' unique hardware circumstances. This sorting result is quite useful. For instance, in the task demanding a quick and rough scan of the environment, robots which like wondering around the environment can be chosen for this particular task, and the efficiency can be improved.

However in some tasks, robots with other types of behavioural characteristics might be preferred. For instance, robots which are likely to cooperate with other members are preferred for the task of moving object cooperatively. As the current proposed sorting technique with the line-following task can only sort the robots along their moving preference, improvement can be made to the sorting technique so that more types of behavioural characteristics can be emphasised, possibly with new sorting tasks other than following the lines.

Appendix A

Typical Tasks for Swarm Robots

In the literature, it is clear that a considerable number of scenarios have been, and are being used by researchers. This section considers those that are most commonly used to explore the structure and capabilities of swarm robots.

A.1 Obstacle Avoidance

Avoiding obstacles is the basic abilities that creature should have. This scenario remains the most widely used testing scenarios for swarm robots. In swarm robotic research, obstacle avoidance is expanded into two scenarios:

- At individual level, any robot should pass or avoid obstacles successfully.
- At system level, the whole swarm should be able to pass obstacles with formation maintained or to be recovered.

For the first level, normally when a obstacle is encountered by a swarm robot, no matter how big the obstacle is, robot will have to find alternative way to pass because that it is less cost-effective to equip extra sensors to detect obstacle's dimension than have the robot move around and choose another path leading to the same target. However implementation do exist by using vision-based approach Ahmed et al. (2012), where a camera is used and a light-weight vision algorithm is also developed due to limited capabilities of swarm robot CPUs.

More Commonly, infra-red emitters and sensors are used to detect obstacles and selecting another way to pass. In such circumstance, the particle swarm optimization (Chyan and Ponnambalam, 2012) and virtual potential field (Song et al., 2009) approaches are often adopted, which can also work well for maintaining and recovering formation of numbers of robots functioning as a swarm.

A.2 Self-Deployment

In swarm robotic research, the scenario of self-deployment is about instructing robots to deploy themselves in unknown environment to achieve maximum coverage of the environment based on their limited perception without global communication.

This scenario is widely discussed in the telecommunication field, in which a group of robot are sent out to initialize a communication path between points. Robots have to rely on themselves to achieve this without help of global viewer.

Standard to evaluate swarm robots' performance are area coverage by the robots and total deployment time. In such applications, communication between neighbouring robots and localization should be maintained. Common solutions to this kind of problem are particle swarm optimization and virtual force method.

Howard et al. (2002b) adopted potential field approach to explore the self-deployment of swarm-like mobile sensor network. Coverage can be maximized by the means of the mechanism that each individual in the swarm can be repelled by other robots or obstacles.Ren and Tse (2012) studied that adopting minimal number of robots to obtain complete monitoring coverage over an arbitrary 3D terrain.

A.3 Foraging

Foraging is one of the common phenomena in nature, in which worker ants are sent out for food searching and bringing it back to net if food is found. In swarm robotic research, robots are sent out for attractors. This is one of the common phenomena in nature, which is widely used to test robots functioning as a swarm.

Typically foraging consists of three different sub-functions including finding, grabbing and homing. The following issues should be considered:

- Environment coverage and covering rate
- Object handling (Robots are not required to move objects in some cases.)
- Energy management
- Robotic homing

Campo and Dorigo (2007) introduced a decision algorithm for robots to decide when to search for object and which objects (different objects have different energy yield and consumption to be carried to nest.) to be retrieved in order to maximize the energy accumulated by the group. Kernbach et al. (2012) studied the energy-constraint scenario where only a limited number of

recharging docks are provided. A threshold model has been provided in order to maximize energy efficient.

A.4 Aggregation

In nature, fishes joints together to form into schools, which can be regarded as aggregation. Swarm robots aggregation is often referred to that a number of swarm robots randomly scattered in the environment aggregate at certain point, which is the opposite of robots' dispersion.

For swarm robot, simply making them go to the direction where density of other robots is the largest is not a good solution since they may initially separated into several small groups.

Common approaches include probabilistic controller, evolutionary algorithms. And it is noticeable that Jeanson et al. (2005) used probabilistic controller with parameters measured from real cockroach larvae and successful proved that aggregation can be achieved based on local information obtained from interaction.

A.5 Pattern Formation

Pattern formation can be defined as that robots functioning as a swarm form certain patterns using decentralized control. This is one of the phenomena widely adopted in nature, like fishes and wild geese which also feature a decentralized control.

Generally, there are two methods to achieve this:

- Predefined motion strategies
- Virtual force strategies (also known as distributed potential field)

Both of the approaches require that each robot should be capable of sensing the distances and directions of nearby robots.

The former approach is implemented as that motion strategies of the robots are predefined and fixed. Each robot's moving direction and moving distance is well-defined and their motion behaviour should be the same under different circumstances. Furthermore, usually each robot can only move according to one specified robot. Thus formation can fail if one robot is surrounded by multiple robots unless each of them is assigned an unique identity code. However this will reduce flexibility and scalability.

In Fredslund and Mataric (2002b,a), the authors developed an algorithm for robot formation using local sensing and minimal communication, which requires that each robot has a unique identity code and "friend" sensor. When pattern formation is ordered, robot turns only to the

robot with lowest ID when more than one robot exist. The friend sensor is used to track the distance and angle between itself and friend robots to form a specific formation. Another example can also be found in Rubenstein et al. (2012).

The latter approach is implemented by adopting virtual forces into the motion control model. A repulsion force is generated if the distance between any two robots is less that a specified range. Both two robots will try to move away from each other. An attraction force is generated if the distance between any two robots are larger that a specified range. Both two robots will try to move near to each other. Bearing these two forces, formation of the robot will maintain stable dynamically.

This method is very flexible and functions well when the number of robots in the swarm goes large. However it is difficult to form specified shape when viewed globally. Implementations of this method can be found in Hashimoto et al. (2008).

A.6 Self-assembly

Self-assembly (self-reconfigurable robots) can be defined as creating more complex structures from large numbers of relatively simple units only with local interactions. Comparing with fix-structure robots, self-assembly robots have the advantages in versatility, robustness, adaptability, scale extensibility and even self-repair.

Not all robots are tested in this scenario because robots in this case are required to be equipped with one or several specially designed joints which are capable of attaching and detaching from other robots.

According to geometric arrangement of the units, self-assembly robots system can be classified into the following three groups Mohan and Ponnambalam (2009):

- Lattice Architecture: Units are arranged and connected in regular three-dimensional pattern.
- Chain Architecture: Units are arranged and connected in chain or tree pattern, which can later be folded up to form any other structures with help of their articulations.
- Hybrid Architecture: This architecture takes advantages of the previous two architectures. Parts of the whole can either be in chain or lattice pattern, which enables the whole structure more complex.

Other than the hardware part, there are two points in term of software which needs to be addressed here while designing self-assembly swarm robots.

