Devobot: From Biological Morphogenesis to Morphogenetic Swarm Robotics

A C D GAGET

PhD 2021

Devobot: From Biological Morphogenesis to Morphogenetic Swarm Robotics

Antoine Claude Dan Gaget

A thesis submitted in partial fulfilment of the requirements
of
Manchester Metropolitan University
for the degree of Doctor of Philosophy

Department of Computing and Mathematics
Manchester Metropolitan University

2021

Contents

List of Figures

List of Publications

Under preparation

- Gaget, A., Montanier, J.M. and Doursat, R. Fostering the Growth of Multi-Robotic Organisms. Submitted to *IEEE Transaction on Robotics*, under preparation.

Published

- Gaget, A., Montanier, J.M. and Doursat, R., 2020, October. Branched Structure Formation in a Decentralized Flock of Wheeled Robots. In *International Conference on Swarm Intelligence* (pp. 42-54). Springer, Cham.

Abstract

Complex systems are composed of a large number of relatively simple entities interacting with each other and their environment. From those entities and interactions emerge new and often unpredictable collective structures. Complex systems are widely present in nature, from cells and living organisms to human societies. A major biological process behind this emergence in natural complex systems is *morphogenesis*, which refers mainly, although not exclusively, to shape development in multicellular organisms. Inspired by morphogenesis, the field of Morphogenetic Engineering (ME) aims to design a system's global architecture and behaviour in a bottom-up fashion from the self-organisation of a myriad of small components. In particular, Morphogenetic Robotics (MR) strives to apply ME to Swarm Robotics in order to create robot collectives exhibiting morphogenetic properties. While most MR works focus on small and cheap hardware, such as Kilobots, only few or them investigate swarms of mobile and more "intelligent" robot models. In this thesis, we present two original works involving higher-end MR swarms based on the PsiSwarm platform, a two-wheeled saucer-size robot running the Mbed operating system. First, we describe a novel distributed algorithm capable of growing a densely packed "multi-robot organism" out of a group of 40 PsiSwarms, based on ME principles. Then, in another study closer to Modular Robotics (MoR), and taking inspiration from "programmable network growth", we demonstrate

the self-organisation of (virtual) branched structures among a flock of robots. Both works use MORSE, a realistic simulation tool, while a path toward crossing the "reality gap" is shown by preliminary experiments conducted using real hardware.

Declaration

No part of this project has been submitted in support of an application for any other degree or qualification at this or any other institute of learning. Apart from those parts of the project containing citations to the work of others, this project is my own unaided work. This work has been carried out in accordance with the Manchester Metropolitan University research ethics procedures, and has received ethical approval number *this student has not complied with ethics*.

Signed:

Date:

Acknowledgements

First of all, I would like to give my most sincere thanks to my Director of Studies, Prof. René Doursat. Throughout my thesis, René guided me through this ambitious project, and allowed me to achieve my full potential when I could not see it. Even though René left Manchester Metropolitan University one year before the end of my PhD to go back to Paris, he continued to remotely help me at the best of his capacity. I would like to believe that his perseverance and rigour were passed onto me, helping me to become a good researcher and a better person. Once again, René, you have my sincere gratitude.

I would also like to greatly thank Dr. Soufiene Djahel, my Director of Studies for the last year of my PhD. He helped me in the writing of this thesis and the article submitted to the *IEEE Transactions on Robotics* journal. Without him, I would not have been able to complete the redaction of this manuscript.

My sincere thanks also goes to the people who helped me complete my work: thank you to Dr. Darren Dancey (Manchester Metropolitan University), my co-supervisor, who helped me to get accustomed to the university administration and who always asked the right questions; thank you to Jean-Marc Montanier (Tinyclues), for his help in setting up the simulation and for supporting me during the first steps of my PhD work; and thank you to Dr. James Hilder (University of York) and Prof. Jon Timmis (University of

Sunderland), for the development of the PsiSwarm robot model—without it, this thesis would not have been possible—and for allowing me to modify and tailor the ARDebug software to my needs.

Throughout my thesis, the Department of Computing and Mathematics at Manchester Metropolitan University and the Centre for Advanced Computational Science (CfACS) provided funding, equipment (among others: PsiSwarm bots and a large office used as a robotic lab), and training for critical skills. I also received all the administrative support I needed, thanks to the dedication of the Research Degrees team at the Faculty of Science and Engineering.

My sincere thanks to my two scrutineers, Dr. Matteo Cavaliere and Dr. Soufiene Djahel, who reviewed my RD2 and gave excellent advise on how to continue and improve my research.

I would like to thank Dr. Matthew Shardlow who allowed me to teach in his module, Introduction to Programming, as a Teaching Assistant, and who helped me become a better teacher and with whom I greatly enjoyed working.

I would like to give a special appreciation you to my fellow colleagues in E132, Joshua P., Andrew C., Nicolas T. and Joel D., who supported me go through the thesis, by helping me solve problems I alone could not. We spent many hours talking, sharing, drinking coffee and beers, which gave me the much needed emotional support a thesis requires.

Along the way, I made a lot of new friends in Manchester, friends I hope to keep in the long run, so thank you to Matthieu, Sam, Kat, Juliette, Margot and all the others. A special thanks to Dominik, who helped me during the first few weeks in Manchester, even though he did not know me very well, your generosity is humbling.

I can not forget my friends who stayed in France, so thank you to Jeremy, Ismael,

Nolwenn, Marie and all the rest of you for your remote support throughout my PhD.

And finally, above all, thank you to my family. Thank you to Stefan and Maria, my brother and my sister, who helped me go through the difficult time that was 2020, and thank you my mum and dad, Magdalena and Michel, who pushed me to do this thesis and without whom I would not write these words.

Abbreviations

ACO	Ant Colony Optimisation
ANN	Artificial Neural Network
CB	Collective Behaviour
CMA-ES	Covariance Matrix Adaptation Evolution Strategy
CR	Collective Robotics
CS	Complex Systems
DevoBot	Developmental Robot
EC	Evolutionary Computing
EH-GRN	Evolving Hierarchical Gene Regulatory Network
GA	Genetic Algorithm
GRN	Gene Regulatory Network
InR	Intelligent Robots
MAS	Multi-Agent System
ME	Morphogenetic Engineering
MR	Morphogenetic Robotics
MoR	Modular Robotics

MU	Morse Unit
SR	Swarm Robotics
UAV	Unmanned Aerial Vehicles
USV	Unmanned Surface Vehicle
UUV	Unmanned Underwater Vehicle

Chapter 1

Introduction

Is designing and creating a multi-robotic system capable of self-organisation using bio-inspired principles achievable? This question is the main focus of the field of Morphogenetic Robotics (MR) (Jin and Meng 2010). MR is a fruitful alliance between Morphogenetic Engineering (ME) (René Doursat, Sayama, and Michel 2012) and Swarm Robotics (SR) (Beckers, Holland, and Deneubourg 1994). The latter is a technological family of complex systems targeting the use of multiple robots capable of performing tasks collectively and dynamically. Boosted by important advances in the field of robotics in recent years, especially cheaper and faster robot hardware, SR proposes large flocks of robots capable of many different feats such as, but not limited to, data collection over large areas; complete decentralisation to resist against local failures; easy replacement of robotic units; less costly communication with neighbours; and so on.

On the other hand, ME explores new methodologies to model and program the bottom-up self-assembly of a swarm of agents into specific functional architectures,

ones that cannot be directly planned top-down. ME takes its inspiration from morphogenesis and embryology, in particular how cells grow and arrange themselves into highly sophisticated structures. By using a set of complex biological mechanisms, such as cell duplication during mitosis, cell-to-cell communication based on chemical products released in the environment, and cell differentiation (the ability of a cell to change its structure and specialise), a single cell can become an oak, a housefly or a human being. The most impressive fact, and a central interest of ME, is that all decisions made by cells, e.g. when to differentiate or what signal to diffuse, are guided by chemical information stored in the DNA, which represents a massive string of elementary instructions decoded by each cell in various ways depending on its location, environment, or type. ME's endeavour is to design similar artificial "DNA's" embedded in each agent so that the swarm is able to achieve certain goals without explicit programming; in other words, it is to "meta-design" the motion control and collective self-assembly of individual agents to make them operate as a single entity.

To provide a brief context: MR can be separated into two main areas. In some systems, a great effort is spent on the design of sophisticated high-tech robotic parts and actuators capable of exact docking. In those cases, a small number of expensive units only permits sparse and precise formations, such as chains and T-junctions, typically by recursive attachment (*e.g.* M-TRAN from Murata et al. (2002)). Other systems, on the contrary, contain a large number of simple and cheap mobile robots, forming a dense mass (*e.g.* Kilobot from Rubenstein, Ahler, and Nagpal (2012)). These units "huddle" together to maintain local communication and form more or less random patterns, typically guided by "chemotaxis" (i.e. following virtual pheromones). In both cases, however, whether chain constructions or crowd flocking, the morphogenetic abilities of

these systems remain difficult to control.

In the present Developmental Robot, or DevoBot project, we aimed to create a multi-robotic system that exhibits morphogenetic abilities. Following the "meta-design" tenet of ME, and applying bio-inspired self-organisation to a large swarm of robots, we achieved the development of what can be called a "multi-robotic organism": an artificial creature composed of a body and several limbs, all made of a swarm of small two-wheeled robots. We managed to meta-design and implement a model capable of growing such a creature using 80 simulated robots within a realistic simulation environment. This meta-design allowed us to control the unfolding of the ME algorithm behind the scene as well as the final shape of the swarm. To further consolidate its viability, we ran a preliminary stability analysis on the simulation, then transposed it to a group of 26 real robots to form the body part of our creature. This was an attempt at crossing the *reality gap*, the fundamental discrepancy between what is feasible in simulation and in reality.

We have also undertaken a second study, where instead of growing a dense multi-robotic organism out of a swarm, we focused on chain formations within a sparser group of robots. Derived from ME principles as described earlier, this work followed a particular instance of it, "programmable network growth" (René Doursat and Ulieru 2008), better suited to smaller and less populated graph-like structures. We established such a model and implemented it both in simulation and in physical experiments, using the same simulation software and physical hardware as in the previous contribution.

In both studies, the idea behind our work is to create new "multi-robotic organisms" ultimately capable of performing tasks. If the swarm can be seen as a tool to achieve a

goal, our work on "meta-designing" the behaviour of the swarm comes from an indirect point of view: we want to show that it is possible to use a swarm to *generate* the needed tools to perform a specific task, then change the swarm shape to change the tool and perform yet another task, and so on. We want to increase flexibility in the swarm in order to increase the number of possible applications with one swarm. Within this ambitious framework, the present work only shows a proof-of-concept, i.e. first steps in this direction.

To summarise, during this thesis we tried to answer the following question: *Can we meta-design and implement a model for a multi-robotic system capable of self-organising into a "creature" or a structure using strong Morphogenetic principles?*

To answer this question, we organised this thesis as follows. First, we conducted an extensive literature review, presented in Chapter 2, describing the context of our contributions and discussing related works. Then, in Chapter 3, we explain our experimental setup, including common principles, tools and hardware used in both studies. We continue in Chapter 4 & Chapter 5 with the core of this thesis: the detailed description of the two studies. We first present the main contribution to the DevoBot project: the development of a multi-robotic creature in Section 4, before explaining our work on the self-assembling chain-like structures in Section 5. Finally, we conclude this thesis in Chapter 6, where we summarise our contributions to the field of Morphogenetic Robotics, expose the limitations and obstacles we encountered during the course of this work and discuss possible solutions and future work.

Chapter 2

State-of-the-art

This chapter outlines the key concepts used in our work, namely Collective Robotics and Morphogenetic Engineering and the concepts orbiting around them, and gives a broad outline of the existing work related to our project. Figure 2.1 illustrates where Morphogenetic Robotics (MR), the main research field of our work, is positioned relative to other fields. We start by reviewing Morphogenesis and Collective Behaviour (CB), introducing some core concepts on which our work is based. Then, we focus on Swarm Robotics (SR) before discussing Morphogenetic Robotics.

2.1 Introduction

Morphogenesis is the biological process referring to the development of the shape of an organism (J. B. Bard and J. Bard 1992). It is one of the two fundamental sides of the discipline of evolutionary developmental biology or "evo-devo" (Hall 2003). Evo-devo aims to understand the correlations between genotype and phenotype in living organisms, and how variations in one create variations in the other. Morphogenesis was

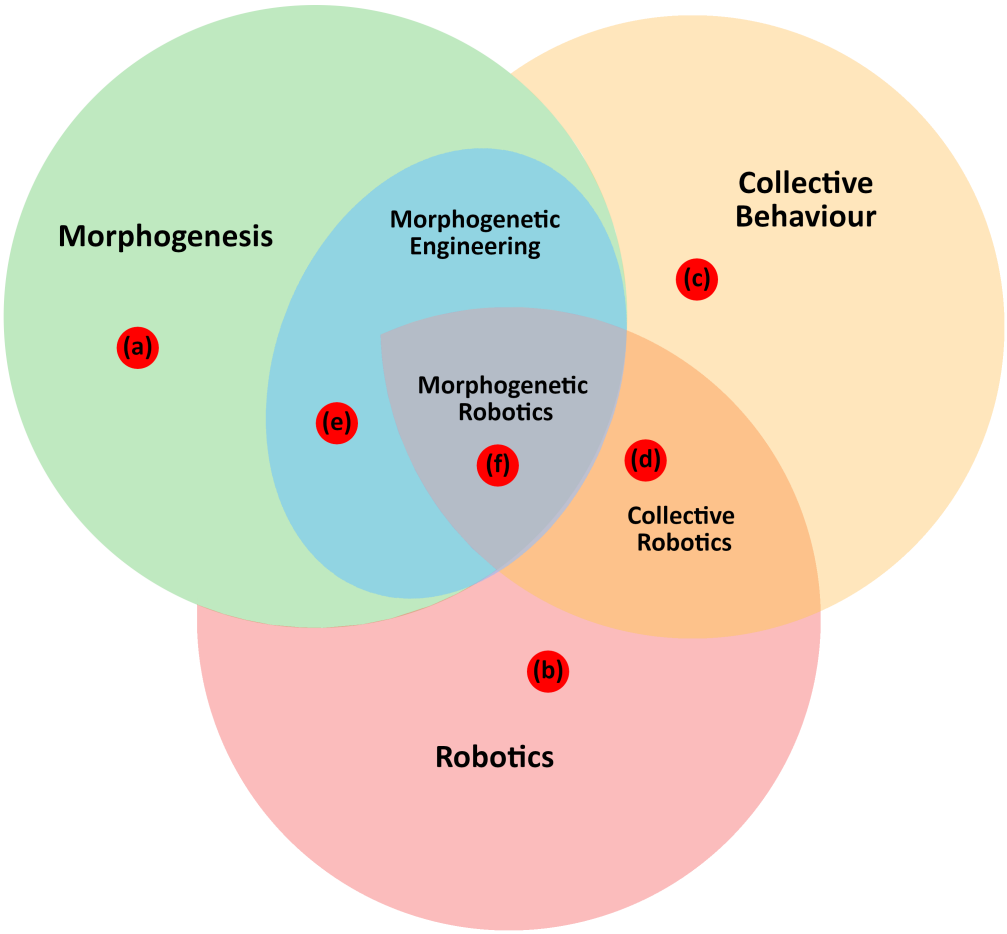

Figure 2.1: **Venn diagram of research fields related to our work.** Morphogenetic Robotics is a Collective Robotics system using principles of Mophogenetic Engineering. Example of research from: *(a)* Morphogenesis, the biological process referring to the development of the shape of an organism: Turing (1952), J. B. Bard and J. Bard (1992), and Ball (2015); *(b)* Robotics, the interdisciplinary study of machines: P. J. McKerrow and P. McKerrow (1991) and Siciliano et al. (2010); *(c)* Collective Behaviour, the study of emerging behaviour from groups of similar agents within Complex Systems: Reynolds (1987), Parrish, Viscido, and Grunbaum (2002), and Olfati-Saber (2006); *(d)* Collective Robotics, the study of large groups of relatively simple robots: Beckers, Holland, and Deneubourg (1994), Rubenstein, Ahler, and Nagpal (2012), and Murata et al. (2002); *(e)* Morphogenetic Engineering, the study of complex systems that self-organise and self-assemble in a non-trivial, controlled fashion: Rene Doursat (2011) and René Doursat, Sayama, and Michel (2012) and *(f)* Morphogenetic Robotics, the study of the application of ME principles to the field of collective robotics: Oh, Shiraz, and Jin (2018), Vergara et al. (2017), and Malley et al. (2020).

discussed by Alan Turing in his famous 1952 paper *The chemical basis of morphogenesis* (Turing 1952), in which he devised a mathematical model explaining how random fluctuations can drive the pattern formation from initial uniformity. His motivation was to answer the question of how a spherical embryo becomes a non-spherical organism, such as a human being (Ball 2015).

In synthetic biology, J. Pascalie (Pascalie et al. 2016) proposed a model of prokaryotic cell with morphogenetic properties that self-assemble into limb- and body-like structures. This is based on gradients of morphogen concentrations, a concept that we will also apply in our own robot experiments (see Chapter 4 & Chapter 5). Cell signalling is one of the pillars of morphogenesis. In Hancock (2003), J. Hancock presented three different ways for cells to communicate: (1) signal molecules diffusing in the surrounding environment of the cell; (2) direct chemical channels between a sender cell and a receiver cell through their membranes; and (3) proteins on the membrane that can be recognised and bound by another cell's membrane. These communication mechanisms have also inspired our model of information propagation through a complex robotic system (see Chapter 4.1).

Genetic Algorithms are part of Evolutionary Computing (EC) (Eiben and Schoenauer 2002; Eiben and Smith 2015). While not directly used in the work we present in this thesis, EC, and evolution in a more general sense, will be a crucial part of any future development based on this thesis (see Chapter 6).

In Dawkins (2003), R. Dawkins proposed a definition of evolution, and claimed that two principles are essential for it: (1) a *genetic* component, a set of replicators, or self-replicating entities on the micro level, also called genotype; and (2) an *embryology*, the

fact that replicators lead to an expression at a macro level on the entity that is evolving (this expression is also called a phenotype) and leading to the success or failure of the said entity within its environment. If the entity succeeds and survives with its set of replicators, its information is passed onto the next generation. To clarify those two terms: *genetic* is the study of the relationship between genotypes in successive generations, whereas *embryology* is the study of the relationship between genotypes and phenotypes within one generation. Most evolutionary computing techniques make use of a *fitness function* and *value*, a mathematical function used by the system to evaluate the quality of a specific individual within one generation. If we take the example of a GA used to optimise the parameters of a function, the fitness is used to select the best parameter sets, the individuals that yielded the best results, and pass them onto the next generation.

For a concrete example of the use of EC, C. Gros, in (Gros, Martin, and Sándor 2017), presented a one-neuron controller for a wheeled-robot, where each actuator, or wheel, is linked to a single neuron. The neuron mimics a steam locomotive propulsion system, using the angle of the wheel as the main input to calculate the force applied to the wheel in order to put it in motion. He implemented his model in a 3D simulation, and showed that the robot was able to achieve simple tasks efficiently with a number of wheels varying from 2 to 10 plus, proving the scalability of his model. K. Harrington devised an agent-based swarm behaviour for a food foraging task (Kyle Harrington et al. 2017). A GRN controls the behaviour of each agent, including obstacle avoidance and food items collection, and is evolved using a GA, where all agents of the swarm act as the population and the parameter sets of the GRN as the genome. Using a fitness function that punishes agents when they collide and rewards them when they successfully forage food items, K. Harrington created a “competitive eco-evolutionnary” simulation

where agents compete with each-other on the food collection task by avoiding collisions, and with the GA evolving the parameter set, improves the general behaviour of the swarm generation after generation.

It is also important to clarify what is "Collective behaviour" (CB) and how it is related to this thesis. CB denotes behaviours at the group level, emerging from the interactions of a large number of entities. These collective behaviours emerge thanks to a process called self-organisation. Self-organisation refers to a broad range of pattern-formation processes that make an initially disordered system ordered, such order arising from local interactions between the entities of the system (Camazine et al. 2020; Yates 2012).

Collective Behaviour can be linked to the study of flocks, what parameters control the behaviour of the entities within it, and the successive stages of flock or crowd formation.

Reynolds (Reynolds 1987) famously introduced three rules to model and simulate these behaviours: *separation*, where entities steer to avoid colliding and crowding surrounding flockmates; *alignment*, where entities correct their direction to align with surrounding flockmates, and *cohesion*, where entities move toward the centre of mass of their surrounding flockmates. He provided better insight into the key mechanisms of this type of natural complex systems.

In Toner and Tu (1998), J. Toner investigated the properties and different states a flock following Reynolds' rules have. To do so, they devised a mathematical model capable of predicting stable states in Reynold's flock, similar to the Navier-Stokes equations for simple comprehensible fluids. They showed that such flocks (in 2 dimensions) always exhibit a state where all the "boids" (elements in the flock) match their direction

and speed, forming a stable state. In his research, D. Fu (Fu et al. 2018) investigated the impact of low density of agents in flock formation, proposing a novel approach for flock behaviour to overcome the issues of low communications in low agent-density environment for flocking behaviour control. Indeed, he showed that flocks were forming more easily when the concentration of agents is high, whereas in a low agent-density environment, interactions are more rare hence flocks are forming more slowly if forming at all. His novel approach consists in a "follow-then-influence" behaviour, letting some agents influence the others in the flock to fly in a desired direction, hence controlling the global movement of the flock. The previously presented works are important because they exhibit a will to understand and control flocking behaviour, and form the first steps to fully decentralised and autonomous flocks of robots.

