Development, evolution and genetic analysis of *C. elegans*-inspired foraging algorithms under different environmental conditions

Gabriela Rabelo Andrade

Submitted in accordance with the requirements for the degree of
Doctor of Philosophy in Mechanical Engineering

The University of Leeds
School of Mechanical Engineering

November, 2019

The candidate confirms that the work submitted is her own, except where work which has formed part of jointly-authored publications has been included. The contribution of the candidate and the other authors to this work has been explicitly indicated below. The candidate confirms that appropriate credit has been given within the thesis where reference has been made to the work of others.

Chapters 5 and 6 of the thesis contains material described in the paper entitled "A minimal biologically-inspired algorithm for robots foraging energy in uncertain environments". The manuscript has been accepted for publication in Elsevier Robotics and Autonomous Systems on 06/03/20. I am the first author and my supervisor, Dr. Jordan Boyle, is my co-author[a].

The Manuscript describes the following: design of a bio-inspired foraging algorithm; design of a simulated virtual environment; optimisation with evolutionary algorithms; speciation and interpretation of results.

My contribution to the paper was: Conceiving the idea (50%), writing the code (100%), running the experiments (100%), interpreting the results (75%), writing the paper (75%).

My co-author's contribution to the paper was: providing high-level guidance, helping to conceive the idea (50%), helping to interpret the results (25%) and helping to write the paper (25%).

[a] The paper is available at <https://doi.org/10.1016/j.robot.2020.103499>.

Acknowledgements

First, I would like to thank my parents. I am sure that I am still far from imagining how many sacrifices they made in order for me to receive the education I received. Without all their effort and love, none of this would be possible. Mom and dad, thank you - I love you.

To my sister, who has always been an example of effort and dedication, and to my nephews, for bringing so much joy. In particular, to my little Gabriel: I hope that one day you will forgive the absence of your godmother that loves you so much.

To the friends I made in Leeds, who managed to bring some sunshine to the rainy Leeds weather: Sil, Sheyla, Rafa, Lu, Diogo, Raíssa, Mohannad, Cleandho, Sara, Fernanda, Jonas, Thiago, Aws, Dani, Kaz, and my colleagues from the office. In particular, I thank my friends Gustavo, Bianca, Bob, Arthur, Gésner, and Anna Carolina, who became my family in Leeds.

To my friends in Brazil, who so often made the distance seem small: Fabiano, Jac, Arthur, Cecília, Hernan, Jana, João, Renato, Kelly, Mari, Merson, Kel, Douglas, Vanessa, Lili, Lu, and so many others. To my friends and fellow researchers at CBEIH and at Instituto Teia. I would especially like to thank my friend Vinícius, another sibling that life gave me and whom I am honoured to know I can always count on.

I am grateful to some of the strong and inspiring women around me, who even in a difficult world, have the elegance and the ability to say 'yes' more than 'no': Maria Edith, Marcela, Helen, Lia, Kelly, Silvia, Maristela, Lucimeire, Iyá, Silvana, Junia, my grandmother Elza, and my mother, Luiza.

To Emma, for taking my hand one day and never letting go, for being my love, my safe haven, and the best friend, girlfriend, wife, and thesis reviewer anyone can imagine. Thank you for the affection, for your patience with my absence and stress, and for always making me believe in myself - even when I couldn't do it myself. I love you!

To my supervisor, Jordan Boyle, whose creativity and ingenuity I admire beyond measure, and to whom I am immensely grateful for his dedication, constant presence and his unwavering ability to guide the way, even in the face of so many adversities that arose along the way. Thank you!

To my co-supervisor Ozz Querin, and to Professor Rob Richardson, who provided critical insights in the early stages of the research.

To Professor Mehmet Dogar, who took the time to participate in my annual evaluations, and who provided such valuable ideas and feedback for my work to move forward, I am incredibly grateful.

To the Professors Jongrae Kim and Fumiya Iida, for finding the time to participate in my Viva. Their generosity, talent, encouragement, and intense contributions to the improvement of my research were invaluable.

To Peter Hayward, for his kindness in providing ways for me to use the computational infrastructure that was essential to my experiments.

Finally, I would like to thank the Brazilian National Council for Scientific and Technological Development (CNPq), through the program Science Without Borders (Grant reference: 207062/2014-5). This research would not have been possible without the financial support provided by this program.

Abstract

In this work 3 minimalist bio-inspired foraging algorithms based on *C. elegans*' chemotaxis and foraging behaviour were developed and investigated. The main goal of the work is to apply the algorithms to robots with limited sensing capabilities. The refined versions of these algorithms were developed and optimised in 22 different environments. The results were processed using a novel set of techniques presented here, named Genotype Clustering. The results lead to two distinct conclusions, one practical and one more academic. From a practical perspective, the results suggest that, when suitably tuned, minimalist *C. elegans*-inspired foraging algorithms can lead to effective navigation to unknown targets even in the presence of repellents and under the influence of a significant sensor noise. From an academic perspective, the work demonstrates that even simple models can serve as an interesting and informative testbed for exploring fundamental evolutionary principles. The simulated robots were grounded in real hardware parameters, aiming at future application of the foraging algorithms in real robots. Another achievement of the project was the development of the simulation framework that provides a simple yet flexible program for the development and optimisation of behavioural algorithms.

Table of Contents

List of Tables

List of Figures

Part I
Introduction

Chapter 1
Introduction

Some great problems require small solutions. Small scale robots may one day be the best way to solve big problems: air pollution, crop pollination, repairing organs or blood vessels, healing wounds, detecting and targeting cancer cells, pathogens, or infections are just some of the examples. Recent developments in robotics, electronics and smart materials are blazing the way for a new generation of robots: simpler, smaller, and able to act either as individuals or as a swarm. With reduced size and cost, and increased capabilities, these robots might be able to tackle problems that are still too expensive, difficult, or even unfeasible for humans or existing robots.

In the open environment, some of the tasks that could be performed by a new class of low-cost minimalistic autonomous robots would be: detecting and/or neutralizing pollutants, harmful organisms or chemicals in the environment[1–7], actuating in hazardous environments[8,9], monitoring natural systems[7,10], exploring the sea[11], pollinating plants[12], or even colonising other planets. In the human body, some applications would be: delivering drugs[13–16], targeting cancer cells, infections, or pathogens, monitoring health conditions and health indicators[17], removing biofilm from surfaces[18], patching up wounds, and performing micro-surgeries[19,20].

Some of the limitations these robots would face are those that arise due to the constraints of its hardware, software and embodiment, and those regarding production costs, as some of these robots would be expendable, or semi-disposable. Another key limitation relates to energy supplies, as some small and/or disposable applications would not suit traditional batteries - either because of payload limitations or because of toxic components in the most common batteries. One solution for that is the development of solutions allowing the robots to harvest energy from the available sources in the environment: light, wind, wave power, sugars[17], biomass, and micro-organisms are some alternatives. One example of the use of biomass and microorganisms to power small devices is the technology of *Microbial Fuel Cells* (MFCs). MFCs use living microorganisms (electrochemically-active bacteria) to convert organic fuels into electricity[21].

As a technology under development, the capacity of generating power of MFCs and similar components is still limited. However, when combined with developments towards miniaturization of electronic components and advances in smart materials and manufacturing techniques[14,15,18,20,22–28], these components could give power to minimalistic autonomous robots, capable of harvesting energy while performing tasks out in the environment. For certain environments and applications, by feeding on the pollutants they are built to eradicate, robots would be able to economize this task. In a recently published work, the researchers successfully developed a robot prototype that consumes

the pollutants it is designed to eradicate and uses them as an energy source[5]. In a similar approach, recent advances successfully used glucose to power micro-devices, envisioning the application of such technology in micro-robots to monitor blood sugar[17].

The capacity to harvest energy from biomass and other sources available in the environment, combined with cheap and disposable robots, is laying the groundwork for autonomous robots that would behave less like machines, and more like living beings - able to feed, perform tasks and eventually decompose or complete their cycle. Across evolution, living beings have evolved ways to find food, even in harsh environments. From single-cell organisms to complex animals, one of the most basic needs is to feed and, with the exception of autotrophs, such tasks usually involve movement, behavioural, and foraging strategies. For those organisms that actively move, foraging also encompasses the trade-off of spending energy to acquire energy[29–31]. In that context, looking for inspiration in biology is a prolific way to design - and evolve - technological solutions[32–38].

Regarding the problem of minimalistic energy-harvesting mobile autonomous robots, a perfect example to look to for inspiration is that of the free-living roundworm Caenorhabditis elegans, one of the first organisms to have its genome and connectome fully mapped. Despite its remarkable simplicity, *C. elegans* displays a considerable repertoire of behaviour including foraging, mating, feeding and threat avoidance, as well as rudimentary learning, memory and social behaviour[39–47]. Its foraging strategy emerges from the interaction of simple sensory information and limited body movements - a combination of runs and turns of varying speed and direction.

Computer models and simulations provide methods for the investigation of natural processes, systems, animal behaviour, among others. These are some of the approaches used in the field of Artificial Life (A-Life). A-Life does not focus on building perfect models of life (as in Biological Sciences), but rather to capture the most general and relevant aspects of a given system and implement them in a simulation[48,49]. The aim of the field is not necessarily only to recreate systems as they are, but also as they could be. A-Life combined with biomimetics and optimisation might provide useful insights and tools for robotics and for the development of algorithms.

The main drive to produce this type of algorithm is that a model based on an organism as simple as *C. elegans* lays the groundwork for similarly simple autonomous robots, capable of performing tasks with minimal, or no human supervision or intervention.

The foraging task was chosen as the testbed for the algorithms, as to remain autonomous a robot must be able to conserve and replenish its batteries. Furthermore, by basing the algorithm testing around the foraging task, robots not only tackle the problem of foraging for energy, but also how to cover large areas with minimal sensing and decision making capabilities.

This work is an investigation in Artificial Life about Evolution and differentiation of species in response to environmental constraints in the context of minimalist bio-inspired behavioural algorithms. The algorithms, inspired by *C. elegans*' chemotaxis and foraging behaviour, are aimed to be applied to a future class of autonomous mobile robots, with energy harvesting capabilities. The results obtained from these studies are expected to contribute to the fields of Robotics, Artificial Life, Biological Sciences, Biomimetics, and Optimisation.

1.1. Aims and Objectives

The aims of this research are presented as follows:

To develop and optimise foraging algorithms inspired by the behaviour of *C. elegans*, for autonomous minimalist mobile robots.

To define a testbed for the problem of foraging energy and avoiding threats in unknown environments.

To test and apply evolutionary algorithms for optimisation to optimise the parameters of the Foraging Algorithms for a given set of environments, furthermore to obtain the best solutions for each environment.

To develop methods to identify, group and classify these best solutions found for each problem by applying concepts from Biology, Evolution, and differentiation of species.

To contribute to biological sciences by providing biomimetic foraging algorithms that may be useful for studying chemotactic behaviour in *C. elegans* and eventually other species.

The specific objectives are to:

- Create a model that captures the most relevant aspects of *C. elegans'* foraging behaviour;
- Develop optimizable foraging algorithms that could be deployed into minimalist robots, capable of working with a single sensor input;
- Create a program to simulate robots using custom behavioural algorithms to forage resources;
- Create standard virtual environments as a testbed for foraging algorithms;
- Deploy the foraging algorithms into virtual robots with specifications grounded in real hardware;
- Identify the best suited Evolutionary Algorithms for the optimisation problem defined;
- Optimise the foraging algorithms using evolutionary algorithms;

- Identify patterns in the best optimisation solutions utilising clustering techniques and group them into species, by genetic similarity.

1.2. Contributions

This work is the first to develop and test biomimetic foraging algorithms based on *C. elegans* into autonomous robots that were simulated in a virtual set of environments, and contributes to the fields of Artificial Life and Robotics by proposing standard problems to test foraging algorithms. It also contributes to the fields of Biological Sciences and Biomimetics by creating minimalistic models of *C. elegans* foraging behaviour (Figure 1.1). These models are optimisable and show how very simple models with very few parameters can solve the problem of finding attractants and avoid repellents in unknown environments.

This work is also the first to propose the use of clustering techniques to identify and classify the solutions optimised with evolutionary algorithms, and contributes to the field of Evolutionary Computation and Optimisation by proposing a new method to identify and classify diversity among evolved solutions.

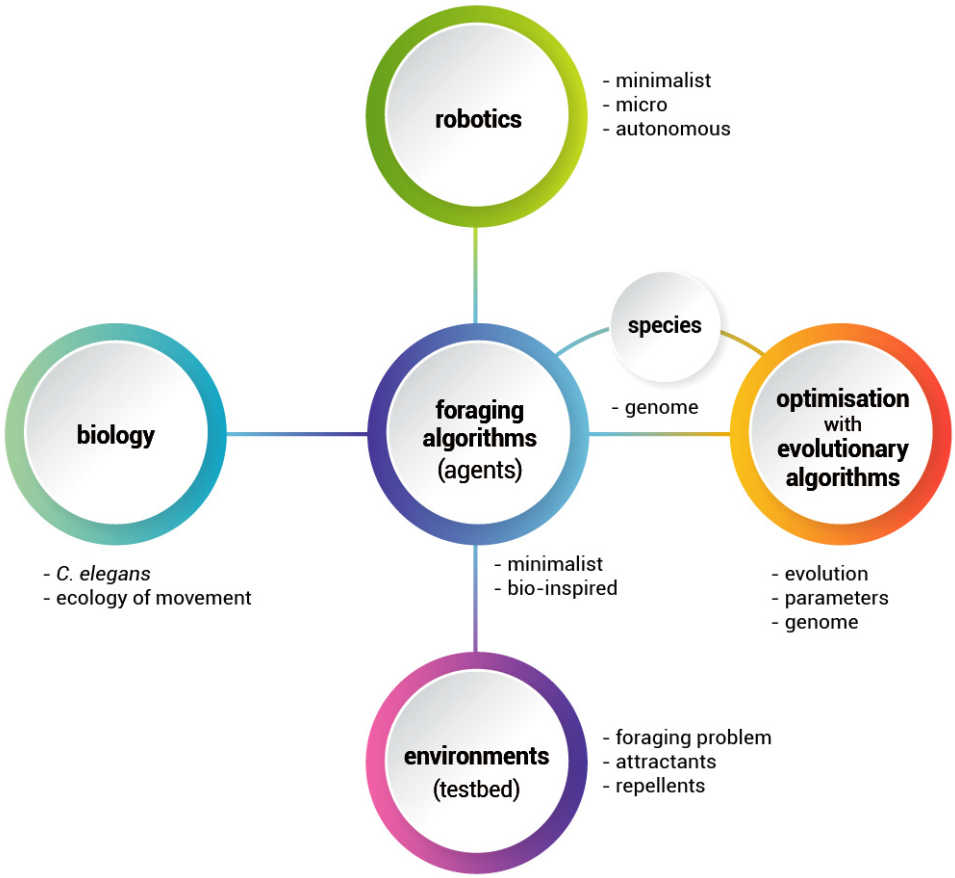


Figure 1.1 Mapping multidisciplinary interactions between several fields/areas involved in this research.

1.3. Applications

Given the transdisciplinarity of this work, the literature review will be organised in two chapters, one focusing on the synthetic aspects of the field being explored, the other dealing with the natural aspects. Chapter 2 covers the topics of Artificial Life, Artificial Intelligence and Robotics while Chapter 3 covers the topics related to Ecology, Movement and Behaviour of animals.

Over the course of my research, I designed six foraging algorithms inspired by the behaviour of *C. elegans* and these were tested over 10 series of experiments. As this work intersects with several fields and many of them are novel, there are no standard problems or methods for evaluating robots utilising foraging algorithms. For that reason, I had to both develop a tailored simulation platform and define standard problems to evaluate the performance of the algorithms. That was done by creating the settings and procedures to generate a set of controlled environments in which the experiments would be performed.

Coding a simulation program, implementing optimisation with evolutionary algorithms, defining the optimisation problem and the heuristics, and finally designing the algorithms was certainly not trivial and therefore it was only after seven of the ten series of experiments that I was able to fine-tune the methods and consolidate an experimental setup. The processes, issues and results that emerged from these seven initial experiment series were essential in defining the algorithm design and the optimisation problems. The outcomes of these experiment series also account for the robustness achieved in the later series, and very often will explain the reasoning behind the methods, techniques and other design and optimisation decisions adopted later in this work.

Given the nature of this work, the entirety of Chapter 4 is focused on these series of experiments, and also the three early foraging algorithms. This is included with the aim of clarifying the chain of decisions which led to the final methods, and also to offer guidance on the issues which were overcome during this period for further research.

The consolidated methods for simulating, optimising, and processing the results of the three refined algorithms, along with three later experiment series, are presented in Chapter 5.

The three refined foraging algorithms inspired by *C. elegans* behaviour are presented separately, in Chapters 6, 7, and 8. The analysis of the results and the discussion about each refined foraging algorithm are presented in the corresponding chapter. Chapter 9 presents the results of a performance comparison between the three refined algorithms.

A general discussion and conclusions are presented in Chapter 10.

Part II
Literature Review

Chapter 2
Literature Review in Artificial Life, Artificial Intelligence and Robotics

2.1. Artificial Life

Artificial life, or A-Life for short, is a field of research that is notable for the multidisciplinarity in its core. Gathering knowledge from Biology, Computer Sciences, Mathematics, Chemistry, Behaviourism, and Robotics, amongst others, the purpose of artificial life studies is double-fold: through simulating biological systems in various scales - molecular, cellular, organismal and population - it may provide researchers with a better understanding of the mechanics of said systems in the context of their natural occurrence, as well as enable them to employ these mechanics in human-made applications not necessarily related to life sciences. With the insights provided by creating and studying artificial living systems designers, computer scientists and even theoretical mathematicians are able to explore a large gamut of possible solutions for varied problems in an efficient and timely manner. It is the role of the researcher to identify fit candidates out of the countless solutions evolved by the natural world to replicate, analyse and convert their inner workings into practical benefits for the application at hand[50].

One of the most important contributors to the field, the physicist Christopher Langton, proposed the name Artificial Life in 1987[51] at a workshop he organized while working at the Los Alamos National Laboratory. The event was intended to bring together researchers from various fields interested in the control and repeatability offered by programmed simulations, instead of the random sampling and extensive observation typical of biological studies[52]. Langton saw in computational simulations a tool able to substitute an entire bio-focused lab. Using computers as "one simple-to-master piece of experimental equipment"[53] made possible doing away with the considerable assortment of specialized apparatus necessary for wet-lab analysis. This way it would be possible to try and synthesize systems analogous to living systems so they could be studied in a more efficient and better defined manner.

Many years before the field got its name, some very notable researchers were already setting the foundation of what would be eventually known as Artificial Life. Pioneer William Grey Walter achieved in 1949 the kind of life-like system defined by Langton with his hardware-centric, analogue tortoises. Two years later Alan Turing was the first to investigate biological growth patterns using digital computer simulations[53].

In this initial research Turing used the first commercial general-purpose digital computer, the Ferranti Mark I. He set out to build his simulations as soon as the newly available machine was installed in the Computing Machine Laboratory at the Victoria University

of Manchester, where he worked at the time[54]. Programming the Mark I was not an issue, since Turing was himself involved with the development of this computer and was already familiar with it, so he got to work right away. The hypothesis he wanted to simulate was also already somewhat developed, being his personal take on the ongoing work of zoologist D'Arcy Thompson. Thompson's idea was that biological forms were actually a direct effect of chemical diffusion and interactions at the molecular level, replicated and spatially amplified by the growth of the system. Just like inorganic forms display fractals and repetitive patterns as a result of chemical interactions ruled by the laws of physics, this mathematical ruling also inevitably emerges in biology[55]. Both Thompson and Turing investigated the astoundingly common occurrence of the Fibonacci series in the distribution of elements in botany and zoology. Turing also had an special interest in the structural organization of the starting point of most living forms: embryos. Turing called "chemical embryology" the hypothesis he experimented on with the Mark I, and it was the beginning of modern concepts like morphogenesis - the origin of the shapes of organic systems and living beings -, the reaction-diffusion model - one that can accurately simulate the formation of visual patterns such as the spots on leopards, stripes on giraffes, and distribution of scales on reptiles -, and even neural networks, that would only become commonplace decades later.

Ranging from the investigation of physiology, and the detailed biophysics and biochemistry involved in living systems, through cellular and tissue scale, up to whole-organism dynamics, artificial life studies can be classified by the support they happen on. The division of support media tends to be closely related to the scale and level of organization of the systems under scrutiny. The three main classes of medium employed are wetware, software, and hardware, and the connection between the scale and medium, ranging from molecular to populational, follows a very practical logic, as follows[50].

When simulations take place in petri dishes - wetware -, having molecules in suspension as their basic building blocks, it is called Wet A-Life. The experiments developed at this scale are the closest to natural systems, but are also the slowest to develop, mainly due to their time scale being defined by the direct participation of a human operator in each of the processes, plus the necessity of waiting for the intended reactions to occur. The most common application of this class of A-Life is to artificially produce, evaluate and select RNA molecules by their capacity of realizing useful work, usually catalysis of various reactions of interest. The working principle in this case is directly linked to statistical mechanics and chemical kinetics, and based on the slew of molecular interactions that can be expected from any practical volume of analytes. A series of more than a thousand RNA variants is put into a solution containing the substrates for the intended reaction. If any of the trillions of copies of these RNA strands is able to catalyse the reaction in question, it will eventually do so, given the sheer amount of chemical interactions taking place per second in each experiment run. From there, the experiment follows as a sort of

artificial evolutionary selection. The more successful strands are recovered from the solution, mutations are introduced to their code, the resulting variants are replicated in bulk and tested again. The cycle is then repeated, and each cycle is akin to a generation in the evolutionary process of a molecular population, until the desired properties are selected[56].

The wet technique sheds light into the possibilities stemming from randomly generating and subsequently selecting something so complex and versatile as a molecule. Because of the open-ended quality of this style of study, the insights achievable are applicable to objective, manufacture-oriented ends, as well as to theoretical questions like the "RNA World" theory for the origin of life[57].

When simulations happen in a software environment, the most common and cheap way to run them, it is called Soft A-Life. The earliest Soft A-Life was the cellular automata (CA), having the first example been presented by physicist, mathematician, and computer engineer John Von Neumann in his lecture at the Hixon Symposium, in 1951. There, he proposed a definition for an automaton: any machine or program that could behave in a way that took inputs from the environment and combined it with its own code or build somehow to produce change - growth - in a series of steps throughout its running time. He also noted that these simple rules are, in a reductionist view, actually what natural living creatures do. That is the basic concept of the CA - a simulation of relatively simple organisms distributed in a colony-like manner making decisions based on their programming and external conditions in sequential steps, growing, replicating and, eventually, dying. The first CA was built by Von Neumann and Stanislay Ulam and consisted of thousands of cells capable of displaying 29 distinct states, a massively complex example of CAs, even to this day. The result was a program capable of, among other feats, replicate while keeping a copy of its original code that could be output when needed. In the infancy of digital computing, their program was able to survive, mutate and reproduce, effectively simulating open-ended evolution. Even while the structure of DNA was yet to be fully understood, a computer could, indeed, be used as a petri dish[58].

The next class of artificial life regards simulations run directly on hardware, which may vary greatly in complexity. A comparison between modern autonomous vehicles and those discussed in the literature review - Tilden's minimalistic BEAM automata, or even Grey Walter's seminal tortoises - exemplify the range of challenges present in design, implementation, and the resulting functionalities. From the hardware basis, this class is called Hard A-Life, and can be considered as one interpretation of the field of Robotics.

The strongest focus of the hard A-Life class is in the simulation of individual living beings, and for good reason. When dealing on this scale, a curious synergy enables massive decrease in the task of setting up the simulation, at the same time it incidentally generates a leap in quality of results. Consider the procedure of replicating the behaviour

of a single life form in a complex environment: if one was to finely model and simulate the vastly complex chemistry, mechanics and physiology of said organism, all while simultaneously modelling and validating the even more complex environment in which the simulation is to run, the computational resources needed would simply be out of the technological capabilities of mankind for the foreseeable future. This can be anticipated due to the immense intricacy of the target environment - the real world - and its potentially infinite properties and the interactions between them - physical, chemical, structural, mechanical, to mention a few. Add to this already gargantuan equation the noise and uncertainties of reality and it only becomes more clear that simulating this scale is non-trivial[50].

A solution to this was proposed by Rodney Brooks in 1986, and it was called the *Subsumption Architecture*[59]. While Brooks' work is discussed in greater detail in the literature review section, for the current explanation it suffices to mention that this architecture simply does away with the heavy lifting of creating a symbolic representation of the ambient, substituting it for an even better alternative: the real world itself. By doing an analogous change to the modelling of the organism, from computer code to actual electromechanical build, one can leave the world building to the electronic detection made in real time by various layers of sensors and instead focus on fine-tuning the interaction between these layers. Instead of arising from a complex control system, complexity derives from the complex environment. Because of all these benefits, it is possible to quickly and effectively experiment with multiple robotic variants with different bases and objectives[60].

This nomenclature is, not surprisingly, reminiscent of the in-silico, in-vitro, and in-vivo classification style common in artificial and natural life sciences[61], serving a similar purpose but with each simulation substrate being fully useful in itself, instead of a step to an ultimate goal of applying the results to living tissues or creatures.

Another noteworthy aspect of the A-Life field is the philosophical questions that arise when dealing with the very definition of what is life. Von Neumann had already pointed at the fact that at the base of behaviours exhibited by natural living organisms are, indeed, simple rules. Researchers must, then, deal with the implications of such definitions on the future of studies in A-Life, since its fundamental tools - programs that emulate survival abilities found in living beings - are expected only to grow in complexity and approximation to their real counterparts. The notion of these simulations becoming indistinguishable from biological, naturally occurring life has been predicted and discussed, and from this discussion arose two lines of thought, called Strong A-Life and Weak A-Life. Further, the discussion includes the question if such division is positive or just the kind of unproductive detour expected of the beginnings of a science field[62].

The Strong notion is closely related to the Strong AI concept, in that it describes life as something immaterial, be it intelligent life and its essential components - namely consciousness, self-awareness, and free will - or the electronic analogue to the smallest, simplest forms of life - the self-replicating computer viruses[63].

According to this logic, life is an abstract concept, a pattern of instructions in space-time , and can exist no matter the substrate, and the consequence of this is a challenging yet rich discussion around the meaning of life and the ethics of A-Life research. This discussion is expected to become particularly relevant when technology reaches the point of producing an artificial consciousness that cannot be told apart from our own. For now notions are still mostly philosophical, and the proposal of definitions to artificial life in terms like “prosthetically controlled thought experiments of indefinite complexity” set the tone[62].

Conversely, according to the Weak A-Life reasoning the simulations can never truly be real life, a status exclusive to biochemical, wet-systems. From this point of view any simulation in the context of A-Life is just another tool for better understanding nature.

2.2. Minimalist Robots

The field of minimal robotics was defined in its beginnings by the employment of almost bare-bones designs, often based entirely on discrete analogue components. One of the pioneering works combining AI and robotics was developed by William Grey Walter around 1950[64,65]. Being originally a neurophysiologist, Walter was particularly interested in the properties of the brain and their relation to the multitude of behaviours observed in living beings. Influenced by recent findings related to the wiring between neurons, which also give rise to the concept of artificial neural networks, Grey Walter started his work in the field of robotics with the development of autonomous robots able to mimic animal behaviours. His famous Tortoise Robots were built around 1950 and consisted of simple sensors and two vacuum tubes controlled by electric circuitry. Elmer and Elsie (acronym names for two of his *M. speculatrix*) were some of the first built and exhaustively experimented robots (Figure 2.1).

Figure 2.1 *M. speculatrix* robots created by Grey Walter.

(A) chasing a light source, (B) Without cover, exposing its circuits and (C) in an experiment interacting with a mirror and driving itself to charging station. Source: Hoggett[66,67].

In Walter's book "The Living Brain" he proposed a list of lifelike behaviours such as: propensity to explore the environment (speculation), positive or negative tropism, distinction between effective and ineffective behaviour (discernment), a tendency to seek out the most favourable conditions, among others[64,66–68].

With a simple combination of sensors and actuators, Walter was able to implement a small set of actions that, combined, lead to more complex behaviours. As discussed by Owen Holland[64], this work was a great contribution to the field, but some of the observable behaviours of the tortoises may have been misinterpreted by Walter when in fact some were just a matter of luck.

Furthermore the field of minimalist robotics had in Valentino Braitenberg one of its most prolific thinkers. The Italian-born neuroscientist saw in computers a way to model, in a simplified manner, the extremely complex connections found between neurons. His work in this field began in the late 1940's and his 1984 book Vehicles: Experiments in Synthetic Psychology describes the fundamentals of minimalist robot design[69].

Braitenberg's minimal intelligence model is based on the concept of very simple vehicles composed of analogue sensors directly wired to motors. As a way to mimic the most basic form of energy-seeking behaviour in simple organisms, such as bacteria, the sensors initially employed in his practical examples are light sensitive, enabling the concept vehicles to perform *phototaxis*. The fundamental design has one sensor and one motor directly connected so that when light is detected by the sensor it causes a directly

proportional acceleration to be applied to the vehicle by the motor. This mono-sensor, mono-motor design may accelerate when exposed to a light source, effectively "fleeing" when in a lit environment, only stopping when its sensor enters darkness again. This most basic setup is devoid of much control and has little practical application besides illustrating the concept, and has behaviours only as complex as a single-cell organism. Taking one step further in complexity, Valentino proposes a vehicle displaying bilateral symmetry of its components, with two motors, each connected to a wheel, and a pair of light-sensitive sensors wired in a "straight" manner, as in the sensor on the right side of the robot is wired to the right-hand motor and wheel assembly. Note that the increase in motor capabilities is also present in the directionality of its sensory input. This robot will display a certain behaviour when exposed to a light source: the side closer to the source will accelerate more, resulting in the robot turning away from the light. The result of this wiring scheme is that the robot will accelerate as fleeing *from* the light, instead of simply fleeing from the lit environment.

If the wiring was to be crossed, so that the right side sensor affects the left-hand side motor assembly, the vehicle would steer into the light source, in a behaviour akin to *seeking*. This simple difference in design causes a significant and seemingly complex change of behaviour, and in the examples given only the simplest vehicle designs are described.

Further exploration of the concept of Braitenberg's vehicles include the use of multiple types of input sensors, and as many kinds of electrical and logical connections made between themselves and various actuators used for whole-body movements, body-part movement, and skills such as jumping, striking, swallowing, and hundreds more complicated in bodily nature and objective.

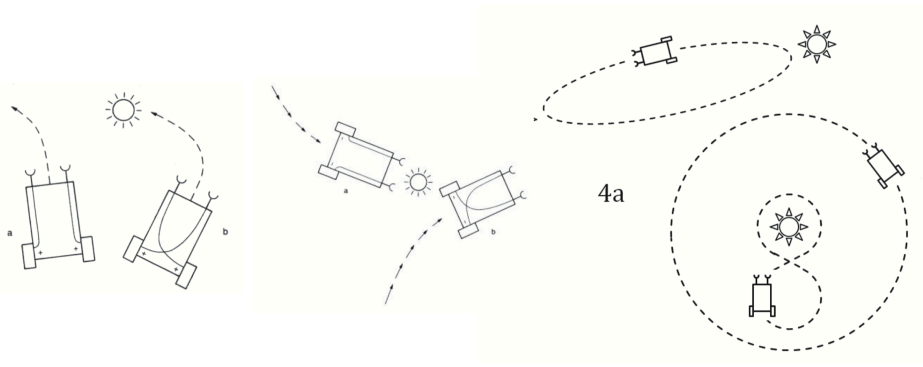


Figure 2.2 Braitenberg's vehicles. Source: Braitenberg[69].

Braitenberg also explores the use of non-linear equations to control the relationship between sensing and acting in his robots. When taking into account delays such as the friction that must be overcome for an electrical motor to produce movement, he likens the observed behaviour - more complex and, thus, harder to distinguish from the interaction

of an organic life form to its physical environment - to *instincts, decisions* and *will.* Patterns he called *love*, *liking, fear,* and *aggression,* arise from the interaction of simple rules very similar to the result of simple neurons acting in a network. The names given to these patterns are similar to the ones defined by William Grey Walter. This analytical reduction equates to a form of reverse engineering of the working structures of living brains and can be very useful in the construction of the simple set of rules that comprises a foraging algorithm.

When considering how a robot locates itself in an environment and perceives the world around it, the expression *behaviour-based* translates into a design paradigm that made a whole new generation of simple robots possible. Building on the sensor-motor relations proposed by Braitenberg and the seemingly intelligent yet simpler tortoises of Grey Walter, the field of behaviour-based robotics had a great contributor in Rodney Brooks.

Taking inspiration from the biological world, Brooks proposed a new robotic architecture in which the artificial intelligence driving the machine did not have to build a detailed model of the world it had to navigate to, then, plan and act accordingly. Instead of high power, intricate AI with a generic scope that has to plan into the future, he envisioned a reaction-based way of navigating that he called the *Subsumption Architecture*. This architecture equates to what today is known as *narrow* or *weak* AI and describes a low intelligence that is very specific and very efficient at the single task it tackles, rather than a higher intelligence that has a generalist approach to problem-solving. A comparison fitting for putting in perspective the difference between the two kinds of intelligence would be the scale and scope of a machine vision and sorting AI - very specific and, therefore, narrow - and one on par with the human brain - capable of reading unknown environments and situations, future planning, and manipulation of high level symbols typical of an *artificial general intelligence*, or even sentience and consciousness, that would put it above a simple computer program and define a *strong AI* [70–72].

More than a robot building technique, the *Subsumption Architecture* requires a change in the whole design mindset, changing the primary abstraction used to define the world to be navigated. Instead of definite states, the world must be defined as a series of tasks, to which the robot must respond by doing the “right thing”. In more technical terms, instead of defining artificial intelligence as the AI ability of manipulating *meaningful symbols* that define the environment - an approach based on the *Symbol System Hypothesis* - the controlling AI must be able to perform in a direct sensing-acting manner, without the use of a representation language or other medium for understanding the world - it is, thus, based on the *Physical Ground Hypothesis*[73].

To design a robot with *Subsumption Architecture*, various routines of instructions to solving very specific tasks - single behaviours - are encapsulated by a human designer and are programmed in layers to be activated in the right circumstances, as defined by the

robot's sensors. At the same time, they must suppress certain other routines, that may be opposite, contradictory or simply pertain to a different behaviour context. From the computational standpoint, these instructions are often remarkably fast, sometimes consisting of simple vector additions and countdown timers. The high energy efficiency of this architecture is also very appropriate for minimal, mobile robots. A major design requirement is that there are enough routines to cover the whole of the robot's operation in the intended environment and the inclusion of the proper cross-routine suppressions. Infinite loops are avoided structurally since all instructions are based on finite state machines which can be augmented with timing elements. From the resulting interaction of basic behaviours emerge new ones, often unexpected and higher in complexity, apparent cognitive level and life-likeness[74–76].

Brooks' idea that a robot does not need to plan ahead, but can navigate by simply sensing its way and dealing with obstacles as it goes, is at the core of a discussion on how to implement AI in robots. His reaction-based robots fared better in navigational tasks than predecessors like the Stanford Cart[77], which was based on a world-building and planning AI, even though they were much simpler in their representation of the world. The argument later formalized by Brooks is that: "The world is its own best model. It is always exactly up to date. It always has every detail there is to be known. The trick is to sense it appropriately and often enough"[75].

From the biomimetic standpoint, the Subsumption Architecture mimics the functioning of animal intelligence in the same layered way it developed from bacteria to human beings: the primitive layers of the system - involved in basic physiological functions common to man and bacteria, for example - do not have to keep track of a large trove of processes pertaining to higher functions, such as cognition and consciousness. In computing terms the main implication of a reaction-based approach is the reduction of the computing power and memory size necessary for the robot to operate: it does not need to compute high-level symbols for representing the world or numerous steps of planning, and it does not need to keep track of changing parameters to track its own state, be it position, attitude, speed, etc[75].

Another example of this logic is presented as a *move from A to B* problem. The proposal is that a tight sensor-motor feedback loops is established, having the environment as the medium for such a loop. When a robot must navigate between points A and B, if a sufficient amount of data is fed into the AI, activating the appropriate subset of instructions, the navigation should happen smoothly and more importantly, adaptatively. The quality (signal to noise ratio) and quantity of data (frequency of sensor reads) captured by the robot is only half the equation, though. The other half, the instructions applied in decoding such data, can be developed by hand or evolutionary algorithms but must be encapsulated by a human[74].

The definition of encapsulated behaviour first appears in the work of Karl Sims, *Evolved Virtual Creatures (EVCs)* and consists of the selection and segregation of certain control skills evolved through genetic algorithms as a way to keep them from being lost in the procedure of mutation between generations of virtual organisms. This approach requires human interference in the selection of the skills to keep and how they interact among themselves, and their encapsulation means that they will be maintained and built upon by consecutive generations and simulations, effectively directing the simulated evolution towards a certain goal. This method is discussed in greater depth in the section Evolutionary Robotics[78–82].

The field of BEAM robotics (BEAM being an acronym for *Biology, Electronics, Aesthetics and Mechanics*) takes another route to the bio-inspired minimalistic robotics. Based on a control concept composed by the *Nv Neuron* and the *Nv Network* and proposed by Mark. A. Tilden, this approach simulates neurons firing in sequence to achieve varying levels of reaction-based behavioural complexity.

The basic electronic structure of the Nv Neuron - “Nv” standing for “Nervous” - is composed of a capacitor, a resistor and an inverter. These neuron models work by discharging the capacitor through the resistor, resulting in a gradual decrease of the capacitor voltage. After this delayed discharge through the resistor, the voltage drops to a value that trips the inverter and triggers the next neuron, resulting in an oscillating voltage. This basic implementation is useful in a myriad of ways, and when combined with actuators enables the use of cyclical *back-and-forth*, *in-and-out*, *left-and-right* and similar aspects of movement and behaviour in a robot with a remarkably low component count[83].

Another principle of BEAM robotics is a periphery-to-core system structure. Tilden proposed an approach similar to Brooks’ *Subsumption Architecture*. In his design theory he referred to the high-complexity real world as the *Fractal World*, which could be directly navigated in an adaptive way by *soft machines* based on a *Biomorphic Architecture.*

Tilden’s architecture can be divided into two main systems. The first is a mechanical periphery comprising of legs and sensors that are physically compliant, literally *soft* and, therefore, well suited in handling the noisy interface with the complex terrains that constitute most environments. The second component is the electronic neuron core, cyclical in nature yet calibrated to receive interference from the peripheral system and change its own output slightly. The use of legs instead of wheels came naturally, as they can more easily be designed to perform sensorial tasks by changing their shape in regards to the environmental context.

The analogue nature of this design logic is evident in the continuous aspect of both elastic deformations suffered by legs, sensors, and the voltage, electrical charge and resistance

variation in the electronics of the core. The resulting machine employs its entirety in the computing of a fractal, unknown world[84,85].

Also, the word “soft” refers to the reactive and dynamic type of navigation BEAM bots are capable of, in clear contrast to robots hardwired to work from lookup tables, branching logic and other more conventional programming techniques.

Another strong focus of this field is on recycling e-waste for mechanical and electrical components. Since the automatons generally work well with the crudest of sensorial signals - physical contact - it is not rare to see implementations of paper clips, guitar strings, or even the leads of electronic components as sensors. The design possibilities are endless while maintaining low build costs and posing as a fun challenge for the many educators and hobbyists involved in this field[83].

2.3. Autonomous Mobile Robots

Autonomous robots are those able to perform some task or behaviour without external assistance. The degree of freedom such a robot displays is variable and depends on several factors that define its context of operation. The range of operational situations is notable, including stationary factory pick-and-place applications, the small but highly specialized context of spacecrafts, the comparatively infinite freedom of a long-ranging autonomous aerial vehicle and the critical activity of explosive devices disposal. These widely varied application environments are also one of the main motivations for the development of this field of robotics, in that it is not always desirable - or possible - to have a human operator in close enough proximity to exert control.

Presently most autonomous robots with complex navigational abilities are, in reality, very specialized vehicles - such as unmanned aerial vehicles, ocean-going vessels, and wheeled land vehicles - instead of bio-inspired robots that mimic the physiology and locomotive mechanics of organic beings. This means that there is a stronger focus on optimizing AI and sensorial tools to the operation of already proven-and-true concepts, as in *autonomously* driving a common car, the likes of which can be seen in cities and streets daily. However, terms such as *autonomous foraging* - the search for an energy source akin to a living being searching for food - exist in the artificial intelligence and robotics fields, indicating that it may be a matter of time until the rise of new designs, materials and building techniques lead to the creation of truly autonomous artificial life.

In direct relation to the physical context of the robot are two of the most critical aspects of its operation: remote control and power input. While fixed assemblies - such as the industrial welding robots found in automobile factories or pick-and-place bots in the electronics industry - allow for direct power from the grid as well as backup generators, however most autonomous *mobile* robots must carry a power source.

The issue of control is similar in nature, as a mobile robot cannot be connected to an external processing unit and must perform all sensing, navigation, actuator control and decision making operations onboard. Equally, it must carry some sort of communications apparatus to provide telemetry and grant human input when necessary. These fundamental challenges in the development of autonomous mobile robots will be discussed in the following section.

2.3.1. A Matter of Control

To be truly autonomous, a mobile robot must perform - independently from any external control - a series of procedures related to keeping its energy reserves, and do so regularly to achieve continued survival. The seeming opposition between *autonomy* and *control* is cleared when the definition of *control* is expanded to high and low forms of control. High level control refers to the definition of tasks and objectives a robot must perform, and until AI research reaches the point of conscious and curious robots, this type of control is exclusively human. Lower levels of control, on the other hand, consist of the direct operation of the machinery integrating the robot (e.g. driving a motor, reading a sensor, etc.) and the decision-making involved in modifying a previously defined step of operation (e.g. changing course to avoid an obstacle, abandoning a task to return to a recharging station, etc.). These processes occur internally and are the kind of continuous control a robot must exercise to be classified as autonomous[86].

This issue is highlighted in the extreme application of robotics in extra-planetary exploration. In this setting, the robot must operate in unmapped environments, under extreme atmospheric and radiation conditions, and at distances far too great for real-time operation. Some specialized designs, namely NASA's Mars rovers MER-A (Spirit, active 2004-2010) and MER-B (Opportunity, active 2004-2018), do account for human intervention and decision-making, but are able to autonomously map terrain in the appropriate spatial resolution with its onboard sensors and compute locally the optimal path to a given destination while avoiding obstacles. A further expansion of these capabilities is currently underway by the international space program ExoMars, whose rover is designed to combine absolute positioning data based on solar positioning, with relative positioning obtained by stereo cameras on the ground. This combination makes the robot especially suited for navigational tasks.

2.3.2. A Matter of Power

One of the main issues with portable electricity sources is that of specific energy. This concept refers to the relation between the net amount of energy obtainable from a power source - measured in Joules - in relation to its mass - measured in grams. Another useful

concept can be derived from the specific energy in relation to the physical properties of the power source, namely the volume, which defines the energy density. The latter is especially useful in mobile applications that must account for the extra mass of the energy source and its impact on mobility, autonomy (in the sense of distance travelled per charge), physical size of the vehicle, and feasibility of the mechanical design for any given application. Further, these properties of the fuel itself or power source units (PSUs) may be used to calculate the absolute power rating of a mobile robot in performing its intended tasks.

One critical aspect of the power issue is that energy released from different substances vary greatly, as it is the result from different types of reactions. From most to least energetically dense, these reactions are: nuclear, chemical, electrochemical and electrical. Directly or indirectly, all these reactions are employed in some type of autonomous vehicle by means of several power collecting and storing technologies. Most present-day robotics operate electronically and rely on electrical signals for control and power, making electrochemical batteries the most common portable source of electricity. The term battery includes several architectures and form-factors, which generate power from electrochemical reactions between their component materials. This energy cycle differs notably from the chemical nature of energy storage in most living organisms, which is derived from the metabolic breakdown of various biomolecules and, consequently, more dense and better suited for mobility. For its direct impact on the development of mobile robotics - among other mobile technologies such as transportation, IOT, and personal computing - the field of battery architecture is of utter relevance, and novel and more energy-dense technologies are in constant research[87].

Regarding vehicles and mobile bots, PSUs generally consist of a battery bank that may be implemented in a discharge-only manner that requires an interruption of operation for recharging, or coupled to onboard devices that allow for on-the-fly recharging, most commonly solar cells. Other technologies such as radioisotope thermoelectric generators (RTGs), fuel cells and alternators can also be used as onboard recharge methods or as standalone PSUs. The many configurations possible point to the complexity of the matter, as each PSU has its strengths and limitations, explored next.

RTGs, having a working principle based on the conversion of heat from nuclear decay reactions into electricity, are able to deliver steady high-power for long operational periods without maintenance, with both factors affected by the radioactive fuel employed. The selection of the fuel is a research field of its own and takes into account the types of radioactive decay displayed by a given isotope, its power density and half-life. Alpha decay is better suited for this application for being more easily absorbed and converted into thermal radiation, while beta decay can emit gamma and X-ray secondary radiation and imposes the need of heavy shielding in the PSU design. The power density of

radioactive fuels is inversely proportional to their half-life, resulting in a design dilemma: longer operational periods equal lower power ratings. This issue becomes even more relevant in regards to their niche application in spacecraft. Given the high costs related to manufacturing, storing and working with radioactive materials, this is the only current application of the technology. It is not uncommon for space probes to travel for years before reaching their destination, and the RTG must remain operational for this period and beyond, while still being able to power as many scientific experiments and sensors as feasible to better make use of the travel time, launch costs and the specific launch windows for rendezvous with selected space bodies. One recent example was NASA's New Horizons mission, launched in 2006 with a primary mission of studying the Pluto system, achieved in 2015. Commonly used isotopes may have an extreme range of half-life versus energy release ratios: Plutonium-238, for example, has a power density of 0.54 W/g and a half-life of 87.7 years and has been extensively used in the form of plutonium oxide as RTG fuel in recent space probes. In contrast, the short-lived Polonium-210 has a power density of 140 W/g and a half-life of only 138 days. The RTG powering the New Horizons probe employs 9.75 kg of plutonium oxide and had a power rating of 245.7W at launch, 202 W by the time it reached Pluto and is expected to be able to transmit until the 2030's due to the 3.5 W (0.8%) annual drop in power output. Opposite to the benefits is the notoriously low efficiency of this system: its fuel outputs approximately 4400 W of thermal power, of which less than 9% are converted to electrical power[88,89].

These types of onboard generators are currently limited to extreme or niche applications such as satellites, space probes, and remote facilities, due to their higher cost. Similarly to cases in which solar panels are used, this combination of battery bank and generator allow for extended operational periods while reducing risks related to refuelling with liquid combustible or corroding chemicals, the logistics involved in setting up and maintaining refuelling stations, the temperature and positional issues related to internal combustion engines, the absence of moving parts, besides the benefits of powering the complete robot - including control electronics, sensors and data transmitters - from a single source[89].

Following nuclear reactions in descending order of released energy come chemical reactions. Fuels in this category include some of the most energetically dense in modern daily use, and include substances from mineral, organic and synthetic sources. Energy can be extracted from these sources by means of combustion - which, being an oxidation reaction, requires and consumes oxygen in the process - and in-vivo metabolism - more specifically complex multi-step catabolic reactions, themselves including oxidation reactions.

The omnipresence of oxidation reactions gives rise to the common point to animal and plant metabolism: respiration. In parallel, apparatus meant to chemically extract energy

from consumable fuels must account for the necessary oxidation process. Space-faring engines are a notable example in that they must carry tanks filled with pure oxygen or other oxygen-containing molecule suitable as oxidizer. The purposeful injection of an oxidizer enables the reaction otherwise impossible in space vacuum, even if the fuel would combust with little to no extra cost in Earth's oxygen-rich atmosphere.

But oxygen is not the single complicating factor. Combustible fuels can be found in gaseous, liquid and solid states, making their density disparities extremely relevant, moreso in the field of mobile robots. Because of this disparity, specific energy and energy density of a substance are not directly proportional, requiring careful consideration in regards to the practicality of onboard storage and transport of each fuel. A clear example is the relation between energy outputs of Hydrogen and Methane combustion. Hydrogen combustion is the chemical reaction of highest energy output per unit mass, with a net specific energy of 119.93 kJ/g (1atm, 25oC). All reactions cited here refer to the LHV - Lower Heating Value -, or the usable energy provided by the combustion after subtracting the latent heat of vaporization of any vapour in the reaction products. Due to its extremely low density, hydrogen has a net energy density of only 10,050 kJ/m3 (1atm, 15oC). Compare these figures to those of the second most energetic combustible gas, the ubiquitous hydrocarbon Methane, that has a net specific energy of 50.02 kJ/g (1atm, 25oC) and a net energy density of 32,560 kJ/m3 (1atm, 15oC). Even if hydrogen is intrinsically more energetic, the relation between the densities make Methane more energetic per volume, when under the same conditions. The effective impact in design is a fuel tank three times smaller when using the less energetic Methane fuel, while maintaining the energetic autonomy granted if hydrogen was used. This example does not take into consideration the machinery required to produce work from the mentioned fuels, focusing on tank size for the sake of simplicity, but it is clear that for mobile robot designs - especially compact and airborne ones - the impact is far from negligible[90,91].

Liquid chemical fuels have greater energy densities, around three times higher than hydrogen, and pose handling and storage issues far less complex. Internal combustion engines are able to operate with a number of fuels and, for their large scale utilisation and long time since first being developed, are well understood and more easily serviceable. For onboard computing, navigation, and communications, though, the kinetic energy produced by these engines still must be converted into electricity. One common way to address this is by using alternators and dynamos - generators able to convert mechanical energy from a rotating engine shaft into alternating and direct current, respectively -, a practice implemented in passenger and cargo vehicles worldwide. Future mobile robotic systems also mix chemical fuels and electronics, pointing to the volumetric efficiency of chemical reactions. Notable examples include the cargo-bearing, self-navigating Big Dog, which is powered by a gasoline engine, and the speed-focused WildCat, which is able to

run on four legs at up to 32 km/h with a Methanol engine and hydraulic actuators. Both systems are under development by United States-based company Boston Dynamics.

Electrochemical reactions enable today's standard for mobile bots: batteries. These come in many sizes, shapes, weights, and energy storage capacity, and the trade-off between volume and capacity constitutes one of the biggest conundrums related to modern mobile electronics design.

Battery types are generally categorized by the chemical components responsible for the movement of electrons that characterize this technology. Many types have been developed over the years, especially since the inception of the first rechargeable batteries in the 1850's. Those were based on the reaction between a concentrated sulphuric acid solution and plates made of metallic lead and lead dioxide for the negative and positive electrodes of a galvanic cell, respectively. Today the term cell refers to the basic voltage-producing component of any type of multi-cell batteries. In the case of a lead-acid cell the reaction responsible for the electric current flow is based on the conversion of both electrodes into lead sulphate, which generates H+ ions and free electrons on the negative plate and consumes H+ ions and electrons on the positive plate. The reaction rate is self-regulating as the electrons produced on the negative side charge the plate, causing it to attract H+ ions and thereby creating an insulating layer if the solution is undisturbed. When electrons are allowed to flow (by connecting the battery to an electric load, for example) the reaction resumes at the same rate that electrons are sourced to the positive plate. There, the reaction between lead oxide and sulphate ions consume electrons and H+ ions, resulting in the formation of water molecules and the dilution of the acid electrolyte. This generates a concentration gradient of sulphuric acid inside the cell and the formation of convection currents, passively replenishing the electrolyte around the electrodes. The creation of the highly energetic H-O bonds found in the water molecule is the driving force of this type of battery. This cell is reversible, meaning that the system can be reverted to the original state - recharged - by reconverting the lead sulphate of both electrodes in the discharged state back to their original compositions. This is done by applying an electromotive force of reversed polarity to that of the cell, while varying the voltage to account for varying levels of conductivity displayed by the system. Non-rechargeable cells are known as primary cells, while rechargeable ones are referred to as secondary cells[92].

Even after more than a century from their initial implementation, lead-acid batteries are still in widespread use. This is due to their low cost and ability to provide short high-current outputs, useful for the operation of the starter motors of motor vehicles worldwide. Because of their ubiquity, capability of recharging, and relatively simple chemistry, they make for a reasonable illustration of some of the more advanced concepts regarding battery selection for designing mobile robots, the main interest of this work.

The electrical potential generated by the lead-acid chemistry is of 2.05 Volts per cell, and multiple cells can be connected in series to provide the appropriate voltages and in parallel to provide the required capacities for various applications. Each chemistry have a characteristic cell voltage and multi-cell battery modules can be configured according to standard voltages of modern microelectronic components (*eg.*: 3.3V, 9V, 12V), while multiple modules connected in parallel make battery packs applicable to high-power applications such as personal transportation and backup sourcing in case of power grid failures and blackouts. Besides varying in voltage, batteries also vary largely in total capacity and discharge rates. The relation between these values can be expressed as a C-rate, a single number that provides a useful measure of capacity at a given discharge rate. A 1C rate indicates that the designed discharge rate for a given battery will discharge it completely in 1 hour, so that a battery with a capacity of 100Amp-hours (Ah) can be discharged at a 100 A current without being damaged. Further, a battery of the same capacity but having a 5C rate could be discharged at 500 A, or if having a 0.5C could be discharged at 50 A, and so forth. Higher C-rates entail lower capacities dues to Power also is a factor of battery design and depends on chemistry and packaging. If power instead of capacity is the critical aspect of a design, the E-rate is a useful ratio: it indicates how much power any given battery can provide in 1 hour[93,94].

Some other common chemistries are Lithium-Ion (Li-ion), Lithium-Polymer (Li-po), Nickel-Cadmium (NiCd), Nickel-Metal Hydride (Ni-MH), Zinc-Air, and Nickel-Iron (Ni-Fe), among many others. In most secondary cell designs an electrolyte, usually in liquid or gel form, is necessary to mediate the electron flow inside the cell, allow for the inversion of the reaction during recharge, and equalize ionic gradients formed by reactions at the electrodes. That is not the case of primary cells, as they only go through one discharge cycle. Some technologies allow for continued use after this first discharge by means of mechanical recharging. Such is the case of zinc-air primary cells. These cells use a pellet or paste of zinc powder and aqueous potassium hydroxide as anode, oxidizing zinc with atmospheric oxygen to produce power. Once depleted, the anode can be replaced by removing zinc oxide formed during discharge and replenishing it with metallic zinc and electrolyte[93,95].

Specific to the field of mobile robotics, the ideal combination of battery features is defined, among other things, by the electrical load of motors and actuators, sensors and processing units, plus physical requirements dependant on the type of medium or terrain to be transposed - ground, liquid bodies at varying depths, aerial spaces at varying altitudes, or even deep vacuum. In accordance to the concepts previously discussed in regards to other energy sources, a robotics engineer or vehicle designer must consider how much energy can a battery provide, at the same time making sure that cell chemistry, packaging, weight and volume are appropriate for the particular design at hand. Even though specific energies and energy densities of cells can be calculated, they are not the

most useful way to synthesize the energetic capabilities of these power sources. In light of the complexity of any galvanic cell, their efficiencies are always lower than the theoretical output of their reactions - a battery is, in fact, a whole electronic circuit with internal electrical resistance, subjected to heat stress, having components that may accelerate or hamper physical processes like ionic and mass diffusion, and that may be susceptible to external contamination or improper use. Because of this, the effective measures of energy that can be stored and released by a battery to accomplish any given workload are the specific power and the power density.

In the realm of mechanical systems design, the physical properties of batteries also play a significant role. In UAV design, for example, chassis weight reduction is one of the main constraints. All other components maintained this translates into higher energy efficiency as the lower weights of static structures equals a better power/weight ratio. This effect becomes even more pronounced for multi-rotor builds like multicopters, VTOL (vertical take-off and landing) vehicles and hybrids capable of vertical launch and fixed-wing continuous flight, in which several components must be scaled in unison to account for higher lift power, and the change must be applied symmetrically throughout the chassis. The basic example is the propeller-motor-ESC (Electronic Speed Controller) combination: since these designs are built around fixed-pitch propellers, the differential lift necessary for attitude control of the vehicle is a factor of propeller speed. In order to operate in an RPM range that still provides enough headspace for the increase in lift necessary for manoeuvring, it is necessary to completely change the propellers if a higher payload is in order. As the propeller grows in mass or pitch, it eventually requires a more powerful motor for correct operation. Since the power of brushless motors common to this application is a direct factor of the electrical current range in which it operates, swapping the motor generally requires updating the ESCs to provide appropriate, higher current. All these changes must be applied to every lift point of the vehicle, usually ranging from 2 to 8, and incur considerable cost.

As design considerations go further, once troublesome properties of the battery pack employed can be turned into strengths. When strategically positioned, batteries can work, for example, as ballasts for aquatic and aerial robots, increasing stability. The insight of employing batteries as precisely positioned weights is also visible in complex walking robots, such as the platform used by Iida et al[96] for studying 4-legged robot rapid-locomotion. In this case, the weight of a battery was used in the morphological design and for tuning the body dynamics of the robot as a whole, enabling the use of a minimal control scheme to achieve stable gaits. The authors point to the importance of mechanical dynamics in adaptive locomotion and the related control systems. A notable example of adaptive locomotion emerged twelve years later, as Honda publicly announced its Riding Assist-e, an electric motorbike capable of balancing itself when at low speeds and of autonomously following a person at walking speed. The feature is useful for manoeuvring

in tight or crowded spaces and for providing safety and comfort to the rider during transitions between moving and stopped states. Its design purposely keeps its batteries in the lowest portion of the vehicle's body as a simple way to lower the centre of gravity and increase stability. At the same time, the front wheel assembly is morphologically dynamic, being able to change the fork angle to provide a larger, more stable wheelbase. The self-balancing control technology employed is based on the company's previous product research, namely the ASIMO bipedal robot.

Thus, the matter of power must be met with well-informed and deliberate decisions in the design of both physical and simulated robots, as to ensure feasibility and modelling relevance, respectively. The impact of such considerations is also discussed in greater depth in later sections.

2.4. Evolutionary Robotics

Evolutionary Robotics (ER) is described by Doncieux *et al*[97] as an area in the field of Robotics consisting of four main approaches, as follows: "(a) ER as an automatic parameter tuning procedure, which is the most mature application and is used to solve real robotics problem, (b) evolutionary-aided design, which may benefit the designer as an efficient tool to build robotic systems (c) ER for online adaptation, i.e. continuous adaptation to changing environment or robot features and (d) automatic synthesis, which corresponds to the automatic design of a mechatronic device and its control system"[97].

The evolutionary concepts may be applied to ER according to three basic perspectives: a) as the evolution of behaviour and control, the robot's "artificial brain"; b) as the evolution of physical body parts or c) as the co-evolution between body and brain. This third concept was first proposed by Karl Sims in 1994 when he presented the cutting-edge project Evolving Virtual Creatures (EVC) that deeply grounded the latter development on Evolutionary Robotics even now, more than two decades later[78,98].

2.4.1. Early Works

The structural concept of the EVC was named by Karl Sims as *connections* and *nodes*, where the morphology of the creatures is built from a recursive set of procedural instructions encoded by the virtual individual's genotype[78,98]. This system allows for variation of phenotypes across generations while still adhering to a standardization for programming simplicity (Figure 2.3).

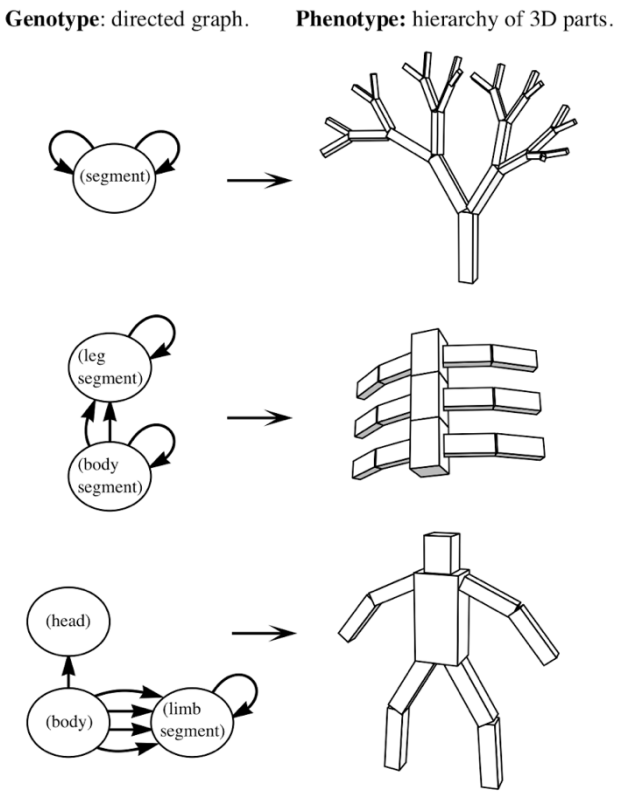


Figure 2.3 Three example creatures generated from procedural instructions encoded in the Genotype (and decoded into a Phenotype).

Left: graph of procedures. Right column: 3D representation of each genotype in the left column. Source: Sims[78].

As the body of the creature may constantly change, most traditional approaches to control would not be able to address such diversity. In order to handle different morphologies, the control of each joint is embodied in the more external body segment. Each joint also had an embedded neural node linked to the nodes of the adjacent parts and to the Central Nervous System - allowing each body segment to have a certain degree of autonomy (Fig. 2.3 and 2.4).

The parts of the body were equipped with three basic types of sensors: *joint angle sensors*, *contact sensors* (on each face of the body segment), and *photosensors* (any three together are capable of providing direction to the light). The sensory inputs are processed by neural circuits constituted of neurons (nested neural nodes), where each neural node performs a single function, where the available set of functions is composed of the following: *sum, product, divide, sum-threshold, greater-than, sign-of, min, max, abs, if, interpolate, sin, cos, tan, log, exp, sigmoid, integrate, differentiate, smooth, memory, oscillate-wave, and oscillate-saw.* Actions are performed by effectors connected to either neurons or to sensors from which they receive a value. This value is scaled by a constant value and then outputted as a movement of the body part.

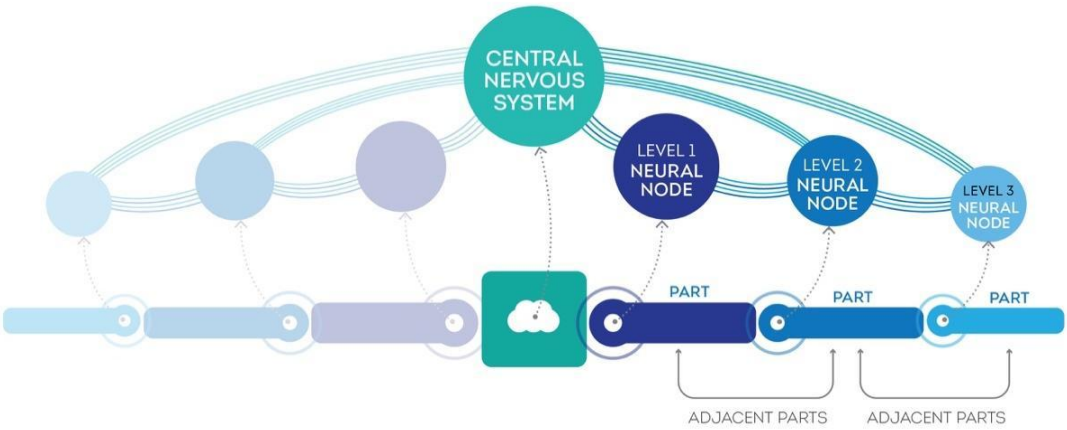


Figure 2.4 Diagram of hierarchy and articulation of body parts in the decision process of control described by Karl Sims[78]

Each neural node is embedded in the body part it is responsible for controlling.

The neural circuitry of individuals is a result of the wiring of the central neural system to the decentralized neural nodes, embedded on the body parts.

The evolution of behaviour was achieved by placing the creatures in a dynamically simulated virtual world and applying traditional methods from Evolutionary Computation. The methods for measuring fitness would be relative to the intended task, such as swimming, walking, jumping or following (an object). The fittest individuals of the population are allowed to breed and/or mutate, giving rise to the next generation. Across successive processes of selection, the fittest individuals' optimised behaviours emerged.

2.4.2. Recent Works

Given the complexity addressed by Sims when evolving body and behaviour together, especially if considering the available computational resources at the time, arguably some sorts of simplifications had to be adopted.

During the years following the publication of EVC by Sims, several researchers took the model some steps further by incrementing the original EVC model by adding complexity, physical accuracy, control, or improving evolutionary steps and procedures.

2.4.2.1. Open-ended ESP

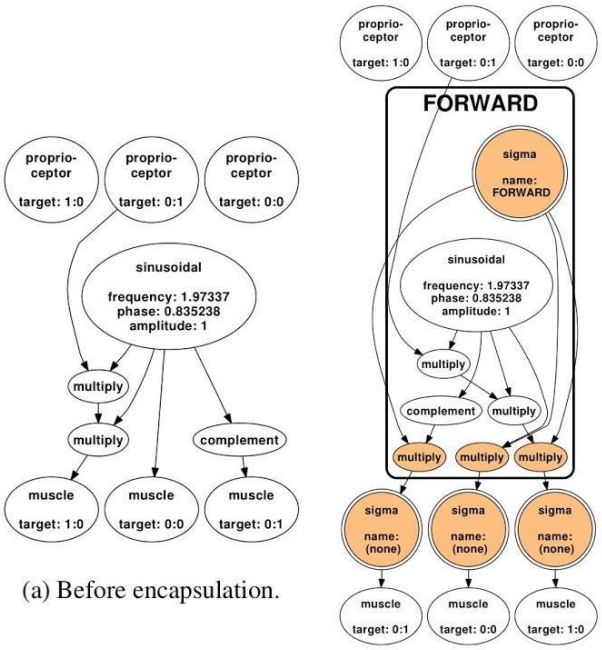
A novel methodological approach on Evolutionary Robotics was first presented and successfully implemented by Lessin *et al*[79]. The open-ended method ESP, named after its three components *Encapsulation*, *Syllabus* and *Pandemonium*, intends to allow complex behaviours to emerge from the simulation of evolving virtual creatures. Similar to Sim's

EVC, Lessin's virtual creatures are intended to develop locomotor skills through simultaneously evolving body and control from a finite set of body parts and 'neural' mathematical operations[79–82]. The step ahead comes with the further human-assisted steps that work on consolidating learned skills and preserving them across the next steps.

Encapsulation: This method preserves learned skills ensuring they will persist throughout evolution by encapsulating both the control skills (neural nodes) and the corresponding body structures. Besides locking the controlling system of the new skill, this human performed procedure adds sigma nodes that multiply input(s) and output(s) working similarly to a switcher and tuning knob (Figure 2.5).

Syllabus: The (human designed) syllabus consists on a hierarchically ordered set of skills from simple to complex levels aimed at assisting the evolutionary process by dividing it into steps. In the model described by Lessin *et al*[80] the creatures are meant to develop basic (low level) skills first to move forward. By gradually raising the complexity of tasks the system sets attainable goals and allows for the mastering of each particular skill, which will later be required to achieve further complex goals. This ordering of steps is also required for optimising time and computational costs. We may point out, however, that this tool must be carefully approached since over-mechanistic paths may trap diversity, eventually leading to ordinary results.

Pandemonium: The third important procedure implemented in ESP is Pandemonium, on which the Syllabus designer has to pick abilities that are mutually exclusive, configuring them as a pandemonium relationship to each other (e.g.: left and right turn). This tool prevents decentralized controllers within the creatures from sabotaging themselves (Figure 2.6).



(a) Before encapsulation.

(b) After encapsulation (with new nodes shaded).

Figure 2.5 Illustration of an encapsulation process of a control system described by Lessin *et al*[80].

(a) artificially evolved neural nodes before encapsulation. (b) wiring of the neural nodes after encapsulation: sigma and multiplier units are added to the model in order to permit modulation. Source: Lessin *et al*[80]

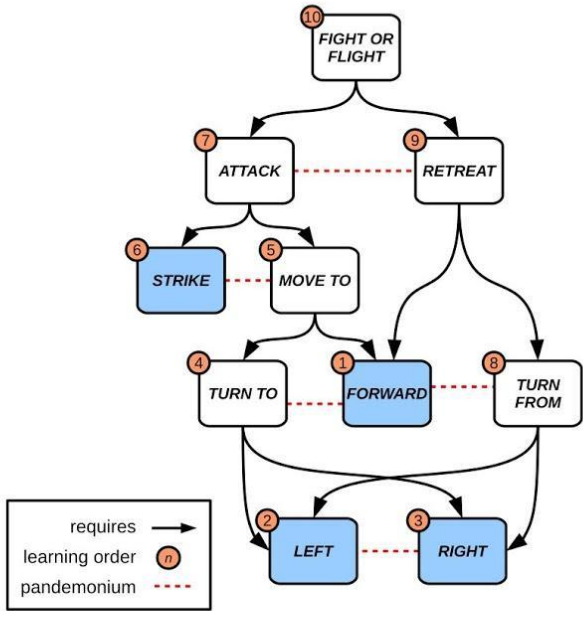
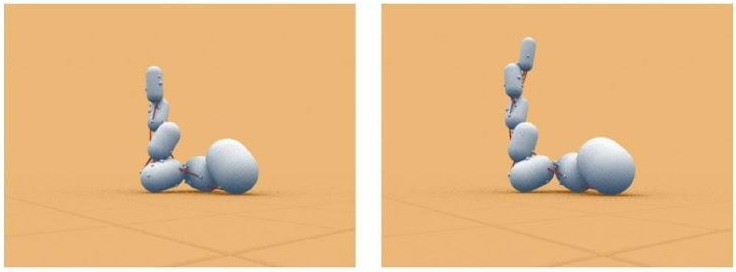


Figure 2.6 Example of Syllabus implemented by Lessin *et al*[80].

The complexity of elements grows bottom up. Complex actions often require more basic actions (straight line arrows). Self-excluding actions are classified as pandemonium relationships (red dotted lines). The learning order (orange "n" circle) does not have to necessarily follow the syllabus from the bottom up. Source: Lessin *et al*[80].

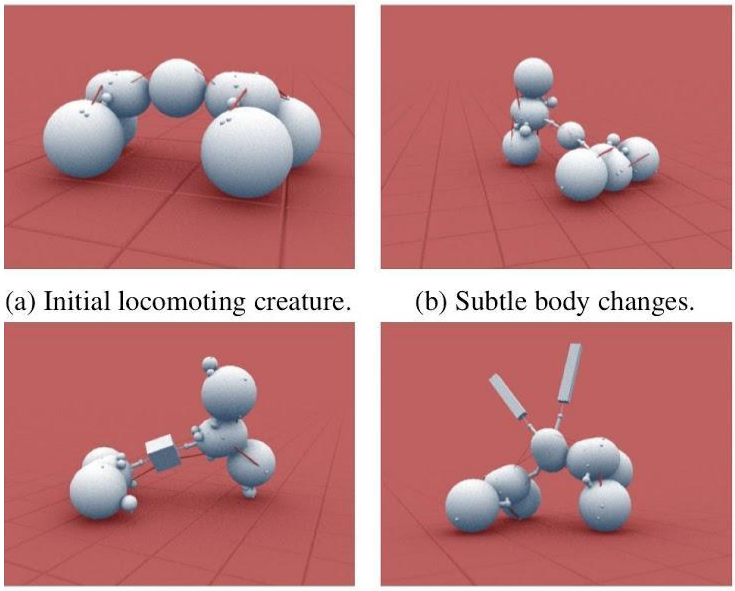
A further development for the ESP method[80] implements an extra step after the development of skills when the creatures are again allowed to evolve their bodies as long as already learnt skills are not lost and their efficiency is not decreased. This novel step presented successful results ranging from slight but fruitful adjustments in the body (Figure 2.7) to very substantial changes in the body, further increasing diversity (Figure 2.8).



(a) Original ESP result. (b) Result in new ESP.

Figure 2.7 The Extended ESP method implements an extra step in which artificial animals are allowed to evolve their bodies after the evolution of the control system.

This step may result into small but significant improvement towards a "fine tuning" of the shape. This figure shows an individual before (a) and after (b) this process. Source: Lessin *et al*[80].



(a) Initial locomoting creature. (b) Subtle body changes.

(c) More obvious body changes. (d) Dramatic changes in morphology.

Figure 2.8 The Extended ESP method[80] implements an extra step in which artificial animals are allowed to evolve their bodies after the evolution of the control system.

A controlling mechanism prevents the species to lose already learnt abilities or to have their efficiency reduced. Figure shows an initially evolved creature's body (a) with subtle (b), more obvious (c) and dramatic (d) body changes, as a result of the late step for morphological improving. Source: Lessin *et al*[80].

2.5. Artificial Intelligence and Optimisation

Artificial Intelligence (AI) was a term coined by John McCarthy in 1955 in reference to the study and development of intelligent systems, able to mimic brain abilities such as planning, problem-solving, and reasoning[99,100]. The seeds of the field date back to the years 1940-1950 especially with McCulloch and Pitts' *Boolean Circuit Model of Brain*, in 1943, and Turing's *Computing Machinery and Intelligence*, in 1950.

The excitement about AI grew up in the years 1950-1970 with the development of the early AI programs such as Samuel's checkers program, Newell & Simon's Logic Theorist, Gelernter's Geometry Engine, in 1950 and Robinson's complete algorithm for logical reasoning, in 1965. Soon in the years from 1970 to 1980 knowledge-based systems started being built and the high expectations on the field lead to a boom on industry. From late 1980's, however, the AI industry experienced a cold shower from the realization that the available technological tools were incapable of supporting the development of the ambitious plans of the time. The exponential growth of data with the increasingly noticeable emergence of the Information Age also required a new approach to AI. The paradigm shift came in the 1990's with the early development of AI systems supported by Statistics[99,100].

Historically, AI has been defined by four approaches: *Thinking Humanly*, *Acting Humanly*, *Thinking Rationally*, and *Acting Rationally*. The human-centred approach intends to recreate human-like behaviours based on empirical observations and hypotheses. The success of a model is measured in terms of its similarity to human cognition or behaviour[a] [101]. Rationalist approaches, on the other hand, involve a combination between Logic, Mathematics and Engineering, with its performance measured based on how close it is from the ideal[100,102].

2.5.1. Intelligent Agents

Modern approaches to Artificial Intelligence are centred in the concept of the Agent, a computational entity that acts based on internal and environmental states. The AI agent is expected to examine the given conditions and autonomously act in a way to maximise the expected outcomes - this is the definition of a Rational Agent[100].

An *Agent* must be capable of perceiving its *Environment* through *Sensors* and act upon this same environment using *Actuators*. The function or behaviour of an agent is

[a] The most traditional test for evaluating AI systems intended to act humanly was developed by Alan Turing in 1950 and remains relevant even 60 year later. The AI system passes the test if, after asking some questions, a human interrogator is still not able to determine whether it is a computer or a human who is writing the answers[101].

implemented as the *Agent Program*. The Agent may keep track of internal information about such as its own condition (e.g. battery level) and external data input - known by the term *Percept*. The combination of internal and input information composes the agent's *State*[29].

The theory behind the behaviour expected from Intelligent Agents look simple at first sight. However, real-world applications often deal with increasing levels of uncertainty in which estimating the outcomes of actions must not be that simple. Diverse approaches have been developed in order to address the increasing complexity presented by different systems, ranging from simple responses to the current stimuli through to memory-based decision processes.

Agents can be either Reflex or Planning Agents depending on its procedures to plan and take actions. Reflex Agents choose their actions based on their internal state and their current perception of the environment. This type of Agent can have a model of the environment based on states and keep track of their actions in the past and even learn from them. However, the concept of Reflex Agents refers to the fact that they do not consider the future consequences of their acts. Planning Agents, on the other hand, plan ahead and take actions based on their possible consequences in the future. The modelling of agent's possible actions and the estimated outcomes of each particular action is based on mathematical and statistical models that can generally be optimized with the implementation of fitness evaluation[29].

At this point it is important to highlight the important distinction between the actual performance of an Agent, and its own measure of performance. The actual performance could only be measured when all the conditions of the Environment are known, whilst the measured performance by the agent itself is nothing but another component of its state of learning plan. In a real problem, measuring the actual performance of an AI agent would be often hard when not impossible. Given the need of assessing the performance of Agents, a considerable part of the development is made in simulated environments. The computer-controlled simulation may provide different scenarios such as varying distributions of resources given, for example, by different *Test Functions* (*Benchmark Functions,* or *Artificial Landscapes*), where all the conditions are known[103].

2.5.2. Problem-solving with AI

Search problems are the core of AI, usually involving a starting state, a set of possible steps (world states), and a Goal. Search problems are usually represented by means of a State Space Graph, where each possibility (*node*) is only represented once and connected to all the possible next steps. SSGs may be expanded into Search Trees: starting from the very top that represents the starting state, each tree node is linked to its *fringe*: a set of the possible next steps. As the complexity of a problem increases, the tree size will follow

- leading eventually to infinite-sized trees. Thus, this graphical tool is only applicable on solving simple problems with few possible states[104].

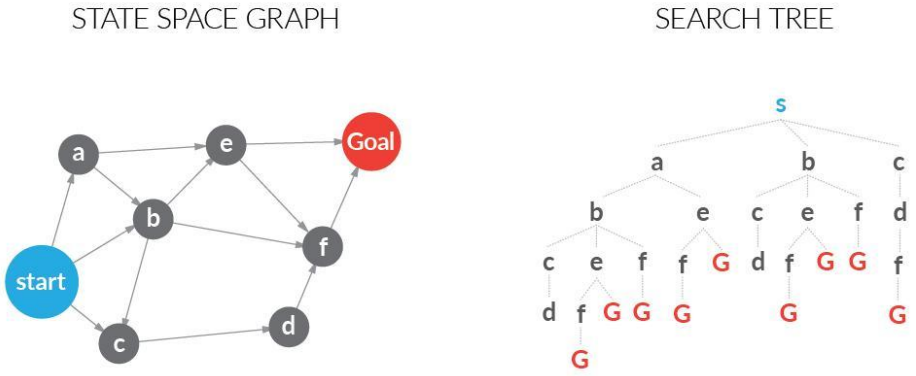


Figure 2.9 Two representations of a same search problem through State Space Graph and Search Tree.