- Control of locomotion of multi-unit structure

- Control of self-configuration

Appendix B

Another Training Arena

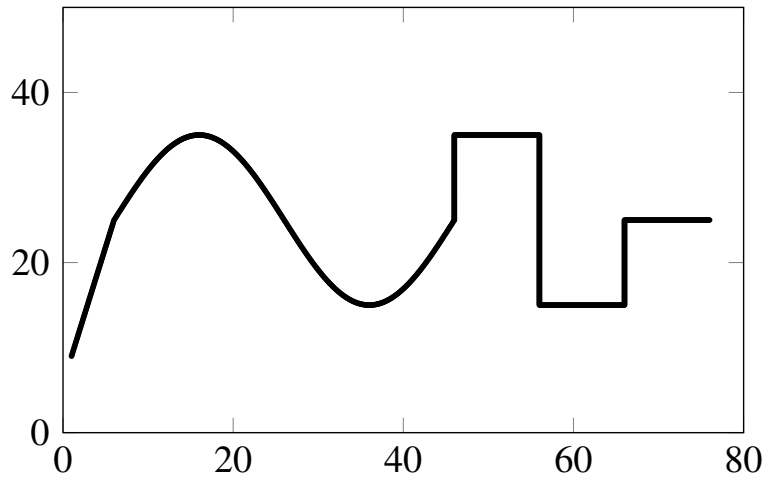


Figure B.1: Another Training Arena: This training arena was also tried to train the robots. The training results did not show much difference compared with the training arena showing in Figure 3.11.

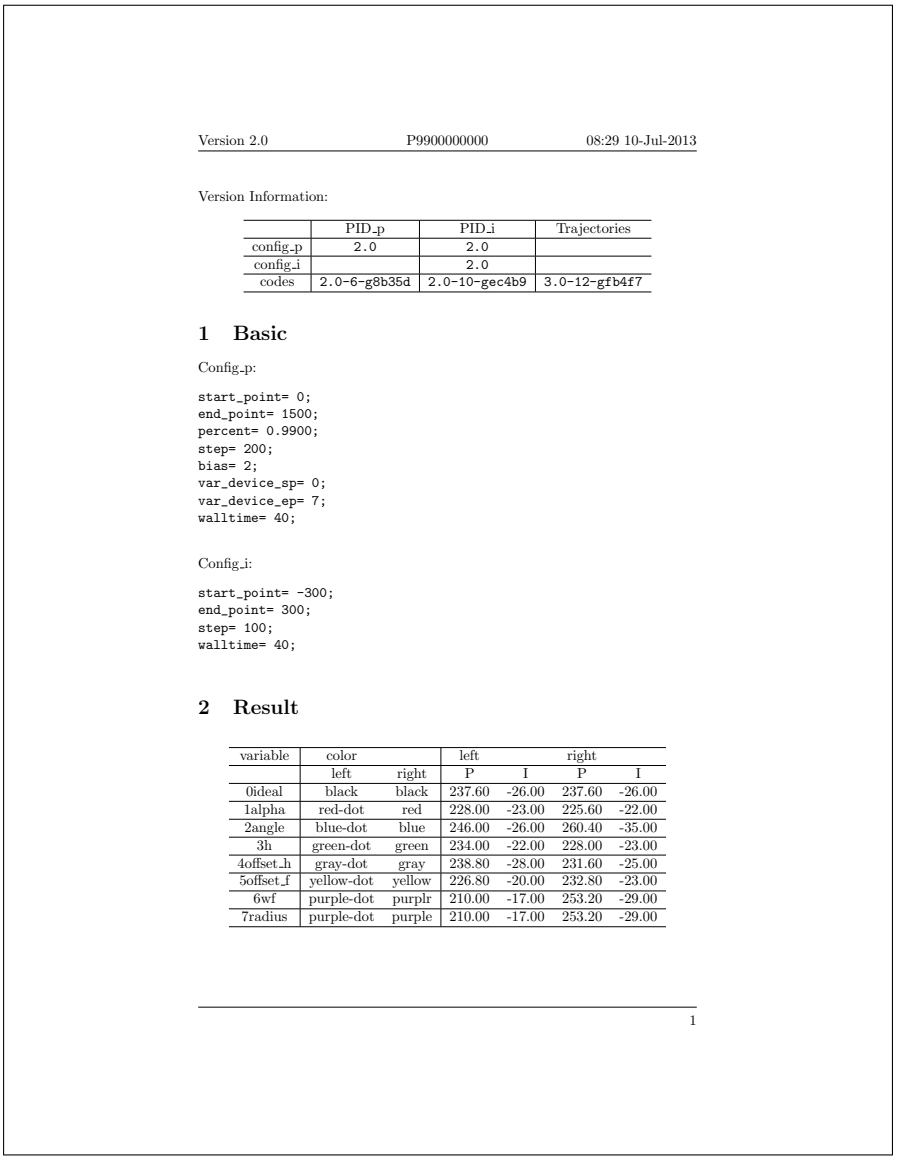
Appendix C

Simulation Jobs Submission Script

```
1  #!/bin/bash
2
3  #PBS -S /bin/bash
4  #PBS -N MX01
5  #PBS -l nodes=1:ppn=16
6  #PBS -l walltime=11:40:00
7
8  matlabroot="/local/software/matlab/2013a"
9
10 function test_MX01 {
11   pid=()
12
13   for (( i=0; i<=15; i++ )); do
14
15     profname="R$(printf '%06d' $(( $i + 1 )))"
16
17     ( ~/iridis_20150527_MixedMl_NoForward_3Ori_Rt0.6/mcc/
       run_simulator_robottraindraw_test.sh \
18     $matlabroot 20150527_MixedMl_NoForward_3Ori_Rt0.6 $profname iridis\
19     >~/iridis_20150527_MixedMl_NoForward_3Ori_Rt0.6/job_out/$profname.out ) &
20
21     pid[$i]=$!
22
23   done
24
25
26   for (( i=0; i<=15; i++ )); do
27     wait ${pid[$i]}
28   done
29 }
30 test_MX01
```

Appendix D

Simulation Report Example



Version 2.0 P9900000000 08:29 10-Jul-2013

Version Information:

	PID_p	PID_i	Trajectories
config_p	2.0	2.0	
config_i		2.0	
codes	2.0-6-g8b35d	2.0-10-gec4b9	3.0-12-gfb4f7

1 Basic

Config_p:

```
start_point= 0;
end_point= 1500;
percent= 0.9900;
step= 200;
bias= 2;
var_device_sp= 0;
var_device_ep= 7;
walltime= 40;
```

Config_i:

```
start_point= -300;
end_point= 300;
step= 100;
walltime= 40;
```

2 Result

variable	color		left		right	
	left	right	P	I	P	I
0ideal	black	black	237.60	-26.00	237.60	-26.00
1alpha	red-dot	red	228.00	-23.00	225.60	-22.00
2angle	blue-dot	blue	246.00	-26.00	260.40	-35.00
3h	green-dot	green	234.00	-22.00	228.00	-23.00
4offset_h	gray-dot	gray	238.80	-28.00	231.60	-25.00
5offset_f	yellow-dot	yellow	226.80	-20.00	232.80	-23.00
6wf	purple-dot	purplr	210.00	-17.00	253.20	-29.00
7radius	purple-dot	purple	210.00	-17.00	253.20	-29.00