Beyond aggregates of social animals, the study of CB expands to larger horizons. Examples include emergence in vehicular traffic flow (Herman and Gardels 1963), a popular field of research nowadays, due to the expected safety and traffic efficiency applications that autonomous vehicles will enable (Jorge and Rossetti 2018). Studies like Nagel and Paczuski (1995), Kerner and Klenov (2009) or Zhu (2020) investigate the emerging properties of traffic. K. Nagel established a traffic flow model and studied the impact of small perturbations and variations on such model. B. Kerner found several states in traffic flow models, similar to states in Reynold's flock models, and showed the different transitions between those states. Finally, L. Zhu improved classic traffic flow models by adding the modelisation of semi-stable states. B. Friedrich showed that autonomous vehicles would improve traffic flow in general, with less traffic jams and an increase in mobility for low-mobility population groups (Friedrich 2016). Studies like Bhavsar et al. (2017) investigated the risks that autonomous vehicles may encounter and/or cause in traffic. P. Bhavsar identified several failure scenarios and proposed

and evaluated strategies to overcome such failures. In J. Wang, Peeta, and He (2019), J. Wang proposed a model for a traffic flow made of both human-driven vehicles and connected and autonomous vehicles.

Collective Behaviour is also studied in the context of human crowds, and is a research area that attracted a lot of attention due to its wide spectrum of potential applications (Thalmann 2007). For example, studies on crowd behaviour during a global panic helped researchers to better understand how the crowd behave and to improve safety measures (Helbing, Farkas, and Vicsek 2000; Shiwakoti and Sarvi 2013; Rockenbach et al. 2018). A Turing test was created to investigate if non-specialists could distinguish a simulated crowd from a real one, and J. Webster (Webster and Amos 2019) showed that they could, but participants were unable to find a real crowd when a choice was given.

Collective behaviour has also been used for other purposes than replicating or understanding behaviour. For example, Ant Colony Optimisation (ACO) (Dorigo and Gambardella 1997; Dorigo and Birattari 2011) uses the ants' collective food foraging behaviour to solve the travelling salesman problem and other combinatorial challenges (Stützle, Dorigo, et al. 1999; Peake et al. 2018; Peake et al. 2019).

In sum, Collective Behaviour appears in many research domains, in behavioural biology with the study of herds and flocks, in sociology with the study of crowd behaviour and in computer science with ACO. Naturally, this field of research also applies to robotics. Ant behaviour can serve as a basis for online area coverage algorithms using stygmergy, as shown in Giuggioli et al. (2016). In the next section, we will focus on the field of robotics. Since our project is focused on the study of swarm of robots, we will not look into research about single entity robots, but instead will focus on exploring

collective robotic systems, more precisely Swarm Robotics (SR).

2.2 Swarm Robotics

Swarm Robotics describes robotic systems composed of groups of small, usually cheap and un-specialised units capable of simple movement and actions. The goal of these systems is to study the emergence of collective behaviours in swarms of robots, and how to design these behaviours to achieve tasks. SR is different from single entity robotics. Single entity robotics aim is to develop a single complex robot capable of performing highly specific tasks, for example: assistive robotics which aim at creating robots capable of social interaction with human beings (Tapus, Mataric, and Scassellati 2007; Feil-Seifer and Mataric 2005); and research linked to space exploration (planetary and space robotics) with development of highly specific and expensive equipment (Yoshida 2009; Weisbin and Rodriguez 2000).

Since swarm robots form complex systems, the difficult challenge is to design rules for the control and coordination of multiple robots so that they can achieve given functions at the swarm level, such as forming a spatial shape (*e.g.* surrounding an area) or acting in coordination (*e.g.* pushing a large object). First, we clarify the distinction between "self-configuration" and "self-assembly" terminologies used in this thesis. The former refers to the ability of the swarm to arrange itself into a final state known in advance by the robots, *i.e.* they make decisions based on both their local perception and a priori knowledge of their final goal. The latter also describes the capacity to form configurations based on local interactions but without any prior knowledge of the goal to reach.

In robotics, using a large number of robots simultaneously is a real challenge, due

to the complex infrastructure needed for communication, experiments and maintenance, but also due to the often high cost of each individual unit. To overcome this challenge, Rubenstein created the Kilobots (Rubenstein, Ahler, and Nagpal 2012), a robot with a 33mm diameter. Its cheap cost allows researchers to use a large number of Kilobots to undertake large scale experiments, and its ease of operation and local communication capability make it a prime choice for MR research.

J. Alonso-Mora (Alonso-Mora, Breitenmoser, Rufli, Siegwart, et al. 2011) described a method of controlled pattern formation in a swarm of robots, using a centralised planification algorithm. The robots used were the "e-puck" model (created by F. Mondada in Mondada et al. (2009)), and were randomly placed in an arena in which they move to achieve a specified pattern, such as a line or a star. The different goal patterns and the optimal robots' positions in these patterns are centrally computed. Then, each robot is assigned an optimised goal position. Finally, the robots will move to their assigned goal position, using a locally chosen velocity and a decentralised collision avoidance algorithm (Alonso-Mora, Breitenmoser, Rufli, Beardsley, et al. 2013).

In our work, the positions of each robot in the final shape will not be centrally computed and assigned to a specific robot. The robot will determine itself its role according to locally obtained information, such as relative positional information.

In Stoy and Nagpal (2004), K. Stoy presented a self-reconfiguration algorithm which uses a spreading gradient system. A gradient is a numerical value passed from robot to robot and used as a relative positional information (see Section 4.1). The algorithm is made for a large group of attached robots in a 3 dimensional space, where each robot has a knowledge about the final configuration. Initially, robots are randomly scattered

and one seed is randomly chosen among the robots and acts as the source of the gradient. The gradient allows the receivers to know the distance between them and the seed, as well as the direction to the seed. Using the *a priori* knowledge, the seed determines which of its neighbouring positions needs to be occupied by a robot and transfers this information to the swarm which will try to fill the holes until the determined structure is complete. Our work will make use of a similar gradient system, but our usage differs (*cf* Chapter 4). Our gradient system will be used to split the group of robots in several regions, and new gradients will emerge from these regions to allow the growth of "limbs" out of the main group of robots. Recently, H. Wang and M. Rubenstein (H. Wang and Rubenstein 2020) developed a model capable of shape formation within a large swarm of Kilobots (Rubenstein, Ahler, and Nagpal 2012). The Kilobots are initially randomly placed in a arena. The desired shape is extracted from an image file and processed by a centralised algorithm that will compute the path for each robot and send to each of them the steps they have to make in order to form the shape without collision. In our work presented in Chapter 4, the exact desired shape of the system is not known in advance. It is derived from the parameters of the system (number of regions, number of limbs, *etc.*) and the swarm will take an approximation of the shape (*e.g.* a round body with 2 limbs), but the exact shape (*e.g.* the exact position of the limbs, their angle compared to the body, *etc.*) is not known in advance.

SR is also present in impressive shows for popular events, where hundreds of Unmanned Aerial Vehicles (UAVs) orderly flock into different shapes in the sky (Kaplan 2016; *Ehang, China* n.d.). In such swarms the whole choreography of drone movements is computed in advance "top-down" and played back—a crucial difference with "bottom-up" autonomous systems.

Even though the work we present in this thesis is related to ground robots, SR is also present in marine environment, for oil spilling management (Kakalis and Ventikos 2008) or marine environmental monitoring (Duarte et al. 2016; Lončar et al. 2019). AquaBotix[1], for instance, developed in 2018 a swarm of Unmanned Surface Vehicle (USV) and Unmanned Underwater Vehicle (UUV) named SwarmDiver[2] (Woolsey, Kitts, and Amend 2019). They are used to accomplish different tasks, from defence goals to ocean research, and are equipped with several sensors and data collection tools. In addition, other works took interest in marine swarms and investigated the synchronicity (Yu et al. 2019) and trajectory planning (Hajieghrary, Kularatne, and Hsieh 2018) of marine vehicles.

As we showed in this section, Swarm Robotics can be applied to a wide range of problems, using self-configuration capabilities to solve issues using large groups of simple robots that a single specialised robot could not resolve. When SR is used in conjunction with growth and developmental principles, as well as limiting global knowledge for the robots in the swarm, SR can achieve self-assembly much like a natural complex system. This opens up a new research field: Morphogenetic Robotics (MR). In the next section, before tackling Morphogenetic Robotics, we will discuss Morphogenetic Engineering in order to give a better overview of the field in which MR is comprised.

[1] www.aquabotix.com
[2] https://www.aquabotix.com/micro-usvs.html

2.3 Morphogenetic Engineering & Robotics

In this section, we shift our focus to Morphogenetic Engineering (ME), a field deriving its principles from Morphogenesis toward artificial life models and innovative bio-inspired technology. Founded by R. Doursat and H. Sayama (Rene Doursat 2011; René Doursat, Sayama, and Michel 2012), ME focuses on a specific class of complex systems that self-organise and self-assemble in a non-trivial and controlled fashion. To model and simulate morphogenetic systems, one should not attempt to design their macroscopic structure directly but rather "meta-design" the microscopic rules by which their components self-organise and evolve, then observe and evaluate the emergent outcome in comparison with the original goals.

In ME systems, communication between entities is based on principles taken from cell communication in biology (Hancock 2003): (1) cells use signalling molecules sent into the surrounding environment to neighbouring cells, and ME uses gradient values that propagate from neighbour to neighbour, which are numerical values representing relative positional information (see Chapter 4.1); (2) via direct connection of a cell to the receiver cell, by opening a "bridge" between the two cells and sending information through it, and in ME, neighbours can send each other specific messages; and (3) via proteins on the membrane that can be recognised by other proteins on another cell's membrane, and in ME, entities in the system can sense information from their neighbours cells such as gradient values.

Doursat extended his work through models of simple and directed cell evolution based on GRNs (René Doursat 2010). He showed multi-scale pattern formation on a growing sheet of cells, demonstrating the possibility of shape creation using ME based

on hierarchical networks of GRNs. Each location on the lattice executes a GRN to transform gradients into output values. The weights of the GRNs represent the genotype, and the calculation gives rise to the phenotype. Hence, changing the GRN parameters results in a change of the pattern and shape. Later, R. Doursat and C. Sanchez created an artificial 3-D creature based on the same mechanisms (René Doursat and Sánchez 2014), including cell division and gradient propagation, and growing in a physics engine. Then, they used genetic algorithms to train it to move in a virtual environment with ambient gravitation, such as climbing stair cases.

In summary, Morphogenetic Engineering twisted the principles of Morphogenesis to create complex systems capable of achieving specific tasks by designing the said system with a bottom-up approach while respecting the genotype/phenotype couple specific to Morphogenesis. In the remaining of this section, we will investigate the core principle of our work, the alliance of Morphogenetic Engineering and Swarm Robotics: Morphogenetic Robotics.

Morphogenetic Robotics (MR) can be seen as the application of Morphogenetic Engineering principles to the field of Swarm Robotics. According to Y. Jin in Jin and Meng (2010), MR "focuses on employing genetic and cellular mechanisms in biological morphogenesis for developing self-organising, self-reconfigurable, and self-adaptive robotic systems, covering a wide range of robotic systems, such as swarm robotic systems, modular robots, and intelligent robots". We will not review Intelligent Robots (InR). Indeed, InR represents single robots capable of performing specific and complex tasks and can make use of highly specific sensors, which is not the focus of our work. As for SR, we will review two sides of the same coin: systems using soft robotics and systems using more classic robotic swarms.

Soft Robotics (Majidi 2014; Whitesides 2018) is a field of robotic that focuses on robots made of flexible materials like gels, fluids and elastometers, mimicking biological materials such as skin or organs. Recent works in engineering, such as Tognato et al. (2019) and Miriyev, Stack, and Lipson (2017) developed flexible materials for soft robots.

Soft robots models are in majority inspired by nature, either on a microscopic scale to reassemble cells, or on a macroscopic level to copy how some animals or insects move and communicate. Vergara et al. (2017) presented a new soft robot design. It mimics cell aggregation with pneumatic cube-shaped elements that can shrink and inflate and are equipped with magnets on every side to attach to each other. A. Vergara showed an organism formed with such robots that can move using actuators and aggregate other robots on itself. He also exposed the possibility of copying cells behaviours such as adhesion or migration, and demonstrated the self-reconfiguration capability of a group of 22 robots. As mentionned earlier, soft robots can be inspired by insects, like the Wormbot (Nemitz et al. 2016), which is inspired by the biology of a worm. In the Eciton Robotica project[3] (Malley 2020; Malley et al. 2020), M. Malley took inspiration from the "army ants" and how they form bridges across gaps of different sizes to create a soft-robot model capable of this same task, in a 2-Dimensional space. Those robots are made of a flexible material, allowing them to efficiently move and reposition themselves. They use a system of claws to attach to a velcro surface or to each other, and use a vibration based system to communicate and form bridges to allow a large swarm of them to efficiently cross a gap.

The works presented so far in Soft Robotics focused on transposing reality into

[3] `ssr.seas.harvard.edu/ecitonswarm`

physical robotics and taking strong inspiration from biology, even copying biological and natural behaviours. "Hard robotics" MR systems (as opposed to soft robotics ones) only take light inspiration from biology and Morphogenesis, and adapt their principles to more abstract SR problems.

One of the famous problems faced in SR is crossing the "reality gap" (Jakobi, Husbands, and Harvey 1995; Ligot and Birattari 2018), the fundamental discrepancy between what is feasible in simulation versus what is feasible physically. To cross this famous gap, from simulated ME systems to MR systems using actual hardware, a large number of robots working together is needed. Indeed, as seen earlier, ME needs a large number of entities to simulate the growth of multi-cellular organisms and morphogenesis principles. Indeed, the reality gap is an important challenge. When simulating a robotic system, one needs to introduce voluntary errors in sensor readings, part breaks in robots, and external events to the simulation or micro-variations in actuators in order to be closer to reality. Even then, creating physical robots from a simulation remains difficult. In out work, the simulations described are used to validate our models, and not how our physical robots react to the real world. That is why our reality gap is not about crossing from a simulated robot swarm to a physical robot swarm, but crossing from a simulated ME model to the physical implementation of the model.
We will now present and discuss relevant works focused on modular structures in robotic swarms in order to introduce the work presented in Chapter 5.

Some swarm systems contain a large number of simple and cheap mobile robots, creating a dense "herd" such as the Kilobot platform (Rubenstein, Ahler, and Nagpal 2012). On the ground, units cluster together to maintain local communication and possibly display patterns, typically guided by "chemotaxis" based on virtual pheromones. Other

works experiment with smaller flocks of unmanned aerial vehicles for indoor exploration (Stirling, Wischmann, and Floreano 2010), also examining self-reconfiguration in case of faulty rotors (Gandhi et al. 2020), or schools of (sub)marine robots performing synchronous encounters (Yu et al. 2019) and cooperative load transport (Hajieghrary, Kularatne, and Hsieh 2018). At the other end of the spectrum, significant efforts were devoted to the design of sophisticated parts and actuators capable of physical attachment to achieve "modular robotics" (Ahmadzadeh, Masehian, and Asadpour 2016). In these cases, a limited number of units generally only permit sparse and precise formations, such as chains and T-junctions, typically by recursive attachment.

Historically, the Modular Transformer (M-TRAN) (Murata et al. 2002) was one of the first self-reconfigurable robotic kits. A group of M-TRANs can be placed in a certain initial state and go through a series of moves to achieve some target shape. Swarm-Bot (Groß et al. 2006) is a self-assembling system comprising smaller mobile robots called s-bots, which use mounted grippers and sensors/actuators (LEDs, camera) to cling to each other or to static objects, following behavioral rules and local perception. Using the s-bot model, SwarmMorph (Christensen, O'Grady, and Dorigo 2008; O'Grady, Christensen, and Dorigo 2009) is a morphogenetic model based on a script language able to produce small 2D robot formations to achieve certain tasks. As for the SYMBRION project (Kernbach et al. 2008), it created an intricate piece of hardware in the form of a cube that could dock precisely with its peers: the vision was to collectively form "symbiotic" robotic organisms that could move in 3D.

In recent years, modular robot systems have become more commonplace, thanks to cheaper and faster hardware. For example, HyMod (Parrott, Dodd, and Groß 2018) is a set of cubic modules with full rotational freedom, which can be combined to create 3D lattice structures such as snakes or wheeled vehicles. The Soldercube (Neubert and

Lipson 2016) is a similar building-block rotational unit equipped with magnetic ties and internal sensors to detect its orientation and occupied faces. Resting on top of a planar lattice of anchors, Soldercubes are able to transform their spatial arrangement by picking up and dropping off each other through attach/turn/detach combinations of moves. The Evo-bot (Escalera et al. 2018) is another modular concept intended to physically implement the growth of artificial "creatures" as compounds of differentiated robotic modules, each one with a specific function such as resource harvesting or motion control. "Soft robotic" designs, as seen earlier in this Chapter, also attempt to mimic cell aggregation with pneumatic and magnetic cubic elements that can shrink and inflate, giving rise to organisms capable of locomotion.

Morphogenetic Engineering in Robotics emphasises works on very large swarms of robots. In his famous article, M. Rubenstein presents the largest swarm robot works: 1024 kilobots swarms together to form a shape (Rubenstein, Cornejo, and Nagpal 2014). Kilobots in his experiment possess a priori knowledge of the goal shape and use a gradient system emitted from 4 anchors, 4 kilobots were placed at a strategic point in the beginning to form the origin of a coordinates system which is used by the other kilobots to know where they are relative to those anchors points. With this positional information and the a priori knowledge of the goal shape, each kilobot moves along the swarm and stops in a correct position, constantly updating its gradient values so other kilobots know where they are and when to stop. This work is a great technology feat and a real advance in swarm robotics and physical experiments. Our work is greatly inspired by it, notably with the usage of gradients in our models, even though the work presented in this thesis aims for more autonomy in the final shape of the swarm, without any a priori knowledge distributed to each robot.

In the SwarmOrgan Project[4] (Oh and Jin 2014b; Oh and Jin 2014a; Shirazi, Oh, and Jin 2014; Oh, Shiraz, and Jin 2016; Oh, Shiraz, and Jin 2018), researchers worked with GRNs and gradient based behaviours to develop a self-assembling swarm of Kilobots capable of tracking a target and herding it to a specific location. Here, the target was acting as a source of gradient propagation which served as a beacon for individuals in the swarm, and the GRNs controlled each robot.

First, they developed an evolving hierarchical GRN (EH-GRN) (E. Davidson and Levin 2005; Erwin and E. H. Davidson 2009) to control a swarm of robots in Oh and Jin (2014b), and used ME principles for pattern formation, precisely chemical based gradients present in the environment (hence using stygmergy). The goal was for the robots to entrap a target whilst avoiding others. The EH-GRN is composed of two layers: the upper layer generates the pattern in which the swarm need to organise and the lower layer controls the robots based on this global pattern.

In Oh, Shiraz, and Jin (2018), the SwarmOrgan project crossed the reality gap and implemented their collective tracking and herding model into Kilobots. One of the robots acts as the target and emits "chemicals" into the environment, symbolised by morphogenetic gradients. Then, other robots can detect these gradients and encircle the target.

In Molins, Stillman, and Hauert 2019, P. Molins presents a trail formation method for large robot swarms inspired by diffusion limited aggregation Kassner 1996. Before discussing his work, a quick aside on multi-agent systems (MAS). Multi-Agent Systems are systems composed of several intelligent agents interacting with each other in order to achieve a goal or solve a problem collectively (Kubera, Mathieu, and Picault 2010). They are different from agent-based model because in an agent-based model (*e.g.*

[4] `http://www.swarm-organ.eu/`

Reynolds' flocks Reynolds 1987), agents are not necessarily "intelligent". Typically, agent-based models are usually used to study an emergent behaviour where MAS can be used to tackle complex computational or engineering problems. Back to P. Molins' work, he made use of a MAS to form trails of robots between a source and an area of interest using a method inspired by diffusion limited aggregation algorithm. Robots perform random walks until they sense neighbours connected to a trail, using virtual pheromone gradients. His model is proven to be able to find the closest area of interest and to adapt to the environment by finding trails avoiding obstacles. This work is focused on the goal of finding the closest area of interest using a morphogenetic robotics swarm, whereas the work we present in this thesis is primarily focused on the shape we aim to form with the swarm rather than the functional goal.

In Slavkov et al. (2018), I. Slavkov used a large swarm of Kilobots to implement the mathematical model of chemical propagation described by Turing in Turing (1952).

Recently, Vasarhelyi (Vásárhelyi et al. 2018) presented a decentralised swarm of UAVs capable of self-reconfiguration, obstacle avoidance and movement coordination, thanks to local perception and communication only. They experimented this behaviour in a swarm of 30 drones using a complex realistic simulation and a Covariance Matrix Adaptation Evolution Strategy (CMA-ES) (Hansen and Ostermeier 2001) to evolve the model parameters.

In Carrillo-Zapata et al. 2019, Z. Carillo presents a Morphogenetic Engineering model for a large swarm of Kilobots (up to 300 physical robots), and based on local gradients, with a reaction-diffusion model to initialise them. During an initialisation phase, the reaction-diffusion model initialise high concentration zone using gradients. After this phase, kilobots are attracted to the closest high concentration location through

local sensing when moving in the swarm with edge following movement. When a kilobot is close to an attraction point (*i.e.* a high gradient concentration zone), it stops and becomes part of the attraction point. Attraction points are also places at the tips of the swarm, so it is influenced to grow out of the main swarm. This work presents a controlled morphogenesis within a swarm of robots through parameter control, with three main parameters: *max hop* (determines the size of the attraction clusters), *min stop* (affects the location of the cluster where the robot stops, *e.g.* more on the sides or in the middle) and *min for attractor* (controls the minimum size of a cluster). In our work, we took inspiration from their work for our limb growth model, phase separation during the unfolding of our model and the control of the desired shape by parameters (see Chapter 4), although we aim to use different robots, larger and in smaller numbers and have even more control over the morphogenesis of the swarm (control over the formation of a body and limbs growing out of the previously formed body).

