ULB

ECOLE
POLYTECHNIQUE
DE BRUXELLES

Automatic design of pheromone-based stigmergy in robot swarms

Thesis presented by Muhammad SALMAN
in fulfilment of the requirements of the PhD Degree in
Engineering Sciences and Technology
("Docteur en Sciences de l'ingénieur et technologie")
Année académique 2023-2024

Supervisor: Professor Mauro BIRATTARI
IRIDIA - Institut de Recherches Interdisciplinaires et de
Développements en Intelligence Artificielle

European Commission
Horizon 2020
European Union funding
for Research & Innovation

DEMIURGE
IRIDIA - Université libre de Bruxelles

erc
European Research Council
Established by the European Commission

In loving memory of my father Bashir Ahmed Akhtar, whose tireless efforts and deep commitment to our education have made this thesis a reality.

Automatic design of pheromone-based stigmergy in robot swarms

Muhammad Salman

IRIDIA, Université libre de Bruxelles, Belgium.

2024

Composition of the jury

Prof. Marco DORIGO (chair)

Research Director of the FRS-FNRS and co-director of IRIDIA, École polytechnique de Bruxelles, Université libre de Bruxelles, Belgium.

Prof. Emanuele GARONE (secretary)

Full Professor and co-director of SAAS, École polytechnique de Bruxelles, Université libre de Bruxelles, Belgium.

Prof. Mauro BIRATTARI (supervisor)

Research Director of the FRS-FNRS at IRIDIA, École polytechnique de Bruxelles, Université libre de Bruxelles, Belgium.

Dr. Gert RASKIN

Senior Research Engineer, Institute of Astronomy, KU Leuven, Belgium.

Dr. Ken HASSELMANN

Senior Researcher, Royal Military Academy, Brussels, Belgium.

Dr. Mary Katherine HEINRICH

Postdoctoral fellow at IRIDIA, École polytechnique de Bruxelles, Université libre de Bruxelles, Belgium.

The thesis

Through a modular automatic design process, robot swarms can take advantage of pheromone-based stigmergy to exhibit spatial organization, memory, and communication.

Summary

This dissertation aims to automatically design pheromone-based stigmergy in robot swarms, addressing two key challenges in swarm robotics: i) the automatic design of stigmergy-based collective behaviors, and ii) the real-world implementation of such systems.

While manual design of stigmergy-based behaviors in swarm robotics is possible, it is a complex and resource-intensive process. Designing collective behaviors for loosely coupled, autonomous robots is inherently difficult, and even more so when environmental modification is necessary. Optimization-based design provides an alternative, but could often requires some human input to fine-tune a produced control software. This dissertation explores a fully-automatic design method that is based on automatic modular design method, striving to eliminate the need for human intervention.

While stigmergy-based behaviors hold great promise for robot swarms, practical implementation faces challenges. Virtual stigmergy aids in simulation, but fails to capture the dynamic, environment-altering nature essential to its true potential. Physical methods like alcohol trails or external tracking systems can be expensive, complex, and potentially compromise robot autonomy. This research introduces `Phormica`, a cost-effective solution using special surfaces and UV LEDs to create artificial pheromones, enabling broader experimentation in real-world scenarios.

To demonstrate these concepts, I introduce `Habanero`, an automatic design method to design stigmergy-based control software. Using a swarm of e-puck robots, I evaluate `Habanero`'s performance across different missions and compared it to alternative design methods like neuroevolution-based design, manual design, and random-walk behavior. Our experimental results show that `Habanero` is a viable approach for pheromone-based stigmergy design. The produced behaviors are comparable to, sometimes outperforming, those created by human designers. Interestingly, while `Habanero`'s modules are mission-agnostic, the devised strategies are mission-specific. The automatic process effectively leverages pheromone-based

stigmergy to create collective behaviors demonstrating spatial organization, memory, and communication.

Contributions

This dissertation presents a number of original research contributions.

Review of the state of the art in swarm robotics: I present an overview of the latest advancements in optimization-based design methods for robot swarms. Specifically, my focus is on the offline optimization-based design approaches, encompassing automatic modular design strategies, neuro-evolutionary design techniques, and others.

Stigmergy-driven swarm robotics: a review of state of the art: I present a comprehensive review of the state-of-the-art in stigmergy-driven swarm robotics, exploring its theoretical foundations, implementation strategies, and applications. I presented a taxonomy of stigmergy-emulating technologies from the available literature. This taxonomy provides a framework to better understand the diverse ways robots can use stigmergy. Rather than being a set of strict rules, it serves as a tool to help researchers and designers think critically about existing systems. It facilitates comparison of different methods and highlights areas where advancements could be made.

`Phormica`: I have developed `Phormica`, a system to conduct experiments in swarm robotics that enables a swarm of e-puck robots to release and detect artificial pheromone. `Phormica` emulates pheromone-based stigmergy thanks to the ability of robots to project UV light on the ground, which has been previously covered with a photochromic material. As a proof of concept, I test `Phormica` on three collective missions in which robots act collectively guided by the artificial pheromone they lay in the target environment and detect.

`Habanero`: I introduce `Habanero`, a novel modular automatic design method to demonstrate that it is possible to generate pheromone-based collective behaviors through an automatic process that is repeatable and generally applicable.

Hardware and software implementations: I, with the coauthors of the related research articles, implemented and maintain the software for the methods presented in the thesis, notably `Phormica`, `Habanero`, `Waffel`, `EvoPheromone`, `Phormica` plugin for the ARGoS3 simulator. I also developed, manufactured and assembled different hardware add-ons for the e-puck robots and the target experiment environments. The software and hardware designs are available as open source material on the public repository of the DEMIURGE project.

`Waffel`: Additionally to the main contributions of this dissertation, I have also contributed to other research studies. In Appendix A, I present a research in which I studied the concurrent automatic design of control software and the automatic configuration of the hardware of robot swarms. I introduced Waffle, a new instance of the AutoMoDe family of automatic design methods that configures the robot hardware, selects the number of robots in the swarm, and produces control software in the form of a probabilistic finite state machine by combining pre-existing modules that are mission independent.

Statement

This thesis presents an original work that has never been submitted to the Université libre de Bruxelles or to any other institution for the award of a doctoral degree. Some parts of this thesis are based on a number of peer-reviewed articles that the author, together with other co-workers, has published in the scientific literature.

Part of the state of the art in Chapter 2 is based on:

- Birattari, M., Ligot, A., Bozhinoski, D., Brambilla, M., Francesca, G., Garattoni, L., Garzón Ramos, D., Hasselmann, K., Kegeleirs, M., Kuckling, J., Pagnozzi, F., Roli, A., **Salman, M.**, and Stützle, T. (2019). "Automatic off-line design of robot swarms: a manifesto". In: Frontiers in Robotics and AI 6, p. 59.

- Garzón Ramos, D., Bozhinoski, D., Francesca, G., Garattoni, L., Hasselmann, K., Kegeleirs, M., Kuckling, J., Ligot, A., Mendiburu, F. J., Pagnozzi, F., **Salman, M.**, Stützle, T., and Birattari, M. (2021). "The automatic off-line design of robot swarms: recent advances and perspectives". In: R2T2: Robotics Research for Tomorrow's Technology.

Chapter 3, `Phormica`, the photochromic pheromone release and detection system designed for swarm robotics research is based on:

- **Salman, M.**, Garzón Ramos, D., Hasselmann, K., and Birattari, M. (2020). "Phormica: photochromic pheromone release and detection system for stigmergic coordination in robot swarms". In: Frontiers in Robotics and AI 7, p. 195.

Chapter 4, the automatic design of pheromone-based stigmergy in robot swarms is based on:

- **Salman, M.**, Garzón Ramos, D., and Birattari, M. (2024). "Automatic design of stigmergy-based behaviours for robot swarms". In: Communications Engineering 3.1, p. 30.

Annex A is based on the research contribution which the author together with co-authors has published in the scientific literature:

- **Salman, M.**, Ligot, A., and Birattari, M. (2019). "Concurrent design of control software and configuration of hardware for robot swarms under economic constraints". In: PeerJ Computer Science 5, e221.

Implementations of software/hardware tools used throughout the work are released as open source software and are described in:

- Garzón Ramos, D., **Salman, M.**, Ubeda Arriaza, K., Hasselmann, K., and Birattari, M. (2022). MoCA: a modular RGB color arena for swarm robotics experiments. Tech. rep. TR/IRIDIA/2022-014. IRIDIA, Université libre de Bruxelles.

- Hasselmann, K., Ligot, A., Francesca, G., Garzón Ramos, D., **Salman, M.**, Kuckling, J., Mendiburu, F. J., and Birattari, M. (2018). Reference models for AutoMoDe. Tech. rep. TR/IRIDIA/2018-002. IRIDIA, Université Libre de Bruxelles.

- Legarda Herranz, G., Rochala, P., Georgiopoulou, P., Ligot, A., **Salman, M.**, and Birattari, M. (2022b). "BRIC: an interactive smart object for swarm robotics research". In: TR/IRIDIA/2022-015.

The author declares that the research presented in this thesis is their original work. To enhance the writing and readability of the text, they employed large language model-based software. This approach allowed for independent refinement of the work, without the assistance of traditional editors.

Acknowledgments

I would like to acknowledge the DEMIURGE project, funded by the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation program (grant agreement No 681872), for their support.

I am deeply grateful to Professor Mauro Birattari for his exceptional mentorship, support, and guidance throughout my PhD journey. His brilliance and passion have greatly inspired me, shaping my growth as both a researcher and an individual. The opportunities he provided and the knowledge he shared have been instrumental in my development.

I extend my sincere gratitude to my jury members. Their valuable expertise and insightful feedback have played a crucial role in refining and improving this thesis.

Big shout-out to everyone at IRIDIA! Working alongside Prof. Marco Dorigo, Prof. Hugues Bersini, and Prof. Thomas Stutzle was definitely a highlight. IRIDIA is not just a lab with a bunch of smart people, it is also surprisingly fun—which is more than I can say for most research groups. To Leslie, Alberto, Mary Katherine, Andreagiovanni, Christian, Weixu (Harry), Marcolino, Volker, Guillaume, Garazi, Alex, and everyone else I inevitably forgot to mention - thanks for making it memorable.

And of course, to the DEMIURGE project team: Federico Pagnozzi, Ken Hasselmann, Antoine Ligot, Fernando Mendiburu, David Garzon Ramos, Jonas Kuckling, Miquel Kegeleirs, Guillermo Legarda Herranz, and Ilyes Gharbi. Your ideas were not always helpful, but they definitely kept things interesting. And hey, I guess I learned something along the way.

Three years back, I hopped onto a new career adventure, and I gotta give a shout-out to my colleagues at the Institute of Astronomy, KU Leuven. Thanks for welcoming me, having my back, and giving me that extra push to cross the PhD finish line!

A massive thank you to my family. Your support and investment in my education

mean the world. To my parents, sisters, and brother: your love, encouragement, and unwavering belief in me have been the bedrock of everything I have achieved. This would not have been possible without you. To my father, you waited six years for this day. I wish you could have been with us just a few more months to witness this accomplishment. I hope you are in a better place, happy and proud of what I have achieved.

To my beloved wife Sarah, words cannot fully capture the depth of my gratitude for your support, understanding, and love throughout this journey. Your contribution to this thesis goes far beyond what any partner could be expected to give. During the challenging COVID lockdown, when our apartment became an arena to perform experiments with robot swarms, you endured this disruption for nearly two years with remarkable patience and grace. Your willingness to share our living space with my research, even in the midst of a global pandemic, speaks volumes about your belief in me and my work.

To my little sunshine, Armelle Salman, though you are too young to understand this now, your giggles and smiles have been the sweetest soundtrack to my work. The past two years have been filled with the most incredible joy because of you. Thank you for being my constant reminder of what truly matters in life.

Salman

Contents

List of Figures

Chapter 1

Introduction

From the creation of the earliest stone tools to the advancement of today's sophisticated technologies, humans have continuously been inspired by nature. They have mimicked its complexity and mechanisms not just to address challenges but also to drive innovation in technology. This enduring fascination has driven significant advancements across multiple disciplines. These include architecture, medicine, mechanical engineering, and material science.

An early example of this human-nature connection can be seen in the development of tools and weapons. Early humans observed the sharp teeth and claws of animals and adapted them to create their own cutting tools. Similarly, the intricate design of spider webs may have inspired early forms of net weaving, revolutionizing trapping techniques and fishing.

This pattern of observing and learning from nature has persisted throughout history, leading to the development of innovative technologies. The shape of wings of birds inspired the design of airplanes, while the camouflage techniques of animals like chameleons and octopuses inspired the development of stealth technology. Even the microscopic structure of lotus leaves has inspired the design of self-cleaning surfaces.

In the field of robotics, bio-inspired innovation has played a particularly significant role. By studying the movement and behavior of animals, researchers have developed robots that can perform tasks that were once thought to be the exclusive domain of living organisms. Bio-inspired designs in robotics manifests in various forms, from mimicking biological locomotion to emulating animals' sensory systems. Here are some specific examples of how nature has inspired robotics. *Biological locomotion:* robots have been designed to mimic the movement of animals such as insects, fish, and mammals. This has led to the development of robots that can

walk, swim, and fly with greater agility and efficiency. *Sensory systems:* robots have been equipped with sensors that mimic the senses of animals, such as vision, hearing, and touch. This has allowed robots to interact with their environment more effectively and gather information about their surroundings. *Cognitive abilities:* robots are being developed with artificial intelligence that allows them to make decisions, operate autonomously, and adapt to their environment. This is inspired by the cognitive abilities of animals, such as learning, memory, and communication. Figure 1.1 shows two bio-inspired robots, highlighting recent advancements in robotics.

Image credit: BostonDynamics

Image credit: Engineered Arts

Figure 1.1: **Examples of Bio-Inspired Robotics. Left:** Boston Dynamics SPOT, a quadrupedal robot inspired by the agility and adaptability of dogs. **Right:** Engineered Arts Ameca, a humanoid robot demonstrating advanced facial expressions and mimicking human-like interaction. These innovative robots illustrate how inspiration from nature drives advancements in fields such as locomotion, sensory systems, and cognition.

Nature's remarkable character is not limited to individual organisms. It also encompasses the complex collective intelligence manifested by various species. This collective intelligence, seen in creatures like bees, ants, birds and fish, demonstrates how group of individuals can work together in sophisticated ways to achieve goals beyond an individual's capacity. This remarkable phenomenon has inspired the development of swarm robotics, where multiple robots collaborate in a distributed way, resembling the coordinated behavior of social insects like ants or termites. In a robot swarm, robots are equipped with simple hardware and work collectively to accomplish complex missions that are beyond what can be achieved by a single robot. They typically operate in a fully decentralized manner, and the individual robots do not have access to global information. Robots gather information about their environment locally through their sensors or short-range communication devices (Dorigo et al. 2014; Şahin 2005).

Recent advancements in swarm robotics are opening up exciting possibilities in various fields. Land-based swarms: research is focused on enabling these swarms to navigate complex terrains, making them ideal for search and rescue missions (Li et al. 2018). Task sequencing: robot swarms demonstrate efficient task sequencing capabilities, showing promise in fields like assembly and logistics (Garattoni and Birattari 2018). Adaptability: ongoing research aims to create robot swarms capable of reshaping themselves, offering adaptability for various environments. This could revolutionize disaster relief and infrastructure construction (Abdel-Rahman et al. 2022; Xie et al. 2019). Underwater exploration: robot swarms are being equipped with unique communication strategies, making them invaluable for marine exploration and research (Berlinger et al. 2021). Aerial applications: aerial swarms are demonstrating advancements in collaborative mapping, dynamic obstacle avoidance, and distributed sensing, making them increasingly valuable for search and rescue, precision agriculture, and environmental monitoring (Soria et al. 2021). Space-based swarms: the recent launch of the Starling Swarm satellite network highlights the growing potential of swarms in space.

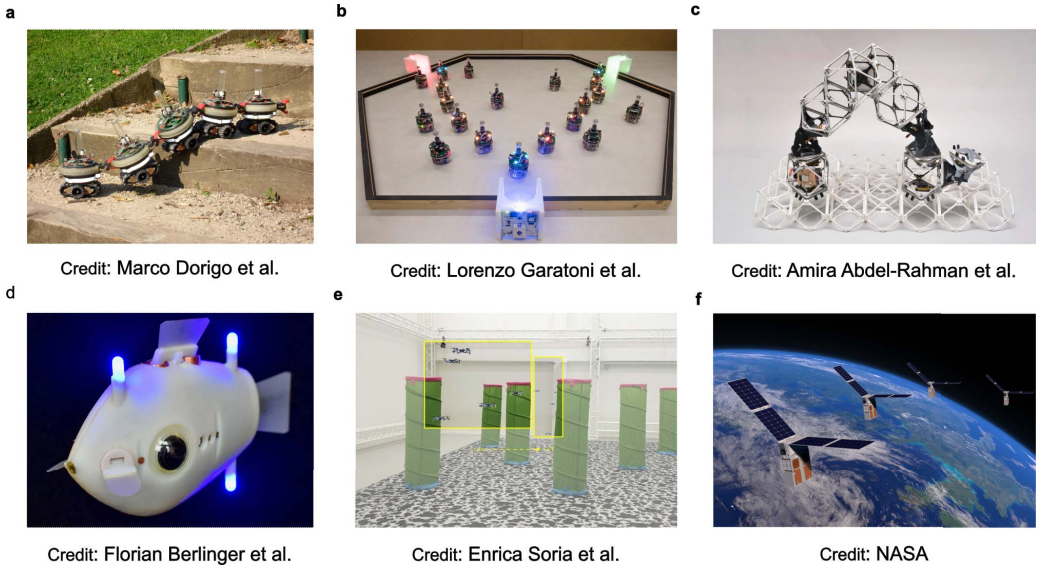


Figure 1.2: **Innovations in Swarm Robotics. a.** A swarm-bot demonstrates chain configuration for tackling obstacles. **b.** E-puck robots engage in task sequencing, illustrating swarm capabilities in logistics and assembly. **c.** A modular assembler robot demonstrates the potential of swarms in construction tasks. **d.** A bluebot showcases biomimicry and implicit communication in underwater swarms. **e.** Drone-swarm navigate a cluttered environment, highlighting advances in collaborative obstacle avoidance. **f.** NASA's Starling CubeSat swarm illustrates autonomous maneuvers for the future of space-based swarm applications.

Communication plays a crucial role in developing coordination in robot swarms, enabling robots to work together effectively without centralized control. Robots can share information about their environment, such as obstacles, targets, and potential hazards. This leads to better decision-making and adaptation. Communication also helps robots coordinate their movements. They can navigate complex environments, form patterns, and avoid collisions. Task allocation is another benefit of communication; robots can assign roles for efficient work. Finally, communication promotes self-organization, allowing swarms to change and adapt their behavior based on the environment.

In swarm robotics, two forms of communication are utilized: direct and indirect. Direct communication involves the active exchange of information between agents at line-of-sight. This can be achieved through infrared messaging, radio, sound, or visual signals like color. Several strategies are employed within direct communication. Broadcasting: where all robots receive all messages, ensuring widespread information distribution. Local communication: where robots communicate with immediate neighbors for consensus-building. Leader-follower communication: where leaders guide followers, aiding task allocation and movement coordination. While direct communication and fixed environment markings have been instrumental in swarm robotics, there are limitations to these approaches. In scenarios where robots are unable to communicate directly due to obstacles or the vastness of the environment, or their communication range is limited, coordination can be disrupted, potentially jeopardizing mission success. To address these challenges, researchers have turned to nature for inspiration, drawing upon the ingenious communication method employed by social insects: stigmergy.

Stigmergy (Grassé 1959; Heylighen 2016b; Wilson 1975) is a coordination mechanism in which agents self-organize through indirect local communication mediated by the environment. When using stigmergy, agents leave indications of their presence or actions in the environment and this stimulates/inhibits the behaviors of their peers (Wyatt 2014). Some animals physically transform the environment thus producing visual cues that influence their peers. For instance, humans leave footprints on the ground and flatten vegetation while walking in the wild, thereby creating visually detectable paths that others tend to follow (Helbing et al. 1997). Other animals secrete chemicals that their peers can detect and to which they react—for instance, Argentine ants lay pheromone trails that are then followed by nestmates (Goss et al. 1989).