1

Figure D.1: An Example of Automatically Generated Simulation Report Page 1

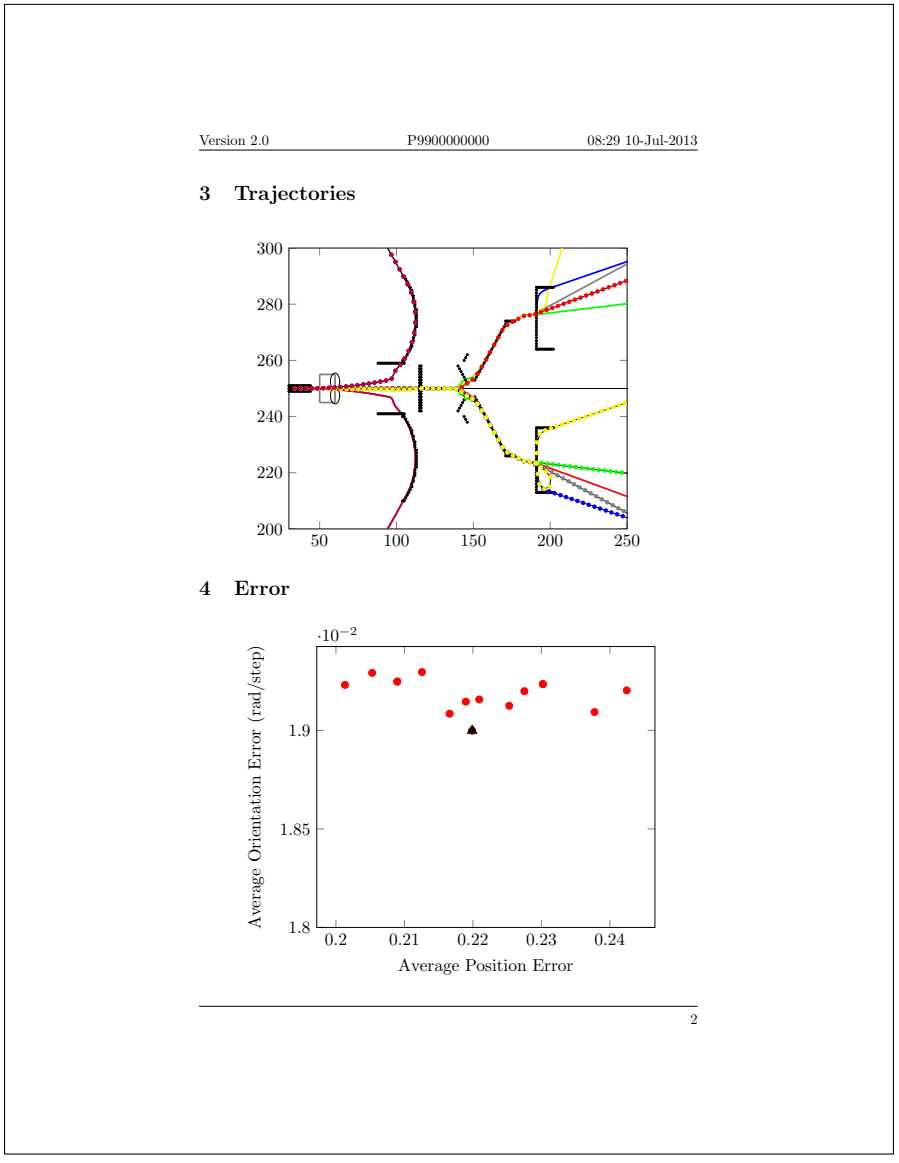
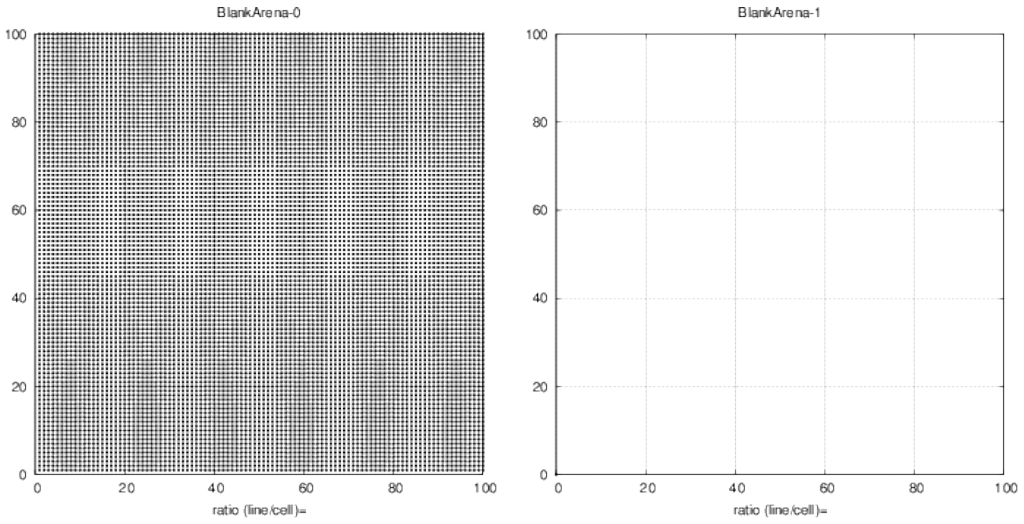


Figure D.2: An Example of Automatically Generated Simulation Report Page 2

Appendix E

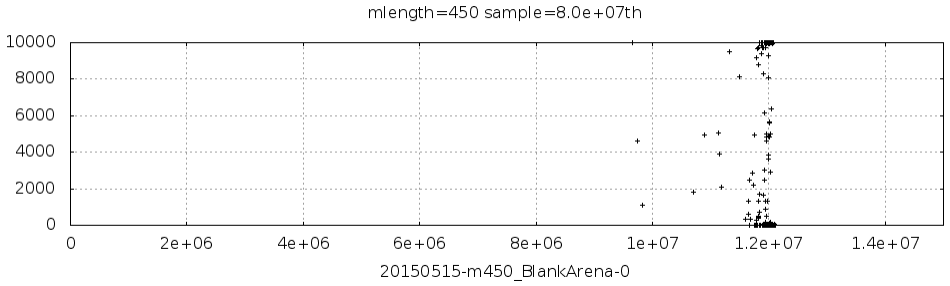
Swarm Chromatography Experiment with Empty Arenas

In this experiment, the same group of robots in Chapter 5 are simulated in two specific arenas. One arena is completely covered with reflective materials. The other one is completely blank. In other words, the arena does not have any reflective material at all.

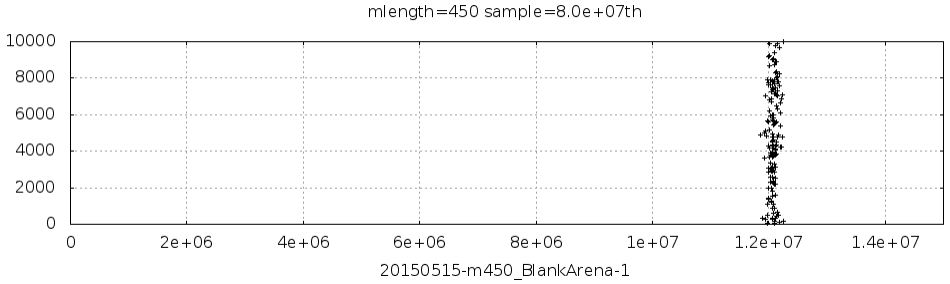


(a) The Arena Fully Covered with Reflective Materials (b) The Arena Covered with No Reflective Material at all

Figure E.1: The Arenas with and without Reflective Materials



(a) Robots' Location in the Arena Fully Covered with Reflective Materials



(b) Robots' Location in the Arena Covered with No Reflective Material at all

Figure E.2: Robots' Location in the Arenas with and without Reflective Materials: The results showed that successful behavioural sorting can not be done without the sorting arena with correct configuration specified in Chapter 5.