Understanding search problems and methods for solving them may require graphical tools to explore the concepts. Some useful graphical tools are *State Space Graphs* and *Search Trees* - mathematical representations of a search problem (Figure 2.9).

Since its early days the field of Artificial Intelligence has been involved in the development of search methods as the framework of problem-solving basically composed by two perspectives: the traditional *Neat AI* and the newer *Scruffy AI*[100,102,104].

Neat AI approaches the problem from a top-down perspective, grounded in logical processes that rationally explain why the problem-solving system works. This paradigm comprises two basic groups of search strategies, Uninformed (Blind) and Informed (Heuristic) search. The first group refers to search algorithms that are given no further information about the problem other than if the goal was reached or not. Effective but usually costly, these search methods could be improved by the addition of Heuristics, a method to equip the algorithms with information about where the goals could be by estimating how close a state is to the goal[100,102,104].

Although Neat AI may provide precise and optimal solutions, this approach becomes costly in terms of computer power, and ultimately unmanageable, as the complexity of problems increase. In this context, the newer concept of Metaheuristics leads the field towards a big leap forward on the pursuit of solutions for multi-dimensional and complex problems. This new paradigm, named Scruffy AI, came up with alternative search methods relying on probabilistic and stochastic methods able to find near-optimal solutions with a reasonable cost of resources[100,102,104].

2.5.2.1. Search Algorithms

Neat AI: Uninformed ("Blind") Search: The traditional approaches on Uninformed Search consist on organizing the problem in a search tree and later expand it according to some criteria, being the strategies classified into: Depth-First Search, Breadth-First Search and Uniform-Cost Search[100,102,104].

> Depth-First Search starts from the very top of a tree, arbitrarily expanding a node in the current fringe at each iteration until it finds a dead end or the goal. When the goal is reached the algorithm stops. This algorithm is relatively efficient on returning a solution, but it does not consider other possible solutions neither the cost values so it will not necessarily find the optimal solution for a problem[100,102,104].
>
> Breadth-First Search starts from the top and explores the whole fringe of each node at each iteration until it finds a solution. The solution found will be the least costly, but this algorithm will probably take a long time to run[100,102,104].
>
> Uniform-Cost Search evaluates each node of the fringe according to the cost to take it (e.g. distance from the current node), named *path cost function*, *g(n)*. The algorithm will go deep from less to more costly paths until it finds the *optimal solution*, considering not just the number of steps taken to find the solution but also the cost associated to each of[100,102,104].

Neat AI: Informed (Heuristics) Search: As the term suggests, in this search strategy the programs are equipped with evaluative tools built according to specific knowledge about the problem. The core concept underlying Informed Search methods is *Heuristics*: a mathematical technique aimed at estimating how close each step/state is to the solution[100,102,104].

While in non-heuristic methods the algorithms would just be able to evaluate whether or not a goal has been achieved, in heuristic-based methods the agents have a clue as to how close the solution might be. Naturally, the better the heuristic is, the higher the efficiency will be. Thus, an essential concept is the idea of Admissibility of a Heuristics: *optimistic (admissible) heuristics* aim at slowing down bad plans while still leaving some degree of freedom for the agent to find the optimal path. When true costs are overestimated as in *pessimistic (inadmissible) heuristics*, good plans may be trapped thus breaking optimality of the algorithm[100,102,104].

The main methods on Informed Search are Greedy Best-first Search and A* Search, this latter combines Greedy Search and Uniform-cost Search being capable of always finding the Optimal Solution.

> Greedy Best-first Search (or simply Greedy Search) is another instance of Tree-Search and/or Graph-Search algorithms with the application of the *heuristic*

function as the *evaluation function*. Starting from the top, this algorithm evaluates each node on a fringe, expanding the one that looks closer to the goal. When correctly applied, this method may quickly find a good solution, however it does not necessarily find the cheapest solution[100,102,104].

A* Search (pronounced "A-star") combines the efficiency of Greedy Best-first Search on quickly finding a solution with the optimality of Uniform-Cost Search. For A* Search algorithms the *evaluation function*, or *cost function* applied consists on a combination between the *heuristic function* and the *path cost measurement*[100,102,104].

Scruffy AI: Natural Computing: The interdisciplinary field of Natural Computing encompasses different aims and approaches, being usually described as having three related fields: Computationally Motivated Biology, Computing with Biology and Biologically Inspired Computing[100,102,104].

Computationally Motivated Biology, as the name suggests, aims at better understanding biological systems by means of computational tools. This field is focused on the simulation of natural phenomena rather than developing mathematical or engineering tools and is closely related to those of Bioinformatics and Computational Biology[100].

Computing with Biology addresses the study of biological substrates or platforms, such as molecules or DNA, as an alternative for implementing computation.

Biologically Inspired Computation aims at finding solutions to computation problems by applying procedures and processes abstracted from nature.

Scruffy AI: Computational Intelligence:

Computational Intelligence is a subfield of Artificial Intelligence aiming at developing adaptive and intelligent systems. Some of the main disciplines composing the field are: Evolutionary Computation, Swarm Intelligence, Fuzzy Intelligence, Artificial Immune Systems and Artificial Neural Networks[102,104–106].

Evolutionary Computation aims at mimicking aspects of biological evolution by natural selection described by neo-Darwinian Theory of Natural Evolution. Some of the most popular evolutionary algorithms are Genetic Algorithms and Differential Evolution, often applied for optimisation with multiple objectives. Genetic Algorithms are based on divergence and convergence of solutions resulting from mating and mutation procedures across generations. Some of the main tools and procedures applied by GAs are breeding, mutation and elitism. Differential Evolution employs most of the methods of GA but modifies the mating procedures and introduces criteria for incorporating or not incorporating the new individual in the next generation[102,104–106].

Swarm Intelligence algorithms were developed based on the solutions emerging from collective behaviour of a group of individuals. The two most popular paradigms in this category are Particle Swarm Optimisation (PSO) and Ant Colony Optimisation (ACO). PSO imitates collectives such as bird flocks and fish shoals, where the emerging behaviour surpasses the simple movement of each particle (animal) individually. ACO emulates the social behaviour and communication among insects to find shorter paths to a solution. Similar to searching individuals from an ant colony releasing and following pheromone trails (that evaporate over time), artificial agents on this algorithm end up concentrating on the most efficient paths[102,104–106].

Artificial Neural Networks (or simply *Neural Networks*) is a *Machine Learning* model aimed at mimicking the biological evidences explaining the natural learning abilities and the wiring in the brain, through the formation of neuronal paths. Neural Network algorithms are often applied to function approximation and pattern recognition in adaptive learning systems[102,104–106].

Fuzzy Intelligence is a reasoning strategy founded on fuzzy logic, in which variables and truth values may be expressed as a range of values from 0 to 1, varying from completely false to completely true, rather than pure definitions of true or false in Boolean Logic (either 0 or 1). Fuzzy Intelligence is often applied in control systems, classification and learning systems[102,104–106].

Artificial Immune Systems are inspired by biological theories and models that explain the adaptive behaviour of mammalian immune systems. Being robust and effective as vertebrate nervous system, immune systems are, though, decentralized and have the property of self-organisation. Some of the traditional AIS algorithms are: Clonal Selection, Negative Selection, Immune Networks and Dendritic Cells[102,104–106].

Chapter 3
Literature Review in Ecology, Movement and Behaviour

3.1. Animal Behaviour and Movement Ecology

All organisms move at some stage of their life cycle; from protozoans to huge whales. Even the organisms and organic structures which are unable to actively move (e.g. bacteria, viruses and plant seeds) depend on the movement of the media to succeed[107]. Some species from the Phylum *Cnidaria* can even alternate between sessile polyps and swimming *medusae* among generations - a phenomenon denominated metagenesis. Considering the dynamics of flow in liquid environments sessile life seems to be more fruitful in aquatic systems[107] being these the only environments where we find sessile animals (*Phylum Animalia*).

At first sight active movement seems to be a superior ability and yet plants are ancient, successful, and essential forms of life. Reduced movement may be an advantageous strategy to avoid energy expenditure as long as it meets the species' energy requirements. As primary producers on grazing food webs plants synthesize biomass from energy and nutrients obtained from sunlight, air, water and soil. Similarly, in a detrital food chain, bacteria and fungi break down dead organic matter transforming it into biomass and passing it ahead to the other trophic[b] levels. The abundant and relatively constant availability of the above-mentioned resources explains the widespread distribution of vegetation, bacteria and fungi all around the globe even in extreme environments[c], with a few exceptions.

Besides a simple proof of success, the widespread distribution of autotrophic[d] individuals is nonetheless one of the reasons behind the diversity of species. Considering the energy loss along food webs the amount of biomass will deeply decrease on each new trophic level. In a simplified picture the food web biomass could be represented by a pyramid with a large base. In other words: the larger the ground-level of the pyramid is the higher the pyramid can be[31,108] (Figure 3.1).

---

[b] Trophic level refers to the role each species or group of species performs in a food web.

[c] Tundra is the coldest terrestrial biome, characterized by average temperatures of 6ºC.

[d] The individuals in the basis of the food webs which produce biomass from inorganic or dead organic matter.

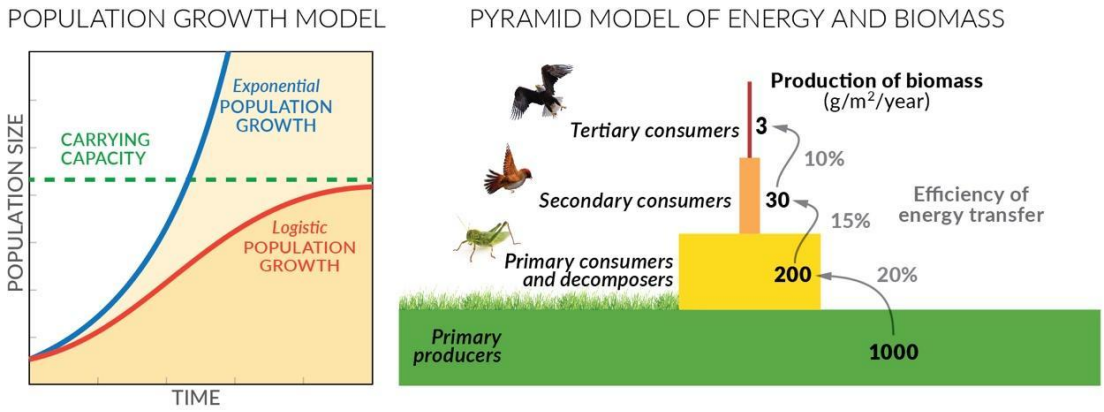


Figure 3.1 Population Growth Model and Energy Pyramid.

Left: Population Growth Model: populations of living organisms tend to grow exponentially however, the carrying capacity of the environment limits population growth to logistic. Right: Production of Biomass and Energy Transfer along the Energy Pyramid model - from primary producers to primary consumers and decomposers to secondary and tertiary consumers. Source: Created by author based on References[31,108].

In general terms, the higher the animal is the food web the larger its habitat has to be in order to meet its needed dietary requirements. The size of the habitat will also be influenced by the animal's size, physiology, feeding habits (herbivore, carnivore, omnivore), and biomechanical properties. Among others, the above-mentioned properties are encompassed by the concept of *Carrying Capacity* of the environment which consists of a limiting factor for the growth of the population explaining why population dynamics draws a logistic curve rather than exponential[31].

Attending its basic metabolic needs (such as obtaining food and water) is definitely a main motivator for animal movement, but there are other motivators such as the individual's will towards mating. Feeding and mating can be thus considered the two major and common life-lasting challenges for individuals and therefore for species. The biological necessity of interactions or "encounters" do not explain why movement first emerged in nature but certainly explain why moving animals may move[30].

Apart from the encounter-driven movements, other factors driving animal's movements are: avoiding threats and/or predators, regulating the body's homeostasis on a daily basis or avoiding major seasonal events such as harsh winters, hot summers, or dry seasons.

In spite of being the manner by which moving animals obtain nutrients, moving is an energy-consuming task itself, leading to a trade-off. This situation led ecologists to develop the concept of Optimal Foraging Theory (OFT), back in 1966. The idea is grounded on the hypothesis that the evolutionary survival of moving species is based in part on the individual's ability to forage for resources and to encounter mates. The most efficient (fittest) behaviours, measured by the energy or nutrient intake per unit of effort, are thus expected to have been selected[29–31].

3.1.1. Basic Concepts and Terminology

Animal's movement can be observed on the most diverse, sometimes surprising, scales: from bees performing a waggle dance within a beehive, through mammals foraging locally and then homing to feed their offspring, to cross-continental migrations completed yearly by certain species of birds. In order to ensure the understanding and to improve the quality of debate most authors highlight the importance of the organization of commonly used terms in the field.

Hansson and Åkesson[107] propose a broad terminology for movement consisting of: Movement, Dispersal, Migration, Homing, and Foraging Movements (Table 3.1). Expanding this previous categorization system, similar schemes of animal movement are proposed by Fox *et al*[109] and Dingle[110], as can be seen in Table 3.2 and Table 3, respectively. Dingle[110] proposes a third category for movements not under control of the organisms encompassing both Accidental Displacement classified by Fox *et al*[109] as "not directly responsive to resources or home range" (Table 3.3).

Table 3.1 Basic terminology for animal movement proposed by Hansson and Åkesson[107].

Movement	Individuals or populations (or parts of populations) that change position at any temporal or spatial scale. Movement includes all other ways of displacement.
Dispersal	Individuals or populations (or parts of populations) that move to reach new areas, but do not return.
Migration	Individuals or populations (or parts of populations) that move between two well-defined habitats on a temporally (reasonably) predictable basis. Migration includes, e.g., the seasonal migrations of birds between wintering and reproduction areas, fish migrations from lakes to streams, but also the once-a-lifetime migration of eels from freshwaters to their natal marine habitat to spawn and then die.
Homing	Refers to when an animal returns to a known goal, e.g. its home.
Foraging movements	Individuals that move between resting places, nest sites, etc., and feeding grounds in a temporally reasonably predictable way, e.g. bees moving from flower fields to the hive or bird parents feeding their nestlings. These types of movements are difficult to distinguish from migration, but generally occur at a shorter time scale. The most striking difficulty when distinguishing between migration and feeding movements is the very well known diel vertical migrations (DVM) of aquatic zooplankton. This type of movements may actually fit better as feeding movement or as diel vertical movements than as actual migration. However, the term diel vertical migration is so established as a research area that any attempts to re-categorize it would lead to a revolution. Hence, in order to avoid such responses, we will here adopt the traditional view while noting the difficulties with such distinctions.

Table 3.2 Classification of animal movement proposed by Fox *et al*[109]:

MOVEMENT	CHARACTERISTICS	EXAMPLES
Movements that are home-range or resource-directed		
A. Station keeping	Movement keeps organism in home range.	
1. Kineses	Changes in rate of movement or turning.	Planarian in shadow.
2. Taxes	Directed movement in response to a stimulus.	Upwind flight of moth in pheromone "plume".
3. Foraging	Movement in search of resources. Ceases when resource encountered.	Search for food, mate, shelter, oviposition site; parasite or parasitoid host seeking.
4. Commuting	Periodic (often daily) forages in search of resources. Ceases when resource encountered.	Albatrosses foraging; vertical "'migration" of plankton.
5. Territorial Behaviour	Movement and agonistic behaviour directed toward intruders in territory. Ceases when intruder leaves.	Many.
B. Ranging	Movement over an area so as to explore it. Ceases when new home range/territory is found.	Dispersal of some mammals; "natal dispersal" of birds.
Movements not directly responsive to resources or home range		
A. Migration	Undistracted movement with cessation primed by movement itself. Responses to resources / home range suspended or suppressed.	Annual journeys of birds, insects etc. Flight of aphids to new hosts. Transport of some seeds to germination sites.
B. Accidental Displacement	Organism does not initiate movement. Ceases when leaves transporting vehicle.	Storm vagrancy.

Table 3.3 Classification of movement in nature proposed by Dingle[110]

MOVEMENT	CHARACTERISTICS	EXAMPLES
Movements home range or resource-directed		
Station keeping	Movements keeping organism in home range	
• **Kineses**	Changes in rate of movement or turning	Moth in a pheromone "plume".
• **Taxes**	Directed movement toward a stimulus source	Insect moving toward a light (positive phototaxis)
• **Foraging**	Movement in search of resources; movement stops when resource encountered	Movement in search of food, mate nesting or oviposition site (animals); modular growth (plants, corals)
• **Commuting**	Movement in search of resources on a regular short-term basis (usually daily or a few days); ceases when resource encountered	Albatross foraging; vertical migration in plankton
Territorial Behaviour	Patrolling territorial boundary, agonistic: response to neighbours and/or intruders (stops when intruder leaves)	Many examples across taxa
Ranging	Movement over a habitat to explore it; ceases when suitable home range located	"Dispersal" of some mammals; natal dispersal" of birds; parasite host seeking
Movements not directly responsive to resources or home range		
Migration	Undistracted movement to new habitat; cessation promoted by movement itself. Responses to resources or home range suspended or suppressed	Annual flights of birds to breeding grounds; flight of aphids to new hosts; movements to breed of diadromous fish; "dispersal" of some seeds

Movements not under control of organism		
Accidental Displacement	Organism does not initiate movement. Movement stops when organism leaves transporting vehicle	Storm vagrancy
"Assisted Migration" or Assisted transport	Accidental or deliberate anthropogenic movement	Horticultural or weedy introductions; biological control species; conservation introductions

3.1.2. Movement Ecology

The interdisciplinary field of Movement Ecology aims at investigating the factors underlying the movements of organisms as well as the trajectories and resulting patterns, among other topics. Some of the motivations for the increasing interest in this field are empirical observations pointing to the existence of patterns in organisms' movements, from rudimentary protozoans to human beings[30].

Random walks and biological encounters are described as reaction-diffusion processes, composed of a diffusive component (e.g. exploratory searches) and a reactive component arising from the interaction with other system elements (e.g. chasing prey). As pointed by Viswanathan *et al*[30] diffusive and reactive processes fundamentally diverge from each other, being guided by different rules. Diffusive processes are linear and could lead to patterns such as Brownian, Random walks, Lévy walks or Lévy Flights (Figure 3.2) whilst Reactive processes rely on interactions with the environment, leading to nonlinear effects. Taking into account the distinction between these two processes could be essential when analysing animal tracking data in order to obtain a tagged dataset. According to the some of the most prominent works on the area of animal movement, the field tracking data of animals usually results in cumulative probability distributions other than the traditional Gaussian (Normal) and Brownian (Uniform), being Lévy Walk, a specific Power Law distribution, the most common.

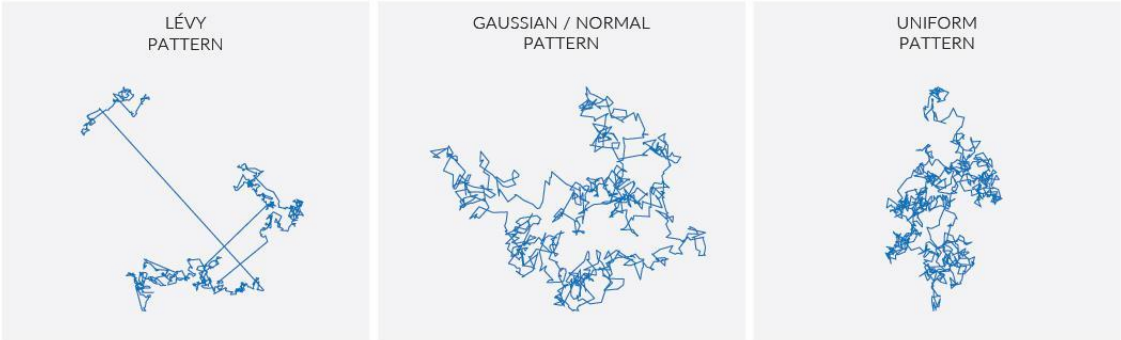


Figure 3.2 Three stochastic moving patterns given by different distributions: Lévy Walks, Gaussian and Uniform.

It is possible that almost all the trajectories of animals would be at some level permeated by interactions with non-desired or even facilitators of the movement. Some forest

animals will have to avoid physical obstacles such as trees in their way, while the movements of animals of arboreal locomotion will be dependent on the distribution of these trees. In the same way, tracking the movements of fishes in a river has to take into account the water movement as in a study developed by Sparrevohn *et al*[111].

Beyond the statistical processing of the results, a major challenge still lies in the preliminary stage of data acquisition. Although it is evolving fast, the available technology still does not allow researchers to track a large number of species, such as small insects or ground burrowing animals due to the size, weight, or signalling requirements of transmitters.

Against all odds, several researches had been successfully undertaken with the outcomes revealing similar results.

3.1.2.1. Tracking Animal Movement

In the field of animal movement, the recent advances in technologies for tracking (such as micro and nanotechnology combined with wireless and satellite communication protocols) made possible to build long-lasting tracking devices capable of keeping track of animal movements across different scales. Animal movement ranges from small movements such as of termites inside a termite mound, or the daily flights of bees looking for nectar, to large-scale movements, across countries or even continents, such as sea turtles on coastal environments or large migratory leaps of migratory birds. Beyond the new possibility of tracking what in the past would seem impossible, increasing resolution and time scale on tracking experiments lead to new discoveries about interactions among groups or species[111,112].

Pigeon flocks, for example, were found to have consistent leadership hierarchies with some individuals influencing the collective movement decisions more than others[113]. Sheep groups on the other hand were shown to respond to global group structures rather than local cues[114]. The possibility of tracking a bigger number of individuals also revealed a considerable diversity in foraging and niche exploration among individuals from the same group or species[115]. Big tracking data also revealed dynamics of interaction, habitat use and patterns of organization among different groups of a same species or from different species, such as territoriality[116] or space partitioning[117–119].

3.1.2.2. Migration

Usually in order to avoid starvation, extreme temperatures or to find more suitable environments to breed, migration is a ubiquitous behaviour in nature. It has been reported in mammals, birds, fishes, reptiles, amphibians, molluscs, arthropods, and surprisingly even in algal protists[107].

Migration also occurs over diverse scales ranging from the outstanding journey of Arctic terns (*Sterna paradise*) flying from high Arctic to South Pole, to the daily vertical migrations of zooplanktons consisting of few meters[107]. Some migrations can also take more than the individual lifespan to be completed such as the transgenerational migration of the painted lady butterfly (*Vanessa cardui*) that takes six generations to complete a cycle from Europe to Africa and back again[107].

The migratory behaviour of individuals may also vary among the same population and in some extreme cases a certain population can be composed of both migrants and entirely resident individuals, what is referred to as Partial Migration[107]. There are three registered types of seasonal partial migration: fractions of the group may share the same habitat on either breeding season or non-breeding season, or in a third case some fraction of the group may skip spawning in certain years thus not coinciding necessarily with the other fraction of the population. Another within-population migratory pattern named Differential Migration describes the variation on timing or destination of fractions of the same population.

Hansson and Åkesson[107] illustrate the concept of the three main patterns of latitudinal population movements performed by different populations of the same species: chain migration, leapfrog migration and telescopic migration. Some animal populations may also migrate regularly in a seasonal cycle but from time to time vary their migratory path according to other major environmental changes ruled by longer periods of time.

The journey size is not necessarily proportional to the size of the animal. However, as it involves the ability to navigate and to move in an optimised manner, the animal's physiology and biomechanics will naturally play a strategic role.

The migration phenomena can be generally explained as a behavioural strategy to deal with drastic weather changes (beyond the physiological limits of the animal) or with an increased competition due to limited resources (forcing poor competitors to seasonally search for new places). For those that remain in the site, physiological differences may account for their ability to deal with extreme changes, allowing the individuals to survive in dry, hot, or cold environments. Similarly, physiological and body adaptations enable individuals to endure exhausting migration journeys to escape the seasonal variation. Ability or luck on finding best paths and sites to migrate may also be rewarded by evolution as well as timing to get to a certain breeding point. The ones that arrive early pick the best breeding sites and promote the evolutionary survival of their genotypes.

3.1.3. Animal Navigation and Orientation

Understanding animal movement from small to large scale requires a comprehension of sensory mechanisms of orientation and navigation, and mind maps - abstract

representations of space inside animal's brains. Some of the questions about how animals find their way have been asked and investigated for centuries, but other advances in this field could only be made recently with the improvement and availability of long-lasting global tracking resources. Although advances have been made there is still much to be yet discovered. As an example, some of the navigational mechanisms (e.g.: Magnetic Sense) have already been identified but the receptors behind them still remain not fully known.

3.1.3.1. Navigation

Navigation describes the process of moving from one place to another - which may be performed by accessing mental land representations, and on external cues given by sensory apparatus or "biological compasses" of orientation (Figure 3.3).

Some cognitive abilities such as path integration (dead reckoning) may help animals to orientate themselves, even for those which experiments suggest not relying on maps for navigation. Among the animals that do not have the ability to build mental land maps, some have been proven to navigate following sequential lists of steps based on self-generated movements tracked, updated by environmental cues through path integration (Figure 3.3).

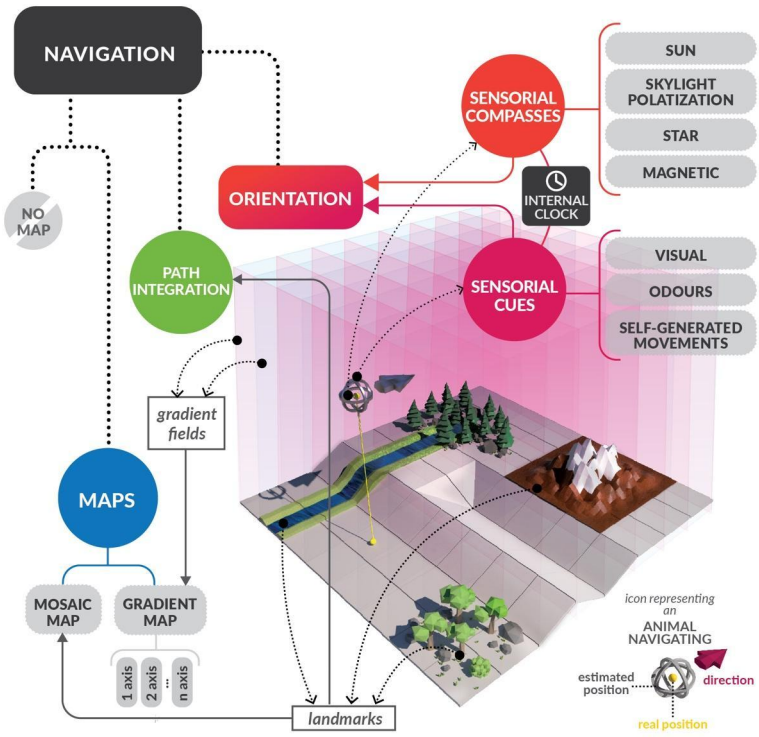


Figure 3.3 Diagram of biological mechanisms involved in animal navigation.

> Navigating presumes a pair of origin (relevant or not) and destination (known or not). Animal's cognitive capacity and sensorial features will provide them with the ability to navigate utilizing maps (mosaic or gradient) and orientation mechanisms such as biological compasses and sensorial cues. The cognitive mechanism of path integration (dead reckoning) improves the navigational ability of an organism's by either updating internal maps or readjusting the route according to previously known environmental clues.

Some animals are born with innate information about where to go, as shown by experiments with socially isolated birds, while some others use learning abilities to perceive and record places of interest such as immediate or long-lasting food and water resources, nesting sites, dangerous places, and so on.

3.1.3.2. Orientation

Some of the sensory mechanisms identified as being involved in animal orientation are organized by Hansson and Åkesson[107] into Biological Compasses and Sensorial Cues (Table 3.4). Biological Compasses enable animals to geographically orientate themselves, in a global reference frame, based on geological and atmospheric features such as the position of the sun and stars, light polarization of the sun on the atmosphere and on Earth's magnetic field. Beyond that, Sensorial Cues such as odours, visible features (e.g.: landmarks) and the tracking of self-generated movements provide vector information between places that may be used for homing, visiting known places or for path integration (dead reckoning).

It has been reported in the literature that several animals possess more than one mechanism of orientation that may be either used alone or in combination.

Table 3.4 Classification of animal orientation mechanism. Organised by author based on Hansson and Åkesson[107].

Biological Compasses
A. Time-compensated Sun compass
B. Skylight polarization compass
C. Star compass
D. Magnetic compass
Sensorial Cues / Sensors
A. Odours
B. Visual Cues
C. [Tracking of] self-generated movements

3.1.3.2.1. Time-compensated Sun Compass

This type of biological compass provides a directional information from the combination of the position of the sun and the time of the day given by the animal's circadian clock. Time-compensated Sun Compasses are widespread across the animal kingdom, being

reported in several species of birds and in diurnal insect species such as honeybees, desert ants and migratory butterflies[107].

3.1.3.2.2. Magnetic Compass

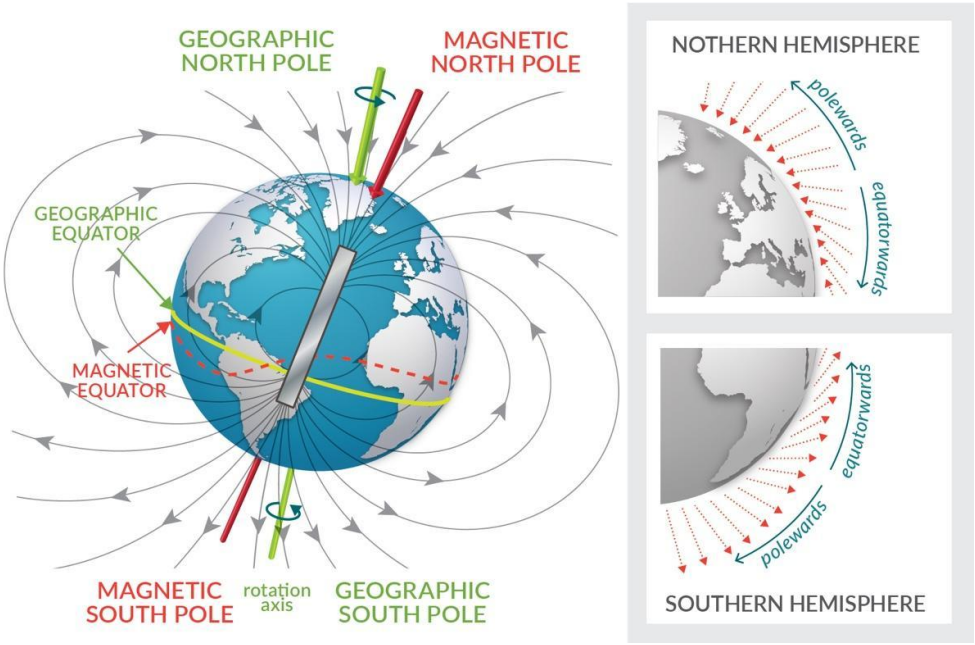


Figure 3.4 Diagram of natural magnetic field lines on Earth, highlighting the singularities of geographic and magnetic hemispheres.

Left: a simplified illustration of the magnetic field around the Earth from magnetic south to magnetic north pole. Right: illustration of the inclination of magnetic field vectors on magnetic north (top right) and south (bottom right) poles.

Some of the main mechanisms by which the Earth's magnetic field can be measured in terms of strength are magnetic induction, magnetic particles and magnetically sensitive biochemical reactions. Animals have been found to make use of the latter two processes by means of both ferromineral-based magnetoreception and radical-pair-based magnetoreception[107]. Magnetic-based biological compasses provides not only the clue for the North but also (and possibly more importantly) the sense of the latitudinal position of the animal, as the magnetic inclination decreases nearing Equator (Figure 3.4).

3.1.3.2.3. Skylight Polarization Compass

When light enters the atmosphere some rays may be scattered by molecules of air, water, dust or aerosol that are smaller than its wavelength. The degree of polarization will vary according to the angle that incident light rays reach the atmosphere, resulting in a *polarized light pattern* of the sky. This pattern is visible through specialized optical equipment and is also perceived by several animals that take advantage of this ability to obtain directional geographical information[107,120,121].

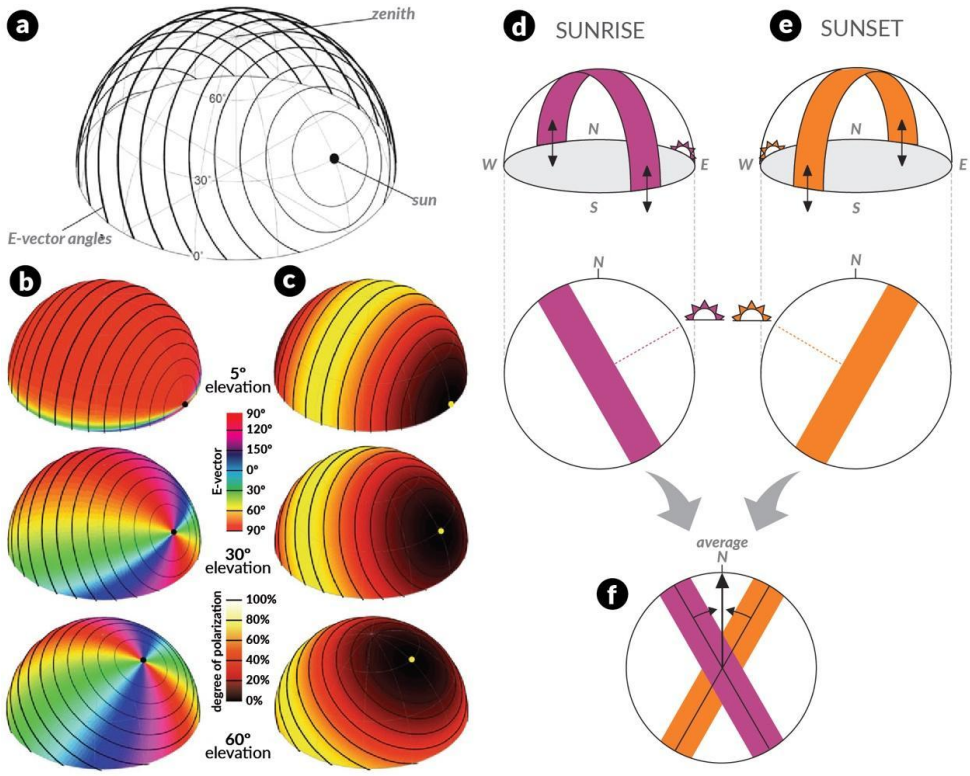


Figure 3.5 Key components and mechanism of Skylight Polarized Compasses.

(a) Single-scattering Rayleigh model of skylight polarization. The orientation of black lines indicate E-vector angles relative to solar elevation and the thickness of the line indicates the degree of linear polarization (b) E-vector angles during different times of the day. (c) Degree of polarization at different solar elevations observable by animal eyes. Source of A, B and C: Horvath[120]. d) 3D and top view of Polarized Light cues at sunrise E) 3D and top view of Polarized Light cues at sunset e) The variation of Polarized Light cues across the day is used to calibrate the skylight polarized compass day after day. Source of D, E and F: Muheim[121].

Similar to the previously described compass this type of biological compass must also compensate the polarized light pattern according to time change along the day and also to variations in altitude (Figure 3.5). Compared to time-compensated sun compass, this mechanism has the benefit of being available even when the sun is obscured by clouds, as long as some blue sky is still visible[107,120].

3.1.3.2.4. Star Compass

Similar to human-made instruments of navigation, several species of nocturnal and migratory birds have been identified as using star reference for orientation. Experimental evidence supports that star patterns are learnt by young birds and may be further involved in three systems: to define the animal's geographical position; as a part of time-compensated star compass; or to define geographical north from the position of the stars according to the rotational point (Figure 3.6)[107]

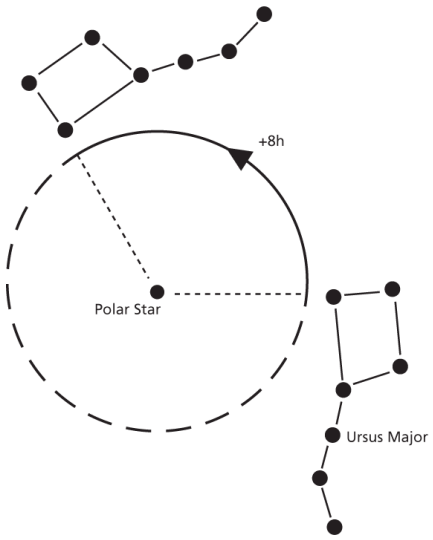


Figure 3.6 Star compass mechanism explained.

Accurate coordinates are obtained by compensating the rotation of the star map according to animal's internal time clock. Source: Hansson and Åkesson[107].

3.1.3.3. Spatial Representations and Maps

Both orientation methods and navigation may rely on internal spatial representations resembling more or less what we call *maps*. So far two basic types of maps and its subsets are described by Hansson and Åkesson[107], as in Table 3.5:

Table 3.5 Classification of animal spatial representation. Organised by author based on Hansson and Åkesson[107].

Mosaic Maps
A. Basic Mosaic Maps
B. Combined Mosaic and Cognitive Maps
Gradient Maps
A. Mono-coordinate Gradient Maps
B. Bi-coordinate Gradient Maps *(combination of two mono-coordinate maps)*

Mosaic Maps are based on the spatial and geometric relationships between home and landscape features surrounding it. Thus, each mosaic map will be as big as the territory explored by each individual and the catalogued landmarks on it. The basic concept of mosaic maps was first proposed by Hans G. Wallraff in 1974[122] and encompasses the paired linking between home and any other targets or landmarks (Figure 3.7). Going beyond it has been speculated that some animals could be able to expand the complexity of Basic Mosaic Maps in Cognitive Maps, by learning or deducing geo-spatial relationships among already known landmarks. Through this skill the animal would be capable of making novel shortcuts between locations, but finding evidence for this ability has proven difficult[107,123].

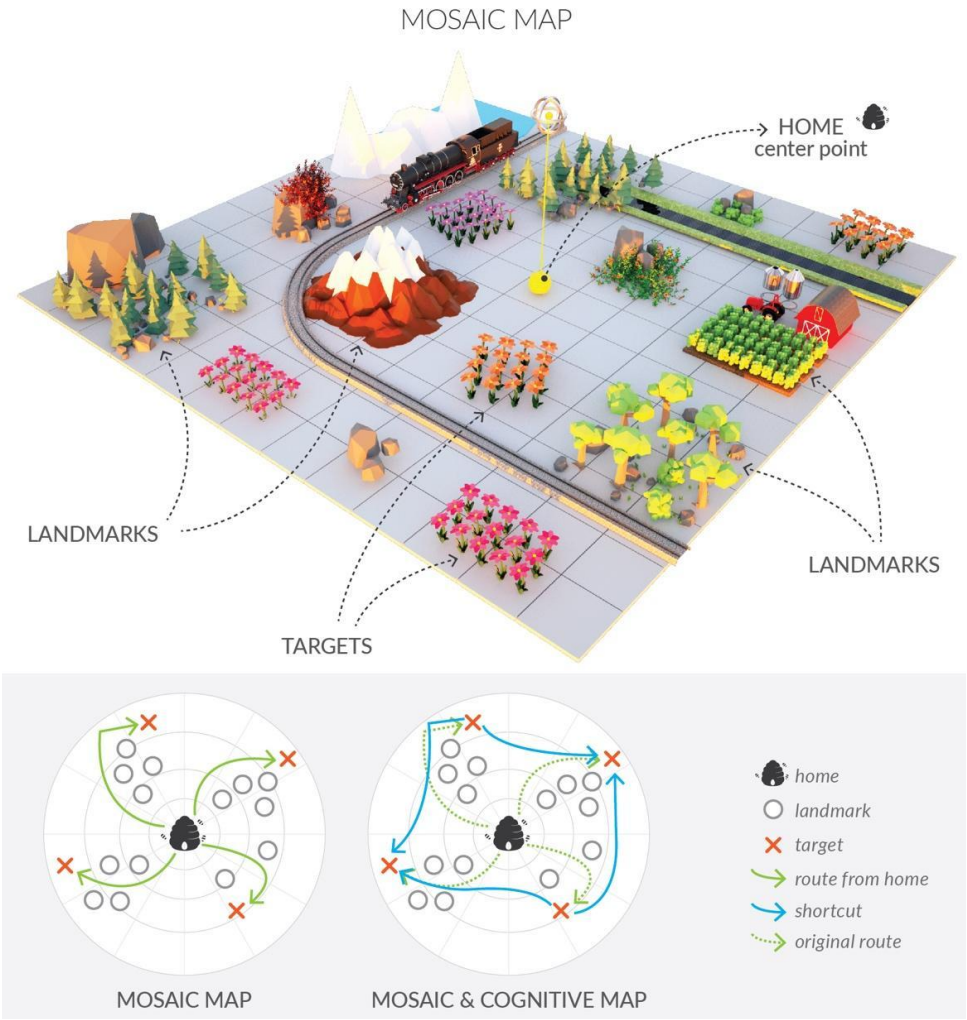


Figure 3.7 Diagram representing how home range, targets and landmarks are believed to be symbolically represented in an animal's mind through mosaic maps.

Gradient Maps, on the other hand, are based on the gradient variation (ideally perpendicularly oriented) of a particular feature along the space. Two feature gradients (disposed in different orientation) may be combined in a Bi-coordinate Gradient Map such as in global avian navigation relying on both geomagnetic inclination and field strength (Figure 3.8). Furthermore, as in Mosaic Maps, Gradient Maps may be built from local experience by learning suitable features to track for later recording[107].

The possibilities of animal navigation based on bi-coordinate maps relying exclusively in geomagnetic sense were analysed by Bostrom *et al*[124] for different regions of the Earth. The study investigated areas around the planet where two gradient maps (total geomagnetic field intensity - TGFI, and geomagnetic field inclination - GFI) could be combined into a bi-coordinate map (Figure 3.9). Each mono-coordinate (gradient) map was first represented as isolines on the map (Figure 3.9a) and the angle between the

isolines from different layers (TGFI and GFI) was utilised as a criteria to map high and low resolution areas (Figure 3.9b).

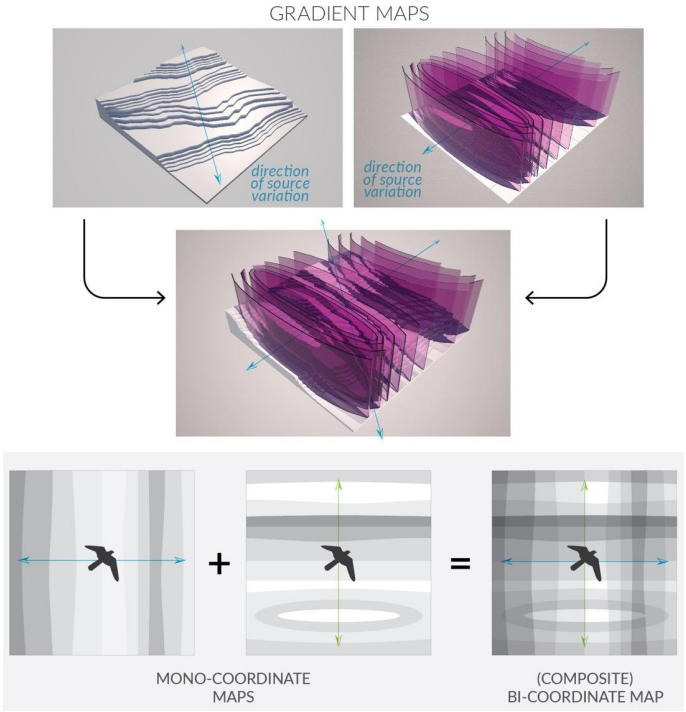


Figure 3.8 Diagram of two different mono-coordinate maps (1 axis) and their combination into a composite bi-coordinate map for navigation.

The areas in which the angle between the isolines were equal or greater than 60º were considered of high resolution and those in which the angle between the isolines ranged from 15º to 60º were labelled as low resolution. Regions with angles below 15º were considered insufficient for navigation utilising only the combination of these two geomagnetic gradient maps analysed.

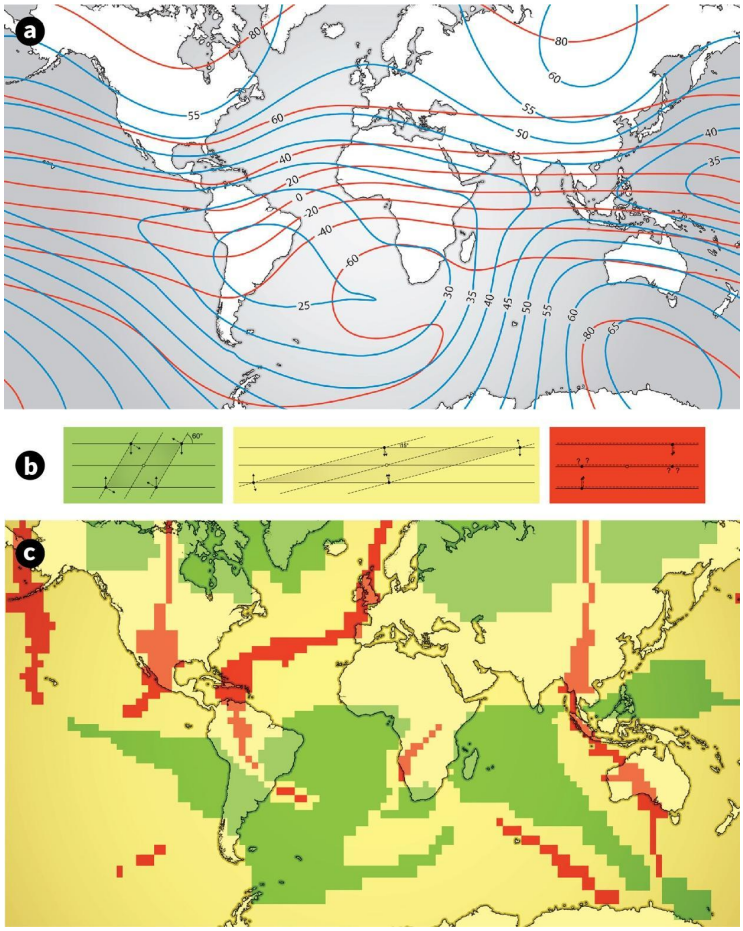


Figure 3.9 Two gradient maps combined into a bi-coordinate map.

A) The map shows blue isolines for total geomagnetic field intensity (μT) and red isolines for geomagnetic field inclination (º) in latitudes between 70ºN and 70ºS, in Mercator projection. B) The angle between the isolines on the map in (A) (geomagnetic field intensity and geomagnetic field inclination) was utilised as a criterion on evaluating the possibility of composing bi-coordinate gradient maps. Green indicates high resolution mapping in which the angle between the isolines is equal or greater than 60º. Yellow indicates low resolution mapping in which the angle between the isolines ranges from 15º to 60º. Red indicates areas in which mapping is not possible. C) Mercator projection map showing navigable and non-navigable regions of the planet utilising bi-coordinate gradient maps composed of geomagnetic field intensity and geomagnetic field inclination. Source: adapted from Bostrom *et al*[124].

3.1.3.4. Path Integration (Dead Reckoning)

Given the imprecision of sensory methods combined to changes in the environment and possible movements beyond the animal's control, such as accidental displacements, cognitive abilities such as path integration may provide an excellent tool for successful navigation. On performing path integration (dead reckoning), by combining sensorial cues to previously known information, animals can keep a continuously updated record of their current position, adjusting their route from time to time and getting rid of incidental disturbances or imprecision[125].

Collett and Graham[125] describe two different kinds of path integration system. The first method regards the animals with fixed dwelling which perform a path integration based on the reference of their nest, hive or other fixed origin point. In this method the animal stores radial vectors with the origin in its nest. The second method regards the animals without a fixed dwelling that rely on either visual or non-visual landmarks. In this latter method the animal stores a network in which vectors with size (distance) and direction connect different places (Figure 3.10).

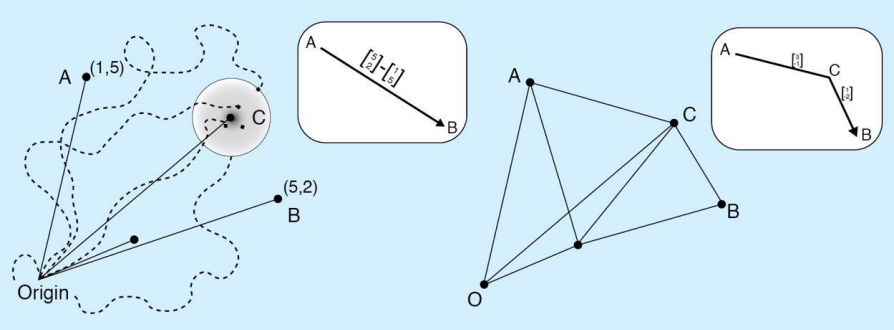


Figure 3.10 Two different kinds of path integration systems described by Collett and Graham[125].

First type (left) describe animals with fixed dwelling that store a beam of radial vectors originated in their dwelling. Second type (right) describe animals without a fixed dwelling that are capable of performing path integration with visible or non-visible landmarks, creating a network of vectors connecting places. Source: Collett and Graham[125].

3.2. Mathematical Models of Behaviour

Mathematical Models are abstractions of real life problems, systems or events which are useful for applications such as:

- explaining complex processes;
- identifying variables involved in complex processes
- modelling future scenarios;
- recreating current or past scenarios;
- simulating consequences of changes in variables.

Since they are abstractions of real-world situations, building models requires the ability to see the big picture of a complex system but also to determine which variables are most relevant to it. The required input data for a model must match, to some extent, the available, collectable or measurable data. If a model requires too much information chances are that no one would actually be able to provide it. Houston and McNamara[29] recommend beginning from simple models capable of providing an understanding of the underlying logic. These models may be developed further and incrementally until they reach a complexity compatible with the modelled situation.

A higher complexity in mathematical models does not necessarily result, however, in a higher quality or accuracy, as Viswanathan *et al*[30] emphasize: "Despite this 'coarse graining,' these [simplified] models lead to statistically robust results, precisely because they do not depend on the particular biological implementation of the search mechanisms. There is a long tradition in statistical physics in which apparently simple models lead to remarkably good agreement with experiment (e.g., the *Ising model* of ferromagnetic phase transitions)"[30].

3.2.1. Foraging Behaviour and the Optimal Foraging Theory

An animal's movement can be observed on the most diverse, sometimes surprising, scales: from bees performing a waggle dance within a beehive, through mammals foraging locally and then homing to feed their offspring, to cross-continental migrations completed yearly by certain species of birds. All organisms move at some stage of their life cycle; from protozoans to huge whales. Even the organisms and organic structures which are unable to actively move (e.g. bacteria, viruses and plant seeds) depend on the movement of the fluid to succeed[29]. As it may seem like a superior ability at first glance, active movement is not necessarily better. Plants, for instance, are some of the most ancient and successful forms of life. Locomotor systems involved in active movement demand energy to run and, in this context, reduced movement may also be an advantageous strategy to avoid energy expenditure. Among the animals with such ability, obtaining food and water, mating and avoiding predators are some of the main motivators to move. Feeding and mating can be thus considered the two major and common life-lasting challenges for genomes and so as for all species[30].

The model assumes that food is distributed into several homogeneous patches of different quality levels. Consider for instance a forager finding a food patch after some time travelling: the quality of the resource and handling time would enable drawing a curve of cumulative energy gain (Figure 3.11) and its rate of return (Figure 3.11b). In order to find the optimal time when energy return would be maximised the model of Marginal Value Theorem (MVT) can be applied with the OFT. The optimal time feeding on this resource can be projected by tracing a tangent line beginning on the time when the travel started[31] (Figure 3.11c).

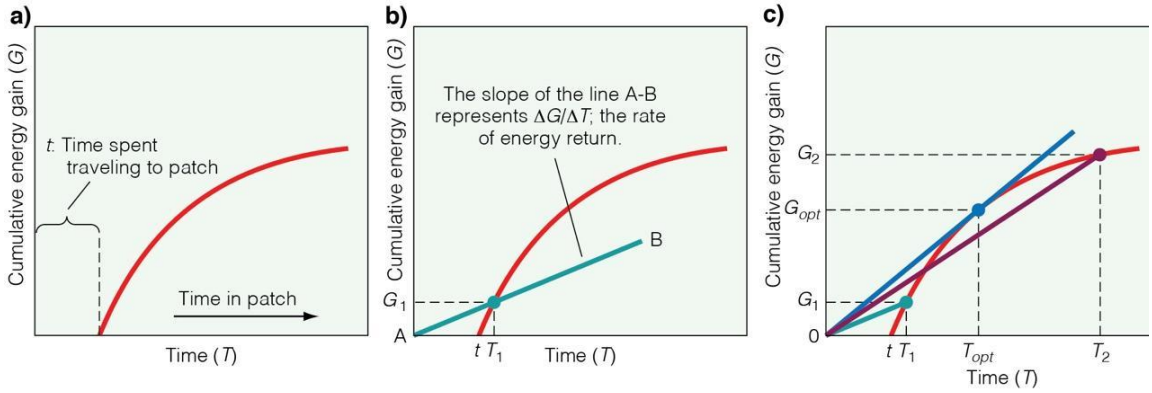


Figure 3.11 Optimal Foraging Theory and Marginal Value Theorem exemplified in charts.

A) Shows the cumulative energy gain curve (red line) and the time spent traveling to the patch (t) in a hypothetical foraging travel. B) The rate of energy return at a particular time (e.g. B) can be calculated by drawing a line from origin (time 0) to the corresponding point of the curve at the desired time. The slope of the line (A-B) gives the rate. C) brings energy return rate calculated from three different moments. The optimal time to leave the patch (T_{opt}) is when the energy return rate line (blue line) has the highest slope, tangent to the cumulative energy gain curve (red line). Turquoise and purple lines represent two other energy return rates calculated at times T_1 and T_2, respectively. Source: Smith and Smith[31].

3.3. The Roundworm *Caenorhabditis elegans*

Caenorhabditis elegans is a free-living roundworm of remarkable simplicity and present in most of the continents (Figure 3.12). In 1963, Sydney Brenner proposed its adoption as a *model organism* for the investigation of neural development in animals, as this is one of the simplest living organisms with a nervous system. Since then, *C. elegans* has been extensively studied, being the first multicellular organism to have its whole genome sequenced and its neuronal wiring diagram completed. The study of the species provides clues on the rudimentary functioning of body, brain and behaviour[41–45].

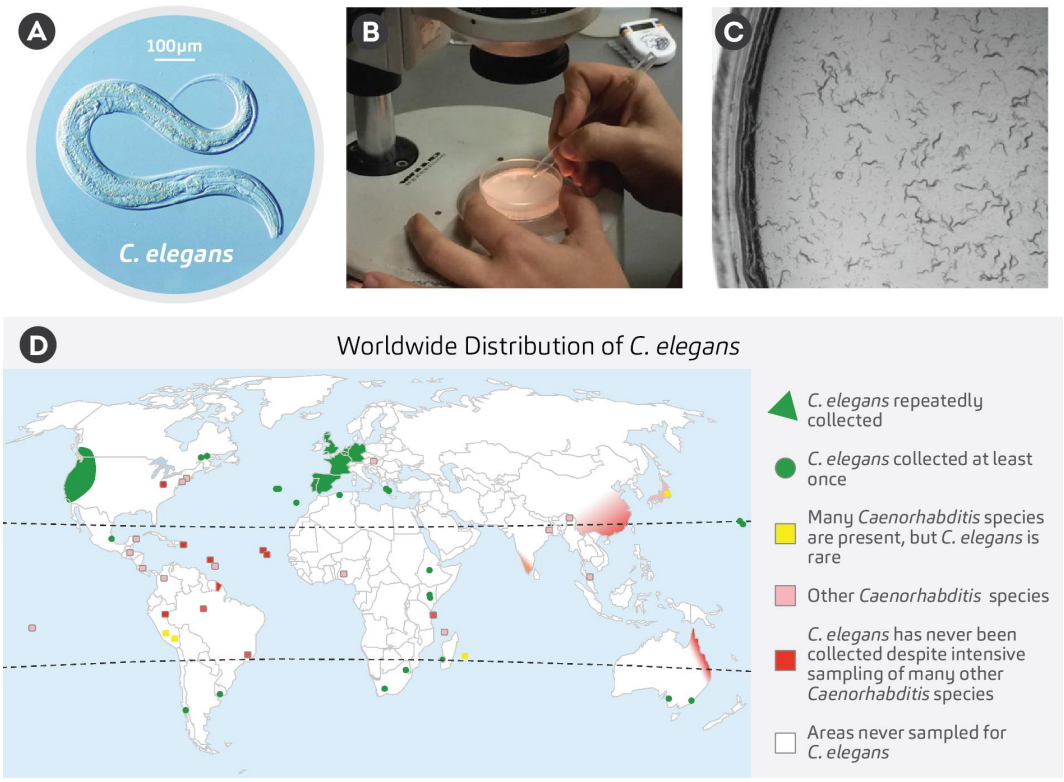


Figure 3.12 C. elegans anatomy and distribution.

(A) adult C. elegans individual. Source: <https://sites.psu.edu/>. (B) Petri dish with C. elegans individuals, that can be observed in more detail in (C). Source: Emmons[45] . D) Worldwide distribution of *C. elegans'* species. Source: Frezal and Felix[44]

There are two sexes of individuals of the species: males and hermaphrodites, the latter being more often studied as it is possible to obtain genetically identical populations from self-fertilization.

The nervous system of hermaphrodite individuals is composed by 302 neurons[e], 60 of each are sensory and 113 are motor neurons. The neural system is composed by nerve ring, head and tail ganglia, dorsal and ventral nerve cords, and a few lateral neurons. *C. elegans*' perception of its environment is mediated by mechanical stimuli, temperature, and chemical cues, accomplished through 24 sensillar organs, and several sensory neurons[41–43].

Four muscle bands along the length of the animal control dorsal and ventral bending, and with the exception of its head, the nematode is not able to bend left of right. The animal is capable of crawling in solid media and swimming in liquid media, either forward or

[e] Against 385 neurons in male individuals (of which 91 are sex-specific). Hermaphrodite individuals have only 8 sex-specific neurons.

backward. As undulatory movements occur from dorsoventral waving, moving individuals lie on their left or right[39,126–128].

Despite the simplicity of its anatomy, *C. elegans* displays a considerable repertoire of behaviour, including foraging, feeding, escaping and avoiding threats, sensory responses to stimuli, as well as learning and memory, mating, and social behaviour[39,40].

3.3.1. Movement Tracking and Models of Exploratory Behaviour

The exploratory behaviour of *C. elegans* emerges from a close interaction of sensory and motor neurons, with a few interneuronal relays in between. The organism moves towards more favourable surroundings by chemotaxis, thermotaxis, and/or aerotaxis, and evade noxious or harmful conditions through avoid or escape behaviours. A total of 60 ciliated sensory neurons compose different sensory systems of the animal and are distributed in tasks like chemosensation (gustation or olfaction), thermosensation, mechanosensation and proprioception. Most of the sensory neurons are concentrated in the head and tail ganglia, being another few distributed alongside the animal.

The exploratory behaviour of *C. elegans* can be decomposed in simple steps as forward and backward movements, turns, and reversals, that can be coupled into more complex movements[39–41,129,130]. In the present work, we adopt the classification in Gray *et al*[39]:

- **Head Swings**: basic component of *C. elegans* movement (Figure 3.21B);
- **Forward movement**: sinusoidal movement with shallow or deep bends, created by dorsoventral flexing;
- **Turn**: general term for change in direction, usually in forward motion; occurs when head swings are stronger in either the ventral or dorsal directions; large head swings result in rapid turns, whereas successive gently head swings result in a gradual curving;
- **Omega Turns**: sharp turns in which the head nearly touches the tail, or a reorientation of $> 135^{o}$ in single head swing (Figure 3.13A, B and C);
- **Reversal**: deliberate reorientation movement, composed by head swings;
- **Short Reversals**: combination of 1 or 2 head swings (Figure 3.13B);
- **Long Reversals**: combination of 3 or more head swings (Figure 3.21B);
- **Pirouettes**: a *Reversal*, most commonly long, coupled with an Omega Turn (Figure 3.13B);

Several works investigating *C. elegans'* exploratory behaviour identified pattern variations according to: the presence or absence of food; its (short-term) memory about the environment; and its internal state of hunger[39–41,130]. Gray *et al*[39] identify three distinct states: Feeding (Dwelling), Local Search, and Dispersal:

- **Feeding** (on food): animals move forward slowly and reverse frequently (mostly in short reversals) - in a pattern called *dwelling*. Reversal movements are usually followed by short turn angles.
- **Local Search** (well fed worms, 1-12' off food): the frequency of short reversals decreases 10- to 20-fold immediately after the animals being removed from food (Figure 3.13E). Similar in magnitude, is the increasing in frequency of long reversals and omega turns (Figure 3.13E). Speed is also increased by an approximated 10-fold, compared to the average speed on food.
- **Dispersal** (starved worms, 35-40' off food): animals keep a high speed but reduce the frequency of all reversals (long and short) and of omega turns. This combination results in long and relatively straight patterns, allowing the individuals to disperse and forage distant paths of food.

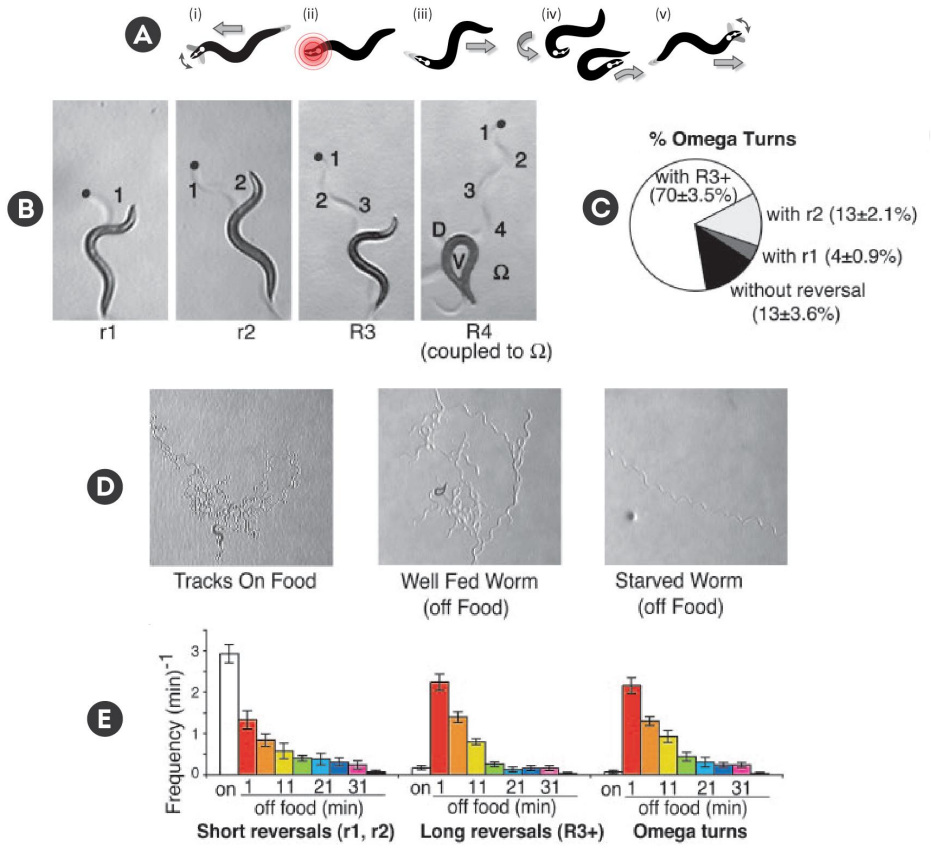


Figure 3.13 *C. elegans* movement tracking and statistics.

(A) Illustration of a *C. elegans* individual performing an Omega Turn (iv) after finding an obstacle in (ii). Source: Broekmans *et al*[131]. (B) Reversal movements captured in microscope: (r1) short reversal with one head swing, (r2) short reversal with two head swings, (R3) long reversal with three head swings, (R4) long reversal with four head swings followed by an Omega Turn. Source: Gray *et al*[39]; 'D' and 'V' indicates, respectively, the dorsal and the ventral sides of the animal. (C) Percentage of Omega Turns isolated or coupled with reversals. (D) Experimental conditions in which Gray *et al*[39] investigated C. elegans behaviour. (E) Frequency of Short or Long Reversals, and Omega Turns during the first 32 minutes of the experiments[39].

Part III
Materials and Methods

Part III of this work is focused on the Methods and Methodological Developments.

- Chapter 4 presents the early series of experiments, and also the three early foraging algorithms. This is included with the aim of clarifying the chain of decisions which led to the final methods, and also to offer guidance on the issues which were overcome during this period for further research.
- Chapter 5 presents the consolidated methods for simulating, optimising, and processing the results of the three refined algorithms, along with three later experiment series. The three refined algorithms will be presented in Chapters 6, 7, 8, and 9.

Chapter 4
Early Experiments and Methodological Developments

In this work, a total of 6 foraging algorithms were developed and tested. The first 3 (SAT-4.3, SAT-4.1, and AT-5) were tested along 7 series of experiments (A-G) and laid the foundation for the three refined foraging algorithms (AT-6, ATRP-8, and ATRP-7) (Figure 4.1). The results of the experiments with these early models are essential to explain the pathway to consolidated Experiment Design, Materials and Methods (Chapter 5), and to the 3 refined algorithms (Chapters 6, 7, and 8). For that reason, these early models and the experiments series with them will be presented along this chapter, aiming to fill the gaps and provide support to the comprehension of the experimental methods and algorithm design choices that will be presented over the chapters 5, 6, 7, and 8.

This chapter describes the methodological developments in the design, experiment, optimisation, and analysis of the *C. elegans*-based foraging algorithms. Here I will cover the main aspects of the 10 series of experiments of multiparameter optimisation with Evolutionary Algorithms (EAs) aimed at calibrating the Foraging Algorithms created. These 10 series of experiments were labelled from A to J, and are summarized in Table 4.1.

Series A-F consisted of experiments and optimisation cycles that contributed to the improvement of:

- The Simulation Platform, entirety coded for the purpose of this research;
- The Programs for running Optimisation with Evolutionary Algorithms: The first version I coded in its entirety (Series A and B), and the second version was adapted from an open-source program for Matlab, that offered a more robust yet flexible solution;
- The EAs and the EA parameters used for the Optimisation (population/pool size, mutation rate, elitism, etc);
- The objectives and costs to be optimised, as well as the techniques of estimating and measuring them;
- The range of each optimizable parameter, that should be wide enough not to restrict the algorithm, but short fitted enough to allow a better resolution and so the optimisation does not 'waste' time with values too far from optimal.

These series of experiments (A-F) also provided a better understanding of the limits of the Foraging Algorithm and of the environments to be optimised (i.e.: minimum and maximum sizes of the field and minimum and maximum qualities of the light sources).

Series G-J implemented the same optimisation methods and model of costs and consisted of experiments with 4 different Foraging Algorithms.

The objectives, methods, and results of series of experiments A-G will be presented and discussed in sections 4.1, 4.2, and 4.3:

- Section 4.1 covers the algorithm design, and describes some of the early and rudimentary models that led to the 3 refined algorithms.
- Section 4.2 details the mechanics of the Simulation Platform, as well as the tested environments and the robotic hardware simulated.
- Section 4.3 describes the methods and improvements in optimisation, covering the evolution of the optimisation problem, calibration and validation experiments.

The refined experimental design, materials and methods will be presented and the optimisation problem will be defined formally in chapter 5. Experiment series H-J will be presented and discussed in depth in chapters 6, 7, and 8.

Table 4.1 Summary of all experiment series indicating the version of the algorithms used, the number of input parameters controlling the algorithm, the number of objectives, the total number of function evaluations (number of runs), and the respective sections describing it in the present document.

EXPERIMENT		FORAGING ALGORITHM	OPTIMISATION		NUMBER OF FUNCTION EVALUATIONS	SECTIONS CONTAINING DESCRIPTION & RESULTS
SERIES	SUBSET		INPUT PARAMETERS	OBJECTIVES		
A		SAT-4.1	5	1 (Fitness)	33,240	4
B		SAT-4.1	5	1 (Fitness)	108,440	4
C		SAT-4.1	5	1 (Fitness)	66,600	4
D		AT-5	5	3	175,744	4
E	E.1	AT-5	5	3	910,322	4
	E.2					
	E.3					
F		AT-5	5	3	58,740	4
G	G.1	AT-5	5	3	767,241	4
	G.2					
	G.3					
	G.4					
H	H.1	AT-6	6	3	250,703	5 & 6
	H.2					
	H.3					
I	I.1	ATRP-8	8	3	197,850	5 & 7
	I.2					
	I.3					
J	J.1	ATRP-7	7	3	54,614	5 & 8
	J.2					
	J.3					
					2,624,694	

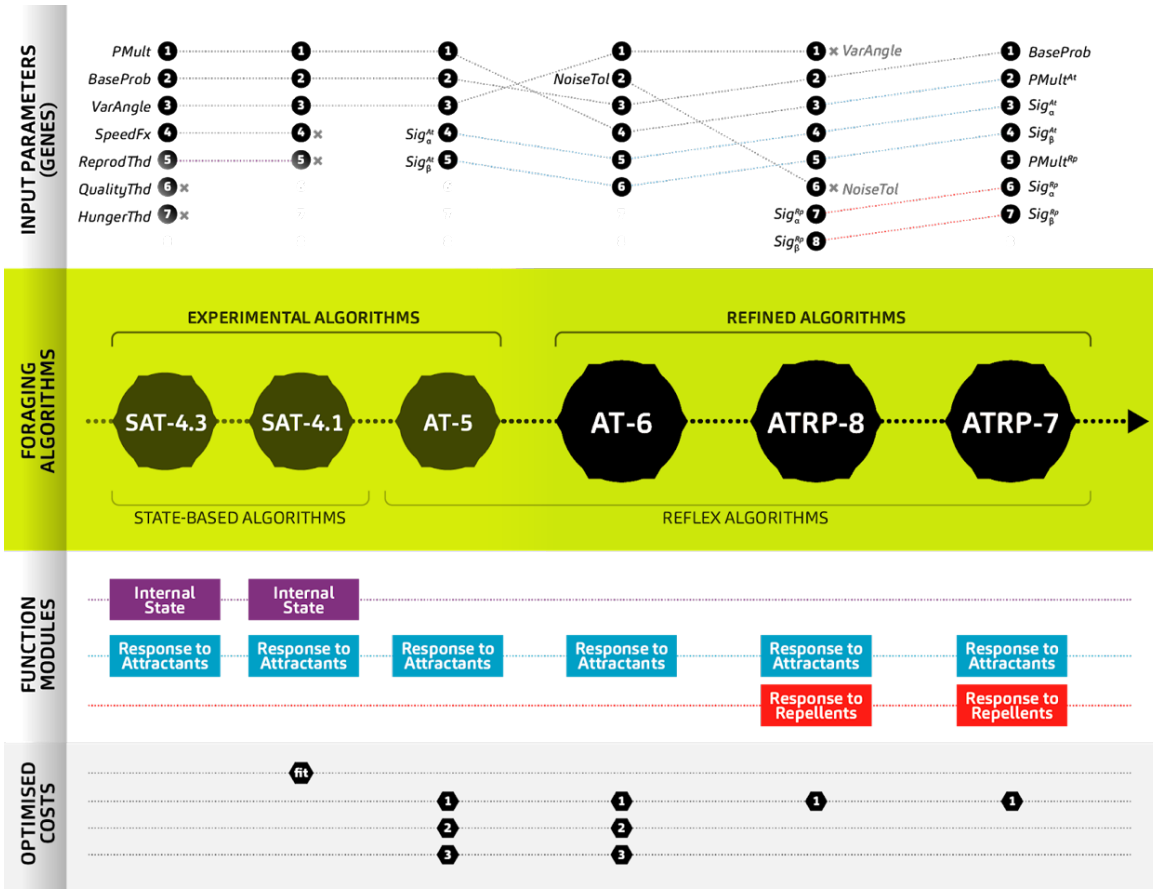


Figure 4.1 Summary of all the experimental and refined algorithms presented in this work, as well as the input parameters (genes), function modules and the optimisation outputs (fitness or costs).

4.1. Methodological Developments in Foraging Algorithm Design

This section will describe rudimentary models of Foraging Algorithms implemented at early stages of this research (SAT-4.3, SAT-4.1, and AT-5) and that laid the ground for the three refined foraging algorithms (AT-6, ATRP-8, and ATRP-7) (Figure 4.1).

Early Models SAT-4.3 and SAT-4.1 (adapted from it) are State-based Foraging Algorithms, and AT-5 is a Reflex-agent Foraging Algorithm. These three models will be covered in the present section (4.1), and the three refined algorithms will be presented individually, in chapters 6, 7, and 8.

4.1.1. Early Model: State-based Foraging Algorithm (SAT-4.3)

This model represents the foraging behaviour of *C. elegans* based on environmental conditions and a rudimentary state *hungriness*, in this case derived from the battery level. This model also implemented a rudimentary reproductive behaviour, in which any agent with a certain level of energy stored would lay an egg. The eggs would not hatch nor generate new robots, but were instead a tool to measure the performance of each robot, as they represented the accomplishment of a task (Figure 4.2). When an agent collects a favourable amount of energy (higher than the *Reproductive Threshold*), one egg is laid, and a certain amount of energy is subtracted from their battery. Laid eggs do not generate new individuals (for simplification purposes), but the number is kept in each agent's inventory.

The sensory capabilities of the agents (robots) are limited to a single sensor, capable of sensing the intensity of light at the spot and only recording the current and the last measurement. The presence of food is visually represented as light spots of varying intensities, according to their quality. In order to save time and to obtain a greater number of samples, each simulation runs with a group of clones that do not interact with each other. At the end, the fitness was calculated based on the average of the population.

The simulation occurs during a time T, scaled by time steps. The graphic console permits the observation of the interaction of the agents in the field, the number of eggs laid by each clone and the dead agents that ran out of battery.

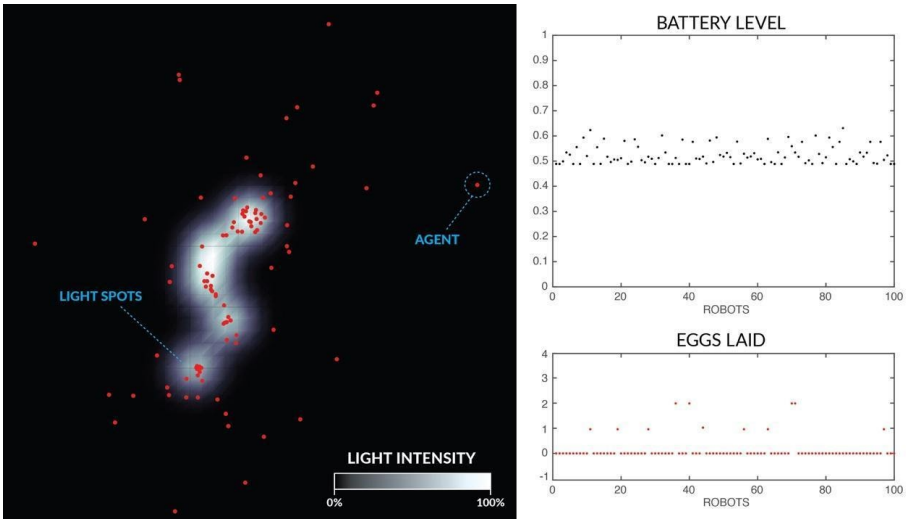


Figure 4.2 Screenshots of the first simulation model.

Left: two-dimensional field with light spots (white gradients) and Reflex agents (red dots). Right: status console with the battery levels of all the 100 robots (top) and the number of eggs laid (bottom).

Speed is updated by a speed modulator, increasing or decreasing according to the raw value of light conditions detected in the environment. The modulation of speed is inversely proportional to the raw value of light detected: high light intensities result in decreasing the speed (dwelling and local search), whereas low light intensities increase it (dispersal) (Fig. 4.3 and 4.4). When judging the relevance of the variation in the intensity of the light (compared to the previous time step), agents use the state variable *Quality Threshold* (Figure 4.3). A rudimentary memory variable was implemented as the *Quality Standards* variable, that constantly records and updates the highest and the average levels of light ever found by the agent. This register helps the setting of quality standards for future exploration in the environment, especially when maximum and average conditions of a source are not fully known. The variation in *Speed* is intensified according to the internal state variable *Hungriness*, dynamically adjusted according to the input parameter *Hunger Threshold* and proportionally to the battery level (Figure 4.3).

At each time step, the agent can either keep or change its direction. When the agent detects an increase in the light intensity (i.e. it is approaching a light source), the *probability of turning* decreases. Similarly, when a decrease in light, or no light, is detected, the *probability of turning* increases (Figure 4.4).

A turn happens when a random number (from a uniform distribution between 0 and 1) is lower than the variable *Probability of Turn*. The *Probability of Turn* of an agent in each time step is adjusted based on the input parameters *Base Probability of Turn* and *Probability Multiplier*, and according to the improvement or decline in light conditions detected in the environment (light gradient). The direction is updated by $180^{o} \pm$ a random (uniform) value between 0 and the *Variability of Angle* parameter value (Figure 4.4). *Variability of Angle* defines the wideness of the angle an agent can turn at once. The agents are allowed to turn both ways and the exact angle is randomly picked.