For many social insects, pheromone-based stigmergy plays an important role in self-organization (Theraulaz and Bonabeau 1999). These insects can sense

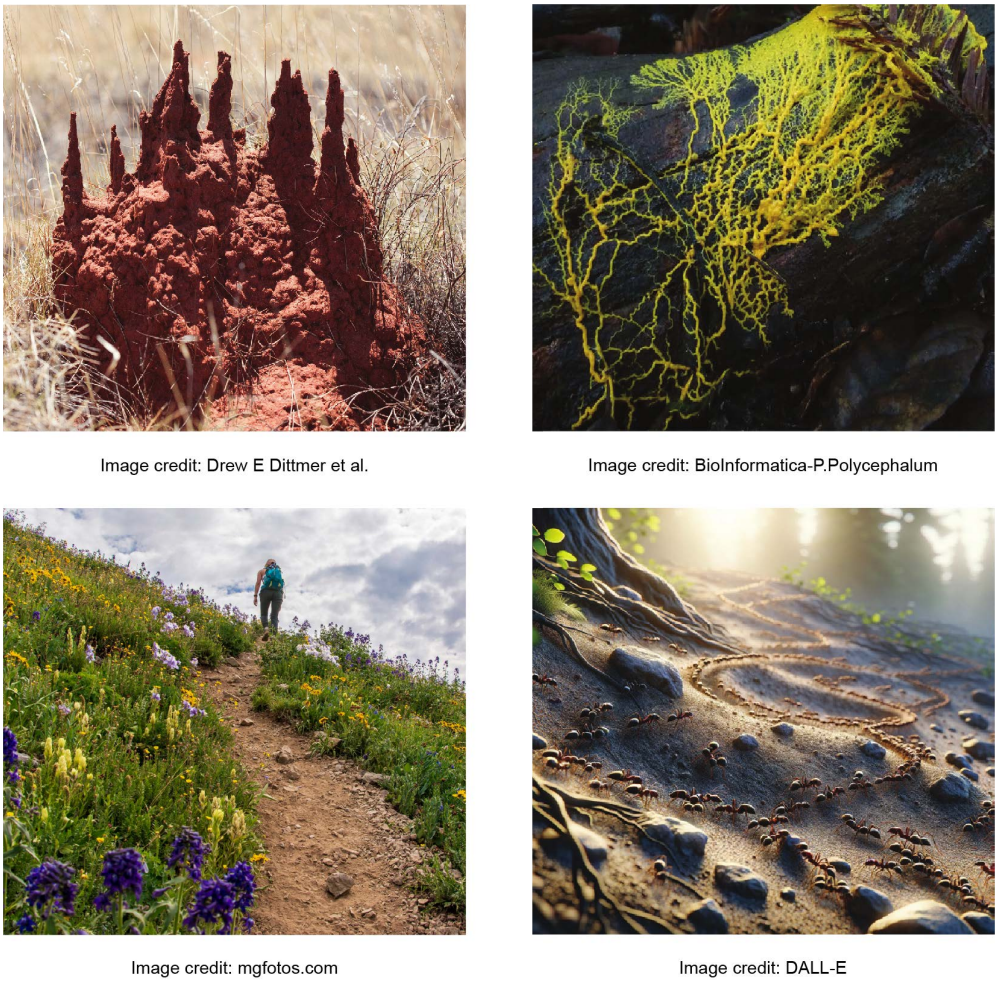


Figure 1.3: **Stigmergy across life-forms.** Top left: Termite mound built through indirect coordination using pheromones. Top Right: Slime mold optimizing its foraging network through trace signaling. Bottom Left: Humans leave trails, unintentionally influencing others' decisions. Bottom Right: Ants using pheromone-based stigmergy for foraging.

environmental features, locally interact with other members of the colony and with the environment, and process information to make decisions (Bonabeau et al. 1999). However, they have short perception and communication ranges, are not aware of the global state of the colony, are unable to remember their actions, and are unable to plan their contributions to the collective activities of the colony (Bonabeau et al. 1999). The pheromones laid in the environment function as a collective and distributed memory: they effectively encode the state of the colony. The pheromones enable coordination, as the individuals can work together and self-organize without the need to communicate directly or receive instruction on the tasks they must perform (Feinerman and Korman 2017; Theraulaz et al. 2003).

A robot swarm, like an insect colony, can use pheromone-based indirect communication mediated by the environment (Garnier et al. 2009). Designers of robot swarms can develop pheromone-based interaction strategies for specific missions. However, giving real robots the capability to mark the environment with indications of their activities is still an open technological challenge (Corne et al. 2012). In some studies, researchers have developed smart environments to enable pheromone-based stigmergy, for instance, by using: (i) a system of stationary devices (e.g., RFID tags) spread throughout the environment to store virtual pheromones (Alfeo et al. 2019; Campo et al. 2010; Goss et al. 1992; Khaliq et al. 2014; Payton et al. 2001), (ii) devices to display or project virtual pheromones on the ground (Garnier et al. 2013; Hunt et al. 2019; Na et al. 2020, 2019), or (iii) augmented reality to immerse the robots in a virtual environment in which they can lay and sense pheromones (Antoun et al. 2016; Reina et al. 2017; Talamali et al. 2020). These systems are flexible, powerful, and enable the implementation of complex coordination mechanisms. However, as these systems rely on external infrastructures (for tracking robots, displaying the pheromones, and storing information), they can be expensive and are only suitable under restricted conditions.

Alternatively, several approaches to physically deposit artificial pheromones have been proposed, using specialized onboard actuators to lay trails of alcohol or wax, without the assistance of any external infrastructure (Fujisawa et al. 2014; Russell 1997, 1999). However, these solutions would be impractical in most real-world applications due to the hazards of using flammable material (alcohol) or heating devices (for melting wax).

To address the issue, I have recently proposed a hardware module for robots that project UV light downwards, laying an artificial pheromone trail on ground that has previously been coated with photochromic material (Salman et al. 2020b). The part of the ground that is exposed to UV light changes in color from white to magenta. Once the UV light is removed, the magenta color fades back to white, in about 50 s, mimicking the evaporation of a pheromone. This approach does not present safety risks and does not rely on complex or expensive infrastructure, however, it still requires the environment to be prepared before deploying the robots.

The technological problem of endowing robots with the ability to lay and sense artificial pheromones is not the only problem to be addressed. The concept of stigmergy is not easily understood intuitively (Heylighen 2016a) and therefore, designing collective behaviors based on stigmergy is itself a challenge. Even without using stigmergy, designing any collective behavior for a robot swarm is already

complex: individuals are autonomous and loosely coupled, and the interactions between individuals and between them and the environment become fully defined only at run time (Brambilla et al. 2013; Winfield et al. 2005). The design problem becomes even more complex if the interaction strategies that enable coordination are regulated by modifications to the environment. No formal design method exists to tell under what conditions and in what amount individuals should release the pheromone, nor how they should react to pheromone trails so that a desired collective behavior emerges.

The design problem in swarm robotics is indeed particularly difficult because a robot swarm is a collective entity and cannot be programmed directly: the designer must program the individual robots so that, together, they perform the desired mission. The design process is laborious because of the complex relation existing between the behavior of the individual robots and the collective behavior that results from their interactions (Brambilla et al. 2013). The most common approach to designing a robot swarm is trial-and-error: a time consuming approach in which individual-level behaviors are implemented, tested, and modified until the desired swarm-level behavior is obtained. Similarly, pheromone-based stigmergy has also predominantly been designed manually to address specific missions under specific conditions (Hamann and Wörn 2007; Khaliq and Saffiotti 2015; Talamali et al. 2020). Although a number of swarms have been successfully designed with this approach, the quality of the results obtained via manual design is not consistent, and greatly depends on the experience of the designer. Typically, a manual design process is not easily repeatable. It is also not directly generalizable to other robotic platforms or missions, even if they are similar (Francesca and Birattari 2016). To overcome these issues, a number of principled manual design methods have been proposed. However, these methods are limited in scope: a universal swarm design methodology does not exist, yet (Brambilla et al. 2014; Hamann and Wörn 2008; López-Ibáñez et al. 2016).

Optimization-based design, or automatic design, is an alternative approach to designing a swarm. In automatic design, the design problem is formulated as an optimization problem that is then solved with an optimization algorithm (Birattari et al. 2019). A design problem of a collective mission is expressed as an objective function, a mathematical equation that measures the performance of the robot swarm. An optimization algorithm steers the search for the control software of the individual-robot that maximizes the performance of the swarm, taking into account the constraints such as hardware limitations of the robots or other environmental restrictions, that are encoded in the form of additional (in)equalities.

Automatic design methods can be classified based on whether the method is applied in an *semi-automatic* or *automatic* manner (Birattari et al. 2020). In *semi-automatic* design, a human designer actively participates in the optimization process. This method, often used offline, allows the designer to apply their expertise in fine-tuning various design parameters through the optimization software, aiming for the best possible outcome. The human expert has the flexibility to modify any aspect of the design process, including the control software architecture, optimization algorithm parameters, simulation models, and the performance measures, based on the evaluation of the swarm's behavior with the current design and the specific requirements of the mission. In *automatic* design, the method remains unchanged and per-mission manual adjustments are not allowed. To be considered fully-automatic, a method must be capable of solving a range of missions, not just a specific one, and its performance must be evaluated across a class of missions without human intervention in any phase of the design. In this context, the design process is completely predetermined and runs autonomously.

Automatic design methods can also be classified into two distinct categories, differentiated by whether the design process is conducted *online* or *offline* (Birattari et al. 2020). In *online* methods, the design process takes place in real-time on robots once they are deployed in the target environment. A distributed optimization algorithm runs directly on the robots themselves. This algorithm continuously fine-tune the control software based on the robots' performance while they execute their mission. This design approach has several drawbacks. For instance, during the early phases of design process, there is a risk of damaging the robots or the environment when robot execute sub-optimal control software. Moreover, the real-time execution of design candidate control software by the robot swarm makes the entire process inefficient and time-consuming. Additionally, the reliance on an external infrastructure that may not be available in all situations introduces an additional layer of complexity and potential limitations. In *offline* methods, the design process takes place in computer simulation prior to the deployment of robot swarm in the target environment. Simulations allow for a safe and cost-effective evaluation of the control software, without the risk of any material damage. Furthermore, simulations can be executed much faster than real-time, which significantly reduces the time required for the design process. Additionally, simulations can be parallelized, which means they can be run simultaneously on multiple computer processors. This further accelerates the simulation process and allows for even faster design iterations.

However, there is an intrinsic and unavoidable difference between the simulations

on the basis of which control software is produced, and the real environment in which the control software is eventually executed. This discrepancy between simulations and reality is often referred to as the 'reality gap'.

The reality gap can arise from various factors, such as the accuracy of the simulation model, the limitations of the simulation software, and the inability to fully replicate the complexity and variability of the target environment (Ligot and Birattari 2020, 2022; Ligot et al. 2022). This highlights the importance of carefully evaluating the results of simulations and conducting thorough testing in real-world conditions.

The reality gap is not a universal problem, but rather one that affects different design methods differently. Francesca et al. 2014b drew an analogy between the reality gap and overfitting in machine learning. They asserted that the reality gap does not arise from the simplicity of the simulator but from the design process inadvertently adapting to certain idiosyncrasies of the simulator, leading to poor performance in real-world scenarios. They likened robustness to the reality gap to the bias-variance trade-off in machine learning, a concept that describes the relationship between bias and variance in learning algorithms. Bias and variance are known to be correlated with the complexity of learning algorithms: typically, high-complexity algorithms have high variance and low bias, whereas low-complexity ones have low variance and high bias. Just as reducing the complexity of machine learning models can improve generalization, restricting the design space of control software can enhance its robustness to the reality gap. High-complexity design methods, such as traditional neuroevolutionary approaches, are more prone to overfitting and thus are more sensitive to the reality gap. To address this challenge, Francesca et al. 2014b proposed a strategy that limits the design process to assembling predefined software modules. This modular design approach reduces the range of possible input-output mappings, thereby decreasing the likelihood of control software overfitting the simulator and enhancing its robustness to the reality gap.

In this thesis, I focus on the automatic design of stigmergy-based collective behaviors for robot swarms. I present a modular automatic off-line design method that belongs to the AutoMoDe family (Birattari et al. 2021). In AutoMoDe, as is customary in automatic off-line design (Birattari et al. 2019; Francesca and Birattari 2016), the design problem is reformulated as an optimization problem that is solved in simulation, prior to the deployment of the robots in their target environment (Birattari et al. 2019, 2020). The solution space of the optimization problem comprises instances of control software that can be obtained by selecting and combining pre-existing software modules (i.e., low-level behaviors and the

conditions to transition between them) into a modular architecture (e.g., finite-state machines, behavior trees) and by tuning the free parameters (Francesca et al. 2014b). Once the optimization process is completed, the selected control software is uploaded to the robots without undergoing any manual transformations, and the robots are eventually deployed in the target environment.

AutoMoDe is a general framework. To define a specific design method that conforms to it and produces control software to address a specific class of missions, the following steps must be taken: (1) select a target robot platform that is appropriate for the given class of missions, (2) define software modules for the selected robot platform, (3) specify the architecture into which the software modules will be assembled, (4) select a simulator to be used in the automatic design process, and (5) define an appropriate optimization algorithm to search the space of the possible ways in which the software modules can be assembled and tuned. My proposed AutoMoDe method, `Habanero`, designs collective behaviors to address missions in which the robot swarm relies on stigmergy to coordinate. The target robot platform is the e-puck (Mondada et al. 2009) augmented with the Overo Gumstix Linux board, the aforementioned hardware module that lays artificial pheromone trails by focusing UV light onto ground coated with photochromic material (Salman et al. 2020b), and an omni-directional camera to detect artificial pheromone trails.

In this thesis, I demonstrate `Habanero` by generating control software for a swarm of eight e-puck robots. I consider four collective missions in which the robots should rely on stigmergy-based coordination: Aggregation, Decision Making, Rendezvous Point, and Stop. To assess the quality of the control software produced by `Habanero`, I compare its performance to that of several alternative design methods: (1) control software produced via neuroevolution (`EvoPheromone`), (2) control software manually produced by human designers (`Human-Designers`), and (3) a random-walk behavior (`Random-Walk`).

The results of the experiment indicate that: (i) `Habanero` is a viable automatic approach to designing pheromone-based stigmergy; (ii) it can produce control software that is comparable to, or even outperforms, control software produced by a human designer; and (iii) although its modules are conceived in a mission-agnostic way, the interaction strategies it devises are mission-specific.

The following is the structure of the thesis. In Chapter 2, I present an overview of the latest advancements in optimization-based design methods for robot swarms. Specifically, my focus is on the offline optimization-based design approaches, encompassing automatic modular design strategies, neuro-evolutionary design techniques,

and others. I further present a comprehensive review of the state-of-the-art in stigmergy-driven swarm robotics, exploring its theoretical foundations, implementation strategies, and applications. In particular, I present a taxonomy of stigmergy-emulating technologies from the available literature. This taxonomy provides a framework to better understand the diverse ways robots can utilize stigmergy. Rather than being a set of strict rules, it serves as a tool to help researchers and designers think critically about existing systems. It facilitates comparison of different methods and highlights areas where advancements could be made. Furthermore, I examine the design principles and challenges associated with developing effective systems for emulating stigmergy for swarm robotics applications. In Chapter 3, I propose `Phormica`, a system to conduct experiments in swarm robotics that enables a swarm of e-puck robots to release and detect artificial pheromone. In Chapter 4 I present `Habanero`, a novel modular automatic design method to demonstrate that it is possible to generate pheromone-based collective behaviors through an automatic process that is repeatable and generally applicable. In Chapter 5, I conclude the thesis with a summary of the contributions and a discussion on future research directions. In Appendix A, I present a research in which I studied the concurrent automatic design of control software and the automatic configuration of the hardware of robot swarms.

Chapter 2

State of the art

This chapter gives an overview of the swarm robotics literature, specifically focusing on the state-of-the-art advancements in two key areas of swarm robotics: the design of robot swarms and the development of pheromone-based stigmergy enabling technologies for collective missions.

Firstly, I focus on the design of robot swarms, investigating the fundamental ideas, and techniques that have been implemented to address the swarm design challenges. The focus will be on optimization-based methods, which offer a promising solution to the current lack of general methodology for the design of robot swarms. Advancements in this area hold the key to the widespread adoption of swarm robotics in situations where conventional methods prove inadequate. If you'd like a deeper dive into swarm robotics, I recommend Hamann 2018, Garattoni and Birattari 2016, Brambilla et al. 2013, and Schranz et al. 2020. For insightful perspectives on the current state and future trajectory of swarm robotics, Dorigo et al. 2021 provide valuable resources.

Secondly, the chapter introduces a taxonomy of methods that enable robots within a swarm to emulate stigmergy. This will be analyzed from three angles: Medium of information transfer: how is information passed within the swarm? Mode of interaction in stigmergy: what are the different ways stigmergy is used? Implementation focus: where is the main emphasis in applying stigmergy? For a historical context of stigmergy, I recommend Theraulaz and Bonabeau 1999. To learn more about stigmergy as coordination mechanism, I refer to Heylighen 2016a, and Heylighen 2016b.

2.1 Swarm robotics

Swarm robotics initially took inspiration from biological swarms like insects or bird flocks. However, the field has broadened significantly and has become an independent engineering discipline. Swarm engineering aims to create arbitrary collective behaviors and not simply replicate swarm-like phenomena observed in nature (Brambilla et al. 2013). Due to their adaptable nature and decentralized control, robot swarms are highly promising for accomplishing complex tasks in real-world scenarios. (Mathews et al. 2017; Rubenstein et al. 2014; Werfel et al. 2014; Xie et al. 2019). Industrial and broader societal applications are anticipated within the next 10 to 15 years (Dorigo et al. 2021).

A robot swarm is defined by its decentralized, self-organized, and highly redundant nature. Unlike systems with centralized control, there is no external entity controlling the swarm's actions. This allows the desired swarm-level behaviors to emerge from local interactions between robots and their environment, which promotes scalability and robustness (Brambilla et al. 2013; Şahin 2005).

Individual robots in a swarm often have specialized roles, but multiple robots are typically capable of performing any given task. This redundancy ensures that the swarm continues functioning even if some individual robots fail. To achieve this, robots within a swarm have relatively simple hardware and software. Each robot operates primarily on local sensing and communication, unaware of the swarm's overall size or global conditions. This characteristic is key to scalability, allowing a swarm to behave similarly whether composed of tens, hundreds, or even thousands of robots

Swarm robotics focuses on how groups of simple robots can achieve complex collective behaviors through local interactions. This emphasis on decentralized operation and robustness makes swarm robotics ideal for real-world tasks where traditional control systems might struggle. To understand how these systems work, following are some basic ways robot swarms achieve complex collective missions.

Aggregation: robots gather together in a specific area. This can be the first step in tasks like cooperative movement or decision-making. Sometimes, they have to select the best aggregation site out of several options. Aggregation is seen in nature too (honeybees or cockroaches).

Dispersion: the opposite of aggregation, where robots spread out to cover an area effectively. This requires them to sense each other and is often linked to tasks like exploration, coverage or surveillance.

Foraging: robots search for items and bring them back to a nest. Researchers

focus on efficient search strategies, sometimes forming chains or using various signals to guide robots in a swarm. It can also involve task-allocation, where some robots search while others transport.

Collective decision making: robots must agree on a choice out of several options, and this gets tricky as they only have local information. Tasks like foraging or finding an aggregation site might involve decision-making about which option is best.

These collective behaviors in swarm robotics emerge from the local interactions between individual robots. This interaction can take two forms: direct interaction, where robots explicitly exchange information, or indirect interaction (stigmergy), where robots modify the environment and others react to those changes. Direct communication offers rapid and precise coordination but can become complex in large swarms. Conversely, stigmergy can provide a more scalable and robust approach, as robots primarily react to their environment rather than coordinating with many individual teammates. This decentralized coordination makes swarm robotics adaptable and resilient to individual failures.

Designing any collective behavior for a robot swarm is complex: individuals are autonomous and loosely coupled, and the interactions between individuals and between them and the environment become fully defined only at run time (Brambilla et al. 2013; Winfield et al. 2005). The design problem becomes even more complex when the robots need to coordinate by modifying their environment, for example, through pheromone trails. No systematic design method exists to determine under what conditions and in what amount individual robots should release the pheromone; and how individuals should react to pheromone trails to achieve collective behaviors.

Understanding the complex relationship between individual robot behaviors and the resulting swarm-level collective behavior is the fundamental challenge in swarm robotics. This complexity underscores the importance of the next step: developing robust design methods that can effectively guide this intricate process of programming a swarm (Brambilla et al. 2013).

2.2 Design methods

The most common way to design a robot swarm still relies heavily on trial-and-error. Designers implement behaviors for individual robots, test the swarm, and then modify those behaviors until they achieve the result they want. This is time-consuming, and even the most experienced designers may need many attempts to

get it right. This approach also is commonly used to design systems which emulate pheromone-based stigmergy. In these systems, researchers manually fine-tune the rules for pheromone release and response. These adjustments are tailored to the specific task and the environment the robots will operate in. (Hamann and Wörn 2007; Khaliq and Saffiotti 2015; Talamali et al. 2020).

While some successful swarms have been created this way, the quality of the results from manual design can be inconsistent. The results are not easily reproducible, and the process does not always transfer well to different robots or tasks, even if they seem similar (Francesca and Birattari 2016).

To address these problems, researchers have proposed many principled design methods. However, these methods are often limited in what they can achieve, and a truly universal swarm design methodology does not exist, yet (Brambilla et al. 2014; Hamann and Wörn 2008; López-Ibáñez et al. 2016).