Appendix F

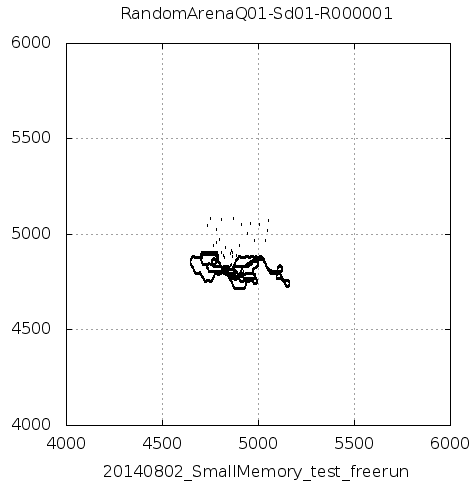
Swarm Chromatography Experiment without Simulated Pressure

The chromatography experiment in Chapter 5 was conducted without simulated pressure. The arena used in this experiment is the same with the chromatography experiment, showing in Figure 5.5. The same group of robots used in Chapter 5 were used in this experiment. All robots started at the coordinate (5000, 5000) in the arena with the same orientation to the right.

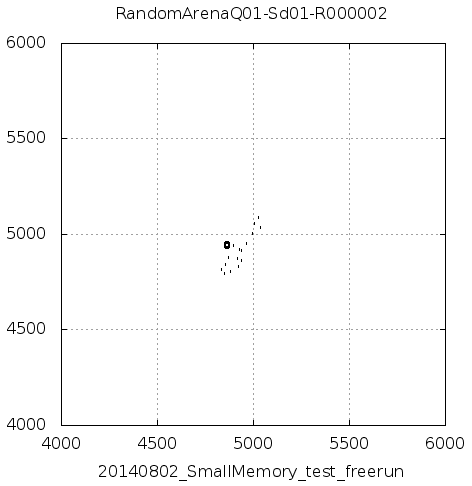
Only the first six robots' trajectories are listed in the following. Trajectories of the robots were sampled every 400 steps. Therefore the sparse points in the figures show that robots went through that part of the trajectory for a limited of times. The thick consecutive lines mean that robots went through that part of trajectory for a large number of times.

According to the figures, all robot set off from the starting point and left sparse dots in the arena, meaning that at the beginning robots run in the arena without repeating the trajectories. After the sparse point part, one can find a thick consecutive lines, which meaning that the robots repeatly run through that trajectories.

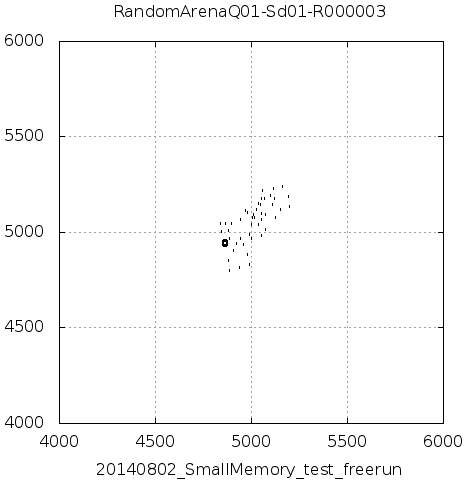
In other words, at the beginning, all robots was able to run freely in the arena. However after some time, robots constantly went through part of the trajectories over and over again.