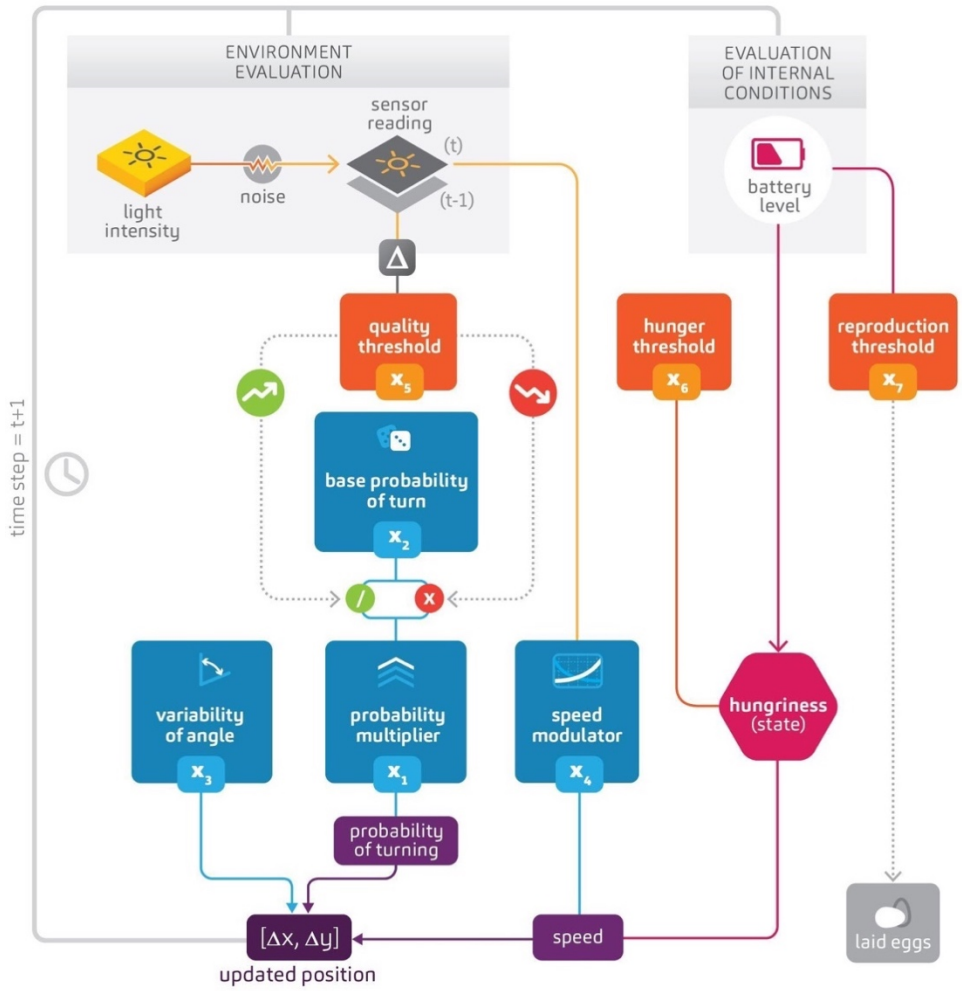


Figure 4.3 Diagram of behavioural parameters ruling the rudimentary state-dependant foraging algorithm SAT-4.3.

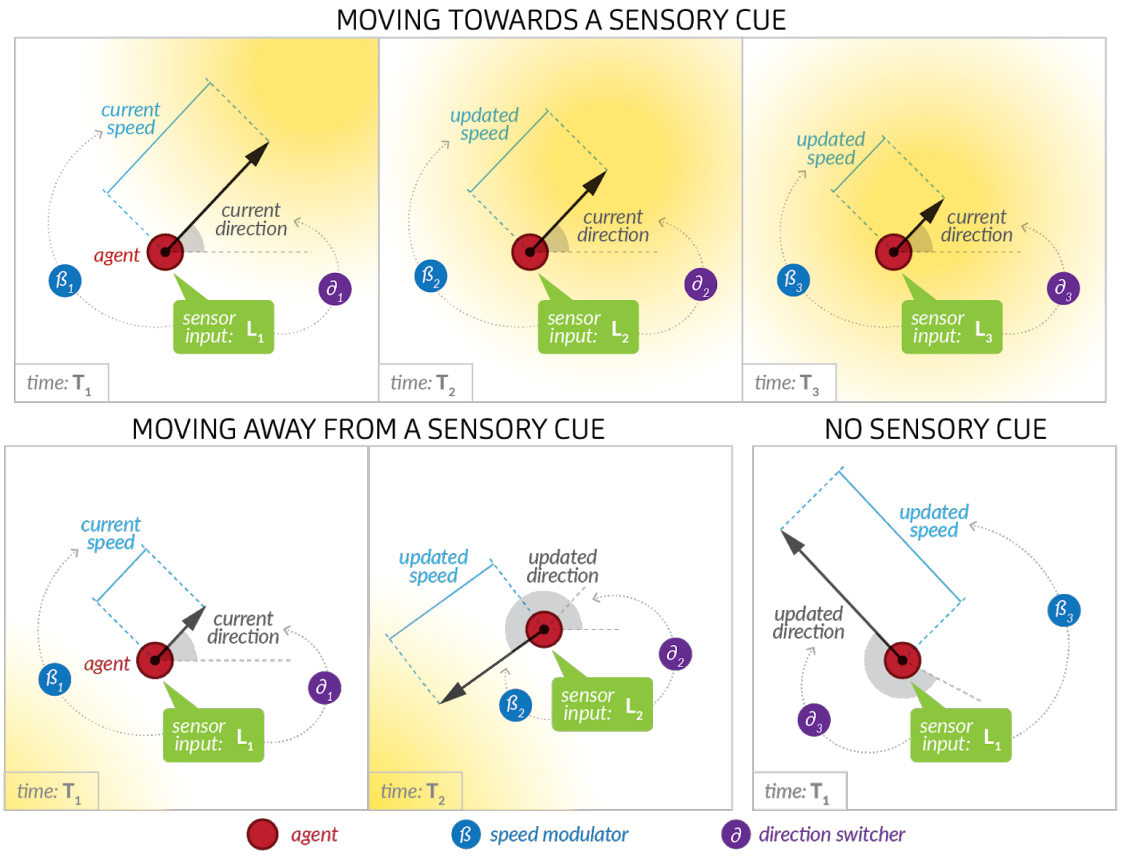


Figure 4.4 Diagram of Agents' behaviour moving towards and away from a sensory cue, and in the absence of a sensory cue.

In the event of an increasing level of light (top row), agents tend to keep the same direction and reduce their speed. In the event of a decreasing sensory cue (bottom row, left), the speed is increased and the probability of turn increases. In the absence of the sensory cue (bottom row, right) (i.e. extinguished light cue) both speed and probability of direction change increase.

4.1.1.1. Module Encapsulation and Multi-parameter optimisation

As part of the early experiments, the program has been re-written with the aim of saving computational resources and achieving modularity. The routines associated with Reasoning and Behaviour have been organised and encapsulated as functions or subroutines. Another big step was the normalization of the input parameters (and later of the cost parameters) to better fit the tools for multi-parameter optimisation. This step was taken carefully in order to prevent or minimise any loss in “resolution” in the numbers: during the conversion of the real parameter value to its normalized version, efforts were made so that the normalized version allows a wide variation within a realistic range of values.

All these developments made it possible for the present algorithm to be optimised. This version of the Foraging Algorithm was used in 3 series of experiments with evolutionary algorithms for optimisation (Series A, B, and C, that will be described in Section 4.3).

Aiming at saving computational resources at that point, parameters *Hunger Threshold* and *Reproduction Threshold* were set to fixed values. To differentiate it from the full implementation (SAT-4.3, encompassing all the 7 parameters), we refer to it as SAT-4.1

4.1.2. Early Model: Reflex-agent Foraging Algorithm (AT-5)

This foraging algorithm (AT-5) kept the main aspects of the previous algorithms (SAT-4.3 and SAT-4.1, presented in the previous section), however without the internal state variables and the input parameters related to them (Figure 4.5). Another key difference from the previous models is that the speed modulator is no longer linear, but controlled by a sigmoid function. A few versions of sigmoid controllers were tested - taking from 1 to 4 controller parameters (plus the variable with light/attractant intensity), resulting in more or less flexible sigmoid curves. The function with 2 optimizable input parameters was chosen for this and the subsequent experiments.

Foraging Algorithm AT-5 was used in four series of experiments (Series D, E, F, and G). The main aspects of it will be covered along the next lines, and the results of the above-mentioned series of experiments will be covered in Section 4.3.

The foraging algorithm inspired by *C. elegans* chemotaxis is composed of two key behaviours: runs and turns, controlled by a set of five parameters ($g_{1:5}$), also referred to in this work as the 'DNA', being: Base Probability of Turning (*BaseProb*: g_1), Probability Multiplier (*PMult:* g_2), Variability of Angle (*VarAngle*: g_3), Speed Sigmoid Controller α (Sig_α: g_4) and Speed Sigmoid Controller β (Sig_β: g_5) (Table 4.2).

Table 4.2 Set of input parameters (genome) for the *C. elegans*' bio-inspired minimalist algorithm AT-5

GENE	PARAMETER	APPLICATION
g_1	*BaseProb*	Sets the base probability of turning, in the absence of any change in sensed light level
g_2	*PMult*	Sets the multiplier (divisor) of the base turning probability when light decreases (increases)
g_3	*VarAngle*	Controls the variability of the angle of a turn;
g_4	Sig_α	Sets the steepness of the sigmoid curve that controls speed according to the sensor reading
g_5	Sig_β	Defines the offset of the sigmoid curve that controls speed according to the sensor reading

At each time step, an agent adjusts its speed and turn probability according to its sensor reading and possibly makes a turn. The agent's battery level is updated according to the

light level it is currently exposed to (i_{in}), as well as how much was spent on moving (i_{out}) and running basic systems (i_{BMR}):

$$Bat_t^R = Bat_{t-1}^R + \left(Light_t^{[x,y]} \times i_{\max}^{in}\right) - Speed_t^R \times i_{\max}^{out} - i_{BMR} \quad [4.1]$$

Also at each time step, the simulation program updates robots' positions, checks for extinguished light spots (replacing them with new ones if necessary) and checks which robots are 'alive' – a robot permanently 'dies' if its battery is depleted.

The reasoning process on each time step (t) starts when the agent acquires the sensor reading ($SensVal_t^R$) for light intensity ($Light$) at its current position [x,y]. The sensor reading ($SensVal_t^R$) is obtained from the actual amount of Light available in the Robot's position, added by a uniform random number between 0 and 1 ($RandNum$) scaled to the sensor noise (Equation 4.2). $\Delta SensVal_t^R$ is then obtained from the current and previous sensor readings in order to calculate the probability of the robot to turn ($PTurn_t^R$) (Equation 4.3).

$$SensVal_t^R = Light_t^{[x,y]} + RandNum \times SensNoise \quad [4.2]$$

$$\Delta SensVal_t^R = SensVal_t^R - SensVal_{t-1}^R \quad [4.3]$$

If $\Delta SensVal_t^R$ is sufficiently positive, $PTurn_t^R$ decreases, whereas if it is sufficiently negative, $PTurn_t^R$ increases. If the current and previous values are approximately equal, $PTurn_t^R$ maintains the value of $BaseProb$ (Figure 4.5).

Once $PTurn_t^R$ is set, a random number (0 to 1, uniform) is generated and, if it is less than or equal to $PTurn_t^R$, the robot will perform a turn (Figure 4.5). When performing a turn, the yaw ($\Delta\theta$) will be calculated using another uniform random number between -1 and 1 ($RandNum$), according to Equation 4.4.

$$\Delta\theta_t^R = 180° + (VarAngle \times RandNum) \quad [4.4]$$

Also, speed is modulated by an inverse logistic function, controlled by the combination of the current sensor reading and the input parameters Sig_α and Sig_β, according to Equation 4.5.

$$Speed_t^R = 1 - \frac{1}{1+(e^{Sig_\alpha \times (SensVal_t^R - Sig_\beta)})} \times V_{\{max\}} \quad [4.5]$$

As the behaviour of the agent is modulated without reference to any internal state variables, this is classified as a reflex-agent model.

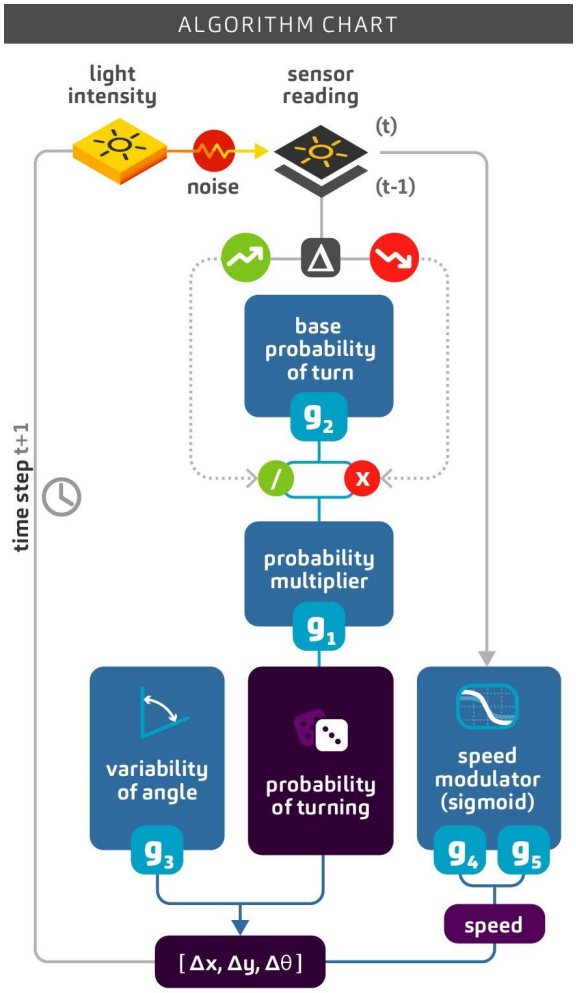


Figure 4.5 Diagram of the reasoning process on Foraging Algorithm AT-5.

4.2. Methodological Developments in Simulation Design

This section covers an overview of the methods and techniques designed for the realization of the experiments with the *C. elegans*-based foraging algorithms. The mechanics of the Simulation Platform, as well as the tested environments and the robotic hardware simulated, will be explained.

Simulated environments capable of recreating key aspects of the real world play an important role in reducing costs and time spent on the development of robotic applications. At the same time, these platforms also permit rigid control and isolation of variables, as well as robust performance measurements[132].

The initial plan proposed the simulation and evolution of bio-inspired robot lineages with the use of Evolutionary Algorithms for Optimisation (EAs). The settings for EAs, as well as the number of generations and of individuals per generation, may vary according to the complexity of the problem and to the number of costs and input parameters. As it turns out, for the complexity of the Foraging Algorithms developed during the course of this research, a single optimisation for each environment required from 8,000 to 30,000

function calls (section 4.3). Hence, the simulation of the foraging algorithms in more complex platforms (ie: *Gazebo*, *v-rep*, *ARGoS*) has proven impractical as some of these use excessive computational resources and time for the simulation of physical aspects (secondary to the present work), being unable to provide the necessary performance in the steps implementation, calibration and optimization of the models.

Thus, it was necessary to develop a specific platform for the implementation of algorithms and for simulating and evolving artificial intelligence agents in different scenarios. The simulation platform was developed allocating the maximum amount of computational resources in the simulation and evaluation, whereas implementing aspects related to physics, locomotion and control motor in a simplified way.

The computational processing time could not be estimated with precision at the beginning of the work, considering that it is proportional to several factors such as hardware and software performance and the complexity of simulation and optimization. The simulation increases in complexity according to the scenario, number of agents, virtual time and calculations performed by the [reasoning] algorithm. Similarly, the time taken by each optimization cycle varies mainly as a function of the optimization algorithm and the number of input variables and goals.

The program I developed using Matlab is capable of:

- Creating, storing and simulating simple environments (2D fields with attractant and/or repellent sources);
- Generating, distributing and storing states and variables of virtual robots (simulation agents);
- Allowing the implementation of other foraging algorithms;
  allowing multi-parameter optimisation by inputting parameters and outputting costs (as well as other results);
- Allowing a rigid control of the conditions of the environments, what is essential for the optimisation;
- Connecting to any Optimisation program compatible with Matlab.

4.2.1. Simulation Platform and Virtual Environments

The simulated environments consist of squared fields of variable size (user input), supplied with attractant or attractant combined with repellent light spots. The light spots are shaped as Gaussian Bells and spread from centre to radius (Figure 4.6). The position, intensity and duration of each light spot is randomly generated according to environmental settings: *x and y* ***limits***, *minimum and maximum* ***intensity***, *minimum and maximum* ***duration***. Their number, position, intensity and duration are unknown by the robots. Attractant light sources are the only resource available for recharging the

robot's batteries, and repellent sources (if existing) cause damage to the robot and deplete their batteries. Failing the purpose to find attractant sources, or contracting too much damage, results on the "death" of the agent. A conceptual representation of the simulations can be seen in Figure 4.6.

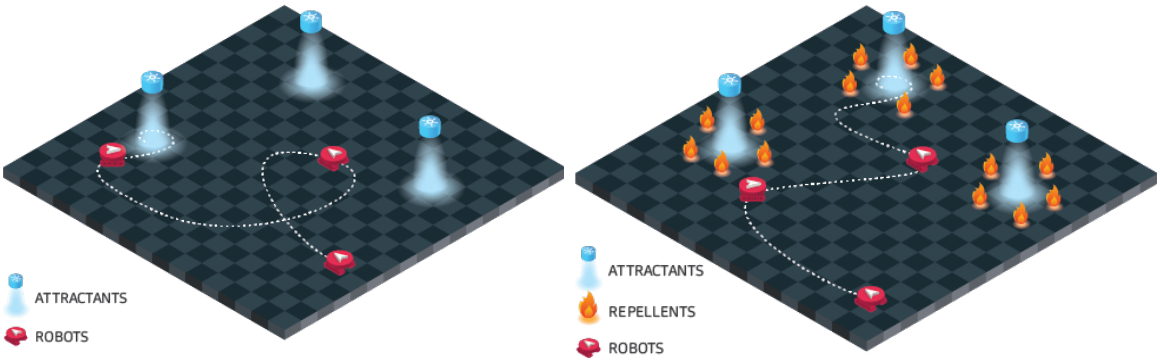


Figure 4.6 Conceptual sketch of the elements of the simulation.

Left: field with attractants only. Right: field with attractants and repellents. Illustration purposes only.

A constant number of light spots are placed in the field, in randomly assigned positions, and each one has a specific intensity and duration parameters. In environments with attractant and repellent sources, the repellent sources are clustered around the attractants, thus forcing a trade-off (Figure 4.6).

Each simulation runs during a virtual time T. In the beginning, a number of robots (Rn), with the same parameter setting (genome), are placed in random positions on the simulation field. There is no interaction between the robots and the purpose of running the simulation with a population of "clones" is to increase the sampling for each parameter setting (Figure 4.7).

Each robot of the simulation is equipped with a single light sensor that is only capable of measuring the intensity of light in the exact spot the robot is. In the implemented simulation, light is the only resource available to recharge the robot's batteries. In terms of movement, the robots are only limited by the maximum speed, moving in all directions and capable of performing turns from 0 to 360 degrees. The field boundaries are wrapped, therefore eliminating the need to manage obstacles.

The refined model of objectives and costs evaluate the robots for accomplishing three objectives:

- Remaining alive for the longest time possible;
- Store as much energy as possible in their batteries up to the end of the simulation;

- Explore the maximum area of the field. For optimisation purposes, the battery limit has been disabled (to increase the resolution of Objective 2).

The performance of each DNA is assessed according to 3 objectives, outputted in the form of Cost variables at the end.

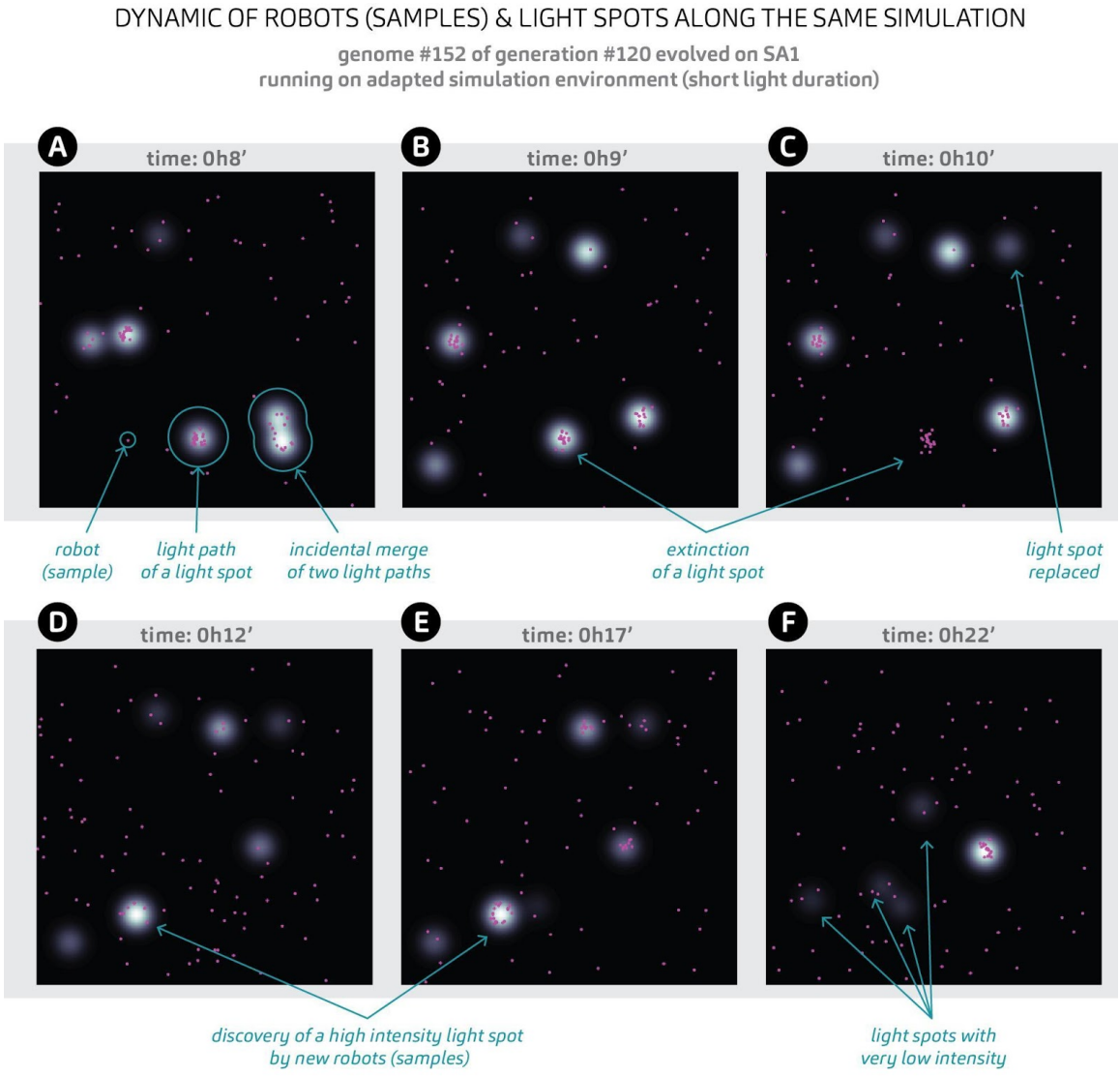


Figure 4.7 Dynamics of Robots (pink dots) and Lights (white glowing spots) in an example simulation.

The level of difficulty of each environment was assessed by the Light Index - a method created to calculate the overall light available in a simulation. The Light Index consists of the total light available in the field, averaged over time and space. The Light Index is a measure of the density of light in a certain environment over time. For environments with attractants and repellents, the light index is calculated separately for each stimuli.

4.2.1.1. Environments with Attractants Only (experiment Series G and H)

For the set of experiments with algorithms AT-5 (Sections 4.1.2) and AT-6 (Chapter 6), on environments with attractants only, all the environments were supplied with 6 light spots of radius 5m, lasting 1 to 3 hours each. When a light spot reaches its duration limit, it disappears from the field, being instantly replaced by a new one, with new random parameters of position, intensity and duration. Low intensities of light may not be enough to provide energy at the same rate as the battery is consumed, even when the robots are not moving (i.e. intensity below 20%). Some of the light spots will then become traps for the robots and it is expected that their genomes will evolve to avoid them. For this reason, environments with average or poor availability of resources may offer an extra challenge for the robots when compared to the ones rich in resources.

4.2.1.2. Environments with Clustered Attractants and Repellents (experiment Series I)

For the set of experiments with algorithm ATRP-8 (Chapter 7), in environments with attractants and repellents, all environments were supplied with 6 light clusters with one attractant source in the centre and ten repellent sources circling it. The radius of the attractant is 5m, and the radius of the repellents is 1.8m. This was found to be the optimum radius as due to the nature of the Gaussian curves used, for both attractants and repellents, any larger diameter of repellent could still affect even the most successful robots which found the centre of the attractant. The distance of the repellents from the centre of the cluster (and also the centre of the attractant) varies randomly between 7m and 10m. Each cluster lasts for a random period of time between 1 and 3 hours.

When a light spot reaches its duration limit, it disappears from the field, being instantly replaced by a new one, with new random parameters of position, intensity and duration (Figure 4.7). Low intensities of light may not be enough to provide energy at the same rate as the battery is consumed, even when the robots are stationary (i.e. intensity below 20%). Some of the light spots will then become traps for the robots and it is expected that their genomes will evolve to avoid them. For this reason, environments with average or poor availability of resources may offer an extra challenge for the robots when compared to the ones rich in resources.

Repellent sources also have a Gaussian or bell curve shape and are clustered around Attractants. The intensity of these sources is proportional to the attractant which they surround, therefore the stronger the attractant the stronger the repellent which surrounds it. Repellent sources are harmful and drain the robot's battery, each unit of repellent causes 40 times more damage to the battery than an attractant of the same intensity would be able to charge it.

4.2.1.3. Environments with Clustered Attractants and Repellents with varying intensity (experiment Series J)

These environments are a variation of the environments presented in the previous section (regular Environments with Clustered Attractants and Repellents) and were used in the experiments with algorithm ATRP-7 (Chapter 8). In these environments, the intensity of the attractants (AT) and repellents (RP) was individually modulated, resulting in 4 environment variations, presented in Table 4.3.

Table 4.3 Four environment variations of MA with Clustered Attractants and Repellents, with varying intensity.

ENVIRONMENT		Intensity of Attractants	Intensity of Repellents
Field Size & Quality	Intensity AT-RP		
MA (225-avg)	ar [1 1]	1x	1x
	Ar [5 1]	5x	1x
	AR [5 5]	5x	5x
	aR [1 5]	1x	5x

4.2.1.4. Virtual Robots' Hardware

The foraging algorithm is embedded as the behavioural module of a virtual agent (robot), capable of moving freely around a field in which light sources are placed.

As the simulation is grounded on real hardware specifications, systems running the robot depend on energy stored in the battery. During each time step, the battery is consumed by a fixed amount taken to keep the sensing and processing hardware functioning (*Basal Metabolic Rate*), and a typically larger amount proportional to the speed at which the robot is moving.

The robots' batteries would be recharged by attractant light sources, and depleted according to their exposure to repellents. The maximum level a repellent will drain is 40A.

Both attractant and repellent sources are detected by a single sensor each, and as a way to keep the model realistic, that sensor is not 100% precise, hence every reading is impacted by a random value representing the sensor noise. In the experiment series A-J, three values of sensor noise were tested: 1%, 0.5% and 0% (no noise).

The initial charge of the robots' batteries varied in different experiment series (100%, 40%, or 20%), as well as the cost for running all basic systems, that was initially set to 0.2A, and later adjusted to 0.1A.

Any robot with enough energy to keep its hardware running is counted as *alive*; this status is changed to *dead* as the battery level reaches zero. Once *dead*, a robot will not perform any movement, or absorb energy to recharge its battery until the end of the simulation.

Table 4.4 List of robot parameters, their respective functions and standard simulation values

ROBOT PARAMETER	UNIT	APPLICATION
$V_{\max}$	m/s	Maximum locomotion velocity
$i_{\max}^{in}$	amps (A)	Optimum solar current
$i_{\max}^{out}$	amps (A)	Motor draw at maximum power
BMR	amps (A)	Basal Metabolic Rate: cost of running systems when not moving
$Bat_{\max}$	Ah	Battery capacity
$BatCharge$	%	Initial battery charge
$SensNoise$	%	Sensor Noise

For the set of experiments presented, the hardware settings of all the robots were kept the same: the battery capacity is 6Ah, the optimum input current is 1A and the maximum output (motor current) is 1.4A.

The virtual robots are capable of moving forward at variable speed and turning to any direction (0-360°), and are equipped with a single light sensor, a battery, and a solar panel. Aiming at future application on physical platforms, all the robot's parameters are grounded in real hardware (Table 4.4).

4.3. Methodological Developments in Optimisation and Genetic Analysis

This section will describe the early experiments and advances leading to the definition of:

- The techniques for adapting the algorithms and the simulation program to the optimisation with evolutionary algorithms;
- The optimisation problem;
- Best suited EAs for the optimisation problem;
- Techniques for evaluating the performance of the robots in order to compose a model of objectives (and costs) to be optimised.
- Environmental settings that would be different enough to give rise to genetic variability in the solutions;
- Methods, tools and techniques to preserve and identify genetic variability among the solutions evolved;

4.3.1. Multiparameter Optimisation with Evolutionary Algorithms

The initial attempts to utilise Evolutionary Algorithms (EAs) to optimise the input parameters of the Foraging Algorithms developed in the present work occurred in Series A and B. Besides the Foraging Algorithm and the Simulation Framework developed by the author, these experiments also utilised a program to perform optimisation with EAs developed by the author. From Series C to J, the program to optimise with EAs was replaced by an open-source program named FEX-GODLIKE[f], that offered more flexibility and control over the optimisation settings. Some initial modifications were made to the original code of the program in order to allow more experimental data to be collected. This improved version was the same used for all the experiments after Series G.

A more stable experimental design could only be achieved after experiment series E and F, which broadly explored the limits of the algorithm regarding the range of parameters and costs, field area and quality of resources available, as well as simulation and robot settings, such as Time Step Size, duration of the simulation, battery size and initial charge.

4.3.2. Foraging Algorithms and Models of Cost

Series A-C tested a state-based version of *C. elegans* Foraging Algorithm, SAT-4.1, with 5 parameters (Tables 4.5, 4.6, and 4.7). In this beta version, the behaviour of the robot would be controlled by the 5 input parameters, and dependant on its internal state and on the conditions of the environment. Depending on the charging status of the robot's battery they would lay an egg (representing the execution of a task, as explained in Section 4.1.1). Also, depending on how low their battery charging level was, their foraging behaviour would be modified. This algorithm is explained in depth on the Section 4.1.1. Also, for these experiments, the objectives of the optimisation were combined in a single *Fitness Function* (*FF*), that outputs a single number encompassing the following: number of living robots at the end of the simulation; total energy collected and stored by the remaining living robots, and the number of eggs laid during the simulation. Three different combinations and calculation methods for each of these values were tested over series A to C. Having a unique Fitness Function speeds up the optimisation process, however, it might force premature decisions about the importance of each objective in relation to the others and will sometimes lead to optimisations. Also, having a single FF

[f] FEX-GODLIKE is available at <https://github.com/rodyo/FEX-GODLIKE>. Copyright © 2018, Rody Oldenhuis. Copyright © 2006, Joachim Vandekerckhove. All rights reserved.

makes it infeasible to decompose the fitness value and to analyse its components separately afterwards.

Series D-G tested a reflex-agent foraging algorithm, AT-5, with 5 input parameters. Unlike SAT-4.1, the decision process of the robots running the algorithm is based only on the conditions of the environment, ignoring any internal state. In this new simulation model, the robotic agents are no longer set to "lay eggs" (representing the execution of a task), but instead evaluated by the amount of energy that they collected over the course of the simulation. Another key difference of that series compared to previous series was that instead of a single Fitness Function summarizing many metrics, the parameters were optimised for independent objectives. On Series D, four objectives were evaluated: number of individuals alive at the end of the simulation; total of energy collected; total remaining energy at the end of the simulation; and total energy spent. On Series E and F, three objectives were optimised: number of individuals alive at the end of the simulation; remaining energy stored in the robots' batteries; and the area of the field explored by the robots. On Series G-J, a new and definitive set of three objectives was optimised: the survival rate of the robots in all the time steps; the remaining energy in the robots' batteries at the end of the simulation; and the ratio of the area of the field explored by the robots. The experiments from series G-J test four different Foraging Algorithms (AT-5, AT-6, ATRP-8, and ATRP-7, respectively), but utilise the same model of costs, and the results are therefore comparable.

4.3.3. Experiment Series A, B, and C (Foraging Algorithm SAT-4.1)

In Series of experiments A, B, and C, eight Field Sizes (100, 140, 150, 170, 175, 200, 225, and 250) and two variations of resource quality (Rich and Average) were tested. In terms of optimisation, two EAs were implemented into two programs, then evaluated, as described earlier in this chapter. The experiments were an attempt to test the basic functioning of the simulation framework and of the heuristics model.

In experiment series A, the optimisation for Environments 100R, 170R and 140A resulted in very similar Costs (Table 4.5). Also, contrary to expectations, the minimised cost for 140R was lower than that of 100R. These results raised questions about the quality of the EA chosen for the optimisation (DE), the implementation of the EA itself, as well as of the methods for calculating Fitness.

Table 4.5 Results of experiment Series A. Minimised Cost was obtained from the Fitness Function.
Column "Number of Function Evaluations" indicates the number of runs.

OPTIMISATION		SIM.	ROBOT	ENVIRONMENT		Number of Function Evaluations	Minimised Cost
EA	Pool Size	Time Step	Sensor Noise	Field Size	Quality of Resources		
DE	100	20	1%	100	R	10,200	183.8
				140		12,720	137.7
				170		10,320	186.6
				140	A	10,920	183.8
					TOTAL:	**33,240**	

Table 4.6 Results of experiment Series B. Minimised Cost was obtained from the Fitness Function.
Results highlighted in red represent issues on Fitness Estimation and conversion to Cost.
Absent results (represented with a Dash) indicate simulations that crashed.

OPTIMISATION		SIM.	ROBOT	ENVIRONMENT		Number of Function Evaluations	Minimised Cost
EA	Pool Size	Time Step	Sensor Noise	Field Size	Quality of Resources		
GA	100	20	1%	100	R	6,700	108.1
				150		9,500	145.7
				175		5,300	198.0
				200		8,200	4,781.3
				225		7,000	447.8
				250		13,200	618.9
GA	100	20	1%	100	A	12,120	132.4
				150		12,000	191.2
				175		10,320	362.0
				200		14,400	99,620,260,807.1
				225		9,700	99,616,434,374.0
				250		-	-
					TOTAL:	**108,440**	

A new series of experiments (Series B) was conducted after a few corrections in the simulation program and on some methods of estimating costs, as well as the implementation of a different EA (GA). Overall, the results for most of the environments appeared to have improved, there was a clearer variation of costs according to the size of the environment and the quality of resources available, as was expected (Table 4.6). A major issue with some of the optimisation results appeared in 200R, 200A, and on 225A,

highly discrepant costs from the rest of the environments tested presented, despite their number of function calls being similar to the others (Table 4.6).

In experiment series C a new program for Optimisation with EAs was tested (FEX-GODLIKE) and since GA seemed to have provided good results on Series B, the same EA was maintained in series C. The issues resulting in costs out of the expected range in experiment series B were also identified and fixed. The results obtained in Series C seemed to be compatible with the variation in size and resource quality of the environments (Table 4.7). The new optimisation program used in series C also seemed to be quicker and more robust than that tested in Series A and B and was used on the experiment series to come.

Table 4.7 Results of experiment Series C. Minimised Cost was obtained from the Fitness Function.

OPTIMISATION		SIM.	ROBOT	ENVIRONMENT		Number of Function Evaluations	Minimised Cost
EA	Pool Size	Time Step	Sensor Noise	Field Size	Quality of Resources		
GA	100	20	1%	150	R	5,400	566.9
				175		8,900	585.2
				200		6,900	654.0
				225		12,400	807.1
				250		7,400	858.5
GA	100	20	1%	150	A	7,400	655.1
				175		10,500	774.9
				200		7,700	890.2
					TOTAL:	**66,600**	

4.3.4. Experiment Series E and F (Foraging Algorithm AT-5)

These series of experiments contributed for:

- The definition of the best suited EAs for the optimisation problem, through a comparison between three classic EAs: Genetic Algorithms, Differential Evolution, and Particle Swarm Optimisation;
- The calibration of the parameters to control the EAs, as well as general optimisation parameters such as stopping criteria and boundaries.
- The definition of standard environment sizes and availability of resources;
- The definition of simulation settings, such as the Time Step.

4.3.4.1. Experiment Series E: Subsets E1 and E2

These experimental sub-sets aimed at finding the best suited Evolutionary Algorithms and parameter settings for this particular problem. Regarding the Simulated Environments, three field sizes were tested (150, 250, and 280), and two levels of Resource Quality: Rich and Average. In terms of Optimisation settings, three EAs were tested: Genetic Algorithm (GA), Differential Evolution (DE), and Particle Swarm Optimisation (PSO). Different Pool Sizes (the total number of individuals composing each generation) were also tested, being: 100, 200, 400, and 800 (Tables 4.8 and 4.9).

For comparison's sake I made adjustments to the following parameters:

- For Field Size 150 Rich in resources, I tested 2 Pool Size settings: 100 and 200. For each Pool Size number.
- For Field Size 250, I tested both Rich and Average resource qualities, both with a Pool Size 200. For each resource quality setting,
- For Field Size 250 with Average resource quality, I tested three Pool Sizes: 100, 200, and 400. For each Pool Size.
- For Field Size 280 and a Pool Size 200, I tested both Rich and Average resource quality parameters. For each resource quality setting.
- For every combination between Field Size, Resource Quality, and Pool Size, all 3 EAs (GA, DE, PSO) were tested

This allowed me to gain a better understanding of the range of costs and the limits of the application of *C. elegans*' inspired Behavioural Algorithm regarding the area and resource availability.

More tests were performed in Environment 250A than in the other environments, since it had an average size and resource quality and the results were expected to be extrapolated to both easier and harder environments.

In Experiment Series E - Subset E1, three Field Sizes were tested (150, 250, and 280), and two Quality of Resources (Rich and Average). For some variations of environments, different Pool Sizes were also tested (100, 200, and 400). For each combination of environment type and Pool Size, all three EAs were tested (GA, DE, and PSO). A compilation of the experiments in Subset E1 is presented in Table 4.8.

All the optimisation cycles for Field Size 150 with Rich quality resources resulted in Cost 1 minimised to zero.

Table 4.8 Compilation of experiments (Series E, Subset E1) with different parameter settings for optimisation with EAs

ENVIRONMENT		OPTIMISATION		Number of Function Evaluations	Minimised Cost 1	Minimised Cost 2	Minimised Cost 3
Field Size	Quality of Resources	Pool Size	EA				
150	R	100	GA	33,397	0.000	0.997	0.967
			DE	31,690	0.000	0.997	0.967
			PSO	4,003	0.000	0.998	0.967
		200	GA	33,398	0.000	0.997	0.967
			DE	32,063	0.000	0.997	0.967
			PSO	8,604	0.000	0.998	0.967
250	R	200	GA	33,400	0.022	0.998	0.954
			DE	32,513	0.034	0.999	0.944
			PSO	4,007	0.147	0.999	0.944
	A	100	GA	33,398	0.183	1.000	0.944
			DE	29,353	0.196	1.000	0.944
			PSO	2,237	0.459	1.000	0.944
	A	200	GA	33,399	0.198	1.000	0.944
			DE	30,867	0.163	0.999	0.944
			PSO	3,185	0.352	1.000	0.944
	A	400	GA	33,599	0.189	1.000	0.944
			DE	-	-	-	-
			PSO	10,355	0.300	1.000	0.944
280	R	200	GA	33,400	0.138	0.999	0.944
			DE	4,856	0.159	0.999	0.935
			PSO	3,604	0.359	1.000	0.936
	A	200	GA	33,400	0.381	0.999	0.935
			DE	32,383	0.349	1.000	0.935
			PSO	3,281	0.596	1.000	0.936
TOTAL:				**500,392**			

As can be seen in Table 4.8 and Fig. 4.8, 4.9, and 4.10, in all of the environments (except the ones with Field Size 150), optimisations with PSO converged too early, achieving worse results than the other EAs tested. Two instances for comparison of the three EAs in identical environmental conditions and Pool Size settings can be seen in detail in Fig. 4.9 and 4.10. In both, PSO converged much earlier, while GA and DE achieved very similar results.

In light of the variance of the results achieved between GA and DE compared to PSO, it was decided that as a further experiment with PSO using a larger Pool Size would be tested.

Experiment Series E - Subset E2 extended the previous investigation (Subset E1) by adding an extra Field Size (320), to explore the limits of the Foraging Algorithm and by

testing a Pool Size of 800 for PSO, as in the previous Series (E2), most of the Optimisations with PSO seemed to have converged early.

As the Optimisation with EAs is stochastic, in this Series, I also aimed at validating the consistency of the results obtained on each optimisation cycle by running replicas of the optimisation cycles. The comparison between two different optimisation cycles under identical environmental and optimisation parameter settings also shed light on possible stochasticity artefacts, as well as on the importance of tuning the optimisation parameters to the best possible settings for the next experimental steps. The results of Subset E2 are presented in Table 4.9.

Further comparisons may also be drawn between the following variations:

- For Field Size 150 Rich in resources optimising with DE with Pool Size 100, I ran 2 independent optimisation cycles (with different random seeds).
- For Field Size 250 Average in resources optimising with GA with Pool Size 100, I ran 2 independent optimisation cycles (with different random seeds).
- For Field Size 280 Rich optimised with PSO, I compared three Pool Sizes: 200, 400, and 800.
- For the new Field Size 320 Rich optimised with GA, I compared two Pool Sizes: 100, and 200. Another instance of Field Size 320 with Average resource quality was also optimised by using PSO.
- PSO was tested on the optimisation for three different Field Sizes (250, 280, 320), all with Pool Size 400.

Table 4.9 Compilation of experiments (Series E, Subset E2) with different parameter settings for optimisation with EAs. Lines shaded in grey indicate results from previous subsets, presented again for comparison.

ENVIRONMENT		OPTIMISATION		Number of Function Evaluations	Minimised Cost 1	Minimised Cost 2	Minimised Cost 3
Field Size	Quality of Resources	Pool Size	EA				
150	R	100	DE	31,690	0.000	0.997	0.967
				30,839	0.000	0.997	0.967
250	A	100	GA	33,398	0.183	1.000	0.944
				33,399	0.547	1.000	0.997
320	R	100	GA	33,391	0.317	1.000	0.927
		200		33,397	0.296	1.000	0.927
280	R	200	PSO	3,604	0.359	1.000	0.936
		400		-	-	-	-
		800		20,471	0.313	1.000	0.935
250	A	400	PSO	10,355	0.300	1.000	0.944
320				8,145	0.661	1.000	0.929
TOTAL:				**238,689**			

Results and Discussion

Increasing the Pool Size for PSO seemed to have slightly improved the minimum cost achieved by it for this particular problem. In 250A, it improved from ~0.459 to ~0.352 and ~0.300 (pool size 100, 200, and 400, respectively). However, even with this improvement, the quality of results was still not satisfactory when compared with the other EAs tested (GA and DE, that achieved ~0.183 and ~0.163, respectively). As can be seen in Figure 4.8, both of the PSO optimisations for Environment 280R converged too early - after 18 (light blue line) and 25 (navy blue line) generations.

Optimisations with GA and DE achieved the best results of the three EAs (Fig. 4.8, 4.9, 4.10, and 4.11), and in most of these cases, both GA and DE ran for a similar number of generations. However, it came to our attention that in one of these cases (280R), DE achieved very similar results to GA after only 25 generations (Figure 4.11). This specific optimisation with DE had to be interrupted earlier due to hardware restrictions. However, as the DE algorithm achieved good results, even in such a short span of generations, it was decided that the matter should be further investigated.

Regarding GA, an interesting fact was noticed in the comparison of optimisation cycles for Environment 250A (Figure 4.12) in which two highly discrepant results were found for optimisations using Pool Size 100: while one of the optimisations minimised Cost 1 to ~0.183 (orange line), another instance only minimised it to ~0.546 (yellow line). This is likely due to the local minima problem. One of the factors contributing to this might be that the individuals of the first generation did not acquire a broad enough sample of the

solution space as the minimum Cost 1 for this generation is ~0.790 (yellow line) - a value much higher than the ~0.658 found in the other optimisation (orange line), and that those found by the other optimisation cycles in the same environment, but with different pool sizes: ~0.552 (green line), and ~0.503 (blue line). This was a one-time event in all the experiments.

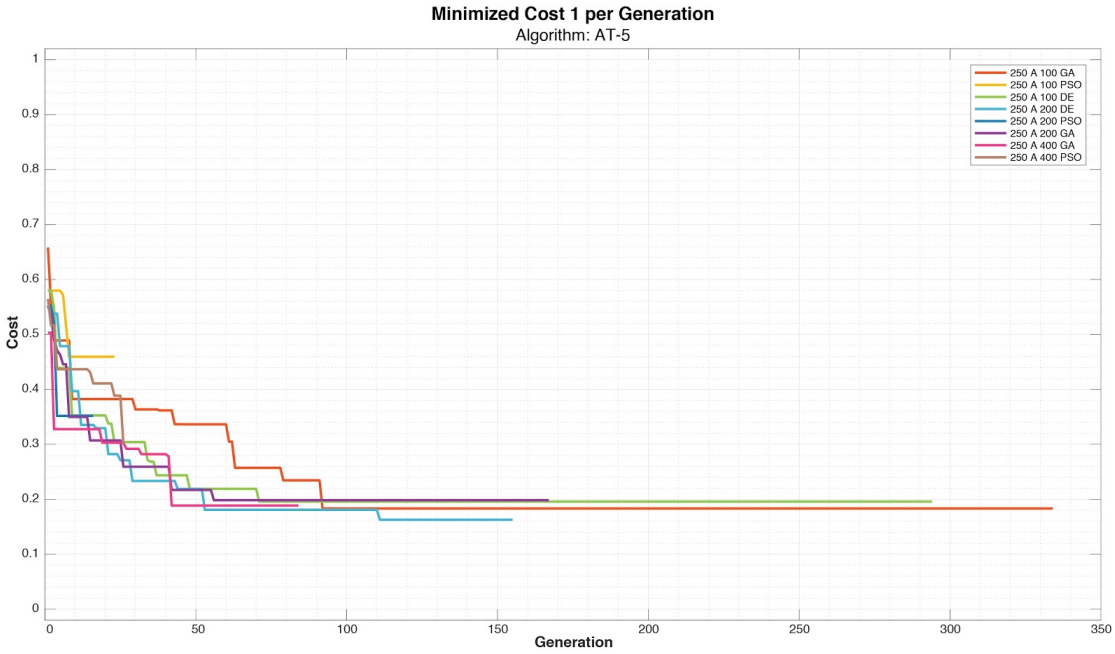


Figure 4.8 Comparison of optimisation cycles for environments of Field Size 250 with Average resource quality, optimised with GA, DE, and PSO, using a Pool Sizes of 100, 200, and 400.

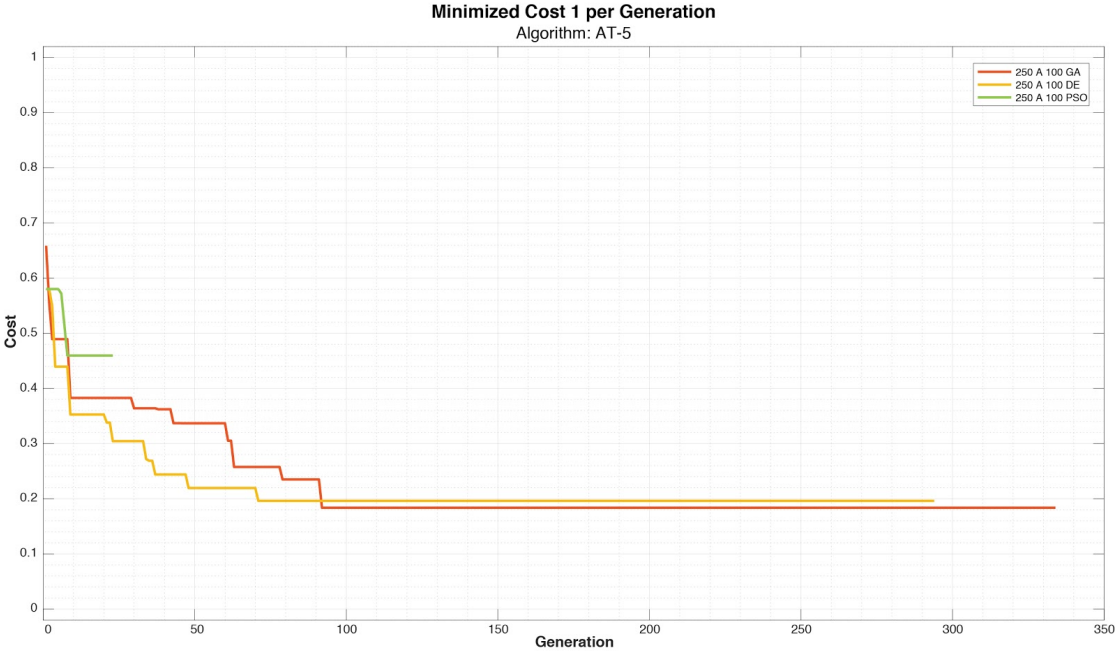


Figure 4.9 One instance of the experiments presented in Table 6.1: comparison of the minimisation of Cost 1 across Generations in 3 Evolutionary Algorithms (GA, DE, and PSO) using a Pool Size of 100 in the same environment (Field Size 250 with Average resource quality).

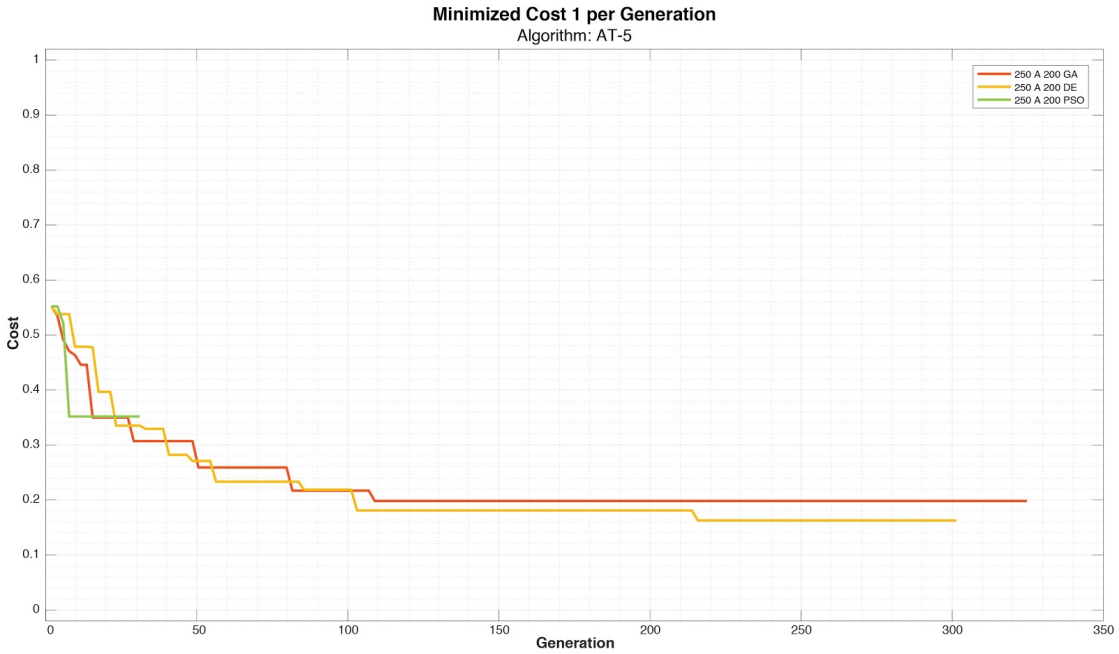


Figure 4.10 One instance of the experiments presented in Table 4.8: comparison of the minimisation of Cost 1 across Generations in 3 Evolutionary Algorithms (GA, DE, and PSO) using a Pool Size of 200 in the same environment (Field Size 250 with Average resource quality).

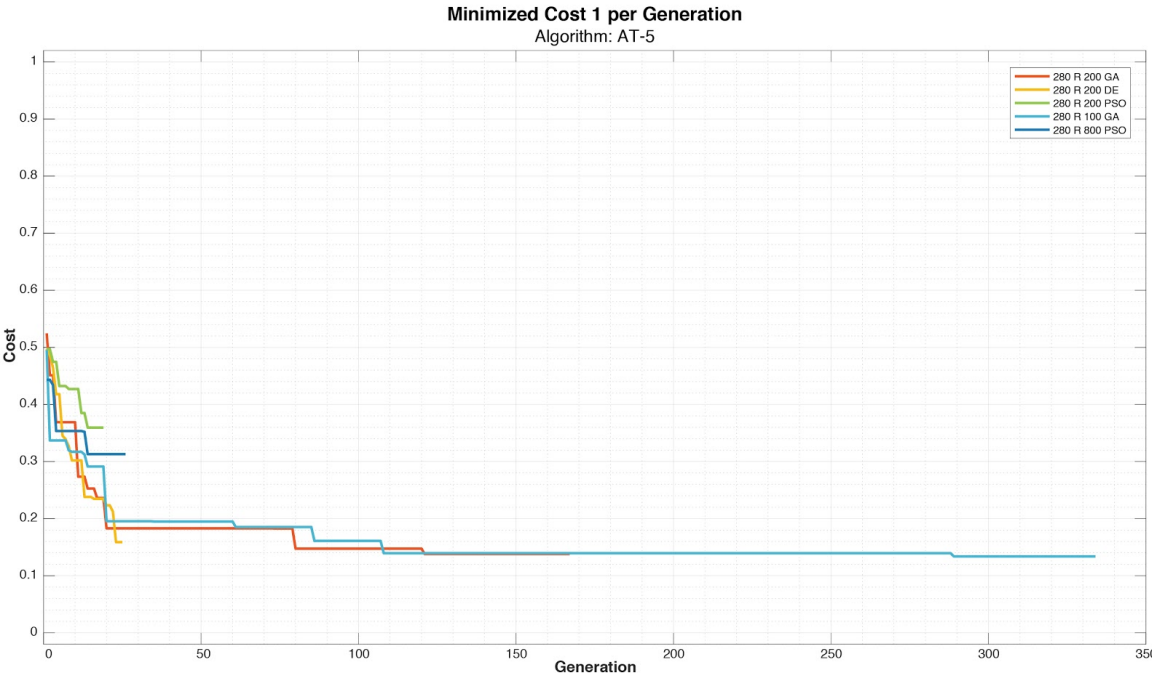


Figure 4.11 Comparison of optimisation cycles for environments of Field Size 280 with Rich resource quality, optimised with GA, DE, and PSO, using mixed Pool Sizes of 100, 200, and 800.

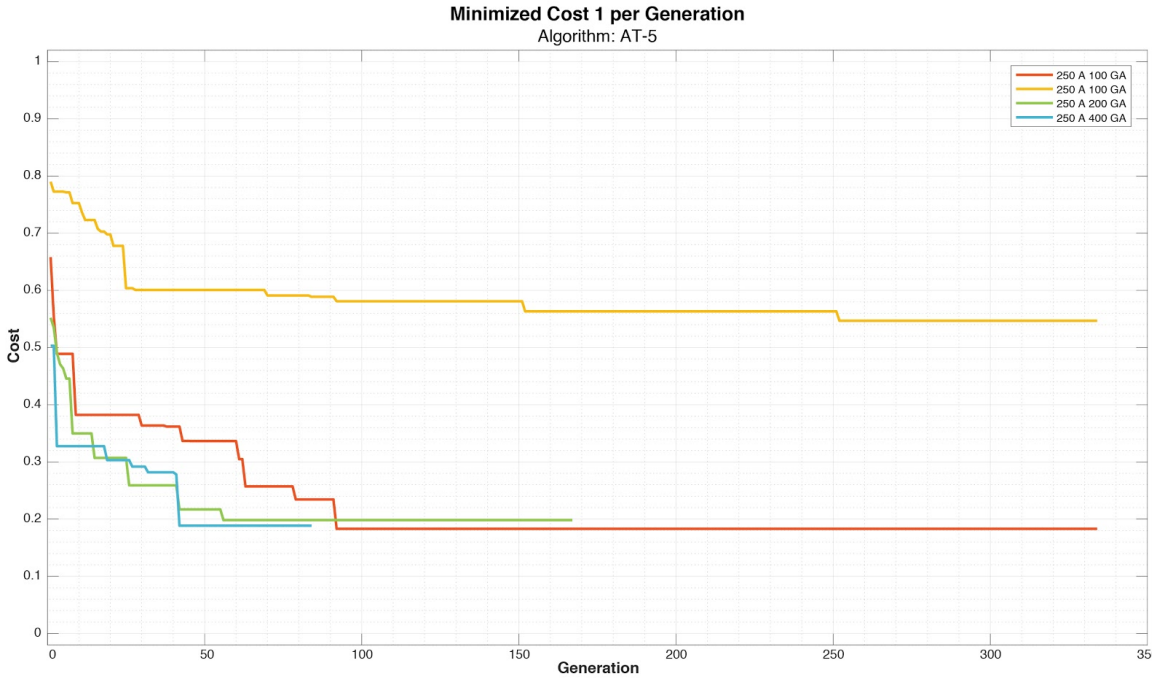


Figure 4.12 Comparison of optimisation cycles for environments of Field Size 250 with Average resource quality, optimised with GA, using Pool Sizes of 100, 200, 400.

4.3.4.2. Experiment Series E, Subset E3

In this subset of experiments I intended to compare different Field Sizes and investigate the limits of the environmental models - to find the most challenging environments for the robots to attempt and evolve within. A compilation of the tests and results is presented in Table 4.10.

Table 4.10 Compilation of experiments (Series E - Subset E3) with varying Environment parameters and constant optimisation with EAs.
Lines shaded in grey indicate results from previous subsets, presented again for comparison.

OPTIMISATION		ENVIRONMENT		Number of Function Evaluations	Minimised Cost 1	Minimised Cost 2	Minimised Cost 3
EA	Pool Size	Field Size	Quality of Resources				
GA	100	150	R	33,397	0.000	0.997	0.967
			A	33,399	0.000	0.998	0.967
		200	R	33,400	0.000	0.998	0.954
			A	33,397	0.035	0.999	0.954
		250	R	33,400	0.022	0.998	0.954
			A	33,398	0.183	1.000	0.944
		280	R	33,394	0.134	0.999	0.936
			A	-	-	-	-
		320	R	33,391	0.317	1.000	0.927
			A	33,391	0.510	1.000	0.929
TOTAL:				**300,567**			

Results and Discussion

After running the experiments, it was noted that even in the most challenging environment tested, with Field Size 320 and Average Resource Quality (320 A), the optimisation still managed to minimise Cost 1 to 0.510 (Table 4.10, Figure 4.13).

Furthermore, the results for environments of Field Size 150R, 200R, and 150A were very similar (Table 4.10). It was therefore decided to make the difference between Environments more noticeable, and to further explore the limits of the algorithm on prolonging the robots lifespan in more challenging environments.

It was decided that within future experiments there would be only 3 variations of Field Size (Small, Medium, and Large), and that a new variation of Resource Quality would be tested, resulting in a total of 3 resource quality parameters: Rich, Average and Poor.

As these experiments (Series E3) ran before the end of Series E2, and with the purpose of saving computational resources, a GA was used for the optimisation, this was comparable to series E1 and E2 as the majority of these series ran the GA. Furthermore as this series was run, before the E2 series completed, the superiority of DE was still unclear. For further experiments, though, DE was chosen as the EA to run the following optimisation cycles of the research, as it seemed capable of achieving better results faster than the other EAs tested.

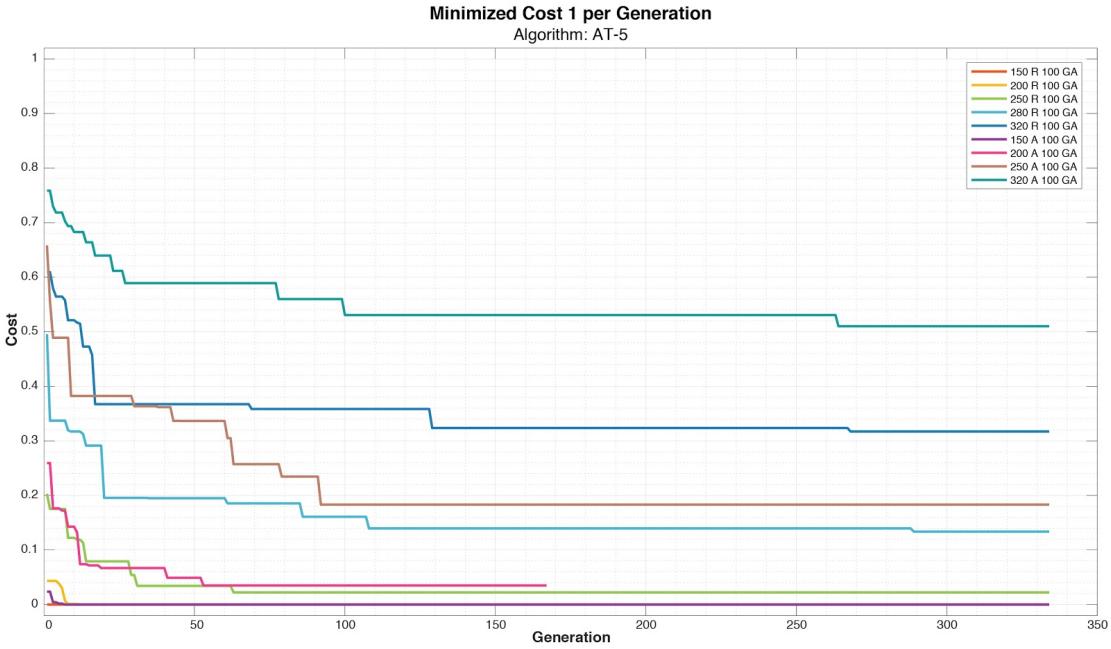


Figure 4.13 Comparison of optimisation cycles for environments of Field Size 150, 200, 250, 280, and 320 with Rich resource quality, and 150, 200, 250, and 320 with Average resource quality. All of them were optimised with GA using a Pool Size of 100.

4.3.4.3. Experiment series F

Here I will cover the methods and results of experiment series F, that compared different values for Time Step Size. This series aimed at testing variations/disturbances in Costs occurred when running the simulation with different Time Step Size values, as well as detecting the limits of the Foraging Algorithm on the current simulation framework regarding the frequency in which the behavioural decisions are made.

Time Step Size (*TSS*) is the minimum Unit of the Virtual Time. Virtual time is the amount of time passed in the simulation (which is different from the time the simulation actually takes to run on a computer). In the framework I built for my simulations, the value of the Time Step Size can be understood as the virtual time interval (in seconds) in which robots would engage in the decision process, the robots would run the algorithm to make decisions each time step. Time Step Size 1 would mean they make a decision every second; value 4, they would make a decision every 4 seconds; and value 0.5, every half a second.

This experiment series was of particular importance because: Running a full cycle of optimisation would usually take about 30,000 function calls and at this point of the research, each full simulation from Series E would take no less than 30 seconds (considering 100 robots simulated on a medium-sized field during a virtual time of 24 hours, with Time Step Size 20). The point of these experiments was to discover how

much would be compromised in the simulation results by using larger Time Step Sizes (as the simulation would lose "resolution" with higher values of Time Step Size). As it is an aim of this word that the code would be made available for other researchers, it seemed important to make sure the program would run on regular computers.

Experiment Design, Results and Discussion

Using the same models of environments and Costs of Series E, the best 80 DNAs evolved in each environment (of Series E) were used for the experiments of the present series. A total of 44 values of Time Step Size (ranging from 0.0001 to 7500) were tested in 1,335 random variations of 2 environments: the first with Field Size 150 and Rich quality of resources (150R), and the second with Field Size 250 with Average quality of resources (250A). The Results obtained with each TSS were later compared to those obtained in the same environment using TSS 1 (which was considered to be the reference value for TSS) (Figure 4.14). The value Cost 3 is affected by both smaller and larger TSS, for two different reasons: as Cost 3 calculates how many spots on a field the robot has visited, so , naturally, with more time steps, the number of unique visited spots (with different x and y coordinates) is expected to increase, and so, in simulations with lower numbers of TSS, the number of spots visited will be naturally lower. For further versions of the program, the method for calculating Cost 3 has been improved to allow more fidelity in simulations in which the Time Step Size has been altered.

An unexpected outcome of this Series of experiments was that it revealed an issue with the method of calculating Cost 2 (that became evident on the test results for environment 250A). The issue was solved on further versions of the Simulation program.

Time Steps of integer values up to 20 presented fairly accurate results for all the Costs, as the mean squared error was below 0.05 in all cases (after the issue with Cost 2 was sorted) (Figure 4.9). For Cost 1 only, regardless, it was decided that on further experiments Time Step Size 1 should be preferred over other alternatives whenever possible. During the realisation of Series E of experiments, several issues related to both hardware and software caused the optimisation cycles to crash and that resulted in the loss of data (as can be seen on Tables 4.6, 4.8, 4.9, and 4.10). In Series E, a full cycle of optimisation would take approximately 16 days, running in parallel on 9 computers (one for each environment). Within the following experiment series, some modules of the simulation and optimisation programs have been re-written, in order to export preliminary results and therefore prevent data loss. Some other modules have also been created for the program as to allow the re-starting of optimisations that crashed.

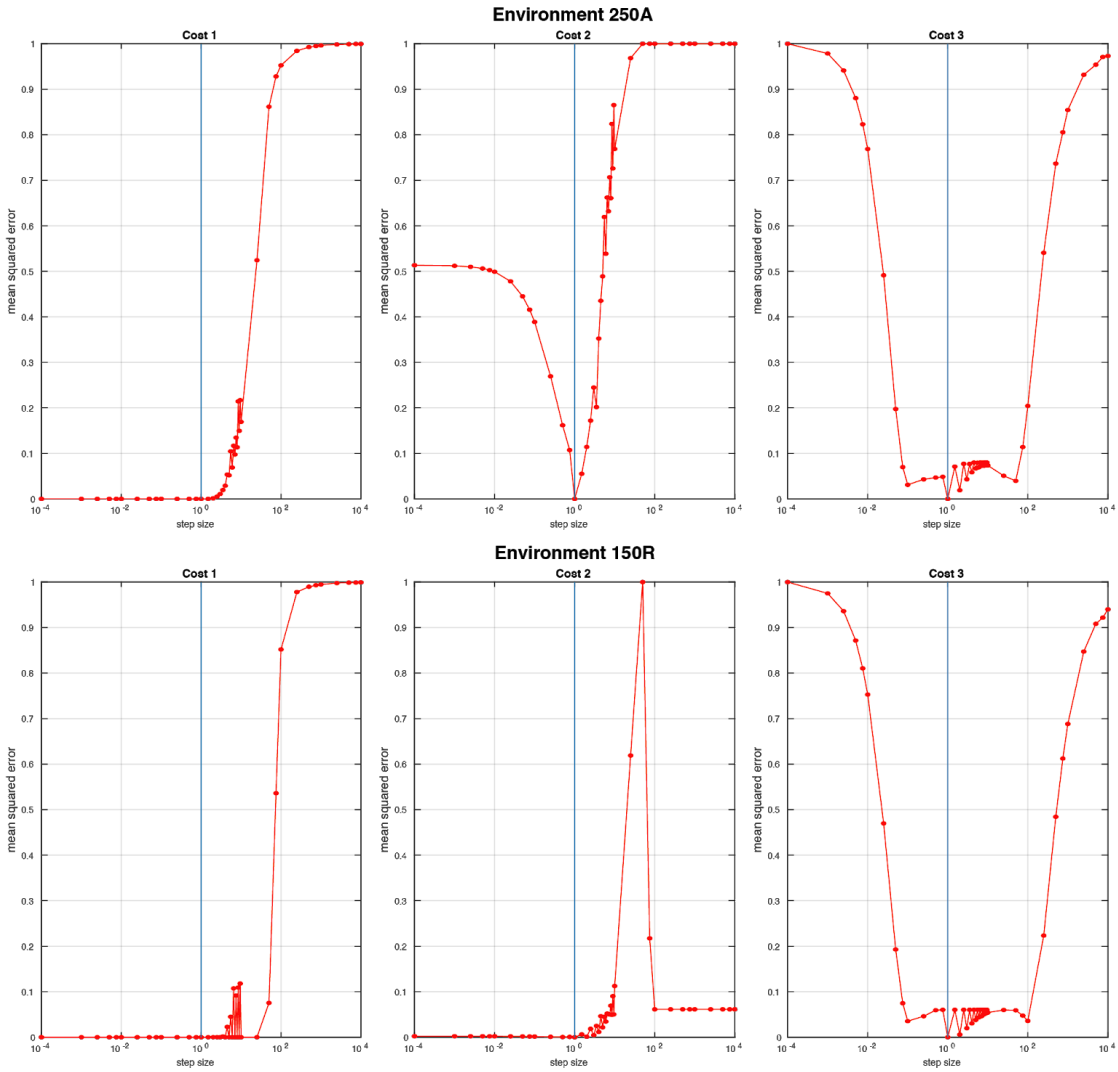


Figure 4.14 Comparison of results obtained with different Time Step Size values for Costs 1 to 3. The variation is measured in mean squared distance from the reference value (Time Step Size of value 1).

Top: Results obtained for Environment 250A. The blue vertical line indicates TSS 1 (reference value). Bottom: Results obtained for Environment 150R; The values tested (x axis) were: 0.0001, 0.001, 0.0025, 0.005, 0.0075, 0.01, 0.025, 0.05, 0.075, 0.1, 0.25, 0.5, 0.75, 1, 1.5, 2, 2.5, 3, 3.5, 4, 4.5, 5, 5.5, 6, 6.5, 7, 7.5, 8, 8.5, 9, 9.5, 10, 20, 25, 50, 75, 100, 250, 500, 750, 1000, 2500, 5000, and 7500.

4.3.5. Experiment Series G (Foraging Algorithm AT-5)

In this series of experiments (Series G), the parameters of Algorithm AT-5 were optimised for 9 variations of environments: SR, MR, LR, SA, MA, LA, SP, MP, and LP. Robots running the Foraging Algorithm AT-5 were tested with three sensor noise settings: 1%, 0.5%, and no sensor noise (Table 4.11).

The results obtained in all 9 environments were compared amongst each other, and to those obtained with three different sensor noise values:

- Value 1% is a realistic sensor noise value.
- Value 0% (no sensor noise) was tested as a proof of concept for the Foraging Algorithm itself, given that no sensor noise would represent ideal conditions for a robot to operate, however that is not very realistic.
- Value 0.5% was tested after the experiments with no sensor noise achieved outstanding results (in all the environments, the cost was reduced to zero), and aimed at providing a better comprehension of how the Foraging Algorithm was affected by different hardware conditions.

To make the experiments viable in terms of the computation time required, the simulation program used on the previous experiment series had to be drastically remodelled. This remodelling involved changes on the format the data is stored (privileging tables), avoidance of *For* Loops whenever possible, pre-allocation of variables, interruption of the simulation in case all the robots die, and encapsulation of functions and operations related to the plotting and recording of the simulation. The optimisation of the code presented a considerable improvement in the computational time required for each simulation and the average time per function call dropped from 218 to 36 seconds (considering 100 robots/samples running with time step size 1).

In this set of experiments, each optimisation cycle took between 1 and 25 days, and the optimisation for each environment ran in parallel on a different computer. As for this step 18 computers were made available enabling me to run the optimisations for 1% and 0% noise in parallel and as some of the optimisations would finish earlier (usually the ones in more challenging environments). I also managed to run several replicas of the optimisation for some environments. At the end of this stage, the replicas with the best results were considered.

The first batch of experiments evolved solutions for the 9 environments running in virtual robots which sensor noise was 1%. As a validation step for the algorithm itself, and for the hardware settings used in the virtual robots, a second batch of optimisation experiments ran in the same environment types, but with robots with sensor noise 0%. The results for this batch of experiments (9 environments and sensor noise 0%) exceeded expectations, and in all 9 environments, all 100 robots managed to complete the entire simulation time (24 virtual hours).

As such a small sensor noise seemed to have changed the results so drastically, a third batch of optimisations tested robots with sensor noise 0.5% and the results of these three optimisation steps (with sensor noise 1%, 0.5%, and 0%) were compared (Table 4.11).

Table 4.11 Summary of the parameters and main results of the experiments in Series G, subset G.1: Optimisation of Foraging Algorithm AT-5 in 9 environments (SR, MR, LR, SA, MA, LA, SP, MP, and LP) with attractants only, for robots equipped with sensors with sensor noise of values 1%, 0.5%, and 0%. The Number of Function Evaluations indicates how many times the simulations ran.

ROBOT	ENVIRONMENT		Number of Function Evaluations	Minim. Cost 1	Minim. Cost 2	Minim. Cost 3	Non-Dominated Solutions	Density of Resources / Light Index (AT)
Sensor Noise	Quality of Resources	Field Size						
1%	R	S	9,253	0.000	0.729	0.657	116	42.4716
		M	11,313	0.147	0.916	0.854	15	19.0452
		L	23,153	0.530	0.988	0.938	5	10.6179
	A	S	13,445	0.179	0.943	0.725	39	27.3432
		M	31,946	0.634	0.996	0.911	28	12.2613
		L	32,097	0.696	0.999	0.957	18	6.8358
	P	S	24,493	0.505	0.999	0.838	67	13.6716
		M	5,112	0.666	1.000	0.931	32	6.1306
		L	6,228	0.735	1.000	0.962	15	3.4179
0.5%	R	S	33,219	0.000	0.683	0.639	48	42.4716
		M	32,852	0.029	0.847	0.841	57	19.0452
		L	30,413	0.386	0.972	0.925	8	10.6179
	A	S	25,035	0.037	0.882	0.697	51	27.3432
		M	30,402	0.478	0.989	0.893	10	12.2613
		L	32,091	0.690	0.998	0.953	41	6.8358
	P	S	20,016	0.504	0.999	0.842	68	13.6716
		M	3,240	0.697	1.000	0.932	10	6.1306
		L	1,143	0.817	1.000	0.964	5	3.4179
0%	R	S	30,653	0.000	0.553	0.632	65	42.4716
		M	23,975	0.000	0.576	0.822	55	19.0452
		L	33,230	0.000	0.638	0.923	50	10.6179
	A	S	33,183	0.000	0.716	0.648	37	27.3432
		M	33,139	0.000	0.771	0.845	46	12.2613
		L	53,570	0.000	0.622	0.888	51	6.8358
	P	S	33,212	0.000	0.918	0.796	78	13.6716
		M	59,593	0.000	0.833	0.891	45	6.1306
		L	57,045	0.000	0.842	0.942	37	3.4179
		TOTAL:	**723,051**					

For this series of experiments, the Robot's hardware parameter maintained the same settings as in Table 4.12. The robots started the simulation with their batteries 40% charged. All the optimisations ran until one of the stopping criteria was reached, following the default parameters of the FEX-GODLIKE toolbox:

- 10^5 function calls;

- diversity loss - if the individuals in a certain population evolved to be too similar to each other;
- after 10 generations the cost did not reduce in more than 10^{-4}.

The results of the three main simulation sets (1%, 0%, and 0.5% Sensor Noise) will be presented separately, in the following sections.

Table 4.12 List of robot parameters, their respective functions and standard simulation values

ROBOT PARAMETER	STANDARD VALUE	APPLICATION
$V_{\max}$	0.5m/s	Maximum locomotion velocity
$i_{\max}^{in}$	1.0A	Optimum solar current
$i_{\max}^{out}$	1.4A	Motor draw at maximum power
BMR	0.2A	Basal Metabolic Rate: cost of running systems when not moving
$Bat_{\max}$	6Ah	Battery capacity
$BatCharge$	40%	Initial battery charge

4.3.5.1. Experiments with Sensor Noise 1%

In this set of optimisation cycles for AT-5, in which the robot sensor has 1% noise, the best minimised Cost 1 was found in SR, MR, SA, SP, LR, MA, MP, LA, and LP, respectively (Table 4.11, Figure 4.15).

Optimisations MP and LP reached the stopping criteria of diversity loss early, after 21 and 32 generations, compared to the others.

When it comes to Cost 2, the environments in which the optimisation achieved the best minimised costs were, respectively: SR, MR, SA, LR, MA, LA, SP, MP, and LP (Figure 4.16). Cost 2 seemed to be in general more sensitive to the quality of light available in the field, given that 3 of the 4 best minimised costs were achieved in rich environments, and 3 of the 3 worst costs were found in environments with poor quality of resources.

On Cost 3, the environments in which the optimisation achieved the best minimised cost were, respectively: SR, SA, SP, MR, MA, MP, LR, LA, and LP (Figure 4.17). Cost 3 seem to be more sensitive to the size of the field, considering that 3 of the 3 best values were found on fields of size small, as well as 3 of the 3 worst results were found in environments of size large. As Cost 3 represents the percentage of the field explored by the robot, it seems natural that robots exploring smaller environments, even when these are more challenging, would do better than those on larger fields.

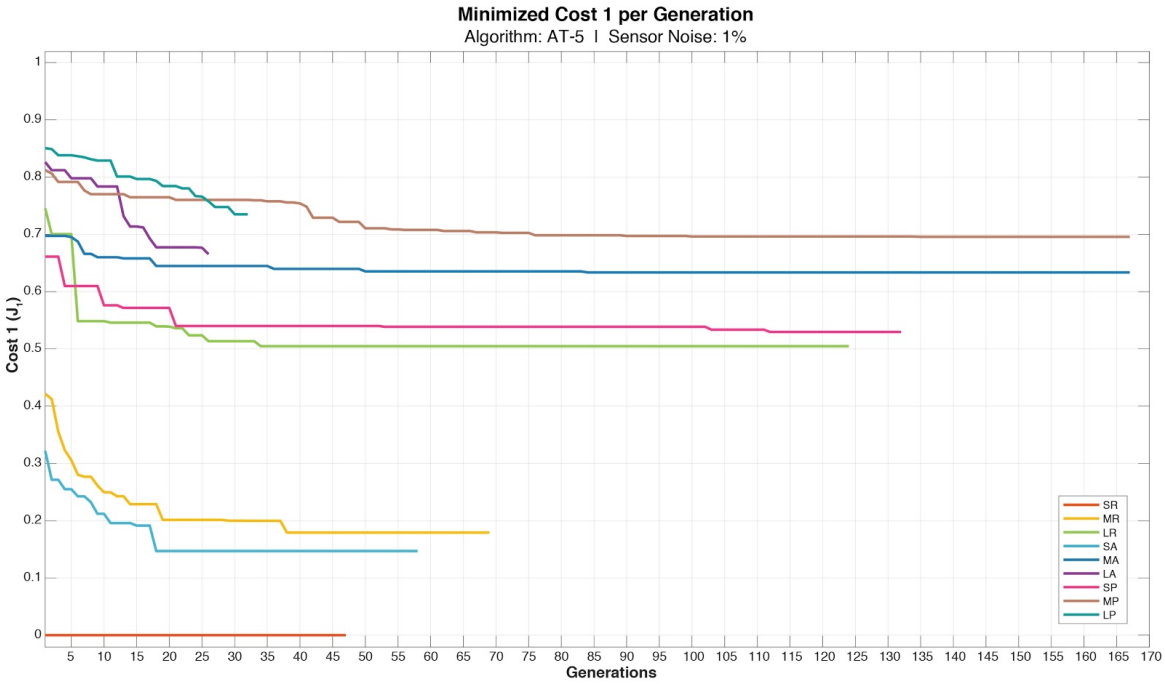


Figure 4.15 Cost 1 (J_1) optimised for Behavioural Algorithm AT-5 and robots with sensor noise 1%, in 9 environments (SR, MR, LR, SA, MA, LA, SP, MP, and LP), with attractants only.