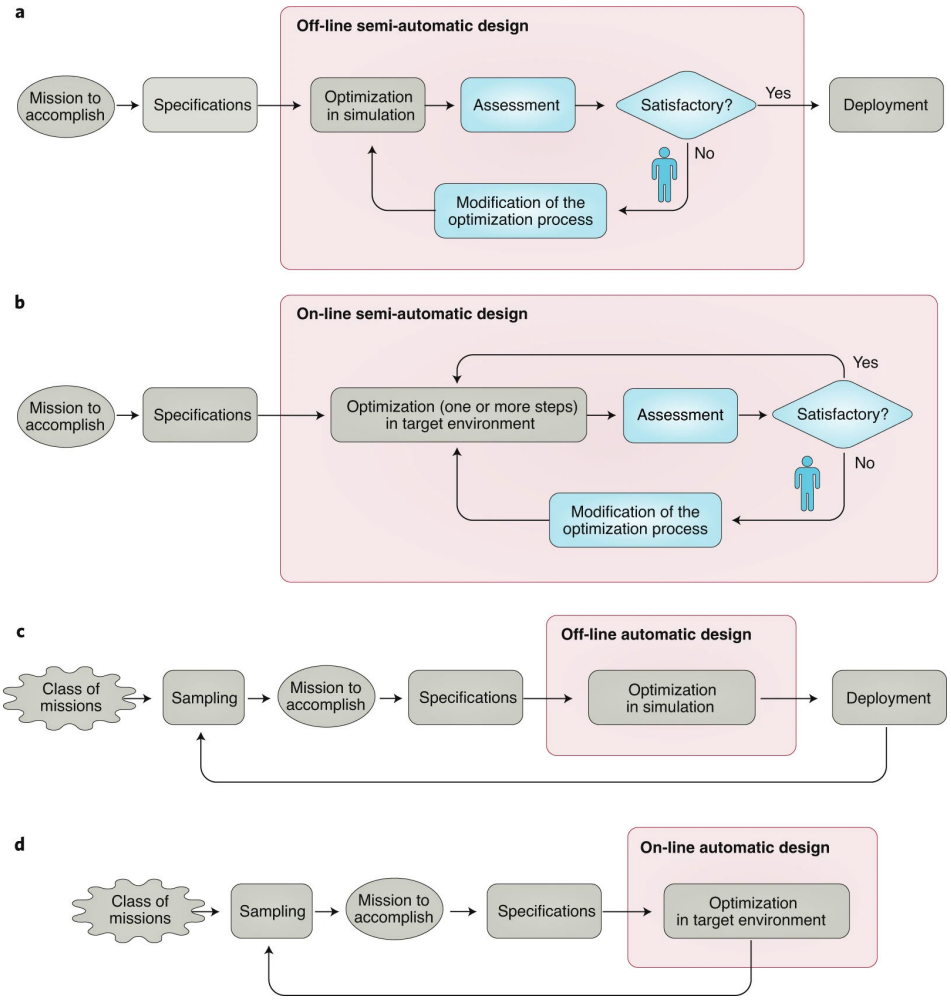
2.3 Automatic design

Optimization-based design, or automatic design, offers an alternative approach to designing robot swarms. In automatic design, the design problem is formulated as an optimization problem and solved using an optimization algorithm (Birattari et al. 2019). Within this automatic design framework, a collective mission is expressed as an objective function, a mathematical equation that measures the performance of the robot swarm. An optimization algorithm searches for individual-robot control software that maximizes the swarm's performance. This search, however, must operate within constraints. These constraints reflect real-world limitations, such as the capabilities of the robot hardware or specific requirements of the swarm's operating environment. Constraints are also expressed mathematically.

We can classify optimization-based design approaches for robot swarms based on two factors: where the design takes place (offline vs. online) and the level of human involvement (semi-automatic vs. automatic). The available literature can be classified into four categories based on the combination of the above-mentioned factors. For more details, I refer readers to Figure 2.1 (originally published by Birattari et al. 2020).

2.3.1 Semi-automatic and fully-automatic design

Optimization-based design methods can be classified depending on whether the method is applied in a *semi-automatic* or *fully-automatic* manner (Birattari et al.



Source: Birattari et al. 2020

Figure 2.1: **Four classes of optimization-based design.** We can classify design approaches for robot swarms based on two factors: where the design takes place (offline vs. online) and the level of human involvement (semi-automatic vs. automatic). This leads to four distinct categories. **(a)** offline semi-automatic: design happens in simulation, with the designer actively guiding or shaping the process. **(b)** Online semi-automatic: design occurs directly on the deployed robots, but the designer still influences how the optimization proceeds. **(c)** Offline automatic: design occurs entirely in simulation, requiring minimal human input. **(d)** Online automatic: robots autonomously adapt and refine their control software within the real-world environment, without direct human intervention. To best of my knowledge, no research on online fully-automatic design is publicly available.

2020).

Semi-automatic design

In semi-automatic design, a human designer actively participates in the optimization process. This method, often used offline, leverages the designer's expertise for fine-tuning outcomes. The designer can modify various design aspects, including control software, optimization algorithm settings, simulation models, and performance evaluation. This adaptability stems from their understanding of the task and the swarm's current behavior.

Typically, the designer runs the optimization process with existing settings and then carefully analyzes the resulting swarm behavior using simulations and/or robotic experiments. Based on this analysis and mission requirements, they adjust optimization settings for subsequent iterations. This cycle continues until the designer is satisfied with the control software or determines that further improvement is unlikely.

Semi-automatic design is a useful tool for exploring the potential of design methods in a research context. Moreover, in the context of expensive swarm robotic systems where trial-and-error approaches are financially prohibitive, semi-automatic design methods offer a cost-effective way to explore a wide design space, optimize system parameters, and reduce the risk of costly design failures. However, it shares drawbacks with manual design in practical settings. Both approaches lack reproducibility and are heavily dependent on the designer's individual skill.

Fully-automatic design

In a fully-automatic design process, human intervention is unwanted. Once the initial settings (like the optimization algorithm, simulation models, and the objective function) are defined, the design process runs autonomously.

To be considered fully-automatic, a design method must be capable of solving a range of missions, not just a single niche one. Consistent performance across different mission types, without human modification, is the hallmark of a truly automatic design.

2.3.2 Online and offline design

Optimization-based design methods can also be classified into two distinct categories, differentiated by whether the design process is conducted *online* or *offline* (Birattari

et al. 2020).

Online design

Online design methods offer a dynamic approach to swarm robotics where the robots' control software undergoes continuous refinement while operating within their target environment. This contrasts with offline methods, where robots rely on simulations to optimize behavior. Instead, online methods allow robots to actively adapt in real-time, often using distributed optimization algorithms. One powerful technique is embodied evolution, where each robot becomes part of a larger evolutionary process, continuously modifying its own software in response to performance (Watson et al. 1999).

This grants robot swarms incredible flexibility, as they autonomously adjust their behaviors to handle unexpected conditions. Additionally, online methods avoid the "reality gap"—the risk that offline designs may fail in unpredictable real-world settings due to simulation inaccuracies. The distributed nature of embodied evolution further allows multiple robots to optimize in parallel, potentially accelerating the design process.

While promising, embodied evolution within swarm robotics is still a developing field. In notable early research, Bianco and Nolfi 2004 created a framework where robots could spontaneously self-assemble and evolve without needing a specific objective. Their setup involved robots operating in a simple environment, each controlled by a basic feed-forward neural network. They demonstrated the potential of this approach through simulations involving 64 robots in three different scenarios.

More recent research investigates the application of "social learning" algorithms to distributed robotic systems, specifically swarms. This approach diverges from centralized machine learning models by having robots operate in a decentralized manner. The primary goal of social learning in swarms is to optimize task performance in real-time, after deployment (Bredeche and Fontbonne 2021).

Several studies further demonstrate the viability of online design methods. Pugh and Martinoli 2009 successfully employed particle swarm optimization for online obstacle avoidance control software design in a Khepera robot swarm, demonstrating its viability as an alternative to evolutionary algorithms and highlighting its adaptability to varying swarm sizes and communication ranges. König et al. 2009 employed embodied neuroevolution and finite-state machines to design collision avoidance and gate passing behaviors for a 26-robot Jasmine IIIp swarm, and their simulated experiments showed that specific genome recombination strategies and

increased parentage improved performance. Silva et al. 2015 developed odNEAT, a distributed neuroevolution algorithm based on NEAT, and demonstrated its performance on par with offline rtNEAT in three simulated robot swarm missions (aggregation, obstacle avoidance, and phototaxis), while also showing superior sensor fault tolerance.

However, online methods have drawbacks. Initially, control software is likely to be suboptimal, posing potential risks of damage to robots or their surroundings. Additionally, limited adaptation time means these methods work best when exploring smaller search spaces or tasks where individual robots can adequately assess the swarm's overall performance.

Offline design

In offline design, robot swarm control software is developed entirely within computer simulations. A simulated environment offers significant advantages. Simulations are much faster and cheaper than real-world testing, allowing designers to explore a vast range of software possibilities in a short time. Unlike physical experiments, simulations provide a comprehensive overview of the entire swarm's behavior, enabling the use of diverse performance metrics. Moreover, simulations create a safe environment for evaluating early designs. Since these initial designs might be suboptimal, using simulations eliminates the risk of damaging robots or the environment during testing.

Optimization-based techniques are common in offline swarm robotics design. Among the most well-established approaches is evolutionary swarm robotics (Nolfi 2021; Trianni 2008). This method employs simulated trials to iteratively fine tune robot control software, enabling swarms to excel in tasks such as foraging, transport, and pattern formation (Brambilla et al. 2013; Schranz et al. 2020).

Neuroevolutionary swarm robotics enjoys particular popularity within this field (Floreano et al. 2008; Lipson 2005; Trianni 2008). In this approach, the control software of robots is based on artificial neural networks. These networks are then meticulously optimized through evolutionary algorithms, resulting in sophisticated swarm behaviors.

Other offline approaches emphasize modularity, where robot swarm control software is assembled from a library of low-level behavioral building blocks. These modules can be generated automatically, hand-designed by engineers, or created through a hybrid approach (Duarte et al. 2016, 2015; Ligot et al. 2020a). Techniques like automatic modular design further enrich the toolkit available for offline

optimization-based designs. This approach enables the creation of complex robot swarms by combining simpler, reusable modules. Additionally, multi-agent reinforcement learning contributes to this toolkit. It's a machine learning approach where multiple agents learn to cooperate and achieve a common goal through trial and error. This means that we can leverage these techniques to create and optimize the design of robot swarms in simulation before deploying them in the real world (Gharbi et al. 2023; Ligot et al. 2020a).

While undeniably powerful, offline design must contend with the inherent reality gap. The reality gap is a relative problem; it's not just about recreating reality within a simulation but also about the broader difference between the training environment and the deployment environment (Hasselmann et al. 2021; Ligot and Birattari 2020). This discrepancy means control software that excels in simulation may perform poorly when deployed on physical robots. Bridging this gap, and carefully crafting objective functions that reliably guide simulations to produce the desired real-world swarm behaviors, remain key challenges within the field (Ligot 2023).

2.3.3 Neuroevolution

In evolutionary swarm robotics, robots' control software is created through an artificial evolutionary process, inspired by natural evolution. Evolutionary algorithms, tailored for a specific mission, evaluate a controller's performance using an objective (or fitness) function, based on predefined metrics. This function measures how well a controller performs based on specific goals. Think of it like digital survival of the fittest: poorly performing controllers are discarded, while the best ones are selected, bred together, and subtly changed until the robots achieve the desired behavior or the maximum time for the process runs out. Typically, every robot within the swarm runs the same control software.

The objective function is the heart of this process—it is the scorecard that determines which controllers are winners and which are losers. Those that perform poorly are eliminated, while successful designs are 'recombined' and slightly modified (mutated) to create new possibilities for testing. By continually exploring a range of control parameters, this process aims to gradually evolve better and better results over time.

Neuroevolution is a pioneering optimization technique within swarm robotics (Dorigo et al. 2003; Trianni et al. 2003). In this approach, an evolutionary algorithm takes on the role of the designer, crafting artificial neural networks that act as the

robots' control software. The key advantage lies in its independence from expert knowledge—the evolutionary algorithm optimizes the neural network without needing in-depth understanding of the specific task at hand. This flexibility has been a focus of early neuroevolutionary design methods (see Brambilla et al. 2013 for a comprehensive review).

The SWARM-BOTS project is a prominent example of the successful use of evolutionary swarm robotics. The project aimed to explore innovative design and implementation techniques for self-organizing and self-assembling robotic systems. Researchers successfully developed complex behaviors like collective alignment, hole avoidance, and self-organizing synchronization by employing evolutionary techniques within simulations. These controllers were later deployed with equal success in both simulated environments and on physical "s-bot" robots equipped with gripper mechanisms. The SWARM-BOTS project demonstrated the power of evolutionary swarm robotics to create complex real-world behaviors.

Interest has recently surged in the systematic use of neuroevolution to design control software for diverse robotic platforms, with a strong focus on creating systems ready for real-world deployment. Duarte et al. 2016 successfully used NEAT (Stanley and Miikkulainen 2002) to develop control software that enabled swarms of aquatic robots to perform tasks like homing and dispersion. Similarly, Gomes et al. 2019 generated control software allowing cooperative foraging robot swarms composed of both aerial and ground robots.

A significant contribution in this field is the comprehensive analysis by Hasselmann et al. 2021. They compared various offline neuroevolution techniques, including NEAT (Stanley and Miikkulainen 2002), CMA-ES (Auger and Hansen 2005), xNES (Glasmachers et al. 2010), and EvoStick, specifically investigating how these approaches perform within the context of common swarm robot missions.

The persistent challenge in neuroevolution is the reality gap. The reality gap in neuroevolution refers to the discrepancies that arise when transferring behaviors developed in a simulated environment to a real-world context. This challenge stems from the inherent limitations of simulations in perfectly replicating the intricacies and unpredictability of the real world (Hasselmann et al. 2021; Ligot and Birattari 2020). Recognizing this, Birattari et al. 2019 emphasized the critical need to test offline, optimization-based design methods on real robots, not just in simulations. Building on this, Hasselmann et al. 2021 directly investigated how the reality gap affects various neuroevolution strategies. They found that without mitigation strategies like incorporating noise into simulations, or adjustments specific to the mission, even sophisticated neuroevolutionary methods perform no better on real

robots than a basic perceptron network.

2.3.4 Reinforcement learning

Multi-agent reinforcement learning (MARL) has emerged as a promising alternative to traditional design methods for automatically creating controllers in robot swarms. Reinforcement learning (RL) allows robots to learn desired behaviors through trial and error, guided by a reward function that optimizes their controllers. Early work by Matarić 1994, 1997 demonstrated RL's potential in swarm robotics, despite challenges like complex state spaces and a centralized approach. Later research explored communication strategies for decentralized RL, addressing a key challenge within multi-agent environments.

Applications of RL in multi-robot systems with small numbers of agents have shown notable success. Chen et al. 2017 presented decentralized collision avoidance algorithms for up to six robots, with performance improving as task complexity increased. Hu et al. 2020 developed a superior multi-robot navigation system using RL for use in realistic domestic settings. Hüttenrauch et al. 2019 pioneered the use of Deep Reinforcement Learning (DRL) within the context of swarm robotics, addressing the issue of dynamic state spaces. While simulation results are encouraging, real-world validation of these techniques remains limited.

Despite the benefits of stigmergy in robot swarms, automatic design methods for developing their control software remain strikingly rare. In fact, there's only one documented instance where deep reinforcement learning was used to generate collision avoidance behavior based on virtual pheromones. While this simulation-based study demonstrated the potential superiority of control software designed through deep reinforcement learning compared to manual methods, it relies on a centralized infrastructure to store and distribute pheromone information. This approach, though a step towards automatic design of stigmergy-based behaviors in virtual environments, has limitations when it comes to real-world scenarios. It cannot be directly applied where robots must physically lay and sense artificial pheromones within their environment (Na et al. 2020).

One major challenge in MARL is the curse of dimensionality. As the swarm size or complexity of the environment increases, the number of possible states and actions the system needs to consider explodes exponentially. This makes it incredibly difficult to find optimal solutions, requiring extensive training time and massive amounts of data. Solutions learned in one scenario might perform poorly when even small changes are introduced, limiting the adaptability of MARL

approaches.

Another key obstacle is the credit assignment problem. When multiple agents achieve a goal, it's difficult to pinpoint which individual actions were most responsible for the success. This fuzzy reward system can lead to unstable learning, where agents focus on safe actions rather than riskier maneuvers with potentially higher rewards. It also becomes harder to accurately evaluate agent performance as the swarm grows, hindering the scalability of MARL frameworks to larger-scale problems.

2.3.5 Reality gap

Reality gap an intrinsic and unavoidable difference in simulations on the basis of which control software is produced, and the real environment in which the control software is eventually executed.

The reality gap can arise from various factors, such as the accuracy of the simulation model, the limitations of the simulation software, and the inability to fully replicate the complexity and variability of the target environment. This highlights the importance of carefully evaluating the results of simulations and conducting thorough testing in real-world conditions.

The reality gap is not a universal problem, but rather one that affects different design methods differently. Francesca et al. 2014b drew an analogy between the reality gap and overfitting in machine learning. They asserted that the reality gap does not arise from the simplicity of the simulator but from the design process inadvertently adapting to certain idiosyncrasies of the simulator, leading to poor performance in real-world scenarios. They likened robustness to the reality gap to the bias-variance trade-off in machine learning, a concept that describes the relationship between bias and variance in learning algorithms.

Bias and variance are known to be correlated with the complexity of learning algorithms: typically, high-complexity algorithms have high variance and low bias, whereas low-complexity ones have low variance and high bias (Francesca 2017). Just as reducing the complexity of machine learning models can improve generalization, restricting the design space of control software can enhance its robustness to the reality gap. High-complexity design methods, such as traditional neuroevolutionary approaches, are more prone to overfitting and thus are more sensitive to the reality gap.

To address this challenge, Francesca et al. 2014b proposed a strategy that limits the design process to assembling predefined software modules. This modular

design approach reduces the range of possible input-output mappings, thereby decreasing the likelihood of control software overfitting the simulator and enhancing its robustness to the reality gap (Kegeleirs et al. 2024).

For a more comprehensive exploration of the reality gap and its implications in robot swarm design, I refer readers to Ligot 2023. This work provides a detailed analysis of the challenges posed by the reality gap, along with potential strategies and approaches to mitigate its effects and enhance the transferability of simulated behaviors to real-world environments.

2.3.6 Automatic modular design

In modular design, instead of relying solely on artificial neural networks as control software, automatic methods utilize multiple software modules to create a more complex control architecture. These modules, often designed by human experts with valuable domain knowledge, can include finite-state machines and behavior trees. Think of these modules as software components, each responsible for a basic robot behavior like "move", "stop", or "follow a color". During the design process, an algorithm carefully assembles these modules, much like building blocks, into a larger structure such as a probabilistic finite-state machine or a behavior tree.

Modular design offers flexibility. Modules can be handcrafted by experts, generated automatically (for example, using neural networks), or utilize a combination of both approaches. Their specific implementation depends on the robot's capabilities and the tasks it needs to perform. Importantly, the integration of domain knowledge not only boosts controller performance but also helps bridge the reality gap.

Ferrante et al. 2013 explored the use of grammatical evolution for designing robot swarm control software in a foraging scenario where robots needed to coordinate to accomplish the collective mission. Their approach breaks down complex behaviors into basic behavioral modules, like moving towards a resource or returning to the nest. Using grammatical evolution, an offline algorithm intelligently combines these basic modules into a control software.

Jones et al. 2018 demonstrated the potential of behavior trees as a control architecture for evolved swarm robots, with an emphasis on human readability. Applying their approach to a foraging task, they manually designed mission-specific behavioral modules. These modules were then assembled into behavior trees using an offline evolutionary algorithm and tested on real Kilobots (Rubenstein et al. 2014).

Hecker et al. 2012 investigated how genetic algorithms could optimize the

behavior of foraging robot swarms. Inspired by ant behavior, they designed a hand-made finite-state machine as the control architecture. Using an agent-based simulation model, they employed a genetic algorithm to fine-tune parameters within the finite-state machine related to environment and simulated pheromone based coordination.

Gomes and Christensen 2018 explored the use of a quality diversity algorithm to evolve general-purpose swarm behaviors, independent of any specific task. Instead of optimizing for a single mission, their approach used task-agnostic behavior metrics to create a repertoire of controllers with diverse characteristics. Notably, when directly compared to controllers evolved for specific missions, controllers from the repertoire achieved comparable or even superior results in five out of eight tasks. Furthermore, they demonstrated that a single repertoire held the potential to solve multiple tasks and contained a variety of potentially useful, and diverse control solutions.

Francesca et al. 2014a introduced the AutoMoDe, a versatile family of methods for the automatic modular design of robot swarm. The initial AutoMoDe-Vanilla approach focused on the automatic assembly of hand-crafted, pre-defined behavioral modules into probabilistic finite-state machines. A powerful advantage demonstrated by modular methods within the AutoMoDe family is the increased resilience to the reality gap. Over the years, the AutoMoDe framework has been continually refined, leading to the development of several distinct and innovative methods (Francesca et al. 2015; Hasselmann and Birattari 2020; Ligot et al. 2020b; Mendiburu et al. 2022; Salman et al. 2019a).

Automatic modular design is a compelling alternative to neuroevolution for designing robot swarm controllers. Early results demonstrate similar performance, and better adaptability when deployed from simulation to real-world environments. The trade-off is that this approach requires careful design of the modules themselves. Neuroevolution can theoretically work with any type of robot input and output, whereas modular design relies on pre-made building blocks by a human. This limits the tasks the swarm can successfully handle.

Recent studies, however, have sought to overcome this limitation by automating the module design process. For instance, Ligot et al. 2020a and Hasselmann et al. 2023 introduced automatic modular design methods where the software modules were also designed automatically in the form of neural networks. Notably, Hasselmann et al. 2023 introduced `Nata`, a novel modular method that combines task-agnostic behaviors generated through novelty search. This approach requires less human intervention than previous methods and produces control software more

robust to the reality gap. While `Nata`'s performance is promising, exceeding some existing methods, future research is required to improve its robustness and identify the types of missions it can effectively handle. Importantly, `Nata` is the first fully automatic design method for robot swarm control that has itself been generated automatically.