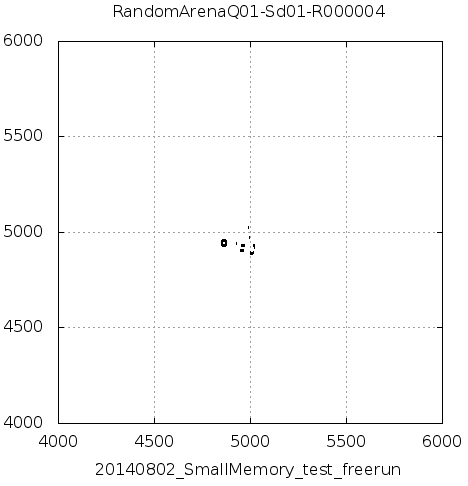
(a) Trajectory of Robot 01



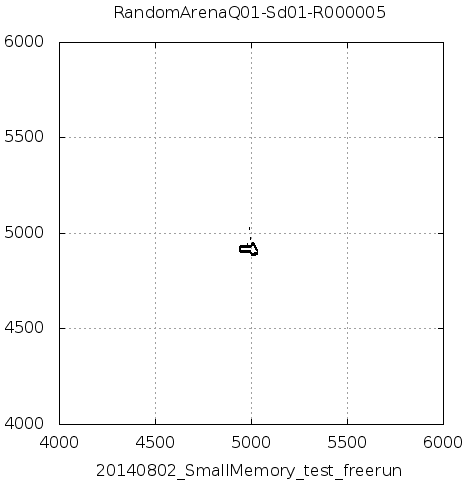
(b) Trajectory of Robot 02



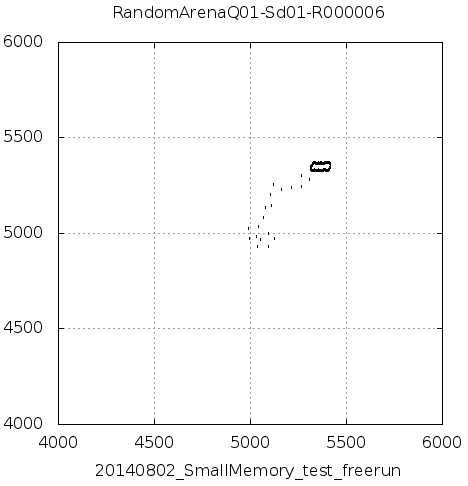
(c) Trajectory of Robot 03



(d) Trajectory of Robot 04



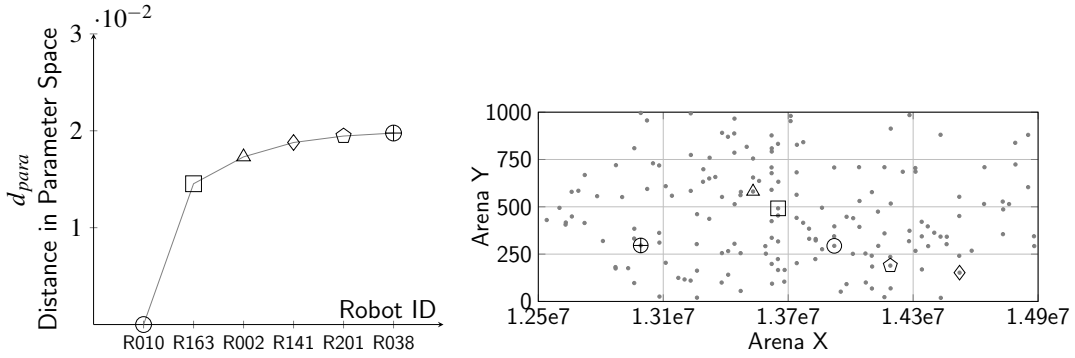
(e) Trajectory of Robot 05



(f) Trajectory of Robot 06

Appendix G

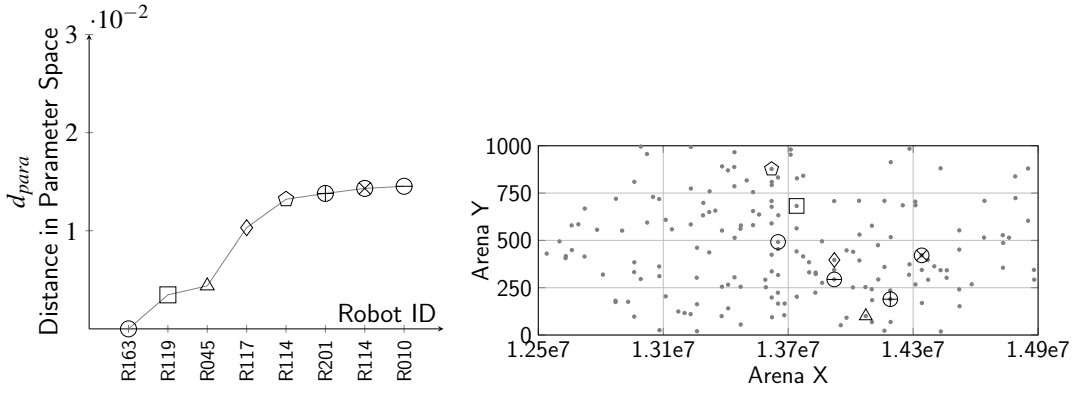
Clustering for R010 and R163



(a) Parameter distance between the robot R010 and similar ones

(b) Locations of robot R010 and similar ones

Figure G.1: Locations of robot R010 and similar ones and the parameter distance between them:



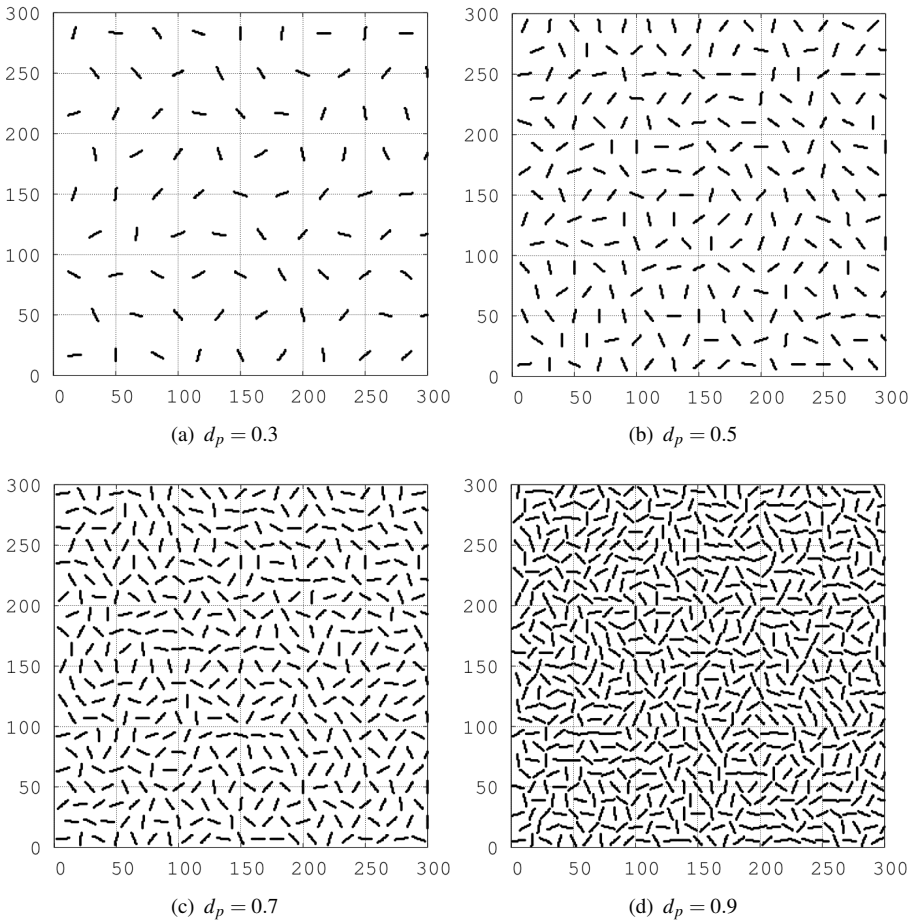
(a) Parameter distance between the robot R163 and similar ones

(b) Locations of robot R163 and similar ones

Figure G.2: Locations of robot R163 and similar ones and the parameter distance between them:

Appendix H

Chromatography Arena with Different Density



(a) $d_p = 0.3$ (b) $d_p = 0.5$

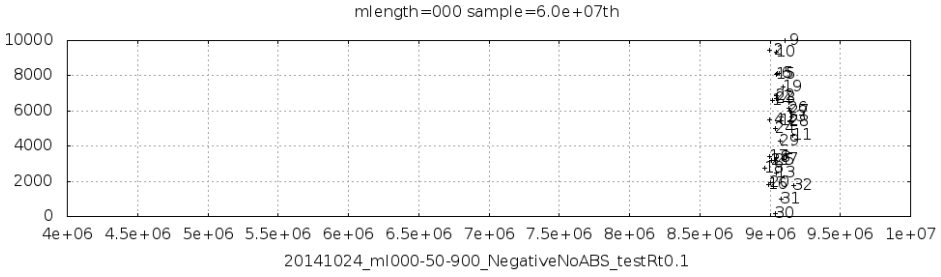
(c) $d_p = 0.7$ (d) $d_p = 0.9$

Figure H.1: Arenas with Different Pattern Densities. Only an area of 300x300 of the arenas is shown to illustrated the different densities of the reflective lines.

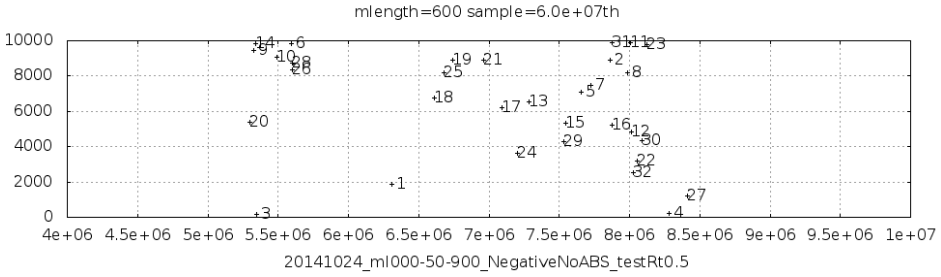
Appendix I

The Drawing of Violin-shape Figure

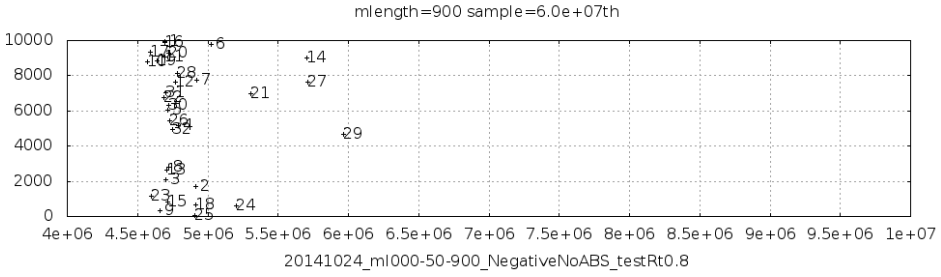
Each of the bell shape illustarted the distribution of the robots' location in the arena.