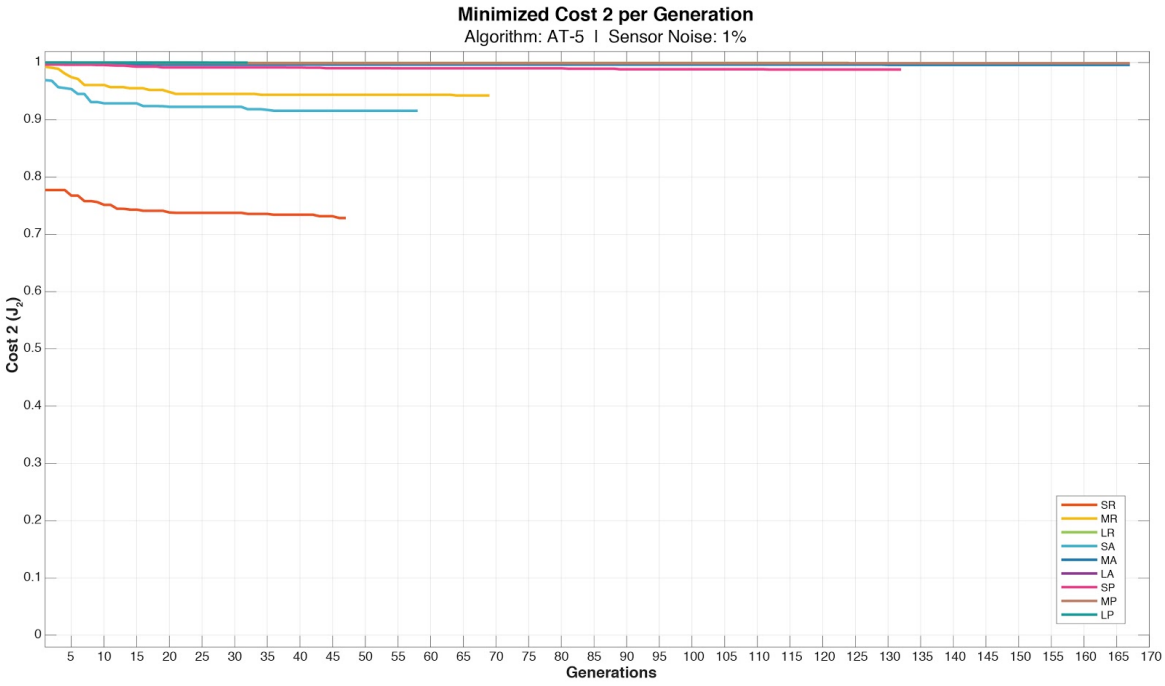


Figure 4.16 Cost 2 (J_2) optimised for Behavioural Algorithm AT-5 and robots with sensor noise 1%, in 9 environments (SR, MR, LR, SA, MA, LA, SP, MP, and LP), with attractants only.

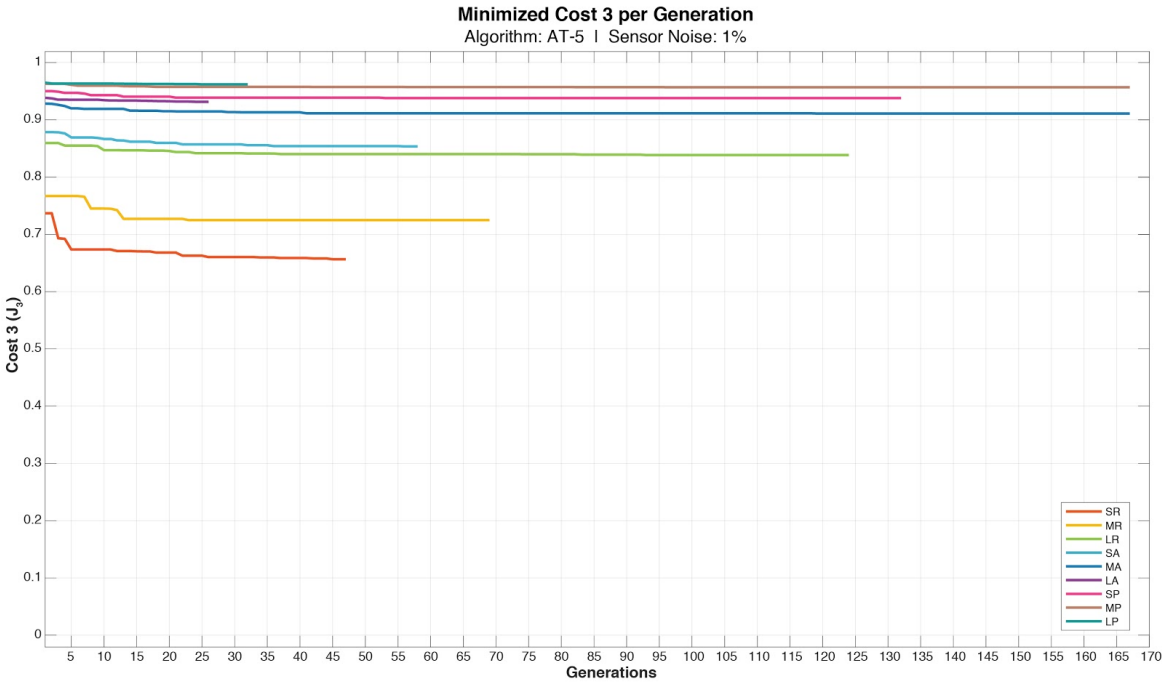


Figure 4.17 Cost 3 (J_3) optimised for Behavioural Algorithm AT-5 and robots with sensor noise 1%, in 9 environments (SR, MR, LR, SA, MA, LA, SP, MP, and LP), with attractants only.

4.3.5.2. Experiments with Sensor Noise 0% (no noise)

As a proof of concept, a version of the algorithm with sensor noise 0%, was deployed and optimised for the same set of environments. After only 7 generations, the individuals in all 9 environments reduced Cost 1 to zero, meaning that all the agents/robots in the simulation were able to discover the light spots quickly and to remain alive during the whole simulation (Figure 4.18).

The best values of Cost 2 were found, respectively, on SR, MR, LA, LR, SA, MA, MP, LP, and SP. This sequence confirms the pattern identified in the experiments with Sensor Noise 1%, in which the quality of the sources is more important than the size of the field (Figure 4.19).

As for Cost 3, the best values were found, respectively, on SR, SA, SP, MR, MA, LA, MP, LR, and LP. This result also confirms the observation that for Cost 3, the size of the field matters more than the quality of the resources (Figure 4.20).

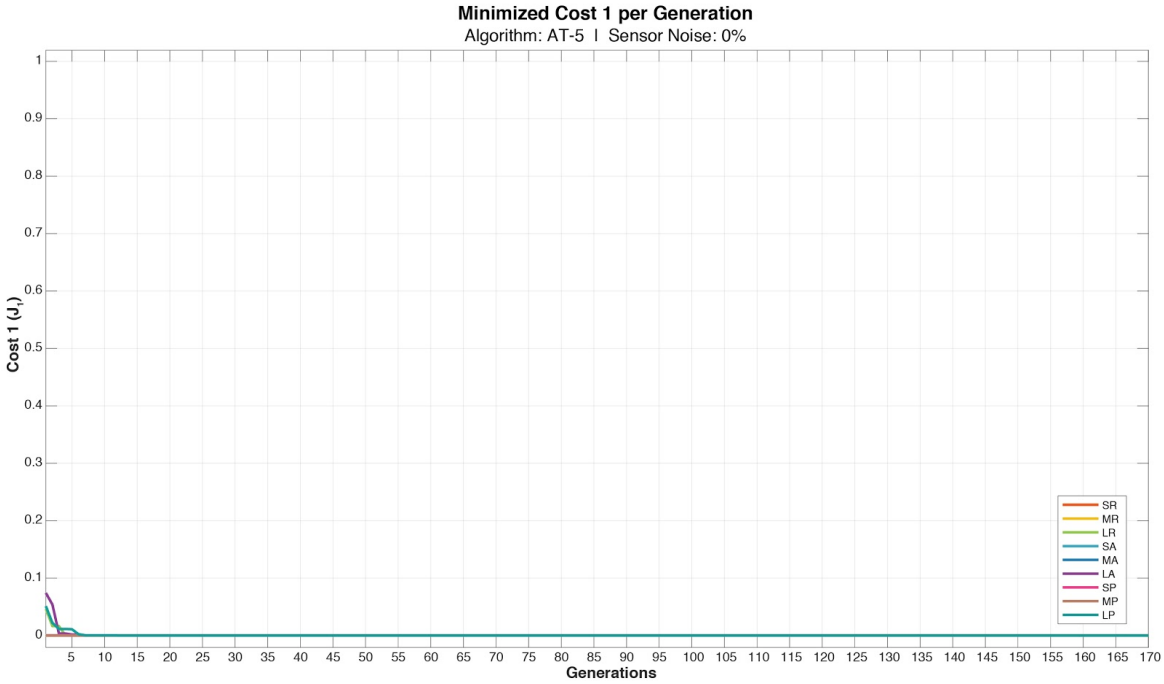


Figure 4.18 Cost 1 (J_1) optimised for Behavioural Algorithm AT-5 and robots with sensor noise 0%, in 9 environments (SR, MR, LR, SA, MA, LA, SP, MP, and LP), with attractants only.

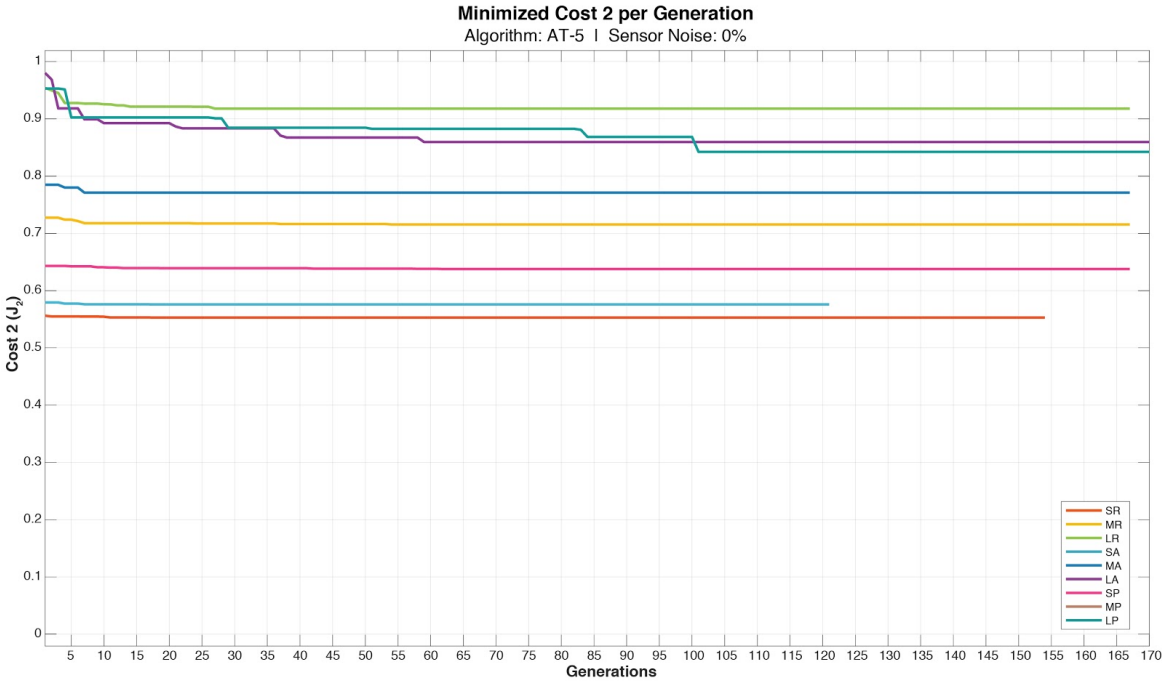


Figure 4.19 Cost 2 (J_2) optimised for Behavioural Algorithm AT-5 and robots with sensor noise 0%, in 9 environments (SR, MR, LR, SA, MA, LA, SP, MP, and LP), with attractants only.

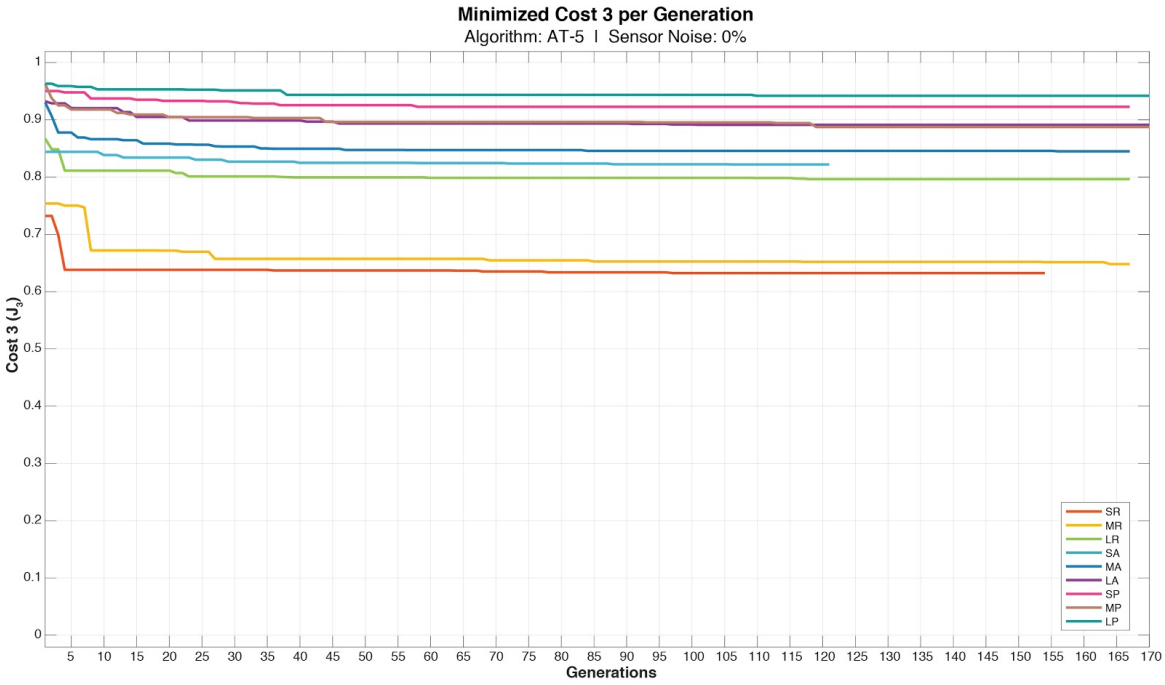


Figure 4.20 Cost 3 (J_3) optimised for Behavioural Algorithm AT-5 and robots with sensor noise 0%, in 9 environments (SR, MR, LR, SA, MA, LA, SP, MP, and LP), with attractants only.

4.3.5.3. Experiments with Sensor Noise 0.5%

The best results for Cost 1 were found, respectively, on SR, MR, SA, LR, MA, SP, LA, MP, and LP (Figure 4.21). These results are mostly consistent with those of the experiments with Sensor Noise 1% (SR, MR, SA, SP, LR, MA, MP, LA, and LP), with the exception of SP, that improved, from the 6th to the 4th position.

For Cost 2, the best results were found, respectively, on SR, MR, SA, LR, MA, LA, SP, MP, and LP (Figure 4.22). These results also confirm that the quality of the resources are more important than the size of the field for Cost 2.

For Cost 3, the best results (minima) were found, respectively, on SR, SA, MR, SP, MA, LR, MP, LA, and LP (Figure 4.23). The results also indicate, as observed previously, that the size of the field seem to be more important than the quality of the resources for Cost 3.

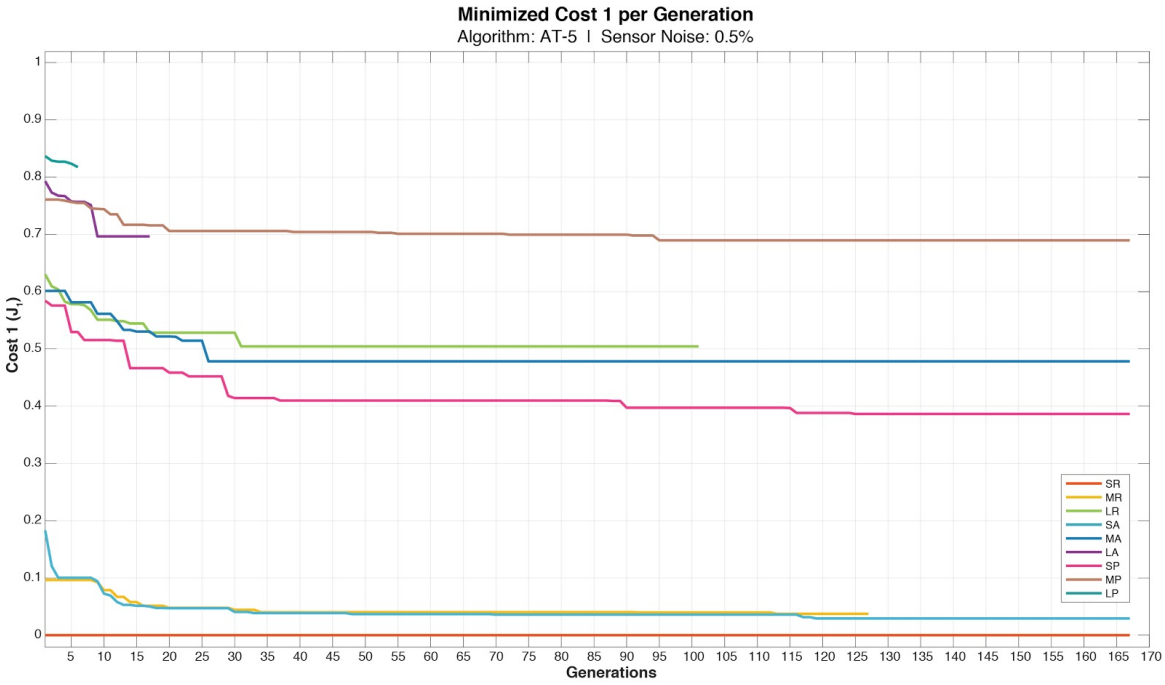


Figure 4.21 Cost 1 (J_1) optimised for Behavioural Algorithm AT-5 and robots with sensor noise 0.5%, in 9 environments (SR, MR, LR, SA, MA, LA, SP, MP, and LP), with attractants only.

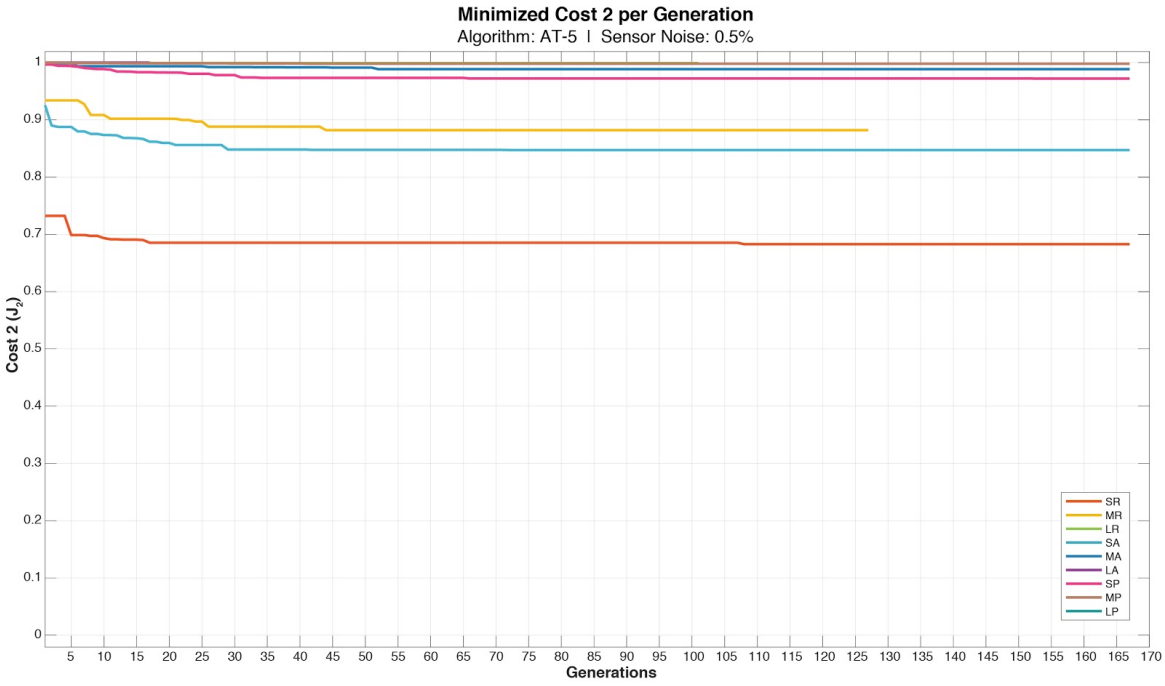


Figure 4.22 Cost 2 (J_2) optimised for Behavioural Algorithm AT-5 and robots with sensor noise 0.5%, in 9 environments (SR, MR, LR, SA, MA, LA, SP, MP, and LP), with attractants only.

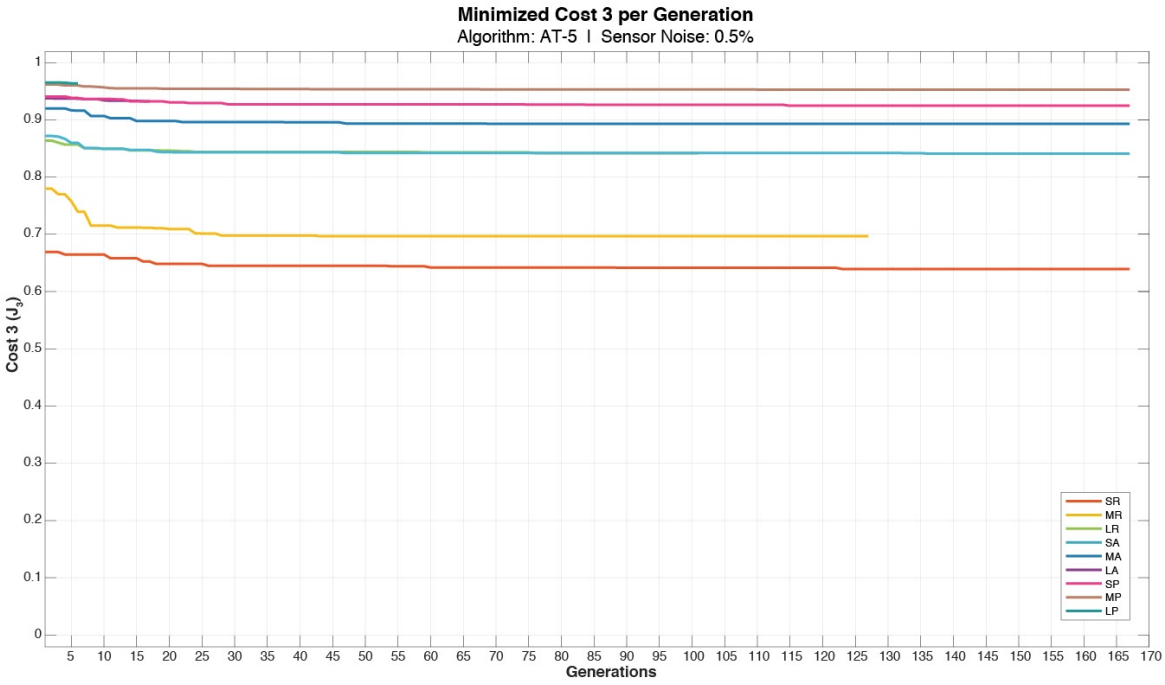


Figure 4.23 - Cost 3 (J_3) optimised for Behavioural Algorithm AT-5 and robots with sensor noise 0.5%, in 9 environments (SR, MR, LR, SA, MA, LA, SP, MP, and LP), with attractants only.

4.3.5.4. Genetic Analysis and Validation Experiments

The optimisation results obtained in Series G also served as the calibration data for a new set of analytical methods proposed in this thesis. The techniques developed consist of a series of experiments to verify the validity of the experiments regarding the following aspects:

- Level of difficulty of each environment as a validation method for the optimisation step;
- Ability of the best individuals evolved in one instance of a certain environment (e.g.: MR with seed 5420) to survive in other instances of the same environment (e.g.: MR with different seeds);
- Ability of the best individuals evolved in each environment to survive in the other 8 environments (cross-environment trials);

The experiments described in this section were only performed with the optimisation results for robots with sensor noise 1%, as this is a more realistic scenario in terms of real robotics hardware.

The level of difficulty of each environment was assessed by the *Light Index*, a measure of density of resources available. The resulting *Light Index* ranking (from the easiest to the hardest) is: SR, SA, MR, SP, MA, LR, LA, MP, and LP. Compared to the ranking of minimised Cost 1 for simulations with sensor noise 1% (SR, SA, MR, SP, MA, MP, LR,

LA, and LP), the only discrepancy is MP, that ranked 6th in cost minimisation, even though it seems to be the 8th most difficult environment. *Light Index* is, however, a simplified calculation and therefore might not capture all the important aspects that would make an environment harder or easier.

Table 4.13 - Environments in Series G, with minimised Costs and *Light Index* values.

ENVIRONMENT		Number of Function Evaluations	Minim. Cost 1	Minim. Cost 2	Minim. Cost 3	Density of Resources / Light Index (AT)
Quality of Resources	Field Size					
R	S	9,253	0.000	0.729	0.657	42.4716
	M	11,313	0.147	0.916	0.854	19.0452
	L	23,153	0.530	0.988	0.938	10.6179
A	S	13,445	0.179	0.943	0.725	27.3432
	M	31,946	0.634	0.996	0.911	12.2613
	L	32,097	0.696	0.999	0.957	6.8358
P	S	24,493	0.505	0.999	0.838	13.6716
	M	5,112	0.666	1.000	0.931	6.1306
	L	6,228	0.735	1.000	0.962	3.4179

4.3.5.5. Problems and Issues

Due to a problem in the simulation module the original versions of the environments in which the optimisations ran were not saved as expected, nor it was their random seed. Although for each unique optimisation cycle all the individuals were tested in the same environment (generated by the same unique seed), and therefore the results are valid, the reproducibility of the exact conditions of any particular simulation were no longer possible (as the random seed of the environment is unknown). Nor would it be possible to resume, or force resume, any of the optimisation cycles. In short: each type of environment was generated from a unique and unknown seed. Fortunately, though, after several crashes experienced in Experiment Series F, the version of the program used on Series G saved the main results, that are the inputs (genome) and outputs (costs) for all the function calls.

This issue was discovered after a first attempt to run the Cross-environment Trials (the step in which the best solutions/genomes evolved in one environment are tested in the other 8 environments) in the optimisations with sensor noise 1%. This way, a few adaptations in the simulation program had to be made ensuring that the environment instances would be saved beforehand. Also, in order to carry on with the experiments for methodological developments, as the optimisation step might take a very long time, the best 100 individuals from each environment were simulated again in environments with random seed (value 5420). After that point, this random seed was considered the original

seed for comparison purposes. Also in order to maintain the reliability of the experiments to come, on every Cross-environmental Trials the best solutions of each environment were tested in 9 environments: 1 of which would be a repetition of the environment they originally evolved in.

4.3.5.6. Native Environment Replicates

In order to assess the validity of the parameter optimisation, some extra validation experiments were performed to investigate that the optimisation of the input parameters of the foraging algorithm did not result in overfitting (when a model or analysis fits too closely a particular set of data, failing to fit or to predict future observations). In that case, these further validation experiments aimed at confirming that the tuning of the parameters fitted not only the particular instance of the environment they were optimised in (seed 5420), but also other instances of environments of the same type - same size and quality of resources, but generated with different seeds (Figure 4.19).

For this experiment, 14 four-digit seeds were randomly generated and used to generate variations of the default environments tested (Table 4.14). Two of these seeds (S8 and S10) have repeating values, as a way to validate the method for generating controlled environments.

Table 4.14 - Default Seed and Validation Seed values.

Seed	Value
Default / S0	5420
S1	0132
S2	0975
S3	0549
S4	0332
S5	1987
S6	0025
S7	0102
S8	2604
S9	1000
S10	2604
S11	1234
S12	0906
S13	1307
S14	0555

The replicate with seed S3 (0549) displayed results with a great variability from the original seed S0 (5420) and to the other 9 replicates in all the environments, except in SR. The same was observed in SA with seeds S9 and S12 (1000 and 0906). Further analysis showed, however, that the total light available during the simulation with seed 5420 was lower in all the 9 environments than most or all the simulations with other

replicate seeds. The *Light Index* for environment SA with seeds S9 and S12 (1000 and 0906) was also lower than the others (Figure 4.24).

These validation experiments also raised attention for the importance of all the environments types running the same seed, as there is a clear consistency among different environment types running the same seed (Figure 4.24).

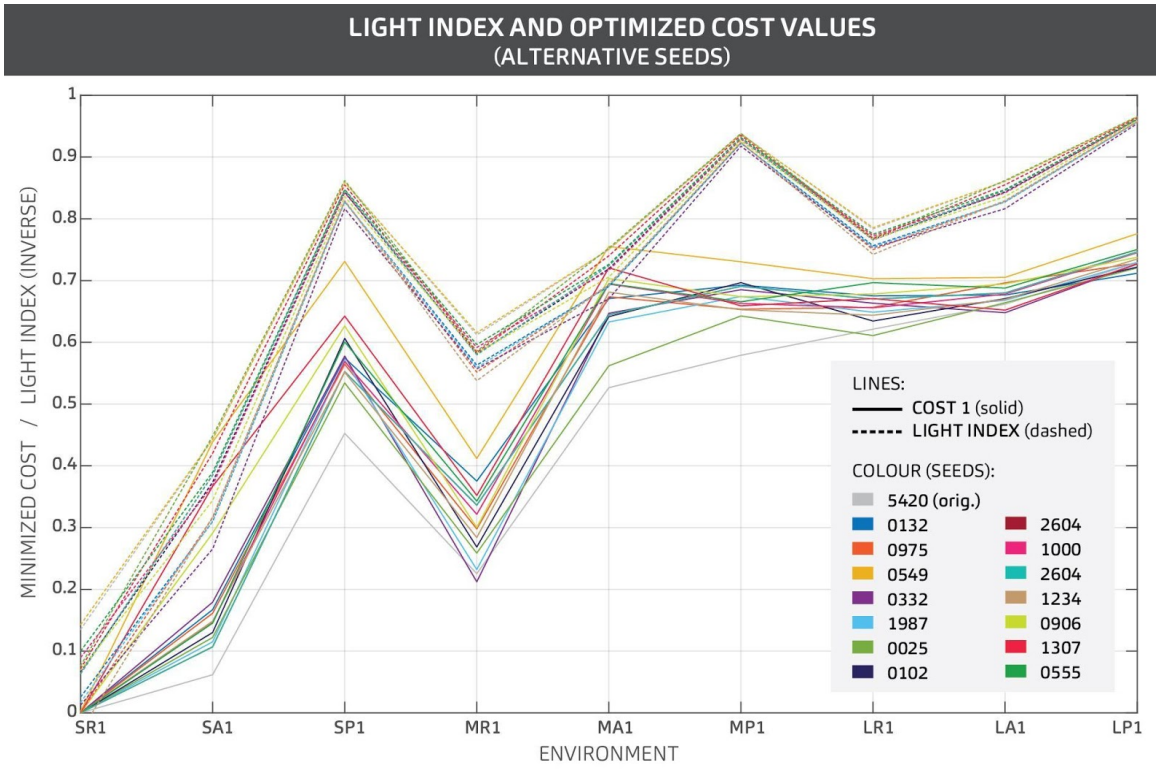


Figure 4.24 Minimised Cost1 (solid lines) and Light Index (dashed lines) of the original optimisation and all the replicates with alternative seeds.

Each colour represents the results for the experiment in an environment generated from a given seed, and grey represents the original seed (used to generate environments in the optimisation step).

4.3.5.7. Cross-Environment Trials

In this step the best 25 individuals evolved in each environment were tested outside their native environment and the value of Cost 1 could be further minimized in 6 of the 9 environments. Non-native (alien) populations achieved better results than the native populations in SA, SP, MR, MA, LR, LA (Fig. 4.25, 4.26, 4.27).

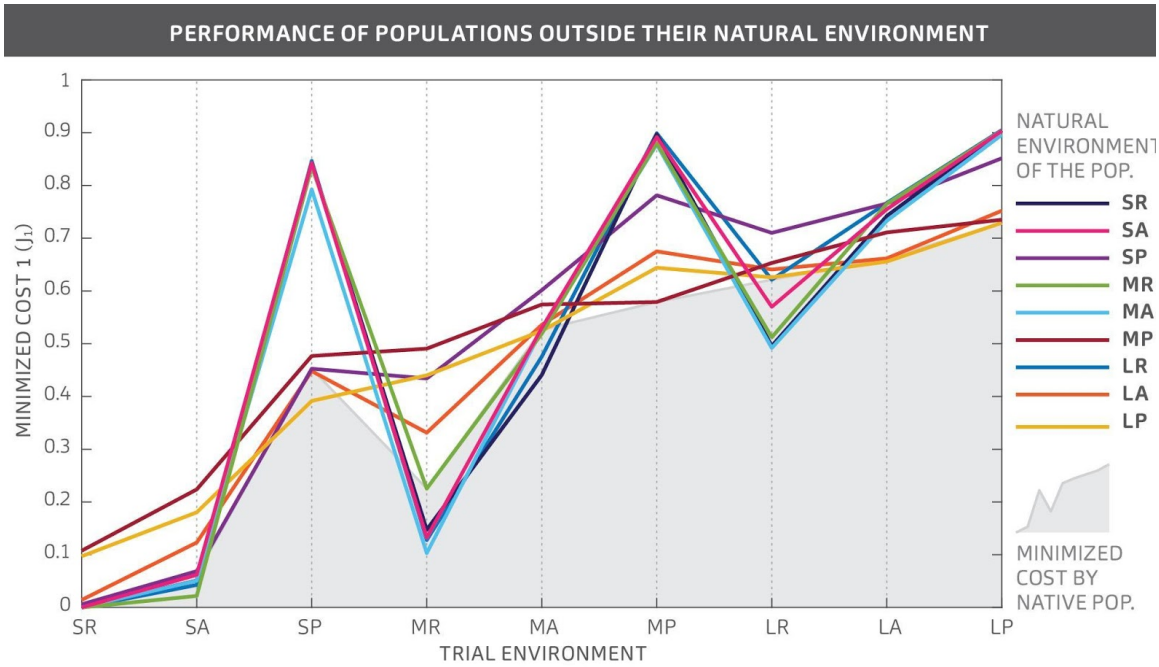


Figure 4.25 Results of the Cross-environment Trials.

Each line represents the native population of a certain environment and its performance in every one of the 9 environments (native and 8 others). The performance is plotted on the y axis (minimized cost 1), and along the (9 environments).

Populations evolved in the easiest environment, SR, outperformed native populations 4 environments (SA, MA, MR, and LR). The best individuals evolved in the second and the third easiest environments, SA and MR, also outperformed native populations in 2 and in 3 environments, respectively (Fig. 4.25, 4.26).

Populations evolved in the hardest environment, LP, outperformed the native in 2 environments (SP and LA). Individuals evolved in the second hardest environment, LA, also performed better than the native ones in SP (Fig. 4.25, 4.26).

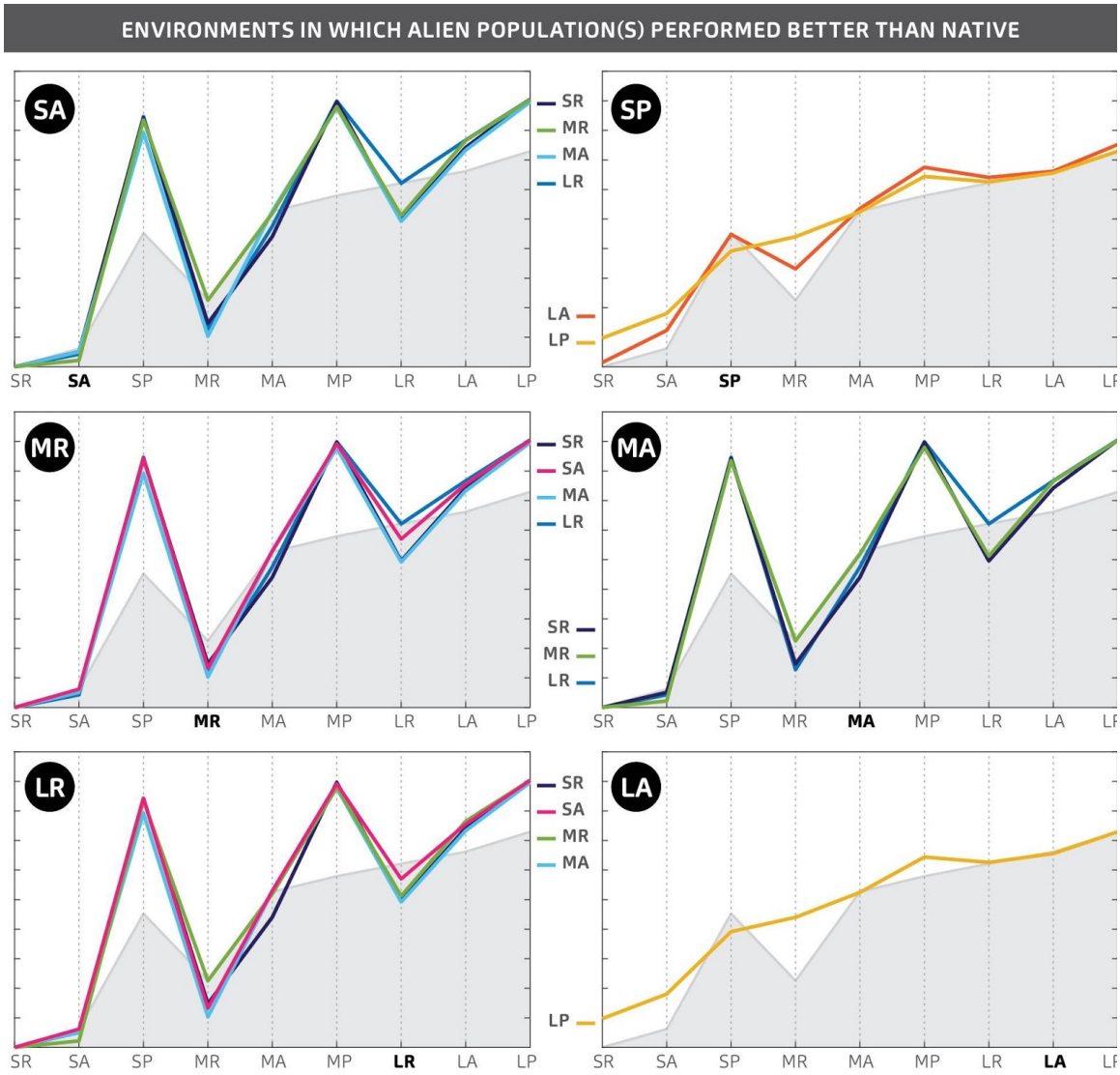


Figure 4.26 Analysis of results of Cross-environment Trials: environments in which non-native populations performed better than the native population.

The two-letter code on top left corner indicates the environment and the coloured lines represent the native environment of the populations who performed better than the native.

Overall, two distinct patterns could be found in data (Figure 4.27):

- **Genomes evolved in SR, SA, MR, MA and LR** displayed a highly variant pattern; while some the group managed to reduce the costs in the environments within the group, their performance was drastically lower (high variance) in environments with poor quality of resources (SP, MP and LP).
- **Genomes evolved in SP, MP, LA and LP** displayed a more stable pattern; lower deviance from the performance of the native populations, even though their performance in easier environments (such as SR and MR) was lower than expected.

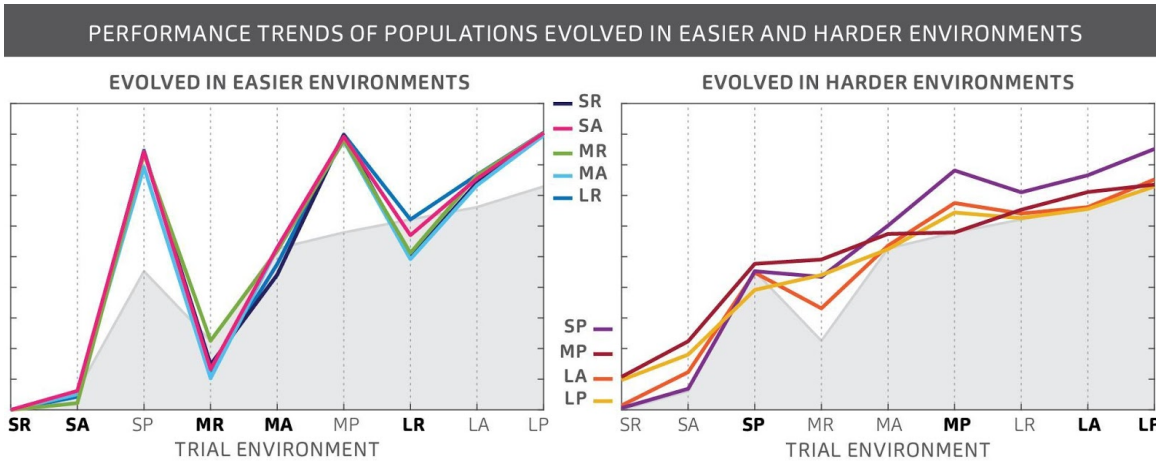


Figure 4.27 Performance pattern found in the results of Cross-environment Trials (cross-trials).

Two performance trends have been identified: solutions evolved in easier (left) and harder environments (right).

As described in section 4.3.5.5 some reproducibility problems occurred in experiment series G and although all the individuals from each environment type evolved in the same instance of this environment, its random seed is unknown. That means some of the results found in the Cross-environment Trials (particularly the ones in which non-native species performed better than the native ones) must be a result of the fact that both the native and non-native were simulated in a different instance of the original environment. As it can be seen in Figure 4.24, although the different instances of the same environment type maintain a level of consistency with each other, some costs might vary.

These results also drew attention to the settings used in the optimisation step. As previously mentioned, the convergence of solutions in harder environments (Figure 4.27) happened much earlier than the others. This way, it is possible that the combination of less forgiving environments with less forgiving optimisation settings might have led to an early convergence due to diversity loss. Some optimisation settings have been further tuned for the following experiment series (series H, I, and J).

4.3.5.8. Cross-Environment Trial Replicates

The experimental step of Cross-Environment Trials were repeated another 10 times (10 replicates), using alternative random seeds. Each replicate consists of 81 sets of experiments, in which the best populations originally evolved are tested in one variation of their native environment and in variations of the other 8 environments.

Similar to the original experiment, the results of each replicate were only compared within the same set (i.e.: with the same random seed). The overall performance variation observed in the original Cross-Environment Trials (Figure 4.25) could also be observed in the replicate experiments (Figure 4.24). The same performance pattern highlighted in

Figure 4.27 was also found in the experiments, in which two groups can be distinguished: (1) genomes evolved in SR, SA, MR, MA and LR, and (2) genomes evolved in SP, MP, LA and LP.

For the same new variations of also ran some extra validation experiments to make sure that the optimisation of the input parameters of the foraging algorithm did not result in overfitting - when a model or analysis fits too closely a particular set of data, failing to fit or to predict future observations. In this case, these further validation experiments aimed at confirming that the tuning of the parameters fitted not only the particular instance of the environment they were optimised in (seed 5420), but also other instances of environments of the same type - same size and quality of resources, but generated with different seeds (Figure 4.28).

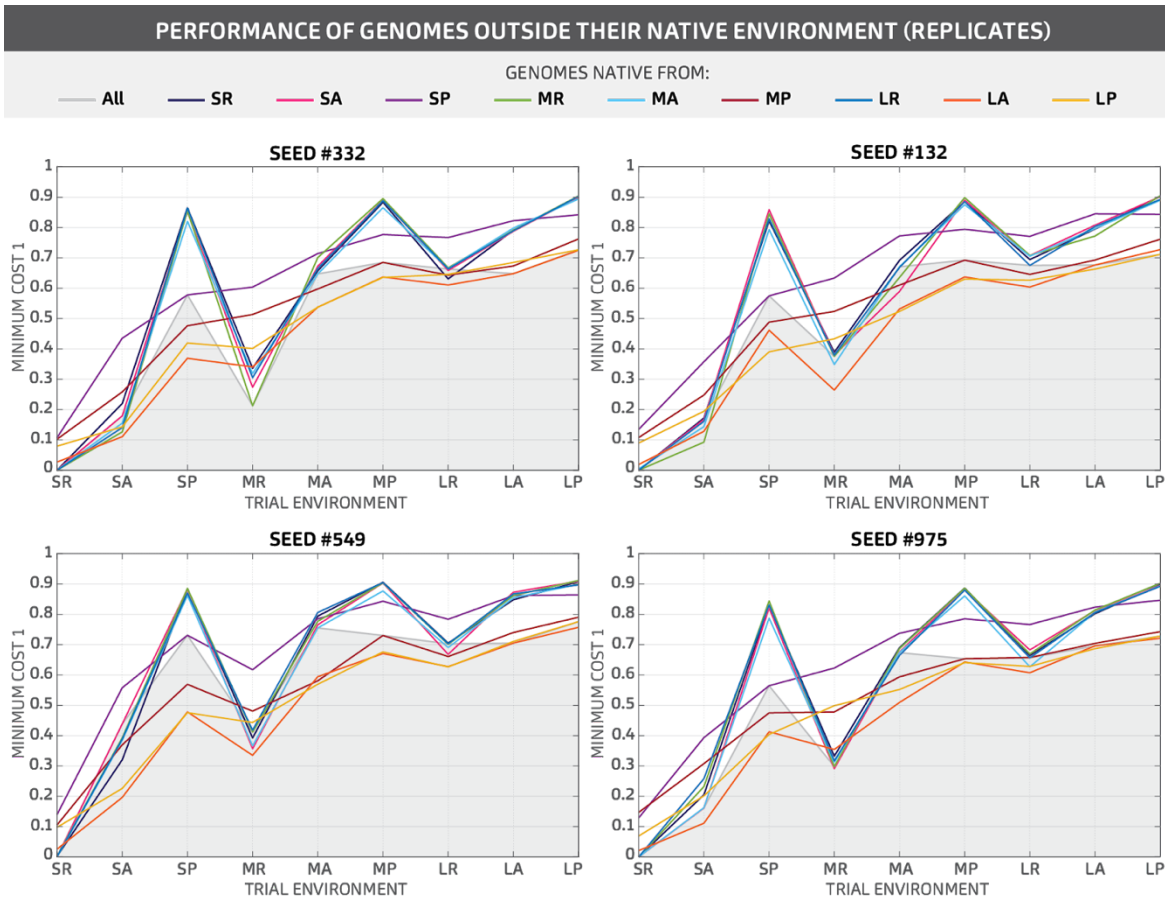


Figure 4.28 Replicate Minimised Costs (Grey Lines) and Replicate Cross-Environment Trial Results, for the best 100 individuals running the behavioural algorithm AT-5 (sensor noise 1%), tested in different instances of the 9 environments (SR, MR, LR, SA, MA, LA, SP, MP, and LP) with attractants only.

Each instance was generated using a different random seed.

Genetic Analysis (Genotype Clustering)

Finally, the clustering algorithm *k-means++*[133] was used to find patterns in the combination of the input parameters (genome). The performance of a clustering process can be assessed by analysing the accumulated sum of point-to-centroid distances with the *Elbow Method*, as presented in Figure 4.29. Five was chosen as the number of clusters (K=5) as it provided a balance between optimal performance and readability of results (Figure 4.30). In order to facilitate the present analysis, each cluster will be referred to as a *species*.

The best populations from SR, SA, MR, MA, and LR were found to be essentially composed by two species: 2 and 5. While species 2 was the majority (more than 70%) in SA, MR, and MA, the opposite happened in LR where species 5 composes approximately 75% of the population. Species 5 also composed the majority of the population in SR, but in a more balanced scenario: 58% (species 5), against 42% of species 2 (Figure 4.30).

The hardest environment, LP, happened to be the most diverse: despite the predominance of species 4 (80%), species 1, 2, and 3 are also present (10%, 4%, and 6%, respectively. The second hardest, LA, was the second most diverse, with species 1, 2, and 4. Three different species were also found in SA, however one of them (species 1) was found to be only 1% of the population (Figure 4.30).

The lowest diversity was found in SP, being all the best individuals of the species 1. The same species (1) was the majority in LA (72%) and MP (52%), sharing LA with species 3 and 4, and sharing MP with species 3 (Figure 4.30).

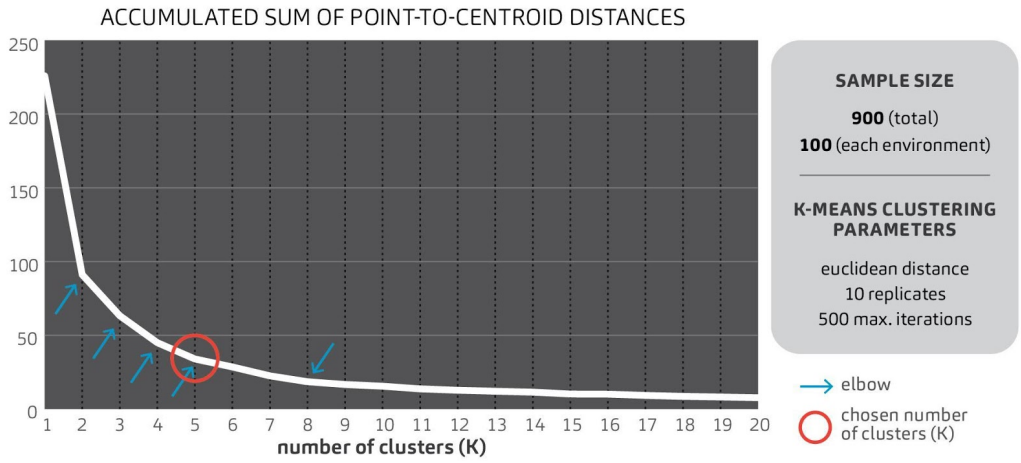


Figure 4.29 Accumulated sum of point-to-centroid distances for different numbers of clusters in a dataset (K).

Five elbows in the graphic are indicated by cyan arrows and the red circle indicates the chosen number of clusters.

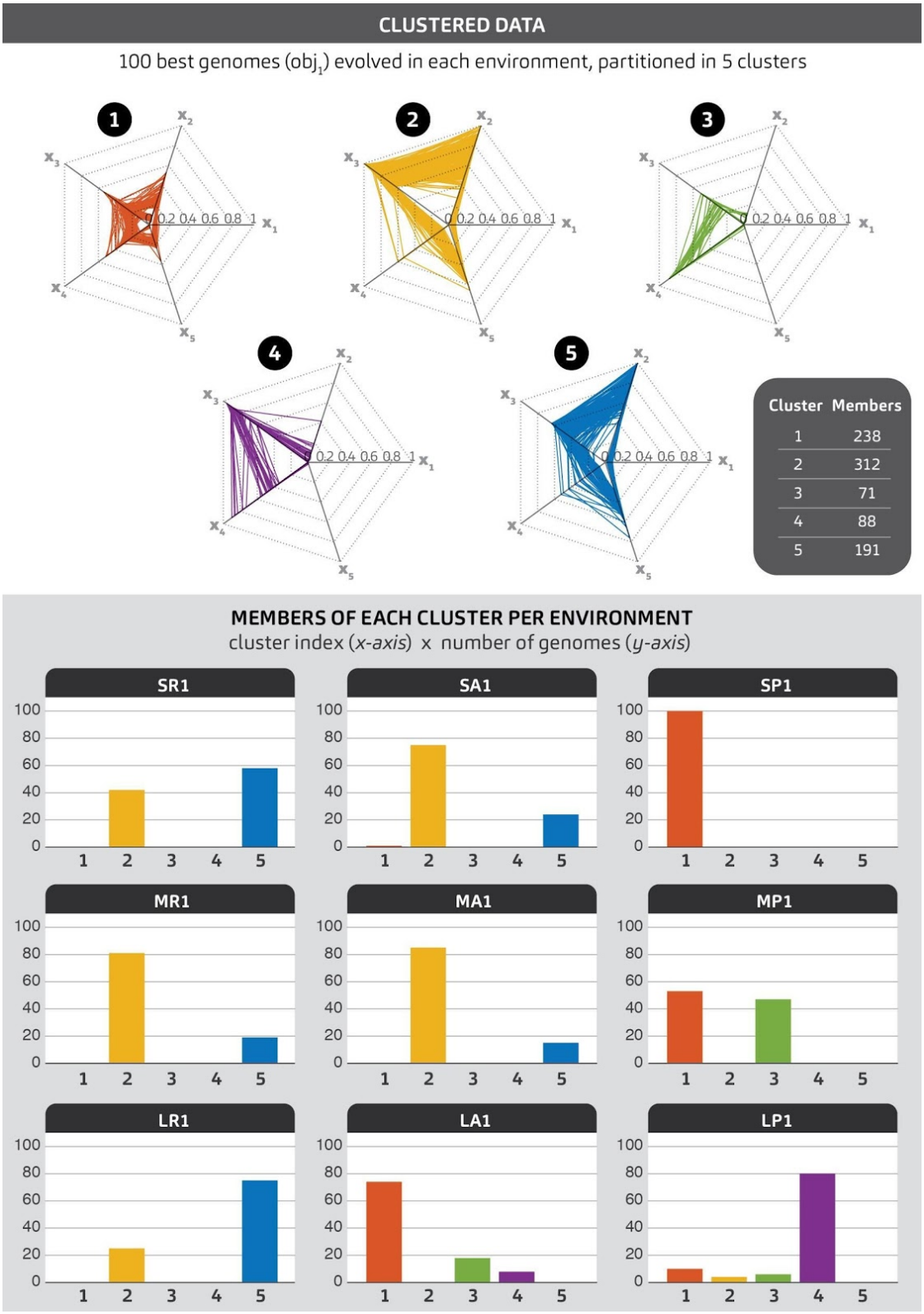


Figure 4.30 Results of Genotype Clustering.

Top: all members of the five clusters plotted into spider plots. Each radial axis of a spider plot represents the range of the value of each gene/input parameter ($g_{1..5}$). Bottom: number of members of each cluster on each environment.

4.3.5.9. Conclusions

After these experiments (series A-G) it was decided that in future series, a unique seed would be used to generate the environments in which the individuals would be simulated, and these environments would be saved beforehand. Second to this, further fine tuning would be made to the optimisation settings to prevent diversity loss, such as reducing elitism and increasing mutation rates. In future experiments, statistical tools will be applied in order to estimate the number of best individuals (best fraction of a population) to be selected for genetic analysis. Finally, further validation methods should also be developed for validating the clusters, as the graphics for the accumulated sum of point to centroid distances frequently contained more than one elbow.

Chapter 5
Experiment Design, Materials and Methods

In this chapter I will cover the consolidated experimental methods for the three refined foraging algorithms: AT-6, ATRP-8, and ATRP-7, assessed in experimental series H, I, and J (Table 4.1). Foraging Algorithm AT-6 has behavioural modules prepared to deal with attractants only, and takes a vector of 6 numbers as the input parameters to control its behaviour. Foraging Algorithms ATRP-8 and ATRP-7 have behavioural modules to deal with repellents, and take vectors of 8 and 7 parameters, respectively, as the input parameters to control their behaviour.

The behaviour of the robots (agents) is a result of the interaction between the environmental conditions and the combination of the input parameters ($g_{1..n}$) tuning the foraging algorithm, which we called the robot's genome. The algorithm controls two basic movement components - runs and turns - by regulating the speed (and hence energy cost) of locomotion and the probability of turning. Speed regulation responds to instantaneous light value, while turn probability responds to the change in light value over time. The intensity of light perceived by the robots is contaminated by normally-distributed sensor noise with variance equal to 1% of the maximum light intensity (aimed at representing the imperfection of the dispersion of chemical cues in water or air, as well as the inherent noise of real sensor hardware). The details of the foraging algorithms and behavioural modules inside it will be explained in detail in Chapters 6, 7, and 8.

All three refined foraging algorithms were simulated and evolved under similar experimental conditions in terms of environments, robot hardware, and optimisation settings. In this chapter I will describe the general aspects, materials and methods in common of these experiments. Each refined foraging algorithm and its respective experimental results will be presented in separate chapters, afterwards.

First, the algorithm parameters are optimised for each environment with Evolutionary Algorithms. Each environment has a different level of difficulty, varying in size, quality levels of light sources, and intensity of repellents (for the experiments with repellents). This step is supposed to be similar to nature, in which species evolve according to the characteristics and constraints of the environments they are in. The best optimised solutions for each Environment are, then tested in the other environments and submitted to genetic analysis. The following experiments and methods will be described:

- Creation of experimental environments;
- Optimisation of solutions for each environment with Evolutionary Algorithms;
- Selection of the best individuals (solutions) evolved in each Environment;

- Tests of solutions evolved in one environment in the other environments (Cross-Environment Trials);
- Characterization of evolved solutions (Genotype Clustering).

5.1. Design of Experiments

Optimisation: for each Environment (Figure 5.1), a population of solutions was obtained with multi-objective optimisation using Evolutionary Algorithms. Differential Evolution (DE) was chosen after showing the best results when compared to Genetic Algorithms (GA) and Particle Swarm Optimisation (PSO) (Figure 5.2).

Cross-environmental Trials: the best 100 solutions optimised for each environment were then tested in the other 8 (non-native) environments and their performance was compared to the native population (Figure 5.3).

Characterization of Evolved Patterns (Genotype Clustering): This step aimed at finding similarities and differences among the best solutions optimised for each environment. The best 100 solutions from each environment (adding up to 900 solutions) have been combined into a dataset to which the clustering algorithm *k-means++* was applied[133]. The investigation aimed at finding patterns on the tuning of the five input parameters of the model ($g_{1:5}$), also referred to as the "genome". The clusters obtained were validated and the origin of its members was analysed (Figure 5.4).

In experimental series G (AT-5), I also ran some extra validation experiments to make sure that the optimisation of the input parameters of the foraging algorithm did not result in overfitting - when a model or analysis fits too closely a particular set of data, failing to fit or to predict future observations. In this case, these further validation experiments aimed at confirming that the tuning of the parameters fitted not only the particular instance of the environment they were optimised in, but also other instances of environments of the same type (same size and quality of resources, but generated with different seeds). The above-mentioned experiments (replicas) as described as follows:

Native Environment Replicates: As in the optimisation step all of the individuals of a given environment were tested under the same conditions (light spots position, duration and intensity), another set of experiments was performed to validate the results. The best 100 solutions of each environment were tested in other 10 instances of the same environment, generated by different random seeds, and the results were compared to the original seed and to sets of 100 random solutions (Figure 5.5).

Cross-environmental Trial Replicates: Further validation experiments also compared the performance of the best solutions in native and non-native environments generated by the same alternative seed (Figure 5.6).

The results of these experiments (series G) have been presented in Chapter 6.

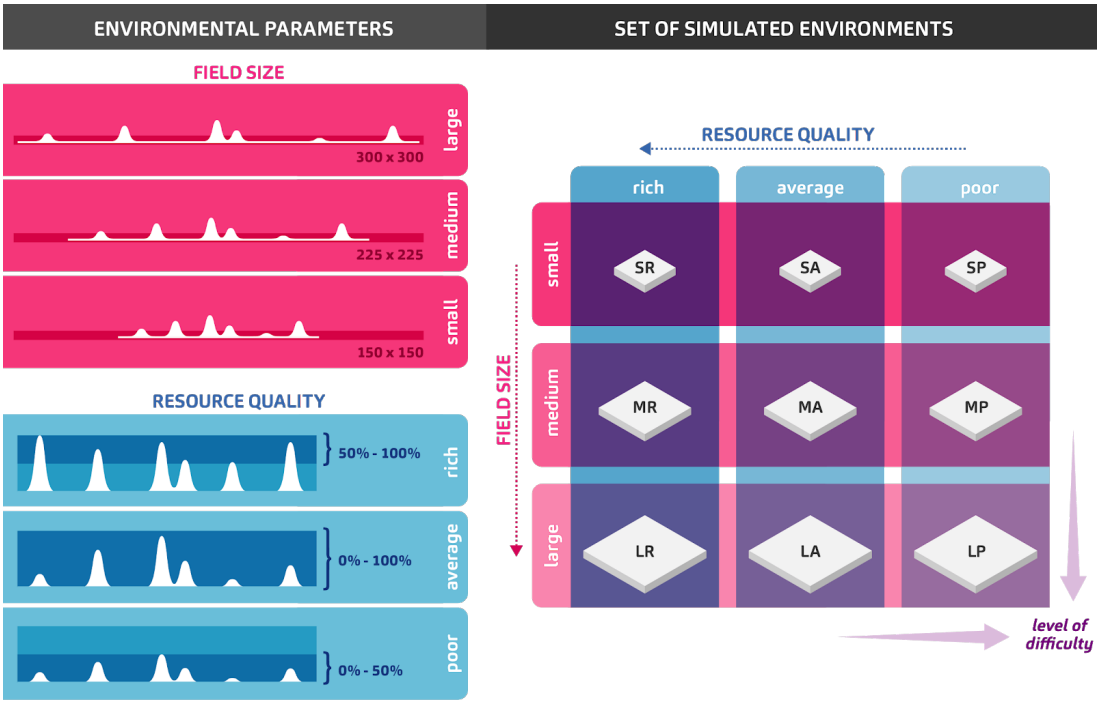


Figure 5.1 Diagram of the Main Parameters defining and Environment and the set of 9 Simulated Environments of type AT, with attractants only.

Left: environmental parameters Field Size (small, medium, and large) and Quality of Resources (poor, average, and rich). Right: Set of 9 Environments resulting from the combination (3-by-3) of the environmental parameters.

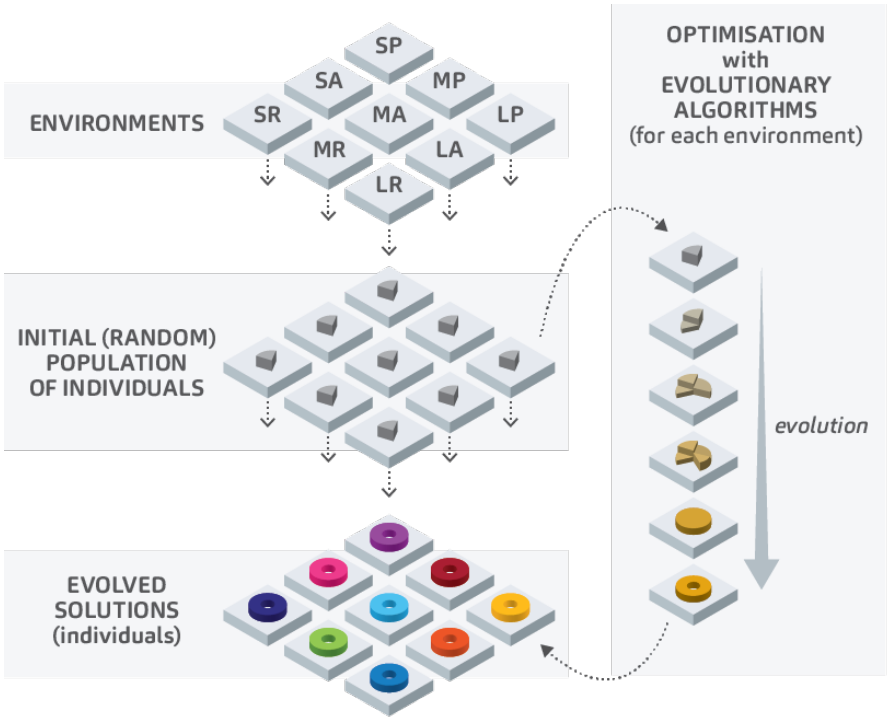


Figure 5.2 Diagram of the process of Optimization with EA, resulting in 9 different populations of solutions evolved for each environment.

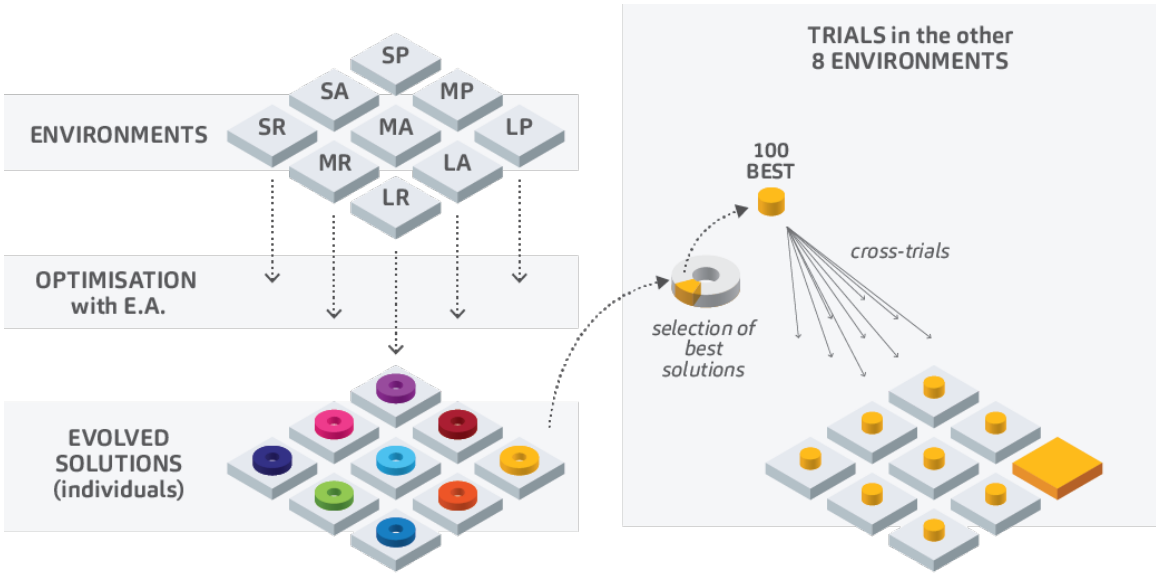


Figure 5.3 Diagram of cross-environment trials.

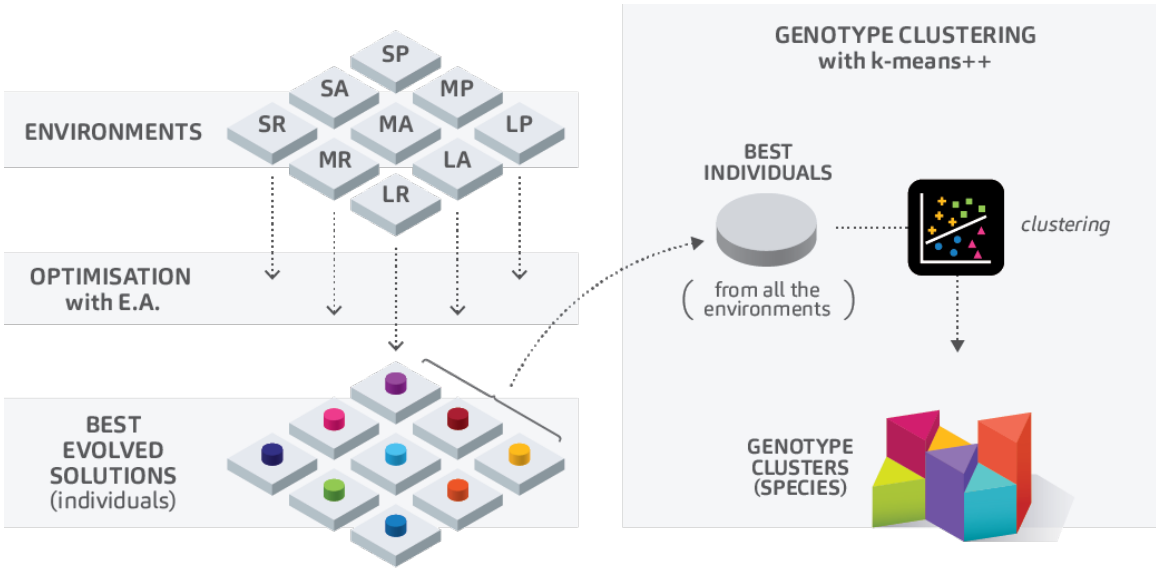


Figure 5.4 Diagram of Genotype Clustering (characterization of evolved patterns).

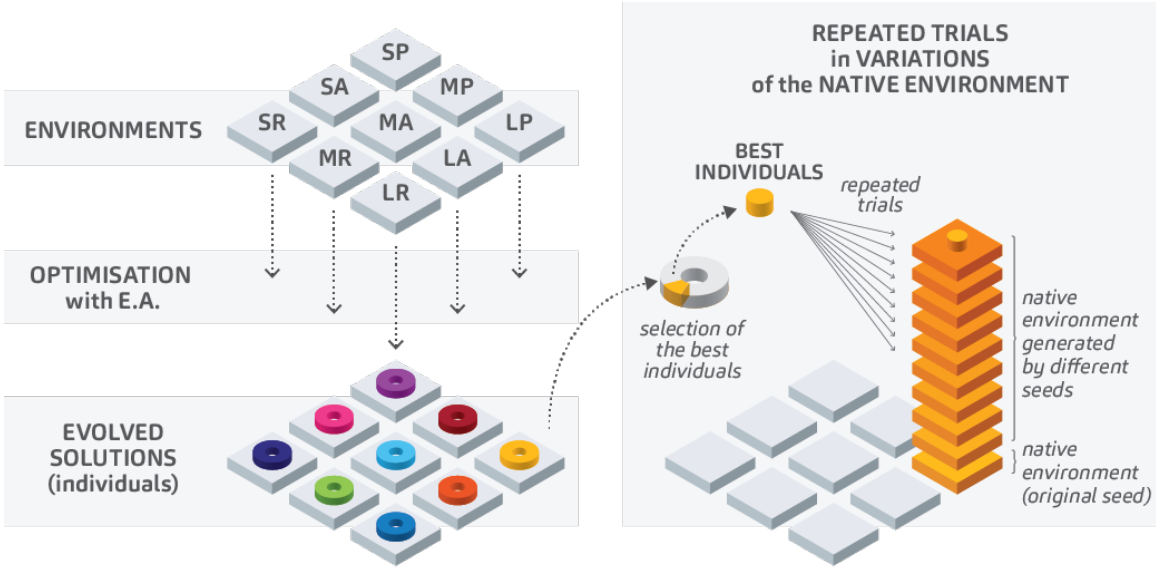


Figure 5.5 Diagram of native environment trials.

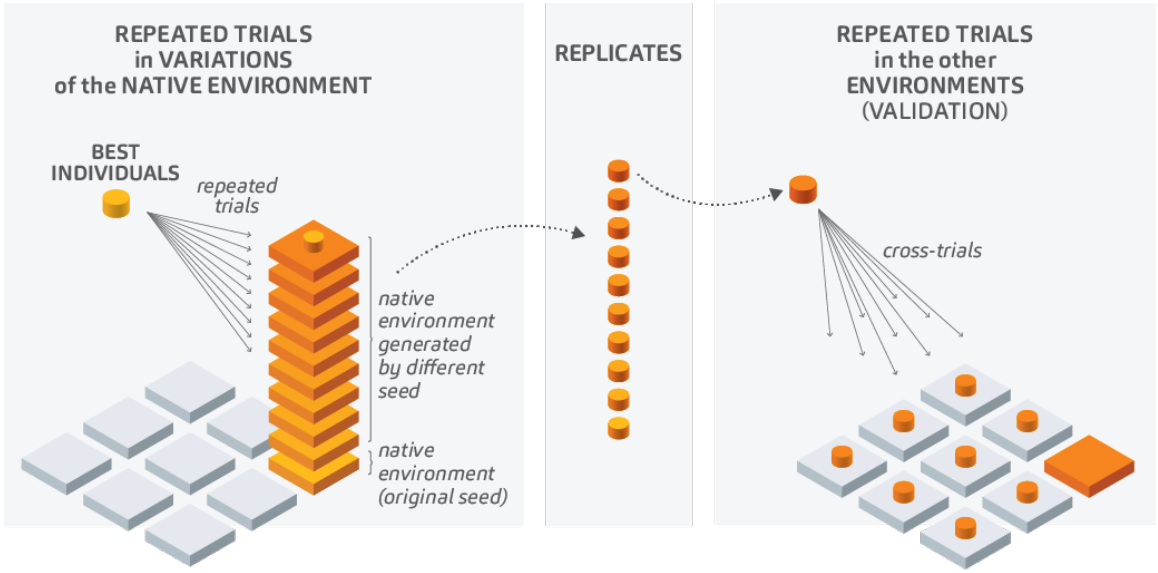


Figure 5.6 Diagram of repeated cross-environment trials.

5.1.1. Optimisation Methods and Heuristics

During the step of Optimisation with Evolutionary Algorithms, the input parameters (g_{1n}) for *C. elegans'* Foraging Algorithms are optimised - fine-tuned - for each environment, separately.

Algorithms AT-5, AT-6 and ATRP-8 (series G, H, and I) were optimised for 9 environments: SR, MR, LR, SA, MA, LA, SP, MP, and LP. Algorithm ATRP-7 was optimised for 4 variations of MA: MA51, MA11 (equivalent to MA), MA55, and MA15.

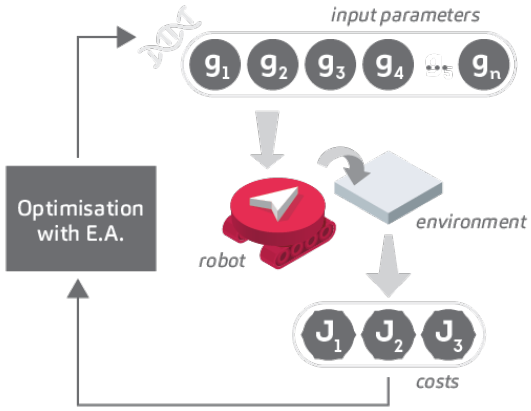


Figure 5.7 Overview of the Optimisation of C. elegans Foraging Algorithm (Simulated with the Simulation Engine) with Evolutionary Algorithms.

The Foraging Algorithm takes **n** input parameters, and the performance of the algorithm (with the given parameter setting) is measured by 3 simulation outputs (Costs).

As in most EAs, the optimisation program starts by creating a population of individuals with randomly generated input vectors and evaluating their *fitness* (performance) through one or more objective functions. Individuals are composed essentially by a vector of input parameters (to which we also refer as its *DNA*, *genes*, or *genome*) and a vector with one or more outputs of one or more evaluated functions (Figure 5.7). This first random population composes the first generation of individuals, that will give rise to the next generation, and so on. The methods to create a new generation from the previous one vary according to the algorithm, and may include *elitism* (keeping clones of the best individuals), *gene mixing/shuffling*, *tournament*, *mutation*, among others. Along the optimisation, it is expected that the overall performance of the group will improve, as well as new best solutions (best individuals) will be found (Figure 5.8).

Differential Evolution (DE) algorithm was chosen after a set of experimental optimisation cycles comparing the performance of this algorithm and two others: Genetic Algorithm (GA) and Particle Swarm Optimisation (PSO)[102,134–137]. In the context of the optimisation problem in this work, DE was able to find lower minima than the other EAs without taking much longer than GA (quickest). The number of individuals per generation was set to 200, as this number achieved better results in most of the calibration experiments when compared to 100, 400 and 800. The number of maximum function evaluations was set to 100,000, although most optimisation cycles converged before 30,000. Differential Evolution specific parameters were set as: 95% probability of a new genome to be inserted in the population and -1.5 and 1.5 as lower and upper boundaries of the scaling parameter.

As a meta-heuristic model, the performance of each parameter setting (DNA) was measured according 3 objectives:

- **Objective 1:** average of the number of time steps each robot was alive during the simulation, normalized by the total number of time steps.
- **Objective 2:** residual energy stored in the battery of all the robots that remain alive at the end of the simulation, averaged by the total number of robots.
- **Objective 3:** percentage of the field explored by each robot, averaged by the number of robots.

The first objective is the most relevant among the three, representing the capacity of robots to survive the environments. The second objective is complementary to the first and was introduced as a form of pushing evolution further in environments in which the robots were able to survive the entire simulation. The third objective was introduced as a way to penalize genomes whose original placement happened to be more favourable. The correspondent Costs ($J_{1:3}$) are obtained by subtracting the normalized objective from 1, according to equations.

Although for the interest of this study Obj_1 prevail over the others, all the three objectives were treated separately in the optimisation step. The multi-objective approach was favoured over the combination of objectives in a single fitness function aiming at improving the diversity of the solutions and keeping the resolution of the results for the further steps of the study. To do so, the method of *Non-Dominated Sorting* (NDS)[138] was applied to rank and compare solutions (genomes) on each generation.

Non-Dominated Sorting consists of organising the population of solutions into fronts ($F_{1:n}$) composed by sets of non-dominated solutions. A particular solution is said to dominate another when it performs better in all of the Objectives. Consider for instance the population of solutions in Figure 5.9: the costs J_1 and J_2 of 13 hypothetical solutions are sorted into three fronts (F_1 to F_3). Each solution was compared against the rest of the population as shown in detail (top-right corner): solution C dominates G and H (among others) given that costs J_1 and J_2 are both lower than J_1 and J_2 of G and H. The same is not true when C is compared to D: J_1 is lower, but not J_2, thus C and D are, mutually, non-dominated. The fronts ($F_{1:3}$) are composed by groups of solutions that are, among themselves, non-dominated.