The tricky part, however, still lies in the module design itself, even when automated. The complexity and quantity of these modules directly determine which problems the swarm can solve effectively. How they are structured and work together fundamentally influences both the swarm's overall performance and its ability to successfully transition from simulation to real-world applications.

2.4 Stigmergy

Stigmergy is a coordination mechanism in which agents self-organize through indirect local communication mediated by the modification of the environment. The concept of stigmergy was first introduced by the French entomologist Pierre-Paul Grassé (Grassé 1959) to describe a unique form of indirect communication he observed in termites. This communication happens through changes made to the environment. When an individual performs a task, it leaves a mark or trace in the environment. This mark then serves as a cue, stimulating the next task to be performed, either by the same individual or another one. This process ensures tasks are carried out in the correct sequence without the need for planning, supervision, or even direct interaction between the individuals involved (Heylighen 2016a).

These environmental changes can manifest in various ways, but one prevalent example is the use of pheromones. Pheromones, primarily derived from fatty acids, esters, and aldehydes, can also include isoprenoids and other compounds, and typically exist as complex mixtures rather than pure substances (Howse et al. 2013). Social insects like ants, termites, bees, and others have evolved sophisticated communication systems heavily reliant on pheromones (Karlson and Lüscher 1959). These chemical signals trigger various behaviors, including foraging, defense, and even caste determination within the colony (Baracchi et al. 2017; Chalissery et al. 2019). Social insects are exceptional in their use of pheromones, with the chemicals acting as an externalized memory for the group despite limited memory within individual insects (Denny et al. 2001). This allows for coordinated behavior and trail optimization.

For instance, Argentine ants use pheromone trails to locate the shortest path

to food (Goss et al. 1989). Pharaoh's ants release multiple pheromones for efficient food foraging in ever-changing environments. Bumblebees mark depleted flowers to optimize the colony's efforts (Eltz 2006). Ant queen pheromones play a crucial role in colony regulation, influencing worker sterility and overall colony stability (Holman 2018). These examples highlight the significant role of pheromones in social insect communication and success.

2.4.1 Stigmergy in swarm robotics

Stigmergy holds immense potential for swarm robotics. This concept, inspired by social insects, allows large groups of relatively simple robots to achieve complex behaviors by leaving and sensing "traces" within their environment.

To fully harness the power of stigmergy, we need a clear way to classify the different methods that enable a robot swarm to mimic these interactions using pheromone-based stigmergy. A structured taxonomy can serve as a road-map, helping researchers choose the right approach for their specific robot swarm and collective mission.

Here, I present a taxonomy based on three primary axes. Please refer to Figure 2.2 to view the taxonomy of stigmergy-enabling methods in swarm robotics in a hierarchical manner. The advantages and disadvantages of different stigmergy enabling methods are summarized in Table 2.1.

- **Medium of information transfer:** how is the information pertaining to stigmergy represented?
- **Mode of interaction in stigmergy:** how do robots interact with the information?
- **Implementation focus:** what are the main task domains for a specific method?

Medium of information transfer

- **Physical:**
  - **Chemical:** Researchers have explored the use of chemical-based artificial pheromone trails for robot communication. These trails often consist of alcohol detected by specialized chemical sensors on the robots. Robots can be equipped with sensors to detect both odorous chemicals and

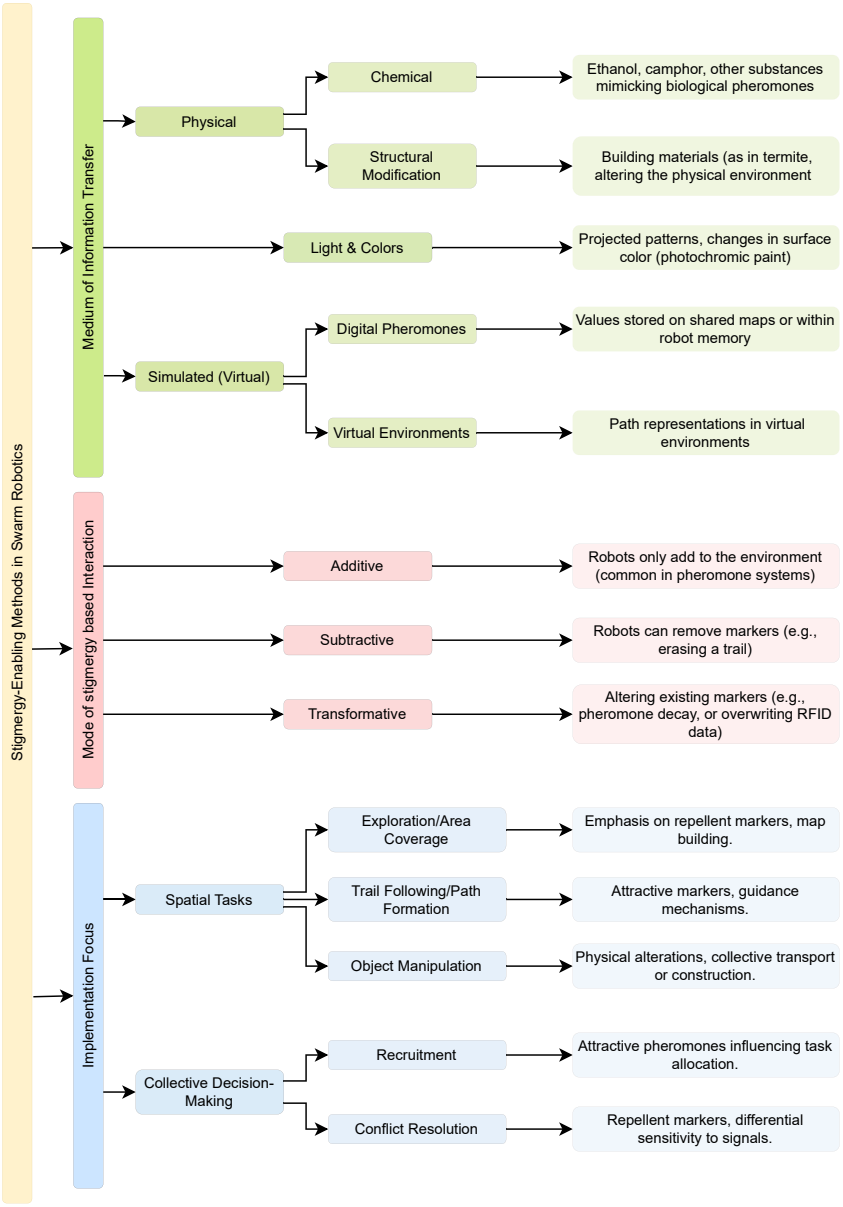


Figure 2.2: **Taxonomy of stigmergy-enabling methods in swarm robotics**

Table 2.1: **Advantages and disadvantages of different stigmergy enabling methods.**

Method	Advantages	Disadvantage	Additional Consideration
Chemical Substance	Close to natural systems, possible memory precursor	Specialized sensors, safety, environmental factors	May not scale well
RFID Systems	Data manipulation, real-world use cases, cue usage	Needs pre-installed tags	
Virtualization	Flexible, supports multiple pheromones, scalable in simulation	Potential delays, mostly simulated so far	Ideal for algorithm testing
Light/color-Based	Flexibility, potential high-resolution, attract/repel	Real-world lighting issues, may not scale well in dense swarms	Good for behavior exploration

alcohol within their surroundings, mimicking the pheromones found in nature by using volatile substances like ethanol.

While this approach offers rapid communication, it presents challenges in terms of spatial precision and control. One experiment involved multiple robots with gas sensors designed to detect alcohol-based chemicals. These robots could release small amounts of chemicals to communicate and detect the presence of their peers. Similarly, researchers have equipped robots with chemical detecting sensors to enable them to follow trails of alcohol-based pheromones (Fujisawa et al. 2014; Purnamadjaja and Russell 2010; Sharpe and Webb 1998). However, airflow in an environment can be unpredictable, making the volatile chemical trails unreliable to follow. Additionally, long-term exposure to alcohol in enclosed spaces risks saturation, which obscures the trail and creates a potential fire hazard.

Researchers have also explored other strategies for artificial pheromone trails. For instance, robots carrying a heating element, or hot paraffin, can leave a heat trail on the ground (Russell 1997). Again, the practicality of the system is questionable. The generation of sufficient heat using the limited battery power available in many mobile robots is not

a power-efficient solution and the idea of robots laying consistent heat trails for a long period is impractical.

Although these systems illustrate a realistic implementation of stigmergy, it remains that the use of volatile and inflammable chemicals in the environment and expensive chemical detection sensors raises questions on the practicality of the systems.

– **Structural modification:**

Unlike termites, whose nest-building offers robustness at the expense of flexibility, robots have the potential to alter the physical landscape in more adaptable ways. However, achieving both robustness and flexibility remains a challenge in robotics.

To illustrate this potential, researchers have studied the clustering and sorting of colored frisbees by a robot swarm (Holland and Melhuish 1999). Using a stigmergy-based coordination mechanism, the robots successfully sorted and clustered different objects, demonstrating the power of self-organization through stigmergy (Werfel et al. 2014).

Additionally, a multi-robot system for construction tasks has been presented (Allwright et al. 2017, 2019). This system leverages a decentralized control strategy inspired by social insects, enabling autonomous robots to build 3-D structures. Key to this system are "stigmergic blocks" whose markings and structural arrangements guide the robot's actions, allowing coordinated construction without centralized direction. The researchers demonstrate the system's capabilities by successfully constructing a staircase, marking a significant step toward swarm robotics construction. Future work aims to extend this strategy for diverse structures and multi-robot collaboration.

- **Light & colors:** Projected patterns or changes in surface luminescence create visual markers that robots can perceive. It offers flexibility and scalability.

  A number of researchers have proposed different approaches in which an external infrastructure is used to partially or completely store the stigmergic information. For instance, the movement of robots is tracked using an overhead camera and the pheromone trails are then either projected on the ground (Garnier et al. 2013; Hamann et al. 2007; Hunt et al. 2019; Talamali et al. 2020) or displayed on LCD screens that serve as the floor on which the robots move (Arvin et al. 2015; Na et al. 2020, 2019). Although the

pheromone information is stored digitally in a centralized infrastructure, it is conveyed to robots using colors displayed on LCDs or projected onto the ground. Robots can detect the projected pheromone trails locally using their sensors and act accordingly. These approaches offer flexibility in displaying multiple pheromone types simultaneously and can even emulate the volatility of pheromones. However, even in these implementations, robots cannot perform stigmergy autonomously and rely on an external infrastructure, such as the tracking system and pheromone projectors. Additionally, these approaches are primarily suitable for experiments with small robot swarms and may not be feasible for large-scale implementations.

In a notable research project, robots operated on a surface covered with phosphorescent material (glow-in-the-dark paint) that illuminates when exposed to UV light. Mayet et al. 2010 developed a hardware add-on for e-puck robots, enabling them to project UV light and create glowing trails, mimicking pheromone deposits. Additionally, they equipped the robots with cameras to detect these glow trails in the environment. However, this system has been primarily evaluated in simulations, with limited testing on a single physical robot to demonstrate feasibility.

- **Simulated pheromones**
  - **Digital pheromones:** Payton et al. 2001 coined the term virtual pheromo- nes in their well-known study titled pheromone robotics. In this study, the virtual pheromone is implemented via messages locally exchanged by robots using short-range infrared communication. Robots locally broadcast messages that inform neighboring peers about the occurrence of an event or a discovery. The neighboring robots act according to the information received, or store the message based on its content, strength, or direction. These messages ultimately act as virtual agents that leave virtual pheromone on the robots (Campo et al. 2010). Many researchers have used this approach to develop communication networks, robot chain formation, and to perform foraging tasks (Ludwig and Gini 2006; Maes et al. 1996; Payton et al. 2005). Yet, this approach uses pheromone as an analogy: pheromone is never stored in the environment and a line-of-sight between robots is necessary to pass the information.

    In an alternative approach, physical memory devices are placed in an

environment that acts as a shared memory for a robot swarm. For instance, a grid of radio-frequency-identification (RFID) tags is embedded into the floor on which the robots operate. Robots read and write the pheromone information on these tags using special equipment at close proximity (Alfeo et al. 2019; Khaliq et al. 2014; Khaliq and Saffiotti 2015). However, the robots can read only one RFID tag at a time and this makes it difficult to conceive strategies based on pheromone gradients.

- **Virtual environments:** Another approach is the one in which researchers implement virtual environments to store the stigmergic information of a robot swarm. Robots can access and modify the virtual environment through virtual actuators and sensors that enable the release and detection of virtual pheromone. For example, Augmented Reality for Kilobots (ARK) is a virtual environment in which an overhead controller based on IR communication tracks the individual robots of a swarm of Kilobots, stores the pheromone information on a computer, and controls the robots accordingly (Reina et al. 2017). Similarly, Antoun et al. 2016 conceived a device named Kilogrid that enables the creation of virtual environments for swarms of Kilobots. The Kilogrid uses a floor made of a grid of communication modules that detect the position of Kilobots above them, store the information of pheromone in a virtual environment hosted in a remote computer, and controls the robots accordingly. Although, ARK and the Kilogrid are capable of executing large-scale collective mission using Kilobots, these systems are difficult to scale for a swarm of larger and faster robots.

Mode of interaction in stigmergy

- **Additive:** This is the most common form of environment modification for establishing stigmergy-based coordination. Information is added but not directly removed. Examples: artificial pheromone lay by individuals of a robot swarm to reinforce a pheromone trail (Mayet et al. 2010).
- **Subtractive:** Removal of existing pheromone information offers precise environmental control but is less common in natural systems. Examples: A robot clearing a pheromone trail marker by erasing pheromone information on an RFID tag (Alfeo et al. 2019; Khaliq et al. 2014.

- **Transformative:** Modifying existing pheromone information rather than outright removal gives dynamic flexibility. Examples: Updating pheromone information on RFID tags, LCD screen, or in virtual environments (Na et al. 2021, 2020).

Implementation Focus

When designing control software for robot swarms using pheromone-based stigmergy, the most important part is deciding how robots should react to the pheromones. The control software design process plays a crucial role in defining the meaning of pheromone traces. It must determine how robots should react to them: should robots avoid the trace, follow it, deposit more pheromone, or perhaps even perform a completely different action like stopping or moving towards light? This decision shapes how the swarm interacts and the types of behaviors that can emerge. Stigmergy itself is a complex concept, and designing how a whole robot swarm operates to implement it is even more challenging.

Traditionally, designing control software for robot swarms using stigmergy has been done manually. This approach made it difficult to consistently replicate successful results and was highly dependent on the designer's expertise. To overcome these limitations, we need an automatic design method capable of creating control software that allows swarms to leverage stigmergy effectively. Such a method would need to address several key aspects of pheromone-based behaviors, which I discuss next:

- **Spatial tasks:**
  - **Exploration/area-coverage:** *Repellent pheromone:* robots might signal "already explored" areas. This prevents redundant effort and promotes efficient coverage. *Collaborative Map Construction:* Robots create shared maps by leaving markers that may decay over time. Newer marks help others prioritize unexplored regions.
  - **Trail following/path formation:** *Attractive markers:* pheromone trails guiding their peers to a source, nest. *Gradient-based Strategies:* the intensity of signals, either pheromones or projected light, can create gradients. Robots follow these gradients to optimize paths or locate the source.
  - **Object manipulation:** *Physical modifications:* Robots might alter the terrain itself to aid movement (e.g., flattening a path) or carry a

building-block for collaborative construction. *"Signposts":* simple markers, like RFID tags or light patterns, placed near objects convey status ("available", "under transport") supporting collaborative manipulation.

- **Collective decision-making:**
  - **Recruitment:** pheromone trails to food sources are the classic example. The stronger the trail, the more recruits follow. This translates to task allocation and prioritization in robot swarms.
  - **Conflict resolution:** *Repellent markers:* these might indicate resource scarcity in an area, alarming others, and preventing over-exploitation. *Differential sensitivity:* Robots may have varied responses to markers depending on their current task or internal state. This helps modulate how readily a robot abandons one task to join another based on various pheromone types and strengths.

2.5 Discussion

This chapter introduced swarm robotics, its applications, and the unique characteristics of this developing field. We explored the ongoing challenge of designing robot swarms and how there is no universal consensus among researchers and engineers regarding the "best" design approach. However, designers can select a reasonably suitable design method based on specific factors like design requirements, resources, and the swarm's intended purpose. We examined various design approaches, noting the niche applications of semi-automatic design and the advantages of automatic design in specific cases. We also discussed how online design methods are well-suited to certain scenarios, while offline methods hold value in particular situations.

We learned that evolutionary design offers benefits but faces the significant challenge of the reality gap caused by overfitting in simulations. While reinforcement learning presents another viable solution, it too has drawbacks. The automatic modular design approach has the potential to produce robust swarm control software, with recent studies demonstrating promising results. In many cases, modular designs outperform other methods, especially when deployed in real-world environments. Currently, only manual design approaches exist for creating robot swarms capable of real-world pheromone-based stigmergy. Other attempts often remain restricted to simulations and/or are tailored for a single, highly specific mission.

In the context of this thesis, I believe the automatic modular design approach is the most suitable for designing fully automatic stigmergy behaviors. In this work, I focus on automatically designing stigmergy-based collective behaviors for robot swarms. I present a modular, automatic, offline design method belonging to the AutoMoDe family (Birattari et al. 2021). As is typical in automatic offline design (Birattari et al. 2019; Francesca and Birattari 2016), the design problem is reframed as an optimization problem solved in simulation before physical deployment (Birattari et al. 2019, 2020). The optimization algorithm explores a solution space comprising control software instances. These instances are created by selecting and combining pre-existing software modules (low-level behaviors and their transition conditions) into a modular architecture (like finite-state machines or behavior trees). The algorithm also fine-tunes the free parameters of these modules (Francesca et al. 2014b). For this work, I opt for the finite-state machine architecture (see Chapter 4 for details).

In the second part of this chapter, I presented a taxonomy of stigmergy-emulating technologies from the available literature. This taxonomy provides a framework to better understand the diverse ways robots can utilize stigmergy. Rather than being a set of strict rules, it serves as a tool to help researchers and designers think critically about existing systems. It facilitates comparison of different methods and highlights areas where advancements could be made.

Stigmergy offers great potential for robot swarms, but creating real-world environments where robots can utilize pheromone-based stigmergy remains an engineering challenge. Virtual stigmergy is a valuable tool for simulations but lacks the dynamic, environment-altering aspect of true stigmergy. Other methods, such as those relying on alcohol trails, colored markings, or external tracking systems, can be expensive, complex, and potentially compromise the robots' independence.

This research introduces `Phormica`, a promising solution that utilizes special surfaces and UV LEDs to create affordable and practical artificial pheromones. This system opens the door for researchers to explore a wider range of stigmergy-based swarm behaviors. Importantly, `Phormica`'s emphasis on ease of use and robot independence addresses some of the limitations faced by older artificial pheromone systems.

In Chapter 3, I present `Phormica`, the photochromic pheromone release and detection system, in detail. I combine `Phormica` with manual modular design to create robot swarms capable of exhibiting stigmergy-based behaviors. Through various missions, I demonstrate both the practicality of `Phormica` and the potential of modular design for successfully creating stigmergy-driven robot swarms. In

Chapter 4, I build upon this foundation, integrating `Phormica` with AutoMoDe for the automatic design of pheromone-based stigmergy in robot swarms.

Chapter 3

Phormica

In this chapter, I introduce Phormica, a novel photochromic pheromone release and detection system designed for swarm robotics research. This system provides a practical and flexible way to emulate pheromone-based stigmergy in real-world environments, enabling the study of complex swarm behaviors. I begin by detailing the construction of the artificial environment, the robotic platform employed in the experiments, and the specialized add-on that allows robots to deposit and detect artificial pheromones. Following this, I describe the experimental setup and methodology. Finally, I present the results of these experiments, demonstrating the potential of Phormica for advancing swarm robotics research.

3.1 Photochromic pheromone release and detection system (Phormica)

In this section, I discuss the construction of the artificial pheromone environment; the robotic platform; and the functionalities of Phormica.

3.1.1 Artificial pheromone environment

In Phormica, the floor of the environment is coated with a photochromic substance that is used to store stigmergic information. Photochromic substances are a class of materials that change color when a light of specific wavelength and intensity hits them. When the light source is removed, they gradually switch back to their original color (Dürr and Bouas-Laurent 2003). In this thesis, I use a commonly available photochromic substance that is of white color in its normal state and switches to magenta when lit with UV light. The color of the substance gradually decays to

white after UV light is removed: this phenomenon is similar to the evaporation of biological pheromone. The saturation and decay time (pheromone evaporation) of the magenta color depends on the intensity of the incident UV light–see Figure 3.1. Further research is required to build the mathematical models for the color decay of these substances in the context of stigmergy for swarm robotics.

Besides magenta, this photochromic substance is also available in other colors. Substances with different colors have a different decay time and color saturation—see Figure 3.1. Indeed, depending on the pheromone color and evaporation time requirements, a `Phormica` environment can be constructed with an appropriate color of the substance.

The photochromism of this substance works fine in most indoor lighting conditions. In the case of outdoor environments, the UV light makes up a small portion of sunlight, but it is still enough to affect the photochromism in the material. The substance does not have an infinite shelf-life. Long exposures of sunlight and high temperature can alter the chemical composition of the substance. In ten months of experiments, I never noticed any inconsistency in periods of color change cycles, but further research is required to determine the actual useful lifespan of the substance.