2.4 Conclusion

As showed in this review, Morphogenetic Robotics is derived from the merging of two large fields of research, Morphogenetic Engineering and Swarm Robotics. Where Morphogenetic Engineering aims at using nature-inspired principles, specifically Morphogenesis principles, to create and meta-design self-assembling and self-reconfigurable systems, being agent-based systems capable of growing artificial creatures or more abstract systems such as Ant Colony Optimisation systems; and Swarm Robotics focuses on the usability and feasibility of large robotic systems composed of cheap and easily replaced units but with no limits on which type of controller is used; Morphogenetic

Robotics meanwhile aims to create large completely decentralised robotic systems using ME principles to self-assemble and achieve specific goals with no external supervision, in the most organic way possible.

The majority of the works in the MR field of research focused either on: (1) the Kilobot model (Oh, Shiraz, and Jin 2018; Slavkov et al. 2018; Gauci et al. 2017; Rubenstein, Ahler, and Nagpal 2012), a cheap and reliable model allowing the realisation of experiments with dozens or hundreds of units; and (2) complex and specific robotic models, handcrafted for the specific research problems researchers try to solve (Malley 2020; Malley et al. 2020; Vergara et al. 2017; Nemitz et al. 2016).

The work presented in this thesis is focused on a swarm exhibiting morphogenetic properties, made of cheap 2-wheeled robots, the PsiSwarm model (see Section 3.2). The Psiswarm model are equipped with embedded hardware functioning with MbedOS[5], allowing the bots to perform complex computation online. Their two-wheeled based movement offers them good movement flexibility, with a full control of their movement in a 2-dimensional plane; and communication wise, Psiswarms are equipped with Bluetooth technology giving them a good communication range compared to other wired-based of Infrared-based communication. However, Psiswarm detection capabilities, based on small infra-red sensors placed around the bot, are limited to a few centimeters, and Bluetooth communication is not the most scalable and reliable technology, and although it allows a good range, it is still limited compared to Wifi or LTE (Long-Term Evolution) technology. On the other hand, Kilobots are cheap and easy to produce to conduct physical robotic experiments with large numbers of entities, giving them the advantage to make the results stochastically relevant for real world applications but they possess very limited movement capabilities and computation capacity. In this work, we

[5] `https://os.mbed.com/mbed-os/`

wanted to show that MR could be used with larger robots in similar situations where kilobots are used, making use of their compute and movement capabilities, better than the kilobots'. Additionally, when Soft Robotics MR works focuses on replicating behaviours seen in nature with hardware as close as possible to nature, "Hard Robotics" MR mainly focuses on using ME principles applied to more classical hardware (*e.g.* PsiSwarms or Kilobots) in order to study natural phenomenons in controlled environments or using these phenomenons and principles to solve a wide range of tasks with a swarm. Our work focuses on using ME and Morphogenesis as it is present in nature, to create new "multi-robotics organisms" themselves capable of achieving tasks. Instead of using the swarm as a tool to achieve a tasks, we use the swarm to create the tool using ME principles, and then use this tool to solve the tasks, allowing more flexibility for the swarm in term of task achievement, and more possible applications with the same swarm.

This concludes the literature review Chapter of this thesis. In the next Chapter (Chapter 3), we will present the experimental setup used for our contributions (Chapter 4 & Chapter 5), describing first the basic principles taken from ME used by our models, then we will describe our simulation tool and the robot model we used for our physical experiments.

Chapter 3

Experimental Setup

In this chapter, we will be presenting the software tools and hardware we used in our works. First, we will discuss the basic principles our work uses: the neighbourhood computation using a trimmed Delaunay triangulation and the spring forces application used to calculate trajectories. Secondly, we will describe the simulation tool we used, the MORSE simulation environment[1], running with Python. As MORSE is a real time simulation, we developed a new tool called *Logvis* to enable replaying the simulations offline. Finally, we will present the robots we used for the physical experiments: the PsiSwarm platform designed by James Hilder and Jon Timmis at the York Robotics Lab[2], and the controller we used to monitor the robots in real time, ARDebug (Millard et al. 2018).

[1] http://www.openrobots.org/morse
[2] https://www.york.ac.uk/robot-lab/psiswarm/

3.1 General principles

Our work is about Collective Robotics and Morphogenetic Engineering, thus it makes use of a flock of robots, interacting with each other in a decentralised manner. These robots communicate with their close neighbours to exchange information and pass messages through the flock. The complex system formed by the flock needs to be represented as an abstract model for us to simulate it and implement our ME model within the flock first in simulation, and then in hardware. Hence, we need to precisely define what is a robot at an abstract level, as well as how its neighbourhood is formed and how it communicates.

We represent our systems as a distributed, multi-agent system where each agent relies only on local perception of the environment to control its behaviour and communicate with the agents that it detects in its vicinity. All the computational logic is embedded in the agents to obtain a fully decentralised system. We chose a MAS for its intrinsic properties (Sycara 1998): (1) each agent has incomplete information or capabilities for solving the problem and, thus, has a limited viewpoint; (2) there is no global system control; and (3) computation and data processing is asynchronous and decentralised. These properties mimic almost perfectly a system composed of multiple independent robots.

In addition to the definition of the agent itself, the definition and computation of each agent's neighbourhood is central to the cohesiveness of a collective robotic organism, as it ensures the proper coordinated propagation of information across the flock. To this aim, we use a hybrid "topological-metric" type of neighbourhood implemented by a modified Delaunay triangulation, chosen for its accurate representation of physical contacts and robustness to change (Shamos and Hoey 1975).

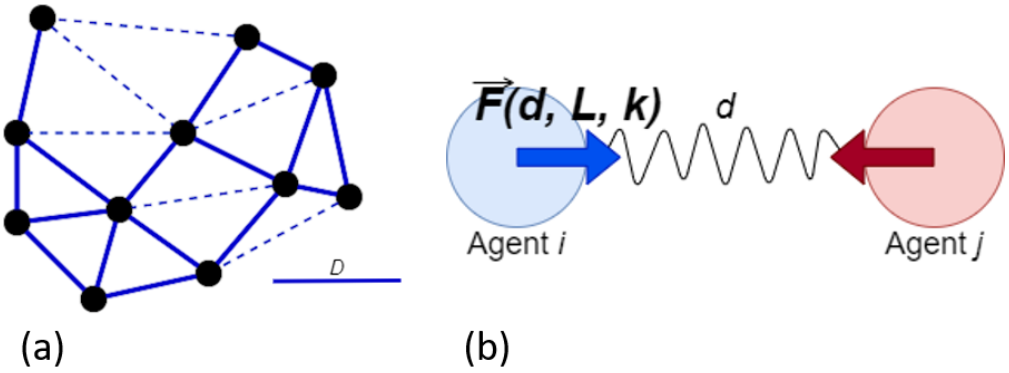

Figure 3.1: **Neighbourhoods and forces.** (a) Example of a Delaunay graph among a dozen agents where connections are trimmed (dashed lines) above a given cutoff distance D. (b) Two connected agents i and j at distance d exert virtual and opposite elastic forces of magnitude $F = k|d - L|$ onto each other (implemented by wheel-based movements), where L denotes a free length and k a rigidity coefficient. If $d > L$, i and j move toward each other; otherwise, they pull apart.

A Delaunay triangulation is the dual graph of the Voronoi diagram of a set of points (Fortune 1995), and forms a set of triangles. Each triangle's circumcenters[3] is a point within the Voronoi diagram. To keep things simple, if two points are connected with an edge in the Voronoi diagram, then the two resulting triangles (with said points as circumcenter) will share an edge in the Delaunay triangulation.

Our modification consists of pruning connections that are longer than a given threshold, set just below the average minimum distance of uniformly distributed agents in space in order to accommodate real-world constraints (Figure 3.1a). Since the Delaunay triangulation is not metric-based, far away robots may also be connected and this is why a cutoff length was introduced to prevent unrealistic long-range communication.

Each neighbourhood connection can also carry a virtual spring creating elastic forces (Figure 3.1b), which translates into wheel-based movements by each agent to stay at a

[3]Circumscribed circle's center

certain optimal distance from its neighbours, neither too close to avoid collisions, nor too far to remain within signal range. Each agent will have a set of polar coordinates sP, containing tuples (a, d) with a an angle and d a distance. Each polar coordinate is calculated from the position of its neighbours. The force F resulting from each polar coordinate is calculated with: $F = k|d - L|$, where k represents the rigidity coefficient (how strong is the virtual spring between the two agents) and L is the free length of the virtual spring (the distance where the spring will exert no forces on the agents).

From sP, a set of forces sF is calculated. The average force F_a is calculated from sF to give the agent the final spring force applied to it by all its neighbours. From F_a, the agent will get a target angle a_t that it has to align to and a target distance $dist_t$ that it has to move in order to apply the final spring force. The agent will then turn until facing a_t and move by $dist_t$. This process is repeated until the agent finds itself into a stable position, where all the spring forces from its neighbours are at free length and exert no more forces.

3.2 PsiSwarms and ARDebug

The PsiSwarm platform, a disc-shaped robot on wheels, was designed by James Hilder and Jon Timmis at the York Robotics Lab[4]. It runs on Mbed OS[5], an open-source real-time operating system for the Internet of Things. Its control code in C++ is uploaded to the board via a USB link. PsiSwarms are equipped with the following components (Figure 3.2): an Mbed LPC1768 Rapid Prototyping Board[6], the heart of the operation containing the code, plus a Micro-USB plug and a Bluetooth emitter/receptor; an LCD

[4]https://www.york.ac.uk/robot-lab/psiswarm/
[5]https://os.mbed.com/mbed-os/
[6]https://os.mbed.com/platforms/mbed-LPC1768/

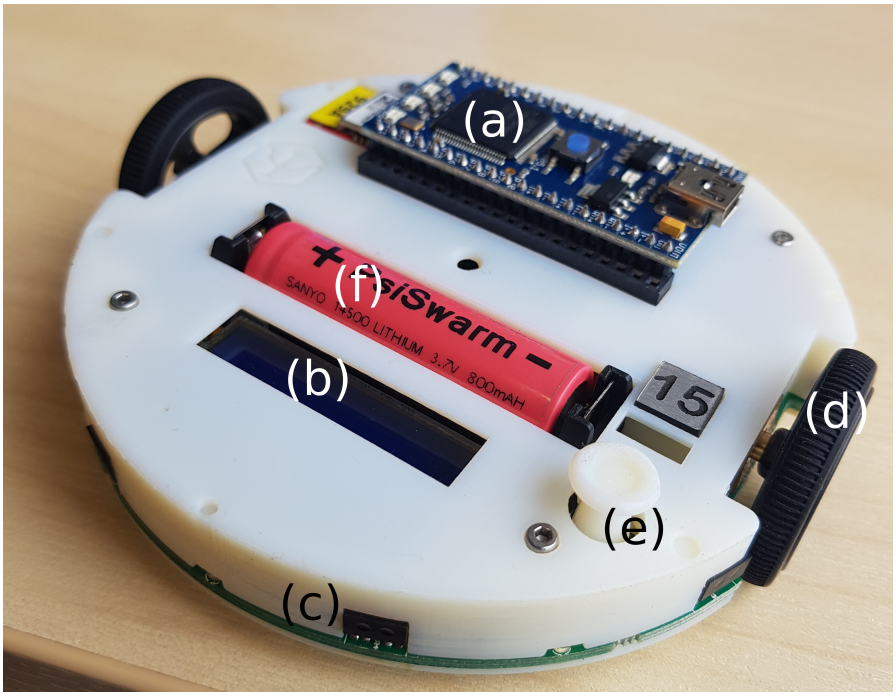

Figure 3.2: **PsiSwarm platform** with (a) mother board Mbed LPC1768 Rapid Prototyping Board, (b) LCD screen, (c) infrared sensors, (d) wheels and motors, (e) joystick for user input, and (f) battery.

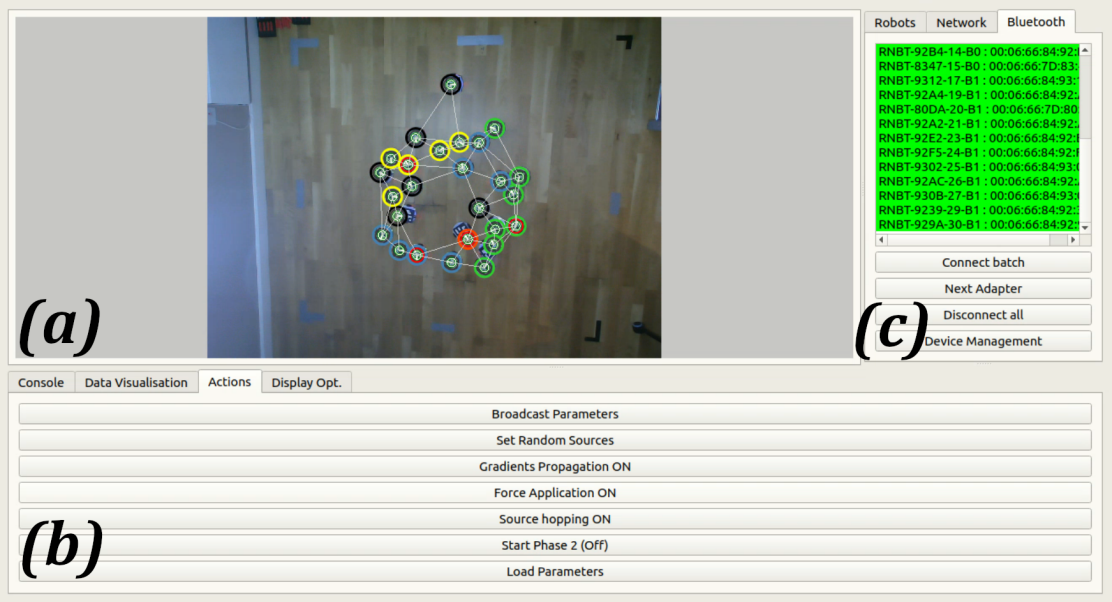

Figure 3.3: **ARDebug software** *(a)* Camera display: PsiSwarms are equipped with ArUco markers and Augmented Reality is used to display additional information; *(b)* Bottom panel: information about the PsiSwarms' internal states and global actions available; *(c)* Right panel: connected PsiSwarms and buttons to communicate with them.

screen; eight infrared sensors placed at quasi-regular intervals around the robot; two wheels and motors; a small joystick to input commands into the robot; and a single battery.

To centrally monitor the PsiSwarms in real time, whether to read out their trajectories or intervene in the experiment, we relied on the ARDebug software (Millard et al. 2018), an augmented-reality tool that can track the robots with ArUco square markers pasted on top of them (Garrido-Jurado, Munoz-Salinas, et al. 2014; Garrido-Jurado, Muñoz-Salinas, et al. 2015; Romero-Ramirez, Muñoz-Salinas, and Medina-Carnicer 2018) (each one carrying a binary pixel matrix that encodes a unique ID number), and can exchange information with them via Bluetooth. The software uses a top-down bird's-eye camera (mounted on the ceiling) to evaluate the location of every PsiSwarm and link this information with the right Bluetooth sockets to communicate with them

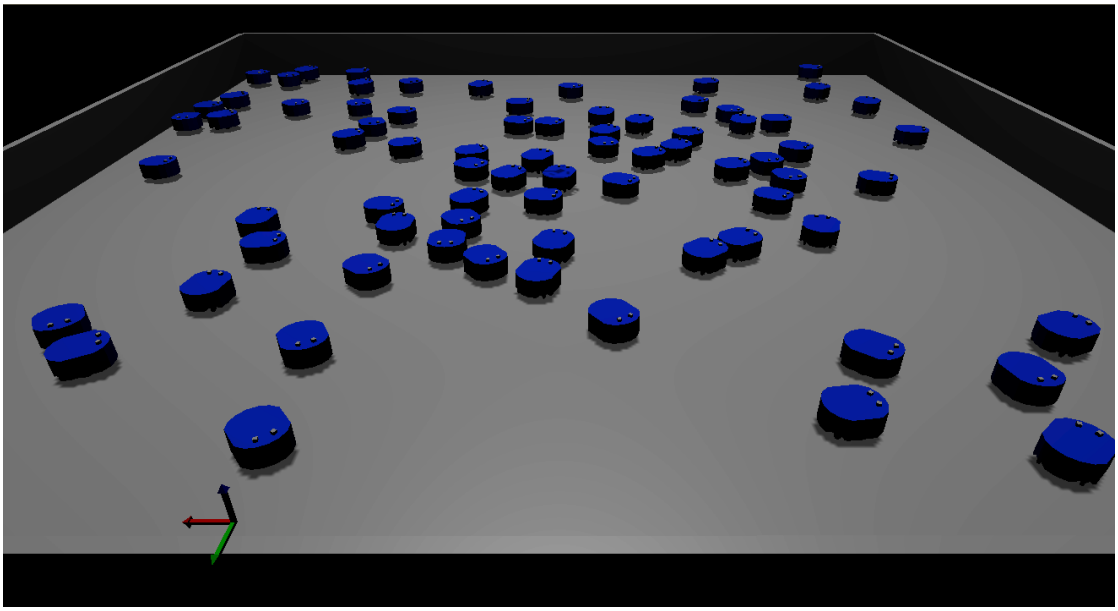

Figure 3.4: **MORSE simulations visual display** Screenshot of the MORSE simulation environment, displaying 80 simulated robots within an arena of $16 \times 16MU$.

individually. This allowed us to send to/receive from each robot data, both in the beginning and during the experiment. We tailored the software to our needs, adding specific messages sent by the user for specific commands (*e.g.* move from phase 1 to phase 2, see Section 4.1) or specific displays depending on the model used (*e.g.* display the links of different colours for our chain-like structures, see Section 5.2.2).

3.3 MORSE simulator

To simulate our work in a controlled environment, we used a simulation tool with a realistic physics engine to simulate robots behaviours as close as possible to reality.

Tools such as NetLogo (Tisue and Wilensky 2004) were first considered to simulate

our model. NetLogo is an interactive and educational multi-agent development environment capable of modelling large groups of agents, homogeneous or heterogeneous, following the principles of complex systems: behaviour based on local perception, strictly local interactions and "stigmergy" (indirect inter-agent communication through changes in the environment). However, such tools are not detailed enough for robotic simulations, especially in term of motion and realistic constraint modelling. This is why we chose the MORSE simulation tool.

The simulation allowed us to test and adjust our models with more flexibility, in order to prepare the ground toward bridging the reality gap with the physical experiments. To achieve this aim, we chose the MORSE simulator environment[7], a platform written in Python and powered by the Blender physics engine[8]. MORSE offers accurate representations, physics simulation and detailed graphic display of robotic components and external objects (Figure 3.4). Within MORSE , all distances are expressed at an arbitrary scale. For future references of distances within the MORSE simulator, we use a unit we refer to as *Morse Unit (MU)*. Each simulated robot has a diameter of $0.5MU$. In the MORSE simulator, 1 *MU* represents 1 meter. However, we chose to treat distances in an arbitrary scale. Our simulation is mainly used to validate the behaviour of our models, in addition to simulate how the PsiSwarms move physically. The physical simulation offered by MORSE , albeit not being used to its full potential, allowed us to tweak and fine tune how our robots react to spring forces in the physical world, and test different gradient propagations for our models (see Chapters 4 and 5).

The MORSE simulation follows the standard agent-based model paradigm: (1) an agent can only rely on its local perception of the environment to control its behaviour;

[7]http://www.openrobots.org/morse
[8]https://www.blender.org/

(2) it can only communicate with neighbouring agents determined by its local perception; and (3) all the logic and computation (the "creature's intelligence") is distributed over, and embedded into, the agents to obtain a fully decentralised system.

Each agent of the simulation is instantiated by an autonomous low-height cylindrical robot endowed with its own virtual control and sensorimotor abilities: two wheels and motors, eight infrared proximity sensors distributed on its periphery, and a communication module for short-range broadcast. We chose these specifications to match the simulated robot model to our real robot platform. In both of our contributions, PsiSwarms make use of a simulated camera to detect the "centre of mass" of the swarm, where the bulk of the other robots are located (see Sections 4.3 & 5.2.2). The turret camera is not simulated but replaced by global information about the centre of mass of the flock, which is sent to the robots that needed it. This way to implement the turret camera was made to be closer to our physical setup. Our robot, the PsiSwarm, is not equipped with a camera or a way to sense robots position in its surrounding. To make them capable of such feat, we decided to use a bird's eye view camera, and use the centre of mass of the swarm as a way to estimate the position of the swarm. To make this solution viable in a fully decentralised system, each robot should have an embedded camera to detect other robots. For example, a depth camera could be used by the robots to detect the distance of others around them, and inferring the swarm's centre position. Another way to make this possible would be to use a camera capable of detecting LEDs on each robot, allowing one robot to estimate where the swarm is positioned compared to its position.

Alongside the MORSE simulation environment, our simulations use a Python script to encode the behaviour of each agent of the simulation. This controller is identical for

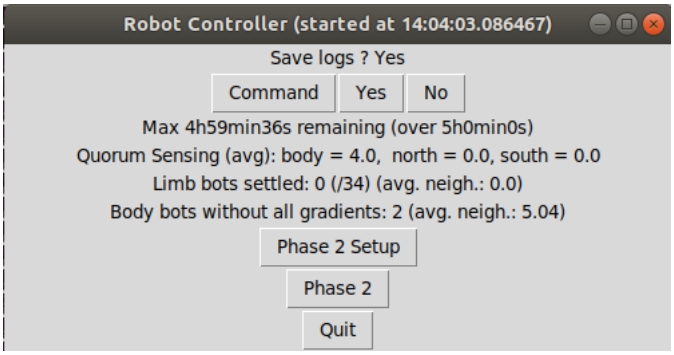

Figure 3.5: **Agent controller for MORSE simulator** Our agent controller display, where basic information about the current simulation is displayed and some actions can be taken in real time.

each agent to respect the agent-based simulation paradigm, and hold additional global information about the simulation for display to the user while the simulation runs (Figure 3.5). Such information can be, for the work presented in Chapter 4, the number of bots forming the body and limbs of the multi-robotic creature, the real time elapsed since the beginning of the simulation or quorum sensing values for limb formation.