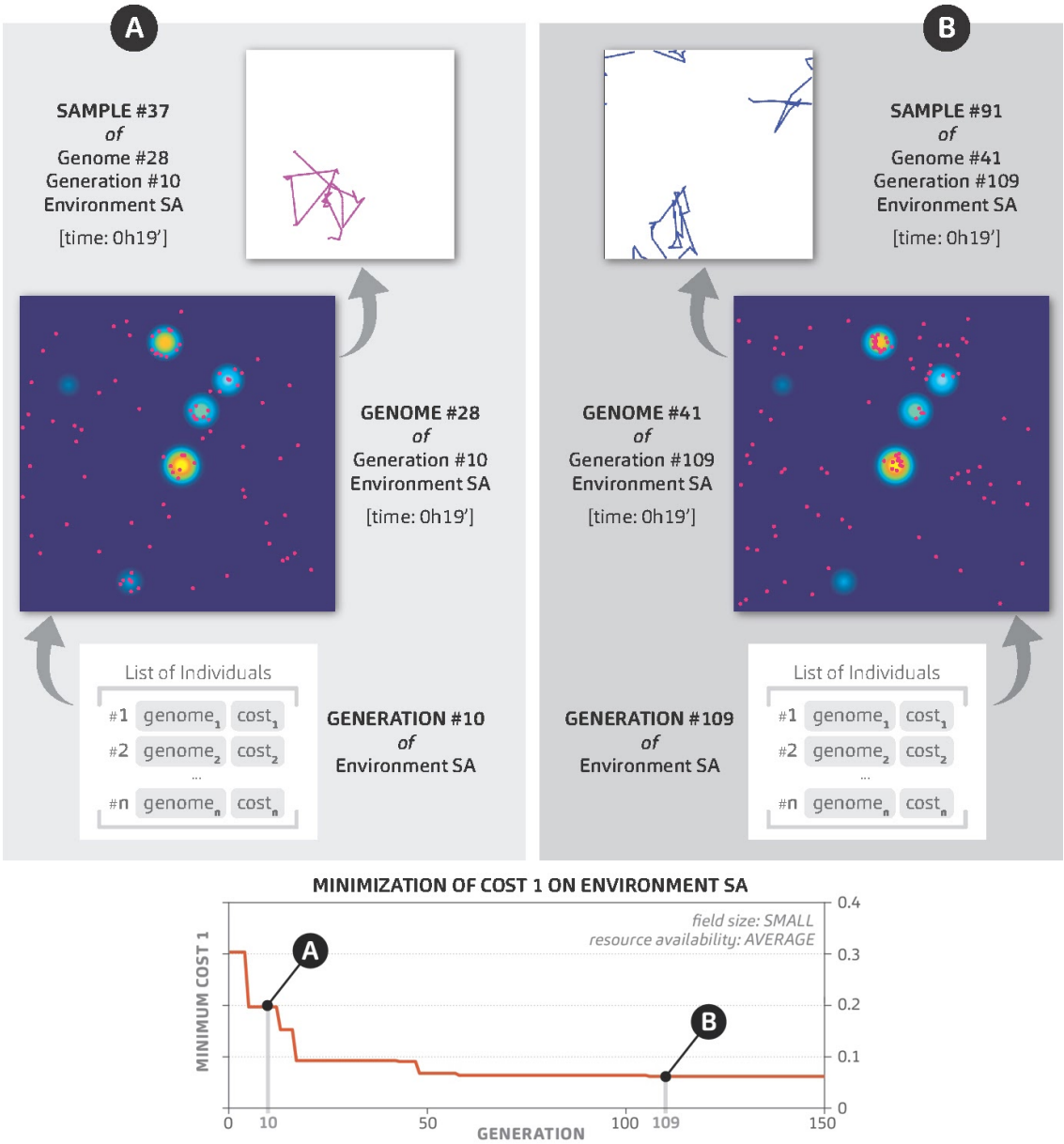


Figure 5.8 Overview of the process of Optimisation with Evolutionary algorithms, highlighting key elements and steps, as well as the correspondent terms used on the present work.

A and B are specimens from an early (A) and a late (B) stage of the same optimisation cycle (SA1). The graphic in the bottom shows the minimization of $Cost_1$ for algorithm AT5 on environment SA over 150 generations. (A) [top row]: path of sample #11 (one of the clone robots with genome #84) at 0h24' of the simulation. A [2nd row]: all the samples (pink dots) at time 0h21' of the simulation of genome #84 (10th generation) in SA1. A [3rd row]: every generation is composed of 200 individuals, each one having a genome vector (input) and a cost vector (simulation output). B [top row]: path of sample #91 (one of the clone robots with genome #41) at 0h12' of the simulation. A [2nd row]: all the samples (pink dots) at time 0h09' of the simulation of genome #84 (10th generation) in SA1. A [3rd row]: every generation is composed of 200 individuals, each one having a genome vector (input) and a cost vector (simulation output).

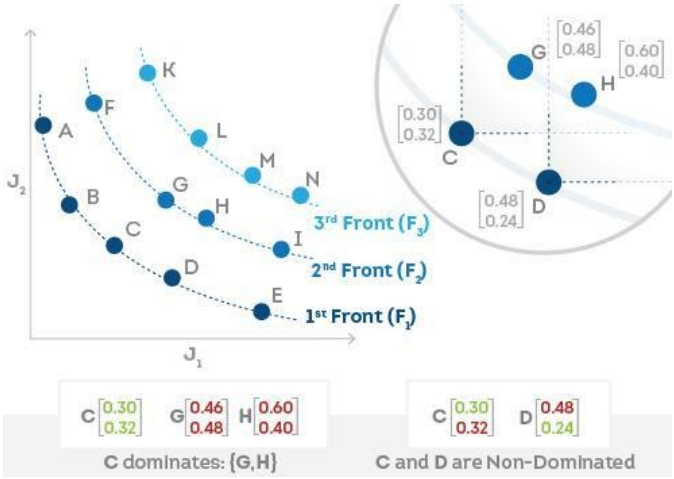


Figure 5.9 Non-Dominated Sorting: individual solutions are ranked according to their performance on all of the Objectives.

As the simulation of *C. elegans* foraging algorithm is stochastic, some measures were adopted to minimise unwanted noise on the output data. Each simulated instance of an environment is created with the same random seed, meaning that the *position*, *intensity* and *duration* of every light spot will behave the exact same way every time a different DNA is simulated on that environment. Also, the score of each DNA is calculated by averaging the scores of 100 samples of robots with the same DNA, starting the simulation in different positions (x and y) in the field.

For simplicity, experiment series I and J did not consider Cost 3 for optimisation, even though the values for every function call were registered.

5.2. Environments

The virtual environments with attractants only consist of simulated 2D fields populated with attractant gaussian light spots of varying peak intensities, which the robots must use to recharge their batteries (Figure 5.1). The position of the light spots is unknown by the robots and changes randomly throughout the simulation, so they must rely on the temporal gradient of local light intensity to tell whether they are moving in an advantageous or disadvantageous direction. The included sensor noise means that only relatively steep gradients can be reliably sensed, effectively limiting the detection radius of these attractants. Environments with attractants and repellents consist of 2D fields populated with attractant spots surrounded by smaller repellent spots, which cause damage to the robots' batteries.

Each simulation includes 50 or 100 identical robots (with the same genome) foraging light sources in a virtual environment over a certain period of time. In Series H (and all previous series), the number of samples (robots) in each simulation was set to 100. Given the

increasing in the time taken for each simulation, in environments with both attractants and repellents, the number of samples had to be reduced to 50 in experiment series I and J. Each environment is a square field with wrapping boundaries populated with 6 light spots of varying intensity and duration. For algorithm AT-6, a set of 3 field sizes and 3 light intensity ranges were tested (Figure 5.10). The environments tested contained only attractant sources. For algorithm ATRP-8, the same set of 3 field sizes and 3 light intensity ranges were tested, this time, containing attractants and repellents (Figure 5.11). For algorithm ATRP-7, due to time restrictions, 4 variations of environment MA (with attractants and repellents) were tested, being one of the variations (MA11) the equivalent of MA, from the experiments with foraging algorithm ATRP-8 (Figure 5.12).

As the simulation starts, the position of each light spot is randomly assigned, along with its intensity (within the range for that environment) and duration. The light intensity at a distance from the centre decreases according to a Gaussian curve. Each extinguished light spot is replaced by a new random instance so the number of light spots is constant during the whole simulation. For use in later analysis, the average amount of light available per square meter was calculated at each time step and averaged over the total simulation time to obtain the *Light Index* of each environment. As can be seen in Fig. 5.10, 5.11 and 5.12, the generation of the environments have been controlled by a randomly selected seed which would remain constant between environments, resulting in an overall consistent set of environments.

Some necessary changes had to be made in the simulation framework to allow the level of clustered attractants and repellents to be tuned individually. For that reason, although the position is the exact same, as well as the proportional intensity of the clusters (Fig. 5.11 and 5.12), there may have been some slight variations in the overall level of attractants available in the field. For that reason, the results obtained with MA (ATRP) and MA11 (ATRP) were not directly compared.

The total simulation time (*T=24 hours*) is divided into time steps (*ts*), in which the control algorithm is applied. Although each simulation runs with 100 identical robots, there is no interaction between them and their reasoning processes are independent of each other. Besides making behavioural patterns more visible in the graphical plots, another reason for simulating many identical robots is to improve the assessment of a candidate genome by averaging individual's scores.

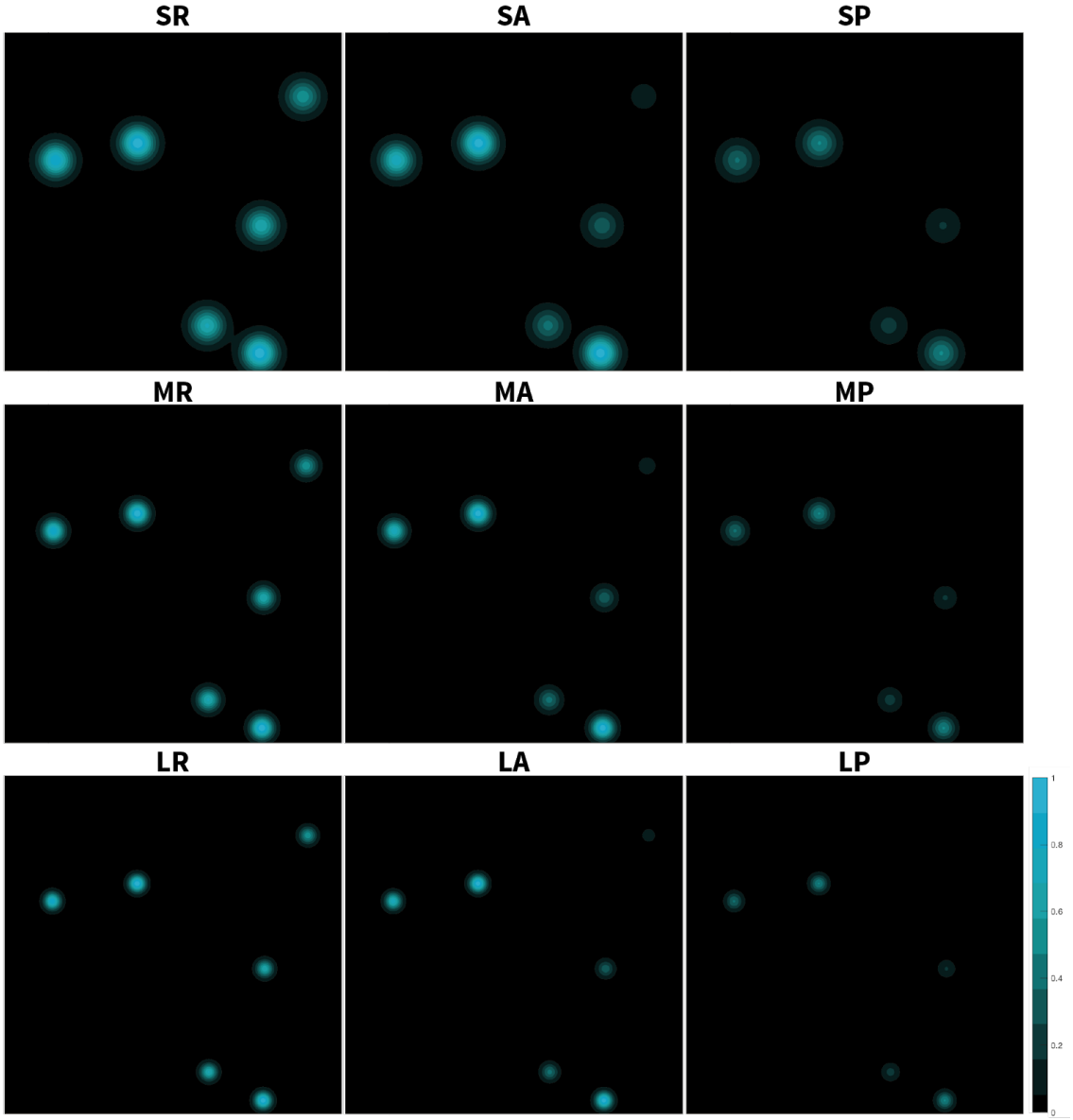


Figure 5.10 Set of Environments of type AT, simulated in Series H, with algorithm AT-6 (SR, MR, LR, SA, MA, LA, SP, MP, and LP).

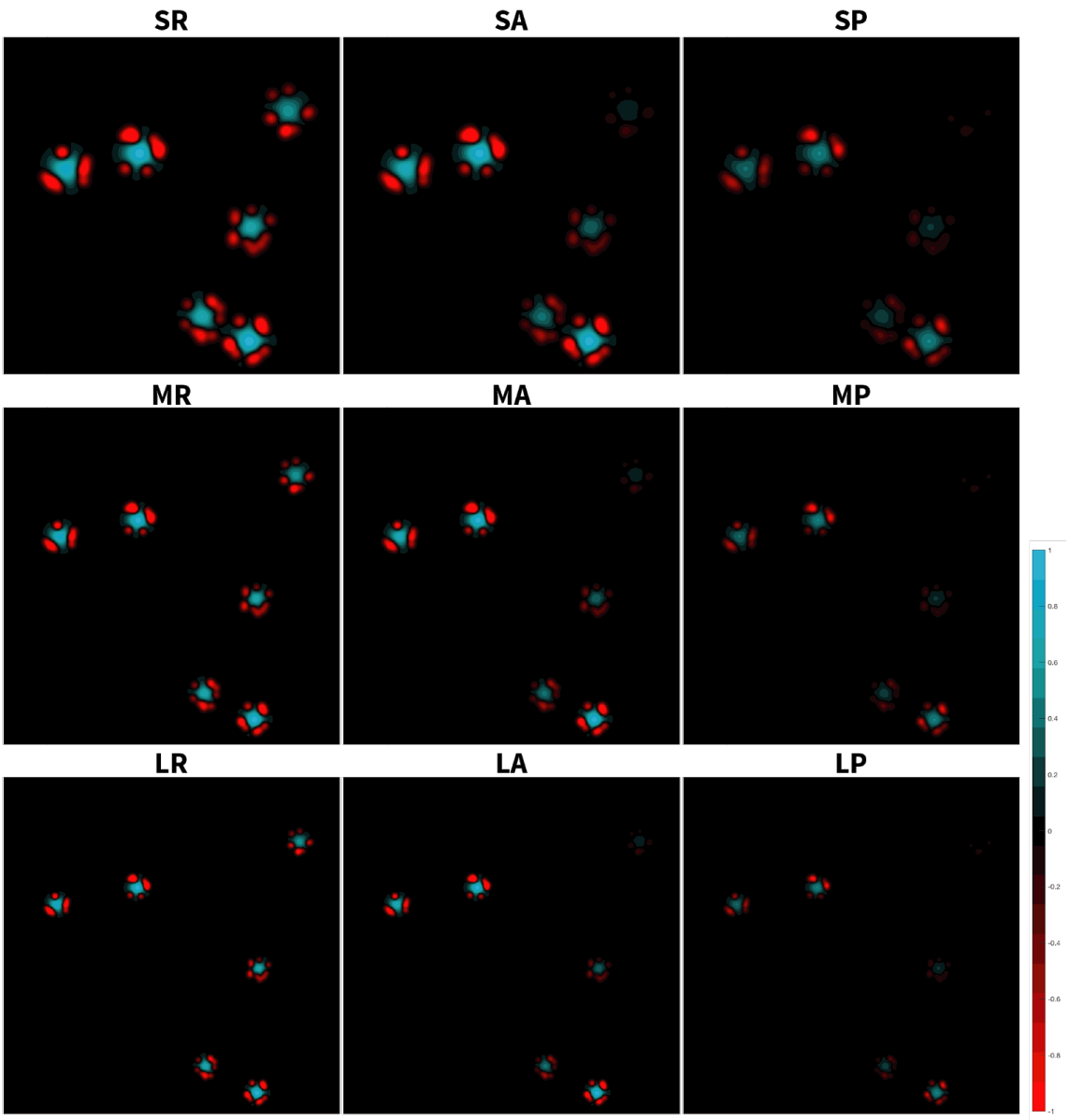


Figure 5.11 Set of Environments of type AT-RP, simulated in Series I, with algorithm ATRP-8 (SR, MR, LR, SA, MA, LA, SP, MP, and LP).

Attractants (centre) are coloured in cyan, and repellents (particles clustered around the attractants) are coloured in red.

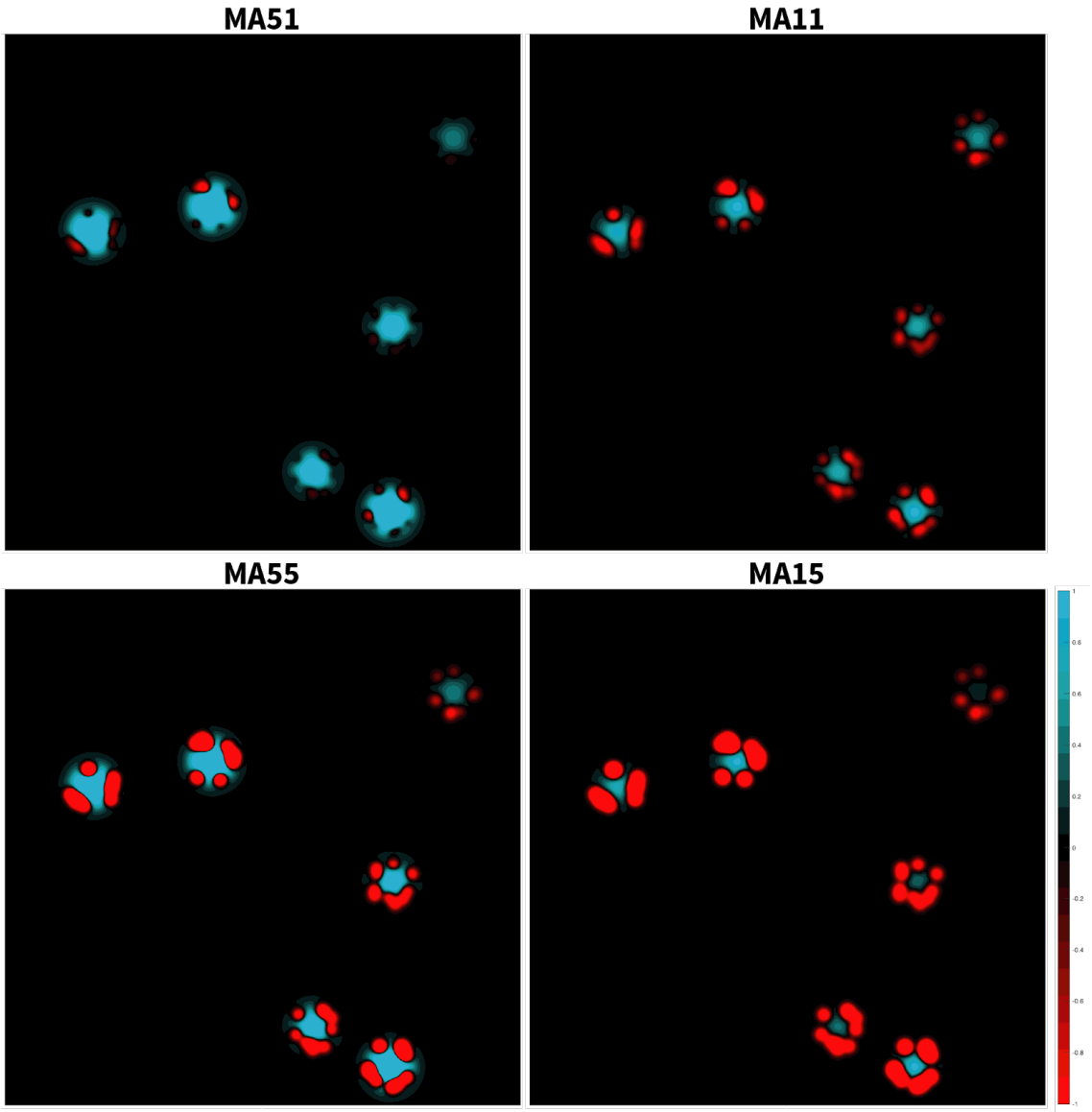


Figure 5.12 Set of Environments of type AT-RP, simulated in Series J with algorithm ATRP-7.

All environments are MA, and their subtypes (MA51, MA11, MA55, and MA15) have different ratios of the light intensity of the attractant (centre, in cyan), and the repellents (particle clusters around the centre, in red).

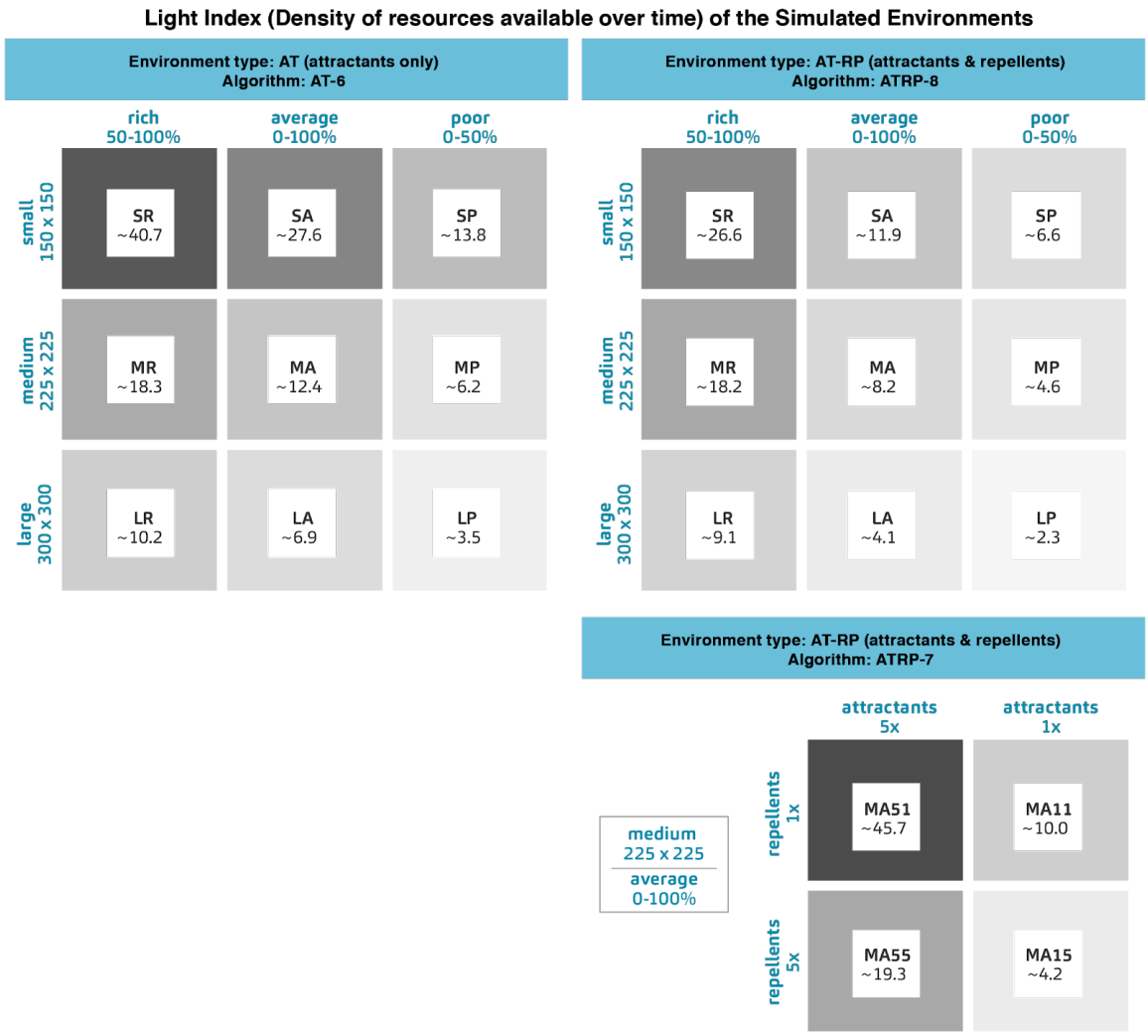


Figure 5.13 Light Index of the environments.

All environments are MA, and their subtypes (MA51, MA11, MA55, and MA15) have different ratios of the light intensity of the attractant (centre, in cyan), and the repellents (particle clusters around the centre, in red).

5.3. Robot Hardware Settings

The foraging algorithms were implemented and tested in robots simulated in virtual environments, as a requirement for the optimization with evolutionary algorithms. The virtual robots are capable of moving forward at variable speed and turning to any direction (0-360°), and are equipped with a single light sensor, a battery, and a solar panel. Aiming at future application on physical platforms, all the robot's parameters are grounded in real hardware (see table 5.2).

Table 5.2 List of robot parameters, their respective functions and standard simulation values

ROBOT PARAMETER	STANDARD VALUE	APPLICATION
$V_{\max}$	0.5m/s	Maximum locomotion velocity
$i_{\max}^{in}$	1.0A	Optimum solar current
$i_{\max}^{out}$	1.4A	Motor draw at maximum power
BMR	0.2A	Basal Metabolic Rate: cost of running systems when not moving
$Bat_{\max}$	6Ah	Battery capacity
$BatCharge$	20%	Initial battery charge
$SensNoise$	1%	Sensor Noise

5.4. Characterization of Evolved Genomes

This step explored the similarities and differences among the best solutions optimised for each environment by using clustering techniques to find patterns in the combination of the input parameters (genomes). We used the *k-means++* algorithm to cluster datasets composed by the best genomes from each environment (all grouped together). Four distance metrics were also tested: squared Euclidean distance; sum of absolute differences; one minus the cosine of the included angle between points (treated as vectors); one minus the sample correlation between points (treated as sequences of values). For each of these methods combined, we tested k values (number of clusters) from 1 to 20. Each Series of Experiments found a specific number of best individuals to be the ideal for this step and that will be explained individually, in each of the following chapters. Usually, the best fit for the clustering was found around 25 of each environment (225 genomes), at least 20 replicates, and up to 500 iterations.

When applying any clustering method the number of clusters is a key parameter, and using too many or too few will make the results uninformative. The quality of any given clustering was quantified in terms of the accumulated sum of point-to-centroid distances. When this was plotted against cluster number (Fig.4), there was no clear single "elbow" on which to base the choice of optimal cluster number. As such, the representativeness of the clusters were further quantified based on a comparison between the performance of the individual cluster members and the cluster centroid in all 9 environments (Cross-environmental Trials).

The amount of best individuals chosen from each environment is also a key parameter on this step, as the sample has to be, at the same time small enough to represent the very best individuals but still broad enough to encompass the diversity of genomes present among the best solutions. Samples too big also introduced too much noise into the clustering results. Some statistical tools have been used to estimate the amount of best individuals to be chosen for the genetic analysis step. This decision was made supported by calculations

of standard deviation from the minima, by grouping costs and parameter values in percentiles, and with the help of graphical tools as box plot diagrams. Hence, the number of both the amount of individuals for the analysis and the number of clusters were chosen based on statistical relevance and at the same time prioritising clarity in the results. Different values of samples were tested for clustering the results of all three foraging algorithms and the best results were usually obtained from samples ranging from 10 to 25 from each environment (adding up to a total of 40 to 225, depending on the number of environments).

Part IV
Results

Part IV presents the algorithm design and experiment results of the three refined foraging algorithms: AT-6, ATRP-8, and ATRP-7, tested over experimental series H, J, and I. The content is organised in four chapters:

- Chapter 6 covers the algorithm design, results and discussion of Experiment Series H, assessing the foraging algorithm AT-6 in environments with only attractants;
- Chapter 7 covers the algorithm design, results and discussion of Experiment Series I, assessing the foraging algorithm ATRP-8 in environments with attractants and repellents in proportional intensity;
- Chapter 8 covers the algorithm design, results and discussion of Experiment Series J, assessing the foraging algorithm ATRP-7 in environments with attractants and repellents in varying intensity;
- Chapter 9 presents a general discussion and comparison of all three refined algorithms, as well as the conclusion and future works.

Chapter 6
First Refined Algorithm: AT-6 (Experiment Series H)

In this chapter, I will present the design of the first refined *C. elegans*-based Foraging Algorithm, AT-6, and the results of experimental series H. The reflex-agent algorithm takes 6 input parameters and it was optimised in 9 environments (SR, MR, LR, SA, MA, LA, SP, MP, and LP), containing attractants only. The EA used was Differential Evolution (DE), with a pool size of 200 (Table 6.1).

Table 6.1 Summary of the parameters and main results of the experiments in Series H, subset H.1: Optimisation in 9 environments (SR, MR, LR, SA, MA, LA, SP, MP, and LP) with attractants only, for robots equipped with sensors with sensor noise of value 1%.

OPT.		SIM.	ROBOT	ENVIRONMENT		Number of Function Evaluations	Minim. Cost 1	Minim. Cost 2	Minim. Cost 3	Non-Dominated Solutions
EA	Pool Size	Time Step	Sensor Noise	Field Size	Quality of Resources					
DE	200	2	1%	S (150)	R	10,636	0.000	0.673	0.673	75
					A	37,093	0.018	0.869	0.869	13
					P	9,534	0.523	1.000	1.000	8
				M (225)	R	37,699	0.000	0.809	0.809	13
					A	38,256	0.426	0.986	0.986	6
					P	9,844	0.724	1.000	1.000	13
				L (300)	R	32,246	0.253	0.952	0.952	12
					A	40,797	0.671	0.998	0.998	11
					P	12,854	0.719	1.000	1.000	32
					TOTAL:	**228,959**				

6.1. Algorithm Design

The foraging algorithm inspired by *C. elegans*' chemotaxis is composed of two key behaviours: *runs* and *turns*, controlled by a set of six parameters ($g_{1\text{-}6}$), also referred to in this work as the genome, as shown in Table 6.2.

Table 6.2 - Set of input parameters (genome) for the *C. elegans*' bio-inspired minimalist algorithm AT-6.

GENE	PARAMETER	APPLICATION
g_1	*VarAngle*	Controls the variability of the angle of a turn;
g_2	*NoiseTol*	Controls the tolerance of the robot to the variation in sensed light
g_3	*BaseProb*	Sets the base probability of turning, in the absence of any change in sensed light level
g_4	*PMult*	Sets the multiplier (divider of the base turning probability when light decreases (increases)

g_5	Sig_α	Sets the steepness of the sigmoid curve that controls speed according to the sensor reading
g_6	Sig_β	Defines the offset of the sigmoid curve that controls speed according to the sensor reading

At each time step (t), each Robot (R) adjusts its speed and turn probability according to its sensor reading and possibly makes a turn. The agent's battery level is updated according to the intensity of light it is currently exposed to (i^{in}), as well as how much was spent on moving (i^{out}) and running basic systems (i_{BMR}):

$$Bat_t^R = Bat_{t-1}^R + \left(Light_t^{[x,y]} \times i_{\max}^{in} \right) - Speed_t^R \times i_{\max}^{out} - i_{BMR} \quad [6.1]$$

Also at each time step, the simulation program updates robots' positions, checks for extinguished light spots (replacing them with new ones if necessary) and checks which robots are 'alive' – a robot permanently 'dies' if its battery is depleted (Figure 6.1).

The reasoning process on each time step (t) starts when the agent acquires the sensor reading ($SensVal_t^R$) for light intensity ($Light$) at its current position [x,y], which includes some sensor noise [6.2]. $\Delta SensVal_t^R$ is then obtained from the current and previous sensor readings (3) in order to calculate the probability of initiating a turn ($PTurn_t^R$).

$$SensVal_t^R = Light_t^{[x,y]} + RandNum \times SensNoise \quad [6.2]$$

$$\Delta SensVal_t^R = SensVal_t^R - SensVal_{t-1}^R \quad [6.3]$$

If $\Delta SensVal_t^R$ is sufficiently positive, $PTurn_t^R$ decreases, whereas if it is sufficiently negative, $PTurn_t^R$ increases. If the current and previous values are sufficiently similar (as determined by $NoiseTol$), then $PTurn_t^R$ maintains the value of $BaseProb$ (Figure 6.1).

Once $PTurn_t^R$ is set, a random number (0 to 1) is generated and, if it is less than or equal to $PTurn_t^R$, the robot will perform a turn (Figure 8.1). When performing a turn, the yaw ($\Delta\theta$) will be calculated using another uniform random number between -1 and 1 ($RandNum$), according to:

$$\Delta\theta_t^R = 180° + (VarAngle \times RandNum) \quad [6.4]$$

Also, speed is modulated by an inverse logistic function, controlled by the combination of the current sensor reading and the input parameters Sig_α and Sig_β, according to:

$$Speed_t^R = 1 - \frac{1}{1+\left(e^{Sig_\alpha \times \left(SensVal_t^R - Sig_\beta\right)}\right)} \times V_{\{max\}} \quad [6.5]$$

As the behaviour of the agent is modulated without reference to any internal state variables, this is classed as a reflex-agent model.

Originally some early experiments were conducted with noiseless sensors and found that the agents were able to survive the entire simulation time in all 9 environments, making for rather uninteresting results. It was therefore decided to include substantial sensor noise

in all further experiments, which is also more realistic since all real sensors (particularly low-cost sensors) are subject to noise.

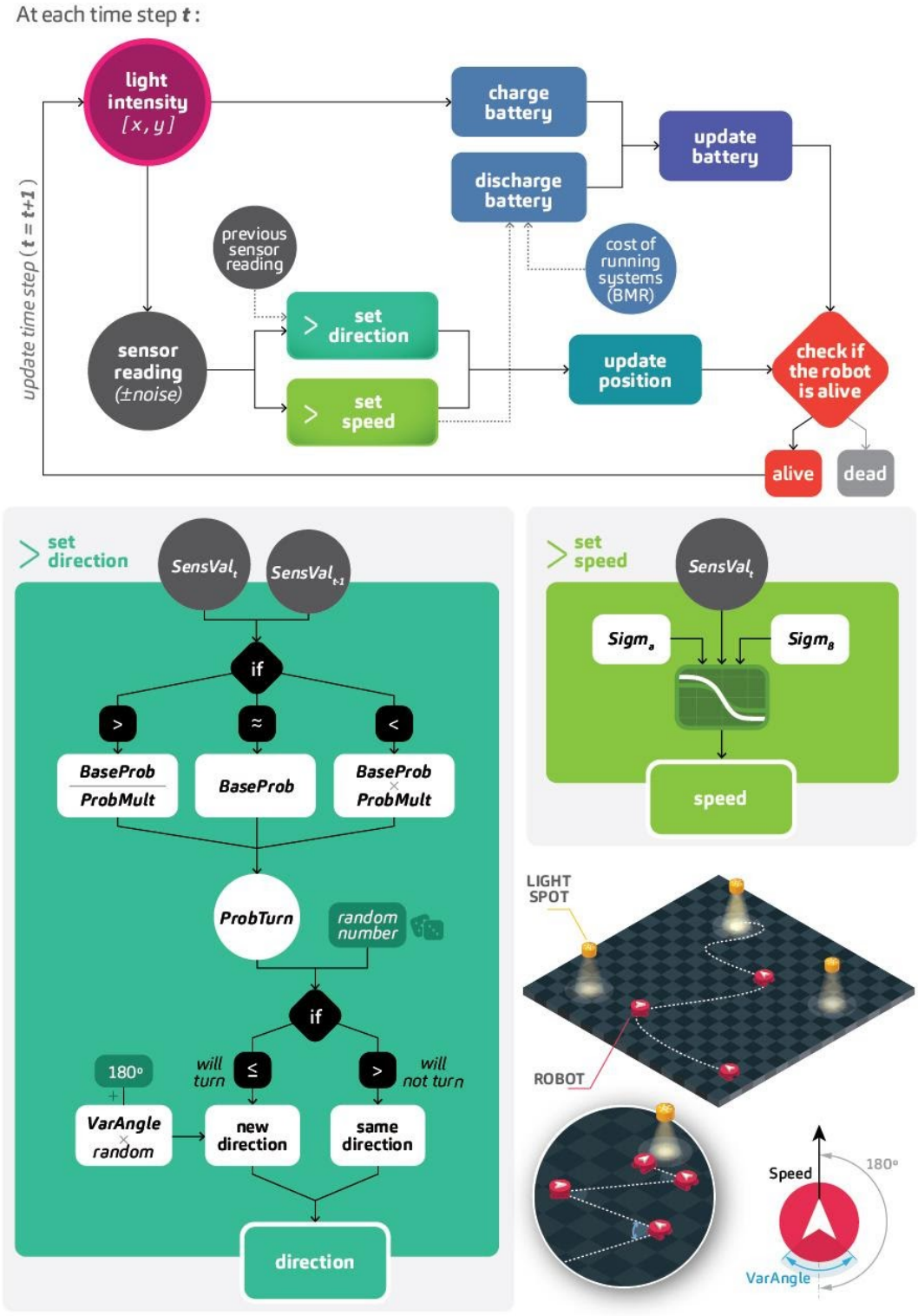


Figure 6.1 Flowchart of the control algorithm applied at each time step of a simulation.

Top: an overview of the simulation process. Bottom: detailed flowchart of processes of setting direction (left) and setting speed (mid-right). Bottom-right: conceptual illustration of the simulation and of the robot.

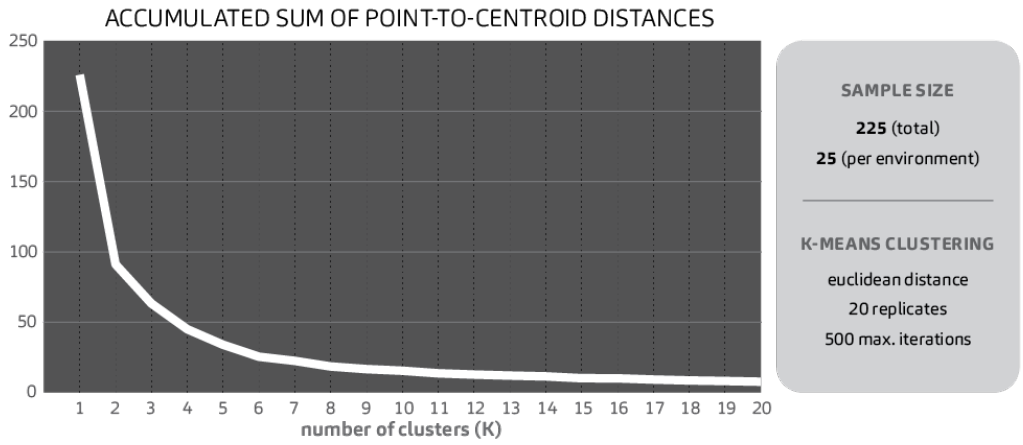


Figure 6.4 The accumulated sum of Point-to-centroid distances for different cluster numbers.

6.2. Experiment Design

The 6 input parameters of the algorithm were optimised in 9 environments (SR, MR, LR, SA, MA, LA, SP, MP, and LP), containing attractants only. The algorithm was implemented in virtual robots with hardware settings as indicated in Table 6.3. Differently from the previous experiment series (A-G), in series H and the subsequent series, the robots started the simulation with 20% battery charge. The EA used was Differential Evolution (DE), with a pool size of 200 and the optimisations ran until one of the stopping criteria was reached:

- 10^5 function calls;
- diversity loss - if the individuals in a certain population evolved to be too similar to each other;
- after 10 generations the cost did not reduce in more than 10^{-4}.

Each simulation ran with 100 robots, meaning that each set of input parameters was tested in 100 samples.

Table 6.3 List of robot parameters for simulations of series H, their respective functions and standard simulation values

ROBOT PARAMETER	STANDARD VALUE	APPLICATION
$V_{\max}$	0.5m/s	Maximum locomotion velocity
$i_{\max}^{in}$	1.0A	Optimum solar current
$i_{\max}^{out}$	1.4A	Motor draw at maximum power
BMR	0.2A	Basal Metabolic Rate: cost of running systems when not moving
$Bat_{\max}$	6Ah	Battery capacity
$BatCharge$	20%	Initial battery charge
$SensNoise$	1%	Sensor Noise

6.3. Results and Discussion

In this set of optimisation cycles for AT-6, virtual robots running this foraging algorithm were evolved in 9 environments containing only attractants. The best solutions found were grouped into 6 clusters by genetic similarity. The clusters were validated by comparing the results of its members on the cross-trials.

6.3.1. Optimisation

In this step, the algorithm's input parameters ($g_{1:6}$) were optimised in 9 different environments. To keep the results consistent, all the simulated instances of a certain environment were exactly the same. All the optimisation cycles converged between 100 and 200 generations before the maximum number of function calls were reached.

The minimised values of Cost 1 were found in SR, MR, SA, LR, MA, SP, LA, MP, and LP, respectively (Table 6.1, Figure 6.5).

Optimisations MP, SP, SR, and LP reached the stopping criteria of diversity loss earlier than the others: between 54 and 69 generations, whereas the others ran until the 250th generation.

When it comes to Cost 2, the environments in which the optimisation achieved the minimised cost were, respectively: SR, MR, SA, LR, MA, LA, SP, MP, and LP (Figure 6.6).

For *Cost 3*, the environments in which the optimisation achieved the minimised cost were, respectively: SR, MR, SA, LR, MA, LA, SP, MP, and LP (Figure 6.7).

There is a strong correlation (-0.83) between the optimised *Cost 1* values and *Light Index* for each environment. However, there is some discrepancy in the order (easiest to hardest) according to the two metrics. Specifically, the ranking of environments in terms of the optimised values of *Cost 1* (low to high) is: SR, MR, SA, LR, MA, SP, LA, MP, LP, while

the ordering with respect to *Light Index* (total light available in the field, averaged by time) is: SR, SA, MR, SP, MA, LR, LA, MP, LP (Figure 6.8). In general, rich environments perform better in practice than their *Light Index* would predict. This is a logical outcome because there are no "traps" (light spots whose peak intensity is too low to provide a net positive battery current over and above the current consumption).

Furthermore, it must be acknowledged that the optimisation process is stochastic so some solutions may be local optima (we are investigating this with more experiments). More generally, the *Light Index* may not capture all relevant aspects of the spatio-temporal light distribution (we are investigating alternative metrics).

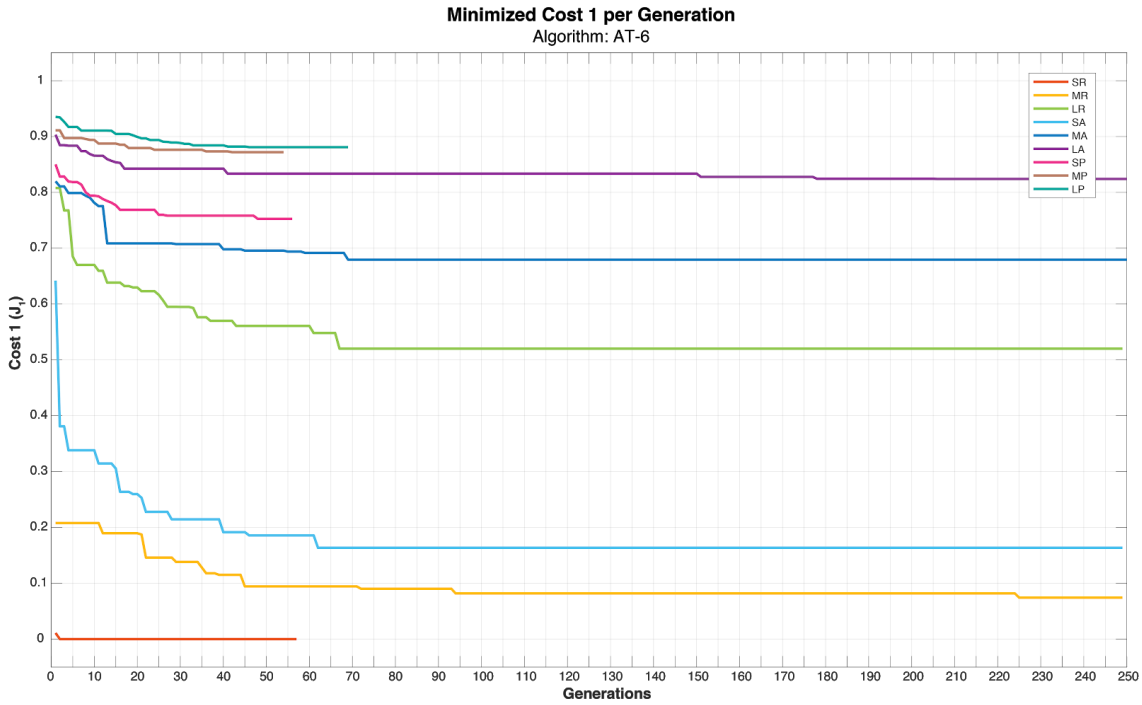


Figure 6.5 Cost 1 (J_1) optimised for Behavioural Algorithm AT-6, in 9 environments (SR, MR, LR, SA, MA, LA, SP, MP, and LP), with attractants only.

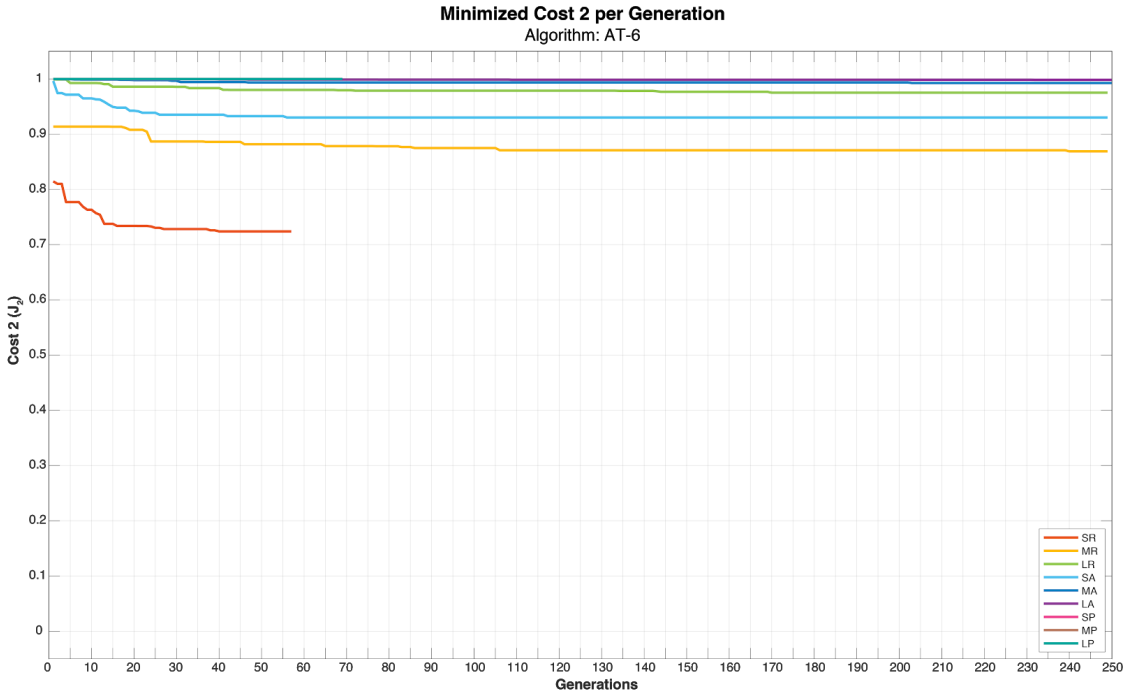


Figure 6.6 Cost 2 (J_2) optimised for Behavioural Algorithm AT-6, in 9 environments (SR, MR, LR, SA, MA, LA, SP, MP, and LP), with attractants only.

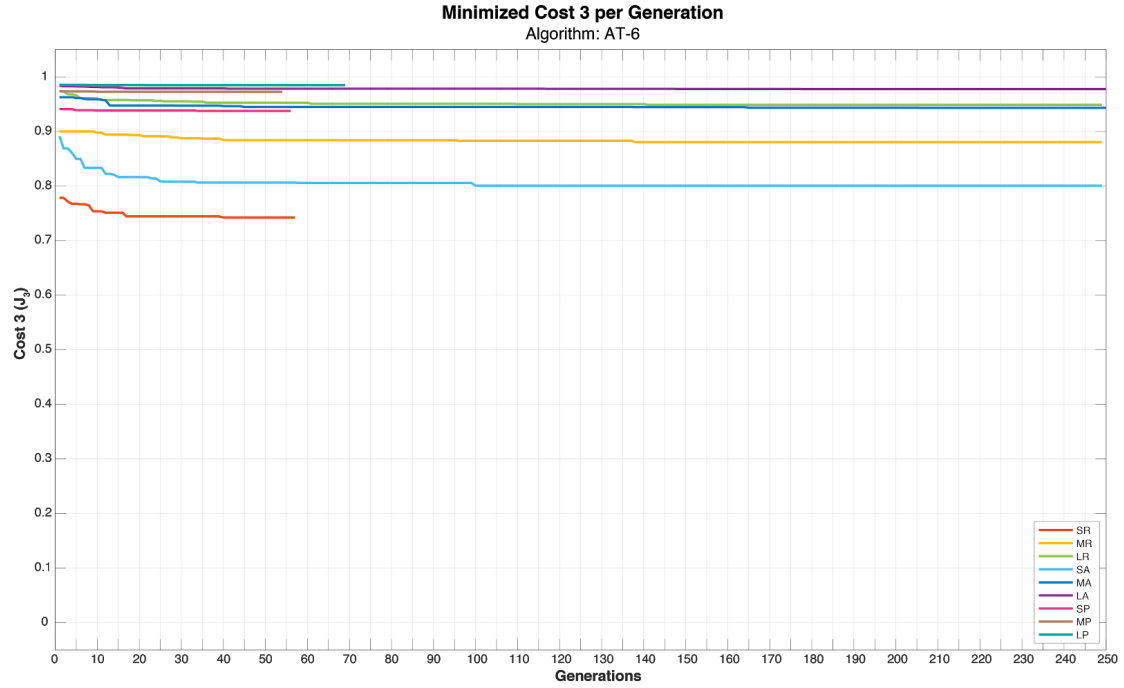


Figure 6.7 Cost 3 (J_3) optimised for Behavioural Algorithm AT-6, in 9 environments (SR, MR, LR, SA, MA, LA, SP, MP, and LP), with attractants only.

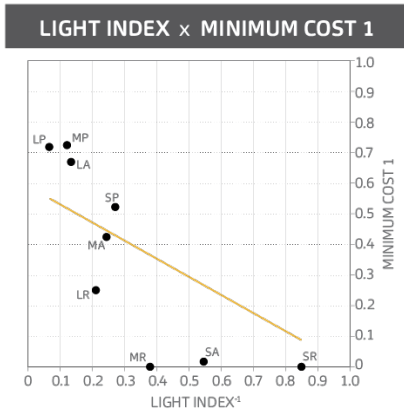


Figure 6.8. Light Index versus the Minimised value of Cost 1 in all Environments. Pearson Correlation Coefficient: -0.826; Spearman Correlation Coefficient: -0.9

6.3.2. Selection of the best individuals

Figure 6.9 shows the best 9500 unique individuals evolved in the 9 environments using Foraging Algorithm AT-6. As can be seen, Cost 1 presents a steep drop in approximately the best 100 individuals (Figure 6.9).

Further experiments explored the variance of Cost 1 in different aliquots of the population and values around 25 individuals presented a good combination between diversity (gene variance) whilst still retaining a narrow range of Cost 1 values for all environments (Fig. 6.10, 6.11, 6.12, and 6.13).

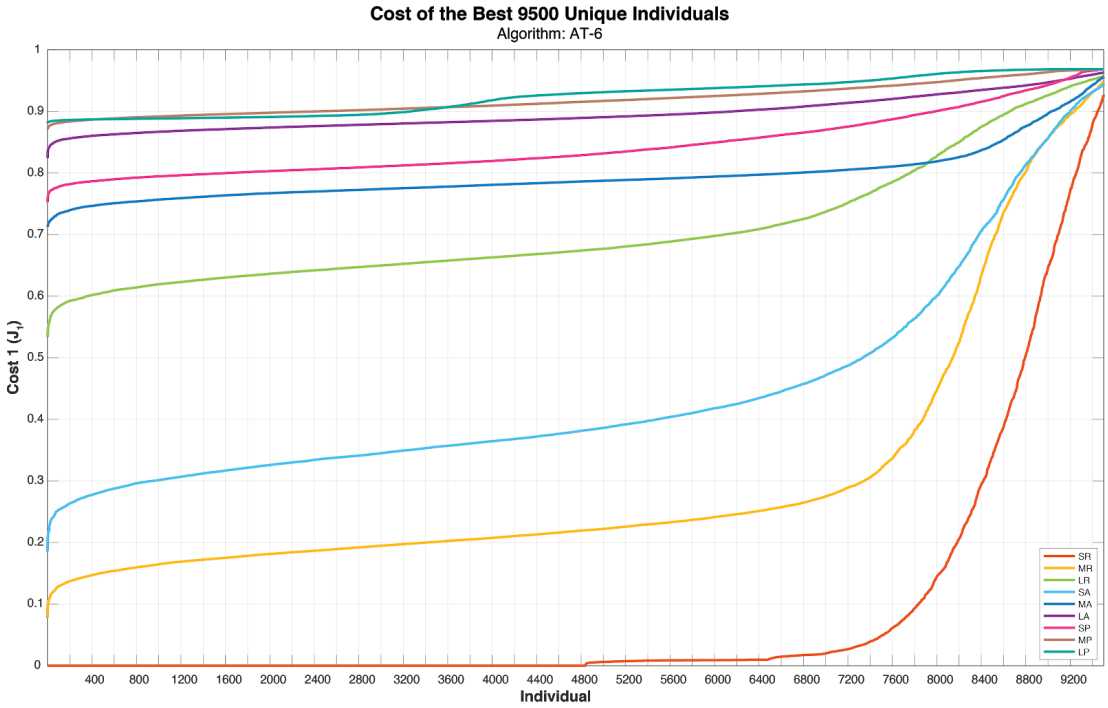


Figure 6.9 Cost 1 (J_1) of the best 9500 individuals running the behavioural algorithm AT-6, evolved in 9 environments (SR, MR, LR, SA, MA, LA, SP, MP, and LP) with attractants only.

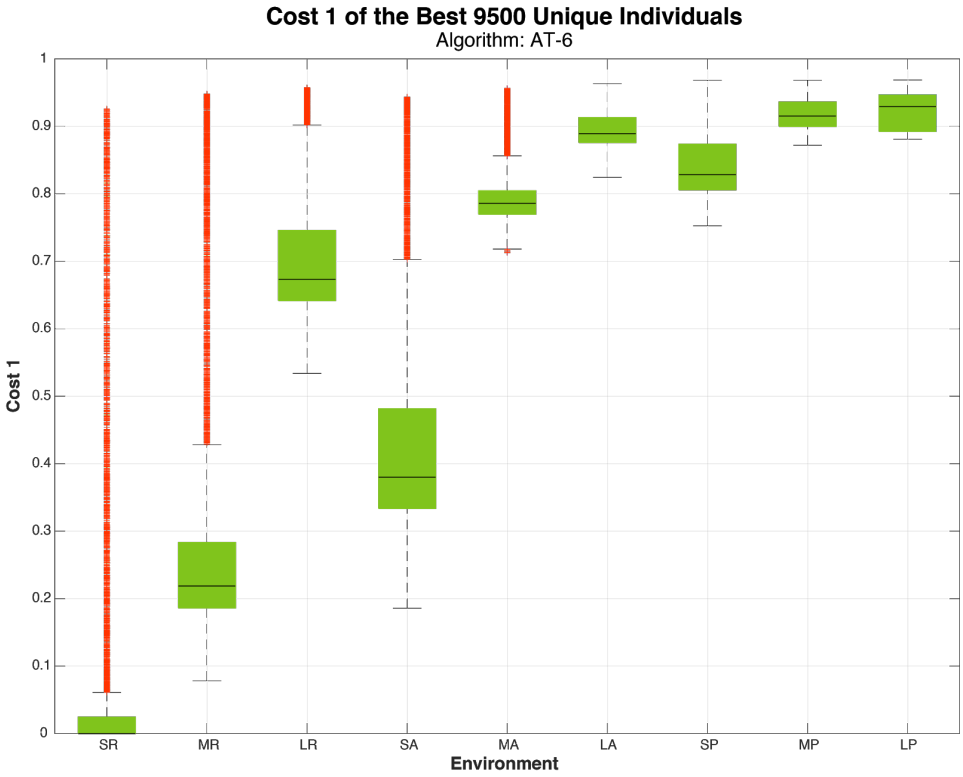


Figure 6.10 Box plot of the Cost 1 (J_1) of the best 9500 individuals evolved running the behavioural algorithm AT-6 in 9 environments (SR, MR, LR, SA, MA, LA, SP, MP, and LP) with attractants only.

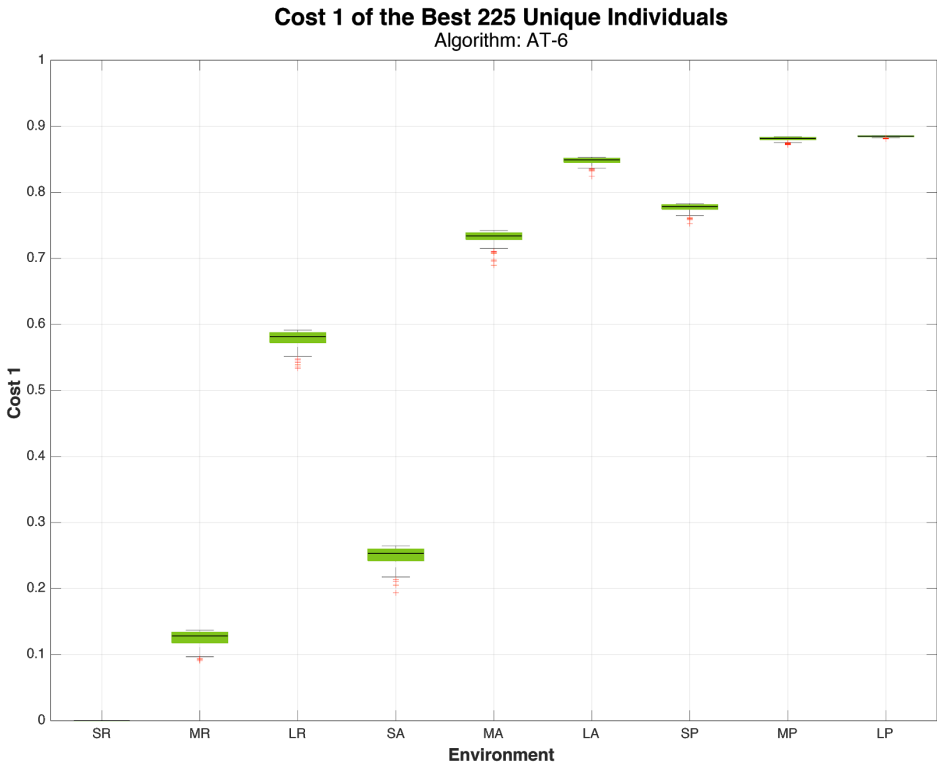


Figure 6.11 Box plot of the Cost 1 (J_1) of the best 225 individuals evolved running the behavioural algorithm AT-6 in 9 environments (SR, MR, LR, SA, MA, LA, SP, MP, and LP) with attractants only.

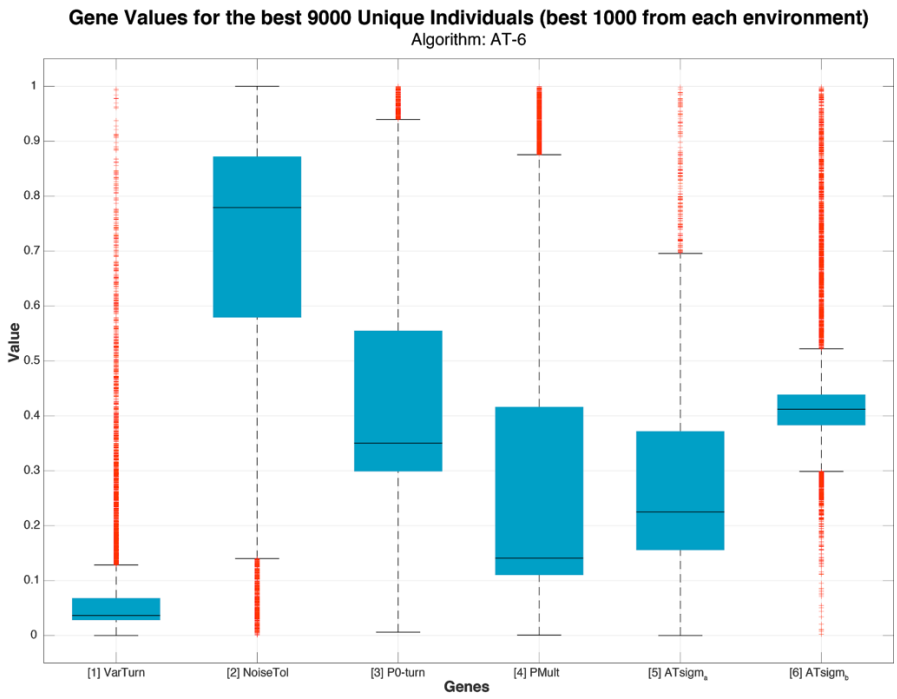


Figure 6.12 Box plot of the values for the input parameters (genes) of the best 9000 individuals (1000 from each environment) evolved in SR, MR, LR, SA, MA, LA, SP, MP, and LP, running the Foraging Algorithm AT-6.

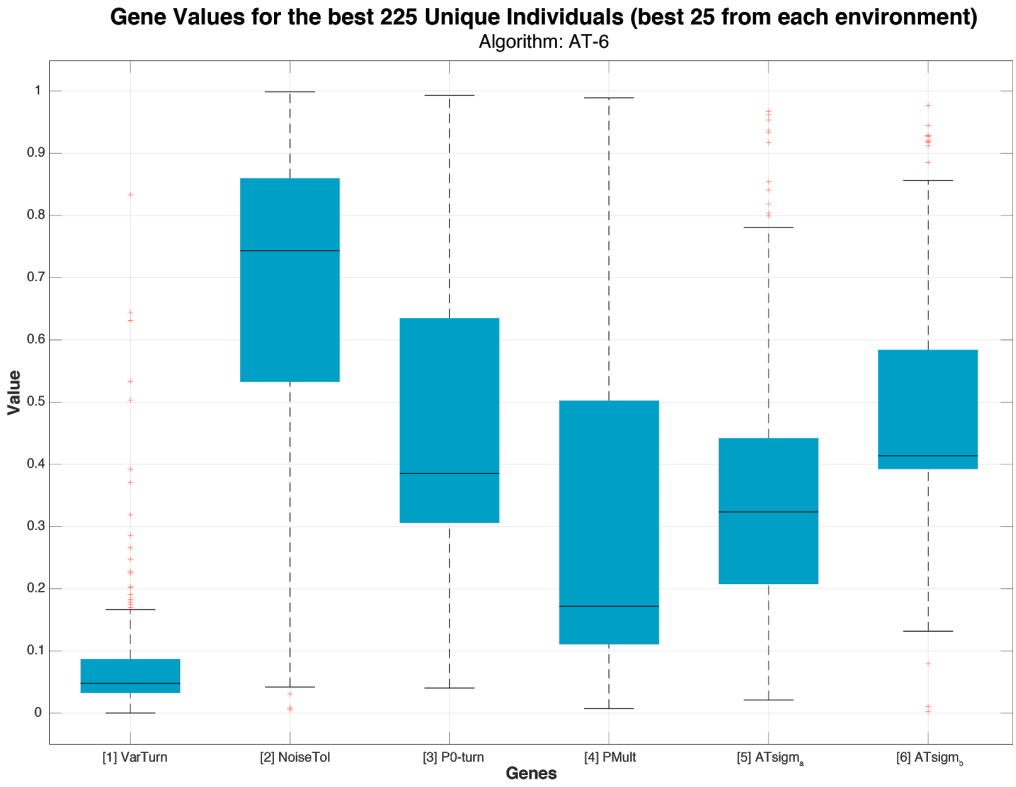


Figure 6.13 Box plot of the values for the input parameters (genes) of the best 225 individuals (25 from each environment) evolved in SR, MR, LR, SA, MA, LA, SP, MP, and LP, running the Foraging Algorithm AT-6.

6.3.3. Cross-Trials

In a broad view, we can distinguish two basic groups: one composed by species evolved in Rich and Average environments, and the other composed by species evolved in Poor environments:

- **Genomes evolved in Rich and Average environments** exhibit quite similar performance across all Rich and Average environments, generally achieving costs quite close to those of the native populations in these environments. They all performed significantly worse than the native populations and the other group in Poor Environments (Figure 6.14).
- **Genomes evolved in Poor environments** exhibit quite a similar performance in the Poor environments, generally achieving costs quite close to those of the native populations, but performing significantly worse than the native populations in Rich and Average environments. Surprisingly, the genomes evolved in LP outperform the native populations in SP and MP (Figure 6.14).

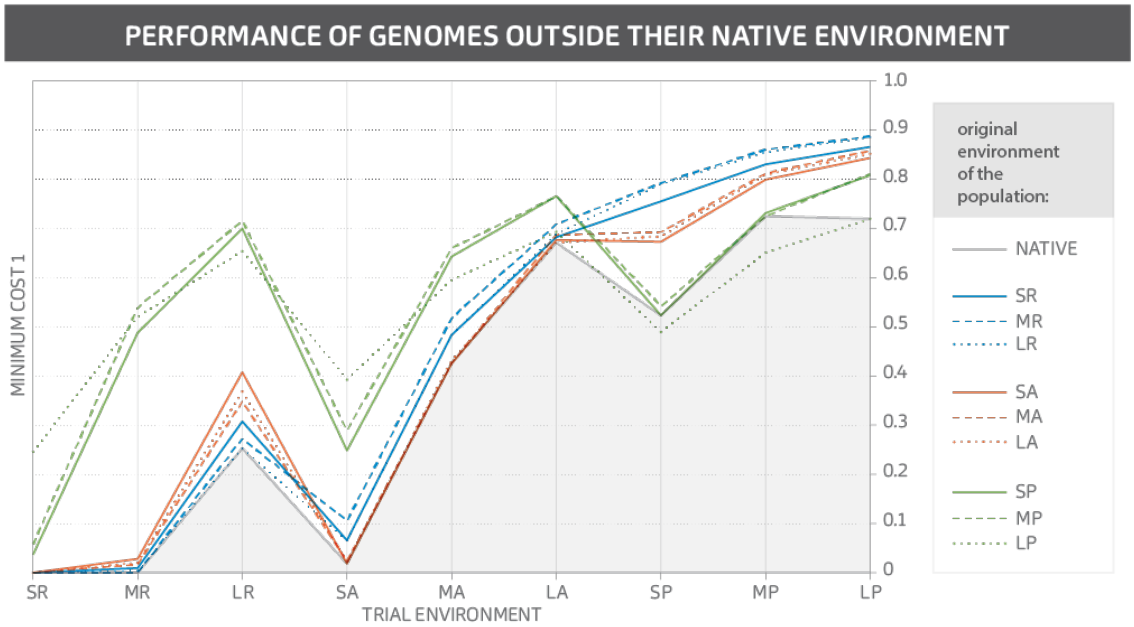


Figure 6.14 Performance of the best genomes (25 of each environment) in all the 9 environments.

6.3.4. Genetic Analysis

In this step, the best 25 individuals from each environment (a total of 225) were tested outside their native environment. Each simulated instance of the 9 environments was exactly the same as the one used in the optimisation step.

The same procedure was adopted for each cluster centroid and the clustering scheme was chosen based on the accuracy of the results obtained for the cluster centroid and all its members (Figure 6.15)

Figure 6.15 Compared performance of cluster centroid (black line) and cluster members (colour lines), each tested in the 9 environments.

Clusters obtained from the best 25 genomes evolved in each of the environments (225 in total). The tight grouping of the cluster members and cluster centroids indicate that the clustering was effective, with each cluster representing class of solution (species).

6.3.5. Characterization of Evolved Genomes

The results of genotype clustering reveal some striking patterns (Figure 6.16, 8.17) (Table 6.4). Firstly, at most two species feature in each of the 9 environments, with a single species present in five of these. It is also noticeable at first glance some pairs of clusters are more similar to each other than to the other clusters (1 and 2, 3 and 4, 5 and 6), which will also be discussed further ahead.

The two species from Large Poor (LP) environments (Sp_5 and Sp_6) appeared to have minimised sigmoid position (g_6) at the same time that they have maximised sigmoid steepness (g_5) (the resulting sigmoids are shown in Figure 6.18). Therefore they evolved to stop suddenly in partially favourable conditions which other species would not consider favourable enough to reduce speed or stop. That explains why they perform more poorly in the other environments because they fall into light spots with low intensity (traps), missing the better quality food sources available in Rich and Medium environments.

These two Species (Sp_5 and Sp_6) have also minimized the value of Base Probability of Turn (*BaseProb*: g_3). This, combined with relatively high values of Noise Tolerance (*NoiseTol*: g_2), means they have evolved an unexpected strategy where they virtually never turn, instead moving straight until they hit a light spot and stop, exploiting the wrapping boundaries of the world. It can be observed that in these two species native to LP, the values of Probability Multiplier (g_4) and Turn Variation (g_1) are distributed across a broad range (particularly when considering Species 5 and 6 together). This makes sense because their evolved strategy means that these genes are irrelevant (because they don't turn). In fact, observation of the simulated behaviour suggests that Species 5 and 6 actually exhibit virtually identical behaviour. Given that only genes 1 and 4 are functionally irrelevant, they are subject to random genetic drift. The clustering algorithm (which is concerned with gene values rather than behaviour) appears to have separated these into two species mainly on the basis of high/low values of g_4.

The Large Poor (LP) environment was the only one in which a strategy using no turns at all evolved, and as a result the genetic profiles of Species 5 and 6 are strikingly different from all others (see Figure 8.16). Focussing on Species 1 to 4, the differences are more subtle but still significant. Most notably, when comparing Species 3 and 4 (evolved in Poor environments SP and MP) to Species 1 and 2 (evolved in Rich and Average environments), there is a striking difference in the sigmoid genes (g_5 and g_6). Comparing the resulting shapes of the sigmoids (Figure 8.18) shows that Species 3 and 4 will stop much more suddenly and in lower quality light sources than Species 1 and 2. This is a necessary adaptation to the low-quality energy sources available in Poor environments.

When comparing Species 3 and 4 to each other, the only somewhat significant difference in again in the sigmoid genes (g_5 and g_6). The light intensity required to bring robots to very low speeds is similar, but Species 4 has a sharper transition (Figure 8.18). The behavioural relevance of this is unclear, and the fact that these two species perform very similarly in cross-trials (Figure 8.14) and are both native to MP (although only Species 3 is native to SP) suggests that it is minimal.

Finally, when comparing Species 1 and 2 the only significant genetic differences are in g_4 and g_6 (see Table 6.4). The difference in g_6 means that Species 1 (found primarily in Rich environments) slows down at somewhat higher light levels than Species 2 (found primarily in Average environments), which is a subtle but logical adaptation to higher consistency of good quality energy sources found in Rich environments. The fact that Species 1 has a somewhat higher value of *PMult* (g_4) is consistent with the fact that Rich environments will, on average, have higher intensity light spots and hence steeper light gradients. This means that a sensed change in light intensity is more likely to be due to the underlying gradient (as opposed to sensor noise), and therefore warrants a stronger effect on turning probability.

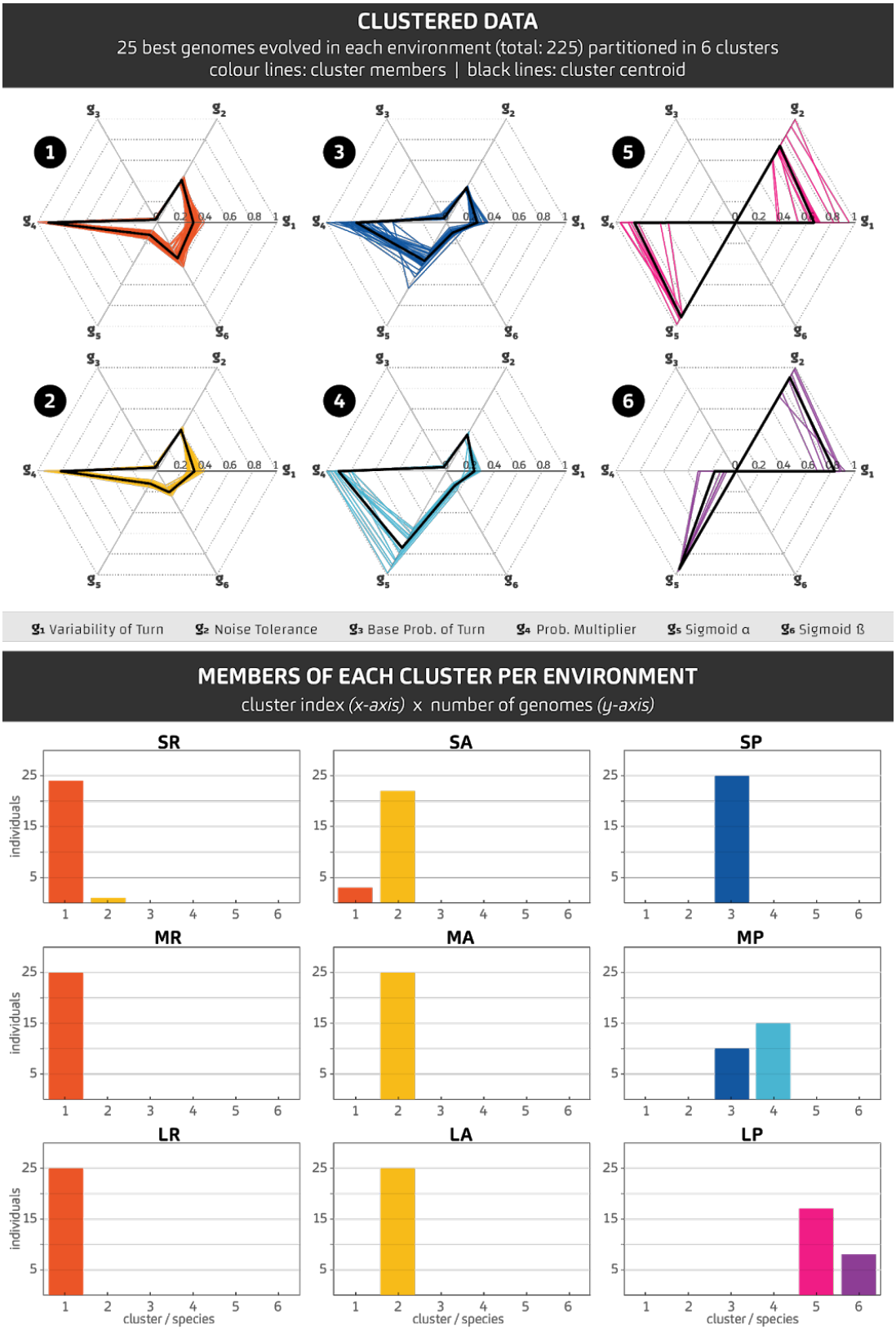


Figure 6.16 Results of Genotype Clustering (Series H: AT-6).

Top: all members of the six clusters represented as spider plots (coloured lines) and the cluster centroid (black lines). Each radial axis of a spider plot represents a value ($g_{1\text{-}6}$) of the input vector. Bottom: Distribution of members of each cluster within the population of each environment.