The photochromic substance is mostly available in pigment form. A binder is thus required to apply this substance to the floor. Depending on the application, the photochromic substance can be used with acrylics, PVC, and other resin-based binders. The type of binder and the material of the floor determine the durability of the artificial pheromone environment.

In `Phormica`, I use an acrylic binder with a 15% (w/w) concentration of photochromic substance Salman et al. 2020a. The acrylic binder has many advantages: it is low cost; easy to use; and it can be removed easily if a change of the pigment color in the same arena is needed. The technical data sheet and supplier information of photochromic substance and acrylic binder are available as online supplementary material (Salman et al. 2020a).

The prototyping cost per square meter of the artificial pheromone environment (including 18 mm thick MDF) is approximately €25. Indeed, the most significant advantage of `Phormica` is the scalability of the environment: extending the size of the artificial pheromone environment only requires inexpensive materials and minimal effort.

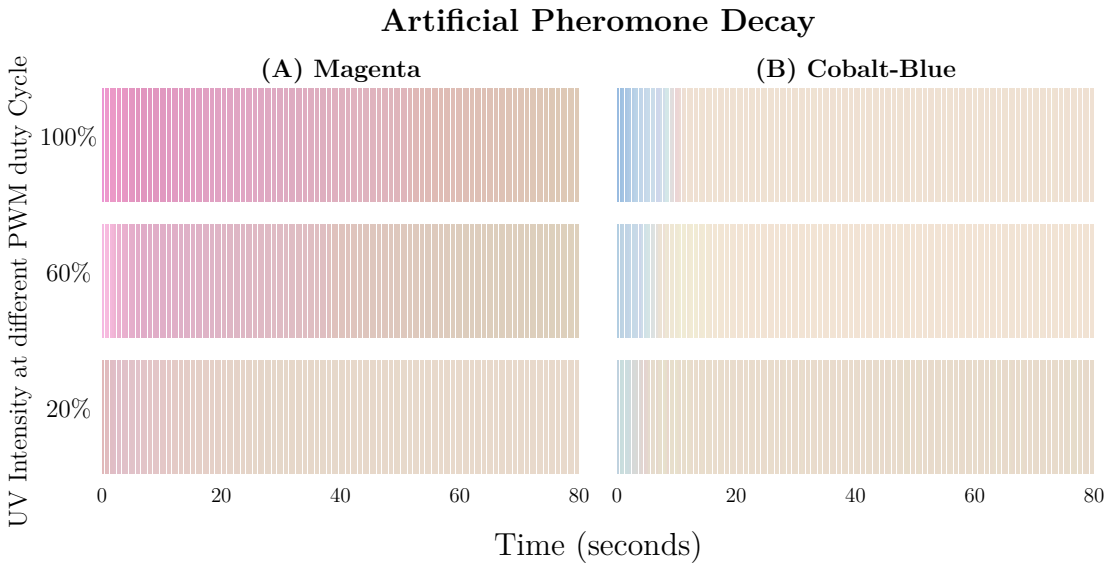


Figure 3.1: Time taken by the photochromic substances (A) magenta and (B) cobalt-blue to switch back to white color after a UV light source is removed. The substances are tested under three different intensities of UV light. The intensity of UV light is controlled by varying the pulse-width-modulation (PWM) duty cycle of UV LEDs.

3.1.2 Robot platform

In this research, I use a swarm of e-puck robots to demonstrate the stigmergic coordination capabilities of `Phormica`. The e-puck robot is a differential-drive robot that is mostly used in swarm robotics research (Mondada et al. 2009). I use the extended version of the e-puck, equipped with the various computer-on-module, sensors, and communication modules: the Overo Gumstix, to run Linux on the robot; the ground sensor module to detect the gray-level color of the floor; the omni-directional camera to perceive its surroundings; and the range-and-bearing module for communication with neighboring peers (Gutiérrez et al. 2009)—see Figure 3.2. I formally describe the characteristics of the e-puck robot with a reference model that I shall call RM4.0 as it is a modified version of a number of previously defined reference models of the e-puck robot (Hasselmann et al. 2018a)—see Table 3.1. The reference model provides the details of the input and output variables of onboard sensors and actuators. The control software of the robot can read and/or update these variables at every control step of 100 ms. The robot can detect obstacles ($prox_i$) in its surroundings using eight infrared transceivers. It can also detect the gray-level color of the floor ($ground_j$) using ground sensors. The robots use the range-and-bearing module to detect the neighboring peers (n) within their

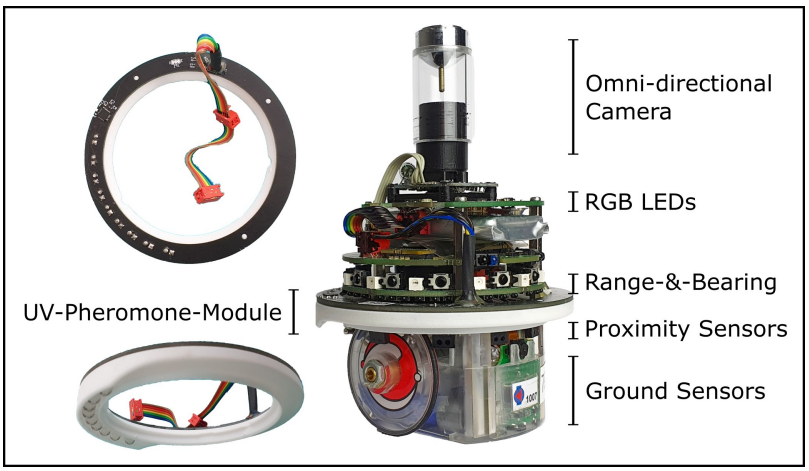


Figure 3.2: Extended version of e-puck equipped with UV-pheromone-module and omni-directional camera.

perception range of 0.5 m, and their aggregated relative position (V). Three RGB LEDs, present on the robot, can display cyan or yellow color. Finally, the robot can move using two wheels whose velocity can be controlled (v_r, v_l).

3.1.3 Pheromone release and detection

In `Phormica`, pheromone release and detection is implemented using two modules: (i) the UV-pheromone-module is used to deposit the pheromone in the environment; and (ii) the omni-directional camera is used to detect pheromone in the environment.

UV-pheromone-module

In order to mimic the release of pheromone, I have developed a UV-pheromone-module to allow e-puck robots to project variable-intensity and variable-width UV light on the floor of the environment. The UV-pheromone-module is ring-shaped and replaces the original e-puck ring used to diffuse the light of the eight LEDs around it—see Figure 3.2. The UV-pheromone-module consists of two parts: (i) a printed circuit board (PCB); and (ii) a 3D-printed ring that acts as a PCB holder, gives the mechanical strength, and helps to project light uniformly on the floor—see Figure 3.3. The UV-pheromone-module is designed in such a way that UV light is projected only downward. A user is only exposed to the UV light if it lifts a functioning robot and turns the emitting UV LEDs towards themselves. The estimated prototyping cost of one UV-pheromone-module is €36: this price

Table 3.1: Reference Model RM4.0. The highlighted part in gray characterizes the pheromone release capabilities of the e-puck. It also represents the novelty with respect to RM3.0 (Hasselmann et al. 2018a)

Sensor	**Input**	**Value**	**Description**
Proximity	$prox_{i\in\{1,...,8\}}$	$[0, 1]$	reading of proximity sensor i
Ground	$ground_{j\in\{1,2,3\}}$	$\{black, gray, white\}$	reading of ground sensor j
Range-&-Bearing	n	$\{0, ..., 5\}$	number of neighboring robots perceived
	V	$([0.5, 5]\,; [0, 2]\,\pi\text{rad})$	their aggregate position
Camera	$cam_{c\in\{R,G,B,C,M,Y\}}$	$\{yes, no\}$	colors perceived
	$V_{c\in\{R,G,B,C,M,Y\}}$	$(1.0; [0, 2]\pi\text{rad})$	their relative aggregate direction

Actuator	**Output**	**Value**	**Description**
Motors	$v_{k\in\{l,r\}}$	$[-0.12, 0.12]$ ms^{-1}	target linear wheel velocity
RGB LEDs	*LEDs*	$\{\phi, C, Y\}$	color displayed by the LEDs
UV-pheromone-module	*phe*	$(\{1, ..., 9\}, [0, 255])$	number and brightness of UV LEDs

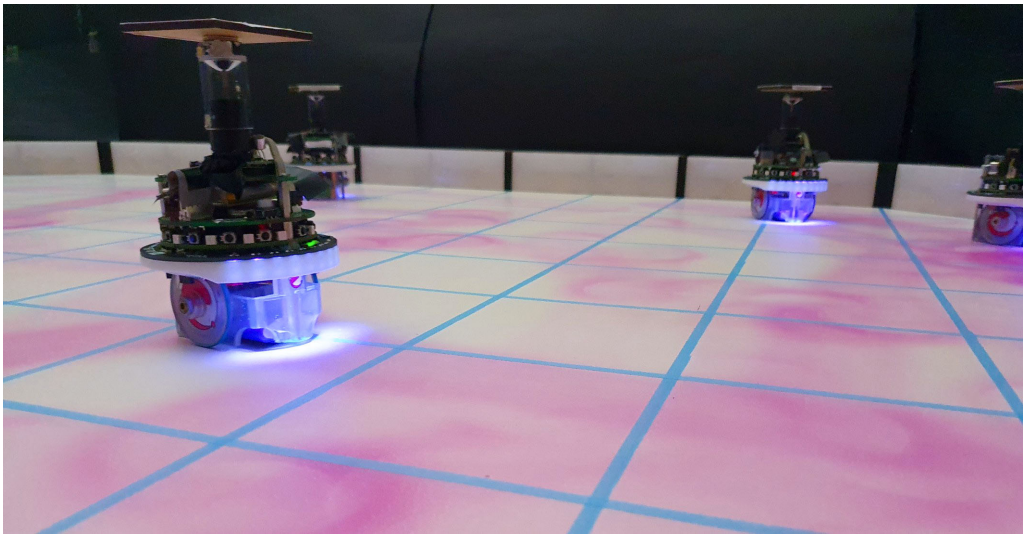


Figure 3.3: A swarm of e-puck robots equipped with UV-pheromone-module and omni-directional camera releases pheromone in the *artificial pheromone environment* to accomplish a collective mission.

includes a 3D printed ring, a PCB, and all electronics components.

The UV-pheromone-module is equipped with nine UV LEDs facing downward and positioned at the rear of the e-puck. A nine-channel LED driver, LP55231, is used to control these LEDs. The LP55231 directly communicates with the main computer of the e-puck through I2C without using any intermediate peripheral. The LEDs are controllable separately or collectively (phe), to vary the intensity and beam-width of UV light—see Table 3.1. The intensity of the UV LEDs is controlled by a PWM (pulse-width-modulation) signal driving to LEDs. This allows the robot to control the intensity and decay time of the pheromone—see Figure 3.1. The UV-pheromone-module can leave a trail of pheromone of width between 30 mm, when only one LED is on, and a maximum of 75 mm, when all nine LEDs are on.

Omni-directional camera

In order to detect the artificial pheromone in the environment, the robots use an omni-directional camera. This camera can perceive red, blue, green, cyan, magenta, and yellow colors (cam_c) in a 360 degree field of view: magenta color is reserved for the artificial pheromone—Table 3.1. Moreover, the field of view of the camera can be controlled to enable the robot to perceive pheromone in a given direction only: in some collective missions, it is appropriate that robots are able to differentiate pheromone released by themselves and pheromone that was already present in the environment. After perceiving the pheromone in the arena, the robot can use this information to move towards that area using a unit vector (V_c) that represents the attraction to the pheromone perceived.

3.2 Experimental setup

In this section, I provide the details of the experiments in the design of collective behaviors for robot swarms that are capable of stigmergic coordination. For the purpose of this thesis, I use a swarm of five e-puck robots to perform experiments on three different collective missions: (1) Coverage, (2) Foraging, and (3) Tasking. For each collective mission, I perform two experiments: (i) the robot swarm performs the collective mission without using artificial pheromone; and (ii) the robot swarm takes advantage of `Phormica` and uses artificial pheromone. In the three missions, the robots operate in a rectangular arena in which modular RGB blocks are placed as walls. Each RGB block is 0.25 m long and can display a color according to the mission requirements. This modular arena system was initially used by Garzón

Ramos and Birattari 2020.

3.2.1 Robot control software

I manually designed the control software of the robots—both, for the experiments in which they release pheromone and for the ones in which they do not. The control software has the form of a probabilistic finite–state machine. In this architecture, states represent low-level behaviors that the robots execute, and transition conditions represent events that trigger the change from one behavior to another. Each low-level behavior and transition condition is a parametric software module. I conceived four low-level behaviors and four transitions. The control software of the robots is obtained by configuring and assembling these software modules into finite–state machines. Table 3.2 describes the low-level behaviors and transition conditions I conceived for experimenting with `Phormica`.

The four low-level behaviors are exploration, go-to-color, avoid-color, and waggle; and the four transition conditions are black-floor, gray-floor, color-detected, and fixed-probability. In exploration, the robot moves randomly describing a ballistic motion behavior (Kegeleirs et al. 2019). Go-to-color and avoid-color are behaviors in which the robot moves towards or away from objects that display a specific color (c)—if the specific color is not perceived, the robot performs exploration. In waggle, the robot rotates in place for a random number of control cycles ($\tau \in \{1, .., 100\}$). All low-level behaviors embed obstacle avoidance, and the selective release of pheromone (phe). The low-level behaviors and transition conditions that use the omni-directional camera to perceive color include a parameter that enables control of the field of view of the omni-directional camera (fov). Black-floor and gray-floor trigger a transition with a certain probability (β) if the robot detects black or gray floor, respectively. Color-detected can trigger if the robot detects a specific color (c). Fixed-probability triggers with a probability β.

Different combinations of these software modules into finite–state machines and different values of the parameters lead the robots to exhibit different collective behaviors. For each mission, I designed finite–state machines that enable the robots to perform the missions at hand.

3.2.2 Coverage

The arena is divided into small cells to make a mesh. This mesh consists of 128 quadrilateral cells and 8 triangular cells. Q_c represents the total number of cells in

Table 3.2: Low-level behaviors and transition conditions used in Phormica. All low-level behaviors are capable of releasing pheromone *phe*—see Table 3.1. The parameter $fov \in [0, 2\pi]$ determines the field of view of the camera. The parameter $\tau \in \{1, .., 100\}$ determines the number of control cycles for which a robot rotates in place: the period of the control cycle is 100 ms. In all transition conditions, the parameter $\beta \in [0, 1]$ determines the probability of transitioning. In Phormica, the values of phe, β, and fov are determined manually by trial and error.

Low-level behaviors	**Parameters**	**Description**
Exploration	phe	Robot moves by random walk.
Go-to-color	c^*, phe, fov	Robot steadily moves toward objects displaying a specific color.
Avoid-color	c^*, phe, fov	Robot steadily moves away from objects displaying a specific color.
Waggle	τ, phe	Robot rotates in place for a random period of time.

Transition conditions	**Parameters**	**Description**
Black-floor	β	Black floor detected
Gray-floor	β	Gray floor detected
Color-detected	β, c^*, fov	Objects of color perceived.
Fixed-probability	β	Transition with a fixed probability

$^*c \in \{R, G, B, C, M, Y\}$

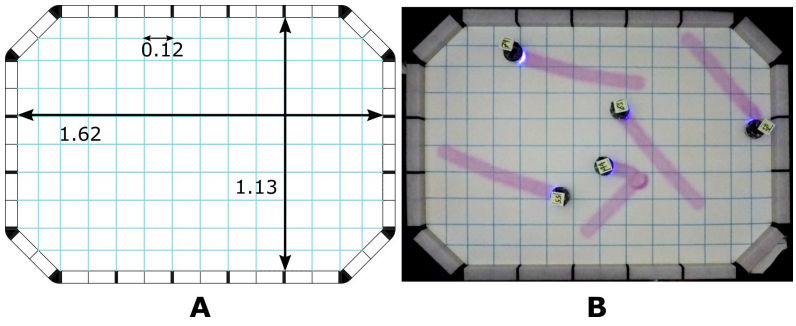


Figure 3.4: Construction of arena for the COVERAGE experiments: (A) technical representation of the arena with dimensions of mesh, and (B) real arena. Measurements are expressed in meters.

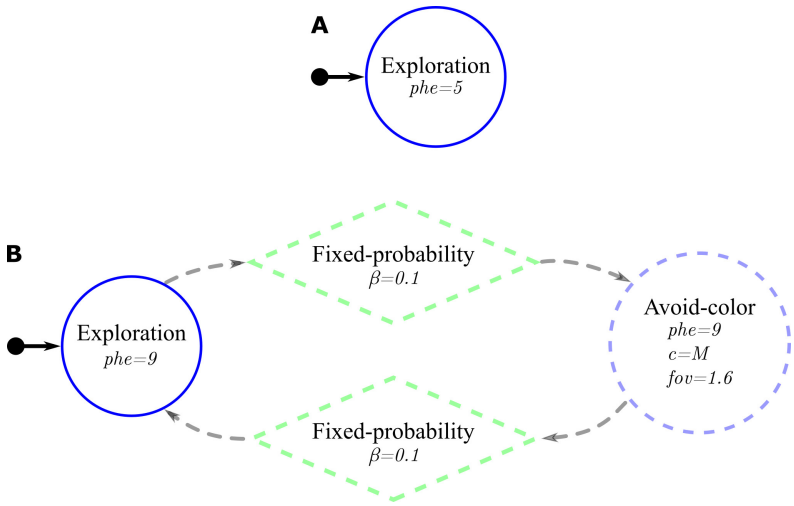


Figure 3.5: Probabilistic finite–state machines for the two COVERAGE experiments: (A) Basic-COVERAGE, and (B) Phormica-COVERAGE—the difference between the two probabilistic finite–state machines is highlighted with dashed-lines. In both finite–state machines, a circle, a lozenge, and an arrow with a small circle represent the states, transition, and initial state, respectively.

the mesh—see Figure 3.4. A robot swarm must visit each quadrilateral cell at least once in one minute of the experiment duration. The performance C_q of the swarm is measured by the following objective function:

$$C_q = \frac{N_c\left(T_{cov}\right)}{Q_c} \times 100, \tag{3.1}$$

where T_{cov} is the total duration of the experiment; $N_c\left(T_{cov}\right)$ is the number of distinctive quadrilateral cells visited by the robots up to time $T_{cov} = 60\,\text{s}$, which is the duration of a run. I perform two experiments: (i) Basic-COVERAGE, and (ii) Phormica-COVERAGE. For each experiment, I perform 20 runs of a control software in the form of a finite–state machine—see Figure 3.5. At the beginning of the experiments, the robots are manually placed in the arena. Their initial position and orientation is selected at random by a human experimenter.

Basic-Coverage

In this experiment, the robots do not employ stigmergic coordination. The robots move randomly in an attempt to traverse the whole area–see Figure 3.5A.

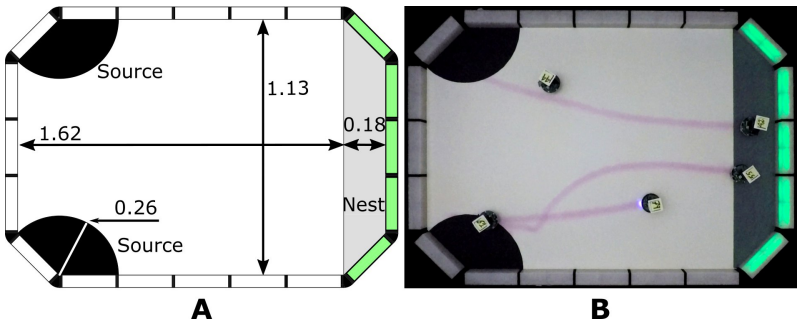


Figure 3.6: Construction of arena for the FORAGING experiments: (A) technical representation of the arena with dimensions and positions of different zones, and (B) real arena. Measurements are expressed in meters.

Phormica-Coverage

In this experiment, the robot swarm uses Phormica to achieve stigmergic coordination. The robots start a random exploration of the arena while releasing the pheromone. With a fixed probability, the robots begin to avoid previously explored areas: the robots detect the pheromone in the arena and go towards the unexplored part (without pheromone) of the arena. Similarly, with the same fixed probability, the robots again switch their behavior to random exploration—see Figure 3.5B.