The MORSE simulator enables real-time monitoring of robots (rendered in Blender) with additional information, such as frame rate or CPU usage. When using MORSE , you can set the seed of the simulation to repeat experiments. However, each experiment can last several hours, and re-running simulations to study how it unfolds can be a difficult process. Therefore we developed an external visualisation tool using Python 2.7 and the PyGame library[9]: *Logvis*. First, we log what happens in MORSE at periodic intervals: robot positions, directions, gradient values, neighbours and other useful information.Then, all log files are parsed and processed in order to replay the simulation in our visualisation tool, with additional useful information such as neighbourhoods,

[9] `www.pygame.org`

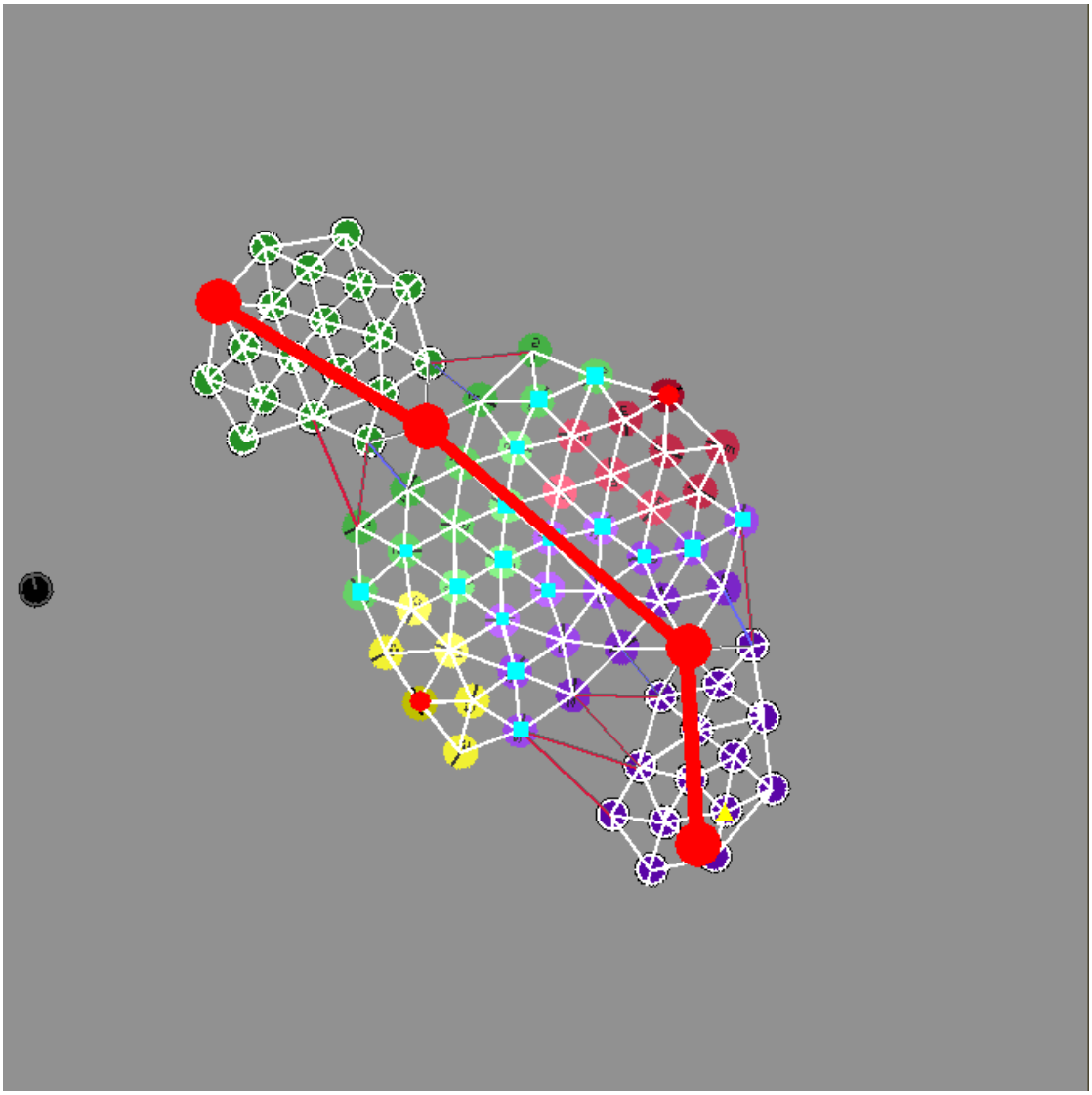

Figure 3.6: ***Logvis*: Log visualisation tool.** Each disc represents a robot, the links between discs represent neighbour connections, with different colours meaning different type of links, the disc colours and markers on the disc represent different information related to the MR model used for this specific simulation.

gradient values, or robot types—otherwise not displayed in MORSE (Figure 3.6).

3.4 Conclusion

In this chapter, we presented all the necessary building blocks we used for our contributions. First, we presented the general principles used by our models: neighbourhood compilation and trajectories computation.

For the first, we used a trimmed Delaunay triangulation. Indeed, the Delaunay triangulation is a renown and robust topological triangulation amongst a group of points, and using a version with links trimmed according to a distance threshold allows us to keep the robustness while having a more realistic neighbourhood. Regarding trajectories computation, we chose a spring force system, were agents have simulated spring forces applied to them, from which resolve a final movement vector that they follow. We chose this way of representing movement in our model for its similarities with how cells in biology move, aggregate and pack together.

Then, we presented our simulation tool, MORSE , chosen for its realistic physics system, and our log system capable of replaying simulation with enhanced information display. Finally, we showed our robot unit, the PsiSwarm, used for our physical experiments and proof of concept.

In the next two chapters, we will present and discuss our contributions in the field of Morphogenetic Robotics. Our first work, presented in Chapter 4, is currently under preparation for submission to a journal. We present a model of Morphogenetic Robotics for a flock of robots, allowing it to self-organise into a multi-robot organism composed of a body and several limbs. Our second contribution, presented in Chapter 5, is based on our paper published at the Twelfth International Conference on Swarm Intelligence

ANTS2020[10] (Gaget, Montanier, and René Doursat 2020), and proposes a model of collective robot dynamics based on ME principles, in particular an algorithm of programmable network growth, and how it allows a flock of self-propelled wheeled robots on the ground to coordinate and function together.

All the work presented in the next chapters makes use of the general principles discussed in this Chapter.

[10]https://www.iiia.csic.es/ants2020/

Chapter 4

Fostering the Growth of Multi-Robotic Organisms

In this chapter, we will present and discuss our first contribution in the field of Morphogenetic Robotics. We present a model of Morphogenetic Robotics for a flock of robots, allowing it to self-organise into a multi-robot organism composed of a body and several limbs. We apply morphogenetic engineering principles to collective robotics with the goal of creating self-assembling "multi-robotic organisms" made of many small mobile robots moving in specific spatial formations. Simulation results show these principles at play in a large group of wheeled robots, while preliminary experimental results pave the way toward physical results.

The literature review for this contribution is covered in Chapter 2.

This contribution is described as follows: in Section 4.1, we present the model used in our work to create a multi-robotic organism. In Section 4.2 we present and discuss the obtained simulation results. Finally, we show in Section 4.3 our preliminary physical experiments results in before discussing our work in Section 4.4.

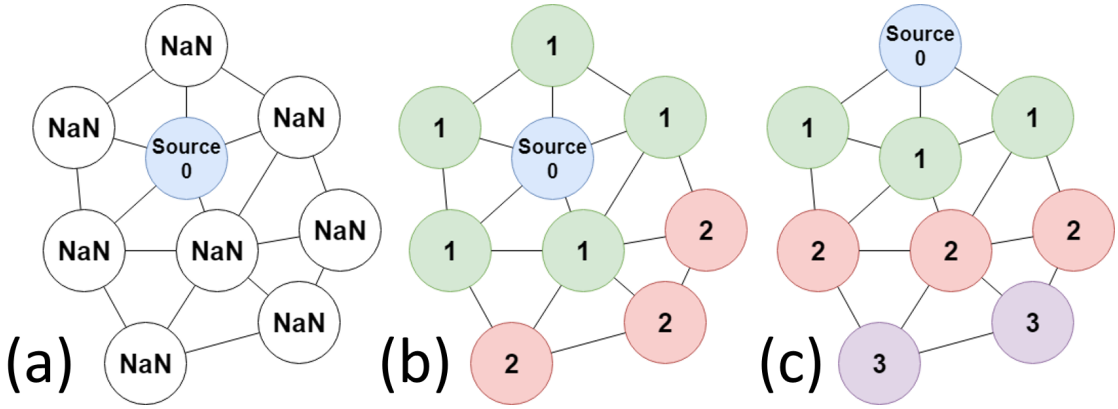

Figure 4.1: **Gradient Propagation.** (a) Initialisation: only the source has a gradient value of 0; all other agents are empty (NaN). Links between agents represent neighbourhoods. (b) Gradient propagation: following the rules described in the text, each agent computes its gradient as a function of its neighbours (here colours code for different integers). (c) Source change: here the former source chooses to pass on its role to a more suitable neighbour, thus changing the whole gradient landscape by ripple effect.

4.1 Model

The "meta-design" methodology applied in this project is composed of hand-made rules distributed in all agents to grow a multi-robot organism. Robots are able to form an organism using a set of rule, by making local decisions based on their local detection and exchange with their neighbours.

The definition and computation of each agent's neighbourhood is based on the description present in Section 3.1 (Figure 3.1).

4.1.1 Positional information: Gradients

In our model, gradients are integer values that spread across the organism conveying relative "positional information", and are referred to as "positional gradients" or "super-gradients" throughout this Chapter. A gradient is akin to a hop counter propagating throughout a group of agents. They originate from one or more source agents that hold

value 0 (Fig. 4.1a) and are incremented by 1 as they hop from cell to cell (Fig. 4.1b). Within one gradient, all agents of the organism, except the sources, update their values via the following rules:

0. Initial state at time t_0: set value to NaN ("Not a Number").

1. If current value is n, send $n + 1$ to all neighbours (if NaN, send nothing).

2. Receive values from neighbourhood (one value per neighbour):

    - If no values are received, reset value to NaN.
    - If own value is n and no received value is n (i.e. agent was not connected to any neighbour holding $n - 1$), reset value to NaN.
    - Compute minimum of received values, m, and adopt m as new value if and only if current value is NaN or $n > m$.

Sources can also change during development (Fig. 4.1c). As stated earlier, a source indicates an agent within the organism which holds a gradient value of 0. This source will always keep a value of 0, and all other agents will update their gradient value according to the distance to the source (see Fig. 4.1). If a source agent of a particular gradient detects another agent around it that is more "suitable" to be a source for that gradient, it will pass on its role to this neighbour. Passing the role of source from an agent to another requires certain conditions to be met. First, the agent which passes the role of source must be a source itself. Then, if the source agent sees a neighbour who would be more suitable as a source, it contacts this agent to make it a source. In practice, it means the current source agent makes itself not a source anymore, and sends a signal to the more suitable neighbour to make it a source. In order to make this possible, each agent must be capable of differentiating a suitable agent from another by looking at

the source suitability. To determine this source suitability, different conditions can be followed according to the design of the experiment, i.e. the shape that the organism is trying to achieve. For instance, the condition can be simply: if a source's neighbour has a greater *other* gradient value (and is not already a source itself). This particular rule will have the effect of pushing sources to the borders of the organism. The rules we are using in our experiments are detailed in Section 4.1.4.*Model development*. The mechanism of the sources moving through the swarm is referred as *source hopping* from now on.

4.1.2 Differentiation

The next processing stage after an agent has updated its positional gradient values is to "differentiate", or calculate its type, as a function of these values. The differentiation function can be denoted by $\tau = f(G_{\mathrm{x}}, G_{\mathrm{y}}, ...)$ where τ is the cell type and $G_{\mathrm{x}}, G_{\mathrm{y}}, ...$ are the super-gradient values. If the super-gradients are $G_{\mathrm{x}}, G_{\mathrm{y}}, ...$, the respective types are noted $\tau_{\mathrm{x}}, \tau_{\mathrm{y}}, ...$ By differentiating into different types, agents in the swarm that have the same type and are close to each other will form regions of this type. If the super-gradients are $G_{\mathrm{a}}, G_{\mathrm{b}}, G_{\mathrm{c}}, ...$, the corresponding regions are noted $R_{\mathrm{a}}, R_{\mathrm{b}}, R_{\mathrm{c}}, ...$ These regions will be used during the development of the organism, as discussed in Section 4.1.4.*Model development*.

4.1.3 Spatial evolution

The growth of a biological organism happens through repeated division, migration and adhesion of its cellular components. In our model of multi-robotic organism, however, robots clearly do not divide like cells do, and literal "adhesion" between robots is also notoriously difficult to achieve via physical means (e.g. mechanical, magnetic

or chemical). Therefore, instead of division and adhesion, our model uses aggregation and proximity: here agents "huddle" together following rules depending on their type, via the gradient values they exchange. Their movements are governed by virtual spring forces between them and their neighbours, as discussed in Section 3.1. These virtual spring forces make robots huddle together into a tight formation, allowing the gradients to correctly propagate through the swarm and the source to move to its ideal position within the swarm. As robots move around because of the spring forces, the background mesh is not static but continually updated (as per Fig. 3.1a) so that new connections may appear and existing ones disappear. If an agent finds itself isolated, *e.g.* by being far away without neighbourhood connections, it uses the camera to search for the bulk of the flock and head in its direction.

Spring forces are not applied similarly for every connection between robots. Some connections apply attraction forces, while others apply repulsion forces or no forces at all. In a similar fashion, gradients do not propagate on every connection. All these subtleties are discussed in the next Section.

4.1.4 Model development

The model development is defined by how gradients are defined within the genome and how the source suitability is calculated, *i.e.* how the source will move from agent to agent according to the positional gradient values. Our model aims to form a multi-robotic organism composed of a central body and limbs growing out of the body. In order to achieve this formation, each agent follow a set of rules, identical for each agents in the swarm, that we call *genome*. All the rules concerning movement, spring forces application and gradients propagation is included in the "genome" and shared by all

agents in the swarm.

As described in Section 4.1.1.*Positional information: Gradients*, we used super-gradients to convey positional information through the swarm. In our model's genome, we have 4 super-gradients, called *North* or G_{n}, *South* or G_{s}, *East* or G_{e} and *West* or G_{w}. *North* and *South* (and *East* and *West*) are called "opposite" gradients. These gradients are integers representing positional information for each cardinal point. Each positional gradient follows the rules described earlier for the propagation, and sources use the following set of rules for the source hopping mechanism: if one of the source's neighbour has a bigger opposite gradient value (*i.e.* is further from the opposite positional gradient's source), if the absolute value of the difference of the other 2 super-gradients is close to 0 (*i.e.* is at equal distance between the 2 other sources) and if the neighbour is not already a source of any positional gradient, then it will pass on the role of source for this specific positional gradient to this neighbour.

When differentiating, an agent will set its $type$ based on its minimum positional gradient value, *e.g.* if an agent have $G_{\mathrm{n}} = 2$, $G_{\mathrm{s}} = 5$, $G_{\mathrm{e}} = 1$ and $G_{\mathrm{w}} = 5$, then this agent will set its $type$ to *East*, or τ_{e}. If several super-gradients have the minimum value, then the $type$ is set by order of priority defined in the genome.

The genome also specifies other important values, all set before the beginning of the experiment:

- N_{BotBody}: the number of agents that will form the body.
- Growing Regions GR: determine which region will grow into a limb, with $GR \in [R_{\mathrm{n}}, R_{\mathrm{s}}, R_{\mathrm{e}}, R_{\mathrm{w}}]$. The number of regions in GR will determine the number of

limbs of the organism.

- N_{BotLimb}: the number of agents that will form each limb.

From the rules stated before comes out a separation of all agents within the organism according to relative "cardinal points", *north* being the top of the body, *south* the bottom, *etc*. In Fig. 4.2 b, we show an example of this separation: green dots represent agents within the body with $type = \tau_{\mathrm{n}}$, red dots agents with $type = \tau_{\mathrm{e}}$, yellow dots agents with $type = \tau_{\mathrm{s}}$ and blue dots agents with $type = \tau_{\mathrm{w}}$.

In order to form the body and the limbs, agents can be one of two sort, called *super-type* T: (1) *Body Bot*, T_{body}: Agents that are "Body Bots" form the body of the organism. They propagate the 4 super-gradients (*North*, *South*, *East* and *West*) and uses the spring forces to form a circle-shaped body with super-gradients evenly dividing it into 4 quarters; and (2) *Limb Bot*, T_{limb}: Agents that are "Limb Bots" form the limbs of the organism, which will grow from the regions GR specified in the genome.

The development of the multi-robotic organism can be divided into two phases.

Phase 1: Body formation

First, agents are randomly scattered in the environment (Fig. 4.2 a). 4 Agents are randomly selected and set as source for one of the 4 super-gradients. This selection is made globally amongst "Body Bot" T_{body}. We also assume that source agents do not fail. Robustness to failure in our work, or rather the lack of, is discussed in the Conclusion in Section 6. All agents then begin phase 1. As long as an agent has one of its positional gradient values set to NaN (or -1), or it has no neighbour, then it will

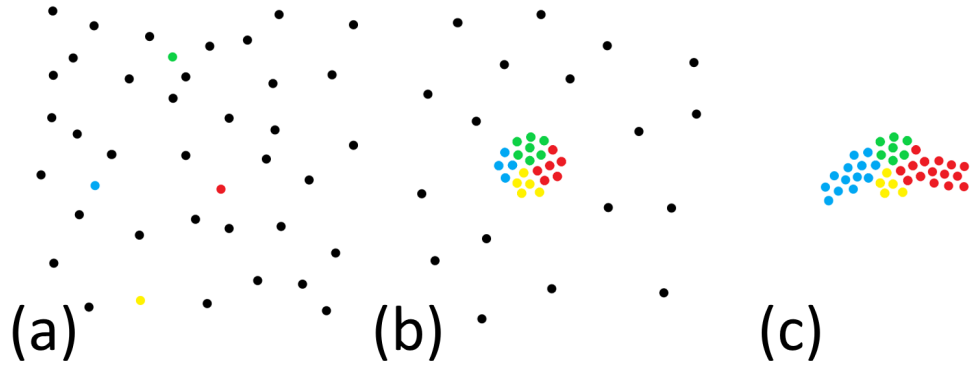

Figure 4.2: **Illustration of the multi-robot developmental stages.** (a) Initialisation: 40 agents are scattered in the environment with empty gradients and types, except four sources (red, blue, green, yellow). (b) End of Phase 1: agents have clustered together in a "body" formation while a few of them are still roaming the environment. Here the green and yellow types no longer accept new agents; only the green and blue one do. (c) End of Phase 2: remaining agents moved in and, as they were pushed away by the non-growing agents, they eventually found the accepting green region and formed a "limb-like" structure sprouting from the body.

use its camera to move closer to the bulk of the swarm. Once it has at least one neighbour and all its gradient values are positive, then it will move according to the spring forces. Through gradual aggregation and by means of the spring forces governing their movements, agents end up forming a disc-shaped organism (fig. 4.2 b). Meanwhile, super-gradients propagate across the swarm and agents also determine their types τ, which create a pattern of distinct regions inside the organism. These regions will play an important role in Phase 2. Once a certain number of agents are connected to each other, or a certain time period has elapsed, the system transitions to Phase 2. The transition condition can vary according to the goal and depend on other measures of cohesion.

Phase 2: Limb Formation

Two new gradient types are introduced in Phase 2: the Inward gradient G^X_{inward} and

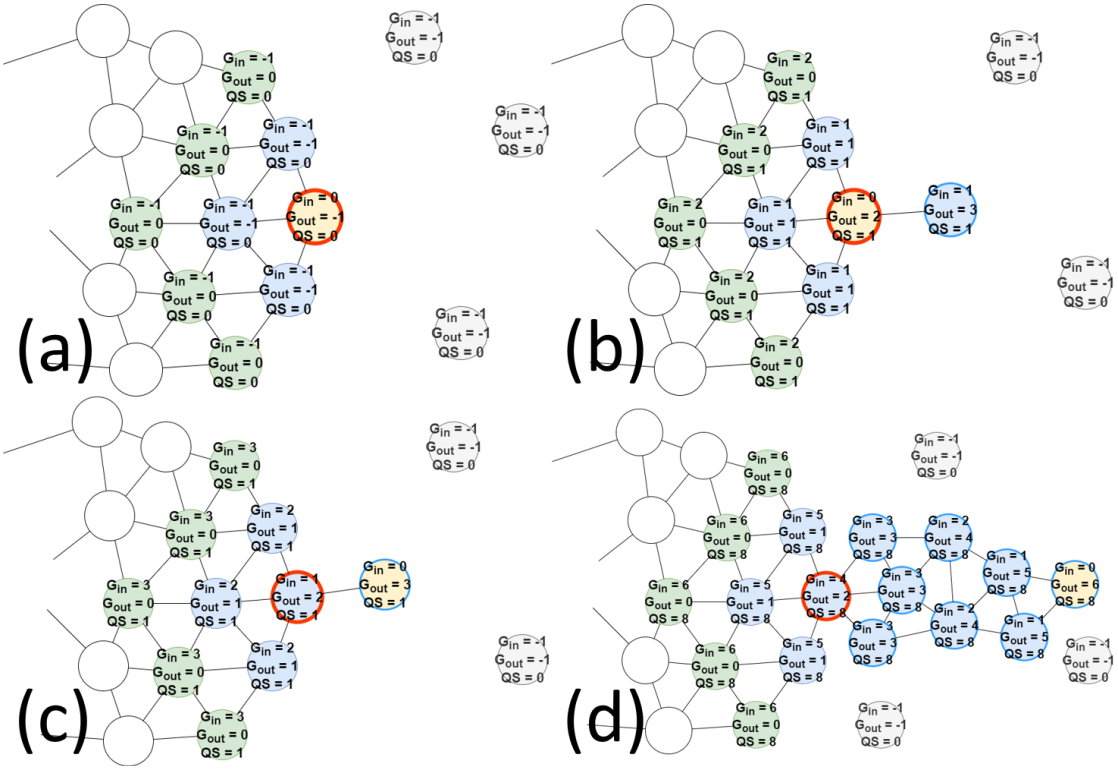

Figure 4.3: **Formation of a limb** Snapshots of an abstract limb formation. Coloured circle represent agents within a growing region, with G_{in} being the Inward gradient value, G_{out} being the Outward gradient value and QS the quorum sensing value. Green circle represent the outward sources, yellow circle the Inward source and grey circle are un-recruited T_{limb} agents. The agent outlined in red is the positional gradient source of the region. The blue outline agents are recruited T_{limb} agents. The genome specifies $N_{\text{BotLimb}} = 8$ (a) End of Phase 1 and before beginning of Phase 2, Inward and Outward sources are set. (b) A T_{limb} agent is recruited by the Inward source as it passes by. (c) As the newly recruited agent has a greater G_{out} value, its neighbour gives it the Inward source role. (d) As time passes, more T_{limb} agents are recruited until the $QS = N_{\text{BotLimb}}$

the Outward gradient G^X_{outward}, where X is the growing region (*e.g.* G^N_{inward} and G^N_{outward} are the inward and outward gradients for R_{n}). The Inward and Outward gradients are numerical values, as the positional gradients, and propagate using the rules described in Section 4.1.1, but only through their growing region. The Inward gradient will propagate from the tip of the limb to the centre of the body, whereas the Outward gradient will propagate from the centre of the body to the tip of the limb.