(a) Robots' memory length 0 and arena density 0.1



(b) Robots' memory length 600 and arena density 0.5



(c) Robots' memory length 600 and arena density 0.8

Figure I.1: Robots' Location with memory length of 0 and arena density=0.1

An example of the robots' location data are shown in the following. The data on each line is the x and y coordinates of the robot at specific simulation time step. In total, there are 32 robots.

```
1760511.5 2091.7
1615347 6117.4
1807051.3 2555.9
1892291.9 3887.4
1721292.3 3430.8
1663751.4 610.7
1648044.6 6592.8
1622264.5 7413.1
1592680.9 5349.9
1713105.8 2341.9
1675545.3 5505.1
1620274.4 8223
1664559.8 2695.1
1574774.1 593.3
1637949.3 785.5
1625473.1 1188.7
1575124.7 9808.5
1668672.2 6313.9
1579832.7 8195.6
1729993.4 7324.4
1631972.2 3533.6
1597286.1 6937.3
1682043.2 8874
1620722.6 5060.1
1666074.3 9904.3
1689767.7 1890.2
1945685.7 6261.6
1705738.7 6664.1
1594031.4 9936.8
1812102.5 905.5
1679487.8 9795.4
1620187 1505.7
```

R is used to generate the bell shape to illustarted the distribution of the robots' location. The R code is in the following.

```
library(ggplot2)

ratio_count=1

ep=1400000

r = seq(0.1, 0.9, by=0.1)
m = seq(000, 900, by=050)


df = data.frame(matrix(vector(), 0, 2, dimnames=list(c(), c("V1", "V2"))),
    stringsAsFactors=F)

for (rt in ratio_count:ratio_count){
  sessionname=sprintf("rt%.1f-ep%.0f",r[rt],ep);
  for (ml in 1:19){
    filename=sprintf("m%03.0f-ep%.0f.txt",m[ml],ep);
    file=sprintf("../../20141024_ml000-50-900_NegativeNoABS_testRt%.1f/temp/%s",r
    [rt],filename);
```

```
18      cat(sprintf("%s\n",file));
19      data <- read.table(file)
20      data <- data[,1:2]
21      data[,2] <-sprintf("%03.0f",m[ml]);
22      df<-rbind(df,data)
23
24    }
25    #data <- read.table('./f_RViolinplot.R.temp')
26    #data <- data[,1:2]
27    #data[,2] <-'ref';
28    #df<-rbind(df,data)
29
30    dev.new(width=20, height=2)
31
32    p<- qplot(factor(V2), V1, data = df,xlab='Memory Length',ylab='Arena Ratio')
33  # p<- qplot(factor(V2), V1, data = df, geom = "violin",ylim=c(4e6,10e6))
34    p<- p + geom_violin(scale='area',fill = "blue", colour = "black",trim=TRUE)
35    p<- p + annotate("text", y = 5e6, x = 1.5, label = sessionname)
36    print(p)
37
38    imgfilename=sprintf("%s.png",sessionname);
39    imgfile=sprintf("../figure/violinbasic/%s",imgfilename);
40    ggsave(filename=imgfile);
41    dev.off();
42  }
```

References

Ahmed, M., Saatchi, R., and Caparrelli, F. (2012). Vision based obstacle avoidance and odometery for swarms of small size robots. *Electronic Letters on Computer Vision and Image Analysis*, 11(1):54–67.

Alici, G. and Shirinzadeh, G. (2006). A Systematic Technique to Estimate Positioning Errors for Robot Accuracy Improvement Using Laser Interferometry Based Sensing. *Mechanism and Machine Theory*, 40:879–906.

Arena, P., Patanè, L., and Vitanza, A. (2012). Autonomous Learning of collaboration among robots. In *Proceedings of IEEE World Congress on Computational Intelligence*, Bribane, Australia.

Bahçeci, E. and Şahin, E. (2005). Evolving Aggregation Behaviors for Swarm Robotic Systems: A Systematic Case Study. In *Proceedings of the IEEE Swarm Intelligence Symposium*, Pasadena, California.

Balch, T. and Hybinette, M. (2000). Social Potentials for Scalable Multi-robot Formations. In *Proceedings of the 2000 IEEE International Conference on Robotics & Automation*.

Bayindir, L. and Şahin, E. (2007). A Review of Studies in Swarm Robotics. *Turkish Journal of Electrical Engineering*, 15(2):115–147.

Benet, G., Blanes, F., Simó, J., and Pérez, P. (2002). Using Infrared Sensor for Distance Measurement in Mobile Robots. *Robotics and Autonomous Systems*, 40:255–266.

Beshers, S. and Fewell, J. (2001). Models of Divsion of Labor in Social Insects. *Annual Review of Entomology*, 46:413–440.

Bonabeau, E., Dorigo, M., and Theraulaz, G. (1999). *Swarm Intelligence : from Natural to Artificial Systems*. New York: Oxford University Press.

Bongard, J. (2007). Reducing Collective Behavioural Complexity through Heterogeneity. In *Proceedings of the Seventh International Conference*, pages 327–336. MIT Press.

Borenstein, J. (1996). Measurement and Correction of Systematic Odometry Errors in Mobile Robots. *IEEE Transactions on Robotics and Automation*, 12:6.

Brooks, R. (1985). A Robust Layered Control System for a Mobile Robot. *IEEE Journal of Robotics and Automation*, 2(1):14–23.

Brutschy, A., Tran, N., Baiboun, N., Frison, M., Pini, G., Roli, A., Dorigo, M., and Birattari, M. (2011). Costs and Benefits of Behavioral Specialization. In *Proceedings of the 12th Annual conference on Towards autonomous robotic systems*, pages 90–101. Berlin: Springer-Verlag.

Campo, A. and Dorigo, M. (2007). Efficient Multi-foraging in Swarm Robotics. *Advances in Artificial Life*, 4649:696–705. Lecture Notes in Computer Science (LNCS).

Chyan, G. and Ponnambalam, S. (2012). Obstacle avoidance control of redundant robots using variants of particle swarm optimization. *Robotics and Computer-Integrated Manufacturing*, 28:147–153.

Conforth, M. and Meng, Y. (2010). Reinforcement Learning Using Swarm Intelligence-trained Neural Networks. *Journal of Experimental and Theoretical Artificial Intelligence*, 22(3):197–218.

Şahin, E. (2005). Swarm Robotics: From Sources of Inspiration to Domains of Application. In Şahin, E. and Spears, W., editors, *Swarm Robotics*, volume 3342 of *Lecture Notes in Computer Science*, pages 10–20. Springer Berlin/Heidelberg.

Deneubourg, J., Goss, S., and amd J. Pasteels, N. F. (1989). The Blind Leading the Blind: Modeling Chemically Mediated Army Ant Raid Patterns. *Journal of Insect Behavior*, 2(5):719–725.

Dong, M. (2009). *Modern HPLC for Practicing Scientists*. Wiley.

Dorigo, M., Floreano, D., Gambardella, L., Mondada, F., Nolfi, S., Baaboura, T., Birattari, M., Bonani, M., Brambilla, M., Brutschy, A., Burnier, D., Campo, A., Christensen, A., Decugnière, A., Caro, G., Ducatelle, F., Ferrante, E., Förster, A., Gonzales, J., Guzzi, J., Longchamp, V., Magnenat, S., Mathews, N., Oca, M., O'Grady, R., Pinciroli, C., Pini, G., Rétornaz, P., Roberts, J., Sperati, V., Stirling, T., Stranieri, A., Stützle, T., Trianni, V., Tuci, E., Turgut, A., , and Vaussard, F. (2013). Swarmanoid: a novel concept for the study of heterogeneous robotic swarms. In *Robotic & Automation Magazine, IEEE*, volume 20, pages 60–71.