3.2.3 Foraging

In FORAGING, the robot swarm must collect a maximum number of objects from two sources and drop them in the nest. I abstract the FORAGING experiment by considering that an object is retrieved when an individual robot visits a source, and the object is dropped when the same robot visits the nest. The two sources in the arena are represented as two black zones, while the nest is represented as a gray zone—see Figure 3.6. A green light is also placed behind the nest as a cue for the robots: the robots use the omni-directional camera to detect the green light (nest). The dimensions and positions of the two source zones and nest are given in Figure 3.6A. I perform two experiments: (i) Basic-FORAGING, and (ii) Phormica-FORAGING. For each experiment, I perform 20 runs of a control software in the form of a finite–state machine—see Figure 3.7. The performance (F_F) of the robot swarm is computed by the following objective function:

$$F_F = N_o(T_{for}), \tag{3.2}$$

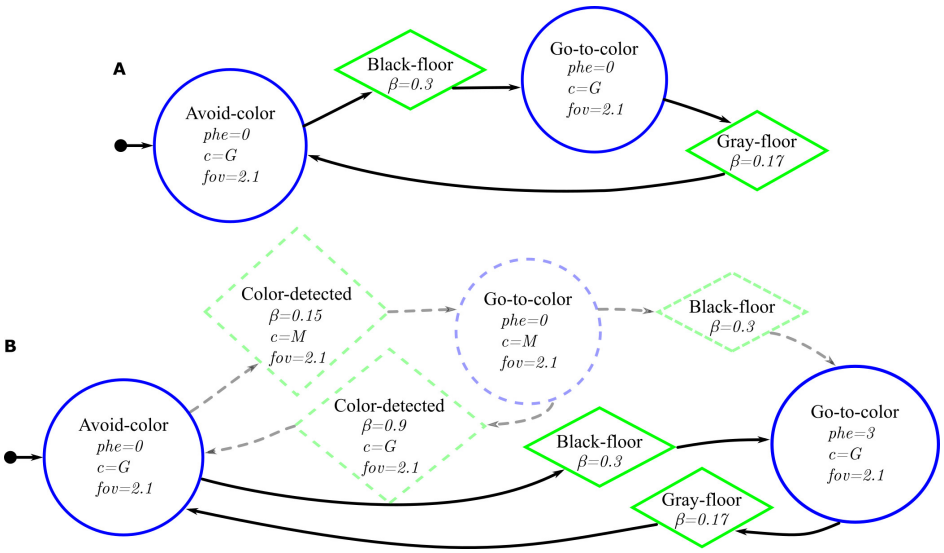


Figure 3.7: Probabilistic finite–state machines for the two FORAGING experiments: (A) Basic-FORAGING, and (B) `Phormica`-FORAGING—the difference between the two probabilistic finite–state machines is highlighted with dashed-lines. In both finite–state machines, a circle, a lozenge, and an arrow with a small circle represent the states, transition, and initial state, respectively.

where T_{for} is the total duration of the experiment, that is, 180 s; and $N_o(T_{for})$ is the number of objects retrieved by the swarm up to time T_{for}. At the beginning of the experiments, the robots are manually placed in the arena. Their initial position and orientation is selected at random by a human experimenter.

Basic-Foraging

In this experiment, the robot swarm does not use stigmergic coordination. The robots start searching for the source (black zone). When a robot reaches the source, it moves toward the nest (green light). After reaching the nest (gray zone), it again starts searching for the source. The finite–state machine that I use in this experiment is shown in Figure 3.7A.

`Phormica`-Foraging

In this experiment, the robot swarm uses `Phormica` to achieve stigmergic coordination and perform foraging. The robots start searching for a source (black zone). Once a robot has reached one, it moves toward the nest (green light) while releasing pheromone. When the robot reaches the nest (gray zone), it stops releasing pheromone and follow the pheromone trail to go back to the source. The

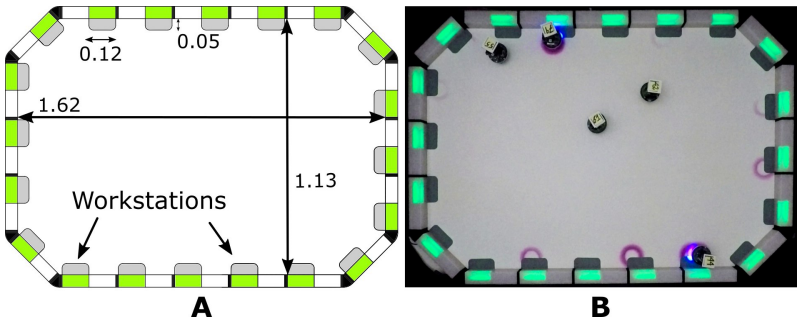


Figure 3.8: Construction of arena for the TASKING experiments: (A) technical representation of the arena with dimensions and position of workstations, and (B) real arena. Measurements are expressed in meters.

finite–state machine that I use in this experiment is shown in Figure 3.7B.

3.2.4 Tasking

Tasking is a mission in which a robot swarm must perform a number of abstract tasks in a manufacturing facility. Robots perform a task in workstations that are indicated by gray patches on the floor and LED blocks displaying the color green. The manufacturing facility has 20 workstations—see Figure 3.8. The dimensions and positions of the workstations are given in Figure 3.8A. I consider that a task is performed when a robot steps into the corresponding workstation and spins on its axis (waggle). The performance of the robot swarm is measured by the number of tasks performed on distinct workstations in two minutes. The robots must visit a workstation only once. Robots receive a penalty if they perform a task on a workstation more than once. The performance T_p of the robot swarm is computed by the following objective function:

$$T_P = T_d(T_{tas}) - T_{rep}(T_{tas}), \tag{3.3}$$

where T_{tas} is the total duration of the experiment, that is, 120 s; $T_d(T_{tas})$ is the number of tasks performed on distinct workstations; and $T_{rep}(T_{tas})$ is the number of tasks performed on a workstation more than once up to time T_{tas}. I perform two experiments: (i) Basic-TASKING, and (ii) `Phormica`-TASKING. For each experiment, I perform 20 runs of a control software in the form of a finite–state machine—see Figure 3.9. At the beginning of the experiments, the robots are manually placed in the arena. Their initial position and orientation is selected at random by a human experimenter.

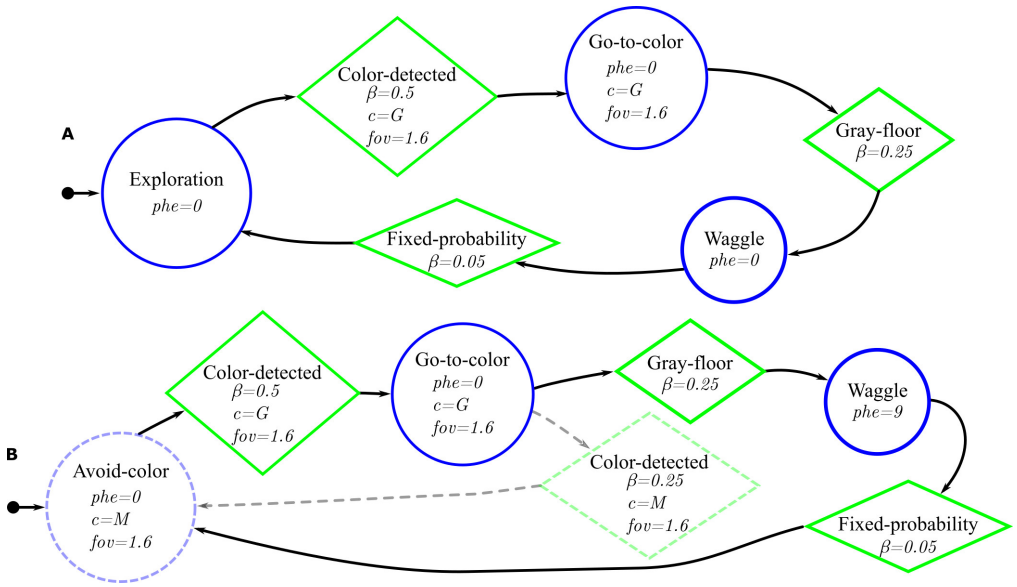


Figure 3.9: Probabilistic finite–state machines for the two TASKING experiments: (A) Basic-TASKING, and (B) `Phormica`-TASKING—the difference between the two probabilistic finite–state machines is highlighted with dashed-lines. In both finite–state machines, a circle, a lozenge, and an arrow with a small circle represent the states, transition, and initial state, respectively.

Basic-Tasking

The robots start searching for a workstation. Once a robot arrives at a workstation, it waggles at the spot. As the robots do not use the UV-pheromone-module, they are unable to mark the task and hence cannot identify the workstations that have been already served. The finite–state machine that I use in this experiment is shown in Figure 3.9A.

`Phormica`-Tasking

In this experiment, the robot swarm uses `Phormica` to release and detect artificial pheromone at the workstations as an indication of a completed task. The robots look for workstations to perform available tasks. If a robot perceives a pheromone indication, it avoids that workstation and looks for another one. Once a robot arrives at a workstation that has no pheromone, it waggles at the spot and releases pheromone to indicate that a task was already performed there. The finite–state machine that I use in this experiment is shown in Figure 3.9B.

3.3 Results

In this section, I present the results on a per-mission basis. The performance scores of experiments are evaluated by manually post-processing the experiment videos.[1] I use box-and-whiskers plots to compare the performance observed when Phormica is used and when it is not—see section Experimental Setup. In these plots, boxes represent the interquartile range, covering the central 50% of the values observed. Whiskers extend from the lower quartile to the lowest recorded performance, and from the upper quartile to the highest one. The horizontal line in the middle of each box plot represents the median performance, and the notches on the box represent a 95% confidence interval on the median. If the notches of two boxes do not overlap, then the difference between their respective medians is significant, with a confidence of at least 95% Chambers et al. 1983. The control software, the data collected, and videos of the experiments are available as online supplementary material (Salman et al. 2020a).

3.3.1 Coverage

In Basic-Coverage, the robots are unable to differentiate between the explored and unexplored regions of the arena. Therefore, many parts of the arena remain unexplored: the robots waste time re-exploring the already visited ones. On the other hand, in Phormica-Coverage, the robots use Phormica to achieve stigmergic coordination that enables them to avoid revisiting the already explored parts of the arena. Consequently, in Phormica-Coverage the swarm explores a significantly wider area than in Basic-Coverage (Wilcoxon signed–rank test, confidence 95%)—see Figure 3.10A.

I observe that the visual assessment of the performance of Basic-Coverage is particularly difficult without an external infrastructure—i.e., a tracking system—see Figure 3.11A. In order to make Basic-Coverage experiments visually comprehensible, the robots turn on their UV LEDs to leave the trails of pheromone while doing random walk: these trails do not stimulate any behavior in robots—see Figure 3.11B. Indeed, the pheromone release feature of Phormica can also be used to visualize the motion of individual robots of a swarm and visualizing the area

[1]The experiments were originally planned to be executed in a well-equipped robotics experiment facility that is also equipped with a tracking system. But due to COVID-19 confinement measures, I was forced to move the experimental setup to my residence where it was unfeasible to set up a tracking system from scratch. Although I evaluated the performance manually, I am confident that my evaluation is accurate and that the results I obtained can be reproduced.

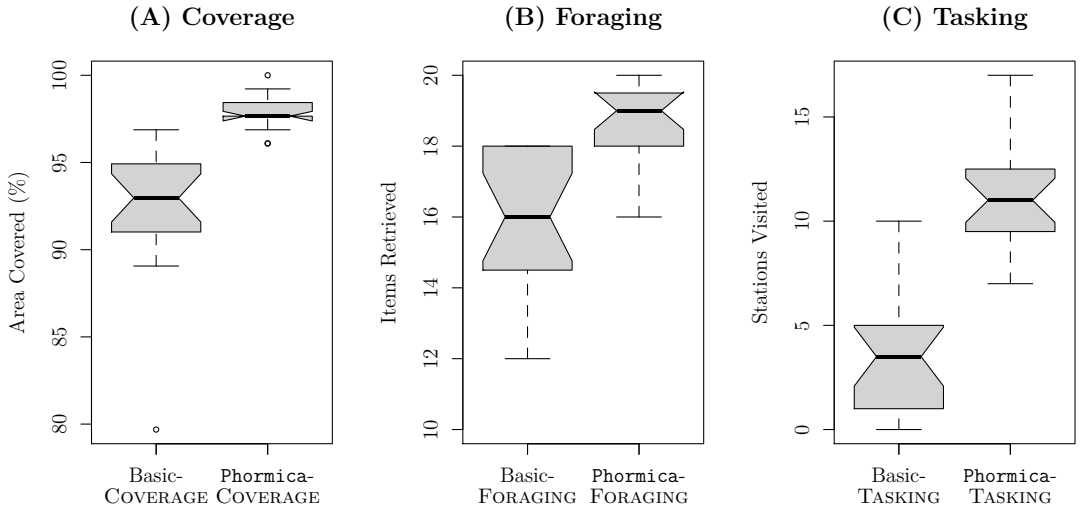


Figure 3.10: Performance obtained in the missions (A) Coverage, (B) Foraging, and (c) Tasking.

covered by a robot swarm—see Figure 3.11.

3.3.2 Foraging

In Basic-Foraging, the robots are not aware of the location of the source in the arena. On the other hand, in Phormica-Foraging, the robots deposit pheromone trails from source to nest that other robots (or same robot) use them to track the location of the source: the robots spend less time searching for the source compared to Basic-Foraging—see Figure 3.12A. The results show that in Phormica-Foraging the robot swarm retrieved significantly more items than in Basic-Foraging (Wilcoxon signed–rank test, confidence 95%)—see Figure 3.10B.

3.3.3 Tasking

In Basic-Tasking, the robots do not use pheromone markers on workstations and are therefore prone to visit a workstation more than once. On the contrary, in Phormica-Tasking, the robots deposit pheromone to differentiate whether a workstation is available or a task has already been performed—see Figure 3.12B. As expected, results show that in Phormica-Tasking the robot swarm obtains a significantly higher score than in Basic-Tasking (Wilcoxon signed–rank test, confidence 95%)—see Figure 3.10C.

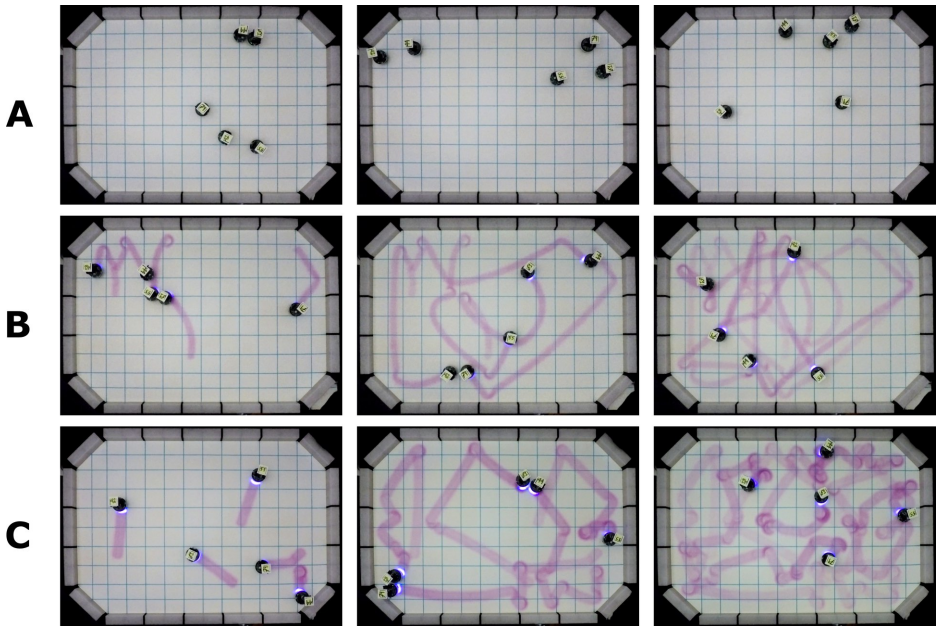


Figure 3.11: Motion patterns evolved during different COVERAGE experiments. (A) e-puck robots performing Basic-COVERAGE: robots tracking is impossible without an external infrastructure. (B) Robots performing Basic-COVERAGE, they use `Phormica` only to release pheromone. (C) Robots performing `Phormica`-COVERAGE: they release pheromone and are also stimulated by its presence.

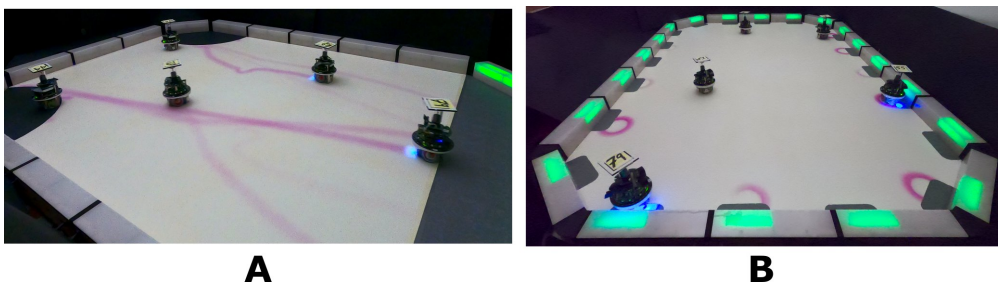


Figure 3.12: (A) a swarm of e-puck robots depositing pheromone trails from source to nest using `Phormica` to perform FORAGING, and (B) a swarm of e-puck robots using `Phormica` to mark the workstations where they perform a task.

3.4 Discussion

In this section, I discuss the current limitations and possible extensions of Phormica; what can be immediately done to enhance its usability; and the prospects of this approach.

In Phormica, the pheromone release and detection system are developed for e-puck robots. Indeed, this system can be implemented for any robot platform that has at least the capability to detect colors in the environment. However, if only the visualization of motion patterns during a collective mission is required, this system can easily be implemented for most of the small robots presently being used in swarm robotics research (Nedjah and Silva Junior 2019). Due to the effectiveness and flexibility of this approach, research communities other than the one of swarm robotics, for example biologists, might also benefit from this approach.

As described earlier, the ground in Phormica is painted with a photochromic substance that changes color from white to magenta when exposed to the UV light. This process is reversible: the color of the substance changes back to white when UV light is removed. However, UV exposure for an extended period can damage the molecular structure of the substance and might reduce its life span: it might no longer exhibit reversible photochromism. Indeed, the time this substance needs to return to white (color-decay-rate) is analogous to the evaporation time of naturally occurring pheromone. However, the formal mathematical models to determine the color-decay-rate of this substance are not available. Therefore, further study with more focus on determining color-decay-rates of all available colors of this type of photochromic substance is therefore suggested.

This substance is usually available in pigment form and could be used to produce tiles or sheets that would facilitate the creation of environments in which robot swarms could operate relying on stigmergic coordination. These tiles or sheets could be installed, for example, in a factory or office floor to enable stigmergic coordination in commercial environments. This could open the possibility of interaction between robots and humans: for instance, robots could draw a line on the floor or wall that humans can follow; and humans could also leave similar markers to interact with the robots.

The research in the field of material sciences is evolving rapidly and we might expect that, in the future, photochromic materials with a longer life span might be developed. We can also expect that a similar photochromic substance could become available that displays different colors depending on the wavelengths of the light to which it is exposed. Robots could then be able to leave or draw different

color markers to communicate different messages to humans and other robots.

3.5 Conclusion

In this chapter, I presented a cost-effective and functional system, Phormica, that enhances the capabilities of a swarm of e-puck robots and enables them to release and detect artificial pheromone in the environment. Phormica is based on a photochromic substance that emulates artificial pheromone when exposed to UV light. I demonstrated Phormica on three collective missions that benefit from stigmergic coordination. The results of my experiments indicated that robot swarms that take advantage of Phormica to achieve stigmergic coordination perform significantly better than the swarm that relies on other collective behaviors to accomplish the mission. Besides enabling stigmergic coordination, the technology I proposed in the thesis can also be used as a visualization tool to observe motion patterns, various random walks, or other behaviors without the help of any complex external infrastructure. The outcomes of this research demonstrate the significance and usability of Phormica in swarm robotics research.

Chapter 4

AutoMoDe-Habanero

In this chapter, I introduce Habanero, an offline fully-automatic design method belonging to the AutoMoDe family. This method offers a powerful tool for designing complex control software for robot swarms. I will first outline the methodology and experimental setup used to evaluate Habanero. Finally, I will present the results of these experiments, showcasing the potential of Habanero to advance swarm robotics research.

AutoMoDe is a general framework. To define a specific design method that conforms to it and produces control software to address a specific class of missions, the following steps must be taken: (1) select a target robot platform that is appropriate for the given class of missions, (2) define software modules for the selected robot platform, (3) specify the architecture into which the software modules will be assembled, (4) select a simulator to be used in the automatic design process, and (5) define an appropriate optimization algorithm to search the space of the possible ways in which the software modules can be assembled and tuned. My proposed AutoMoDe method, Habanero, designs collective behaviors to address missions in which the robot swarm relies on stigmergy to coordinate. The target robot platform is the e-puck (Mondada et al. 2009) augmented with the Overo Gumstix Linux board, the aforementioned hardware module that lays artificial pheromone trails by focusing UV light onto ground coated with photochromic material (Salman et al. 2020b), and an omni-directional camera to detect artificial pheromone trails—see Chapter 3. The software modules of Habanero are based on those previously defined for TuttiFrutti (Garzón Ramos and Birattari 2020), an other AutoMoDe method that generates control software for robots that can display colors via RGB LEDs and react to them. The main difference between TuttiFrutti and Habanero is that the latter features some original hardware and software devices

to lay and detect pheromone trails. The architecture into which these modules are assembled are probabilistic finite-state machines. The simulator used in the design process is ARGoS (Pinciroli et al. 2012) with an original library for the simulation of pheromone trails. The optimization algorithm utilized is irace (López-Ibáñez et al. 2016), as originally used in `TuttiFrutti` (Garzón Ramos and Birattari 2020) and in `Chocolate`, the state-of-the-art AutoMoDe method (Francesca et al. 2015). See Fig. 4.1 for a graphical illustration of `Habanero`, Fig. 4.2 for a description of the platform for which `Habanero` was developed, Fig. 4.3 for experimental setup and the Methods section (4.1) for further details. The collective behaviors designed by `Habanero` enable the robots to operate in a fully autonomous and distributed way without requiring any form of centralized control and coordination.