At the end of Phase 1, just before beginning Phase 2, all agents that are part of the body (*i.e.* that have all gradients with positive values and have at least one neighbour) set their *super-type* T to T_{body}. All the remaining agents set $T = T_{\text{limb}}$. All T_{body} agents that are part of a growing region and within the periphery of the region (*i.e.* has a neighbour with a different type) start as a source for the Outward gradient, $G^X_{\text{outward}} = 0$, and the source of the positional gradient of the growing region is set as source of the Inward gradient, $G^X_{\text{inward}} = 0$ (see fig. 4.3a). While Outward sources are fixed during all of Phase 2, the Inward source can be passed to a more suitable neighbour. If one of the source's neighbour has a greater Outward gradient value, then the source will be moved to it, maximising the distance between the Inward source and the centre of the body (see fig. 4.3b-c). The fixation of Outward sources should not impact adaptability of the limbs because all Outward sources are situated in the body, and the body do not move or change during Phase 2. Because of the redundancy of Outward source, if one should fail the gradient will still propagate through the limb. However, if all sources should fail, there is no fallback strategy.

Additionally, each agent in the system initialise a "quorum sensing" mechanism (see fig. 4.3). The quorum sensing value QS_{X}, where X is the growing region, is set for each growing region and initialised at 0 and used to count how many agents are part of a limb (*e.g.* $QS_{\text{N}} = 5$ means there is 5 T_{limb} agents for R_{n}). This "self-counting" method is based on nothing else but the quorum sensing values of each agent, which are set to 0 once at the end of Phase 1. The quorum sensing value QS_{X} is propagated as followed throughout the whole system, hence all agents have access to this information. The propagation system for the quorum sensing is very similar to to how gradients propagate:

0. Initial state at the end of Phase 1: set $QS_{\mathrm{X}} = 0, for X \in [N, S, E, W]$

1. Send current $QS_{\mathrm{X}} for X \in [N, S, E, W]$ value to all neighbours.

2. Received values from neighbourhood:

   - If $QS_{\mathrm{X}} <$ *min(list of received values)* then $QS_{\mathrm{X}} =$ *min(list of received values)*
   - Else, no changes.

During the development of Phase 2, T_{limb} agents engage in a random walk until they get recruited into a growing region. The recruitment into a growing region is made by the Inward source of this region (see fig. 4.3b). If an un-recruited T_{limb} agent enters into the communication range of the Inward source and $QS_{\mathrm{X}} < N_{\mathrm{BotLimb}}$, where X is the growing region of the Inward source, then it sends a recruitment message containing the growing region X and QS_{X} to the T_{limb} agent. If the recruitment is accepted, the T_{limb} agent sets its growing region to the one received, and sets $QS_{\mathrm{X}} = QS_{\mathrm{X}}^{received} + 1$ and become recruited, thus applying spring forces and propagating Inward and Outward gradients through its neighbourhood. Concerning neighbourhood, a recruited T_{limb} agent has for neighbours only other recruited T_{limb} agents from the same growing region, and the agent that recruited it. All other agents are ignored when calculating spring forces and propagating gradients.

4.2 Simulations

Before trying our model with real robots (see Section 4.3), we implemented it in a realistic simulation. This allowed us to test and adjust the model more flexibly, in order

to prepare the ground toward bridging the reality gap with the physical experiments. To this aim, we chose the MORSE simulator environment[1], which is described further in Section 3.3. A Python script encoding the behaviour of the simulated robots, including their genomic rules of self-assembly, is running in parallel with the MORSE engine. Within this script, each robot is linked to an agent representation. All important parameters are described in Table 4.1. All distance related parameters are expressed in *Morse Unit MU*, used by MORSE and Blender to calculate distances.

Table 4.1: **Simulation parameters**

Name	Description	Value
$nbBotsBody$	Number of robots in the body	46
$nbBotsLimb$	Number of robots per limb	17
$nbLimb$	Number of limbs	2
$nbTotalRobots$	Total number of robots	80
k_{attr}	Rigidity of the virtual attraction springs	1.0
L_{attr}	Free length of the virtual attraction springs	0.8
k_{rep}	Rigidity of the virtual repulsion springs	0.28
L_{rep}	Free length of the virtual repulsion springs	1.6
D_{body}	Cutting distance of the Delaunay triangulation for robots within the body	1.94
D_{limb}	Cutting distance of the Delaunay triangulation for robots within the limbs	1.94
GR	Regions from which the limbs will grow	$[R_{\mathrm{n}}, R_{\mathrm{s}}]$
$A_{\mathrm{h}} \times A_{\mathrm{w}}$	Height and Width of the arena	16×16

Parameters in Table 4.1 were set for different reasons. $nbTotalRobots$, $nbBotsBody$ and $nbBotsLimb$ were set based on the number of robots we wanted in our simulations. $nbBotsBody$ was chosen by making several trials with different number of robots to

[1] http://www.openrobots.org/morse

form the body and achieve a uniform round shape. Too few robots would yield a mis-shape body. $A_{\mathrm{h}} \times A_{\mathrm{w}}$ were chosen to have enough space so all robots could move around. $nbLimb$ and GR are part of the genome. Setting those two parameters as described in Table 4.1 will result in a multi-robotic organism with two limbs growing from regions *North* and *South*. k_{attr}, L_{attr}, k_{rep} and L_{rep} were set to give the best virtual spring parameters possible, to avoid robots bumping into each other if the values are too high, and to avoid robots not moving if the values are too low. Finally, D_{body} and D_{limb} were chosen by testing the impact of the cutting distance on the formation of the body.

4.2.1 Detailed simulation run

During initialisation, robots super-type T is set. Robots are taken randomly and allocated a super-type. $nbBotsBody$ robots are set to T_{body} and $nbBotsLimb \times nbLimb$ robots are set to T_{limb}. Then, robots are semi-randomly distributed within an arena of $16 \times 16MU$. T_{body} robots are randomly scattered at the centre of the arena, in an area of size proportional to the number of T_{body} robots compared to the total number of robots. T_{limb} robots are randomly scattered in the periphery of the central area, where T_{body} robots are not present.

Robots can have 3 types of neighbour: (1) *Attraction neighbour* if two robots are of the same super-type T or one is of super-type T_{limb} and the other is an Inward Source, both robots are attraction neighbours. They can exchange messages (*i.e.* enable gradient propagation) and an attraction force $\overrightarrow{F}(L_{\mathrm{attr}}, k_{\mathrm{attr}})$ is applied; (2) *Repulsion neighbour* if two robots are of different super-type T and none of them is a Inward Source, both

robots are repulsion neighbours. They do not exchange message (*i.e.* no gradient propagation) and a repulsion force $\overrightarrow{F}(L_{\text{rep}}, k_{\text{rep}})$ is applied; (3) *Hook* All neighbours to a positional gradient's source within a GR are designated as "hooks". If a T_{limb} robot is neighbour with a hook, it applies an attraction force $\overrightarrow{F}(L_{\text{attr}}, k_{\text{attr}})$, but they do not exchange messages. Hooks allow the limbs to grow around and from the positional gradient's source and keep a tight base around the body while allowing the Inward and Outward gradients to propagate only through the positional gradient's source.

To automatically run a simulation, we set a overarching centralised system called *Overseer*. The Overseer uses three timers to decide when to transition from phase 1 to phase 2 and when to end the simulation: the "transition timer", the "end timer" and the "ultimate timer". The "transition timer" is used to go from phase 1 to phase 2. It is set to 12 minutes. Every time a robot within the body changes its positional gradient value or a positional gradient source changes robot, the Overseer reset the transition timer. When the timer ends, the Overseer stops phase 1 and start phase 2. The "end timer" is started as soon as phase 2 begins. It is set to 45 minutes. Every time an un-recruited T_{limb} robot is recruited, the timer is reset. When the timer ends, the Overseer ends the simulation. Finally, the "ultimate timer" is set to 5 hours and started at the very beginning of the simulation. When this timer ends, the Overseer ends the simulation regardless of which phase it is in. This timer is a safety net in case the "transition timer" or the "end timer" keep getting reset during the simulation, preventing it to advance normally or end. The *Overseer* was put in place so several simulations could run without human input. It was made to advance the simulation to Phase 2 and stop it as closely as a human would do. It does not interact directly with the simulation, and has no power over anything else expect continuing and stopping the simulation. In theory, it has no impact on the

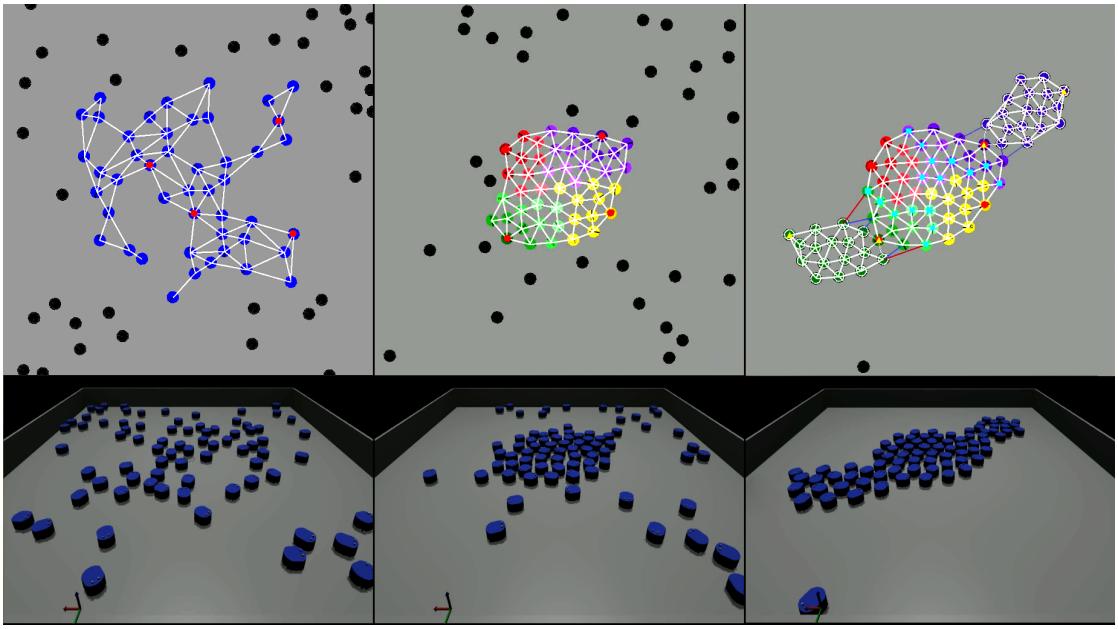

Figure 4.4: **Multi-robotic organism growth in simulation** Bottom: screenshots of the MORSE display at $t = 0min., 39min., 138min.$. Top: custom 2D visualisation tool based on log files at same time. Links: white links represent attraction neighbours, red links represent repulsion neighbours and blue links represent hooks. Parameter values described in Table 4.1. From left to right: Initial state, blue robots are body bots and black robots are free robots ready for recruitment. Red dots represent the one of the gradient's source; End of Phase 1: the body is formed thanks to the spring forces. Colours represent which type $(\tau_{\mathrm{n}}, \tau_{\mathrm{s}}, \tau_{\mathrm{e}}, \tau_{\mathrm{w}})$ a robot is; End of Phase 2: Using recruitment described in Section 4.1.4, the 2 limbs form where the GR are. Yellow triangles represent G^X_{inward} sources and blue squares represent G^X_{outward} sources, where X is its growth region. Video available at `tinyurl.com/DevobotSuppMat`

simulation. However, some problems could arise, such as the "transition timer" being too short, which could result in misshaped bodies and failed experiments. All the times were chosen based on past experience with the simulation, so they would have little to no impact.

The simulated experiment shown in Figure 4.4 uses the parameters values described in Table 4.1, and combine gradient propagation and spatial motion relying on the exploration behaviour and the spring forces, as explained in Section4.1.

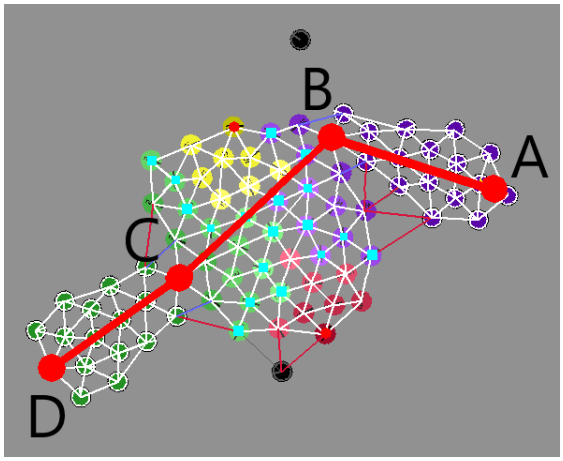

Figure 4.5: **Fitness representation** Representation of the points used to calculated the fitness.

.

4.2.2 Results

To validate our model, we decided to run two medium-scale experiments: (1) A 20-runs experiment using parameter values from Table 4.1 to evaluate the model's stability; (2) A 16-runs experiment where we varied parameters D_{body} and D_{limb} to study the impact of those parameters on our model. We could only run short numbers of simulation because of the computational cost of each simulation (around 5-8 hours per simulation) on the equipement accessible to us when running the experiments.

First, we put in place a fitness value specific for our case, a multi-robotic organism with 46 robots in its body and 2 limbs of 17 robots each (see Fig 4.5. The fitness is composed of a spatial component from the simulation logs and set using 4 points within the multi-robotic organism, and a cohesion component. The fitness value is comprised between 0 and 10. The fitness is calculated as follows:

$$Distance = |\vec{BC}| + (|\vec{AB}| + |\vec{CD}|) \times 1.5 \tag{4.1}$$

$$Angle = (\langle \vec{BA}, \vec{BC} \rangle + \langle \vec{CB}, \vec{CD} \rangle) \times 2.5 \tag{4.2}$$

$$Diff = ||\vec{AB}| - |\vec{CD}|| \tag{4.3}$$

where A is the coordinate of the tip of the North limb, B the coordinate of the north positional gradient's source, C the coordinate of the south position gradient's source, D the coordinate of the tip of the South limb.

$$GeometryFactor = \frac{Distance + Angle - Diff}{Ideal} \times 10 \tag{4.4}$$

Where $Ideal$ is the ideal value for the fitness, and $Distance$, $Angle$ and $Diff$ are calculated as described in Equation 4.1, 4.2 and 4.3, respectively. The closer to 10 the fitness is, the better the multi-robotic organism is formed according to $Ideal$.

$$CohesionFactor = C_B + C_L + C_T \tag{4.5}$$

where C_B is a factor taking into account the number of neighbours T_{body} bots have, C_L is a factor taking into account the number of neighbours T_{limb} bots have and C_T is the average difference between the number of robots of each type τ.

$$F = GeometryFactor - CohesionFactor \tag{4.6}$$

Thus, $GeometryFactor$ (Equation 4.4) takes into account the shape of the creature formed where $CohesionFactor$ (Equation 4.5) looks into more defined factors. Note that the closer to 0 $CohesionFactor$ is, the better, hence the subtraction from

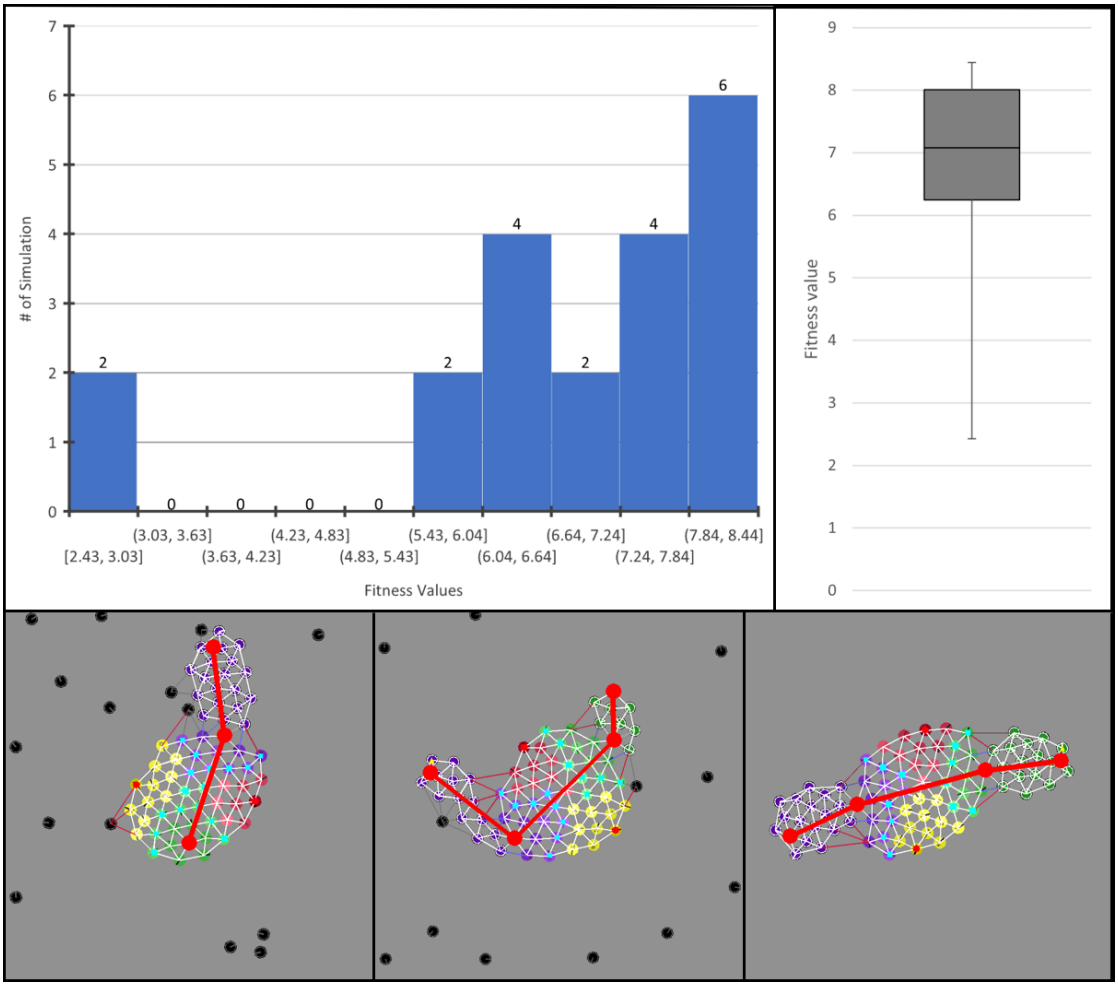

Figure 4.6: **Stability evaluation** Top: Barchart and Boxplot compiling results for 20 simulation runs using parameter values from Table 4.1. Bottom: From left to right, end of simulation's logs visualisation. Robots colour code identical as Fig. 4.4. Red dots and lines represent the points and vectors used to calculate the fitness. *Left* $F = 2.43$. Here, the τ_s source ended up within the body and not at the periphery of the body, preventing the τ_s limb to start its growth hence forming a misshaped organism; *Middle* $F = 5.51$. Here both τ_n and τ_s limbs successfully formed but grown close to the body and did not expend outward, thus forming a crooked organism; *Right* $F = 8.44$. Here, τ_n and τ_s limbs formed and grew straight out of the body, forming a nicely shaped multi-robotic organism. Full results available at `tinyurl.com/DevobotSuppMat`

.

$GeometryFactor$ when calculating the fitness F (Equation 4.6). Figure 4.6 presents the results for our first experiment. As the bar chart shows, most of the results have a

fitness value comprised within $(6.04, 8.44]$. Moreover, the box plot indicates that more than 50% of the results are included in $[6.24, 8.01]$.
These results are encouraging, despite the low stability in higher fitness values, we can see that our model yields correct results and misshaped organism can be explained (see Figure 4.6). The lowest fitness values are due to a limb not growing in one region. Indeed, in some cases, the super-gradient source of one region end up enclosed in the body and not at its edge. This prevents the limb associated to the region to grow. This occurs most likely because the super-gradient propagation during Phase 1 was not ideal and a plateau was hit by the gradient during the propagation, preventing the source to move to an edge robot. To prevent this issue, increasing the number of robots in the body limits the risk of creating such plateaus within the gradient values.