Elliott, T. and Shadbolt, N. R. (2003). Developmental robotics: Manifesto and application. *Philosophical Transactions of the Royal Society of London, Series A*, 361:2187–2206.

Ettre, L. (1993). Nomenclature for Chromatography. *Pure and Applied Chemistry*, 65(4):819–872.

Fredslund, J. and Mataric, M. (2002a). A General Algorithm for Robot Formations Using Local Sensing and Minimal Communication. *IEEE Transactions on Robotics and Automation*, 18(5):837–846.

Fredslund, J. and Mataric, M. (2002b). Robots in Formation Using Only Local Sensing and Control. In *The 7th International Conference on Intelligent Autonomous Systems*, pages 25–27, Marina del Rey, California, USA.

Hamilton, W. (1963). The Genetical Evolution of Social Behaviour. *Journal of Theoretical Biology*, 7:1–52.

Harwood, L. and Moody, C. (1989). *Experimental Organic Chemistry: Principles and Practice*. Wiley-Blackwell.

Hashimoto, H., S., A., Yokota, S., Sasaki, A., Ohyama, Y., and Kobayashi, H. (2008). Cooperative Movement of Human and Swarm Robot Maintaining Stability of Swarm. In *Proceedings of the 17th IEEE International Symposium on Robot and Human Interactive Communication*, pages 249–254, Munich, Germany. IEEE.

Hauert, S., Winkler, L., Zufferey, J., and Floreano, D. (2008a). Ant-based Swarming with Positionless Micro Air Vehicles for Communication Relay. *Swarm Intelligence*, 2:167–188.

Hauert, S., Zufferey, J., and Floreano, D. (2008b). Evolved swarming without positioning information: an application in aerial communication relay. *Autonomous Robots*, 26(1):21–32.

Hayes, A. and Dormiani Tabatabaei, P. (2002). Self-Organized Flocking with Agent Failure: Off-Line Optimization and Demonstration with Real Robots. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 3900–3905, Washington DC, USA.

Hilder, J., Horsfield, A., Millard, A. G., and Timmis, J. (2016). The psi swarm: A low-cost robotics platform and its use in an education setting. *Towards Autonomous Robotic Systems*, 9726:158–164.

Howard, A., Matarič, M., and Sukhatme, G. (2002a). An Incremental Self-Deployment Algorithm for Mobile Sensor Networks. *Autonomous Robots*, 13(2):113–126.

Howard, A., Matarič, M., and Sukhatme, G. (2002b). Mobile Sensor Network Deployment using Potential Fields: A Distributed, Scalable Solution to the Area Coverage Problem. In Zhang, H., Olariu, S., Cao, J., and Johnson, D., editors, *Proceedings of the 6th International Symposium on Distributed Autonomous Robotics Systems*.

Hristoskova, A., Fortuny, E., and Turck, F. (2011). Subsumption Architecture for Enabling Strategic Coordination of Robot Swarms in a Gaming Scenario. In *Proceedings of the 2nd International Conference on Adaptive and Intelligent Systems*, volume 6943, pages 145–156.

Ijspeert, A., Martinoli, A., Billard, A., and Gambardella, L. (2001). Collaboration through the Exploitation of Local Interactions in Autonomous Collective Robotics: The Stick Pulling Experiment. *Autonomous Robots*, 11(2):149–171.

Jacobi, N. (1997). Evolutionary Robotics and the Radical Envelope-of-Noise Hypothesis. *Adaptive Behavior*, 6(2):325–368.

Jakobi, N. (1998). *Minimal Simulations For Evolutionary Robotics*. PhD thesis, University of Sussex.

Jakobi, N., Husband, P., and Harvey, I. (1995). Noise and the reality gap: The use of simulation in evolutionary robotics. In *Proceeding of 3rd European Conference on Artificial Life*, pages 704–720. Springer-Verlag.

Jeanson, R., Rivault, C., Deneubourg, J., Blancos, S., Fourniers, R., Jost, C., and Theraulaz, G. (2005). Self-Organized Aggregation in Cockroaches. *Animal Behaviour*, 69:169–180.

Kernbach, S., Nepomnyashchikh, V., Kancheva, T., and Kernbacha, O. (2012). Specialization and generalization of robot behaviour in swarm energy foraging. *Mathematical and Computer Modelling of Dynamical Systems: Methods, Tools and Applications in Engineering and Related Sciences*, 18(1):131–152.

Koestler, A. and Bräunl, T. (2004). Mobile Robot Simulation with Realistic Error Mode. In *2nd International Conference on Autonomous Robots and Agents*, Palmerston North, New Zealand.

Kuremoto, T. and M. Obayashi, K. K. (2009). Adaptive Swarm Behavior Acquisition by a Neuro-fuzzy System and Reinforcement Learning Algorithm. *International Journal of Intelligent Computing and Cybernetics*, 2(4):724–744.

Labella, T., Dorigo, M., and Deneubourg, J. (2004). Efficiency and Task Allocation in Prey Retrieval. In *Proceedings of the 1st International Workshop on Biologically Inspired Approaches to Advanced Information Technology, Lecture Notes in Computer Science*, 32-47. Springer Verlag, Heidelberg, Germany.

Lerman, K., Martinoli, A., and Galstyan, A. (2004). A Review of Probabilistic Macroscopic Models for Swarm Robotic Systems. In Sąhin, E. and Spears, W., editors, *Proceedings of the Swarm Robotics Workshop*, Los Angeles.

Li, L., Martinoli, A., and Abu-Mostafa, Y. (2002). Emergent specialization in swarm systems. In *Proceedings of Intelligent Data Engineering and Automated Learning*, pages 261–266.

Li, L., Martinoli, A., and Abu-Mostafa, Y. (2004). Learning and Measuring Specialization in Collaborative Swarm Systems. *Adaptive Behavior - Animals, Animats, Software Agents, Robots, Adaptive Systems*, 12(3-4):199–212.

Liu, W., Winfield, A., Sa, J., Chen, J., and Dou, L. (2007). Towards Energy Optimization: Emergent Task Allocation in a Swarm of Foraging Robots. *Adaptive Behavior*, 15:3.

Lobo, J. and Dias, J. (2007). Relative Pose Calibration Between Visual and Inertial Sensors. *International Journal of Robotics Research*, 26(6):561–575.

Lüscher, M. (1961). Air-Conditioned Termite Nests. *Scientific American*, 205(1):138–145.

Lyon, A. (2013). Why are normal distributions normal? *The British Journal for the Philosophy of Science*, 65(3):621–649.

Malheiros, P., Gonçalves, J., and Costa, P. (2009). Towards a more Accurate Infrared Distance Sensor Model. In *International Symposium on Computational Intelligence for Engineering Systems.*, Porto.

Martinoli, A. and Easton, K. (2002). Modeling Swarm Robotic Systems. In *Proceedings of the 8th International Symposium on Experimental Robotics*, pages 285–294, Sant Angelo dlschia, Italy. Springer Tracts in Advanced Robotics.

Martinoli, A., Easton, K., and Agassounon, W. (2004). Modeling Swarm Robotic Systems: A Case Study in Collaborative Distributed Manipulation. *International Journal of Robotics Research*, 23(4):415–436.