In this thesis, I demonstrate `Habanero` by generating control software for a swarm of eight e-puck robots. I consider four missions in which the robots should rely on stigmergy-based coordination: Aggregation, Decision Making, Rendezvous Point, and Stop. See Fig. 4.4 and the Methods section for details. To assess the quality of the control software produced by `Habanero`, I compare its performance to that of several alternatives: (1) control software produced via neuroevolution (`EvoPheromone`), shown in Fig. 4.5; (2) control software manually produced by human designers (`Human-Designers`) shown in Fig. 4.6; and (3) a random-walk behavior (`Random-Walk`), shown in Fig. 4.7.

The results of the experiments indicate that: (i) `Habanero` is a viable approach to designing pheromone-based stigmergy; (ii) it can produce control software that is comparable to, or even outperforms, control software produced by a human designer; and (iii) although its modules are conceived in a mission-agnostic way, the interaction strategies it devises are mission-specific.

4.1 Methods

4.1.1 Arena

All experiments were performed in a rectangular arena whose walls were realised with modular RGB blocks that display colors according to the mission requirements (Garzón Ramos and Birattari 2020; Garzón Ramos et al. 2022)—see Fig. 4.3. The technical diagrams of the arenas used in the thesis are shown in Fig. 4.4. The floor of the arena was white and coated with a photochromic material that acts as a medium to encode the pheromone trails (Salman et al. 2020b). The coating was realised using an acrylic binder with a 20% (w/w) concentration of photochromic

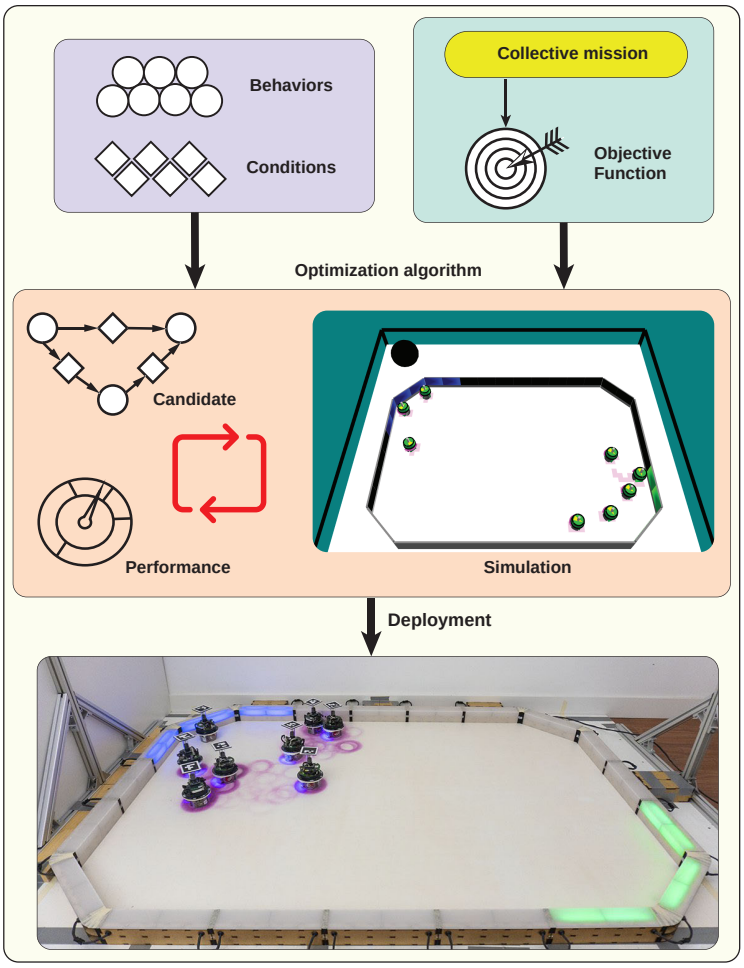


Figure 4.1: **AutoMoDe-Habanero.** Habanero automatically produces control software for e-puck robots by assembling predefined and mission-independent software modules into a probabilistic finite-state machine. A set of seven low-level behaviors and six transition conditions function as states and edges of the finite-state machine, respectively are listed in Table 4.2. Using the irace algorithm, the design process determines the topology of the finite-state machine by maximizing the performance of the robot swarm. The performance of an instance of control software is assessed in simulation, before the swarm is deployed.

pigments. Technical information to reproduce the arena is provided as Supplementary Note 5 (Salman et al. 2024). The photochromic material adopted turns magenta when exposed to UV light. Once the UV light is removed, the magenta color gradually fades and the floor returns white in about 50 s—see Supplementary Video 5 (Salman et al. 2024).

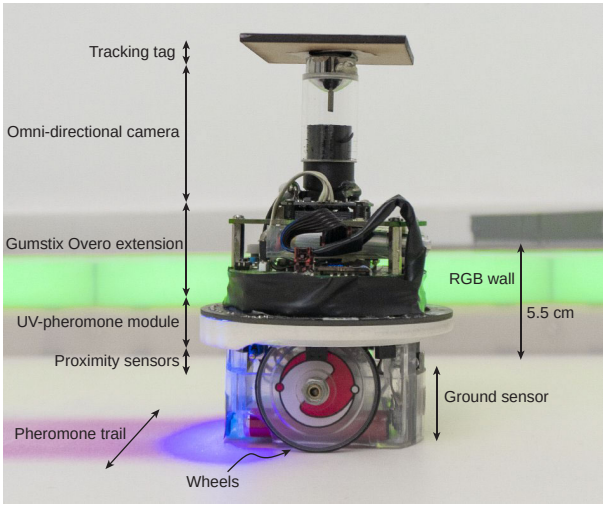


Figure 4.2: **The e-puck robot.** An e-puck robot equipped with a Linux board, a hardware module to focus UV light onto the ground, and an omni-directional camera.

4.1.2 The e-puck robot

The experiments were performed with e-puck robots—small sized differential-drive robots that are widely adopted in swarm robotics research (Allen et al. 2020; Mondada et al. 2009). I used an extended version of the e-puck that is equipped with the Overo Gumstix computer-on-module to run Linux on the robot; the ground sensor module to detect the gray-level color of the floor; a UV-light module and an omni-directional camera to deposit and detect artificial pheromone trails, respectively. The UV-light module is a ring shaped add-on module for e-puck that is equipped with nine down-facing UV LEDs positioned at the rear of the robot (Salman et al. 2020b). A picture of the hardware configuration of the e-puck robot adopted in the research is given in Fig. 4.2a. The capabilities of the e-puck for laying and detecting the artificial pheromone are illustrated in Supplementary Video 5 (Salman et al. 2024).

Reference model: the extended version of e-puck adopted is described by reference model RM 4.1, which formally defines the input and output variables associated with sensors and actuators, respectively—see Table 4.1. The control software of the robot reads/writes the input/output variables at every control step, which has a duration of 100 ms (Hasselmann et al. 2018a).

Simulator: all simulations were performed using ARGoS3 Version 48, along with the argos3-epuck-phormica library—see section Code Availability of Salman et al. 2024. ARGoS was specifically developed to simulate robot swarms (Pinciroli

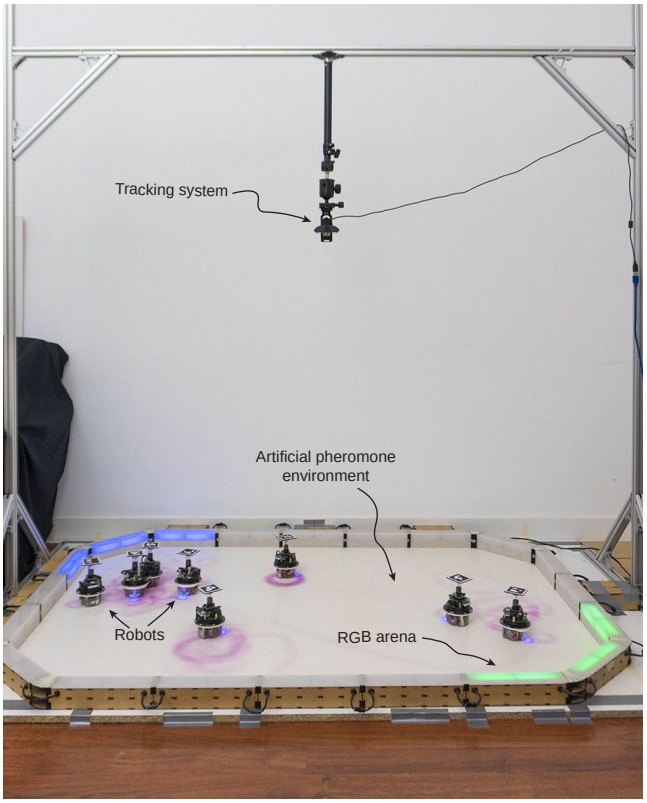


Figure 4.3: **The experimental setup.** The experimental arena. The floor is coated with photochromic material. It changes in color from white to magenta when exposed to UV light, and gradually returns to its normal white color when the UV light is removed. The walls of the arena are constructed using modular RGB (Red, Green, Blue) blocks, which have the ability to display various colors using the RGB color code. A tracking system is used to automatically measure performance indicators.

et al. 2012); the argos3-epuck-phormica library enables the cross-compilation of control software for the e-puck so that it can be ported to the robots without any manually applied modification.

4.1.3 `Habanero`

`Habanero` is an instance of AutoMoDe (Birattari et al. 2021) specialized in the design of swarm of robots that can lay and detect pheromone trails. `Habanero` produces control software by assembling predefined software modules into probabilistic finite-state machines in which states are low-level behaviors performed by the robots and transitions are enabled by conditions on the contingencies experienced by the robot.

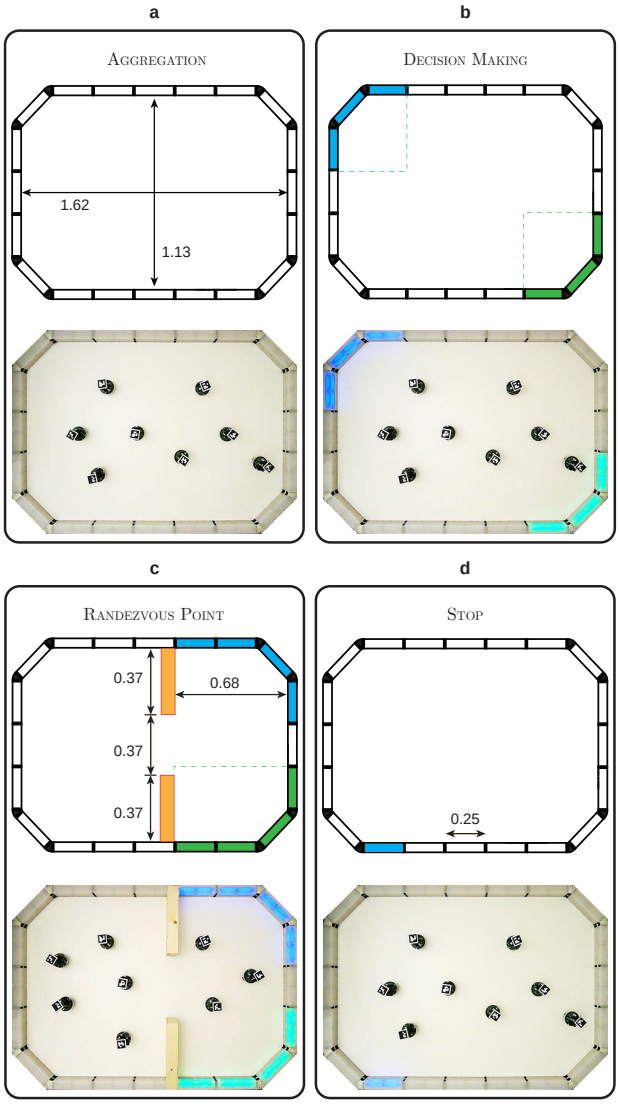


Figure 4.4: **Construction of the arenas for the four missions.** Technical drawings of the arena with dimensions and positions of different regions, along with photos of the real arena, in the four mission configurations: (**a**) Aggregation, (**b**) Decision Making, (**c**) Rendezvous Point, and (**d**) Stop. All measurements are expressed in meters. The missions are described in the Methods section.

`Habanero` operates on seven low-level behaviors and six conditions. Both low-level behaviors and conditions have free parameters that affect their functioning. The space of solutions that `Habanero` can produce comprises all the possible probabilistic finite-state machines—with at most 4 states and at most 4 outgoing transitions per state—that can be obtained by assembling the available modules and by fine-tuning their free parameters. There are a total of 105 parameters to be tuned—with categorical parameters for the selection of software modules;

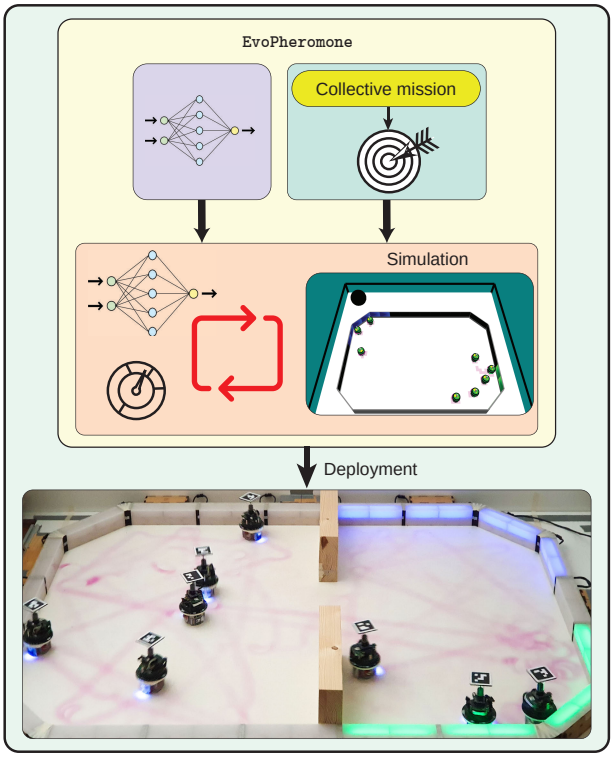


Figure 4.5: **Pictorial representation of `EvoPheromone`.** `EvoPheromone` is an implementation of the neuroevolutionary approach.

and categorical, integer and real parameters that affect their functioning. The optimization problem is mixed-variable in nature (Liao et al. 2014). `Habanero` searches this space using irace (López-Ibáñez et al. 2016) with the goal of maximizing a given mission-specific objective function. irace samples, fine-tunes and selects candidate solutions performing simulations in ARGoS3. There is a limited number of simulations available to `Habanero` to produce an instance of control software—a simulations budget. Once the budget is exhausted, `Habanero` returns the best control software found up to that moment. A pictorial representation of `Habanero` is given in Fig. 4.1.

The seven low-level behaviors are: exploration, stop, go-to-color, avoid-color, go-to-pheromone, avoid-pheromone, and waggle. The six conditions are: white-floor, gray-floor, black-floor, color-detected, pheromone-detected, fixed-probability—see Table 4.2. All the low-level behaviors and the conditions interact with the e-puck hardware (sensors and actuators) via the input/output variables defined in reference model RM 4.1—see Table 4.1.

I chose irace to conduct `Habanero`'s optimization process as, for historical

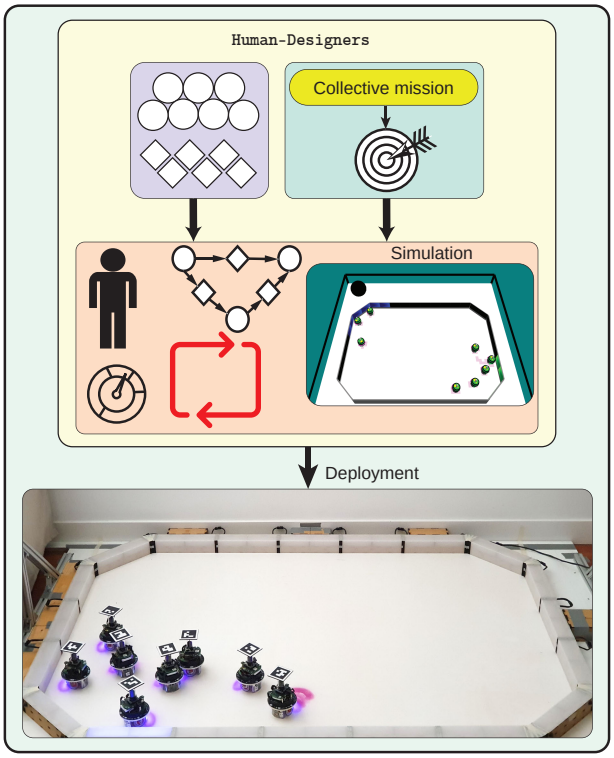


Figure 4.6: **Pictorial representation of the `Human-Designers` design method.** `Human-Designers` is a manual design method.

reasons, it is the de facto standard optimization algorithm in the AutoMoDe family. Notably, irace outperformed human experts in the modular design of control software for robot swarms (Francesca et al. 2015). Moreover, irace was successful when applied to the problem of producing collective behaviors with a diverse set of AutoMoDe methods (Birattari et al. 2021). irace has properties that make it suitable to tackle problems in the automatic modular design of control software. Particularly, it was conceived for the statistical selection of candidate solutions when (i) the problem instances are stochastic and (ii) the solutions comprise discrete and continuous parameter spaces (Balaprakash et al. 2007; Birattari et al. 2010; López-Ibáñez et al. 2016). Recent studies have shown that other optimization algorithms are suitable for the AutoMoDe family—e.g., simulated annealing (Kirkpatrick et al. 1983) and sequential model-based algorithm configuration (Hutter et al. 2011; Lindauer et al. 2022). However, there is no evidence that indicates that they offer a definite advantage over irace—see (Kuckling 2023) for a recent in-depth discussion.

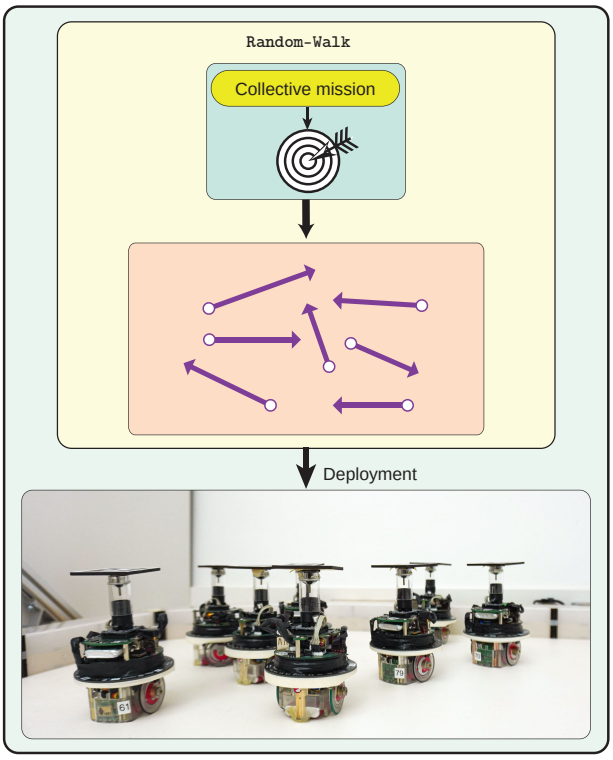


Figure 4.7: **Pictorial representation of `Random-Walk`.** Although `Random-Walk` is not a design method, I include it to serve as a lower bound of performance. See the Methods section for the details.

4.1.4 Comparisons

`EvoPheromone` is an adaptation of `EvoStick`, which is a standard neuroevolutionary method to design robot swarms (Francesca et al. 2014b). `EvoPheromone` produces control software for an extended version of the e-puck robot formally described by reference model RM 4.1—same as `Habanero`. The architecture of the control software is a fully connected feed-forward artificial neural network. The neural network has 61 input nodes, 7 output nodes, and no hidden layer. The input and output nodes are directly connected by synaptic connections with weights. There are a total of 427 parameters to be tuned—all real values, which encode the synaptic weights. The optimization problem is continuous in nature (Liao et al. 2014). `EvoPheromone` tunes the synaptic weights of the neural network via elitism and mutation (Francesca et al. 2014b). The evolutionary process is based on simulations executed in ARGoS3 with the argos3-epuck-phormica library—same setting as `Habanero`. The design process ends when a predefined simulation budget is exhausted. I developed `EvoPheromone` on the basis of `EvoStick`, as the latter

Table 4.1: **Reference Model RM 4.1.** The reference model RM 4.1, which formally describes the interface between the robot and the control software. The difference between RM 4.0 and RM 4.1 is highlighted in gray. In RM 4.1, the brightness of UV LEDs is constant, and the robots can lay either a thin or thick trail of pheromone, or no trail at all.