For our second experiment, we decided to vary D_{body} and D_{limb} in order to study their impact on our model. Both these parameters dictate the distance at which the Delaunay triangulation will be trimmed, thus influencing the neighbourhood of robots. Behind the scene, these parameters are calculated with $D_{\mathrm{m}} \times f$, where D_{m} is the average distance between N_{BotBody} robots within an arena of size $A_{\mathrm{h}} \times A_{\mathrm{w}}$ and f is a multiplying factor. For this experiment, we used values for $f = [1, 2, 3, 4]$. We chose these values to cover different cases: with a value of 1, robots can detect one or two other robots as neighbours and we wanted to compare this case with the other extreme, 4, where robots can detect neighbours at great distances. A value of f lower than 1 would prevent the robots to form a proper connected component, hence preventing any communications and capability of forming the multi-robotic organism, and a value of f greater than 4 would result in a near-complete Delaunay triangulation, defeating the purpose of trimming certain connections. The chosen f result in value for

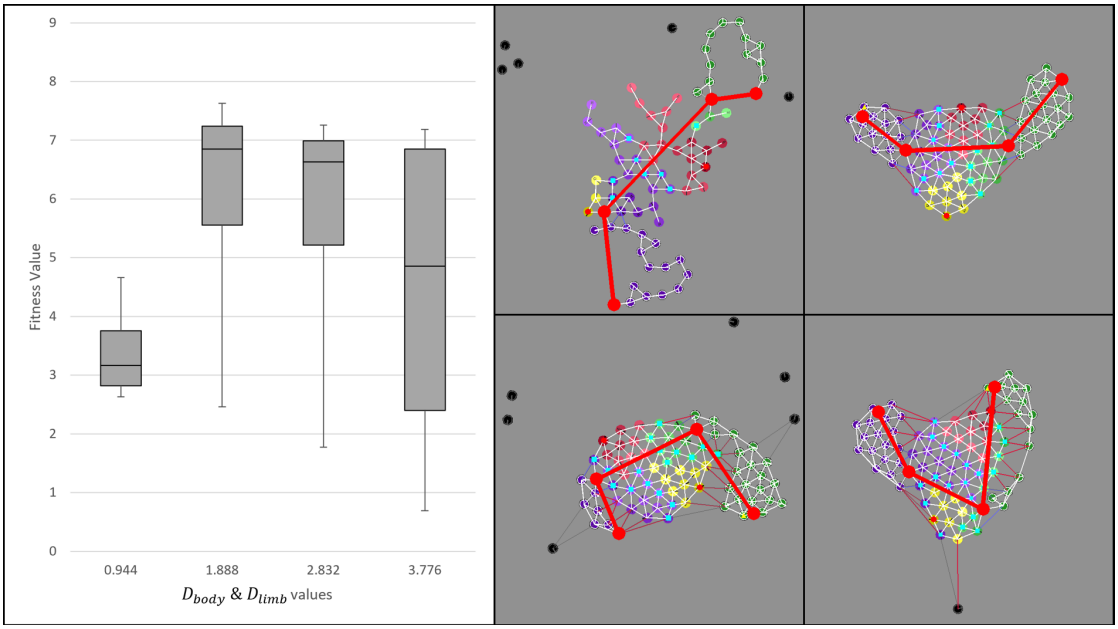

Figure 4.7: **Parameter exploration** Left: Boxplots compiling the results of the parameter exploration. Each parameter value (x axis) compile 4 experiments. Right: From top-left to bottom-right, end of simulation's logs visualisation. Robots colour code identical as Fig. 4.4. Red dots and lines represent the points and vectors used to calculate the fitness. *Top-left* $D_{body} = D_{limb} = 0.944$ and $F = 3.46$; *Top-right* $D_{body} = D_{limb} = 1.888$ and $F = 7.11$; *Bottom-left* $D_{body} = D_{limb} = 2.832$ and $F = 6.36$; *Bottom-right* $D_{body} = D_{limb} = 3.776$ and $F = 6.74$. The boxplots show that values $D_{body} = D_{limb} = 1.888$ and 2.832 yield higher quality results than lower values of D_{body} & D_{limb}, whereas higher values demonstrate less stability in the results. Full results available at `tinyurl.com/DevobotSuppMat`

$D_{body} = D_{limb} = [0.944, 1.888, 2.832, 3.776]$ in MU. Figure 4.7 presents results for 4 runs with each value of f, for a total of 16 runs. As shown on the right part of Figure 4.7, $f = 1$ result in deconstructed organism, misshaped and scattered, whereas $f = 4$ force the creature's limbs to form around the body resulting in a crooked organism. $f = 2$ and $f = 3$ seems to be the sweet spot for these parameters, and further, more refined experiments are needed to optimise their values. Concerning the body formation, low values of f influence its formation, but with $f = [2, 3, 4]$, issues with the creature formation always come from the limb formation and the body manage to grow unimpeded.

4.3 Physical experiments

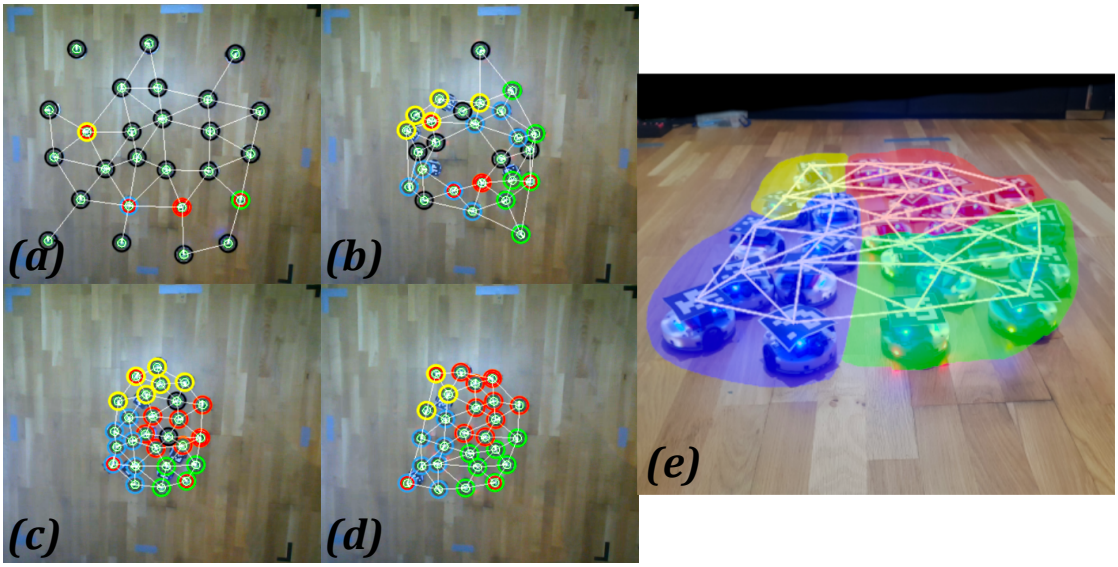

Figure 4.8: **Body formation (phase 1) in a flock of wheeled robots.** (a-d) 26 PsiSwarms robots execute the algorithms described in Section 4.1 inside a 170×180cm arena. Top views from the ceiling camera at times $t = 0, 30, 151, 314$s. Neighbourhood (thin white) are automatically visualised by ARDebug in real time. Red inner circle indicate the robot is a super-gradient source. Coloured outer circle's colour indicate its type τ, while black outer circle indicate a neutral robot. (a) At initialisation, robots are randomly scattered throughout the arena and 4 robots are chosen as initial source for super-gradients. (b-c) Search behaviour and spring forces bring robots closer, while the super-gradients start propagating within the swarm. (d) The body continued its formation until it was complete and stabilised (robots stopped moving) under the attractive/repulsive forces, with parameters $D_{body} = 25$cm, $L_{\text{attr}} = 11$cm, $k_{\text{attr}} = 0.6$. (e) Same final state in perspective view. Coloured areas indicate robots' type τ and red dots indicate super-gradient sources robot. Videos of experiment available at `tinyurl.com/DevobotSuppMat`

The PsiSwarm platform, a disc-shaped robot on wheels, was designed by James Hilder and Jon Timmis at the York Robotics Lab[2]. To centrally monitor the PsiSwarms in real time, whether to read out their trajectories or intervene in the experiment, we relied on the ARDebug softwareMillard et al. 2018, an augmented-reality tool that can track the robots with ArUco square markers pasted on top of them Garrido-Jurado,

[2]https://www.york.ac.uk/robot-lab/psiswarm/

Munoz-Salinas, et al. 2014; Garrido-Jurado, Muñoz-Salinas, et al. 2015; Romero-Ramirez, Muñoz-Salinas, and Medina-Carnicer 2018 (each one carrying a binary pixel matrix that encodes a unique ID number), and can exchange information with them via Bluetooth. PsiSwarms, ARDebug and how they are used in our physical experiments are described in details in Section 3.2.

Toward our goal of shape formation, we faced three technical issues: (1) the IR sensors are not powerful enough to detect the positions of neighbouring robots beyond a few cm; (2) PsiSwarms are not equipped to communicate locally with each other; (3) they also lack a turret camera to spot the flock from afar. To compensate for these shortcomings, we had to go against the principle of decentralisation by resorting to a centralised method for detection & communication. Thanks to the ceiling camera and ArUco markers, (1) the Delaunay neighbourhoods were computed centrally by ARDebug and fed back to the robots (in the form of relative polar coordinates); (2) this information also served for ARDebug to broker peer-to-peer requests via the Bluetooth links; and (3) stray robots received from ARDebug the direction back toward the group by providing a set of polar coordinates to the centre of mass of the flock.

On the other hand, we also made sure to keep the intervention of ARDebug to a minimum, i.e. only provide the robots with the raw, low-level information strictly from their surroundings, that they could have otherwise gathered by themselves with more hardware. In no instance was ARDebug actually controlling the robots and telling them what messages to send and how to move; these calculations and decisions were made by each of them. Based on the polar coordinates of its neighbours (obtained from its fictive detectors via ARDebug), and its internal table of structural links and graph connections,

each robot could compute its total vectorial force and next move. Three resulting shape formation are shown in Fig. 4.8.

4.4 Discussion

In conclusion, we proposed a morphogenetic engineering model and a demonstration of self-organised multi-robotic organism formation among small identical wheeled robots, based on local neighbourhood perception and communication only. We showed that it was possible to implement an abstract model of morphogenetic multi-robotic creature growth in simulation and partly in physical experiments.

The technical problems encountered in the experiments were essentially due to limitations in the PsiSwarm's capabilities, but other issues were involved. These issues are discussed further in Section 6.1, alongside the workaround used in this work and possible solutions. In Section 6.2, we investigate the future directions this project could take.

In the next Chapter, we will present a parallel study to DevoBot, *Branched Structure Formation in a Decentralised Flock of Wheeled Robots*, where our aim shift from forming a multi-robotic organism with a flock of robots to a specific and flexible chain-like structure formation. This work was published at the *Twelfth International Conference on Swarm Intelligence* ANTS2020[3] (Gaget, Montanier, and René Doursat 2020).

[3]https://www.iiia.csic.es/ants2020/

Chapter 5

Branched Structure Formation in a Decentralised Flock of Wheeled Robots

In this contribution we focus on the process of *morphogenesis* per se, i.e. the programmable and reliable bottom-up emergence of *shapes* at a higher level of organisation. This is based on a paper published at the Twelfth International Conference on Swarm Intelligence ANTS2020[1] (Gaget, Montanier, and René Doursat 2020). We show that simple abstract rules of behaviour executed by each agent (their "genotype"), involving message passing, virtual link creation, and force-based motion, are sufficient to generate various *reproducible and scalable* multi-agent branched structures (the "phenotypes"). On this basis, we propose a model of collective robot dynamics based on "morphogenetic engineering" principles, in particular an algorithm of programmable network growth, and how it allows a flock of self-propelled wheeled robots on the ground to coordinate and function together. The model is implemented in simulation and demonstrated in physical experiments with the PsiSwarm platform.

[1] https://www.iiia.csic.es/ants2020/

Our focus is the following: we want to show here that simple abstract rules of behaviour executed by each agent (their "genotype"), involving message passing, link creation, and force-based motion, are sufficient to generate various *reproducible and scalable* multi-robot structures (the "phenotypes") by aggregation. Ideally, agent rules are independent of its physical embodiment—but of course we also present a proof of concept using real robots.

In summary, between swarm and modular robotics, the goal of the present work is to create flexible, yet at the same time highly specific spatial formations within a larger group of small wheeled robots, based on Morphogenetic Engineering (ME) principles. The field of ME (René Doursat, Sánchez, et al. 2012; René Doursat, Sayama, and Michel 2013) investigates the subclass of complex systems that self-assemble into nontrivial and reproducible structures, such as multicellular organisms built by cells, or the nests built by colonies of social insects. These natural examples can serve as a source of inspiration for the meta-design of self-organising artificial and techno-social systems. In particular, we will follow here Doursat's abstract ME algorithm of "programmable network growth" (René Doursat and Ulieru 2008)—which was later modified and hypothetically applied to the autonomous deployment of emergency response teams forming chains of agents using IoT devices (Toussaint, Norling, and René Doursat 2019).

The remainder of this Chapter is organised as follows: In Section 5.1, we describe the abstract model of collective robot dynamics based on ME principles applied to network growth. We present the underlying mechanisms allowing a small swarm of self-propelled robots on the ground to coordinate and function together to create non-physical chains within a swarm of robots. Then, we show how the model unfolds in

simulation (Section 5.2.1) and physical experiments (Section 5.2.2) to generate different structures.

5.1 Model

The "meta-design" methodology of this project consists of hand-made rules programmed in all agents to foster the development of a multi-robot structure. Given different rules, robots are able to form different target shapes by making local decisions based on what they detect and exchange with their neighbours.

The definition and computation of each agent's neighbourhood is based on the description present in Section 3.1 (Figure 3.1).

Network Components: Ports, Links and Gradients. The morphogenetic core of the model is derived from Doursat's original algorithm of programmable network growth René Doursat and Ulieru 2008. It involves input/output *ports* on the nodes, *links* between nodes (on top of their neighbourhood connections), and *gradients* (integer values) sent and received by the nodes over the links through the ports. All agents are endowed with the same set of pairs of input/output ports, denoted by $(X_{\text{in}}, X_{\text{out}})$, $(Y_{\text{in}}, Y_{\text{out}})$, etc. A port can be in one of three states: "open", where it accepts (in input) or creates (in output) links with neighbours; "busy", where it is already linked to a maximum number of agents (generally one) and cannot accept or create new links; and "closed", where it is disabled and devoid of links. An open input port on agent i can accept link requests originating only from its mirror output port located on a neighbouring agent j, for example: $X^i_{\text{in}} \leftarrow X^j_{\text{out}}$ (but not $X^i_{\text{in}} \leftarrow Y^j_{\text{out}}$ or $X^i_{\text{in}} \leftarrow X^j_{\text{in}}$).

Each type of pair of ports is associated with a gradient field across the network, composed of integer values representing important positional information about the nodes relative to each other within the topology (essentially their hop distance), and denoted by x_g^i, y_g^i, etc. When a link $i \leftarrow j$ is created between two agents, the gradient associated with the ports is propagated through this link from j to i via an increment, i.e. $x_g^i = x_g^j + 1$ if it concerns the X ports. Then both agents switch the corresponding ports to the busy state.

In the context of collective robotics, this abstract port-link-gradient framework translates into the self-organisation of branched structures made of chains of robots (Figures 5.1 and 5.2). These structures are a subset of the background communication mesh described above. Therefore, at every time step each agent may have two types of neighbours: ones that are simply within signal range, or "connected" (thin black edges), and ones that are formally and durably "linked" to it (thick green edges)—albeit not physically for lack of hooks or magnets.

Within a static connectivity graph, network growth proceeds by peer-to-peer recruitment and aggregation of agents as follows: if agent j is already part of the growing structure and has an open output port X_{out}^j, it will look if one of its neighbours i has a corresponding open input port X_{in}^i (i.e. is not yet in the structure) to request a link creation—which it does by sending requests to each neighbour in turn.

The specifics of the growth process (which ports to open or update, how many links to create in a chain, etc.) are prescribed by an identical set of rules, or "genome", executed by each agent. The genome dictates how an agent should behave, i.e. the local decisions it should make at every time step, which will vary depending on its current neighbourhood configuration and the gradient values it carries. In essence, a genomic ruleset is composed of a list of *condition→action* clauses, where conditions are based

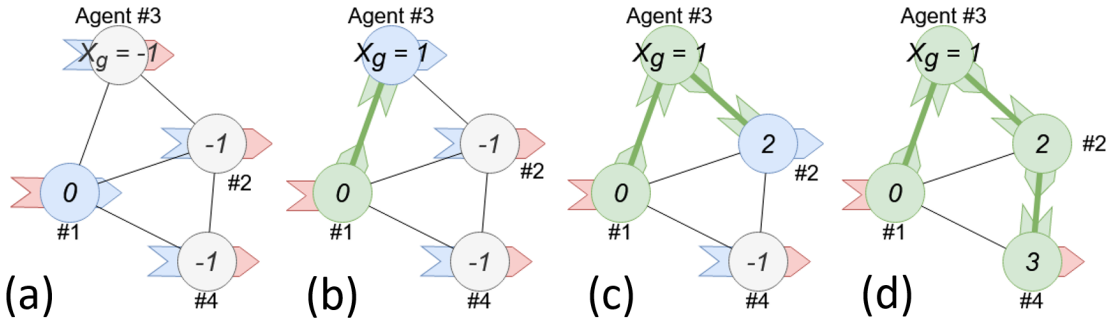

Figure 5.1: **Simple chain formation among static agents.** Each agent executes Alg. 1 (non-bracketed parts) with $x_N = 4$. Ports are symbolised by thick arrows (inputs as tails, outputs as heads) and colour-coded states: open in blue, busy in green, closed in red. Nodes are also coloured: recruiting in blue, accepting in grey, integrated in green. Edges can be of two types: neighbourhood connections in black, structural links in green. (a) Initial state: gradient x_g^1 is set to 0 in a seed Agent 1 and undefined everywhere else. (b) Agents 1 and 3 agree on creating a chain link and propagate the gradient, i.e. $x_g^3 = 1$. (c,d) The chain continues to grow, with Agent 3 recruiting 2, and 2 recruiting 4 in turn. This results in $x_g^2 = 3$, which reaches the given threshold x_N (the maximum length) prescribed in the genome, therefore shuts the output port X_{out}^2 and ends the chain.

on gradients and port states, and actions update the ports. Examples of genomes and structures developed from them are shown in the next section.

In the beginning, agents are scattered at random across the arena. One agent is arbitrarily chosen to be the seed of the structure and is initialised differently from the others. Typically its input port is closed, its output port open, and its gradient value set to 0. Conversely, all other agents start with open inputs, closed outputs, and undefined gradients at -1. Then, each agent repeatedly executes four main steps in a loop: (a) port states are changed according to the genomic rules; (b) links are created where possible; (c) gradient values are propagated and updated; and (d) the robot moves by applying spring forces and/or a search behaviour. The latter step is explained below in the section about mobile network growth.

$x_N =$ prescribed chain length
$y_N =$ branch length
if $t = 0$ **then**
  if *is seed* **then** {close X_{in}, open X_{out}; $x_g = 0$; close $Y_{\text{in}}, Y_{\text{out}}$; $y_g = -1$}
  else {open X_{in}, close X_{out}; $x_g = -1$; open Y_{in}, close Y_{out}; $y_g = -1$}
  return
if $x_g = x_N - 1$ **then** close X_{out}
else if $x_g \geq 0$ *and* X_{out} *is closed* **then** open X_{out}
if x_g *is odd* **then** {open Y_{out}; $y_g = 0$}
if $[[y_g = y_N - 1$ **then** close $Y_{\text{out}}]]$

Algorithm 1: Genome of a simple chain/branched structure growth

Examples of Genomes and Structures. In this paragraph, we give two examples of abstract network growth among static agents on top of their background communication graph, omitting spring forces and motion. The first system involves four agents forming a simple chain based on one pair of X ports (Figure 1). The genome is described in Algorithm 1 (non-coloured parts only), where x_N is set to 4. As explained above, at first ($t = 0$) the unique seed agent is initialised differently from the other agents. Then, as soon as an agent is recruited into the structure, its gradient x_g becomes positive by propagation and triggers the opening of the output port X_{out}, unless $x_g = x_N - 1$, which means that it found itself at the end of the chain and should close X_{out}.