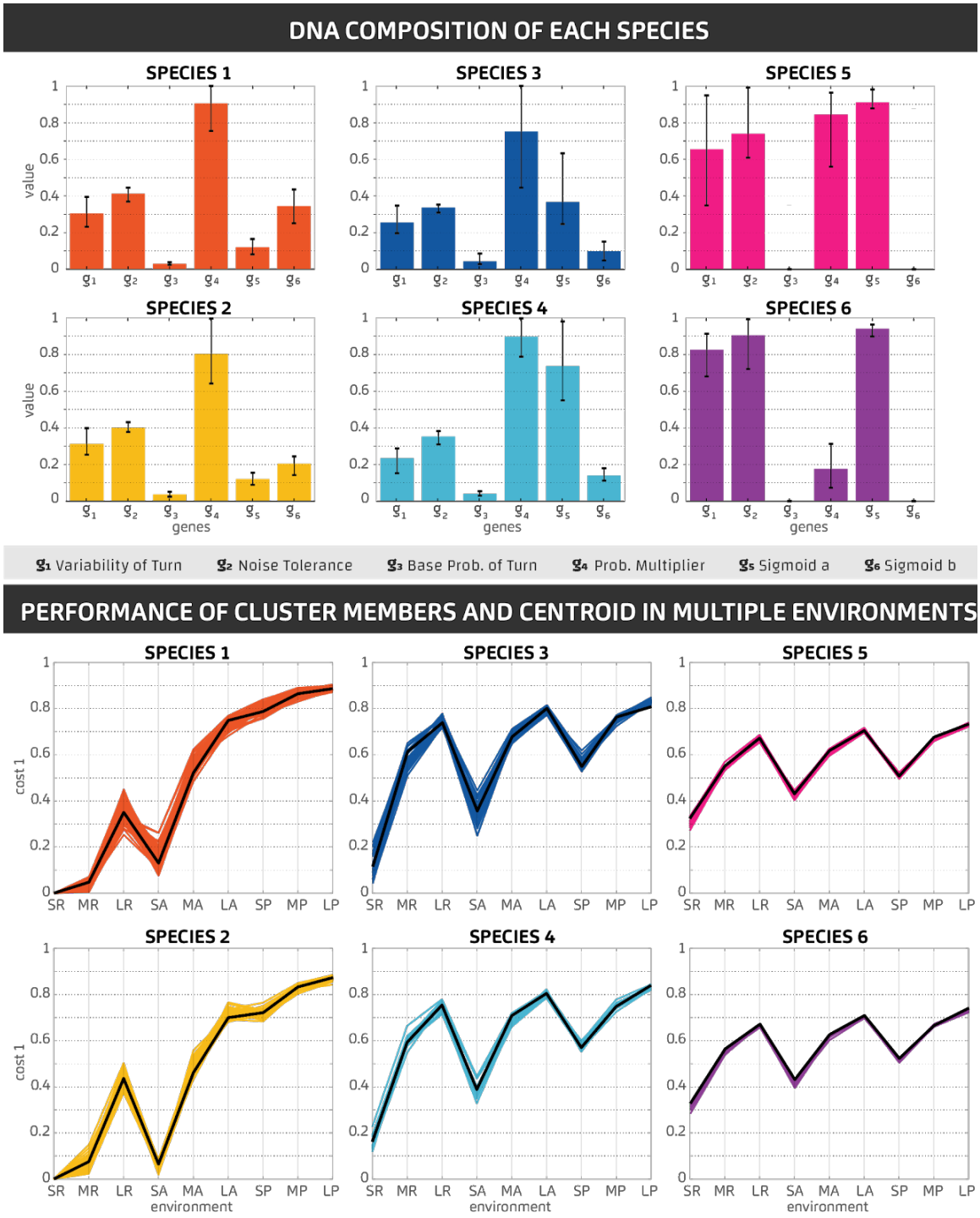


Figure 6.17 Environmental distribution and DNA composition of species.

Top: Bar chart with the number of members of each species present on each environment. Bottom: Bar chart with DNA composition of each species.

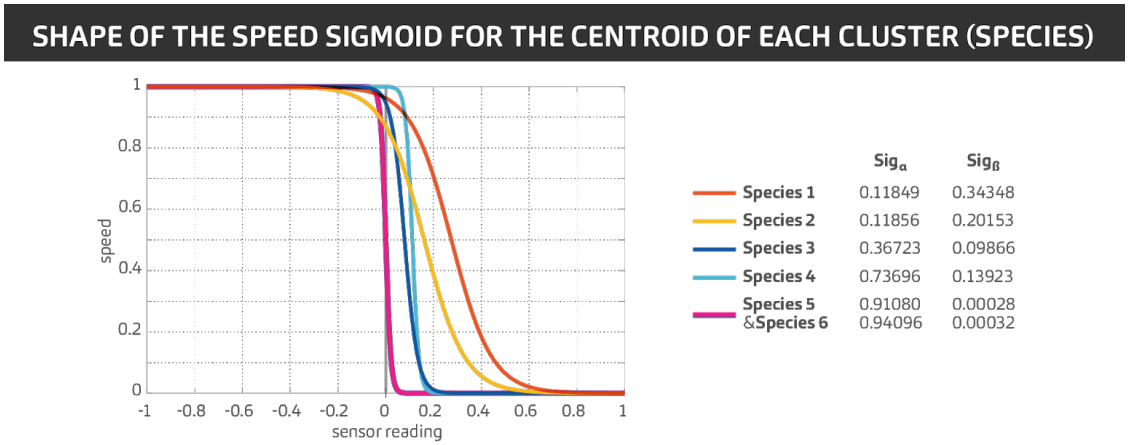


Figure 6.18. Variation in the shape of the speed sigmoid function for the centroid of each cluster.

Table 6.4 Decoded Centroid Genome of each species and respective standard deviation

		GENE					
		g_1	g_2	g_3	g_4	g_5	g_6
		VarAngle	*NoiseTol*	*BaseProb*	*PMult*	Sig_α	Sig_β
SPECIES	1	109º ± 33º	1.64 ± 0.17	0.003 ± 0.001	180.7 ± 29.9	11.8 ± 4.52	0.27 ± 0.07
	2	112º ± 31º	1.59 ± 0.12	0.003 ± 0.001	160.4 ± 38.7	11.8 ± 3.49	0.16 ± 0.05
	3	92º ± 33º	1.35 ± 0.11	0.004 ± 0.004	150.3 ± 61.2	36.7 ± 26.60	0.08 ± 0.04
	4	85º ± 29º	1.41 ± 0.17	0.004 ± 0.001	179.7 ± 19.8	73.7 ± 24.41	0.11 ± 0.03
	5	236º ± 110º	2.95 ± 1.01	0.000 ± 0	169.0 ± 56.8	91.1 ± 7.06	0.00 ± 0
	6	298º ± 52º	3.62 ± 0.73	0.000 ± 0	35.2 ± 27.5	94.1 ± 4.22	0.00 ± 0

6.4. Conclusions

The work replicates, in a very simplified form, the way that modest changes in environmental conditions can lead to evolutionary adjustments, while sufficiently extreme changes can result in the same building blocks being used in fundamentally different ways (see differences between Species 5/6 and 1 to 4). Furthermore, the approach of grouping solutions evolved in different environments and then analysing these on the basis of genetic (as opposed to behavioural) similarity is novel, to the best of my knowledge, and yielded interesting results that aligned very well with behavioural outcomes.

Optimisations in LR and in MA seem to have converged too early, what is suggested by the processes converging before the other processes, and by the pattern of the graphic of cost minimization over generations (Figure 6.5). Instead of several smooth transitions

between plateaus, $Cost_1$ drastically drops (around generations 20^{th} and 70^{th}, respectively), displaying no further relevant evolution - that might indicate the convergence to a local optimum and/or a loss in diversity, followed by an Early Convergence. This conclusion is also supported by the dissonance between the minimized costs and the *Light Index* values calculated for LR (Figure 6.8). As optimisation results are essential for all subsequent experimental steps, future optimisation cycles may encompass at least one replicate (per environment) running in parallel, and thus ensuring the best possible set of results for the whole process.

Contrary to what was expected in the Cross-environment experiments, populations evolved in the easiest environments were able to outperform native ones in 4 of the 9 environments, while the populations evolved in the hardest environments only performed better than the native in only 2 of them. This result might be explained by the fact that easier, hence more forgiving, environments allow a smoother streamlining process, where individuals with a weaker (but acceptable) performance may lead to a successful offspring. Some of the newer approaches to Problem-Solving and Evolutionary Algorithms implement strategies to slowly update the level of demand/difficulty, as well as elitism and penalties applied to individual *fitness values*. Some of these could improve future processes of Optimization.

Some of the results obtained in the Native Environment Trial Replicates could be explained by the variance of the *Light Index*. However, it is also likely that the individuals evolved in the Optimisation Step also present some level of *overfitting* for the original instance of the environments they evolved in. In future works, the Optimisation Step may benefit from evaluating each genome in more than one instance of the environment.

Chapter 7
Second Refined Algorithm: ATRP-8 (Experiment Series I)

This chapter will present the design of the second refined *C. elegans*-based Foraging Algorithm, ATRP-8, and the results of experimental series I. The 8 input parameters of the foraging algorithm were optimised in 9 environments (SR, MR, LR, SA, MA, LA, SP, MP, and LP), containing both attractants and repellents (Table 7.1). These environments present varying values for combined *Light Index* values of attractants and repellents (Light Index AT+RP), but a constant ratio between them (Light Index AT/RP), as seen in Table 7.2.

Table 7.1 Summary of the parameters and main results of the experiments in Series I: Optimisation in 9 environments (SR, MR, LR, SA, MA, LA, SP, MP, and LP) with attractants and repellents, for robots equipped with sensors with sensor noise of value 1%

OPT.		SIM.	ROBOT	ENVIRONMENT		Number of Function Evaluations	Minim. Cost 1	Minim. Cost 2	Minim. Cost 3	Non-Dominated Solutions
EA	Pool Size	Time Step	Sensor Noise	Quality of Resources	Field Size					
DE	200	4	1%	R	S	11,506	0.578	0.979	0.964	19
					M	18,060	0.630	0.989	0.973	20
					L	37,638	0.716	0.989	0.987	16
				A	S	17,386	0.724	0.994	0.958	16
					M	41,540	0.740	0.993	0.978	24
					L	38,599	0.792	0.995	0.987	18
				P	S	9,120	0.750	1.000	0.962	13
					M	9,965	0.815	1.000	0.981	6
					L	8,168	0.859	1.000	0.988	14
				TOTAL:		191,982				

Table 7.2 Simulated environments, minimised Costs and Light Index in Series I, subset I.1 (Foraging Algorithm ATRP-8).

ENVIRONMENT		Minim. Cost 1	Minim. Cost 2	Minim. Cost 3	Light Index (AT)	Light Index (RP)	Light Index (AT+RP)	Light Index (AT/RP)
Quality of Resources	Field Size							
R	S	0.578	0.979	0.964	68.868	-89.265	-20.397	-0.771
	M	0.630	0.989	0.973	30.885	-40.029	-9.143	-0.772
	L	0.716	0.989	0.987	17.219	-22.316	-5.097	-0.772
A	S	0.724	0.994	0.958	47.271	-61.272	-14.001	-0.771
	M	0.740	0.993	0.978	21.200	-27.476	-6.276	-0.772
	L	0.792	0.995	0.987	11.819	-15.318	-3.499	-0.772
P	S	0.750	1.000	0.962	23.635	-30.636	-7.001	-0.771
	M	0.815	1.000	0.981	10.600	-13.738	-3.138	-0.772
	L	0.859	1.000	0.988	5.910	-7.659	-1.749	-0.772

7.1. Algorithm Design

The foraging algorithm ATRP-8 extends the foraging algorithm presented in the previous chapter. This algorithm is also composed of two key behaviours: runs and turns, controlled by a set of eight parameters ($g_{1:8}$), also referred to in this work as the 'genome', as shown in Table 7.3.

This algorithm encompasses both positive and negative chemotactic behaviours, as the agents react to attractant and repellent stimuli distributed in the field.

Table 7.3 Set of input parameters (genome) for the *C. elegans*' bio-inspired minimalist algorithm for positive and negative chemotaxis in Foraging Algorithm ATRP-8.

GENE	PARAMETER	PROCESS	APPLICATION
g_1	$VarAngle$	Positive and Negative Chemotaxis (both)	controls the variability of the angle of a turn;
g_2	$BaseProb$		sets the base probability of turning, in the absence of any change in sensed light level
g_3	$PMult$		sets the multiplier (divisor) of the base turning probability when the level of attractant decreases, and the divisor (multiplier) of the base turning probability when the level of repellent increases.
g_4	Sig_{α}^{At}	Positive Chemotaxis (Attractants)	sets the steepness of the sigmoid curve that controls speed according to the sensor reading for attractants.
g_5	Sig_{β}^{At}		defines the offset of the sigmoid curve that controls speed according to the sensor reading for attractants.
g_6	$NoiseTol$	Positive and Negative Chemotaxis	controls the tolerance of the robot to the variation in sensed light

		(both)	
g_7	Sig_{α}^{Rp}	Negative Chemotaxis (Repellents)	sets the steepness of the sigmoid curve that controls speed according to the sensor reading for repellents.
g_8	Sig_{β}^{Rp}		defines the offset of the sigmoid curve that controls speed according to the sensor reading for repellents.

As in the previous experiment series, the battery level of each robot is updated according to the intensity of light (attractant) the robot is currently exposed to (i^{in}), as well as how much was spent on moving (i^{out}) and running basic systems (i_{BMR}):

$$Bat_t^R = Bat_{t-1}^R + \left(Light_t^{[x,y]} \times i_{max}^{in}\right) - Speed_t^R \times i_{max}^{out} - i_{BMR} \quad [7.1]$$

As the environments tested in this series also contain repellents that are harmful, the battery is further depleted according to the damage the robot absorbed, as a result of getting exposed to repellents:

$$Bat_t^R = Bat_t^R - \left(Repellent_t^{[x,y]} \times dmg_{max}^{in}\right) \quad [7.2]$$

Also similarly to the previous experiments, the simulation program updates robots' positions, checks for extinguished light spots (replacing them with new ones if necessary) and checks which robots are 'alive' – a robot permanently 'dies' if its battery is depleted.

The reasoning process on each time step (t) starts when the agent acquires the sensor readings for the intensity of Attractants ($SensAt_t^R$) and Repellents ($SensRp_t^R$) at its current position [x,y], obtained from the actual intensity added by a (negative or positive) sensor noise, obtained from the multiplication of the set sensor noise ($SensNoise$) by a uniformly distributed random number between -1 and 1 ($RandNum$) [9.3], [9.4]. $\Delta SensAt_t^R$ and $\Delta SensRp_t^R$ are then obtained from the current and previous sensor readings [9.5], [9.6] in order to calculate the probability of initiating a turn ($PTurn^{R}_{t}$).

$$SensAt_t^R = Attractant_t^{[x,y]} + RandNum \times SensNoise \quad [7.3]$$

$$SensRp_t^R = Repellent_t^{[x,y]} + RandNum \times SensNoise \quad [7.4]$$

$$\Delta SensAt_t^R = SensAt_t^R - SensAt_{t-1}^R \quad [7.5]$$

$$\Delta SensRp_t^R = SensRp_t^R - SensRp_{t-1}^R \quad [7.6]$$

At this point, the calculation will happen separately for attractant and repellent stimuli. For attractant stimulus, if $\Delta SensorStimuli_t^R$ is sufficiently positive, $PTurnAt_t^R$ decreases, whereas if it is sufficiently negative, $PTurnAt_t^R$ increases. If both current and previous values are sufficiently similar (as determined by $NoiseTol$), then $PTurnAt_t^R$ maintains the value of $BaseProb$ (Fig.9.1).

For repellent stimulus, if $\Delta SensorStimuli_t^R$ is sufficiently positive, $PTurnRp_t^R$ increases, whereas if it is sufficiently negative, $PTurnRp_t^R$ decreases. If the current and previous values are sufficiently similar (as determined by *NoiseTol*), then $PTurnRp_t^R$ maintains the value of *BaseProb* (Fig. 9.1). The pseudocode is presented as follows:

for Attractant Stimulus:

```
if (ΔSensAt_t^R > SensNoise × NoiseTol) then
    PTurnAt_t^R = BaseProb ÷ PMult
else if (ΔSensAt_t^R < SensNoise × NoiseTol) then
    PTurnAt_t^R = BaseProb × PMult
else
    PTurnAt_t^R = BaseProb
end
```

for Repellent Stimulus:

```
if (ΔSensRp_t^R > SensNoise × NoiseTol) then
    PTurnRp_t^R = BaseProb × PMult
else if (ΔSensRp_t^R < SensNoise × NoiseTol) then
    PTurnRp_t^R = BaseProb ÷ PMult
else
    PTurnRp_t^R = BaseProb
end
```

The final $PTurn_t^R$ is rationed by averaging $PTurnAt_t^R$ and $PTurnRp_t^R$.

As in AT-6 (chapter 8), once $PTurn_t^R$ is set, a random number (0 to 1) is generated and, if it is less than or equal to $PTurn_t^R$, the robot will perform a turn (Fig.9.1). When performing a turn, the yaw ($\Delta\theta$) will be calculated using another uniform random number between -1 and 1 (*RandNum*), according to:

$$\Delta\theta_t^R = 180° + (VarAngle \times RandNum) \qquad [7.7]$$

Speed Modulators are calculated separately for Attractant and Repellent stimuli, each using its specific set of parameters, as seen in Table 7.2. Similar to algorithm AT-6 (Chapter 8), speed is modulated by an inverse logistic function and a logistic function, controlled by the combination of the current sensor reading and the pair of input parameters Sig_α and Sig_β, according to:

$$SpeedModAt_t^R = 1 - \frac{1}{1+(e^{Sig_\alpha^{At} \times (SensAt_t^R - Sig_\beta)})} \times V_{max} \qquad [7.8]$$

$$SpeedModRp_t^R = 1 - \frac{1}{1+(e^{Sig_\alpha^{Rp} \times (SensRp_t^R - Sig_\beta)})} \times V_{max} \qquad [7.9]$$

The resulting speed is then obtained by averaging the results of equations [7.8] and [7.9].

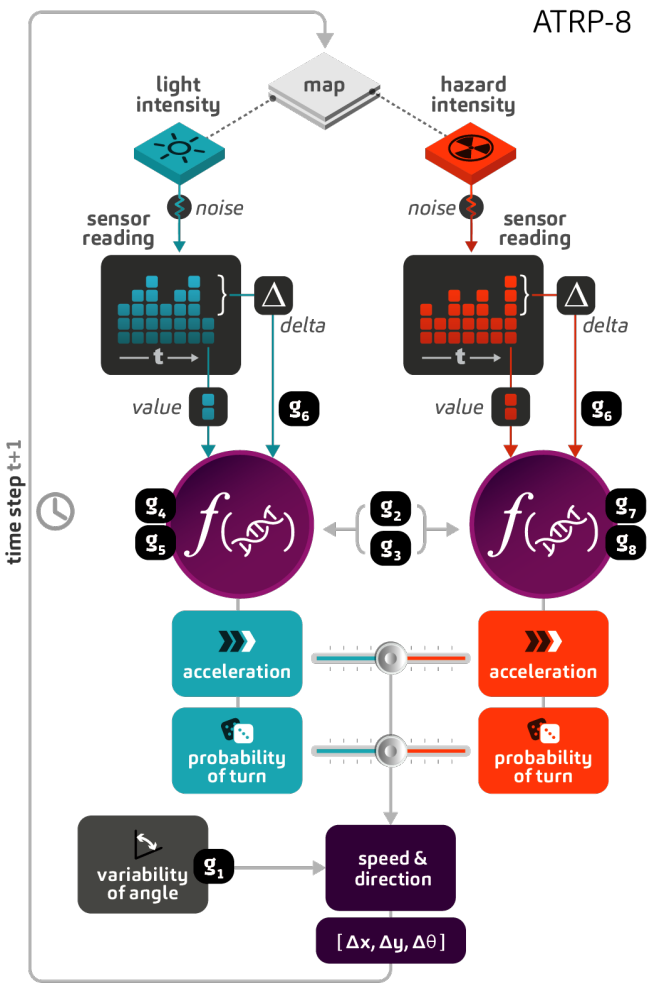


Figure 7.1 Flowchart of the control algorithm ATRP-8.

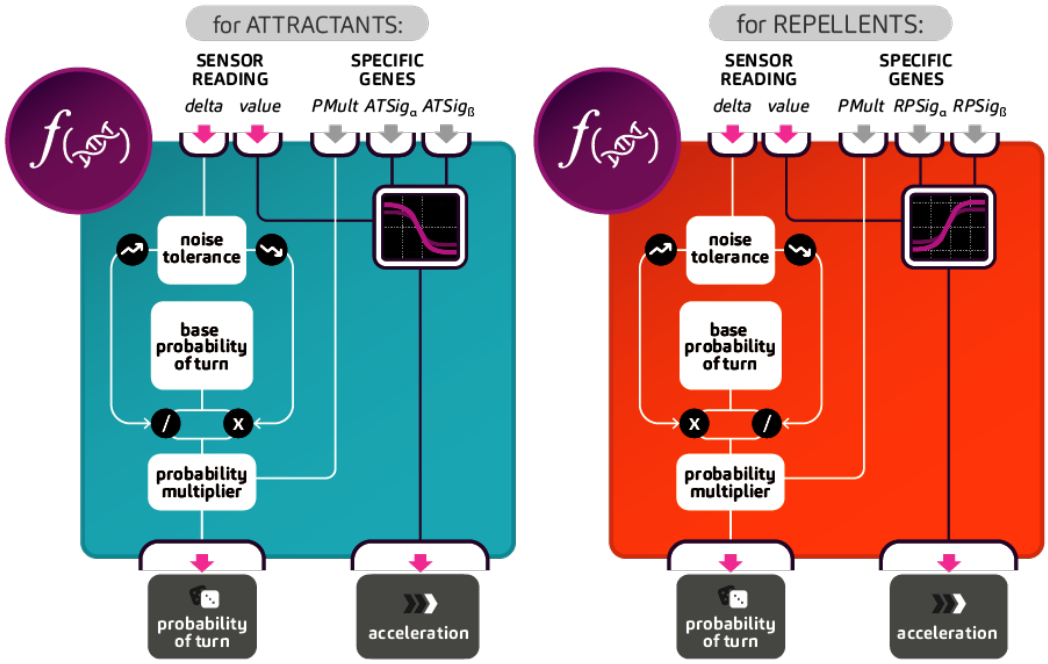


Figure 7.2 Diagram of the function outputting the probability of turn and acceleration. The same function is used for both attractants and repellents.

7.2. Simulation and Experiment Design

The algorithm was evolved for 9 different environments, created from the combination of three sizes (small, medium, and large) and three resource quality types (rich, average, and poor). The environments are populated with attractants and repellents in a balanced ratio, as described in Section 5.2.

7.3. Results and Discussion

In this set of optimisation cycles for ATRP-8, virtual robots running this foraging algorithm were evolved in 9 environments containing attractants and repellents. The best solutions found were grouped into 3 and 7 clusters by genetic similarity. The clusters were validated by comparing the results of its members on the cross-trials.

7.3.1. Parameter Optimisation for Different Environments

The best values of Cost 1 were found in SR, MR, LR, SA, MA, SP, LA, MP, and LP, respectively (Table 7.1, Figure 7.3).

Optimisation in MP reached the stopping criteria of diversity loss earlier than the others: after 60 generations. Optimisations in SP, LP and SR converged before 70 generations, and

optimisations in LA, MA, and LR ran for longer than the other optimisations, for more than 250 generations.

When it comes to Cost 2, the environments in which the optimisation achieved the best cost were, respectively: SR, LR, MR, MA, SA, LA, SP, MP, and LP (Figure 7.4).

As it has been mentioned earlier in this chapter, the input parameters were not optimised for Cost 3, even though the value of it was registered. The best values found (but not optimised) were, respectively: SA, SP, SR, MR, MA, MP, LR, LA, and LP.

In terms of availability of attractants, measured by the *Light Index*, the environments with more attractant light available are, respectively: SR, SA, MR, SP, MA, LR, LA, MP, and LP (Table 7.2).

Unlike the previous experiment series, in which the minimised costs followed a combination of field size and quality of resources, in this series (Series I), the minimised Costs 1 and 2 are closely related to the quality of resources.

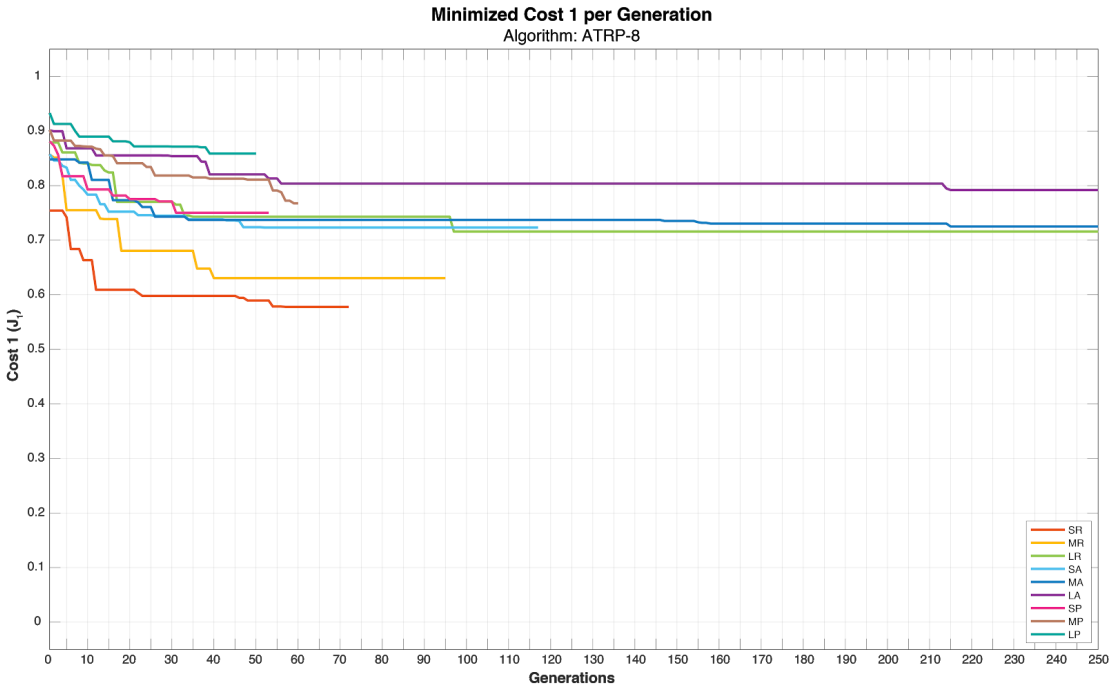


Figure 7.3 Cost 1 (J_1) optimised for Behavioural Algorithm ATRP-8, in 9 environments (SR, MR, LR, SA, MA, LA, SP, MP, and LP), with attractants and repellents.

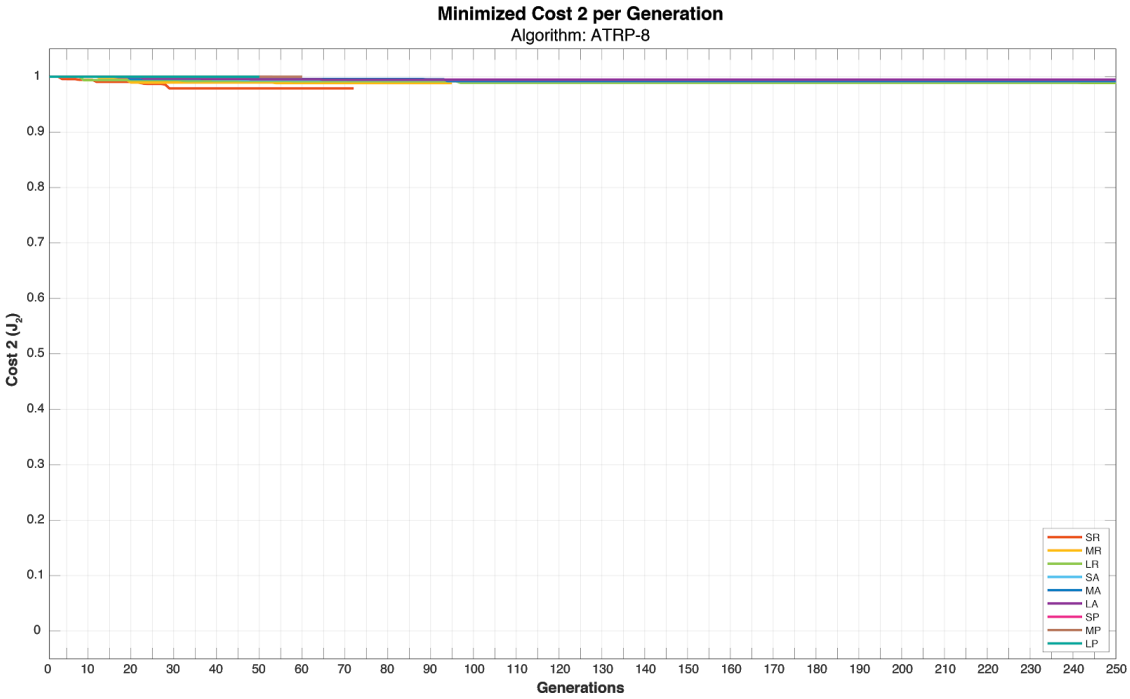


Figure 7.4 Cost 2 (J_2) optimised for Behavioural Algorithm ATRP-8, in 9 environments (SR, MR, LR, SA, MA, LA, SP, MP, and LP), with attractants and repellents.

7.3.2. Selection of the best individuals

Figure 7.5 shows the best 8000 unique individuals evolved in the 9 environments using Foraging Algorithm ATRP-8. As can be seen, Cost 1 presents a steep drop in approximately the best 100 individuals (Figure 7.5).

Further experiments explored the variance of Cost 1 in different aliquots of the population and values around 25 individuals presented a good combination between diversity (gene variance) whilst still retaining a narrow range of Cost 1 values for all environments (Fig. 9.6, 9.7, 9.8, and 9.9). This number was also maintained as it matched the number of best individuals chosen in the previous experiment series (Series H).

In all environments, g_6 (*NoiseTol*) converged to a very narrow range. The same happened to g_3 (*VarAngle*), as seen in Figure 7.10 and in Table 7.4.

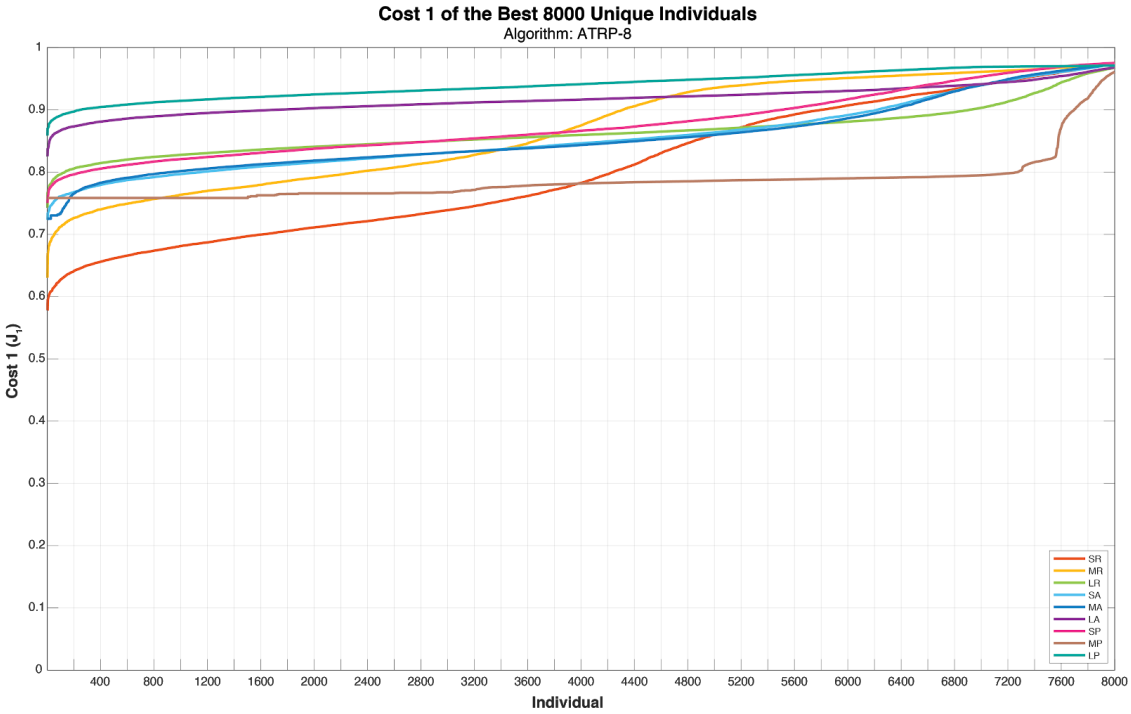


Figure 7.5 Cost 1 (J_1) of the best 8000 individuals running the behavioural algorithm ATRP-8, evolved in 9 environments (SR, MR, LR, SA, MA, LA, SP, MP, and LP) with attractants and repellents.

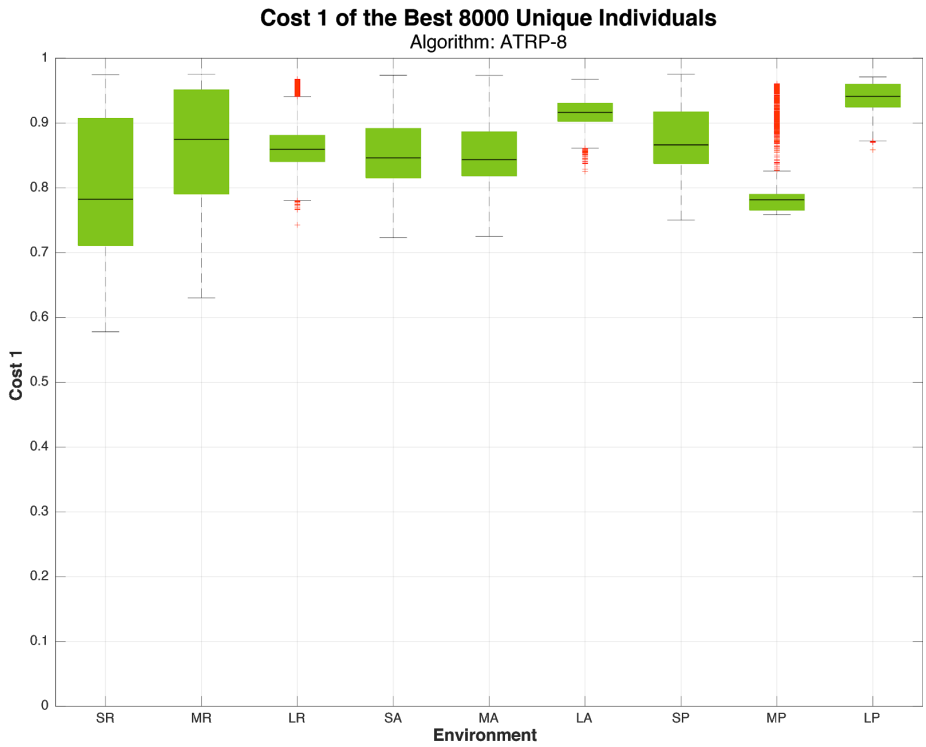


Figure 7.6 Box plot of the Cost 1 (J_1) of the best 8000 individuals (from each environment) evolved running the behavioural algorithm ATRP-8 in 9 environments (SR, MR, LR, SA, MA, LA, SP, MP, and LP) with attractants and repellents.

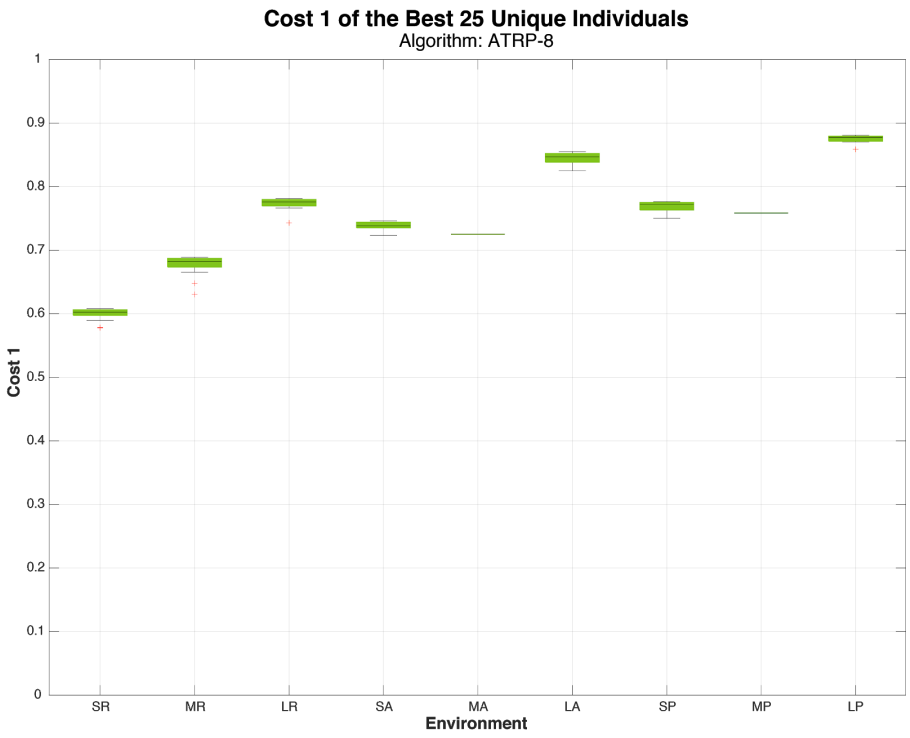


Figure 7.7 Box plot of the Cost 1 (J_1) of the best 25 individuals (from each environment) evolved running the behavioural algorithm ATRP-8 in 9 environments (SR, MR, LR, SA, MA, LA, SP, MP, and LP) with attractants and repellents.

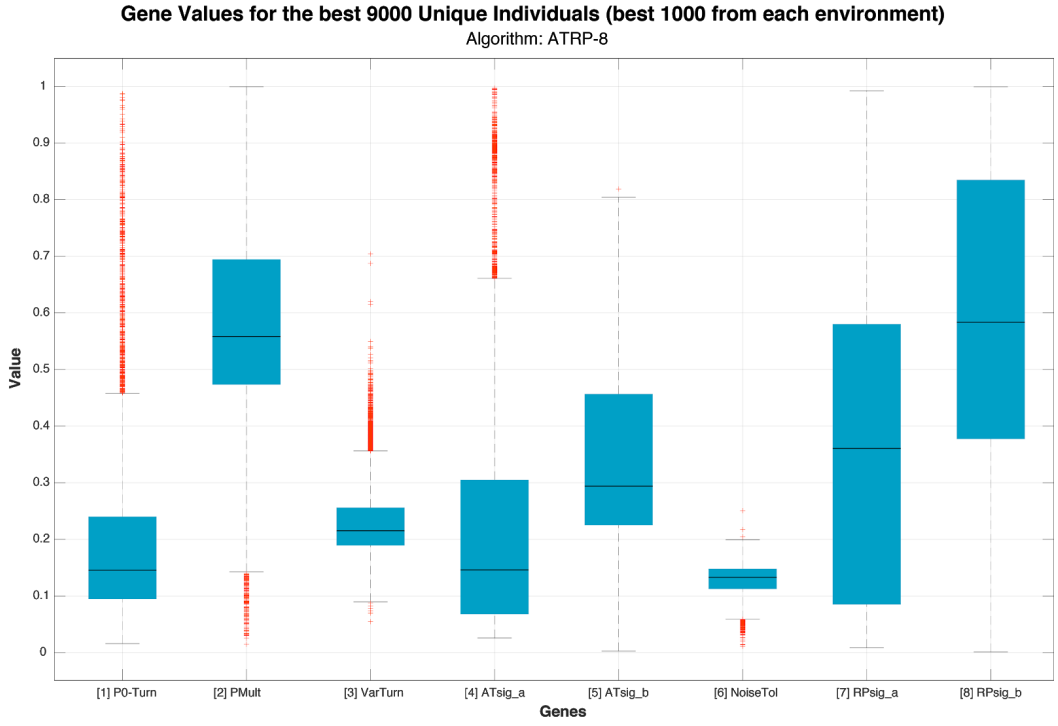


Figure 7.8 Box plot of the values for the input parameters (genes) of the best 9000 individuals (100 from each environment) evolved in SR, MR, LR, SA, MA, LA, SP, MP, and LP, running the Foraging Algorithm ATRP-8.

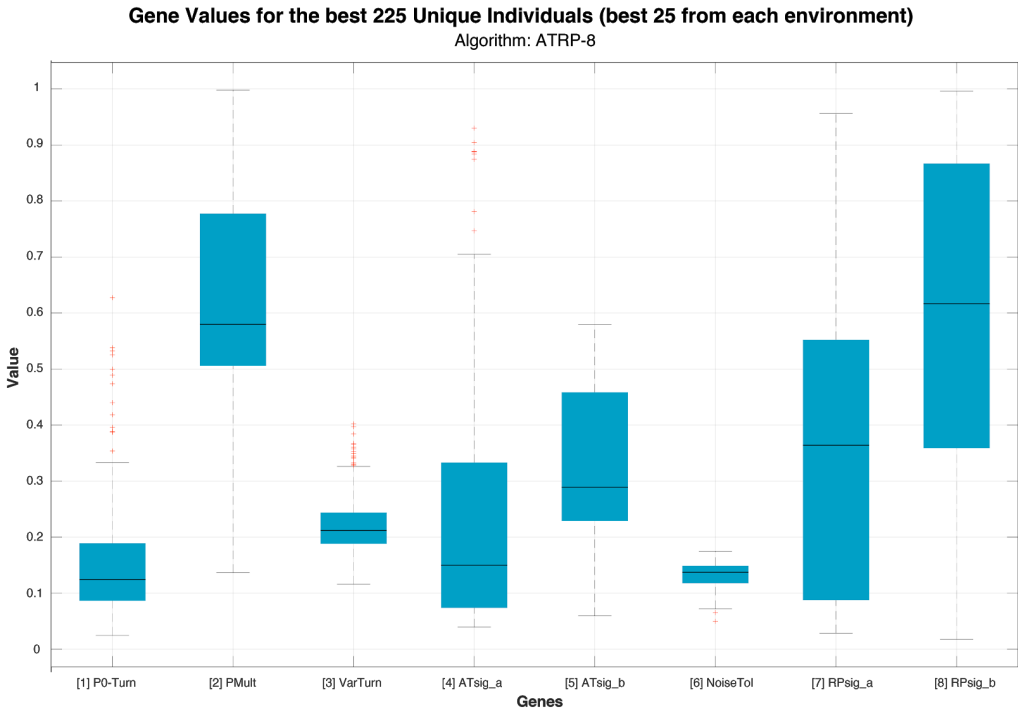


Figure 7.9 Box plot of the values for the input parameters (genes) of the best 225 individuals (25 from each environment) evolved in SR, MR, LR, SA, MA, LA, SP, MP, and LP, running the Foraging Algorithm ATRP-8.

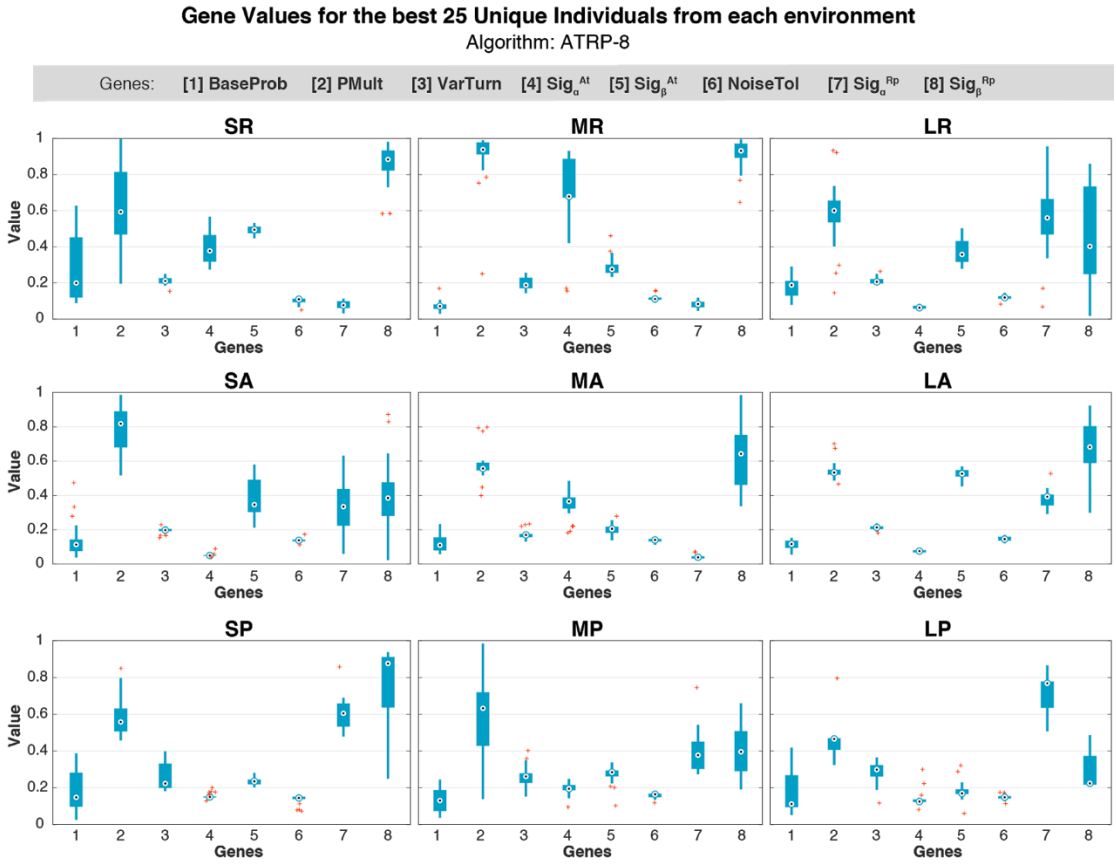


Figure 7.10 Box plot of the values for the input parameters ("DNA") of the best 25 individuals evolved in each environment, running the Foraging Algorithm ATRP-8.

Table 7.4 Percentiles 25%, 50% (median), and 75% of the gene values of the best 25 individuals evolved in each environment with ATRP-8.
Cells in blue indicate pairs of genes related to Speed Modulation regarding the presence of Attractants and the cells in red indicate those related to Repellents.

ENV.	GENES								Percentile
	g1	***g2***	***g3***	***g4***	***g5***	***g6***	***g7***	***g8***	
	BaseProb	*PMult*	*VarAngle*	Sig_{α}^{At}	Sig_{β}^{At}	*NoiseTol*	Sig_{α}^{Rp}	Sig_{β}^{Rp}	
SR	0.1184	0.4677	0.1971	0.3169	0.4756	0.0924	0.0603	0.8224	25%
	0.1992	**0.5931**	**0.2103**	**0.3772**	**0.4943**	**0.1093**	**0.0766**	**0.8838**	**50%**
	0.4522	0.8151	0.2256	0.4656	0.5119	0.1119	0.0982	0.9344	75%
MR	0.0555	0.9112	0.1693	0.6719	0.2556	0.1066	0.0642	0.8928	25%
	0.0703	**0.938**	**0.1875**	**0.6788**	**0.2751**	**0.1106**	**0.0826**	**0.9319**	**50%**
	0.0813	0.9782	0.2286	0.8878	0.3014	0.1192	0.095	0.9715	75%
LR	0.1287	0.5349	0.1965	0.0566	0.316	0.1111	0.4679	0.2487	25%
	0.1877	**0.6005**	**0.2057**	**0.0622**	**0.3574**	**0.1186**	**0.5603**	**0.4021**	**50%**
	0.211	0.6568	0.2203	0.073	0.4318	0.1259	0.6642	0.7342	75%
SA	0.0751	0.6804	0.191	0.0472	0.3027	0.1321	0.2237	0.2801	25%
	0.1132	**0.8185**	**0.1978**	**0.0494**	**0.3461**	**0.1382**	**0.3343**	**0.385**	**50%**
	0.143	0.8906	0.2048	0.0511	0.4909	0.1421	0.4372	0.4768	75%
MA	0.0785	0.5454	0.1558	0.3228	0.1811	0.1313	0.0316	0.4614	25%
	0.1098	**0.557**	**0.1698**	**0.365**	**0.2049**	**0.1405**	**0.0392**	**0.6428**	**50%**
	0.155	0.5908	0.1741	0.3879	0.2183	0.1463	0.0448	0.7527	75%
LA	0.0936	0.5207	0.2046	0.0673	0.5095	0.136	0.3417	0.5884	25%
	0.1166	**0.5336**	**0.213**	**0.0753**	**0.5267**	**0.1451**	**0.3923**	**0.6826**	**50%**
	0.1361	0.5502	0.2195	0.0811	0.5475	0.159	0.4055	0.8037	75%
SP	0.096	0.5059	0.1979	0.1466	0.2184	0.1367	0.5321	0.6359	25%
	0.1471	**0.5577**	**0.2227**	**0.1495**	**0.2344**	**0.1439**	**0.6046**	**0.8758**	**50%**
	0.2818	0.631	0.3313	0.1518	0.2449	0.151	0.6588	0.9114	75%
MP	0.0725	0.4275	0.225	0.1856	0.2608	0.1478	0.302	0.2895	25%
	0.1298	**0.6321**	**0.2608**	**0.1941**	**0.2841**	**0.1641**	**0.3769**	**0.3954**	**50%**
	0.1869	0.7196	0.278	0.2143	0.2937	0.1671	0.451	0.5082	75%
LP	0.0933	0.4058	0.2604	0.1215	0.1647	0.141	0.6342	0.2157	25%
	0.1113	**0.4652**	**0.2981**	**0.1242**	**0.1688**	**0.1474**	**0.7678**	**0.2235**	**50%**
	0.2678	0.4695	0.323	0.1361	0.1908	0.1539	0.7782	0.3729	75%

Figure 7.11 shows the shapes of different sigmoid functions evolved in different environments, the shape of the sigmoid curve is calculated based on median values of the genes of the best 25 individuals.

LR, SA, and LA have shallower curves for attractants, showing that they adjust their speed more slowly when they find attractants.

SR has a delayed reaction to the presence of attractants, compared with all other individuals evolved in other environments. SR, MA and MR have very steep curves, which

means they respond to attractants more quickly than the individuals evolved in other environments. Likely because they evolved in the richest environments, they specialise in locating attractants with high intensity, therefore higher risk may equal higher reward. This may be due to the fact that the more intense repellents are clustered around more intense attractants, some individuals may have evolved to make use of this pattern.

SP, MP, MA, and LP respond very quickly to attractants, with environments SP, MP, and LP showing shallower curves than MA. Indicating a more gradual response to attractants, however not as gradual as LR, SA, and LA.

In regards to the repellent reactions, MA has the shallowest curve, indicating the most gradual response, followed by SR and MR which also show more shallow curves.

LR, SA, LA, SP, MP and LP, all exhibit steeper curves, indicating a more rapid response to repellent stimuli. However, as can be seen in Figure 7.11 the threshold for response to repellents is different between the individuals evolved in these environments. SP responds more quickly to lower levels of repellents, followed by MP and LR. LP shows the most delayed response to repellents only responding when it reaches ~0.75.

In ATRP-8 the probability of a robot turning (*PTurn*) is controlled by 2 genes simultaneously, genes 1 and 2, which set the base probability of turn and the probability multiplier respectively (*BaseProb* and *PMult*). *PTurn* is also affected by noise tolerance (g_6). Noise tolerance (*NoiseTol*) converged to a very narrow range, which suggests an optimal level of tolerance to variations in the intensity detected by the sensor. Meaning that when the sensor noise is 1% and time step size (*TSS*) is 1, most of the best individuals only consider relevant a variation in intensity ($\Delta Light$) higher than ~0.1 (~%1). The probability of turn (*PTurn*) will then be set as a number different from the Base Probability of Turn (*BaseProb*). ATRP-8 is set to reason about attractants and repellents individually and later combining and averaging the preliminary probabilities of turn obtained. Therefore, the influence of each initial Probability of Turn (*PTurnAt* and *PTurnRp*) in the Resulting Probability of Turn is 50%. Table 7.5 shows the contribution of each reasoning process to the Resulting *PTurn*.

All of the best individuals in all environments evolved to have a lower value for gene 1 and higher value for gene 2, indicating that they prioritise a decrease in attractants over an increase; and an increase in repellents over a decrease. Meaning that they will begin to turn back as attractant levels decrease, returning to the centre. In the same way, they are more likely to run away as soon as an increase in repellent is detected.

In environments SR and LR the probability of turn is so certain that the value for just one stimulus is above 2, so when combined response to attractants and repellents the resultant probability of turn is still above 1, meaning it will turn.

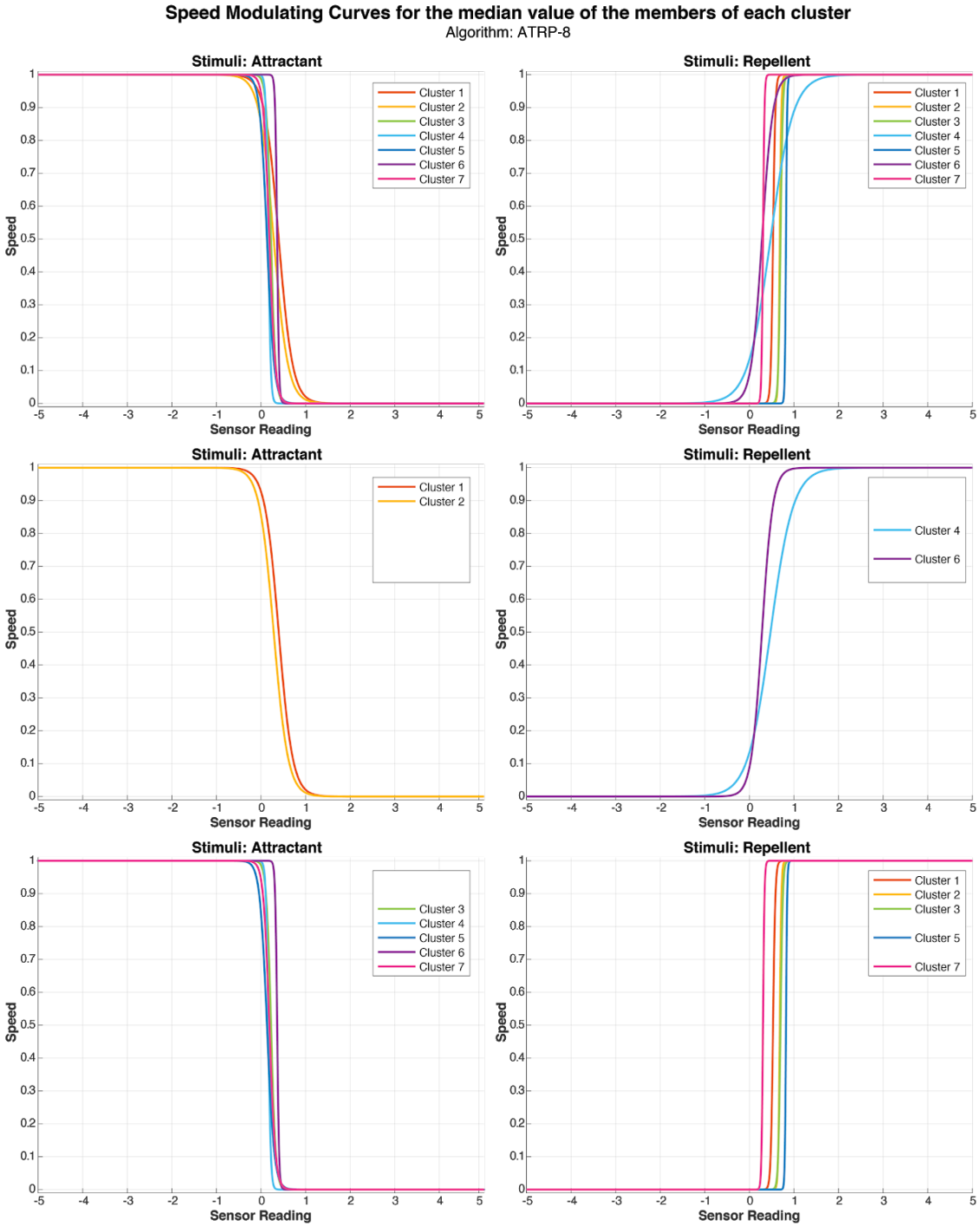


Figure 7.11 Sigmoid Function controllers related to Attractants and Repellents for individuals evolved in 9 environments (SR, MR, LR, SA, MA, LA, SP, MP, and LP), running the Foraging Algorithm ATRP-8.

Each line represents the shape of the sigmoid function for the median values (of the best 25 individuals) of genes 4 and 5 (attractants) or 7 and 8 (repellents) in each environment. The functions are always calculated using the absolute value of the stimulus and these plots show negative values only so the full length of the sigmoid can be seen. The first row shows the curves for all the environments combined and the other rows show similar curves grouped, for clarity.

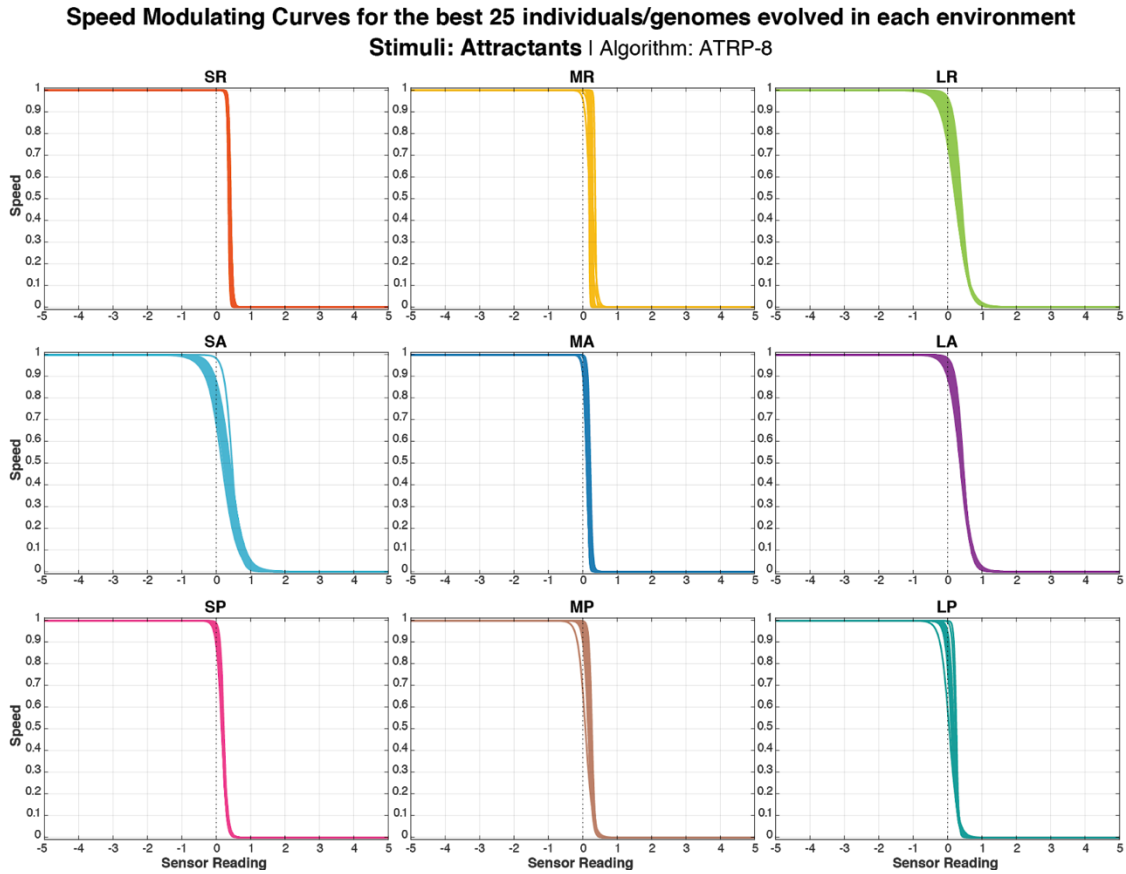


Figure 7.12 Sigmoid Function controlling the Speed Modulation related to Attractants of the best 25 individuals evolved in 9 environments (SR, MR, LR, SA, MA, LA, SP, MP, and LP), running the Foraging Algorithm ATRP-8. Each line represents one of the 25 best individuals evolved in each environment.

In environments SA SP and MP the probability is between 0.82 and 0.93, meaning that although highly likely a turn is not certain. In the remaining environments MR MA LA and LP, the probability is always above 0.5, meaning a turn is likely, but again not certain.

This also happened in AT-6, and for the following series of experiments the foraging algorithm was adapted so it would have different parameter for attractants and repellents.

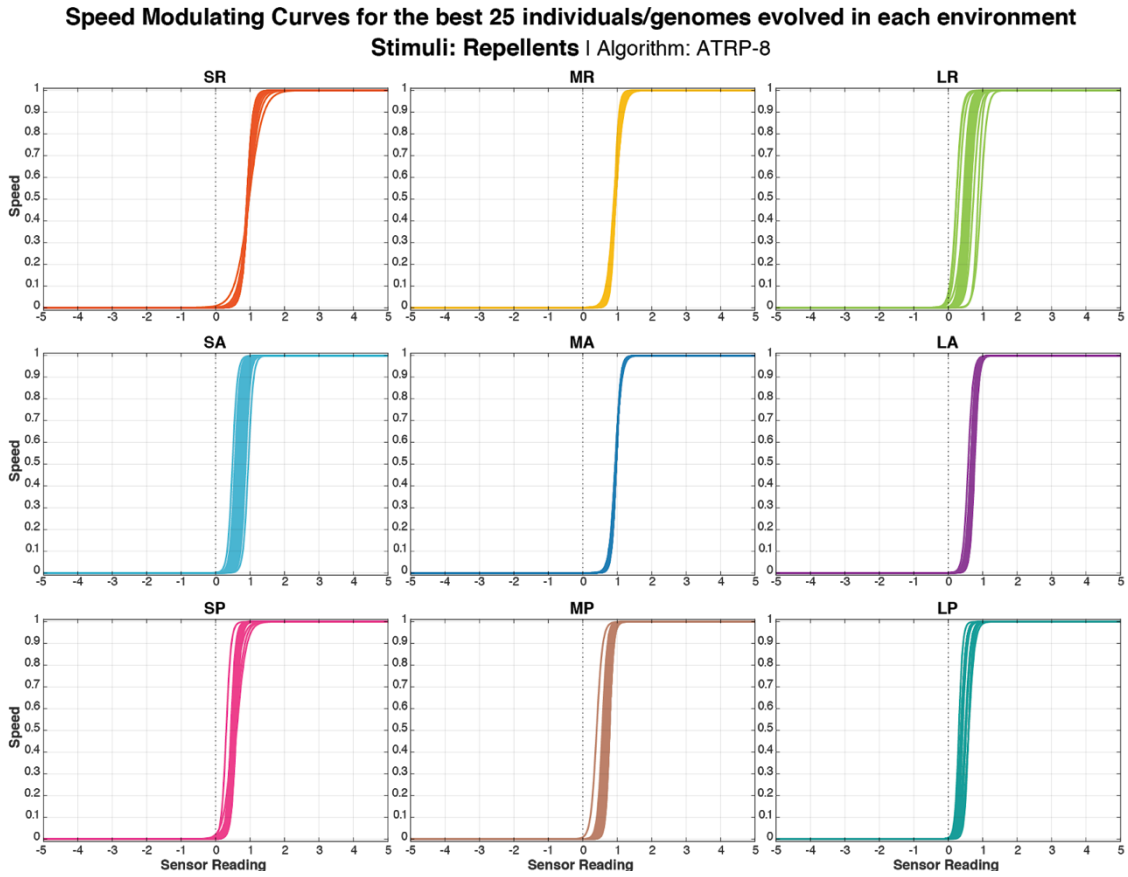


Figure 7.13 Sigmoid Function controlling the Speed Modulation related to Repellents of the best 25 individuals evolved in 9 environments (SR, MR, LR, SA, MA, LA, SP, MP, and LP), running the Foraging Algorithm ATRP-8. Each line represents one of the 25 best individuals evolved in each environment.

Table 7.5 Resulting values for variable *PTurn* calculated from the median of the genes of the best 25 individuals evolved in each environment with ATRP-8.
The value ~0.1 (relevant variation in light intensity) is calculated taking into account the standard sensor noise (1%), the value of gene 6 (*NoiseTol*), and the *Time Step Size* (TSS=4). As a high convergence for *NoiseTol* was achieved, in all the environments the reference value for comparing the *ΔIntensity* against resulted in ~0.1. Values highlighted in red show a probability of 100% (certainty) of turning.

	Contribution for the Resulting *PTurn*					
Median of individuals evolved in:	**PTurnAt / 2**			***PTurnRp / 2***		
	increase (more than ~0.1)	**decrease** (more than ~0.1)	**else**	**increase** (more than ~0.1)	**decrease** (more than ~0.1)	**else**
SR	0.000	1.181	0.010	1.181	0.000	0.010
MR	0.000	0.659	0.004	0.659	0.000	0.004
LR	0.000	1.127	0.009	1.127	0.000	0.009
SA	0.000	0.927	0.006	0.927	0.000	0.006
MA	0.000	0.612	0.005	0.612	0.000	0.005
LA	0.000	0.622	0.006	0.622	0.000	0.006
SP	0.000	0.820	0.007	0.820	0.000	0.007
MP	0.000	0.820	0.006	0.820	0.000	0.006
LP	0.000	0.518	0.006	0.518	0.000	0.006

7.3.3. Characterization of Evolved Genomes

In this section I am going to present the clustering for the best 25 individuals evolved in each environment in two sections: firstly those organised in 3 clusters, followed by those organised in 7 clusters.

7.3.3.1. 3 Clusters

This section presents the clustering of the DNA of the best individuals, the characterization of the native environments the validation, and the interpretation of the clustering results for the best 25 individuals from all 9 environments combined, grouped in 3 clusters with *k-means++* (Fig. 7.14, 7.15, 7.16, 7.17).

The first grouping of 3 clusters gives a more general overview of the similarities between individuals which evolved in different environments. A larger number of clusters would give a more detailed view of the behavioural patterns which evolved in these environments, however for comparison's sake these 3 clusters group behaviours into the pattern they most closely resemble, of course there will be more variance in the data and also outliers, but these are exceptions.

Some of the more distinctive features of this grouping are that gene 4 in the first cluster shows a higher value than the other clusters, and a lower value for gene 7. Gene 4 controls the steepness of the sigmoid which responds to attractants, whereas gene 7 controls the steepness of the sigmoid which responds to repellents. In both cases a steeper curve will

indicate a more rapid response. The members of this cluster combine a rapid response to attractants with a more gradual response to repellents.

One difference to note is that in cluster 2 when compared with the other clusters, a lower value for gene 5 can be seen, which indicates the threshold for response to attractants is lower. Hence they will respond immediately even to weak attractants. However in cluster 1 the values for gene 8 are higher than the others, which indicates a higher tolerance for repellents, meaning they will not react until a certain level of repellent is detected. Cluster 1 shows better results in richer and smaller environments, especially SR, MR, SA, MA, SP, and to some extent MP.

Cluster 3 is somewhat similar to Cluster 1 and performs well in environments that are at the same time richer and smaller. Unlike Cluster 1, it performs more favourably in larger richer environments yet the performance in poor environments is less successful. Of the three, Cluster 3 is the one that performs least favourably in poor environments.

Cluster 2 performs better than all three in poor environments, but its performance suffers in rich and average environments. While it would be expected in natural environments that species evolved with less abundant resources would perform better in both poor and rich environments, possibly by becoming more resourceful or better hunters or gatherers, we do not see this pattern with the robots. This may be due to the fact that the robots become content to settle for resources of lower intensity.

Cluster 1 performs well in smaller and richer environments, it performed reasonably well in MP and SP compared to cluster 3, indicating a more balanced behavioural pattern, however it is not as suited to large environments as cluster 3. Cluster one evolved in SR MR MA and SA, so that it finds larger environments more challenging is to be expected.

Cluster 1 evolved in rich environments yet also performs well in poor environments, this pattern would not be expected, although from the perspective of 'practice makes perfect' this make sense, a species with more experience finding attractants will develop better strategies.

In contrast Cluster 2, which evolved in poor environments, does best in poor environments, which is to be expected as this would be considered its 'natural habitat', but its performance suffers in rich environments. This could be indicative of less experience in finding attractants purely because there were less attractants to find. Due to the nature of the optimisation many individuals would die in poor environments before having the opportunity to evolve, as these environments are less forgiving.

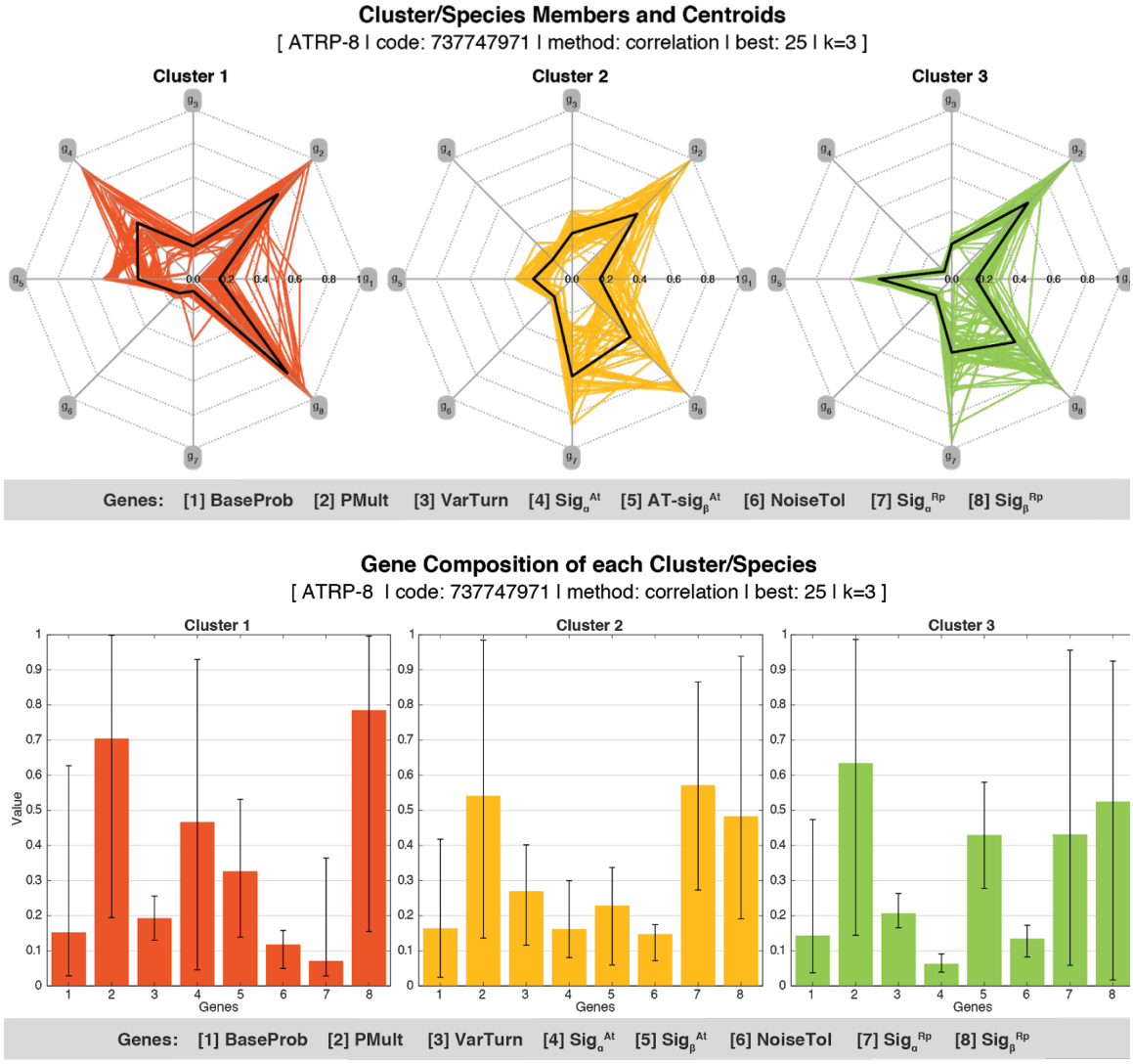


Figure 7.14 Clustering results of the best 25 from each environment (a total of 225), grouped in 3 clusters using *k-means++* and correlation.

Top row: Each subplot (spider plot) shows the cluster members and centroids. Each axis represents one input parameter and the values are distributed centrewise, from 0 to 1. Cluster members are plotted in coloured lines, and the centroid is plotted in black. Bottom row: Each subplot shows the DNA composition of the members of each cluster, and the respective standard deviation.

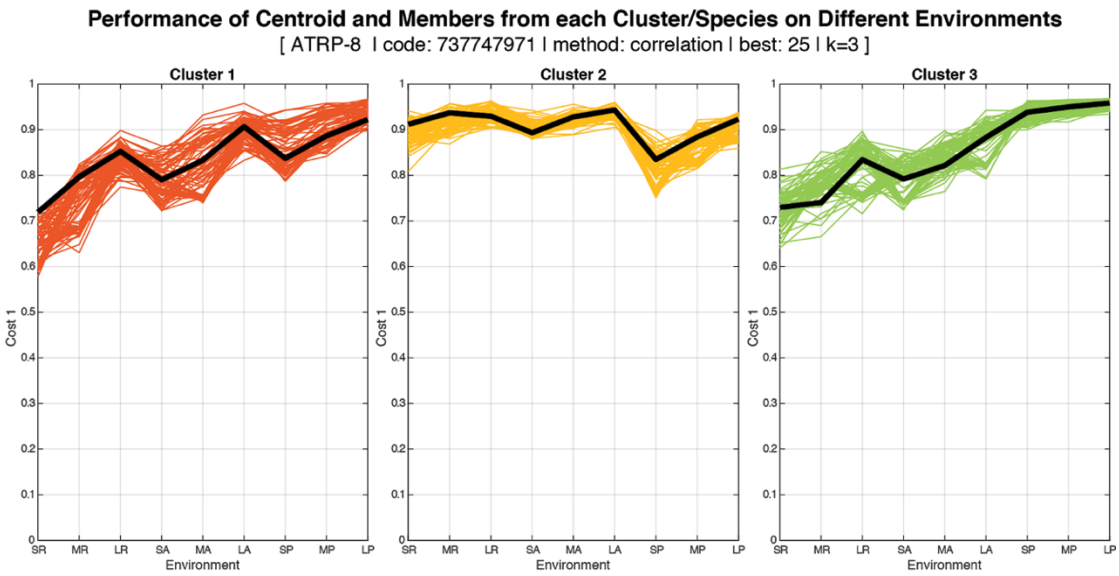


Figure 7.15 Clustering results of the best 25 from each environment (a total of 225), grouped in 3 clusters using *k-means++* and correlation (Series I).

Each subplot shows the cluster members and centroids. Each vertical axis represents the value of Cost1 in one environment. Cost 1 of cluster members are plotted in coloured lines, and Cost 1 of the centroid is plotted in black.

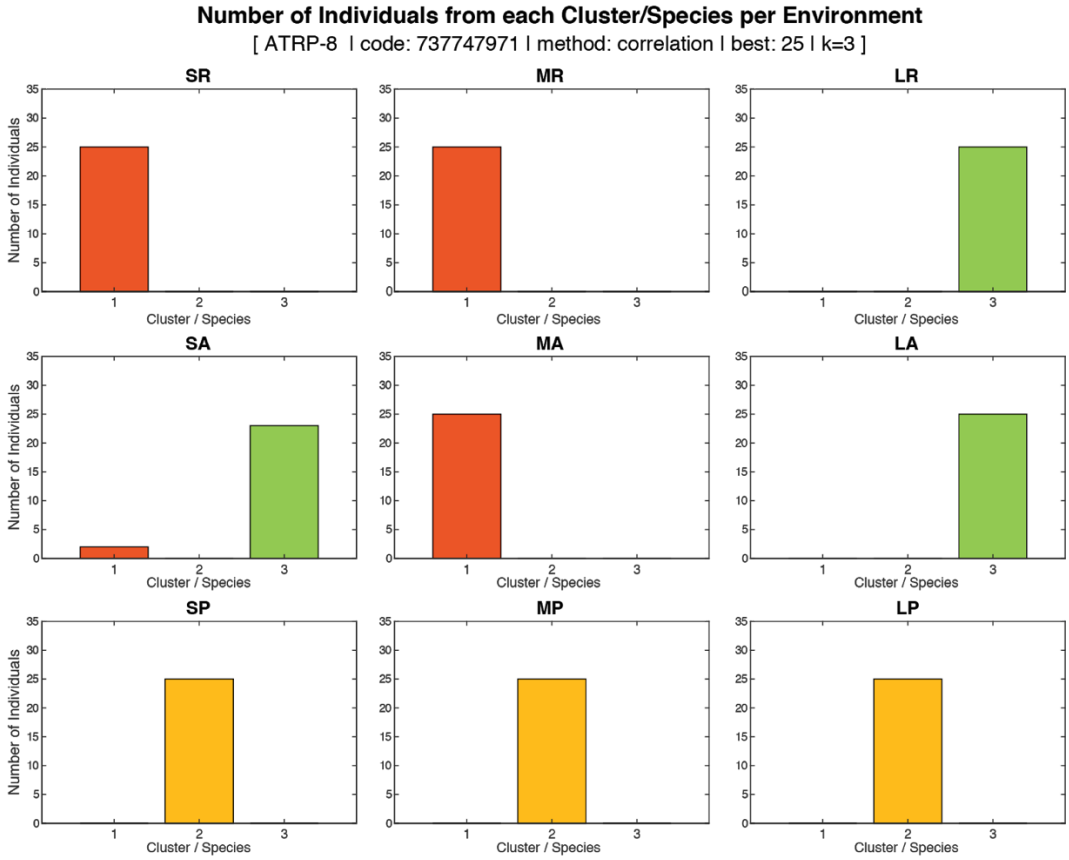


Figure 7.16 Clustering results of the best 25 from each environment (a total of 225), grouped in 3 clusters using *k-means++* and correlation (Series I).

Each subplot shows the presence of members from different clusters in each environment.

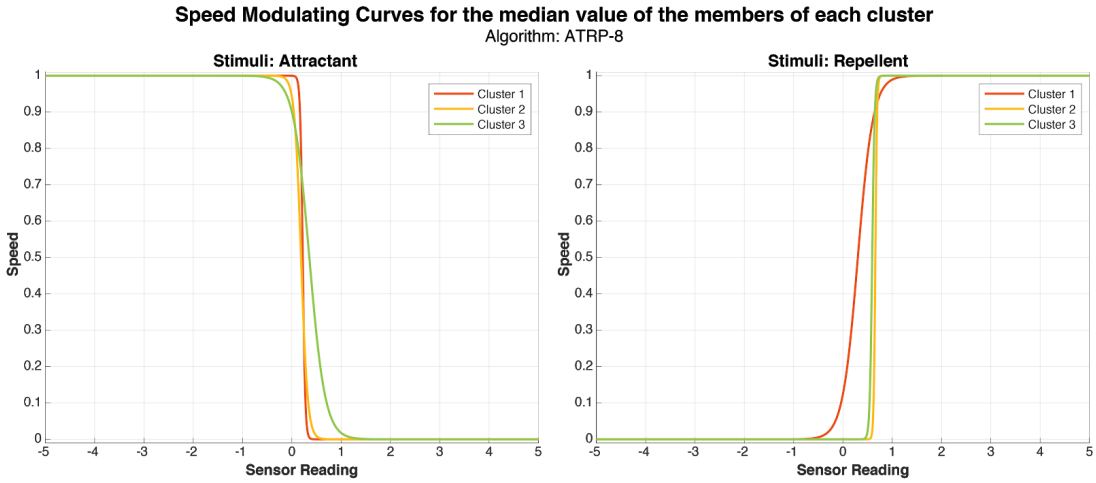


Figure 7.17 Sigmoid Function controllers related to Attractants and Repellents for each cluster in a scenario with 3 clusters.