Matariˆc, M. (1997). Reinforcement Learning in the Multi-Robot Domain. *Autonomous Robots*, 4:73–83.

Miller, J. (2009). *Chromatography: Concepts and Contrasts*. Wiley-Blackwell.

Mohan, Y. and Ponnambalam, S. (2009). An Extensive Review of Research in Swarm Robotics. In *Proceedings of the World Congress on Nature Biologically Inspired Computing*, pages 140–145.

Mondada, F., Guignard, A., Bonani, M., Bär, D., Lauria, M., and Floreano, D. (2003). Swarm-bot: From Concept to Implementation. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, volume 2, pages 1626–1631.

Murciano, A. and Millán, J. (1996). Learning Signaling Behaviors and Specialization in Cooperative Agents. *Adaptive Behavior*, 5(1):5–28.

Murciano, A., Millián, J., and Zamora, J. (1997). Specialization in multi-agent systems through learning. *Biological Cybernetics*, 76(5):375–382.

Novischi, D. and Florea, A. (2016). Ant Intelligent Robot: A Versatile and Low Cost Miniature Mobile Robot Platform for Swarm Robotics Research and Education. In *The 9th EAI International Conference on Bio-inspired Information and Communications Technologies*, pages 256–263, New York, US.

Parker, L. (1994). *Heterogeneous multi-robot cooperation*. PhD thesis, MIT.

Patil, M., Abukhalil, T., Patel, S., and Sobh, T. (2016). UB Robot Swarm - Design, Implementation and Power Management . In *The 12th IEEE International Conference on Control and Automation*, pages 577–582, Kathmandu, Nepal.

Payton, D., Dally, M., Estkowski, R., Howard, M., and Lee, C. (2001). Pheromone robotics. *Autonomous Robots*, 11:319–324.

Pini, G., Brutschy, A., Frison, M., Roli, A., Dorigo, M., and Birattari, M. (2011). Task partitioning in swarms of robots: an adaptive method for strategy selection. *Swarm Intelligence*, 5(3-4):283–304.

Potter, M., Meeden, L., and Schultz, A. (2001). Heterogeneity in the coevolved behaviors of mobile robots: The emergence of specialists. In *Proceeding of 7th International Conferent Artificial Intelligency*.

Pugh, J. and Martinoli, A. (2007). Parallel Learning in Heterogeneous Multi-Robot Swarms. In *IEEE Congress on Evolutionary Computation*, pages 3839–3846, Singapore.

Ren, H. and Tse, Z. (2012). Investigation of Optimal Deployment Problem in Three-Dimensional Space Coverage for Swarm Robotic System. In *Social Robotics*, volume 7621 of *Lecture Notes in Computer Science*, pages 468–474. Springer Berlin Heidelberg.

Rettenmeyer, C. (1963). Behavioral studies of army ants. *The University of Kansas Science Bulletin*, 44(9):281–465.

Roth, Z., Mooring, B., and Bavani, B. (1987). An Overview of Robot Calibration. *IEEE Journal of Robotics and Automation*, 3(5):377–385.

Rubenstein, M., Cornejo, A., and Nagpal, R. (2014). Programmable Self-Assembly in a Thousand-Robot Swarm. *Science*, 345(6189):795–799.

Rubenstein, M., Hoff, N., and Nagpal, R. (2012). Kilobot : a Low Cost Scalable Robot System for Collective Behaviors. In *IEEE International Conference on Robotics and Automation*.

Snyder, L. and Dolan, J. (2010). *Introduction to Modern Liquid Chromatography, 3rd Edition*. John Wiley & Sons, New York, USA.

Sobol, I. and Levitan, Y. (1976). The production of points uniformly distributed in a multidimensional cube. Technical Report 16, Institute of Applied Mathematics, USSR Academy of Sciences.

Sobol, I. M. (1976). Uniformly distributed sequences with an additional uniform property. *U.S.S.R. Comput. Maths. Math. Phys.*

Song, P., Li, K., Han, X., and Qi, G. (2009). Formation and Obstacle-Avoidance Control for Mobile Swarm Robots Based on Artificial Potential Field. In *Proceedings of IEEE International Conference on Robotic and Biomimetrics*.

Soysal, O. and Şahin, E. (2005). Probabilistic Aggregation Strategies in Swarm Robotic Systems. In *Proceedings of the IEEE Swarm Intelligence Symposium*, Pasadena, California.

Spears, D., Thayer, D., and Zarzhitsky, D. (2009). Foundations of Swarm Robotic Chemical Plume Tracing from a Fluid Dynamics Perspective. *International Journal of Intelligent Computing and Cybernetics*, 2(4):745–785.

Spears, W., Spears, D., Hamann, J., and Heil, R. (2004). Distributed, Physics-Based Control of Swarms of Vehicles. *Autonomous Robots*, 17(2-3):137–162.

Toliyat, H., Levi, E., and Raina, M. (2003). A review of RFO Induction Motor Parameter Estimation Techniques. *IEEE Transactions on Energy Conversion*, 18(2):271–283.

Trianni, V. and Dorigo, M. (2005). Emergent Collective Decisions in a Swarm of Robots. In *Proceedings of the IEEE Swarm Intelligence Symposium (SIS 2005)*, number 8-10, pages 241–248.

Trianni, V., Groß, R., Labella, T., Sąhin, E., and Dorigo, M. (2003). Evolving Aggregation Behaviors in a Swarm of Robots. In *Proceedings of the 7th European Conference on Artificial Life, Lecture Notes in Artificial Intelligence*, volume 2801, pages 865–874.

Trianni, V., Labella, T., Grob, R., Şahin, E., Dorigo, M., and Deneubourg, J. (2002). Modeling Pattern Formation in a Swarm of Self-Assembling Robots. Technical report, Universit Libre de Bruxelles, Bruxelles, Belgium.

Trianni, V., Nolfi, S., and Dorigo, M. (2005). Cooperative Hole Avoidance in a Swarm-Bot. *Robotics and Autonomous Systems*, 54(2):97–103.

Turgut, A., Çelikkanat, H., Gökçe, F., and Şahin, E. (2008). Self-organized Flocking in Mobile Robot Swarms. *Swarm Intelligence*, 2(2-4):97–120.

Valdastri, P., Corradi, P., Menciassi, A., Schmickl, T., Crailsheim, K., Seyfried, J., and Dario, P. (2006). Micromanipulation, Communication and Swarm Intelligence Issues in A Swarm Microrobotic Platform. *Robotics and Autonomous Systems*, 54(10):789–804.

Ward, A. and Hart, P. (2003). The Effects of Kin and Familiarity on Interactions between Fish. *Fish and Fisheries*, 4(4):348–358.

Wolton, I. (2012). Iridis 3 Wiki: Job Submission and Scheduling on Iridis 3. https://cmg.soton.ac.uk/community/wiki/iridis/Job_Submission.

Zhang, D., Xie, G., Yu, J., and Wang, L. (2007). Adaptive task assignment for multiple mobile robots via swarm intelligence approach. *Robotics and Autonomous Systems*, 55(7):572–588.

Zhang, G., Fricke, G., and Garg, D. (2013). Spill Detection and Perimeter Surveillance via Distributed Swarming Agents. *IEEE/ASME Transacation on Mechatronics*, 18(1):121–129.

Zlatich, A. (2013). Photograpgic sequence of a chromatographic column. Technical report. urlhttps://commons.wikimedia.org/wiki/File:Cromatografia_su_colonna.jpg.