The second example shows a slightly more complicated branched structure, or "creature" composed of a "body" chain of five agents and two short "leg" chains of two agents each, sprouting from the even-positioned body agents (Figure 5.2). Ports X are used to form the body, while different ports Y support the legs. The genomic rules are described in Algorithm 1 (coloured parts included), with $x_N = 5$ and $y_N = 3$. Compared to the previous example, the added complication consists of managing ports Y and their associated gradient y_g depending on certain values of x_g. Here, if $x_g = 1$ or 3 it means

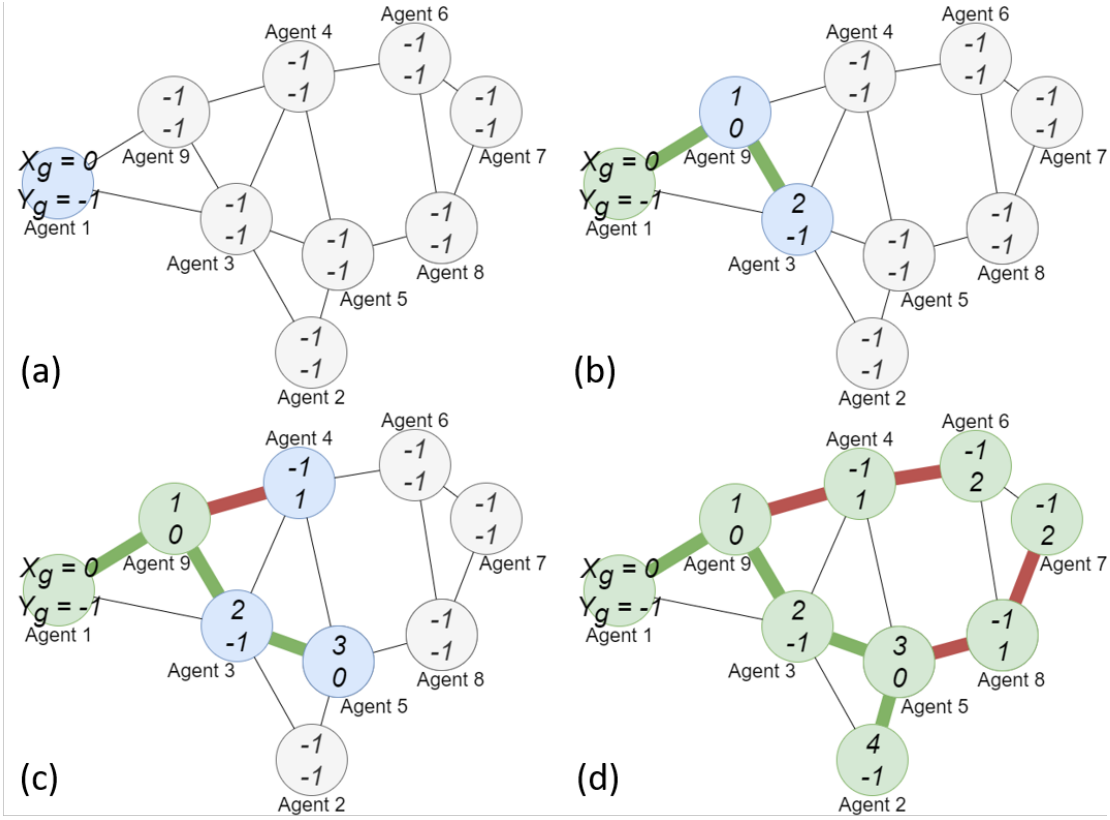

Figure 5.2: **Branched structure formation among static agents.** Each agent possesses two pairs of ports, X and Y (not represented), and executes Algorithm 1 with $x_N = 5$, $y_N = 2$. Colour coding is the same as Figure 5.1, with red lines symbolising Y links (leg branches). (a) Initial state with agent 1 as seed, having an output X port set to 0 (start of the chain); (b) Agent 1 recruits Agent 9 since the latter had a free input port on its pair of port X. Agent 1 thus pass to green because all its ports are either closed or busy. Agent 9 then create a new link on its pair of port X with agent 3, but remains blue because it still can create a link on its pair of port Y; (c) Agent 9 recruit agent 4 on its pair of port Y, hence creating a red link and a sub-chain in the structure. Additionally, agent 3 continue to recruit into the main chain (green links); (d) Finished structure, where all agents have satisfied the rules and no more recruitment is attempted.

that the agent is second or fourth along the main chain, therefore it should open its other output port Y_{out} and set $y_g = 0$ to start a branch by recruiting free agents via Y_{in}. For branch termination, the same condition is used in Y, i.e. closing Y_{out} when $y_g = y_N - 1$.

Mobile Network Growth in Space. In reality, as robots move around, the background mesh is not static but continually updated (as per Figure 3.1a) so that new connections may appear and existing ones disappear. In spite of this, already created structural links will persist: if communication between linked robots is accidentally interrupted, they keep tabs on each other and resume regular gradient exchange whenever possible. This should rarely happen, however, as elastic forces tend to keep them close to each other, as if physically attached.

To maximise matching opportunities, agents not yet recruited navigate toward, and stay close to, the existing structure. If an agent finds itself isolated far away without neighbourhood connections, it uses the camera to search for the bulk of the flock and head over there. When its front proximity sensors detect a close obstacle, then two scenarios can happen: (α) in *simulation* (Section 5.2.1), it initiates a clockwise exploration behavior by turning left and keeping its right-side sensors active until it receives a link request; (β) in the *physical experiments* (Section 5.2.2), it just sticks near the first encountered neighbor(s) by applying default elastic forces. In this last case, an added condition is to receive a "connected-component" flag propagated from the seed agent over the graph connections: if it does not get it, then it moves again toward the flock's centre.

To be more precise, different types of springs are used or not depending on the local state of neighbouring nodes. Three cases can be distinguished: (i) if both nodes are integrated into the structure and linked to each other, then a strong attractive elastic force is applied between them with a coefficient k_{att} and a length L_{att} significantly smaller than the cutoff communication distance D to keep them close; (ii) if both nodes belong to the structure but are not directly linked (yet spatially close, e.g. if the chain is folded), then a weak repulsive force is used to pull them apart, with a coefficient k_{rep} and a length L_{rep}

greater than or about equal to D; (iii) if one or both nodes are outside of the structure, then two variants happen: (iii.γ) in *simulation*, no spring force is applied and the free agents rely on proximity sensing for their search behaviour (the linked agents ignore them when calculating their forces); (iii.δ) in the *physical experiments*, the repulsion force $L_{\text{rep}}, k_{\text{rep}}$ is used to keep them at an optimal distance.

Altogether, this combination of attractive and repulsive elastic forces leads the robot flock to form a tight chain-like structure visible to the naked eye (although without physical links) while at the same time making this structure unfold in space.

5.2 Results

In this section, we present our result obtained both in simulation and in real physical experiments.

5.2.1 Simulations

Before trying our model with real robots (see Section 5.2.2.*Physical Experiments*), we implemented it in a realistic simulation, as described in Section 3.3. Within the agent controller (Figure 3.5) of the simulation, we implement the genome of Algorithm 1. Each agent have access to an internal table listing its own neighbours and their last known states. This table allows the agent to send a regular update of its state to its neighbours, as well as send messages to ask for a recruitment if it's needed. This table is updated via the Delaunay computation unit that calculate the trimmed Delaunay for the flock. Additionally, agents keep track of their current links and their types in order to apply the correct spring forces.

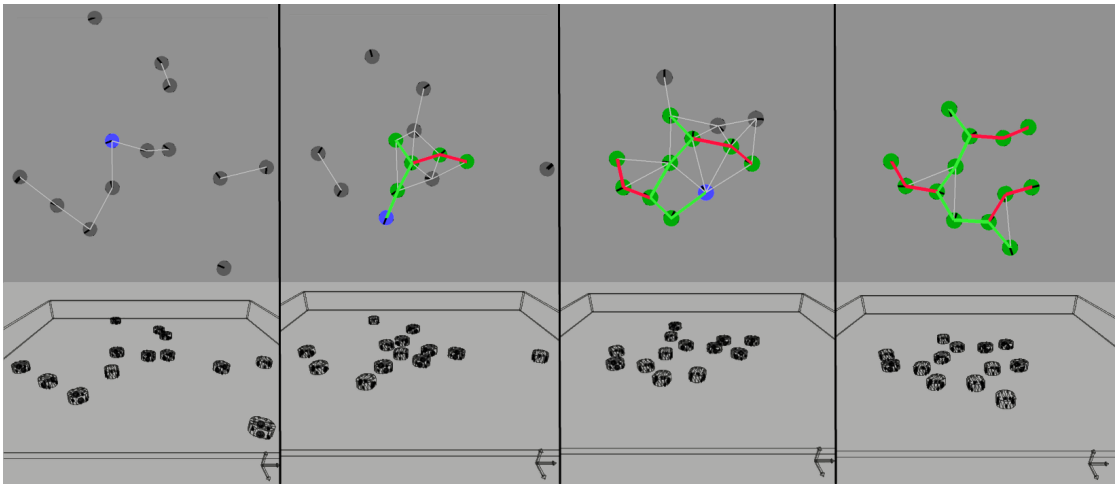

Figure 5.3: **Branched structure formation in simulation.** Bottom: screenshots of the MORSE display at time steps $t = 0, 13, 94, 385$. Top: custom 2D visualisation tool based on log files at same time steps with colour code of Figure 5.2. Each virtual robot executes Algorithm 1 with $x_N = 7, y_N = 3$. The 13 robots self-organise into a 7-robot chain body with three 2-robot legs at odd-numbered positions. This network structure also unfolds in space under the influence of the spring forces with parameters $D = 3.56d, L_{\text{att}} = 1.8d, k_{\text{att}} = 1, L_{\text{rep}} = 5.4d, k_{\text{rep}} = 0.5$, where d is a robot's diameter and $d = 0.5$ MORSE unit. The simulation stops at $t = 427$ when robots cannot form new links and elastic forces have reached equilibrium. Videos available at `https://tinyurl.com/gaget20`.

The simulated experiment shown here is a flock of 13 robots forming a branched structure based on the complete genome of Algorithm 1 (Figure 5.2). In addition to the networking rules, spatial motion relied on the exploration behaviour and the spring forces as explained above at the end of Section 5.1.*Mobile Network Growth in Space* in items (α) and (i-iii.γ) with the parameter values specified in the caption.

5.2.2 Physical Experiments

The PsiSwarm platform, a disc-shaped robot on wheels, and the ARDebug controller were both described in details in Section 3.3 and details about the setup and limitation of the PsiSwarms can be found in Section 4.3.

We ran 3 experiments with different genomes, showed in Fig. 5.4-f, g and h. No repetition or statistical analysis have been conducted so far, but such improvement and stability proofs are discussed in Section 6.2.

Once again, in no instance was ARDebug actually controlling the robots and telling them what messages to send and how to move; these calculations and decisions were made by each of them. Based on the relative positions of its neighbours (obtained from its fictive detectors via ARDebug), and its internal table of structural links and graph connections, each robot could compute its total vectorial force and next move, as per items (i-iii.δ) above—or head for the flock and apply protocol (β) if it was stranded far away. Three resulting formations are shown in Fig. 5.4a-f.

5.3 Discussion

In conclusion, we proposed a morphogenetic engineering model and a demonstration of self-organised branched structure formation among small identical wheeled robots, based on local neighbourhood perception and communication only. We showed encouraging results of the implementation of an abstract model of programmable network growth both in simulation and physical experiments, which demands further study to validate the stability and scalability of our model.

The technical problems encountered in the experiments were essentially due to limitations in the PsiSwarm's capabilities. Its lack of hardware for mid-range peer-to-peer detection & communication, and flock recognition, had to be remedied by the central monitoring system ARDebug, which tracked robots and brokered information and message-passing among them. ARDebug's role, however, remained minimal in the sense that it only emulated the neighbourhood data that would otherwise be handled by

extra sensors and emitters, while the core computation and decision-making modules remained on board. These issues are discussed further in Section 6.1.

The experiments presented so far in this work are encouraging, although at this point they only constitute a proof of concept. To complete this study, an extended statistical analysis over many trials, whether exploring different genotypes or variable random conditions on the same genotype, should be conducted to adjust parameters and establish the resilience of the model in real-world settings. In addition, simulations and experiments with more robots must be conducted to insure the scalability of our model. Further future work are explored in Chapter 6.2.

In this Chapter, we took inspiration from artificial network growth and showed the self-organisation of branched structure formations, based on a single genome distributed among a flock of small and identical robots, both in simulation and in physical experiments. In the next and final part of this thesis, Chapter 6, we will summarise our work, expose its flaws, limitations and obstacles we encountered and propose possible solutions, and discuss possible direction this work could take in the future.

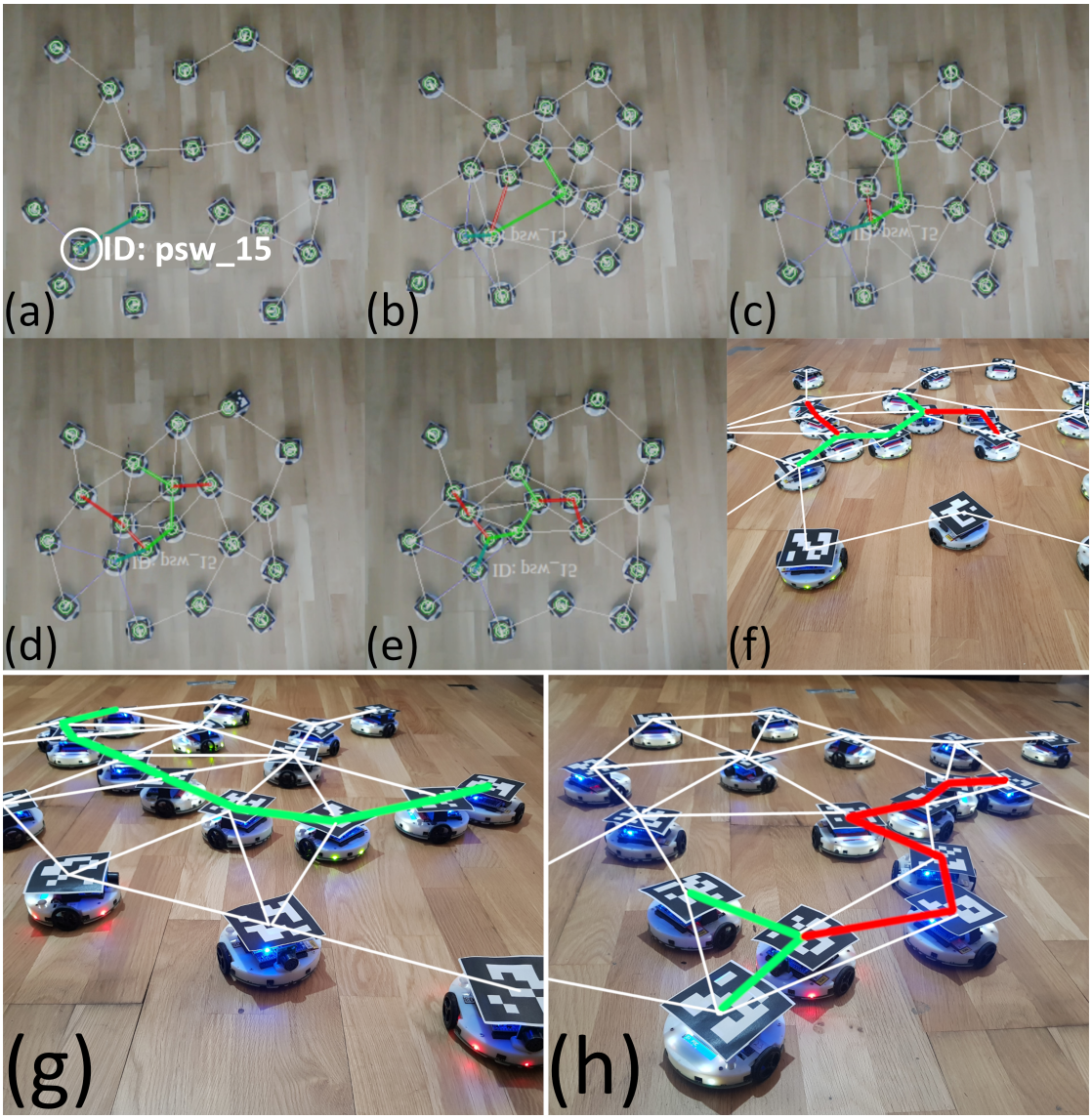

Figure 5.4: **Formation of linked structures in a flock of wheeled robots.** (a-f) 20 PsiSwarm robots execute Algorithm 1-branched structure growth (coloured parts included) with $x_N = 5, y_N = 3$ inside a 170×180cm arena. (a-e) Top views from the ceiling camera at times $t = 4, 13, 26, 45, 100$s. Structural links (thick green & red) and graph connections (thin white) are automatically visualised by ARDebug in real time. (a) Shortly after initialisation, the seed robot *psw_15* created a first link with a neighbour. (b) Search behaviour and spring forces bring robots closer, while two more body links (green) and one leg link (red) appeared. (c-e) The branched structure continued growing until it was complete and stabilised (robots stopped moving) under the attractive/repulsive forces, with parameters $D = 43.2$cm, $L_{\text{att}} = 11$cm, $k_{\text{att}} = 0.6$, $L_{\text{rep}} = 32.4$cm, $k_{\text{rep}} = 0.42$. (f) Same final state in perspective view. (g,h) Other examples of "phenotypes" based on Algorithm 1-branched structure growth: (g) a simple 9-robot chain and (h) a 3+6-robot T-shape. Videos of all three experiments available at `https://tinyurl.com/gaget20`.

Chapter 6

Conclusion and Future Work

In this Chapter, we first summarise and review the work presented in this thesis (Section 6.1). We give a quick summary of both contributions, then expose the limitations and obstacles we encountered and discuss possible solutions. We also shed light on potential future work that could be conducted within the DevoBot project, and explore possible applications both in the near future and from an open-ended perspective.

6.1 Conclusion and discussion

The research undertaken during this thesis aimed at answering the following question: *Can we meta-design and implement a model of multi-robotic system able to self-organise into a "creature" or a structure following Morphogenetic Engineering principles?* In practical terms, our goal was to design and implement multi-agent models of collective shape-formation applicable to flocks of robots, i.e. contributing to the field of Morphogenetic Robotics. To attempt answering this question, we designed two models: one for the growth of a "multi-robotic organism", formed of a body and two limbs

emerging from a flock of identical robots; and another one for the self-assembly of branched chain-like structures among the same group of robots.

For our main contribution to the DevoBot Project, we designed a multi-agent model for a flock of robots developing into a multi-robotic organism composed of a body and two limbs. Using several gradients (numerical values representing relative-position information) propagating throughout the flock, and simulated spring forces for trajectory computation, we achieved the full growth of a multi-robotic body separated into four regions (North, South, East and West in the model we presented). This multi-robotic body growth was achieved both in simulation and in physical experiments using PsiSwarms, a group of small two-wheeled robots. Once this body was grown, some of its regions differentiated to allow the propagation of "sub-gradients", allowing the recruitment of free robots roaming around the body to incorporate into the limbs. Therefore, the limbs developed via incremental aggregation of robots on "attractive" hot spots. The limb-growing phase, also referred to as phase 2, however, was only realised in a simulation environment. During this study, we also conducted a few statistical analyses on our simulations, producing encouraging results on the stability of our model and exploring the impact of some parameters on the unfolding of the simulation and the results obtained. Concerning the impact of parameters, however, the fitness function used to evaluate the results was not ideal, as it was based on a geometry factor so that modifying D_{limb} and D_{body} would deteriorate the fitness value. Therefore, the calculation was biased toward the specific parameter values used in our first experiment. A more generic fitness should be used for parameter exploration in order to better classify the results.

For our second contribution, we designed a multi-agent model of robot self-assembly producing branched chain-like structures. We were inspired by Doursat's "programmable network growth", and implemented it into a robotic system. All robots were equipped

with virtual input and output ports, allowing them to establish connections among each other if a pair of matching input and output ports was available. When these connections were established, links were formed and through these links, spring forces were applied and robots organised themselves into graph-like structures made of nodes and links. The rules dictating the formation of links and application of spring forces were encoded by a "genome" present in each robot of the system, supporting a complete decentralisation of the computation.

We encountered a number of obstacles due to limitations in the PsiSwarm's capabilities. The lack of hardware for mid-range peer-to-peer detection and communication, and flock recognition, had to be remedied by the central monitoring system ARDebug, an existing tracking software for the PsiSwarm that we modified and tailored to our need. ARDebug tracked robots and brokered information among them by message-passing. Its role, however, remained minimal in the sense that it only emulated the neighbourhood data that would otherwise be handled by extra sensors and emitters, while the core computation and decision-making modules remained on board the PsiSwarms.

Our models also presented one algorithmic problem, in addition to the PsiSwarm's physical constraints: the "trimmed Delaunay triangulation" used for neighbourhood computation required a bird's eye view of the entire flock, i.e. a certain degree of centralisation. If a pure local neighbourhood computation was used instead, one based on distance only for example, links between neighbours could cross, whereas within our trimmed Delaunay computation this could not happen. Having such crossing links within the neighbourhood of the flock could result in erratic gradient propagation behaviours, and an overall review of the gradient systems we used would be necessary.

Concerning the first study on the growth of a "multi-robotic organism", the low number of robots used in physical experiments makes the different body areas unequal

in size, giving rise to unbalanced shapes. In simulation, the possibility of separating link properties using "hook" functions, such as ones that apply the spring forces do not propagate gradient information, adds a layer of complexity to the model that could be removed by using a significantly larger number of robots. Finally, inward G^R_{inward} and outward G^R_{outward} gradients used in Phase 2 (limb formation) allow only the formation of simple protrusions with no complex structure and limited possibilities for expansion. In order to grow more complex limbs and reach intricate shapes and structures, a more complex gradient system should be established, but it would increase the complexity of the computation and put more strain on the limited hardware of the PsiSwarm platform. Moreover, in-depth statistical analyses also need to be conducted to further prove the stability of our model because the present analysis was limited to 20 runs. The same improvements can be made in the parametric search, by including more parameters and using a more appropriate method of selection of their values.

Regarding our work on the self-organisation of graph structures, as stated in Section 5.3, further statistical analyses must also be conducted to prove the stability and scalability of our model.

6.2 Future perspective and applications

This thesis presented two main contributions to the field of Morphogenetic Robotics: the controlled growth of a "multi-robotic organism" and the self-assembly of chain-based structures among a flock of identical and autonomous robots.

To extend our work on chain-like structures, more complex branched chains or loops involving other port types and a larger swarm could be attempted. The purpose of such experiments would be to validate the scalability of our model of branch formation in

robotic swarms. Last but not least, the loose flocking structures thus created should demonstrate their usefulness by moving across space and behaving as single cohesive "creatures". Even without physical attachments, the robots should be able, for example, to encircle and push bulky objects—or interact in any other way with their environment via specialisation and division of labour, similarly to multicellular organisms.