Each line represents the shape of the sigmoid function for the median values of genes 4 and 5 (attractants) or 7 and 8 (repellents), for all the cluster members. The functions are always calculated using the absolute value of the stimulus and these plots show negative values only so the full length of the sigmoid can be seen.

7.3.3.2. 7 Clusters

This section presents the clustering of the DNA of the best individuals, the characterization of the native environments the validation, and the interpretation of the clustering results for the best 25 individuals from all 9 environments combined, grouped in 7 clusters with *k-means++* (Fig. 7.18, 7.19, 7.20, 7.21, 7.22).

The second grouping to be discussed is that of 7 clusters. This grouping provides a more detailed overview of the different strategies evolved in each environment. One reason for clustering this data is to show the links between the genes, for example g_4 (Sig_{α}^{At}) and g_5 (Sig_{β}^{At}) which work as a pair to control the sigmoid function to respond to attractants. Without this clustering it would not be as clear as to which genes were affecting the behaviours, and which genes evolved together.

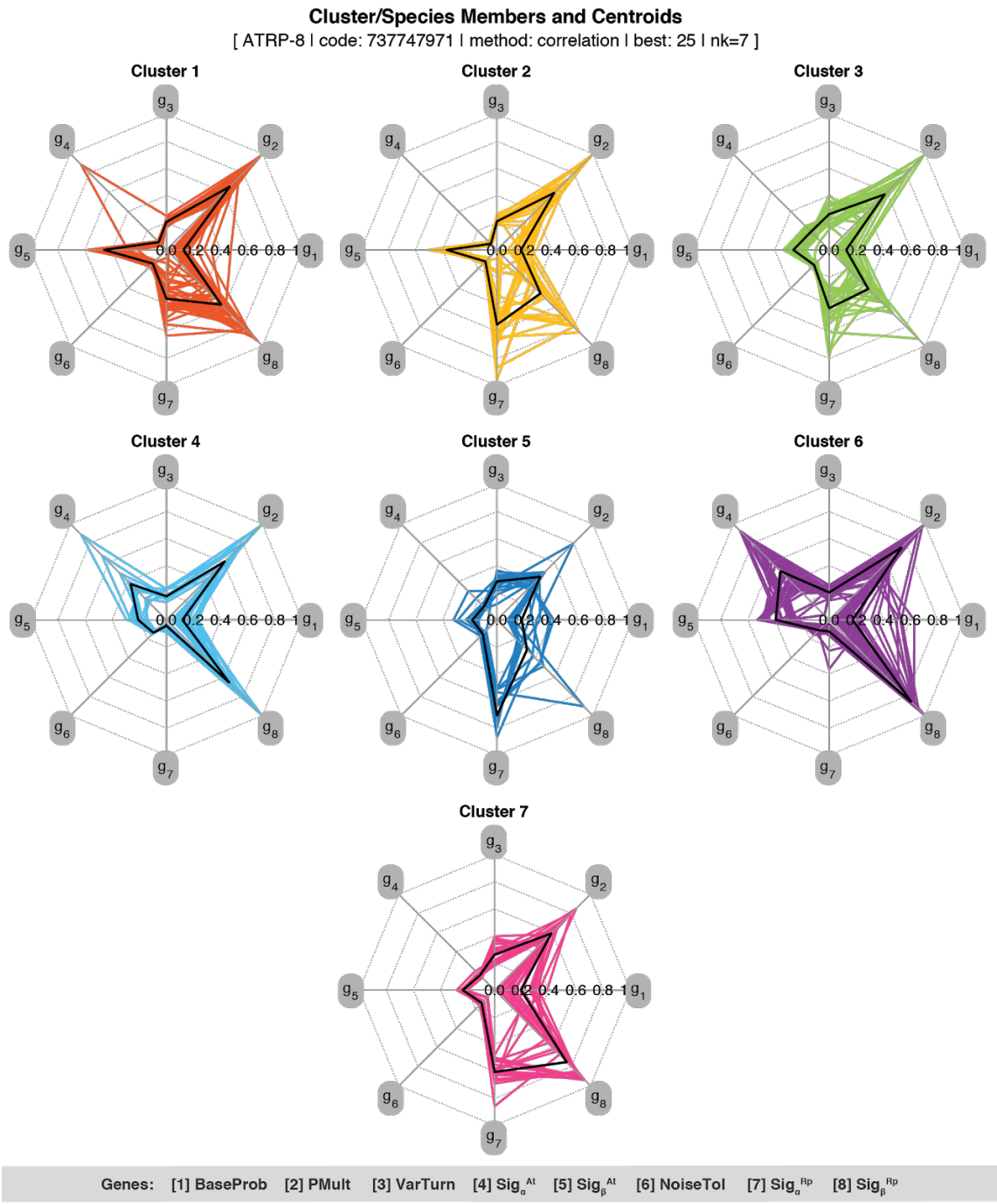


Figure 7.18 Clustering results of the best 25 from each environment (a total of 225), grouped in 7 clusters using *k-means++* and correlation (Series I).

Each subplot (spider plot) shows the cluster members and centroids. Each axis represents one input parameter and the values are distributed centrewise, from 0 to 1. Cluster members are plotted in coloured lines, and the centroid is plotted in black.

Species 1, 2 and 7 have low values of g4 (Sig_α^{At}), meaning a gradual response to attractants. Whereas species 4 and 6 have low values of g$_7$ (Sig_α^{Rp}), meaning a gradual response to repellents.

Species 1, 2, 3, 5, and 7 have higher values of g_7, resulting in a steeper curve, meaning a rapid response to repellents, with varying thresholds: Species 7, then 1, then 2 and 3, then 5 (Figure 7.23).

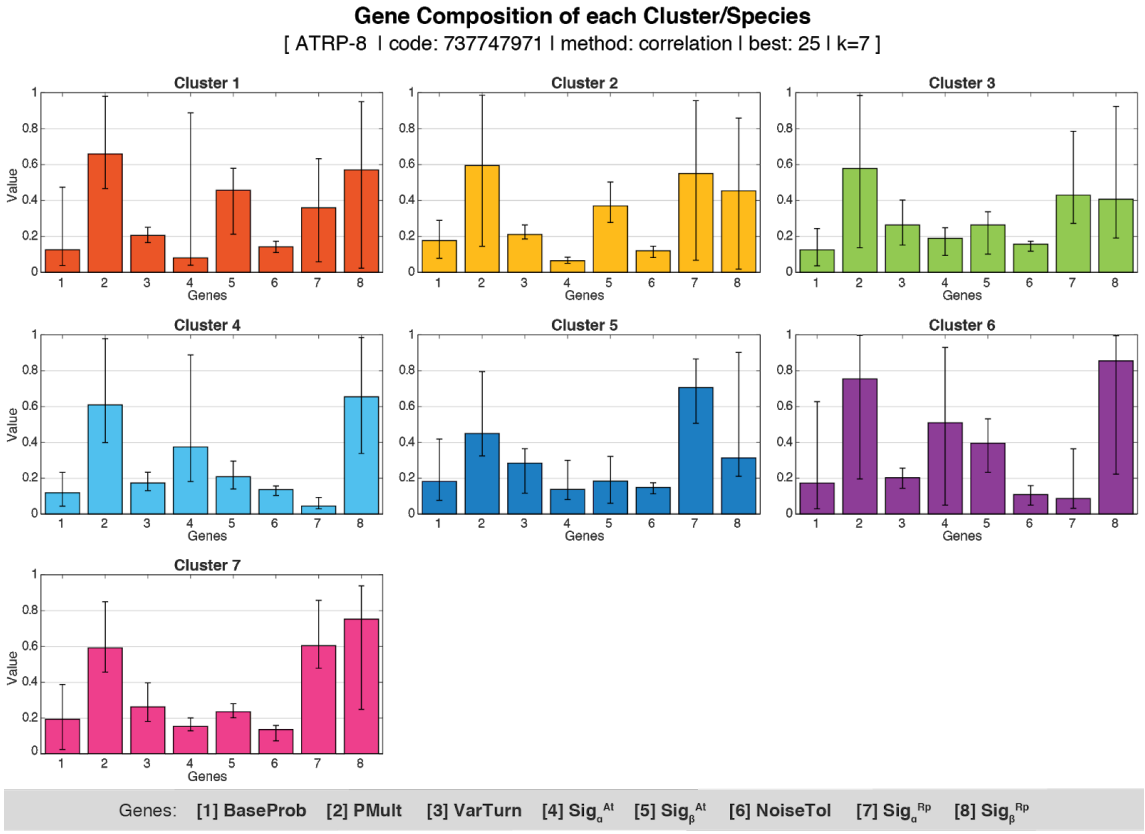


Figure 7.19 - Clustering results of the best 25 from each environment (a total of 225), grouped in 7 clusters using *k-means++* and correlation (Series I).

Each subplot shows the DNA composition of the members of each cluster and the respective standard deviation.

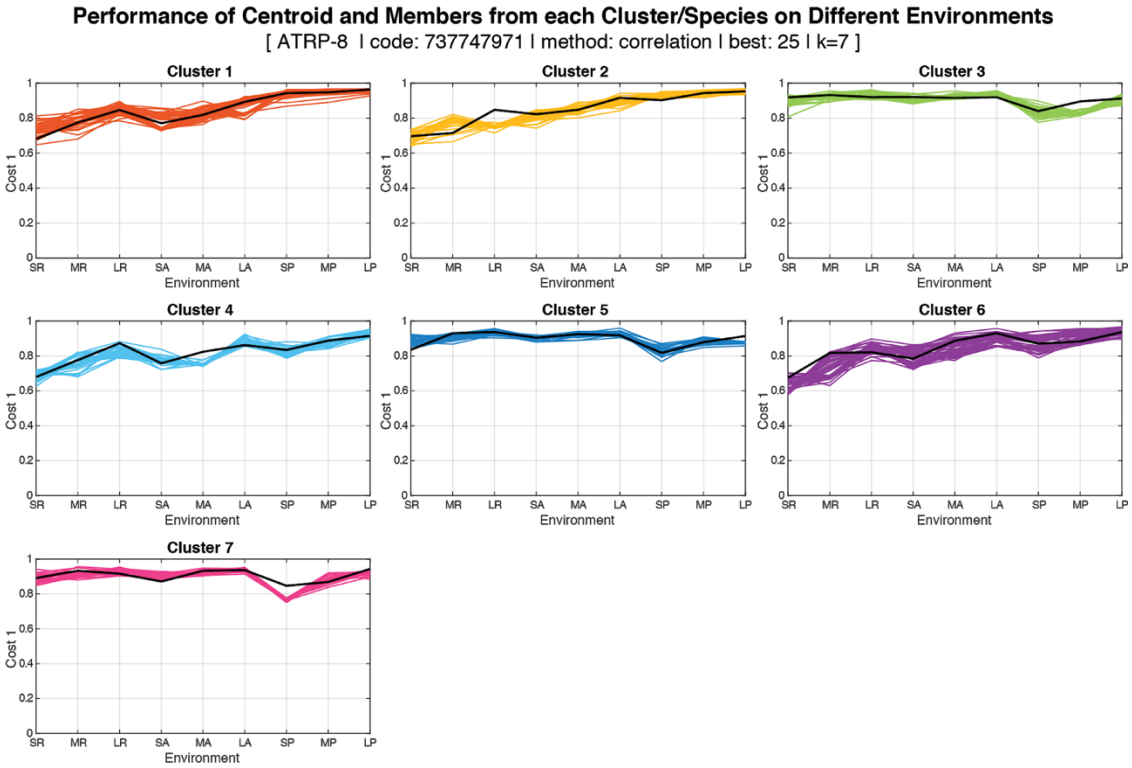


Figure 7.20 Clustering results of the best 25 from each environment (a total of 225), grouped in 7 clusters using *k-means++* and correlation (Series I).

Each subplot shows the cluster members and centroids. Each vertical axis represents the value of Cost1 in one environment. Cost 1 of cluster members are plotted in coloured lines, and Cost 1 of the centroid is plotted in black.

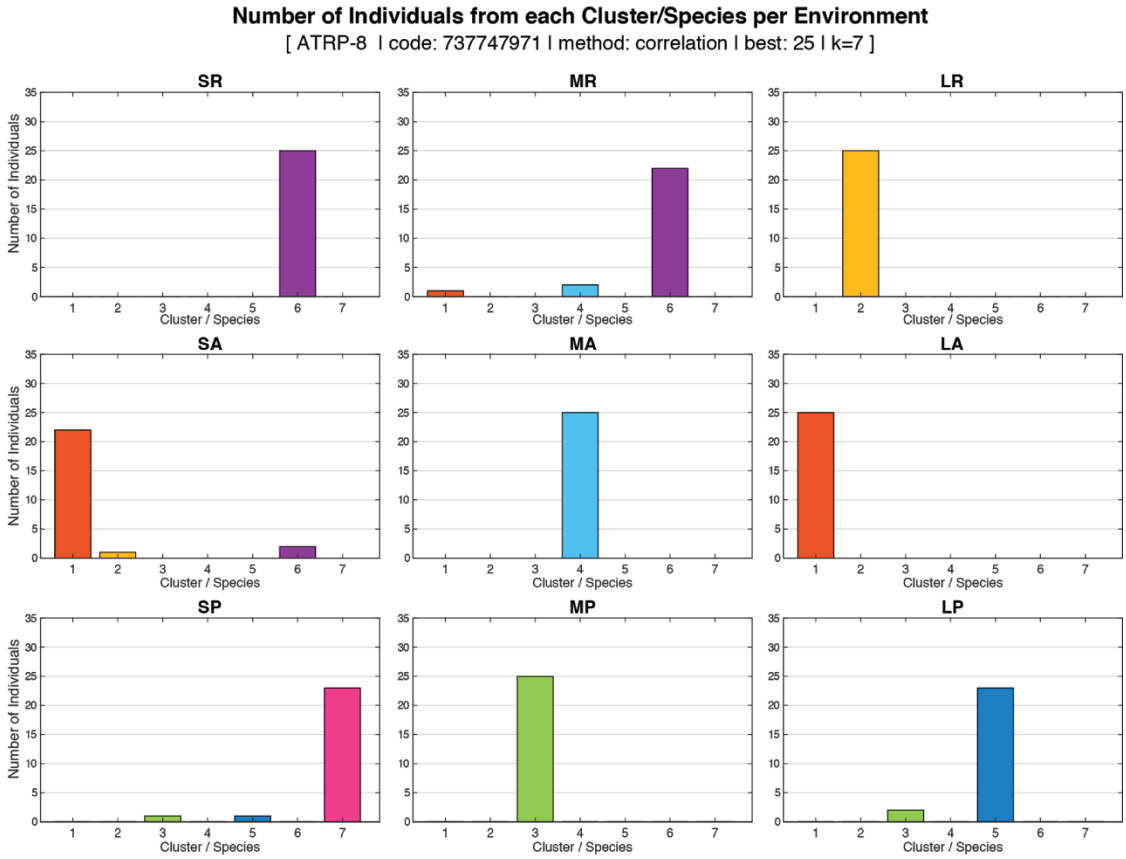


Figure 7.21 - Clustering results of the best 25 from each environment (a total of 225), grouped in 7 clusters using *k-means++* and correlation (Series I).

Each subplot shows the presence of members from different clusters in each environment.

Members of Cluster 6 are mainly native of SR and MR. Members of Cluster 1 are mainly native of SA and LA. Meaning that sometimes in similar environments similar strategies evolved. It would be expected that SA and MA would be more similar or even MA and LA, however this was not the case. The species evolved in SA is more similar to the one evolved in LA than that evolved in MA. It is not unusual that in nature different species will evolve similar features even in vastly different locations, for example as Walruses and Manatees evolved to have a very similar body shape for swimming and similar features, despite being genetically very different, a Walrus's is closer to that of a Wolf and a Manatee's is closer to that of an elephant, yet they are physically very similar despite evolving in vastly different locations.

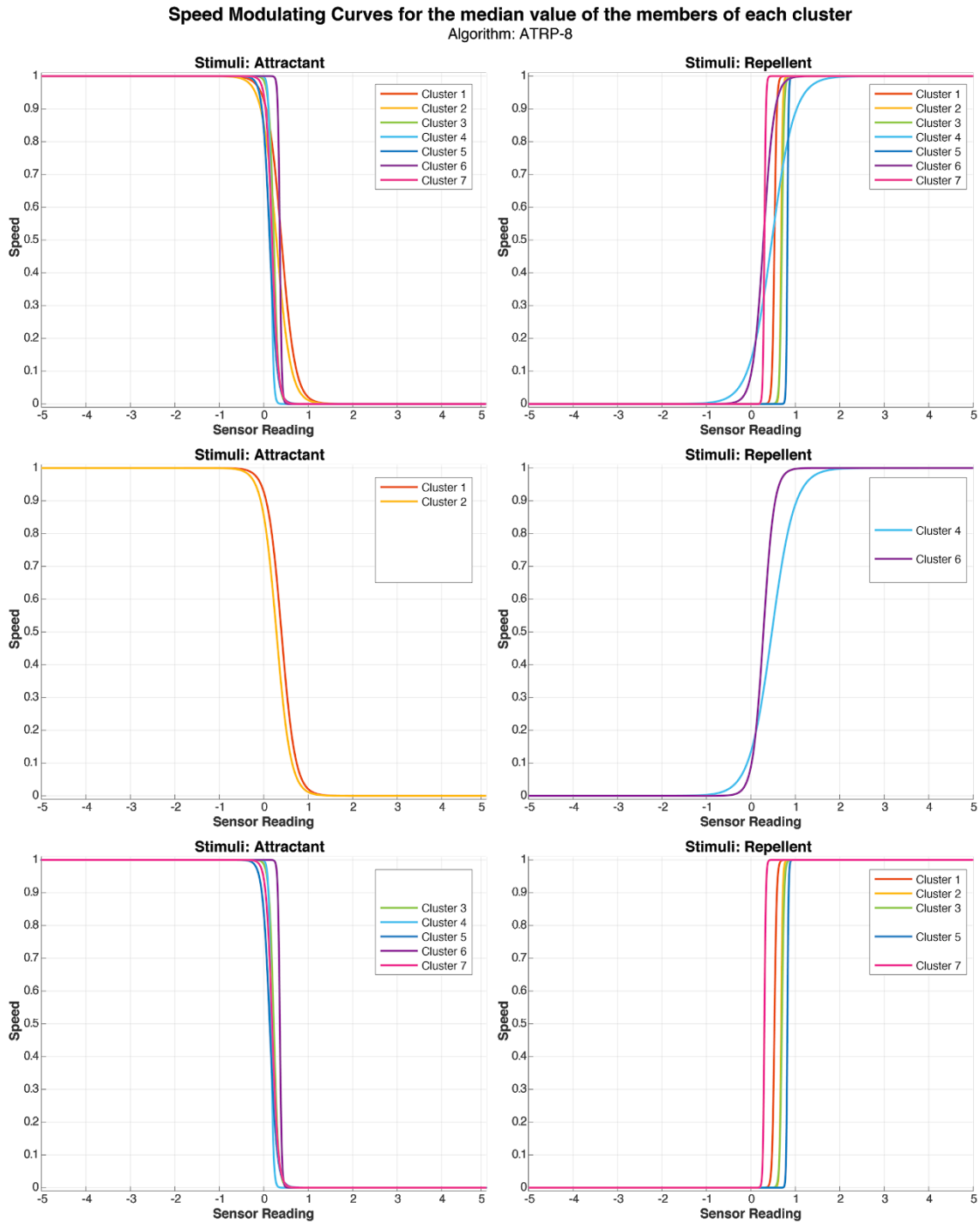


Figure 7.23 - Sigmoid Function controllers related to Attractants and Repellents for each cluster in a scenario with 7 clusters (Series I).

Each line represents the shape of the sigmoid function for the median values of genes 4 and 5 (attractants) or 7 and 8 (repellents), for all the cluster members. The functions are always calculated using the absolute value of the stimulus and these plots show negative values only so the full length of the sigmoid can be seen. The first row shows the curves for all the environments combined and the other rows show similar curves grouped, for clarity.

7.4. Conclusions

Within the 9 environments many different strategies evolved. Furthermore, the genetic analysis seemed to capture some of the most relevant aspects of the individuals evolved. To maintain clarity, the number of species was limited to 7. By forming 7 clusters the similarities between strategies and environments became more apparent.

However, in some cases (Cluster 2 in LR, Cluster 3 in MP, Cluster 4 in MA, Cluster 5 in LP, Cluster 6 in MR, and Cluster 7 in SP), the performance of the centroid differed to that of the members of the cluster (when tested in all environments, as seen in Figure 7.20). This means that each cluster may contain more than one species, which appear similar.

This algorithm was successful in keeping a significant portion of the group of robots alive to the end of the simulation. However, one small issue which was encountered in this series of this algorithm was that the robots running it must use the same value for *PMult* to respond to both attractants and repellents, in future experiments it was decided that they would have different parameters regarding the probability of turn for attractants and and another for repellents. Giving them the opportunity to respond differently to attractants and repellents.

Chapter 8
Second Refined Algorithm: ATRP-7 (Experiment Series J)

In this chapter, I will present the design of the third refined C. elegans-based Foraging Algorithm, ATRP-7, and the results of experimental series I. The Foraging Algorithm takes 7 input parameters which were optimised in 4 environments: MA11, MA51, MA55, and MA15 (Table 8.1). These environments are variations of MA, used in experiment series I (Chapter 7), and have different values for combined *Light Index* values of attractants and repellents (Light Index AT+RP), as well as different ratios between them (Light Index AT/RP), as seen in Table 8.2.

Table 8.1 Compilation of the experiments of Series J, evolved in 4 different environments (MA51, MA11, MA55, and MA15), with attractants and repellents, for robots equipped with sensors with sensor noise of value 1%.

OPT.		SIM.	ROBOT	ENVIRONMENT		Function Eval.	Minim. Cost 1	Minim. Cost 2	Minim. Cost 3	Non-Domin. Solutions
EA	Pool Size	Time Step	Sensor Noise	Field Size & Quality	Subtype					
DE	200	4	1%	MA (225-avg)	MA51	13,635	0.553	0.991	0.960	15
					MA11	14,537	0.659	0.993	0.979	13
					MA55	16,962	0.807	0.995	0.984	14
					MA15	7,144	0.904	1.000	0.982	11
					TOTAL:	52,278				

Table 8.2 Minimised Costs and Light Indexes for the 4 environment subtypes (MA51, MA11, MA55, and MA15), used in Experiment Series J.

ENV. Subtype	Intensity of Attract.	Intensity of Repell.	Minim. Cost 1	Minim. Cost 2	Minim. Cost 3	Light Index (AT)	Light Index (RP)	Light Index (AT+RP)	Light Index (AT/RP)
MA51	1x	1x	0.553	0.991	0.960	50.152	-27.476	22.676	-1.825
MA11	5x	1x	0.659	0.993	0.979	23.396	-27.476	-4.080	-0.852
MA55	5x	5x	0.807	0.995	0.984	50.152	-65.001	-14.850	-0.772
MA15	1x	5x	0.904	1.000	0.982	23.396	-65.001	-41.606	-0.360

8.1. Algorithm Design

The foraging algorithm ATRP-7, inspired by *C. elegans*' chemotaxis, improves and simplifies the foraging algorithm presented in the previous chapter (Chapter 7). This algorithm is also composed of two key behaviours: *runs* and *turns*, controlled by a set of seven parameters ($g_{1:7}$), also referred to in this work as the 'genome', shown in Table 8.3.

This algorithm encompasses both positive and negative chemotactic behaviours, as the agents react to attractant and repellent stimuli distributed in the field.

ATRP-7 is an upgrade of the foraging algorithms developed earlier in this work and there are two main differences. The first difference is that the probability multiplier for attractants and repellents is set individually by two separate input parameters (genes). The second difference is that the two of the input parameters controlling the variability of the angle of the turns (*VarAngle*) and the noise tolerance (*NoiseTol*) have been suppressed, and the values have been fixed to the optimised values found for AT- previous experiments. As in previous experiments (Series G, H, and I), a strong convergence could be observed in parameters *VarAngle* and *NoiseTol*, these two "genes" have been suppressed in this version of the foraging algorithm. The values were fixed at the average parameter value obtained by the best individuals from the previous series of experiments ($VarAngle = 0.20124$ and $NoiseTol = 0.1376$).

Table 8.3 - Set of input parameters (genome) for the *C. elegans*' bio-inspired minimalist algorithm for positive and negative chemotaxis in Foraging Algorithm ATRP-7.

GENE	PARAMETER	PROCESS	APPLICATION
g_1	$BaseProb$	Positive and Negative Chemotaxis (both)	sets the base probability of turning, in the absence of any change in sensed light level
g_2	$PMult^{At}$	Positive Chemotaxis (Attractants)	sets the multiplier (divisor) of the base turning probability when the level of attractant decreases (increases)
g_3	Sig_{α}^{At}		sets the steepness of the sigmoid curve that controls speed according to the sensor reading for attractants.
g_4	Sig_{β}^{At}		defines the offset of the sigmoid curve that controls speed according to the sensor reading for attractants.
g_5	$PMult^{Rp}$	Negative Chemotaxis (Repellents)	sets the divisor (multiplier) of the base turning probability when the level of repellent increases (decreases)
g_6	Sig_{α}^{Rp}		sets the steepness of the sigmoid curve that controls speed according to the sensor reading for repellents.
g_7	Sig_{β}^{Rp}		defines the offset of the sigmoid curve that controls speed according to the sensor reading for repellents.

The method for updating the battery level of each robot is the same as described in Chapter 7, in Equations [7.1] and [7.2].

Also similarly to the previous experiments, the simulation program updates the robots' positions, checks for extinguished light spots (replacing them with new ones if necessary) and checks which robots are 'alive' – a robot permanently 'dies' if its battery is depleted.

The reasoning process on each time step (t) starts when the agent acquires the sensor readings for the intensity of Attractants ($SensAt_t^R$) and Repellents ($SensRp_t^R$) at its current position [x,y]. The methods for obtaining sensor reading and for the program to add sensor noise are the same as described in Chapter 7. $\Delta SensAt_t^R$ and $\Delta SensRp_t^R$ are then

obtained in order to calculate the probability of initiating a turn ($PTurn_t^R$) according to Equations 7.3, 7.4, 7.5, and 7.6.

As opposed to the previous algorithm (ATRP-8), in ATRP-7 the two probabilities of turn ($PTurnAt_t^R$ and $PTurnRp_t^R$) are calculated independently with each using a different Probability Multiplier ($PMult^{At}$ *and* $PMult^{Rp}$), related specifically to attractants or repellents, as described in the following pseudocode:

for Attractant Stimulus:

```
1  if (ΔSensAt_t^R > SensNoise × NoiseTol) then
2      PTurnAt_t^R = BaseProb ÷ PMult^At
3  else if (ΔSensAt_t^R < SensNoise × NoiseTol) then
4      PTurnAt_t^R = BaseProb × PMult^At
5  else
6      PTurnAt_t^R = BaseProb
7  end
```

for Repellent Stimulus:

```
1  if (ΔSensRp_t^R > SensNoise × NoiseTol) then
2      PTurnRp_t^R = BaseProb × PMult^Rp
3  else if (ΔSensRp_t^R < SensNoise × NoiseTol) then
4      PTurnRp_t^R = BaseProb ÷ PMult^Rp
5  else
6      PTurnRp_t^R = BaseProb
7  end
```

The final $PTurn_t^R$ is rationalised by averaging $PTurnAt_t^R$ and $PTurnRp_t^R$. As in AT-6 and ATRP-8 (chapters 8 and 9), once $PTurn_t^R$ is set, a random number (0 to 1) is generated. If the random number is less than or equal to $PTurn_t^R$, the robot will perform a turn (Fig.10.1). When performing a turn, the yaw ($\Delta\theta$) will be calculated using another random number (between -1 and 1), according to equation [8.7].

Similar to algorithms AT-6 and ATRP-8 (Chapters 6 and 7), *Speed* is calculated by averaging the results of two Speed Modulators - calculated separately for Attractant and Repellent stimuli, each using its specific set of parameters, as seen in Table 8.3.

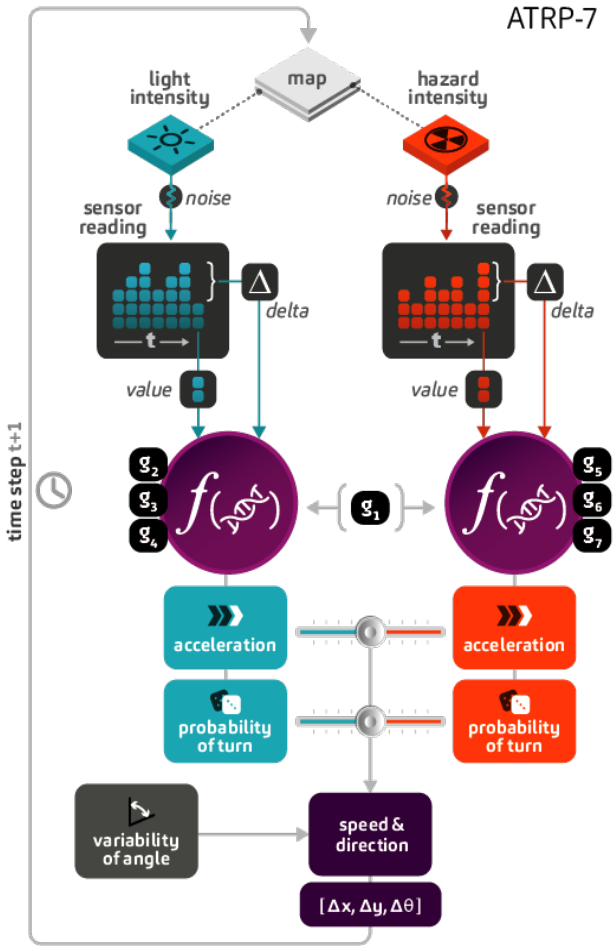


Figure 8.1 Flowchart of the control algorithm ATRP-7.

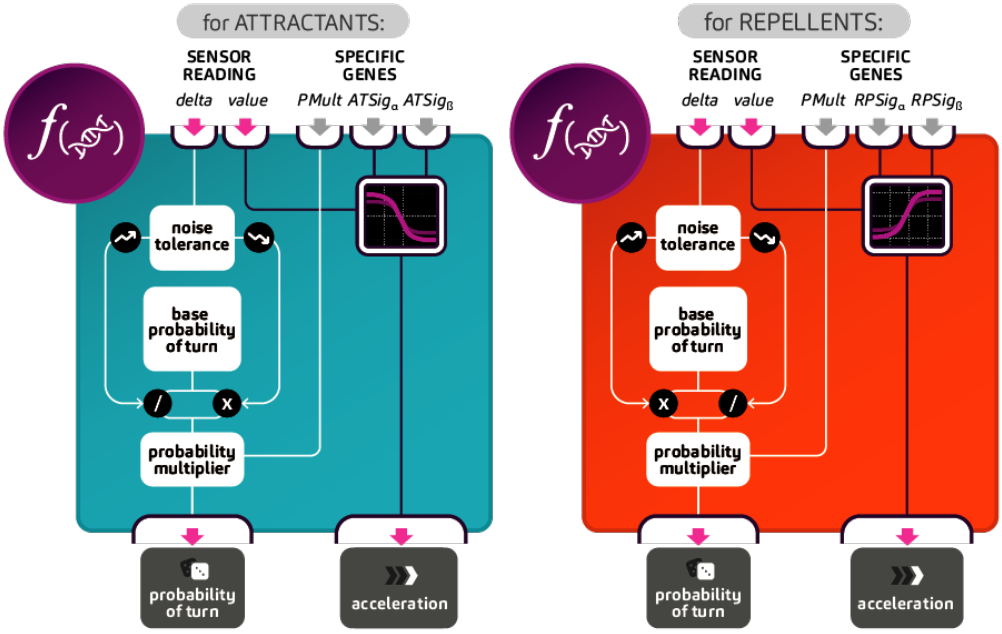


Figure 8.2 Diagram of the function outputting the probability of turn and acceleration. The same function is used for both attractants and repellents.

8.2. Simulation and Experiment Design

The algorithm was evolved for 4 different environments of Medium Size populated with attractants and repellents in different proportions, as described in Section 5.2.

8.3. Results and Discussion

In this set of optimisation cycles for ATRP-7, virtual robots running this foraging algorithm were optimised for 4 environments of the same size and containing the same number of attractants and repellents, but in different proportions. The best solutions found were grouped into 3 and 7 clusters by genetic similarity. The clusters were validated by comparing the results of its members on the cross-environment trials (cross-trials).

8.3.1. Parameter Optimisation for Different Environments

Optimisation in MA15 reached the stopping criteria after 40 generations, whilst in the other environments, the optimisations ran for longer than 70 generations (only being interrupted due to time restrictions). The best values of Cost 1 were found in MA51, MA11, MA55, and MA15, respectively (Table 8.1, Figure 8.3). The same sequence was found for Cost 2 (Figure 8.4). These results hold a strong correlation with both Combined and Ratio *Light Index* values (Table 8.2).

As it has been mentioned earlier in this chapter, the input parameters were not optimised for Cost 3, even though the value of it was registered. The best values found (but not optimised) for Cost 3 are, respectively: MA51, MA11, MA15, and MA55 (Table 8.1, Figure 8.5). Compared to the results found for Costs 1 and 2 and to the values of the Light Index, it was expected that MA15 and MA55 would be inverted. It is possible that the robots in MA15 had to explore more of the field before finding attractant spots and this behaviour might have led to the values obtained.

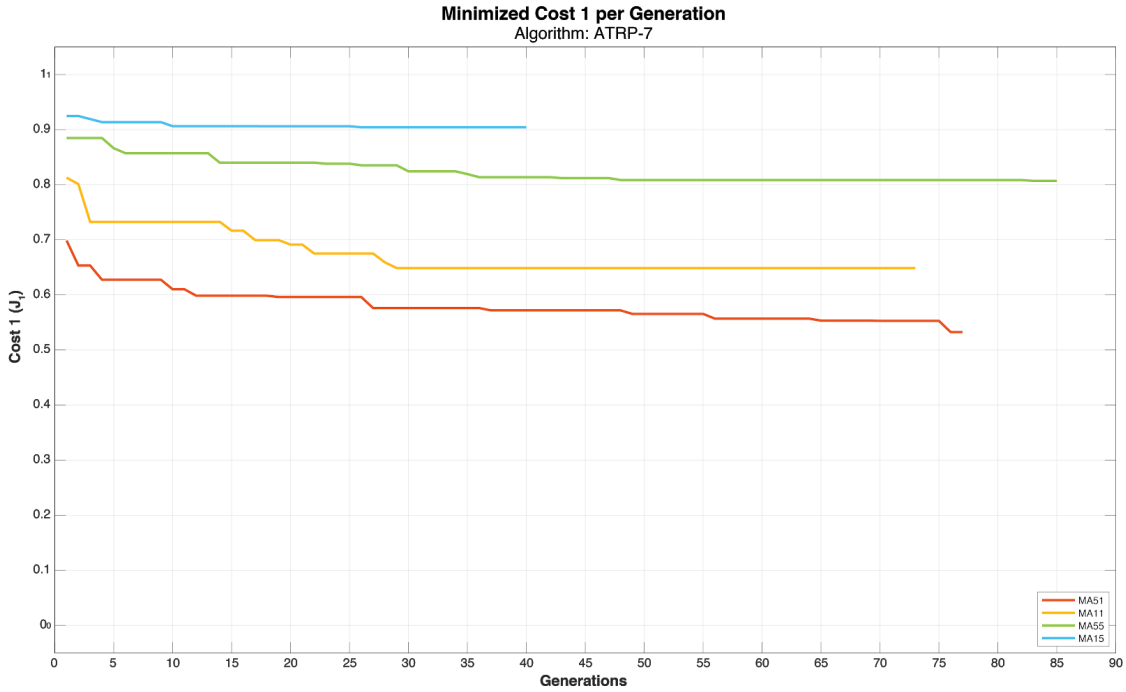


Figure 8.3 Optimisation of Cost 1 (J_1) in 4 environments (MA51, MA11, MA55, and MA15) with attractants and repellents in varying intensity ratios (Series J). MA environments have Field Size 225 and Average resource quality. Solutions were optimised with DE using a Pool Size of 200.

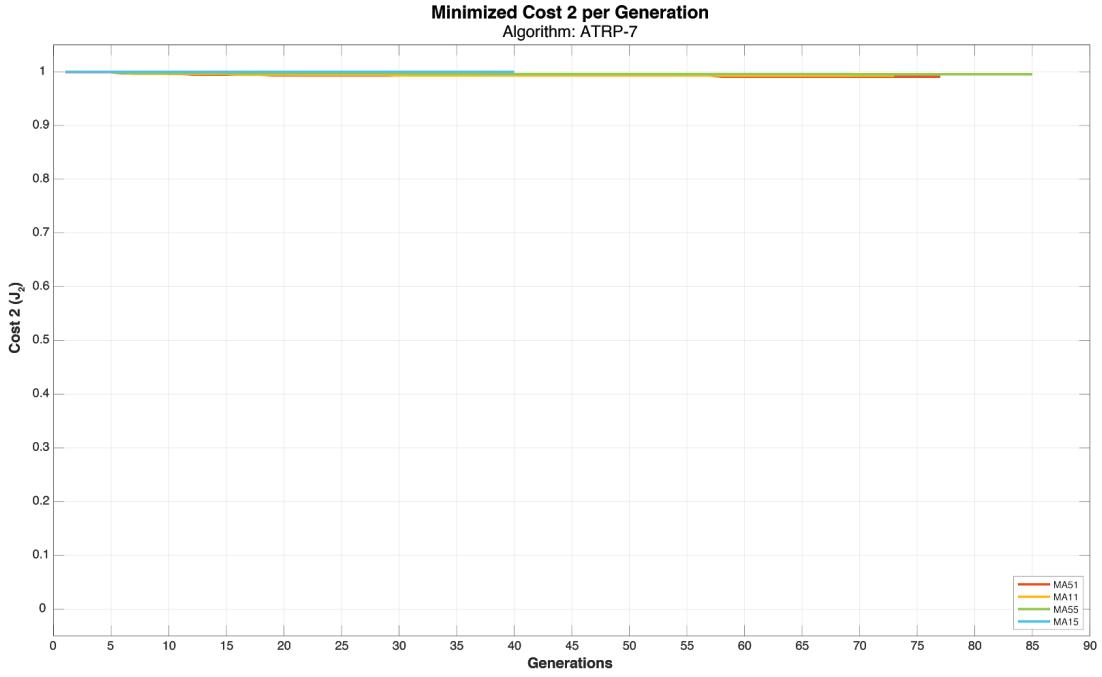


Figure 8.4 Optimisation of Cost 2 (J_2) in 4 environments (MA51, MA11, MA55, and MA15) with attractants and repellents in varying intensity ratios. MA environments have Field Size 225 and Average resource quality. Solutions were optimised with DE using a Pool Size of 200.

8.3.2. Selection of the best individuals

Figure 8.5 shows the best 7000 unique individuals evolved in the 4 environments using Foraging Algorithm ATRP-7. As can be seen, Cost 1 presents a steep drop in approximately the best 100 individuals (Figure 8.5).

Further experiments explored the variance of Cost 1 in different aliquots of the population and values around 25 individuals presented a good combination between diversity (gene variance) whilst still retaining a narrow range of Cost 1 values for all environments (Fig. 8.7, 8.8, 8.9, and 8.10). This number was also maintained as it matched the number of best individuals chosen in the previous experiment series (Series H and Series I).

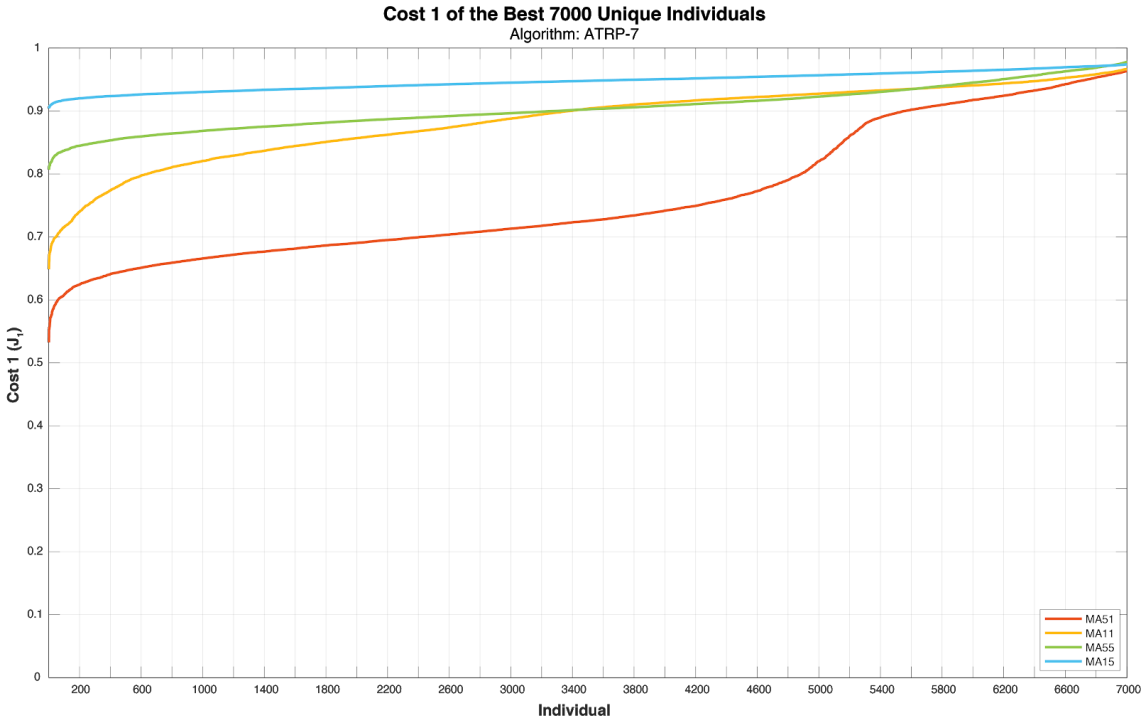


Figure 8.5 Cost 1 (J_1) of the best 7000 individuals running the behavioural algorithm ATRP-7, evolved in 4 environments (MA51, MA11, MA55, and MA15) with attractants and repellents in varying intensity ratios.

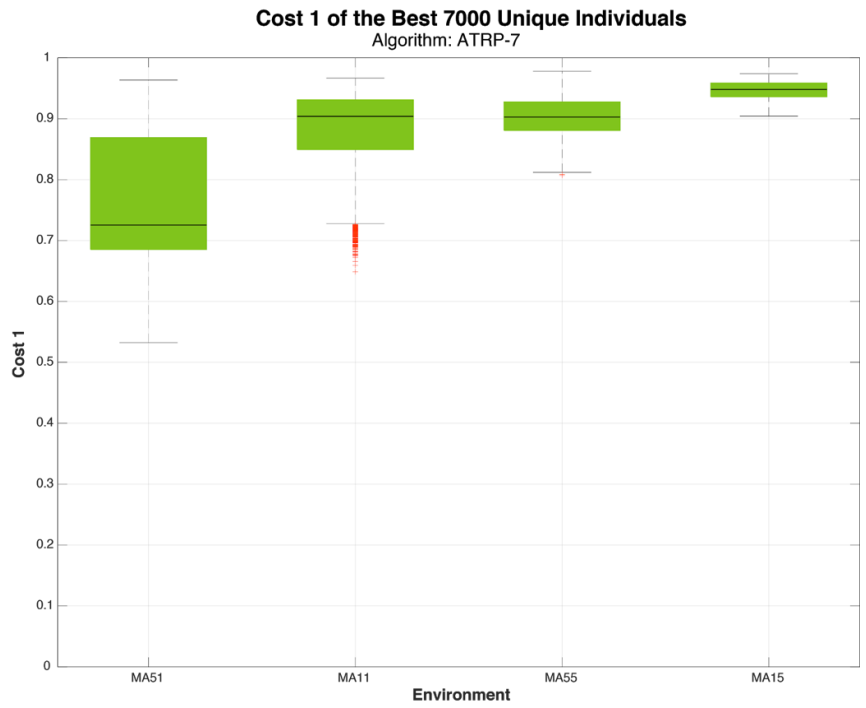


Figure 8.7 Box plot of the Cost 1 (J_1) of the best 7000 individuals evolved running the behavioural algorithm ATRP-7 in 4 environments (MA51, MA11, MA55, and MA15) with attractants and repellents in varying intensity ratios.

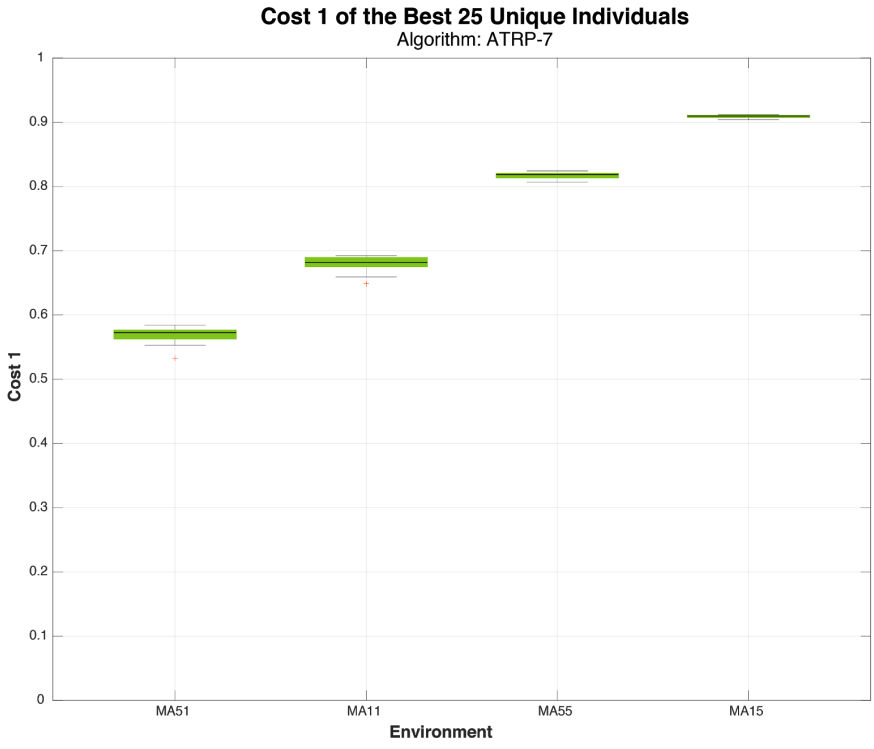


Figure 8.8 Box plot of the Cost 1 (J_1) of the best 25 individuals evolved running the behavioural algorithm ATRP-7 in 4 environments (MA51, MA11, MA55, and MA15) with attractants and repellents in varying intensity ratios.

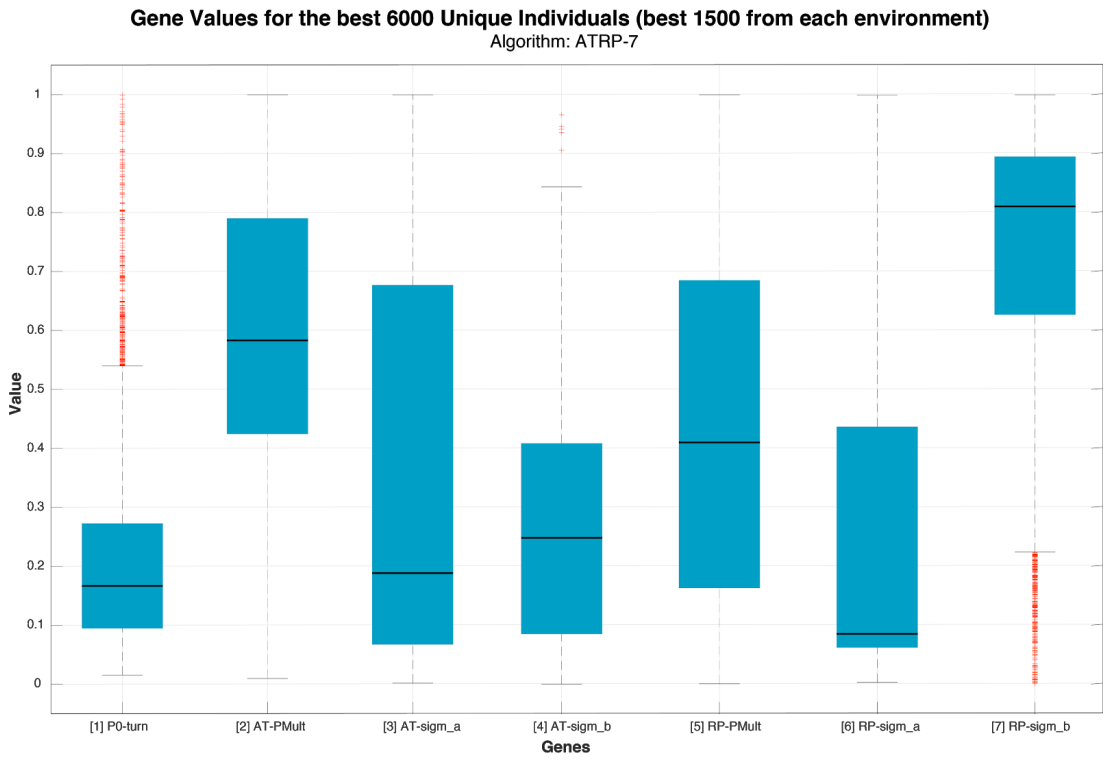


Figure 8.9 Box plot of the values for the input parameters (genes) of the best 6000 individuals (1500 from each environment) evolved in MA51, MA11, MA55, and MA15, running the Foraging Algorithm ATRP-7.

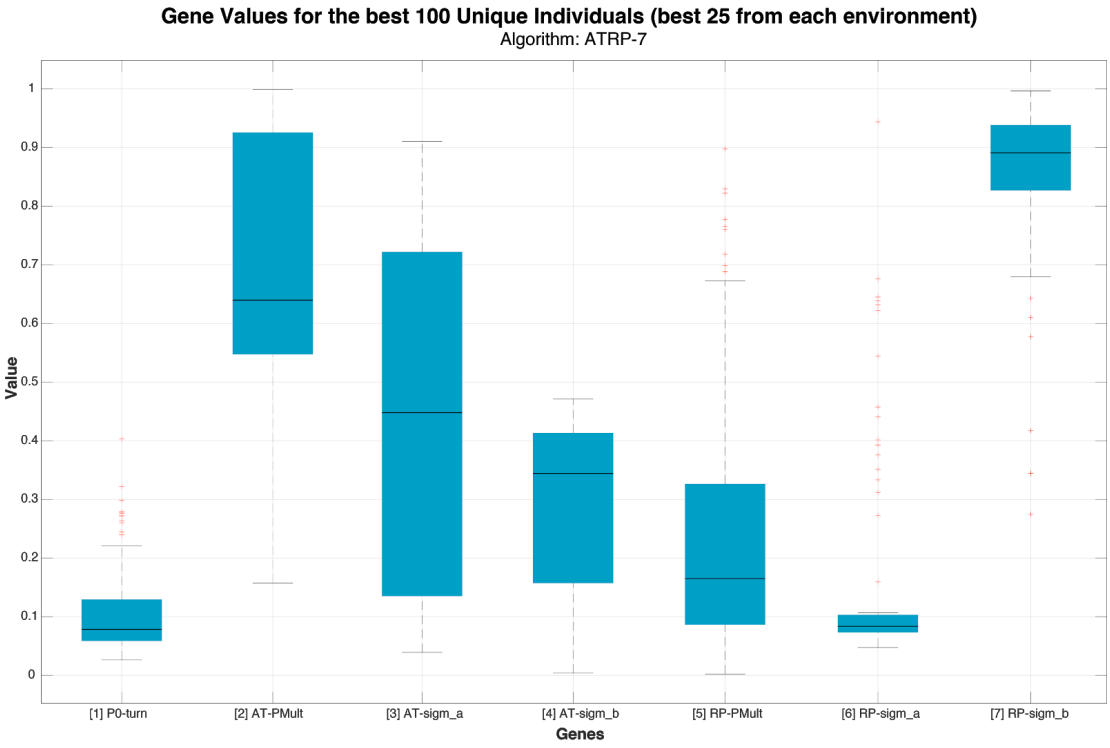


Figure 8.10 Box plot of the values for the input parameters (genes) of the best 100 individuals (25 from each environment) evolved in MA51, MA11, MA55, and MA15, running the Foraging Algorithm ATRP-7.

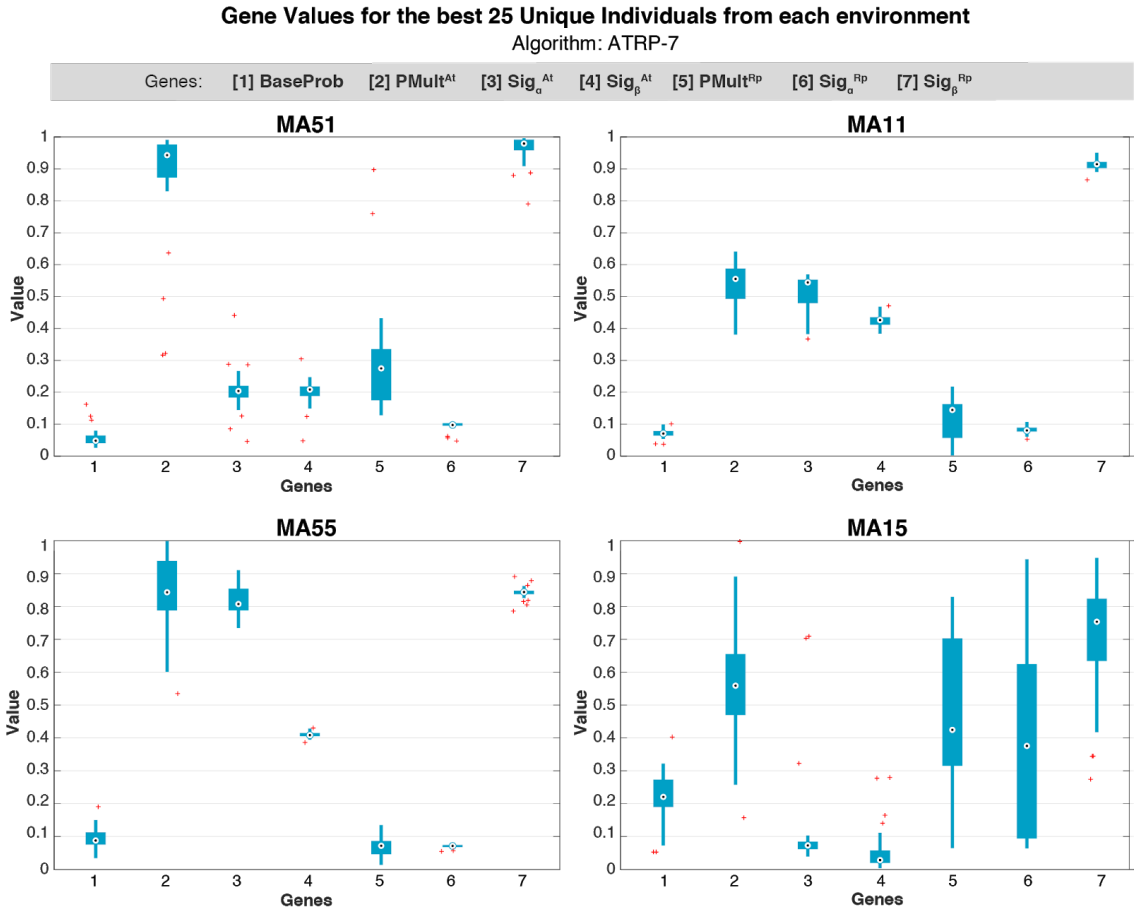


Figure 8.11 Box plot of the values for the input parameters ("DNA") of the best 25 individuals evolved in each environment, running the Foraging Algorithm ATRP-7.

The parameters (genes) of the individuals evolved in MA51, MA11, and MA55 present less variance than of those evolved in MA15 (Figure 8.11). This might be due to the difficulty of MA15, in which the optimisation probably did not find a unique convergent strategy. Hence, the best population evolved in MA15 is more diverse than those evolved in other environments.

Gene 2 ($PMult^{At}$) sets the probability multiplier for turning behaviour in the presence of attractants and gene 5 ($PMult^{Rp}$) sets the probability multiplier in the presence of repellents.

The best individuals evolved in MA51 and MA55 tend to have a higher $PMult^{At}$ (50% of the data fitting between ~0.79 and ~0.98), whereas the ones evolved in MA11 and MA15 tend to have a medium $PMult^{At}$ (50% of the data fitting between ~0.47 and ~0.66) (Table 8.4).

Individuals evolved in MA11 and MA55 have smaller values for g_5 ($PMult^{Rp}$), as 50% of the data fitting between ~0.05 and ~0.16. In MA51, values of g_5 are still small (between ~0.18 and ~0.34), whereas in MA15, the values are spread in a wider range: 50% of the data fits between ~0.32 and ~0.70, and the median is ~0.42 (Table 8.4, Figure 8.11).

Genes 3 and 4 control the sigmoid function that modulates the speed according to the presence of attractants. This pair of parameters presented itself very differently in different environments and in all of them, the range of values was compact (Table 8.4, Figure 8.11). Contrary to what was expected as these are the easiest and the hardest, environments MA51 and MA15 presented very similar pairs of g_3 and g_4: both have smaller values for both genes (approximately ~0.20 for both genes in MA51 and ~0.10 also for both in MA15). In MA11, both genes are also of similar value amongst each other, but with average values (median ~0.55 and ~0.43). In MA55, a higher value of g_3 is coupled with an average value of g_4 (median ~0.81 and ~0.41, respectively), as seen in Table 8.4 and Figure 8.11.

Genes 6 and 7 control the sigmoid function that modulates the speed according to the presence of repellents. The optimised values for MA51, MA11, and MA55 are very consistent: combining a low value of g_6 with a high value of g_7. The range for these values is also very compact in all three cases. In MA15, the values of g_6 are widely spread across the range, and the values of g_7 are also spread on the top half of the range. For g_6, 50% of the data is spread between ~0.09 and ~0.62 and for g_7, this interval is between ~0.63 and ~0.82 (Table 8.4, Figure 8.11).

Table 8.4 Percentiles 25%, 50% (median), and 75% of the gene values of the best 25 individuals evolved in each environment with ATRP-7.
Cells in blue indicate genes related to Attractants and cells in red indicate those related to Repellents.

ENV.	GENES							Percentile
	g1	g2	g3	g4	g5	g6	g7	
	BaseProb	$PMult^{At}$	Sig_{α}^{At}	Sig_{β}^{At}	$PMult^{Rp}$	Sig_{α}^{Rp}	Sig_{β}^{Rp}	
MA51	0.040	0.873	0.184	0.188	0.175	0.095	0.958	25%
	0.048	0.943	0.204	0.209	0.275	0.097	0.980	50%
	0.064	0.976	0.220	0.218	0.335	0.103	0.992	75%
MA11	0.064	0.493	0.479	0.412	0.057	0.077	0.903	25%
	0.070	0.555	0.544	0.426	0.144	0.080	0.915	50%
	0.079	0.588	0.553	0.435	0.163	0.089	0.922	75%
MA55	0.075	0.788	0.788	0.405	0.046	0.068	0.838	25%
	0.088	0.843	0.807	0.408	0.072	0.071	0.844	50%
	0.113	0.939	0.854	0.415	0.087	0.074	0.848	75%
MA15	0.190	0.470	0.061	0.020	0.315	0.094	0.634	25%
	0.221	0.559	0.073	0.028	0.425	0.376	0.754	50%
	0.273	0.655	0.084	0.058	0.703	0.624	0.824	75%

Figure 8.12 shows the results of sigmoid calculated from the median values of pairs of genes 3 and 4 (attractants) or 6 and 7 (repellents). The median was obtained from the best 25 individuals evolved in each environment.

In regards of attractants, MA15 shows the shallowest curve indicating a more gradual response, followed by MA51, which shows a steeper curve than MA15, yet not as steep as

MA11 and MA55. The curves for MA11 and MA55 were both steeper than the others, and closely resembled one another. This steeper curve pattern indicates a more rapid response to attractants. The threshold for MA11 and MA55 appears to be ~0.27, after attractants of this intensity are detected speed decreases rapidly.

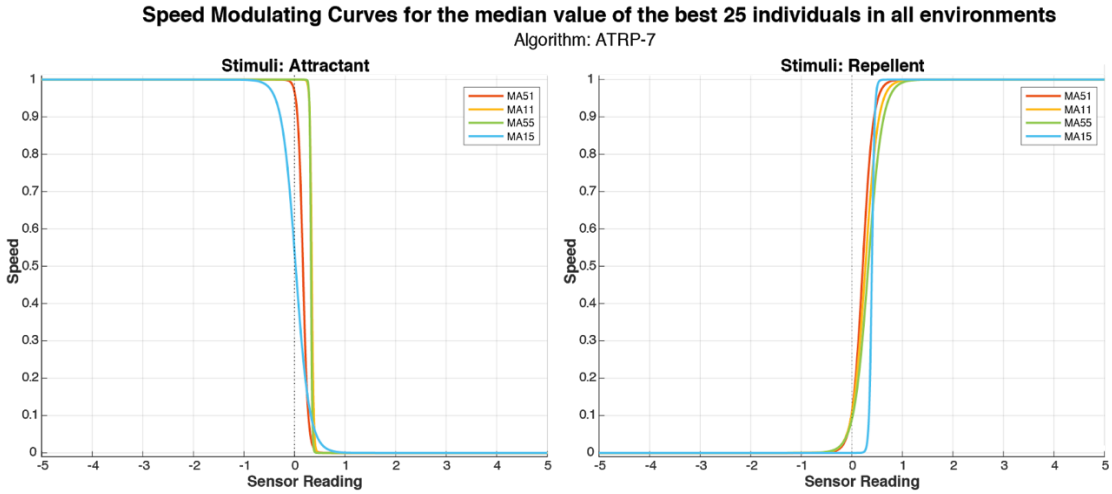


Figure 8.12 - Sigmoid Function controllers related to Attractants and Repellents for individuals evolved in 4 environments (MA51, MA11, MA55, and MA15), running the Foraging Algorithm ATRP-7.

Each line represents the shape of the sigmoid function for the median values (of the best 25 individuals) of genes 3 and 4 (attractants) or 6 and 7 (repellents) in each environment. The functions are always calculated using the absolute value of the stimulus and these plots show negative values only so the full length of the sigmoid can be seen.

In terms of the repellent curves, MA51, MA11, and MA55 all show shallower curves than that of MA15, which shows a more extreme curve. Indicating that they (MA51, MA11 and MA55) respond more gradually to repellents and also their threshold for response is much lower, their speed begins to increase after any sensor reading above 0 is detected.

MA15 has a higher tolerance for repellents, yet once its threshold is reached its acceleration is also much more rapid than the first group.

The curve for MA15 shows that in the absence of attractants the cruising speed is halved, as soon as any level of attractant is detected speed decreases gradually. MA51 also responds to any level of attractants with a slightly quicker response than MA15.

While MA15 shows a rapid response to attractants, its response to repellents is delayed, it still maintains a minimum speed. As repellents are clustered around attractants, this riskier behaviour may pay off as a repellent source may be disguising an attractant.

MA11 and MA55 show a higher threshold to respond to attractants, meaning that these individuals will maintain a higher speed for a prolonged period of time around attractants. This behaviour is coupled with a response to repellent stimuli which results in a gradual

increase in speed. A repellent of more than ~0.3 will result in a fight or flight response, whereby a robot will either give a final push to reach an attractant, which may be disguised by a repellent, or it will flee the repellent with increased speed. In both counts, this flight or fight response is also affected by gene 5, which controls the probability multiplier for repellents, so it affects the probability of a robot fleeing or pushing through to a possible attractant.

MA55 shows a more gradual response than MA11, indicating a higher tolerance for repellents, possibly due to the fact that, although the levels are balanced in MA55, the repellents cause more damage to the battery levels than the attractants can charge in the same amount of time. Meaning that MA55 is a more challenging environment than MA11, this increased difficulty results in a more tolerant behaviour to repellents as by maintaining a low speed near repellents, attractants may be discovered.

Regarding the turning behaviour, in ATRP-7 the probability of turn is set by 3 genes, genes 1, 2 and 5. Similarly to ATRP-8, the robots evolved to respond to decreasing levels of attractants and increasing levels of repellents, meaning they will turn back towards an attractant source, and will turn away from a repellent source. Even though in this series of experiments the parameters controlling this were set individually for attractants and repellents. The individuals evolved in MA15 were the only ones to evolve a definite probability of turn, with a resulting value of ~1.23. Similarly in this environment the response to repellents is also strong, approximately ~0.94. This is the most challenging of all the environments so the robots have evolved to avoid missing an attractant source at all costs, considering their scarcity. Also the repellents are so strong in this environment that they evolved to almost certainly flee a repellent source.

In all environments except MA15, the individuals evolved low probabilities (~0.06 to ~0.13) of turning back from repellents.

In MA11 and MA51 the response to attractants are very close in value, ~0.39 and ~0.45, respectively, this similarity may be due to the decreased intensity of repellents, and as these environments have more abundant attractants compared to repellents, so as they are less likely to go without any attractant resources they are more likely to explore further.

For MA55 the resultant probability of turning in response to attractants is ~0.74, indicating that turns are likely. This may be due to the intensity of repellents, as repellents are stronger in this field the robots become restricted by them and limit their explorations.

As can be seen in Figure 8.11 and Table 8.4, individuals evolved in MA15 present more variability in the range of some genes (2, 5, 6, and 7) than those evolved in other environments. This is especially true for genes 6 and 7, that modulate the speed according to the presence of repellents. In all environments except MA15, the range of values of these

genes is very compact: ~0.009 and ~0.018 (MA51); ~0.010 and ~0.015 (MA11); 0.006 and 0.009 (MA55), whereas in MA15 the range is ~ 0.532 and ~0.201 (Figure 8.11, Table 8.4).

The discussions continued here refer to the median values of the best individuals of each environment, however, this is not to say that there were no other successful strategies employed in these environments. As can be seen in Figure 8.11 (notice the variance of some genes) and in Fig. 8.13 and 8.14, different strategies emerge in the same environment. These similar strategies emerging in different environments will be covered in the Genetic Analysis.

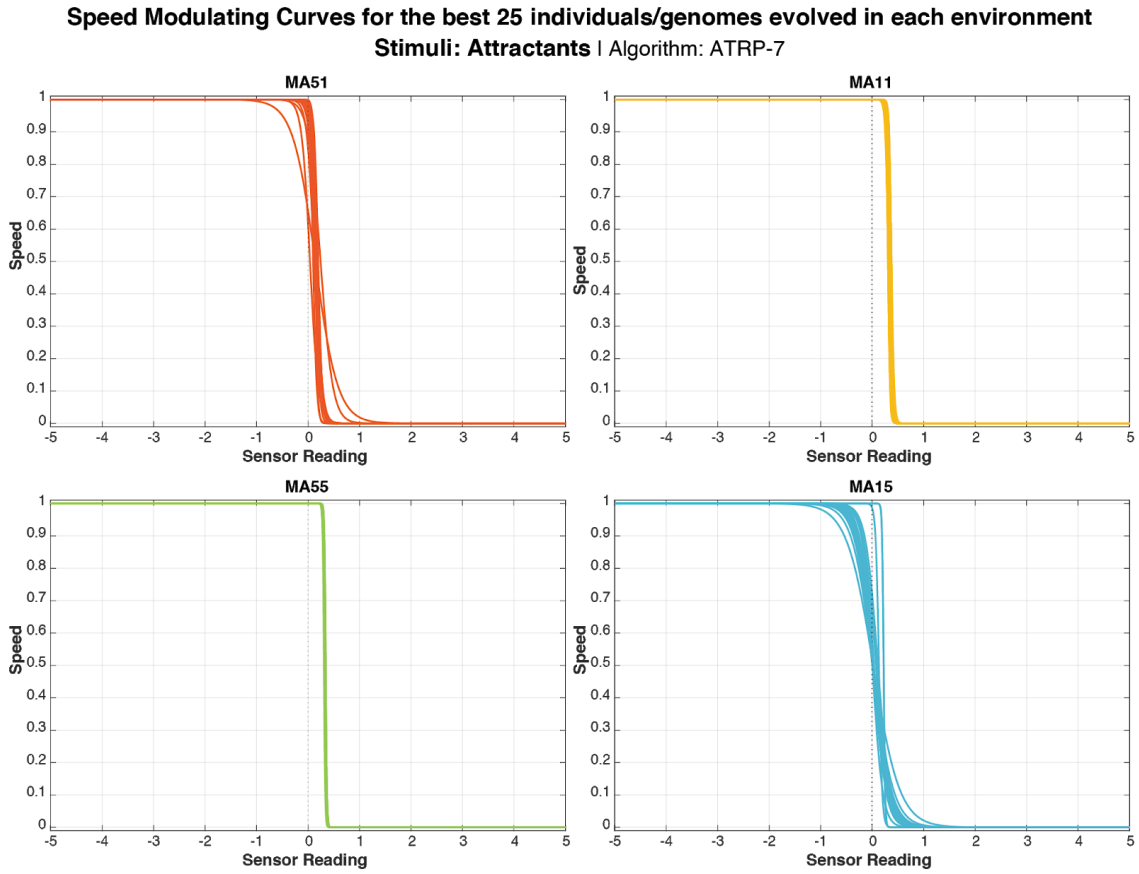


Figure 8.13 Sigmoid Function controlling the Speed Modulation related to Attractants of the best 25 individuals evolved in 4 environments (MA51, MA11, MA55, and MA15), running the Foraging Algorithm ATRP-7. Each line represents one of the 25 best individuals evolved in each environment.

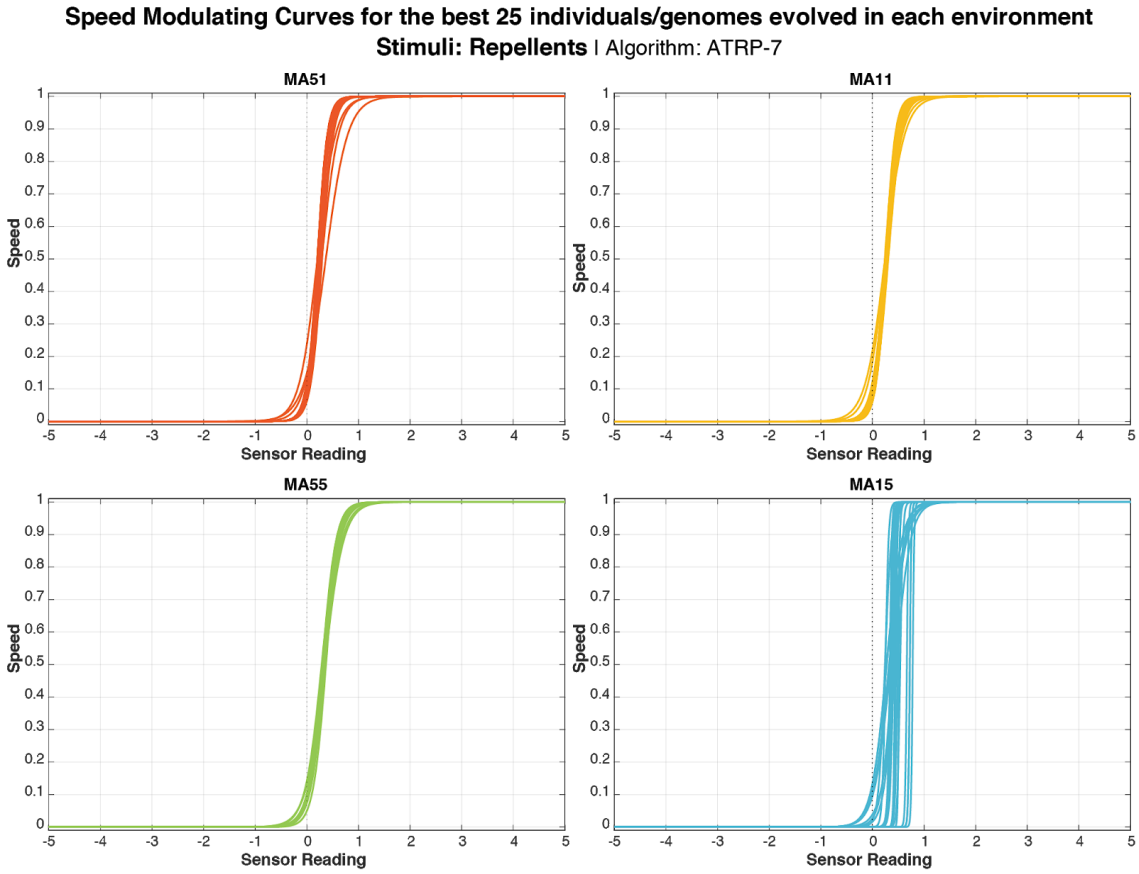


Figure 8.14 - Sigmoid Function controlling the Speed Modulation related to Repellents of the best 25 individuals evolved in 4 environments (MA51, MA11, MA55, and MA15), running the Foraging Algorithm ATRP-7. Each line represents one of the 25 best individuals evolved in each environment.

Table 8.5 Resulting values for variable *PTurn* calculated from the median of the genes of the best 25 individuals evolved in each environment with ATRP-7.
The value ~0.1 (relevant variation in light intensity) is calculated taking into account the standard sensor noise (1%), the value of *NoiseTol* (optimised in previous experiments), and the *Time Step Size* (*TSS=4*). Values highlighted in red show a probability of 100% (certainty) of turning.

	Contribution for the Resulting *PTurn*					
Median of individuals evolved in:	***PTurnAt* / 2**			***PTurnRp* / 2**		
	increase (more than ~0.1)	**decrease** (more than ~0.1)	**else**	**increase** (more than ~0.1)	**decrease** (more than ~0.1)	**else**
MA51	0.000	0.455	0.002	0.132	0.000	0.002
MA11	0.000	0.391	0.004	0.102	0.000	0.004
MA55	0.000	0.741	0.004	0.063	0.000	0.004
MA15	0.000	1.233	0.011	0.937	0.000	0.011

8.3.3. Characterization of Evolved Genomes

In this section, I am going to present the clustering for the best 25 individuals evolved in each environment in two sections: firstly those organised in 3 clusters, followed by those organised in 7 clusters.

8.3.3.1. 3 Clusters

This section presents the clustering of the DNA of the best individuals, the characterization of the native environments the validation, and the interpretation of the clustering results for the best 25 individuals from all 4 environments combined, grouped in 3 clusters with *k-means++* (Fig. 8.15, 8.16, 8.17, 8.18).

The speed modulating strategy towards repellents is similar on Clusters 1 and 3, and characterized by a similar threshold and by a more gradual response than in Cluster 2. However, the speed modulating strategy towards attractants in Clusters 1 and 3 differ: in Cluster 1, the threshold is lower and the curve is shallower, meaning a gradient response starting as soon as any level of attractant is spotted. Whereas Cluster not only has a delayed response combined with a steeper curve, meaning that once a certain level of attractants is sensed, the robots' response is rapid.

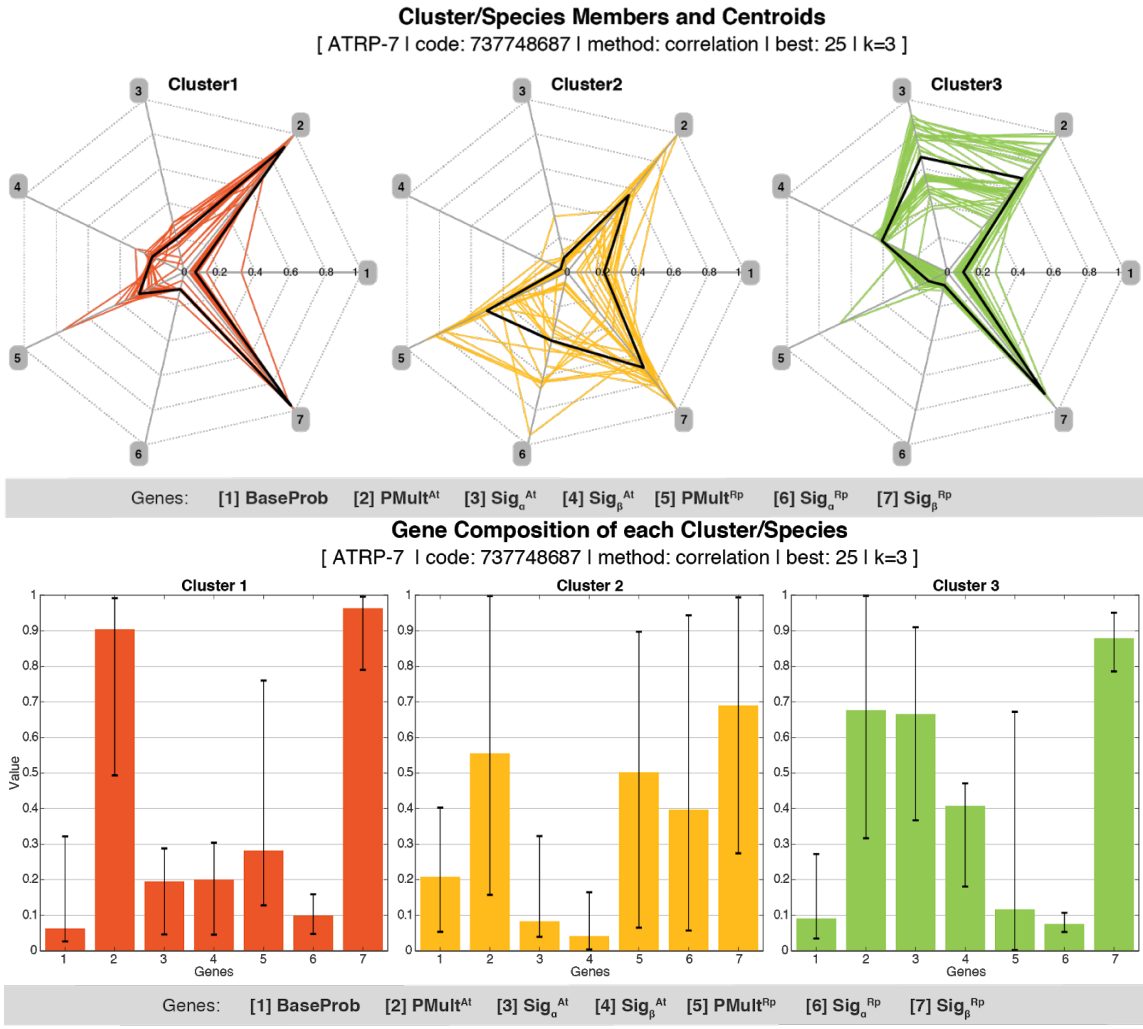


Figure 8.15 Clustering results of the best 25 from each environment (a total of 100), grouped in 3 clusters using *k-means++* and correlation (Series J).