The first extension we can imagine for our "multi-robotic organism" is the implementation of the full growth of the organism with the PsiSwarms, hence fully crossing the reality gap and achieving the full growth of a multi-robotic organism similar to a multi-cellular organism. Before going further, one missing feature would need to be implemented: robustness to failures. Indeed, as of now, if one robot within the body or the limbs fails, other robots can ignore it and continue the growth of the multi-robotic organism. However, if a key robot fails, such as a super-gradient source, there is no fallback strategy or self-healing process and the growth will be impeded. A solution would be that all sources' neighbours would know of the source, and if they detect that the source is missing, they can elect a candidate amongst them to take the role of the source. Additionally, a more refined system can be devised for the limb formation, for example by improving the existing gradient system, in order to grow more complex limbs. A creature could use these limbs to achieve different tasks, for instance grabbing and moving an object like a ball. Within the same scope, the creature should be able to move as a whole in different directions and keep the same overall structure. This should enable the completion of more complex tasks, like grabbing a ball and bringing it to a goal location.

In both studies, evolutionary computation should be used to optimise several parts of our models. For instance, the genomes of our branched structures could evolve via a genetic algorithm to better adapt the structure to solving specific tasks. Such algorithms

should also be used for parametric exploration, and generally to finely tune the models.

If we let our imagination run a little wild, we could imagine a large swarm of mini-robots with strong communication and sensing capabilities, using gradient based algorithms and developmental programming to self-organise into very complex creatures capable of accomplishing complex tasks, like construction work or search and rescue missions, and automatically adapt to any situation without reprogramming.

References

Ahmadzadeh, Hossein, Ellips Masehian, and Masoud Asadpour (2016). “Modular robotic systems: Characteristics and applications”. In: *Journal of Intelligent & Robotic Systems* 81.3-4, pp. 317–357.

Alonso-Mora, Javier, Andreas Breitenmoser, Martin Rufli, Paul Beardsley, et al. (2013). “Optimal reciprocal collision avoidance for multiple non-holonomic robots”. In: *Distributed Autonomous Robotic Systems*. Springer, pp. 203–216.

Alonso-Mora, Javier, Andreas Breitenmoser, Martin Rufli, Roland Siegwart, et al. (2011). “Multi-robot system for artistic pattern formation”. In: *Robotics and Automation (ICRA), 2011 IEEE International Conference on*. IEEE, pp. 4512–4517.

Ball, Philip (2015). “Forging patterns and making waves from biology to geology: a commentary on Turing (1952)‘The chemical basis of morphogenesis’”. In: *Phil. Trans. R. Soc. B* 370.1666, p. 20140218.

Bard, Jonathan BL and Jonathan Bard (1992). *Morphogenesis: the cellular and molecular processes of developmental anatomy*. Vol. 23. Cambridge University Press.

Beckers, Ralph, OE Holland, and Jean-Louis Deneubourg (1994). “From local actions to global tasks: Stigmergy and collective robotics”. In: *Artificial life IV*. Vol. 181, p. 189.

Bhavsar, Parth et al. (2017). “Risk analysis of autonomous vehicles in mixed traffic streams”. In: *Transportation Research Record* 2625.1, pp. 51–61.

Camazine, Scott et al. (2020). *Self-organization in biological systems*. Princeton university press.

Carrillo-Zapata, Daniel et al. (2019). “Toward controllable morphogenesis in large robot swarms”. In: *IEEE Robotics and Automation Letters* 4.4, pp. 3386–3393.

Christensen, Anders Lyhne, Rehan O’Grady, and Marco Dorigo (2008). “SWARMORPH-script: a language for arbitrary morphology generation in self-assembling robots”. In: *Swarm Intelligence* 2.2-4, pp. 143–165.

Davidson, Eric and Michael Levin (2005). “Gene regulatory networks”. In: *Proceedings of the National Academy of Sciences* 102.14, pp. 4935–4935.

Dawkins, Richard (2003). “The evolution of evolvability”. In: *On growth, form and computers*, pp. 239–255.

Dorigo, Marco and Mauro Birattari (2011). “Ant colony optimization”. In: *Encyclopedia of machine learning*. Springer, pp. 36–39.

Dorigo, Marco and Luca Maria Gambardella (1997). “Ant colony system: a cooperative learning approach to the traveling salesman problem”. In: *IEEE Transactions on evolutionary computation* 1.1, pp. 53–66.

Doursat, René (2010). “The growing canvas of biological development: multiscale pattern generation on an expanding lattice of gene regulatory nets”. In: *Unifying Themes in Complex Systems*. Springer, pp. 205–210.

Doursat, Rene (2011). “Morphogenetic engineering weds bio selforganization to human-designed systems”. In: *PerAda Magazine*.

Doursat, René and Carlos Sánchez (2014). “Growing fine-grained multicellular robots”. In: *Soft Robotics* 1.2, pp. 110–121.

Doursat, René, Carlos Sánchez, et al. (2012). "Embryomorphic engineering: Emergent innovation through evolutionary development". In: *Morphogenetic Engineering*. Springer, pp. 275–311.

Doursat, René, Hiroki Sayama, and Olivier Michel (2012). "Morphogenetic engineering: Reconciling self-organization and architecture". In: *Morphogenetic Engineering*. Springer, pp. 1–24.

— (2013). "A review of morphogenetic engineering". In: *Natural Computing* 12.4, pp. 517–535.

Doursat, René and Mihaela Ulieru (2008). "Emergent engineering for the management of complex situations". In: *Proceedings of the 2nd International Conference on Autonomic Computing and Communication Systems*, pp. 1–10.

Duarte, Miguel et al. (2016). "Application of swarm robotics systems to marine environmental monitoring". In: *OCEANS 2016-Shanghai*. IEEE, pp. 1–8.

Ehang, China (n.d.). http://ehang.com/formation. Accessed: 2018-09-14.

Eiben, Agoston E and Marc Schoenauer (2002). "Evolutionary computing". In: *Information Processing Letters* 82.1, pp. 1–6.

Eiben, Agoston E and James E Smith (2015). *Introduction to evolutionary computing*. Springer.

Erwin, Douglas H and Eric H Davidson (2009). "The evolution of hierarchical gene regulatory networks". In: *Nature Reviews Genetics* 10.2, pp. 141–148.

Escalera, Juan A et al. (2018). "Evo-bots: A simple, stochastic approach to self-assembling artificial organisms". In: *Distributed Autonomous Robotic Systems*. Springer, pp. 373–385.

Feil-Seifer, David and Maja J Mataric (2005). "Defining socially assistive robotics". In: *9th International Conference on Rehabilitation Robotics, 2005. ICORR 2005.* IEEE, pp. 465–468.

Fortune, Steven (1995). "Voronoi diagrams and Delaunay triangulations". In: *Computing in Euclidean geometry*, pp. 225–265.

Friedrich, Bernhard (2016). "The effect of autonomous vehicles on traffic". In: *Autonomous Driving*. Springer, pp. 317–334.

Fu, Daniel Y et al. (2018). "Influencing Flock Formation in Low-Density Settings". In: *arXiv preprint arXiv:1804.08667*.

Gaget, Antoine, Jean-Marc Montanier, and René Doursat (2020). "Branched Structure Formation in a Decentralized Flock of Wheeled Robots". In: *International Conference on Swarm Intelligence*. Springer, pp. 42–54.

Gandhi, Neeraj et al. (2020). "Self-reconfiguration in response to faults in modular aerial systems". In: *IEEE Robotics and Automation Letters* 5.2, pp. 2522–2529.

Garrido-Jurado, Sergio, Rafael Munoz-Salinas, et al. (2014). "Automatic generation and detection of highly reliable fiducial markers under occlusion". In: *Pattern Recognition* 47.6, pp. 2280–2292.

Garrido-Jurado, Sergio, Rafael Muñoz-Salinas, et al. (Oct. 2015). "Generation of fiducial marker dictionaries using Mixed Integer Linear Programming". In: *Pattern Recognition* 51. DOI: 10.1016/j.patcog.2015.09.023.

Gauci, Melvin et al. (2017). "Error cascades in collective behavior: a case study of the gradient algorithm on 1000 physical agents". In: *Proceedings of the 16th Conference on Autonomous Agents and MultiAgent Systems*, pp. 1404–1412.

Giuggioli, Luca et al. (Nov. 2016). "From Ants to Birds: A Novel Bio-Inspired Approach to Online Area Coverage". In: *Proceedings of 13th International Symposium on Distributed Autonomous Robotic Systems*.

Gros, Claudius, Laura Martin, and Bulcsú Sándor (Sept. 2017). "A self-organized one-neuron controller for artificial life on wheels". In: *Proceedings of 14th European Conference on Artificial Life*, pp. 184–185.

Groß, Roderich et al. (2006). "Autonomous self-assembly in swarm-bots". In: *IEEE Transactions on Robotics* 22.6, pp. 1115–1130.

Hajieghrary, Hadi, Dhanushka Kularatne, and M Ani Hsieh (2018). "Differential geometric approach to trajectory planning: Cooperative transport by a team of autonomous marine vehicles". In: *Annual American Control Conference*. IEEE, pp. 858–863.

Hall, Brian K (2003). "Evo-Devo: evolutionary developmental mechanisms." In: *International Journal of Developmental Biology* 47.7-8, pp. 491–495.

Hancock, John T (2003). "The principles of cell signalling". In: *On growth, form and computers*, pp. 64–81.

Hansen, Nikolaus and Andreas Ostermeier (2001). "Completely derandomized self-adaptation in evolution strategies". In: *Evolutionary computation* 9.2, pp. 159–195.

Harrington, Kyle et al. (Sept. 2017). "Competitive Dynamics in Eco-evolutionary Genetically-regulated Swarms". In: *Proceedings of 14th European Conference on Artificial Life*, pp. 190–197.

Helbing, Dirk, Illés Farkas, and Tamas Vicsek (2000). "Simulating dynamical features of escape panic". In: *Nature* 407.6803, p. 487.

Herman, Robert and Keith Gardels (1963). "Vehicular traffic flow". In: *Scientific American* 209.6, pp. 35–43.

Jakobi, Nick, Phil Husbands, and Inman Harvey (1995). "Noise and the reality gap: The use of simulation in evolutionary robotics". In: *European Conference on Artificial Life*. Springer, pp. 704–720.

Jin, Yaochu and Yan Meng (2010). "Morphogenetic robotics: An emerging new field in developmental robotics". In: *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)* 41.2, pp. 145–160.

Jorge, Carolina Centeio and Rosaldo JF Rossetti (2018). "On Social Interactions and the Emergence of Autonomous Vehicles." In: *VEHITS*, pp. 423–430.

Kakalis, Nikolaos MP and Yiannis Ventikos (2008). "Robotic swarm concept for efficient oil spill confrontation". In: *Journal of hazardous materials* 154.1-3, pp. 880–887.

Kaplan, Ken (2016). "500 Drones Light Night Sky to Set Record". In: *Retrieved July* 5, p. 2017.

Kassner, Klaus (1996). *Pattern formation in diffusion-limited crystal growth*. World Scientific.

Kernbach, Serge et al. (2008). "Symbiotic robot organisms: REPLICATOR and SYMBRION projects". In: *Proceedings of the 8th workshop on performance metrics for intelligent systems*, pp. 62–69.

Kerner, Boris S and Sergey L Klenov (2009). "Phase transitions in traffic flow on multilane roads". In: *Physical Review E* 80.5, p. 056101.

Kubera, Yoann, Philippe Mathieu, and Sébastien Picault (2010). "Everything can be Agent!" In.

Ligot, Antoine and Mauro Birattari (2018). "On mimicking the effects of the reality gap with simulation-only experiments". In: *International Conference on Swarm Intelligence*. Springer, pp. 109–122.

Lončar, Ivan et al. (2019). "A Heterogeneous Robotic Swarm for Long-Term Monitoring of Marine Environments". In: *Applied Sciences* 9.7, p. 1388.

Majidi, Carmel (2014). "Soft robotics: a perspective—current trends and prospects for the future". In: *Soft Robotics* 1.1, pp. 5–11.

Malley, Melinda (2020). "Army Ant Inspired Adaptive Self-Assembly with Soft Climbing Robots". PhD Thesis.

Eciton robotica: Design and Algorithms for an Adaptive Self-Assembling Soft Robot Collective (2020).

McKerrow, Phillip John and Phillip McKerrow (1991). *Introduction to robotics*. Addison-Wesley Sydney.

Millard, Alan G et al. (2018). "ARDebug: An augmented reality tool for analysing and debugging swarm robotic systems". In: *Frontiers in Robotics and AI* 5, p. 87.

Miriyev, Aslan, Kenneth Stack, and Hod Lipson (2017). "Soft material for soft actuators". In: *Nature communications* 8.1, pp. 1–8.

Molins, Pere, Namid Stillman, and Sabine Hauert (2019). "Trail formation using large swarms of minimal robots". In: *Cybernetics and Systems* 50.8, pp. 693–710.

Mondada, Francesco et al. (2009). "The e-puck, a robot designed for education in engineering". In: *Proceedings of the 9th conference on autonomous robot systems and competitions*. Vol. 1. IPCB: Instituto Politécnico de Castelo Branco, pp. 59–65.

Murata, Satoshi et al. (2002). "M-TRAN: Self-reconfigurable modular robotic system". In: *IEEE/ASME transactions on mechatronics* 7.4, pp. 431–441.

Nagel, Kai and Maya Paczuski (1995). "Emergent traffic jams". In: *Physical Review E* 51.4, p. 2909.

Nemitz, Markus P et al. (2016). "Using voice coils to actuate modular soft robots: wormbot, an example". In: *Soft robotics* 3.4, pp. 198–204.

Neubert, Jonas and Hod Lipson (2016). "Soldercubes: A self-soldering self-reconfiguring modular robot system". In: *Autonomous Robots* 40.1, pp. 139–158.

O'Grady, Rehan, Anders Lyhne Christensen, and Marco Dorigo (2009). "SWARMORPH: Multirobot morphogenesis using directional self-assembly". In: *IEEE Transactions on Robotics* 25.3, pp. 738–743.

Oh, Hyondong and Yaochu Jin (2014a). "Adaptive swarm robot region coverage using gene regulatory networks". In: *Conference Towards Autonomous Robotic Systems*. Springer, pp. 197–208.

— (2014b). "Evolving hierarchical gene regulatory networks for morphogenetic pattern formation of swarm robots". In: *Evolutionary Computation (CEC), 2014 IEEE Congress on*. IEEE, pp. 776–783.

Oh, Hyondong, Ataollah R Shiraz, and Yaochu Jin (2016). "Morphogen diffusion algorithms for tracking and herding using a swarm of kilobots". In: *Soft Computing*, pp. 1–12.

— (2018). "Morphogen diffusion algorithms for tracking and herding using a swarm of kilobots". In: *Soft Computing* 22.6, pp. 1833–1844.

Olfati-Saber, Reza (2006). "Flocking for multi-agent dynamic systems: Algorithms and theory". In: *IEEE Transactions on automatic control* 51.3, pp. 401–420.

Parrish, Julia K, Steven V Viscido, and Daniel Grunbaum (2002). "Self-organized fish schools: an examination of emergent properties". In: *The biological bulletin* 202.3, pp. 296–305.

Parrott, Christopher, Tony J Dodd, and Roderich Groß (2018). "HyMod: A 3-DOF hybrid mobile and self-reconfigurable modular robot and its extensions". In: *Distributed Autonomous Robotic Systems*. Springer, pp. 401–414.

Pascalie, Jonathan et al. (2016). “Developmental design of synthetic bacterial architectures by morphogenetic engineering”. In: *ACS synthetic biology* 5.8, pp. 842–861.

Peake, Joshua et al. (2018). “Vectorized candidate set selection for parallel ant colony optimization”. In: *Proceedings of the Genetic and Evolutionary Computation Conference Companion*, pp. 1300–1306.

— (2019). “Scaling techniques for parallel ant colony optimization on large problem instances”. In: *Proceedings of the Genetic and Evolutionary Computation Conference*, pp. 47–54.

Reynolds, Craig W (1987). “Flocks, herds and schools: A distributed behavioral model”. In: *ACM SIGGRAPH computer graphics*. Vol. 21. ACM, pp. 25–34.

Rockenbach, Gabriel et al. (2018). “Simulating crowd evacuation: From comfort to panic situations”. In: *Proceedings of the 18th International Conference on Intelligent Virtual Agents*, pp. 295–300.

Romero-Ramirez, Francisco, Rafael Muñoz-Salinas, and Rafael Medina-Carnicer (June 2018). “Speeded Up Detection of Squared Fiducial Markers”. In: *Image and Vision Computing* 76. DOI: 10.1016/j.imavis.2018.05.004.

Rubenstein, Michael, Christian Ahler, and Radhika Nagpal (2012). “Kilobot: A low cost scalable robot system for collective behaviors”. In: *IEEE Conference on Robotics and Automation*, pp. 3293–3298.

Rubenstein, Michael, Alejandro Cornejo, and Radhika Nagpal (2014). “Programmable self-assembly in a thousand-robot swarm”. In: *Science* 345.6198, pp. 795–799.

Shamos, Michael Ian and Dan Hoey (1975). “Closest-point problems”. In: *16th Annual Symposium on Foundations of Computer Science (sfcs 1975)*. IEEE, pp. 151–162.

Shirazi, Ataollah Ramezan, Hyondong Oh, and Yaochu Jin (2014). "Morphogenetic self-organization of collective movement without directional sensing". In: *Conference Towards Autonomous Robotic Systems*. Springer, pp. 139–150.

Shiwakoti, Nirajan and Majid Sarvi (2013). "Understanding pedestrian crowd panic: a review on model organisms approach". In: *Journal of transport geography* 26, pp. 12–17.

Siciliano, Bruno et al. (2010). *Robotics: modelling, planning and control*. Springer Science & Business Media.

Slavkov, I et al. (2018). "Morphogenesis in robot swarms". In: *Science Robotics* 3.25.

Stirling, Timothy, Steffen Wischmann, and Dario Floreano (2010). "Energy-efficient indoor search by swarms of simulated flying robots without global information". In: *Swarm Intelligence* 4.2, pp. 117–143.

Stoy, K and Radhika Nagpal (2004). "Self-reconfiguration using directed growth". In: *In Proc. 7th Int. Symp. on Distributed Autonomous Robotic Systems*. Citeseer.

Stützle, Thomas, Marco Dorigo, et al. (1999). "ACO algorithms for the traveling salesman problem". In: *Evolutionary algorithms in engineering and computer science* 4, pp. 163–183.

Sycara, Katia P (1998). "Multiagent systems". In: *AI magazine* 19.2, pp. 79–79.

Tapus, Adriana, Maja J Mataric, and Brian Scassellati (2007). "Socially assistive robotics [grand challenges of robotics]". In: *IEEE Robotics & Automation Magazine* 14.1, pp. 35–42.

Thalmann, Daniel (2007). "Crowd simulation". In: *Wiley Encyclopedia of Computer Science and Engineering*.

Tisue, Seth and Uri Wilensky (2004). "Netlogo: A simple environment for modeling complexity". In: *International conference on complex systems*. Vol. 21. Boston, MA, pp. 16–21.

Tognato, Riccardo et al. (2019). "A stimuli-responsive nanocomposite for 3D anisotropic cell-guidance and magnetic soft robotics". In: *Advanced Functional Materials* 29.9, p. 1804647.

Toner, John and Yuhai Tu (1998). "Flocks, herds, and schools: A quantitative theory of flocking". In: *Physical review E* 58.4, p. 4828.

Toussaint, Nicolas, Emma Norling, and René Doursat (2019). "Toward the Self-Organisation of Emergency Response Teams Based on Morphogenetic Network Growth". In: *Artificial Life Conference Proceedings*. MIT Press, pp. 284–291.

Turing, Alan Mathison (1952). "The chemical basis of morphogenesis". In: *Philosophical Transactions of the Royal Society of London. Series B, Biological Sciences* 237.641, pp. 37–72.

Vásárhelyi, Gábor et al. (2018). "Optimized flocking of autonomous drones in confined environments". In: *Science Robotics* 3.20, eaat3536.

Vergara, Andrea et al. (2017). "Soft modular robotic cubes: toward replicating morphogenetic movements of the embryo". In: *PloS one* 12.1, e0169179.

Wang, Hanlin and Michael Rubenstein (2020). "Shape Formation in Homogeneous Swarms Using Local Task Swapping". In: *IEEE Transactions on Robotics*.

Wang, Jian, Srinivas Peeta, and Xiaozheng He (2019). "Multiclass traffic assignment model for mixed traffic flow of human-driven vehicles and connected and autonomous vehicles". In: *Transportation Research Part B: Methodological* 126, pp. 139–168.

Webster, Jamie and Martyn Amos (2019). "A Turing Test for Crowds". In: *arXiv preprint arXiv:1911.06783*.

Weisbin, C. R. and G. Rodriguez (2000). "NASA robotics research for planetary surface exploration". In: *IEEE Robotics Automation Magazine* 7.4, pp. 25–34. DOI: 10.1109/100.894030.

Whitesides, George M (2018). "Soft robotics". In: *Angewandte Chemie International Edition* 57.16, pp. 4258–4273.

Woolsey, Max, Christopher A Kitts, and Finn Amend (2019). "A Diving Autonomous Surface Vehicle for Multi-Robot Research and Development". In: *OCEANS 2019 MTS/IEEE SEATTLE*. IEEE, pp. 1–7.

Yates, F Eugene (2012). *Self-organizing systems: The emergence of order*. Springer Science & Business Media.

Yoshida, Kazuya (2009). "Achievements in space robotics". In: *IEEE Robotics & Automation Magazine* 16.4, pp. 20–28.

Yu, Xi et al. (2019). "Synchronous rendezvous for networks of marine robots in large scale ocean monitoring". In: *Frontiers in Robotics and AI* 6, p. 76.

Zhu, Liuhua (2020). "Criterion for the Emergence of Meta-Stable States in Traffic Systems". In: *Journal of Applied Mathematics and Physics* 8.6, pp. 976–982.