Top Row: each subplot (spider plot) shows the cluster members and centroids. Each axis represents one input parameter and the values are distributed centre wise, from 0 to 1. Cluster members are plotted in coloured lines, and the centroid is plotted in black. Bottom row: Each subplot shows the DNA composition of the members of each cluster, and the respective standard deviation.

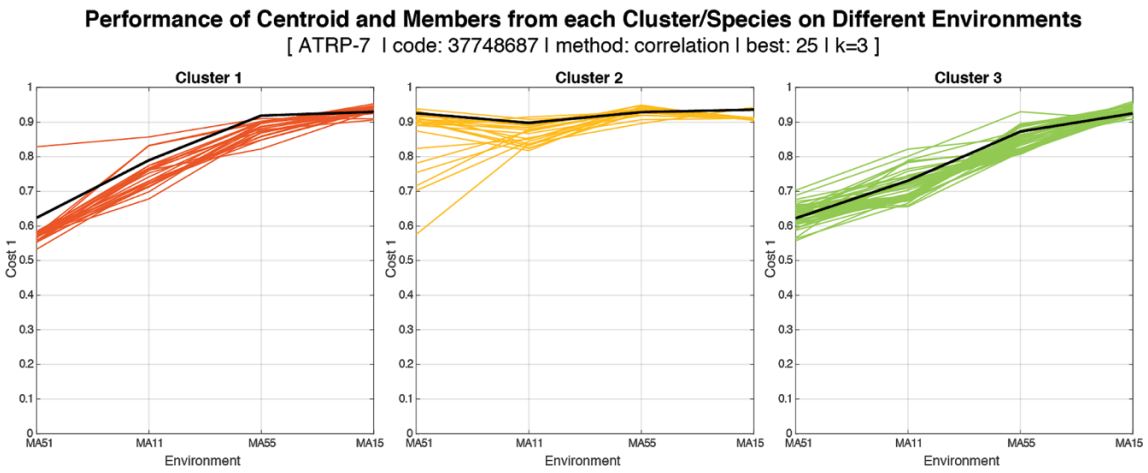


Figure 8.16 Clustering results of the best 25 from each environment (a total of 100), grouped in 3 clusters using *k-means++* and correlation (Series J).

Each subplot shows the cluster members and centroids. Each vertical axis represents the value of Cost1 in one environment. Cost 1 of cluster members are plotted in coloured lines, and Cost 1 of the centroid is plotted in black.

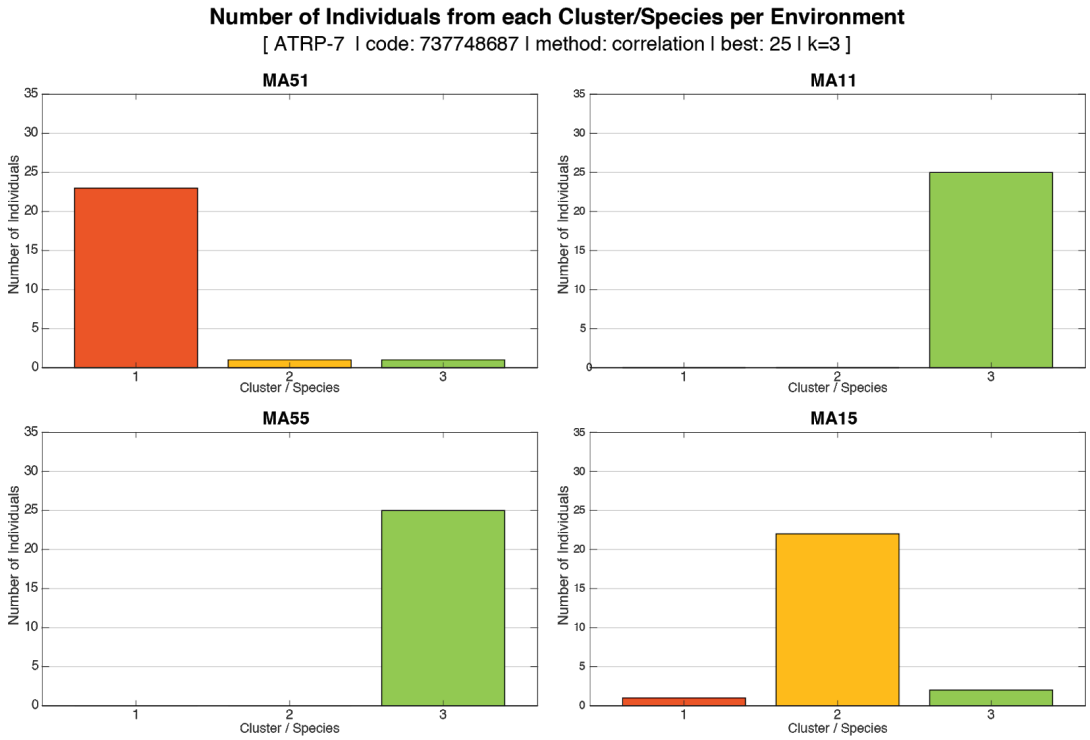


Figure 8.17 Clustering results of the best 25 from each environment (a total of 100), grouped in 3 clusters using *k-means++* and correlation. Each subplot shows the presence of members from different clusters in each environment.

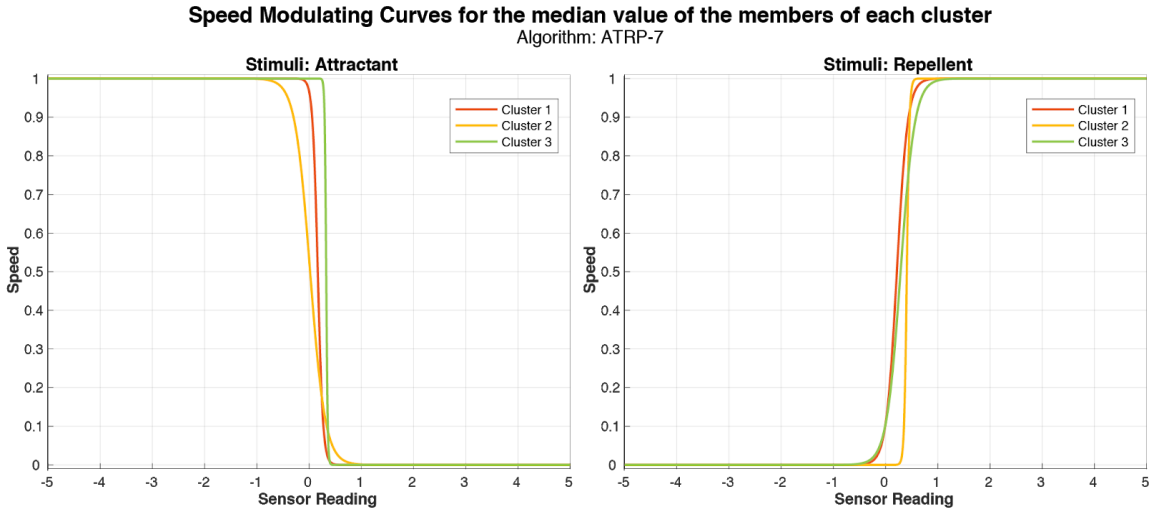


Figure 8.18 Sigmoid Function controllers related to Attractants and Repellents for each cluster in a scenario with 3 clusters.

Each line represents the shape of the sigmoid function for the median values of genes 3 and 4 (attractants) or 6 and 7 (repellents), for all the cluster members. The functions are always calculated using the absolute value of the stimulus and these plots show negative values only so the full length of the sigmoid can be seen.

8.3.2.2. 7 Clusters

This section presents the clustering of the DNA of the best individuals, the characterization of the native environments the validation, and the interpretation of the clustering results for the best 25 individuals from all 4 environments combined, grouped in 7 clusters with *k-means++* (Fig. 8.19, 8.20, 8.21, 8.22).

Species 2, 3 and 7 have low values of g_3 (Sig_{α}^{At}), meaning a gradual response to attractants. Species 1, 4, 5, and 6 have evolved steeper curves, with slight variations in slope and threshold, meaning a rapid response to attractants once the threshold is reached. Species 2, 3, 4, and 7 start to react to attractants as soon as the slightest intensity is detected, whereas species 5, 1, and 6 display a delayed (but rapid) response.

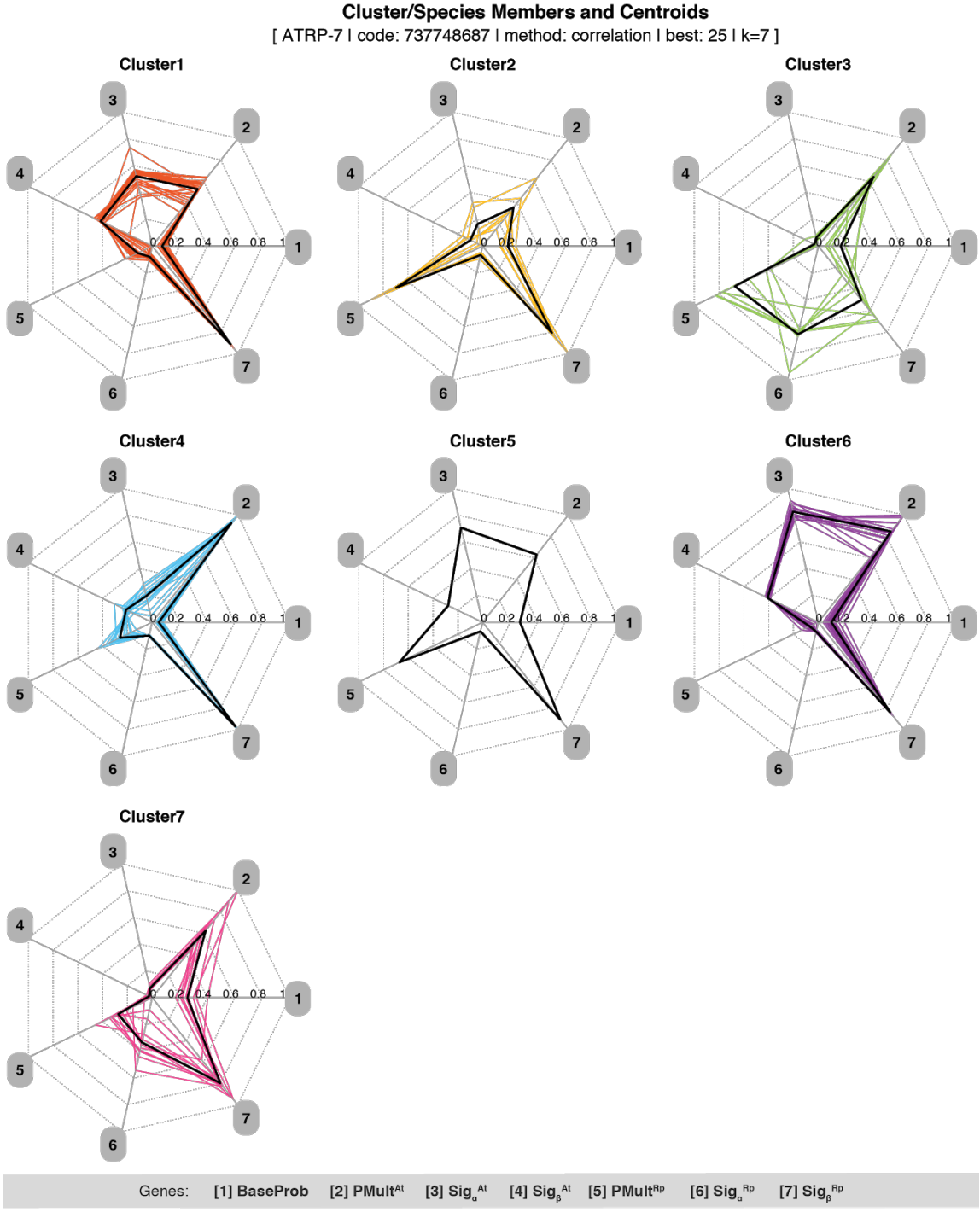


Figure 8.19 Clustering results of the best 25 from each environment (a total of 100), grouped in 7 clusters using *k-means++* and correlation.

Each subplot (spider plot) shows the cluster members and centroids. Each axis represents one input parameter and the values are distributed centre wise, from 0 to 1. Cluster members are plotted in coloured lines, and the centroid is plotted in black.

Species 1, 2, 4, 5, and 6 have low values of g_6 (Sig_{α}^{Rp}), meaning a gradual response to repellents. Species 3 and 7 evolved very steep curves with differing thresholds from one another and from the other curves. Species 1, 2, 4, 5 and 6 also have a lower threshold,

meaning that they will start to react as soon as any level of repellent is detected. Species 3 and 7 have higher thresholds, meaning a delayed yet rapid response, once the threshold is reached, due to the steepness of the curves.

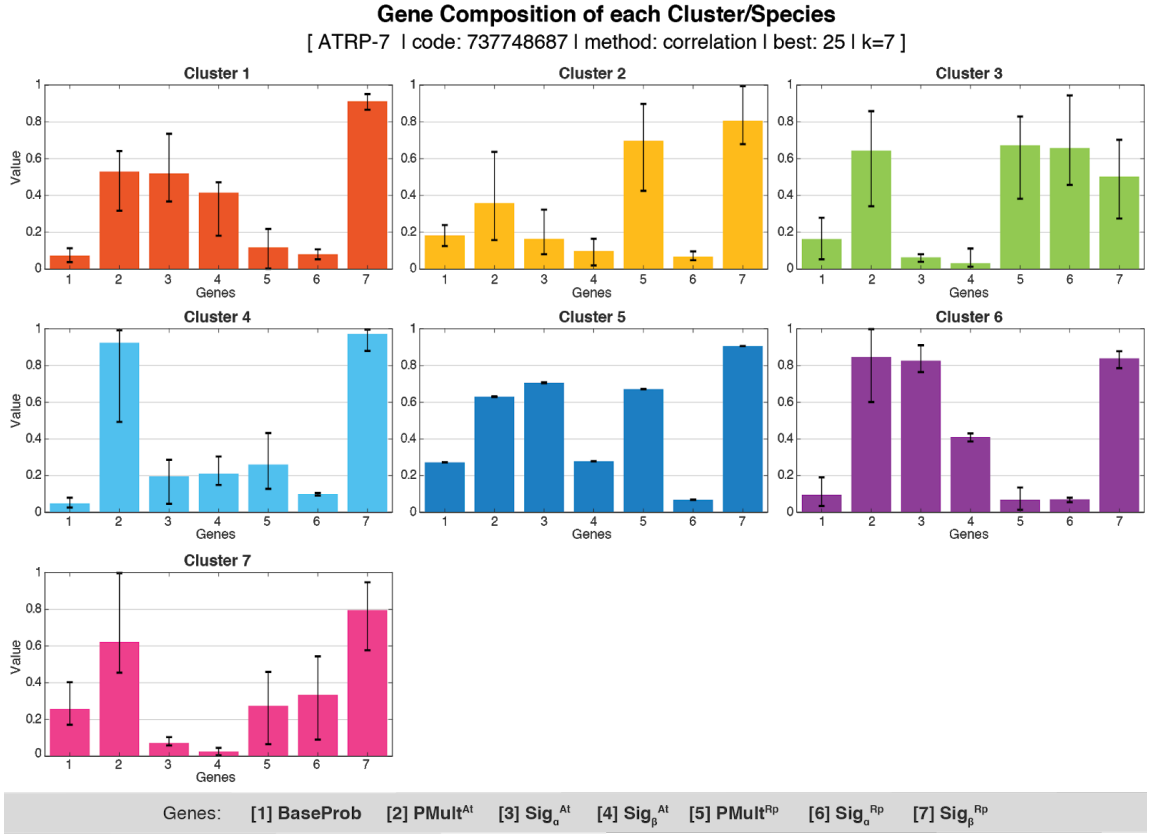


Figure 8.20 Clustering results of the best 25 from each environment (a total of 100), grouped in 7 clusters using *k-means++* and correlation.

Each subplot shows the DNA composition of the members of each cluster, and the respective standard deviation.

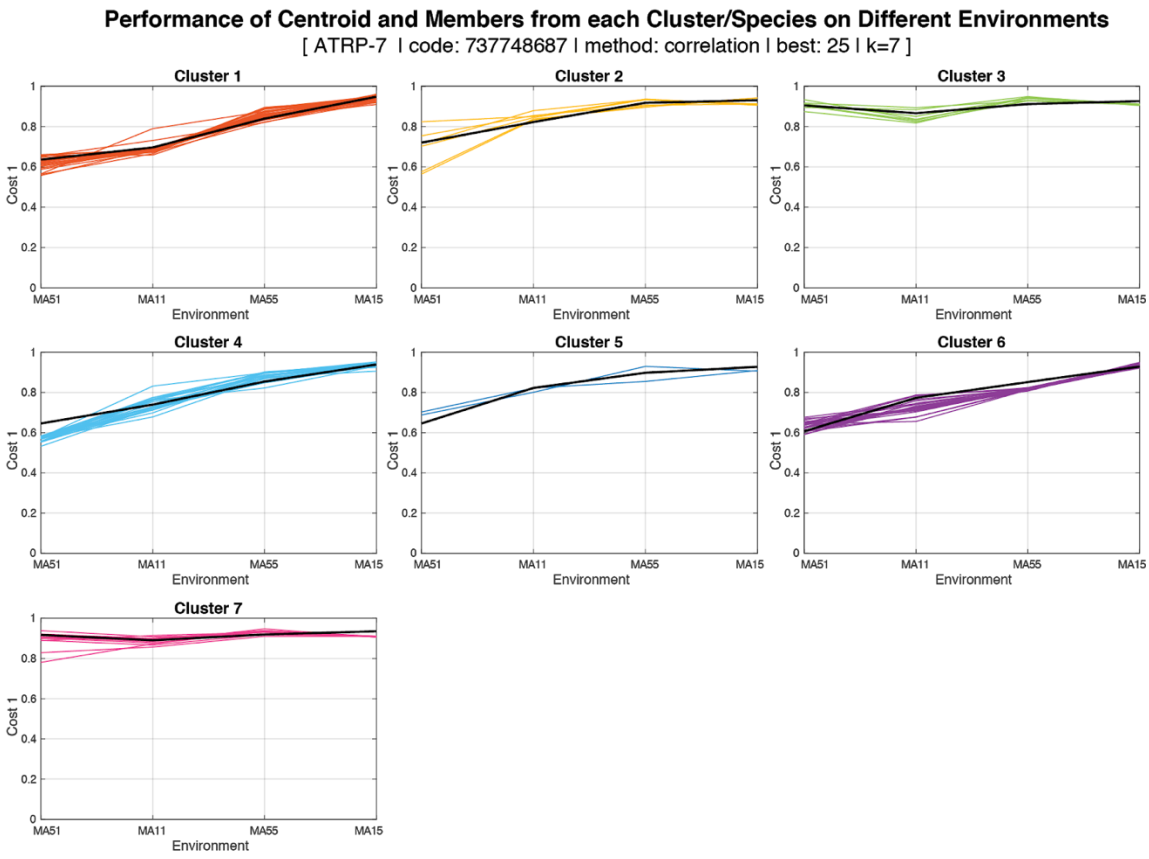


Figure 8.21 Clustering results of the best 25 from each environment (a total of 100), grouped in 7 clusters using *k-means++* and correlation (Series J).

Each subplot shows the cluster members and centroids. Each vertical axis represents the value of Cost1 in one environment. Cost 1 of cluster members are plotted in coloured lines, and Cost 1 of the centroid is plotted in black.

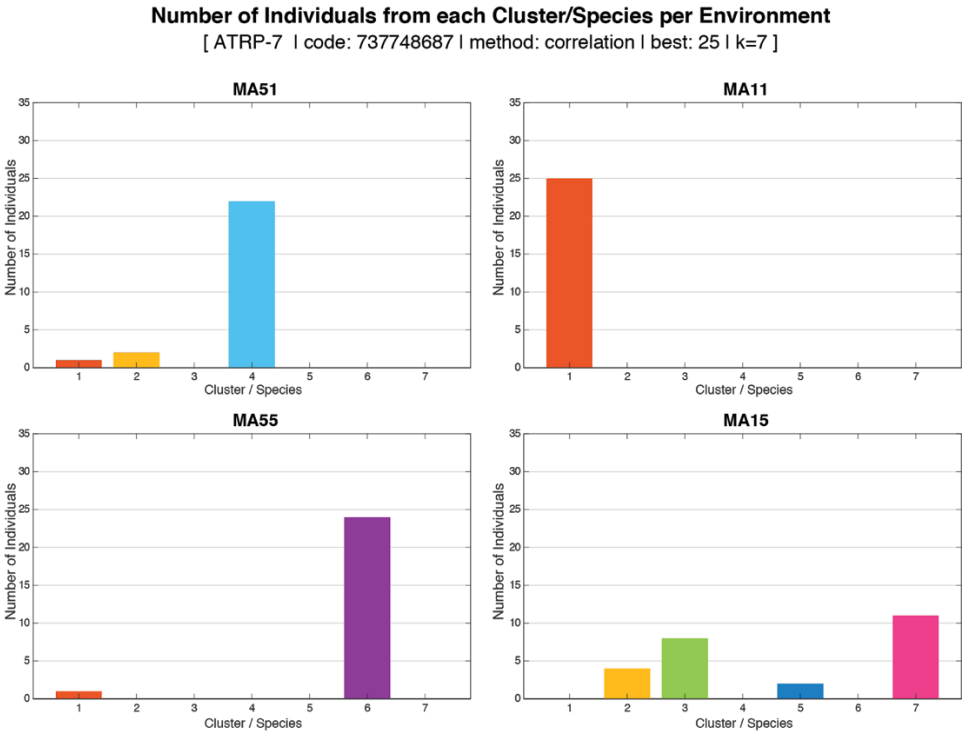


Figure 8.22 Clustering results of the best 25 from each environment (a total of 100), grouped in 7 clusters using *k-means++* and correlation.

Each subplot shows the presence of members from different clusters in each environment.

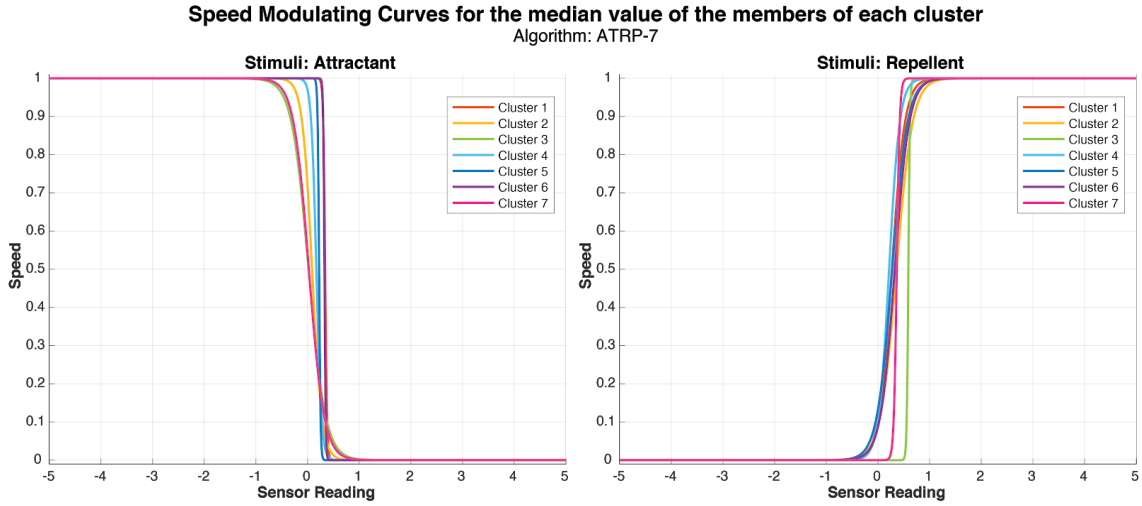


Figure 8.23 - Sigmoid Function controllers related to Attractants and Repellents for each cluster in a scenario with 7 clusters (Series J).

Each line represents the shape of the sigmoid function for the median values of genes 3 and 4 (attractants) or 6 and 7 (repellents), for all the cluster members. The functions are always calculated using the absolute value of the stimulus and these plots show negative values only so the full length of the sigmoid can be seen.

8.1. Conclusions

Overall, this algorithm functioned as well and in many cases better than the previous algorithms. Furthermore, it was shown to be more flexible given that it out-performed the other algorithms in most environments (Figure 8.4). In this case the algorithm benefitted from evolving in a more forgiving environment as can be seen in MA55, likely due to the algorithm having a longer runtime in terms of generations. Although this foraging algorithm did not outperform the others in every scenario, its performance was favourable in all but 6 environments, out of 22.

In previous algorithms the probability for turn to attractants and repellents was fixed at the same value, however in this version the module for determining the behaviour towards attractants is completely separate to that dealing with repellents.

As can be seen in Table 8.5 the values for controlling attractants and repellents evolved to be very different, unlike can be seen in the previous algorithm when these values were forced to align. For further development it is important that the modules for dealing with different stimuli work independently of each other. Even with a small range of environments, using just 4, we could still see different patterns of emergent behaviour, for

example using just 7 clusters we were able to find at least 7 different behavioural patterns during genetic analysis.

Chapter 9
Comparison between AT-6, ATRP-8, and ATRP-7

In this step, the best individual evolved in each of the 22 environments was tested on its own and in all of the other environments. The results are presented in Figure 9.1, and the performance of the individuals running separate foraging algorithms is highlighted in Fig. 9.2, 9.3, and 9.4.

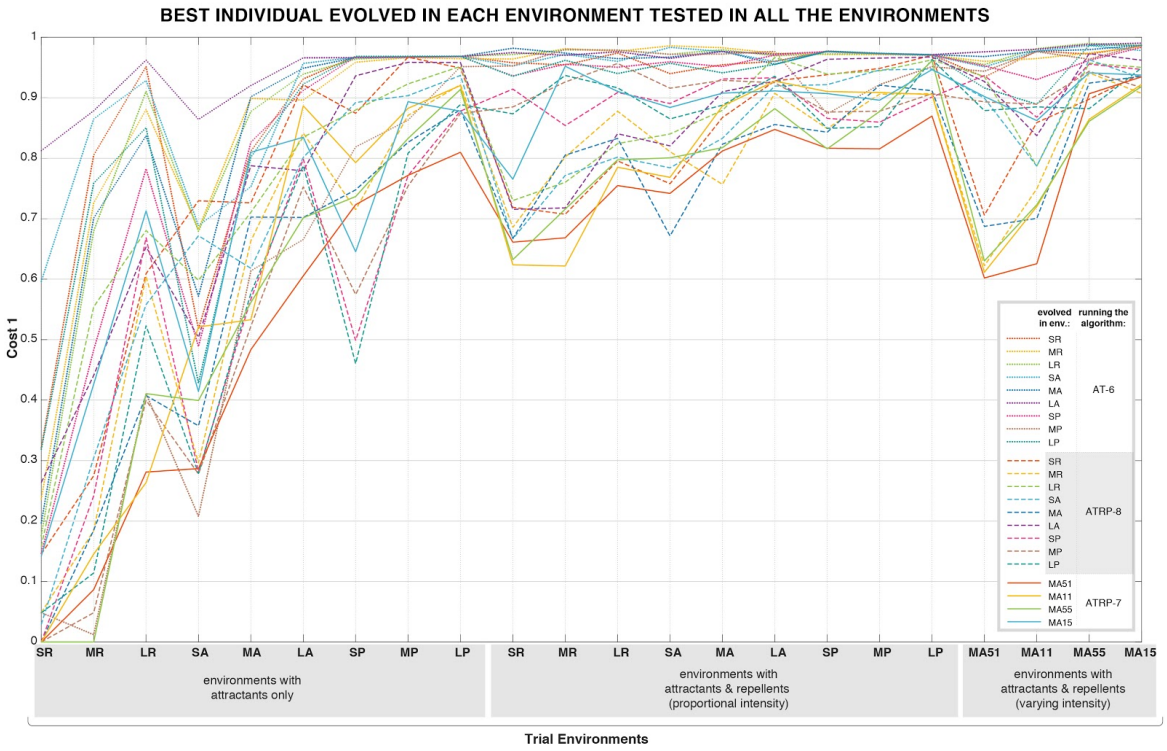


Figure 9.1 Performance of the best individuals evolved in series H, I, and J (running foraging algorithms AT-6, ATRP-8, and ATRP-7) tested in all 22 environments.

Dotted lines indicate the best individuals evolved using Foraging Algorithm AT-6 in environments with attractants only (Series H). Dashed lines indicate the best individuals evolved using Foraging Algorithm ATRP-8 in environments with attractants and repellents (Series I). Solid lines indicate the best individuals evolved using Foraging Algorithm ATRP-7 in environments with attractants and repellents of varying ratios (Series J).

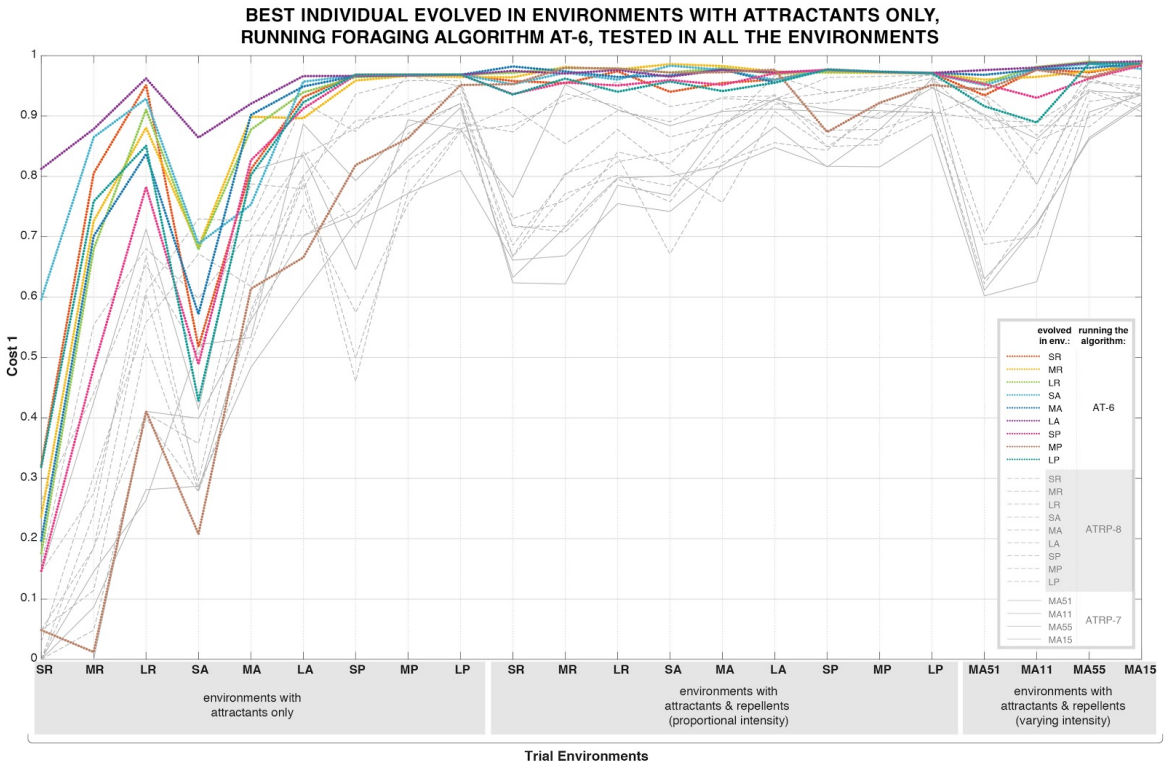


Figure 9.2 Highlight of the performance of the best individuals evolved in series H (running foraging algorithm AT-6) tested in all 22 environments.

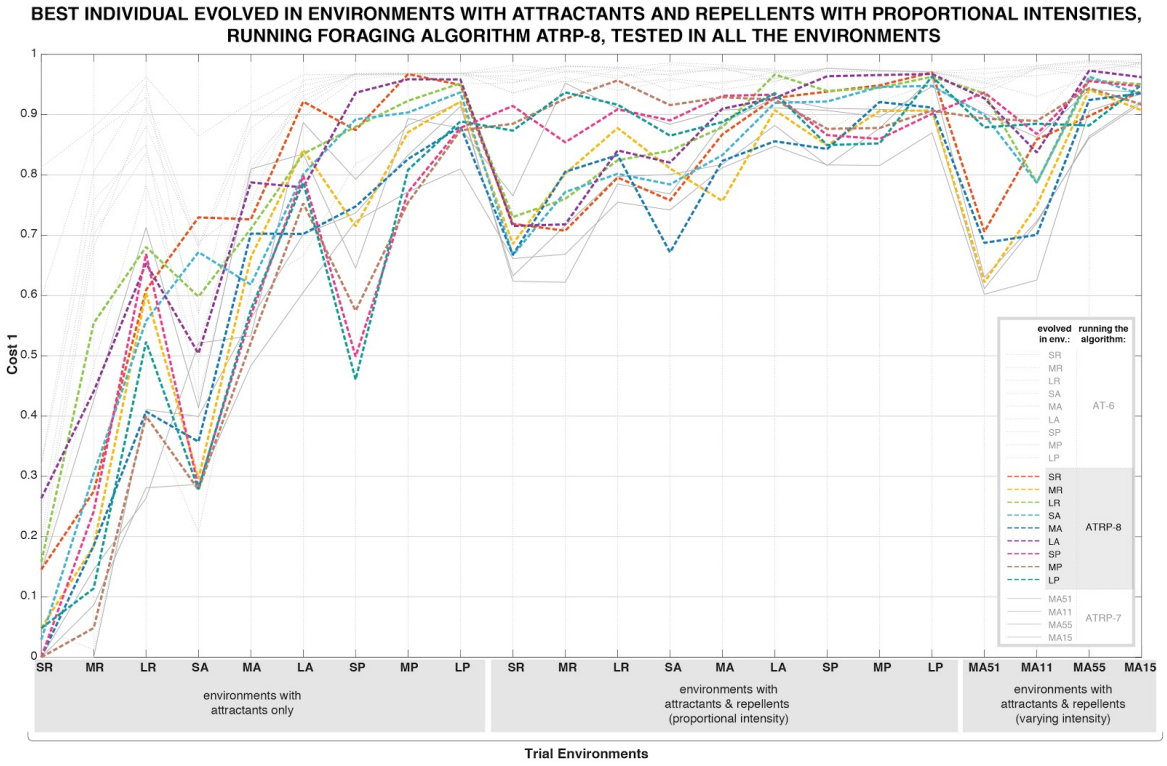


Figure 9.3 Highlight of the performance of the best individuals evolved in series I (running foraging algorithm ATRP-8) tested in all 22 environments.

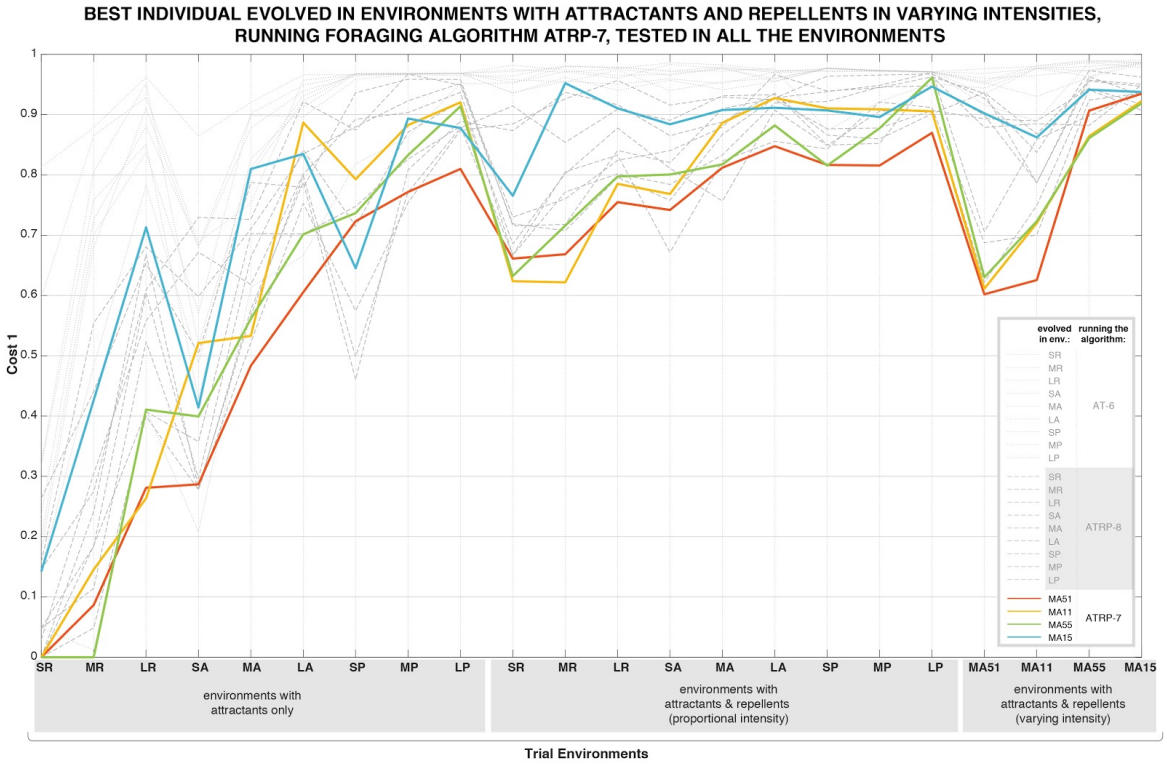


Figure 9.4 Highlight of the performance of the best individuals evolved in series J (running foraging algorithm ATRP-7) tested in all 22 environments.

The latest algorithm (ATRP-7) outperformed the other two (ATRP-8 and AT-6) in most cases when tested within the same environments (Fig. 9.1 and 9.4). However ATRP-8, was more successful in the following environments, SP (with attractants only), SA, MA and MA15 all with both attractants and repellents (Fig. 9.1 and 9.3). AT-6 only outperformed the other algorithms in SA, with attractants only (Fig. 9.1 and 9.2). Although, this may be down to chance, as it appears to have outperformed all of the other individuals evolved with attractants only. This series of experiments ran with the very best individual (genome) from each environment and 20 samples (each robot in a simulation is considered to be one sample), the usual number of samples for a series such as this would be 100.

The results obtained from the individuals running ATRP-7 and evolved in MA51, MA55, and to some extent MA11, and for the individuals running ATRP-8 and evolved in LR, MR, SR, all have less performance variation, which appears as smoother curves. This indicates they are less specialised and behave in a more generalist manner (Figure 9.1, 9.3, 9.4).

In general, with the exception of SA, individuals running AT-6 had overall poor performance compared to the other algorithms, even in their native environments. This would be expected in environments with repellents, as they do not a way to process these stimuli. However, it was unexpected that they would be outperformed in their native environments, as it happened in LR, MA, SP, MP, and LP. That indicates that the second and third refined algorithms (ATRP-8 and ATRP-7) function well even in environments

without repellents. The third refined algorithm was tuned with the results for Variability of Turn and Noise Tolerance and these parameters were fixed according to the optimised values from earlier series of experiments (Series H and I), which may explain, to some extent, the success of this algorithm.

Individuals running ATRP-8 evolved in MA, SP, and MP and those running ATRP-7 evolved in MA51, MA55, and MA11 performed well in SP with attractants only (reaching cost 0). The first group (ATRP-8) is composed of individuals that evolved in environments of average difficulty, whereas the second group (ATRP-7) is composed of those evolved in the easier environments. For the first group, that might indicate the influence of some of the genes in the module that deal with attractants being shared with the module that deals with repellents. As in ATRP-7 these modules are independent, the performance is more consistent, meaning that the individuals evolved in easier environments perform well in other easy environments.

Those running ATRP-8 evolved in MP and those running ATRP-7 evolved in MA51, MA55, and MA11 performed well in SP with attractants only (reaching cost 0), indicating they evolved to specialise in finding attractants.

Those evolved in poor environments with attractants and repellents, (SP MP and LP) became more specialised in finding attractants, especially those of low intensity.

Chapter 10
General Discussion and Conclusions

The work presented here leads to two distinct conclusions, one practical and one more academic. From a practical perspective, the results suggest that, when suitably tuned, minimalist *C. elegans*-inspired foraging algorithms can lead to effective navigation to unknown targets even in the presence of significant sensor noise. Furthermore, it suggests that this approach could be used to achieve long-term autonomy by allowing robots to seek out energy sources in their environments, even in the presence of harmful sources. The simulations used realistic physical parameters (speed, battery capacity, current, etc), but these results would benefit from validation on real hardware.

The behavioural modules to detect attractants contained in all three Reflex-agent *C. elegans* Foraging Algorithms have proven to be effective at foraging energy from light spots in several field sizes, even with very simple hardware and behavioural algorithms. The behavioural algorithms have been proven to be sound, as using a perfect sensor (or the ideal hardware) all robots survived the entirety of the simulation (as seen in series G). Further experiments considering a realistic hardware approach encompassing sensor noise also obtained successful results that were proportional to the difficulty of the environment.

From an academic perspective, the work demonstrates that even simple models can serve as an interesting and informative testbed for exploring fundamental evolutionary principles. Despite the refined algorithms having only six to eight free parameters, results representative of the evolutionary differentiation of species have been achieved. The work replicates, in a simplified and tractable way, how modest changes in environmental conditions lead to evolutionary adjustments. When the changes are more substantial, these adaptations are akin to differentiation of species.

As an unexpected outcome of this research, the Genetic Analysis (Genotype Clustering) approach displayed promising results as a tool for Optimisation and Evolutionary Robotics. As a novel approach, unseen in the literature, the methods will be refined and prepared for publication.

Furthermore, the approach of grouping solutions evolved in different environments and then analysing these on the basis of genetic (as opposed to behavioural) similarity is novel, to the best of my knowledge, and yielded interesting results that aligned very well with behavioural outcomes.

Overall, the results obtained up to present demonstrate a promising application of bio-inspired behavioural algorithms on the development of autonomous mobile robots. As all three refined algorithms are inspired by a remarkably simple organism, and therefore would

not require complex or costly technologies, these results may also contribute to the advance of energy-efficiency and low-cost autonomous robots.

In terms of the methodology, and the results achieved; the clustering and cluster validation methods worked very well for the input parameters and for the cross-environment trials in the first algorithm, which processes only attractants, likely because it was a more simple experiment focused on optimising a single behaviour (seeking attractants). For the second and third foraging algorithms (ATRP-8 and ATRP-7), it became more difficult to find smaller numbers of clusters which would be representative of the genes and achieve similar performance when tested in all the environments. This was somewhat expected, as these algorithms have more input parameters and encompass twice the number of behavioural modules than the first refined algorithm. This is also likely due to the complexity of the environments in which the experiments with these algorithms were performed: in addition to the layer of attractants with varying levels of intensity, a layer of harmful repellent spots clustered around the attractants was added to the model.

If attractants and repellents were in existence separately within the same field it is likely the complexity would double. However as in this scenario repellents are grouped around attractants it becomes much more challenging for the robots to navigate towards the attractants without becoming trapped in a repellent source. The attractants are always surrounded by harmful sources, forcing robots to balance their behaviour, a small risk may result in a better pay off, but a greater risk may drain the robots battery completely (with "fatal" consequences). This trade off can be seen in nature, for example an ant leaving the nest to look for food would also expose itself to predators, or risk.

In experiment series I, with ATRP-8, the intensity of repellents is proportional to the attractant source they are surrounding. A stronger attractant will be surrounded by stronger repellents, so a bigger risk will result in a bigger reward for the robots choosing to take this risk. However, this may encourage less natural behavioural patterns, as robots will adapt to view strong repellents as a sign of strong attractants and therefore be more willing to approach a repellent instead of avoiding it.

In experiment series J, with ATRP-7, different intensity ratios between attractants and repellents have been tested. Not all strong attractants were surrounded by strong repellents. Ideally, this scenario in which attractants and repellents of varying intensity ratios are clustered together would have been tested on both ATRP-8 and ATRP-7. However, due to time restrictions, this was not feasible due to the additional CPU time it would have required.

The MA environment from series I was repeated in series J (as MA11) as a control to compare results between series, especially between the foraging algorithms ATRP-8 and ATRP-7. These results would otherwise have been less comparable without cross trials in

all environments. To save time, one environment was selected to improve the validity and track behavioural changes between experiments.

Finally, ATRP-7 appeared to be the most capable of the refined algorithms, resulting in lower costs when compared to ATRP-8 and AT-6. Experiments running ATRP-7 against others in MA11 resulted in lower costs in most environments (16 out of 22). Although this algorithm was not successful in every single environment, it outperformed the other algorithms in most environments. Using a novel combination of methods (which I termed genetic analysis) which were developed in this research allows us to analyse the data in a more isolated fashion, allowing specific strategies to be examined more closely. Furthermore it allowed more than one successful strategy to develop within each environment, proving there is more than one way to succeed in these environments, as in nature.

In terms of the simulation framework; it offers a robust, efficient solution, and regarding the computational requirements it can be run on relatively simple machines. The current version is still in its early and very simple stages, yet remains flexible enough to produce a multitude of qualitatively and quantitatively different environments.

I consider this framework in and of itself to be a key contribution to the field, as it will save researchers countless time. A framework like this was not available to me at the start of this research, so I had hoped and considered the introduction of such a framework as an option for the standardisation of further algorithms. The simulation framework will be a key component of any future work, alongside the algorithms and the genotype clustering, as is discussed below.

10.1. Contributions and Future Work

Throughout the development of this work it became clear that research in the field of Artificial Life is strongly influenced by a host of more specific fields. From biomimetics and the tools rooted in design used to reverse engineer organic systems to the comprehension of molecular dynamics, electrochemical reactions, software programming, and the statistical treatment of data, many areas were involved. It is only natural that the ramifications into future work to be developed, both as direct continuation of processes here explored and by less obvious by-products, are extremely varied.

With this in mind, this section lists areas in which some level of work can be developed following a process initiated by the research here documented. The type of possible future works can be segmented by proximity with the original piece - artificial life, computational simulations, biomimetic design and genetic algorithms being some of the most direct connections, while robot swarms, animal physiology, biostatistics, and ecology studies are more distant applications. Another option is to divide by chronological feasibility, for some applications are candidates for immediate implementation, while

others exist only in the hypothetical realm as of today - think of robotic interstellar and intrabody exploration.

Beginning with the most direct applications of the ideas initiated by this work, the continued work on development of algorithms inspired by animal behaviour is one of the most promising tools to be further developed. The analysis and conversion of *C. elegans'* foraging behaviour into several algorithms made clear that organic systems can have their behaviour analysed and thoroughly documented by biological sciences (like zoology, ecology, or more specific areas) and then reinterpreted as synthetic behaviours coded into a simulation. In a similar fashion to the source material used in this work, the organic behavioural patterns may have been already documented, lacking only its conversion into algorithms. Considering this, the tools proposed and developed for the analysis of the simulated *C. elegans* could be readily adapted and become even more appealing in analysis both for extrapolating observational data as to corroborate it.

On the same principle, the simulation framework built for the *C. elegans* algorithm can be developed into a modular toolbox for biological analysis, including tools for individual and population-based studies. Such a toolbox can be built around combinations of simple actions that, when combined, describe in detail the behaviour in focus. This would be an evolution from the current algorithm building model, in which the behavioural elements are hardcoded into the program, and a direct expansion of the style of documentation referenced by this work, which used terms like "omega turns" and "inversions" for describing the movements and foraging patterns of *C. elegans*. If a high level of modularity is achieved, the toolbox could become universal enough to also be useful as a behavioural documentation assistant, facilitating the investigation and transcribing of patterns exhibited by living organisms. Concomitantly, a beneficial result of this software design is that for each time the toolbox is employed as documentation assist it produces a new entry in an ever-improving library of species upon which to base future simulated organisms.

Still in the computational field, one important outcome from the development of life simulations is the better definition of methods for biomimetic analysis and subsequent conversion into algorithms. The proper selection of the pertinent parameters to be captured by the simulation is intrinsically linked to the efficacy of said simulation. Developments on the toolbox and the procedures it entails are bound to generate a better comprehension of the priorities in algorithm development. One such development is the Genotype Clustering method, developed in this work to characterize the end scenarios and results of the simulated evolution processes. The method provided tools to verify the specialization of populations of simulated organisms to certain tasks, based on the conditions of their simulated world.

Finally, advances in the development of the toolbox can be made into open source repositories in platforms like github.com and filed under Copyleft distribution guidelines to foster collaboration and make it easier for individual, specific tools to be added and further elaborated.

In order to take full advantage of the various possible tools previously discussed, one must envision how and where to apply the algorithms they so handily facilitated creating. Besides the pure Soft A-Life context, biomimetic algorithms may be used to power subsystems of Hard A-Life. For this application, an algorithm is synthetically evolved under specific strains, resulting in an optimized solution for that strain type. If the algorithm is then structured as an instruction set to control one of the composing systems of a mobile robot or any other hardware-based apparatus, it could be switched on whenever the bot identifies that strain in the field. Such an architecture can be visualized as a modular compartmentalized system that has multiple operational algorithms for each subsystem – motor driving, walking gait, radar scanning pattern, odour plume tracking, visual processing, wind speed measurement, to name a few – and is able to identify ambient conditions and select the appropriate control algorithm for that moment, changing it on the fly. The modularity of the resulting autonomously adaptive system can cut costs in incremental firmware development by isolating components and developing each separately. Rules that define a clear layer hierarchy, like those used in Brooks' *Subsumption Architecture*, could ensure that no conflicts or incompatibilities arise between algorithms that were independently evolved. Some target applications beyond mobile autonomous robots include operation of subsystems in manned vehicles that run without user input but produce changes in user handling of the craft, like fly-by-wire in airplanes, adaptive driving modes and automatic gear shifting in cars and motorcycles, and even the development of intuitive controls for niche vehicles like lunar landers, jetpacks and powered wingsuits.

Concerning more unconventional research and industry fields, many of them still require some technological advancement to achieve the widespread popularity that comes with well understood working principles and low cost. Yet, it is possible to visualize how the tools and processes here discussed are applicable to some of them, including those not feasible in the foreseeable future. The study of such fields may or may not already employ tools from A-Life and robotics, so the applications here proposed ought to be expansive and exploratory. Considering the ever expanding technologies upon which these fields are built, it is pertinent to divide these application proposals into two groups: one relevant for technology already in use or in development, expected to become more universally adopted in the near future; the other regarding technology yet to come into fruition, only hypothetically explored today, such as advanced medicine, interstellar travel and planetary colonization.

Microbots working inside the circulatory system or other internal areas of a patient could perform microsurgeries on like-sized structures, such as removing blood clots, removing varicose veins, installing stents, delivering drugs in finely controlled fashion. One promising area is that of specific cell targeting, in which the robots could perform controlled chemical releases. The practice of externally induced thermal, mechanical or electromagnetic forces is evolving to become a field of medicine with high efficiency, relatively low cost, and low incidence of side-effects to the patient. The use of gold nanoparticles is one of such solutions currently under development, having been employed as transducers of electromagnetic excitation applied from outside the body to disrupt only the cells they are in contact with - usually cancerous cells.

If the microbots are equipped with microbial fuel cells, the ability to harvest energy sources from the environment could compound very well with the nutrient rich intra-bodily ambient. Taking advantage of this, and considering the appropriate sensors are developed or miniaturized enough, the robots could provide continuous, real-time monitoring of endogenous risk factors, amongst these are: blood contents, like glucose levels in diabetic patients, histamine levels in allergic patients, and physical properties such as blood pressure, dissolved oxygen, clotting ability and pH. From this point, a series of options unfolds: it could be possible to provide immediate intervention in the form of insulin or epinephrine release for the diabetes and allergy cases respectively, communicate physicochemical aspects relevant for athletes or chronic patients, or coordinate several robots in swarm behaviours for physical interventions, to name a few. For all these tasks, the combination of simple control algorithms based on foraging and/or chemosensing should be able to run the appropriate procedures.

Besides monitoring endogenous factors, the same logic applies to exogenous pathogens. Circulatory microbots could detect toxins from spoiled food, airborne viruses (such as zoonotic virus emerging in today's environment such as SARS, MERS and recently SARS-CoV-2) identify metabolites from infectious organisms, and act accordingly by releasing the correspondent serum or antidote in the stream, or simply by communicating via change in its visual properties, or by sending signals to an external device. Imagine being able to use these microbots to cover surfaces and PPE to be able to combat pathogens before they come into contact with a vector.

Perhaps, one day, microbots could be able to identify and promptly synthesize in-situ the solutions from the detected pathogens, replicating at a smaller scale the proficiency of a living organism's immune system. Moreover, a bio-inspired algorithm could be the key to the complex task of correctly combining the right organic molecules, a process that can be favourable when the reactants are abundantly available.

Another promising microrobotics-related area is *swarm sensing* - using a swarm of inter-communicating or independent robots as passive mobile sensors. Some uses include cases

such as robots being deployed into bodies of water, or from high altitudes into atmospheric systems. After being released in the desired area, the swarm travels through space at the same time it collects and relays relevant data. “Passive mobile” refers to the fact that, even though the robot is not equipped to apply forces to the environment in order to move itself, it is still able to experience motion by being carried, similarly to seed pods catching in the breeze or to tumbleweeds traveling through a desert Its design would be optimized to make the most of environmental features and phenomena. Sensors like accelerometers and gyroscopes could provide relative positioning and acceleration info for each robot. When coupled with communication capabilities or some level of spatial awareness of other members of the swarm, this method could produce an interesting effect: instead of a single sensor probe being moved around space to take sequential measurements and generate a 3D point cloud, each robot behaves as one of the data points itself, plotting its position through time and space while providing measurements by any other sensors onboard. The major difference is that the single probe produces a single time series of data points that are distributed through the period necessary for the whole measuring operation, while the swarm provides multiple concurrent measurement points for each timestep. The relatively fast clock cycle commanding the sensor reading, allied to the continuous data logging, takes the plot another step ahead and produces something that could be called a dynamic, real-time, or 4D plot. The robots can be inserted into the water stream, current, or tide by various means, including being individually placed from unmanned aerial or aquatic vehicles as well as dispersed in clusters.

Another proposal is the use of microbots as space probes. When traveling in swarms they could provide redundancy of data collected while also acting as data relays for each other. The autonomous operation of space probes could use optimized algorithms to cover a vast array of possibilities in their operation, therefore increasing mission success chances. This case is in accordance with Brooke’s idea[60] that, in a context where machine failures cannot be accessed for repair, a higher amount of smaller space probes equals a higher mission success rate. By employing swarms consisting of thousands of microbots this effect can be, itself, optimized.

To summarise, the applications of this work are as varied as nature itself, and as such this research is not meant to be viewed necessarily as a final destination but definitely step on the way to a greater use for minimalistic robots.

List of References

1. Atsumi, S., Higashide, W. & Liao, J. C. Direct photosynthetic recycling of carbon dioxide to isobutyraldehyde. *Nat. Biotechnol.* **27**, 1177–1180 (2009).

2. Fan, H., Zhao, N., Wang, H., Li, X. & Xu, J. One step preparation of polyaniline micro/nanohierarchical structures with superhydrophobicity. *Mater. Lett.* **78**, 42–45 (2012).

3. Wei, N., Quarterman, J. & Jin, Y.-S. Marine macroalgae: an untapped resource for producing fuels and chemicals. *Trends Biotechnol.* **31**, 70–77 (2013).

4. Carney, L. T. *et al.* Microbiome analysis of a microalgal mass culture growing in municipal wastewater in a prototype OMEGA photobioreactor. *Algal Res.* **4**, 52–61 (2014).

5. Philamore, H., Rossiter, J., Stinchcombe, A. & Ieropoulos, I. Row-bot: An energetically autonomous artificial water boatman. *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* 3888–3893 (2015) doi:10.1109/IROS.2015.7353924.

6. Lovley, D. R. Electrically conductive pili: Biological function and potential applications in electronics. *Curr. Opin. Electrochem.* **4**, 190–198 (2017).

7. Pessina, L.-A. Pinpointing sources of water pollution with a robotic eel. 1–4 https://actu.epfl.ch/news/pinpointing-sources-of-water-pollution-with-a-robo/ (2017).

8. Gross, L. Cultivating Bacteria's Taste for Toxic Waste. *PLoS Biol.* **4**, 1288–1289 (2006).

9. Schwager, M., Dames, P., Rus, D. & Kumar, V. A Multi-robot Control Policy for Information Gathering in the Presence of Unknown Hazards. in *Robotics Research* vol. 100 455–472 (Springer, 2016).

10. Blain, L. Steerable paper planes and maple seeds the basis for life-saving, disposable UAVs. *New Atlas* 1–9 https://newatlas.com/disposable-uav-paper-plane-maple-seed/28323/ (2013).

11. Xu, N. W. & Dabiri, J. O. Low-power microelectronics embedded in live jellyfish enhance propulsion. *Sci. Adv.* **6**, eaaz3194 (2020).

12. Chechetka, S. A., Yu, Y., Tange, M. & Miyako, E. Materially Engineered Artificial Pollinators. *Chem* **2**, 224–239 (2017).

13. Jang, B. *et al.* Undulatory Locomotion of Magnetic Multilink Nanoswimmers. *Nano Lett.* **15**, 4829–4833 (2015).

14. Lu, H. *et al.* A bioinspired multilegged soft millirobot that functions in both dry and wet conditions. *Nat. Commun.* **9**, 1–7 (2018).

15. Guo, Z. *et al.* Biocatalytic self-propelled submarine-like metal-organic framework microparticles with pH-triggered buoyancy control for directional vertical motion. *Mater. Today* **28**, 10–16 (2019).

16. Lee, S. *et al.* A Needle-Type Microrobot for Targeted Drug Delivery by Affixing to a Microtissue. *Adv. Healthc. Mater.* **9**, 1901696–1901697 (2020).

17. Ohayon, D. *et al.* Biofuel powered glucose detection in bodily fluids with an n-type conjugated polymer. *Nat. Mater.* **36**, 1–19 (2019).

18. Hwang, G. *et al.* Catalytic antimicrobial robots for biofilm eradication. *Sci. Robot.* **4**, eaaw2388 (2019).

19. Seyfried, J. *et al.* The I-SWARM Project: Intelligent Small World Autonomous Robots for Micro-manipulation. in *Advances in Artificial Life: Darwin Meets von Neumann. 10th*

European Conference, ECAL 2009. vol. 3342 70–83 (Springer Berlin Heidelberg, 2005).

20. Breger, J. C. *et al.* Self-Folding Thermo-Magnetically Responsive Soft Microgrippers. 1–8 (2015) doi:10.1021/am508621s.

21. Sivasankar, V., Mylsamy, P. & Omine, K. *Microbial Fuel Cell Technology for Bioelectricity.* (Springer, 2018).

22. Zhu, W. *et al.* 3D-Printed Artificial Microfish. *Adv. Mater.* **27**, 4411–4417 (2015).

23. Chen, A. S., Zhu, H., Li, Y., Hu, L. & Bergbreiter, S. A paper-based electrostatic zipper actuator for printable robots. *ICRA* 5038–5043 (2014) doi:10.1109/ICRA.2014.6907597.

24. Bi, C., Guix, M., Johnson, B. V, Jing, W. & Cappelleri, D. J. Design of Microscale Magnetic Tumbling Robots for Locomotion in Multiple Environments and Complex Terrains. *Micromachines* **9**, 68 (2018).

25. Qiu, T. *et al.* Swimming by reciprocal motion at low Reynolds number. *Nat. Commun.* **5**, 1–8 (2019).

26. Huang, H. *et al.* Adaptive locomotion of artificial microswimmers. *arXiv.org* **cond**-mat.**s**, eaau1532 (2019).

27. Kim, D., Hao, Z., Ueda, J. & Ansari, A. A 5 mg micro-bristle-bot fabricated by two-photon lithography. *J. Micromechanics Microengineering* **29**, 105006–105008 (2019).

28. Chen, Y. *et al.* Controlled flight of a microrobot powered by soft artificial muscles. *Nature* 1–22 (2019) doi:10.1038/s41586-019-1737-7.

29. Houston, A. I. & McNamara, J. M. *Models of Adaptive Behaviour.* (Cambridge University Press, 1999).

30. Viswanathan, G. M., Da Luz, M. G. E., Raposo, E. P. & Stanley, H. E. *The physics of foraging: an introduction to random searches and biological encounters.* (Cambridge University Press, 2011). doi:10.1017/CBO9780511902680.

31. Smith, T. M. & Smith, R. L. *Elements of Ecology.* (Benjamin-Cummings Publishing Company, 2012).

32. Benyus, J. M. *Biomimicry: Innovation Inspired by Nature.* (2002).

33. Bar-Cohen, Y. Biomimetics: Biologically Inspired Technologies. 1–579 (2005).

34. Bar-Cohen, Y. Nature as a Model for Mimicking and Inspiration of New Technologies. *Int. J. Aeronaut. Sp. Sci.* **13**, 1–13 (2012).

35. Vincent, J. F. V, Bogatyreva, O. A., Bogatyrev, N. R., Bowyer, A. & Pahl, A.-K. Biomimetics: its practice and theory. *J. R. Soc. Interface* **3**, 471–482 (2006).

36. Nachtigall, W. *Bionic as Science.* (Springer Berlin Heidelberg, 2010). doi:10.1007/978-3-642-10320-9.

37. Nachtigall, W. & Wisser, A. *Bionics by Examples.* (2015).

38. Gruber, P. *et al. Biomimetics - Materials, Structures and Processes.* (2011).

39. Gray, J. M., Hill, J. J. & Bargmann, C. I. A circuit for navigation in Caenorhabditis elegans. *Proc. Natl. Acad. Sci. U. S. A.* **102**, 3184–3191 (2005).

40. de Bono, M. Molecular approaches to aggregation behavior and social attachment. *J. Neurobiol.* **54**, 78–92 (2002).

41. Allen, E. N., Ren, J., Zhang, Y. & Alcedo, J. Sensory systems: their impact on C. elegans survival. *Neuroscience* **296**, 15–25 (2015).

42. Bargmann, C. I. Chemosensation in C. elegans. *WormBook* 1–29 (2006) doi:10.1895/wormbook.1.123.1.

43. WormAtlas. http://www.wormatlas.org (2018).

44. Frézal, L. & Félix, M.-A. C. elegans outside the Petri dish. *Elife* **4**, 381 (2015).

45. Emmons, S. W. The beginning of connectomics: a commentary on White et al. (1986) 'The structure of the nervous system of the nematode Caenorhabditis elegans'. *Philos. Trans. R. Soc. B Biol. Sci.* **370**, 20140309 (2015).

46. Boyle, J. H., Johnson, S. & Dehghani-Sanij, A. A. Adaptive Undulatory Locomotion of a C. elegans Inspired Robot. *IEEE/ASME Trans. Mechatronics* **18**, 439–448.

47. Boyle, J. H., Berri, S. & Cohen, N. Gait Modulation in C. elegans: An Integrated Neuromechanical Model. *Front. Comput. Neurosci.* **6**, 1–15 (2012).

48. Redko, V. G. Artificial Life Evolutionary Models. in *Principia Cybernetica Web* (eds. Heylighen, F., Joslyn, C. & Turchin, V.) 1–4 (1999).

49. Aguilar, W., Bonfil, G. S., Froese, T. & Gershenson, C. The Past, Present, and Future of Artificial Life. *Front. Robot. AI* (2014) doi:10.3389/frobt.2014.00008/abstract.

50. Taylor, C. E. & Jefferson, D. Artificial Life as a Tool for Biological Inquiry. *Artif. Life* **1**, 1–13 (1994).

51. Langton, C. G. Studying artificial life with cellular automata. *Phys. D Nonlinear Phenom.* **22**, 120–149 (1986).

52. Langton, C. G. *Artificial life : the proceedings of an Interdisciplinary Workshop on the Synthesis and Simulation of Living Systems*. (Redwood City, Calif. : Addison-Wesley Pub. Co., Advanced Book Program, 1989).

53. Turing, A. M. & Copeland, B. J. The Essential Turing: Seminal Writings in Computing, Logic, Philosophy, Artificial Intelligence, and Artificial Life, Plus the Secrets of Enigma. 1–622 (2006).

54. Copeland, J. Colossus: The Secrets of Bletchley Park's Code-breaking Computers (Popular Science). 1–495 (2006).

55. Thompson, D. W. *On Growth and Form*. (Cambridge University Press, 1992).

56. Beaudry, A. & Joyce, G. Directed evolution of an RNA enzyme. *Science (80-.).* **257**, 635–641 (1992).

57. Joyce, G. F. RNA evolution and the origins of life. *Nature* **338**, 217–224 (1989).

58. Rocha, L. M. *Von Neumann and Natural Selection* . (2013).

59. Brooks, R. A robust layered control system for a mobile robot. *IEEE J. Robot. Autom.* **2**, 14–23 (2001).

60. Brooks, R. A. & Flynn, A. M. Fast, Cheap and Out of Control: a Robot Invasion of the Solar System. *J. Br. Interplanet. Soc.* **42**, (1989).

61. Sullins, J. P. Ethics and Artificial life: From Modeling to Moral Agents. *Ethics Inf. Technol.* **7**, 139–148 (2005).

62. Dennett, D. Artificial Life as Philosophy. *Artif. Life* **1**, 291–292 (1994).

63. Spafford, E. H. Computer Viruses as Artificial Life. *Artif. Life* **1**, 249–265 (1994).

64. Gomi, T. *Evolutionary Robotics. From Intelligent Robotics to Artificial Life*. (Springer, 2003).

65. Bar-Cohen, Y. & Hanson, D. The Coming Robot Revolution: Expectations and Fears About Emerging Intelligent, Humanlike Machines. 1–180 (2010).

66. Hoggett, R. W. Grey Walter and his Tortoises. *cyberneticzoo.com* 1–3 (2011).

67. Hoggett, R. & Holland, O. W. Grey Walter Tortoises – Picture Gallery #2. *cyberneticzoo.com* 1–26 (2010).

68. Palm, G., Aertsen, A., Palm, G. & Aertsen, A. Brain Theory. 259 (2012) doi:10.1007/978-3-642-70911-1.

69. Dennett, D. C. & Braitenberg, V. Vehicles: Experiments in Synthetic Psychology. *Philos. Rev.* **95**, 137 (1986).

70. Searle, J. R. *The Rediscovery of the Mind*. (MIT Press, 1992).

71. Newell, A. & Simon, H. A. *Human Problem Solving*. (Echo Point Books & Media, 2019).

72. Pfeifer, R. & Scheier, C. *Understanding Intelligence*. (MIT Press, 2001).

73. Nilsson, N. J. The Physical Symbol System Hypothesis - Status and Prospects. *50 Years Artif. Intell.* **4850**, 9–17 (2006).

74. Brooks, R. A. A Robust Layered Control System for a Mobile Robot. *IEEE J. Robot. Autom.* **2**, 14–23 (1986).

75. Brooks, R. A. Elephants don't play chess. *Rob. Auton. Syst.* **6**, 3–15 (1990).

76. Simon, H. A. *The Sciences of the Artificial*. (MIT Press, 1996). doi:10.7551/mitpress/12107.001.0001.

77. Moravec, H. P. The Stanford Cart and the CMU Rover. in *Autonomous Robot Vehicles* vol. 22 407–419 (Springer New York, 1990).

78. Sims, K. Evolving virtual creatures. *the 21st annual conference* 15–22 (1994) doi:10.1145/192161.192167.

79. Lessin, D., Fussell, D. & Miikkulainen, R. Open-ended behavioral complexity for evolved virtual creatures. *GECCO* (2013).

80. Lessin, D., Austin, Fussell, D. & Miikkulainen, R. Adopting Morphology to Multiple Tasks in Evolved Virtual Creatures. *Artificial Life 14: Proceedings of the Fourteenth International Conference on the Synthesis and Simulation of Living Systems* 247–254 (2014) doi:10.7551/978-0-262-32621-6-ch041.

81. Lessin, D. G. Evolved virtual creatures as content: increasing behavioral and morphological complexity. (2015).

82. Lessin, D., Fussell, D. & Miikkulainen, R. Trading control intelligence for physical intelligence. *the 2014 conference* 705–712 (2014) doi:10.1145/2576768.2598290.

83. Tilden, M. The Design of 'Living' Biomech Machines: How low can one go? 1–23 (1999).

84. Hasslacher, B. & Tilden, M. W. Living machines. *Rob. Auton. Syst.* **15**, 143–169 (1995).

85. Hasslacher, B. & Tilden, M. W. Living Machines. 1–37 (1996).

86. Bekey, G. A. Autonomous Robots: from Biological Inspiration to Implementation and

Control. *Robotica* **24**, 271 (2006).

87. Association, W. N. Heat Values of Various Fuels. 1–2 https://world-nuclear.org/information-library/facts-and-figures/heat-values-of-various-fuels.aspx (2020).

88. Blanke, B. C., Birden, J. H., Jordan, K. C. & Murphy, E. L. *Nuclear Battery-thermocouple Type Summary Report*. http://www.osti.gov/servlets/purl/4807049-6bvOmJ/ (1960) doi:10.2172/4807049.

89. Bennett, G. L. *et al.* Mission of Daring: The General-Purpose Heat Source Radioisotope Thermoelectric Generator. *4th International Energy Conversion Engineering Conference and Exhibit IECEC, June , San Diego, California* 1–23 (2006).

90. Lanz, A., Heffel, J. & Messer, C. *Hydrogen Fuel Cell Engines and Related Technologies*. vol. 03 http://books.google.com.br/books?id=4lk0HQAACAAJ&dq=intitle:Hydrogen+Fuel+Cell+Engines+and+Related+Technologies&hl=&cd=1&source=gbs_api (2001).

91. Verhelst, S. & Wallner, T. Hydrogen-fueled internal combustion engines. *Prog. Energy Combust. Sci.* **35**, 490–527 (2009).

92. Schmidt-Rohr, K. How Batteries Store and Release Energy: Explaining Basic Electrochemistry. *J. Chem. Educ.* **95**, 1–10 (2018).

93. Crompton, T. R. & Crompton, T. P. J. *Battery Reference Book*. (Newnes, 2000).

94. Team, M. I. T. E. V. A Guide to Understanding Battery Specifications. 1–3 (2008).

95. Linden, D. & Reddy, T. B. *Handbook of batteries*. (McGraw-Hill Professional, 2002).

96. Iida, F., Gomez, G. & Pfeifer, R. Exploiting body dynamics for controlling a running quadruped robot. *ICAR'05. , 12th International Conference on Advanced Robotics, 2005* 229–235 doi:10.1109/ICAR.2005.1507417.

97. Doncieux, S. *New Horizons in Evolutionary Robotics*. (Springer Science & Business Media, 2011).

98. Sims, K. Evolving 3D Morphology and Behavior by Competition. *Artif. Life* **1**, 353–372 (1994).

99. Jones, M. T. *Artificial Intelligence*. (Infinity Science Press LLC, 2008).

100. Russell, S. J. & Norvig, P. *Artificial Intelligence*. (Prentice Hall, 2010).

101. Cronin, L. *et al.* The imitation game - a computational chemical approach to recognizing life. *Nat. Biotechnol.* **24**, 1203–1206 (2006).

102. Brownlee, J. Clever Algorithms. *Nature-Inspired Program. Recipes* 1–437 (2011).

103. Mo, H. Handbook of Research on Artificial Immune Systems and Natural Computing: Applying Complex Adaptive Technologies . 1–634 (2010).

104. Downing, K. L. *Intelligence Emerging*. (MIT Press, 2015). doi:10.2307/j.ctt17kk8tv.

105. Engelbrecht, A. P. *Computational Intelligence: An Introduction*. (2008).

106. Kita, E. *Evolutionary algorithms*. (2011).

107. Hansson, L.-A. & Åkesson, S. *Animal Movement Across Scales. Animal Behaviour* vol. 102 (Oxford University Press, USA, 2014).

108. Nyberg, D. Energy in Ecosystems. in 17–23 (2016).

109. Fox, C. W., Roff, D. A. & Fairbairn, D. J. *Evolutionary Ecology*. (Oxford University Press, 2001).

110. Dingle, H. *Migration: The Biology of Life on the Move*. (OUP Oxford, 2014).

111. Sparrevohn, C. R., Nielsen, A. & Støttrup, J. G. Diffusion of fish from a single release point. *Can. J. Fish. Aquat. Sci.* **59**, 844–853 (2002).

112. Kays, R., Crofoot, M. C., Jetz, W. & Wikelski, M. Terrestrial animal tracking as an eye on life and planet. *Science (80-.).* **348**, aaa2478–aaa2478 (2015).

113. Nagy, M., Akos, Z., Biro, D. & Vicsek, T. Hierarchical group dynamics in pigeon flocks. *Nature* **464**, 890–893 (2010).

114. King, A. J. *et al.* Selfish-herd behaviour of sheep under threat. *Curr. Biol.* **22**, R561–R562 (2012).

115. Bolnick, D. I. *et al.* Why intraspecific trait variation matters in community ecology. *Trends Ecol. Evol.* **26**, 183–192 (2011).

116. Crofoot, M. C., Gilby, I. C., Wikelski, M. C. & Kays, R. W. Interaction location outweighs the competitive advantage of numerical superiority in Cebus capucinus intergroup contests. *Proc. Natl. Acad. Sci.* **105**, 577–581 (2008).

117. Laidre, K. L. & Heide-Jorgensen, M. P. Spring partitioning of Disko Bay, West Greenland, by Arctic and Subarctic baleen whales. *ICES J. Mar. Sci.* **69**, 1226–1233 (2012).

118. Meyer, C. G., Papastamatiou, Y. P. & Holland, K. N. A multiple instrument approach to quantifying the movement patterns and habitat use of tiger (Galeocerdo cuvier) and Galapagos sharks (Carcharhinus galapagensis) at French Frigate Shoals, Hawaii. *Mar. Biol.* **157**, 1857–1868 (2010).

119. Wakefield, E. D. *et al.* Space partitioning without territoriality in gannets. *Science (80-.).* **341**, 68–70 (2013).

120. Horváth, G. *Polarized Light and Polarization Vision in Animal Sciences*. (Springer, 2014).

121. Muheim, R. Behavioural and physiological mechanisms of polarized light sensitivity in birds. *Philos. Trans. R. Soc. B Biol. Sci.* **366**, 763–771 (2011).

122. Wallraff, H. G. *Avian Navigation: Pigeon Homing as a Paradigm*. (Springer Science & Business Media, 2005). doi:10.1007/b137573.

123. Wehner, R., Boyer, M., Loertscher, F., Sommer, S. & Menzi, U. Ant navigation: one-way routes rather than maps. *Curr. Biol.* **16**, 75–79 (2006).

124. Boström, J. E., Åkesson, S. & Alerstam, T. Where on earth can animals use a geomagnetic bi-coordinate map for navigation? *Ecography (Cop.).* **35**, 1039–1047 (2012).

125. Collett, T. S. & Graham, P. Animal Navigation: Path Integration, Visual Landmarks and Cognitive Maps. *Curr. Biol.* **14**, R475–R477 (2004).

126. Broekmans, O. D., Rodgers, J. B., Ryu, W. S. & Stephens, G. J. Resolving coiled shapes reveals new reorientation behaviors in C. elegans: figures and supplements. **5**, 411–428 (2016).

127. Boyle, J. H. & Cohen, N. Caenorhabditis elegans body wall muscles are simple actuators. *Biosystems* **94**, 170–181 (2008).

128. Cohen, N. & Boyle, J. H. Undulatory Locomotion. *arXiv Prepr. arXiv0908.2769* (2009).

129. Kim, D., Park, S., Mahadevan, L. & Shin, J. H. The shallow turn of a worm. *J. Exp. Biol.* **214**, 1554–1559 (2011).

130. Donnelly, J. L. *et al.* Monoaminergic Orchestration of Motor Programs in a Complex C. elegans Behavior. *PLoS Biol.* **11**, e1001529-17 (2013).

131. Broekmans, O. D., Rodgers, J. B., Ryu, W. S. & Stephens, G. J. Resolving coiled shapes reveals new reorientation behaviors in C. elegans. *Elife* **5**, 411–428 (2016).

132. Carpin, S., Noda, I., Pagello, E. & Reggiani, M. *Simulation, Modeling, and Programming for Autonomous Robots.* (Springer Science & Business Media, 2008).

133. Arthur, D. & Vassilvitskii, S. k-means++: The Advantages of Careful Seeding. *SODA Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms* 1027–1035 (2007).

134. Coello, C. A. C., Lamont, G. B. & van Veldhuizen, D. A. Evolutionary Algorithms for Solving Multi-Objective Problems (Genetic and Evolutionary Computation). *Evol. Algorithms Solving Multi-Objective Probl. (Genetic Evol. Comput.* (2006).

135. Deb, K., Agrawal, S., Pratap, A. & Meyarivan, T. A Fast Elitist Non-dominated Sorting Genetic Algorithm for Multi-objective Optimization: NSGA-II. in *Advances in Artificial Life: Darwin Meets von Neumann. 10th European Conference, ECAL 2009.* vol. 1917 849–858 (Springer Berlin Heidelberg, 2000).

136. Deb, K., Pratap, A., Agarwal, S. & Meyarivan, T. A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Trans. Evol. Comput.* **6**, 182–197 (2002).

137. Goldberg, D. E. *Genetic Algorithms in Search, Optimization, and Machine Learning.* (1989).

138. Srinivas, N. & Deb, K. Multiobjective Optimization Using Nondominated Sorting in Genetic Algorithms. *Evol. Comput.* **2**, 221–248 (1994).