Sensor	**Input**	**Value**
Proximity	$prox_{i\in\{1,\dots,8\}}$	$[0,1]$
Ground	$ground_{j\in\{1,2,3\}}$	$\{black, gray, white\}$
Omni-directional camera	$cam_{c\in\{R,G,B,C,M,Y\}}$	$\{yes, no\}$
	$V_{c\in\{R,G,B,C,M,Y\}}$	$(1.0; [0,2]\,\pi\text{rad})$

Actuator	**Output**	**Value**
Motors	$v_{k\in\{l,r\}}$	$[-0.12, 0.12]\,\text{m}\,\text{s}^{-1}$
UV-Pheromone-Module	phe	$\{none, thin, thick\}$

is a readily available method for the e-puck that has served as a yardstick to apprise the performance of AutoMoDe methods in the past (Francesca et al. 2015, 2014b). `EvoStick` is the only neuroevolutionary method that has been tested in the automatic design of robot swarms for several missions, without undergoing any mission-specific modification (Hasselmann et al. 2021). Moreover, `EvoStick` served as a starting point to develop other neuroevolutionary methods for robots endowed with enhanced capabilities—see, for example, adaptations of `EvoStick` to study direct communication (Garzón Ramos and Birattari 2020; Hasselmann and Birattari 2020) and spatial organization (Mendiburu et al. 2022). `EvoStick`, and therefore `EvoPheromone`, are simple and straightforward implementations of the neuroevolutionary approach. I do not consider more advanced neuroevolutionary methods—e.g., `CMA-ES` (Hansen and Ostermeier 2001), `xNES` (Glasmachers et al. 2010), and `NEAT` (Stanley and Miikkulainen 2002)—as previous research has shown that they do not provide any performance advantage over `EvoStick` when applied off the shelf (Hasselmann et al. 2021).

`Human-Designers` is a manual design method in which 10 human designers were requested to produce control software using the software modules of `Habanero`. In a sense, a human designer acts as an optimization agent that assembles a finite-state machine and fine-tunes its parameters. `Human-Designers` produces control software for an extended version of the e-puck robot formally described by reference model RM 4.1—same as `Habanero`. The human designers who participated in this research had various levels of expertise in swarm robotics—ranging from bachelor

students to post-doctoral researchers in swarm robotics. Seven of them had previous experience with real robots, seven had previous experience with ARGoS3, and six had experience with the e-puck—either in simulation or reality. I provided the designers with a visualization tool to produce and manipulate finite-state machines, to visualize simulations, and to compute the value of the objective function (Kuckling et al. 2021). All simulations were executed in ARGoS3 with the argos3-epuck-phormica library—same setting as `Habanero`. The designers were allotted 4 hours per mission—see Supplementary Note 4 (Salman et al. 2024). The guidelines and experimental description given to the designers are provided as Supplementary Note 3 (Salman et al. 2024).

`Random-Walk`, although not an automatic design method, is included in the thesis as a lower bound on the performance of robot swarms. In `Random-Walk`, the robots move straight in the arena, when they encounter an obstacle, they rotate for a random number of control steps and then resume their straight motion. `Random-Walk` was conceived for an extended version of the e-puck robot formally described by reference model RM 4.1—same as `Habanero`.

4.1.5 Missions

The empirical study is based on four missions. Each mission must be performed within $T = 180\,\text{s}$ by a swarm of $N = 8$ robots. The size of the swarm was determined in accordance with the number of robots available for the experiments.

Aggregation: initially, the robots are randomly placed in the arena—see Fig. 4.4a. The robots must approach one another to form a cluster and remain close until the end of the mission. Formally, the mission is specified by the following objective function, which must be minimised:

$$F_a = \sum_{t=1}^{T/100\,\text{ms}} d_{\text{avg}}(t). \tag{4.1}$$

At each control step t, the average distance d_{avg} between the robots is added to F_a.

Decision Making: initially, the robots are randomly placed in the arena—see Fig. 4.4b. The robots must select between a green and a blue region: at every control step t, the score is increase by $+1$ for every robot that is in the green region, and by $+2$ for every robot that is in the blue one. Both green and blue light signals disappear after a random amount of time, which is uniformly sampled between 70 and 90 s. Formally, the mission is specified by the following objective function,

which must be maximised:

$$F_d = \sum_{t=1}^{T/100\,\text{ms}} \sum_{i=1}^{N} I_i(t); \qquad I_i(t) = \begin{cases} 1 & \text{if robot } i \text{ is in green region,} \\ 2 & \text{if robot } i \text{ is in blue region,} \\ 0 & \text{otherwise.} \end{cases} \tag{4.2}$$

Rendezvous Point: initially, the robots are placed in the left side of the arena. The robots must reach the green region and stay there until the end of the mission. A blue region is added as a decoy to possibly confuse the robots—see Fig. 4.4c. Both green and blue light signals disappear after a random amount of time, which is uniformly sampled between 70 and 90 s. Formally, the mission is specified by the following objective function, which must be maximised:

$$F_r = K_{\text{in}} - K_{\text{out}}; \tag{4.3}$$

where K_{in} is the number of robots inside the green region at the end of the mission, and K_{out} is the number of robots outside.

Stop: initially, the robots are randomly placed in the arena. A blue light signal appears after a random amount of time $\bar{t}$, which is uniformly sampled between 70 and 90 s—see Fig. 4.4d. All the robots must stop as soon as the signal appears, but not before. Formally, the mission is specified by the following objective function, which must be minimised:

$$F_s = \sum_{t=1}^{\bar{t}} \sum_{i=1}^{N} \bar{I}_i(t) + \sum_{t=\bar{t}+1}^{T} \sum_{i=1}^{N} I_i(t); \qquad I_i(t) = \begin{cases} 1 & \text{if robot } i \text{ is moving,} \\ 0 & \text{otherwise;} \end{cases} \qquad \bar{I}_i(t) = 1 - I_i(t). \tag{4.4}$$

In the absence of well-established benchmark missions, we chose a set of missions that allowed us to estimate the expected performance of `Habanero` in typical swarm robotics tasks. Aggregation, Decision Making, Rendezvous Point and Stop are missions that belong into the same class—they allow the pheromone-based coordination of robots. Yet, they are sufficiently different to benefit from a tailored design—they vary in the nature of their goals and in the presence of reference points of interest. By selecting a varied set of missions, I also aimed at testing `Habanero`'s ability to handle diverse challenges without undergoing any mission-specific adjustment.

It is worth noting that these missions—likewise `Habanero`—are not suitable for drawing conclusions on whether automatic methods can handle more complex

missions or design relatively more complex stigmergy-based interactions. For instance, missions that require precise behavioral control via careful modulation of the pheromone deposition and response, or missions that involve more complex communication strategies through various types of pheromones.

4.1.6 Protocol

All experiments were executed without any human intervention or any mission-specific modification in the design process. For both `Habanero` and `EvoPheromone`, for each mission, I independently executed the design process 10 times to obtain 10 instances of control software. Both methods operated with a budget of 100 000 simulation runs for each execution of the design process. I executed all automatic design processes on a high-performance computational cluster with about 1500 computing cores. In case of `Human-Designers`, 10 human designers were involved and each of them produced one instance of control software for each mission. After obtaining all the instances of control software, I assessed their performance once in simulation and once in reality. I varied the initial position of the robots when assessing instances of control software of a single method, and I used the same set of initial positions across the four methods. To perform the experiments in reality, the instances of control software, regardless of the design method that produced them, were automatically cross-compiled and deployed on the e-puck robots without undergoing any manually-applied modification.

Tracking System. I used a tracking system to automatically compute the performance of a robot swarm during each run of a real-robot experiment (Legarda Herranz et al. 2022). The tracking system uses an overhead camera to record the positions of the robots by recognizing squared markers mounted on the robots. I also used the overhead camera to record videos of the experiments—see Supplementary Video 6 (Salman et al. 2024). The overhead camera was used only to measure the performance of the swarm and was not used to provide any information to the robots.

4.1.7 Statistics

I present the performance of the different methods with notched box-and-whiskers plots on a per-mission basis. In these plots, boxes represent the interquartile range, covering the central 50 % of the values observed. Whiskers extend from the lower quartile to the lowest recorded performance, and from the upper quartile to the

highest one. The horizontal line in the middle of each box plot represents the median performance, and the notches on the box represent a 95% confidence interval on the median. If the notches of two boxes do not overlap, then the difference between their respective medians is significant, with a confidence of at least 95% (Chambers et al. 1983). For each method, I present the performance obtained in simulation and in real-robot experiments using thin and thick boxes, respectively. I executed a mission-specific comparison of the performance of methods with Wilcoxon paired rank sum tests at 95% confidence (Conover 1999).

I also performed a Friedman rank sum test (Conover 1999) that aggregates the performance of each method across all four missions. More precisely, I applied a Friedman two-way analysis of variance to the performances recorded in the experiments with physical robots, across all missions, and for all methods. The Friedman test is nonparametric and implements a block design. In my protocol, the treatment factor is the method under analysis and the blocking factor is the mission. By operating on the ranks, the Friedman test is invariant to the magnitude of the objective functions of the missions considered. Also, due to its nonparametric nature, it can be applied with no assumption on the distribution of the performance. These properties are instrumental for aggregating the performance observed across the four missions. I present the results of the test with the average rank of each method (computed across all missions), and its 95% confidence interval. A method is significantly better than other if it has a lower average rank and the confidence interval of the two methods do not overlap.

4.2 Results

`Habanero` designed stigmergy-based collective behaviors that proved to be effective: the robots used the artificial pheromone to complete each mission in a way that is meaningful and appropriate to the mission considered. Statistical analysis shows that the control software generated by `Habanero` performed significantly better than the alternatives included in the empirical study. In the following sections, I first present the results on a per-mission basis, and then I aggregate them across all missions. Simulation-only experiments with different swarm sizes are provided as well. I also provide an analysis of the robustness to the reality gap.

4.2.1 Aggregation

In this mission, the robots must aggregate anywhere in the arena. To aggregate, the robots cannot rely on any form of direct communication nor on the ability to directly sense the presence of their peers in their vicinity. The only way in which they can coordinate is the laying and detecting of artificial pheromone trails. They can leverage this ability to attract their peers and aggregate using stigmergy. However, as all robots could release some pheromone at the same time in different areas, they could saturate the environment and/or be trapped in the local accumulation of their own pheromone emissions.

`Habanero`, `EvoPheromone`, and `Human-Designers` produced control software that performed equivalently well in simulation—see Fig. 4.8a. However, when transferred to the real robots, the control software produced by `Habanero` performed significantly better than the one produced by all other design methods. `Habanero`

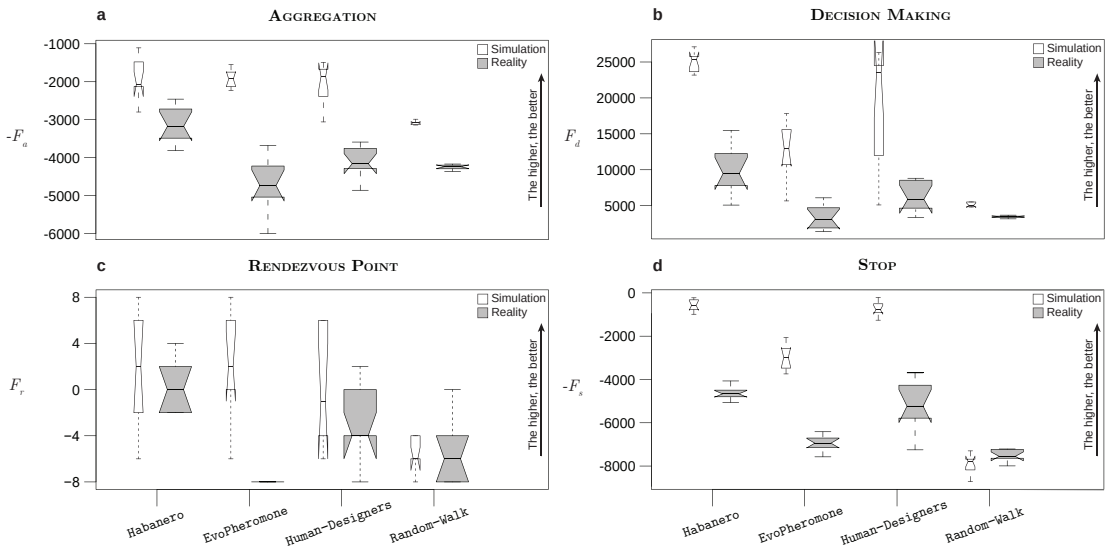


Figure 4.8: **Results of the empirical analysis**. I report results of the evaluation of 160 instances of control software, 10 per method and per mission. All instances of control software were evaluated once in simulation and once with physical robots—more details on the protocol are provided in Methods. The results are presented using boxplots on a per-mission basis: (**a**) Aggregation, (**b**) Decision Making, (**c**) Rendezvous Point, and (**d**) Stop. In all missions, for each method, I report the performance obtained in simulation and with physical robots using thin and thick boxes, respectively.

produced collective behaviors in which the robots laid pheromone trails only for short periods of time and kept searching the environment for pheromone traces left

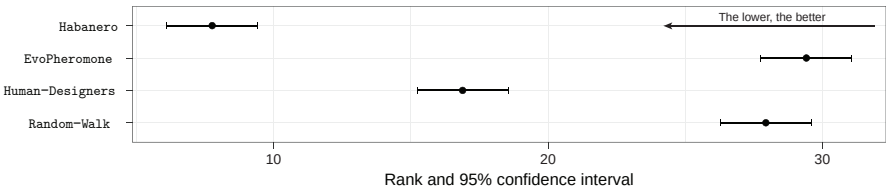


Figure 4.9: **Aggregate performance**. Friedman rank sum test on real-robot performance to aggregate the overall performance of each method across the four missions—the lower the rank, the better. An explanation of the graphical convention adopted in the boxplots and in the Friedman test are provided in the Methods section under the heading Statistics.

by their peers. By laying pheromone trails only intermittently, the robots avoided saturating the environment and marked only isolated spots, which then served as aggregation points. Around these points, they eventually gathered in clusters—see Fig. 4.10 and Supplementary Video 1 (Salman et al. 2024).

`EvoPheromone` produced a different strategy: the robots laid pheromone trails while moving along a circular trajectory and followed the pheromone trails to gather at places where pheromone concentration was high. This strategy produced good results in simulation but not on the real robots. The robots did not properly avoid the walls and failed to reproduce the behavior observed in simulation.

The control software produced by `Human-Designers` continuously laid pheromone trails with the expectation that all robots would gather at one place. Results were good in simulation but failed to transfer to reality. In the real-robot experiments, the robots remained trapped in local pheromone accumulations. Eventually, they gathered in separate clusters.

4.2.2 Decision Making

In this mission, the robots must make the decision to congregate in one of two regions of the arena, designated by RGB blocks that display blue or green color, respectively—see Fig. 4.4b. Each robot scores one point for each time step spent in the green region and two points for each time step spent in the blue one. Halfway through each run of the experiment, the blue and green RGB blocks are switched off, leaving the robots without any visual cue to identify the two regions. In order to maximise the score, the robots must quickly congregate in the region that provides the highest score per time step—i.e., the blue one—and remain there even once

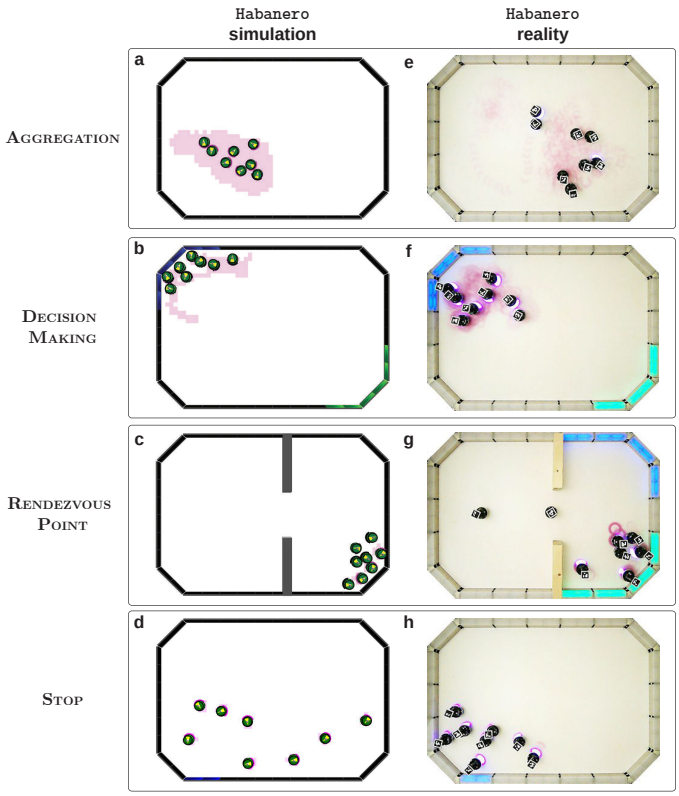


Figure 4.10: **Execution in simulation and reality**. For each mission, I show: a snapshot of robots executing an instance of `Habanero` control software in (**a-d**) simulation and (**e-h**) real-robot experiments.

the environmental cues are removed.

When evaluated in simulation, the control software produced by `Habanero` and `Human-Designers` performed equally well, and significantly better than the one produced by `EvoPheromone`—see Fig. 4.8b. However, in the real-robot experiments, the control software produced by `Habanero` performed significantly better than that of `Human-Designers`. The control software produced by both `Habanero` and `Human-Designers` performed significantly better than that of `EvoPheromone`, which obtained results comparable with those of `Random-Walk`.

In all experimental runs, the robot swarm designed by `Habanero` correctly selected the blue region to congregate. The robots relied on stigmergy not only to attract other robots to the blue region, but also to stay there after the cues were removed. The behavior displayed in the real-robot experiments was qualitatively similar to the one displayed in simulation—see Fig. 4.10 and Supplementary Video 2 (Salman et al. 2024). However, in the real-robot experiments, some

robots that gathered in the blue region spilled out of the boundaries of the region, although remaining in its vicinity. Because of this, the performance in the real-robot experiments was lower than that in simulation. The robot swarm generated by `EvoPheromone` was unable to congregate in a single region: the robots stayed in the first region in which they entered. Consequently, the score was significantly worse than the one obtained by other design methods. The robot swarm produced by `Human-Designers` was able to correctly congregate in the blue region but was unable to remain there once the cues were removed.

4.2.3 Rendezvous Point

In this mission, a wall with a narrow gate laterally divides the arena into two sections: the left side, where the robots are deployed at the beginning of the experiment; and the right side, which contains two regions designated by RGB blocks that display blue or green color, respectively—see Fig. 4.4c. Similar to Decision Making, halfway through each run of Rendezvous Point, the blue and green RGB blocks are switched off, leaving the robots without any visual cue to identify the two regions. The robots must cross the narrow gate to gather in the green region. The score is given by the number of robots that, at the end of the experimental run, are positioned in the green region.

When evaluated in simulation, the control software produced by all design methods performed equally well—see Fig. 4.8c. However, in the real-robot experiments, the control software produced by `Habanero` performed significantly better than the one produced by all other methods. Moreover, the one produced by `EvoPheromone` performed significantly worse than that produced by all other methods.

The robot swarms designed by `Habanero` relied on random walk to cross the gate and find the green region. Once the robots reached the green region, they took advantage of stigmergy to attract their peers and to keep themselves inside the region even when the green light was removed. The robots laid pheromone trails to mark the green region and kept laying the pheromone trails at that place to avoid fading—see Fig. 4.10 and Supplementary Video 3.

In the control software produced by `EvoPheromone`, the robots do not randomly search for the narrow passage. Instead, they move along the walls of the arena to eventually cross the gate and reach the green region—see Supplementary Video 3 (Salman et al. 2024). Although this behavior worked effectively in simulation, it failed in the real-robot experiments: the robots were unable to move along the walls and remained stuck. Consequently, they were unable to cross the gate.

In the real-robot experiments, the performance of the robot swarm designed by EvoPheromone was even significantly worse than that of Random-Walk.

In the control software produced by Human-Designers, the robots were mostly able to reach the green region. However, the swarm produced by Human-Designers was not always effective in using stigmergy to remain in the green region, especially after the green light was removed.

4.2.4 Stop

In this mission, the robots must halt and stand still as soon as a stop signal is perceived. The stop signal is a (random) RGB block that switches on at a random moment in time and emits blue light—see Fig. 4.4d. Before the signal, each robot scores one point for each time step during which it moves. After the signal, each robot scores one point for each time step during which it stays in place. As the robots considered in this thesis are incapable of direct communication, the individuals that detect the signal can only rely on stigmergy to inform any peers that are in a position from which the signal cannot be seen.

The control software produced by Habanero and Human-Designers performed similarly well when evaluated both in simulation and reality, and performed significantly better than the one produced by EvoPheromone—see Fig. 4.8d.

In the robot swarms designed by Habanero, the robots kept moving to search for a block emitting the stop signal. As soon as a robot detected the signal, it stopped or started waggling in place, while laying a pheromone trail to alert its peers. Other robots also stopped and started laying pheromone trails either after detecting the signal or the pheromone trails laid by their peers—see Supplementary Video 4.

Human-Designers produced collective behaviors similar to those generated by Habanero, and so no significant difference in the performance could be observed—see Fig. 4.10.

The collective behaviors produced by EvoPheromone achieved good scores in some cases, but were unable to accomplish the mission in its true sense. The robots took advantage of stigmergy to gradually repel each other, approach the walls, and eventually stop against them. The evolutionary process tuned the timing of the behavior to match the typical amount of time that elapsed between the beginning of the experiment and the moment when the blue signal appeared. This allowed the robots to score points by moving towards the walls before the appearance of the signal and remaining still against the walls after the appearance of the signal.

Although this behavior was reasonably well synchronized with the typical case, its failure to properly react to the appearance of the signal prevented it from achieving good scores consistently. Consequently, the performance achieved by EvoPheromone is significantly worse than the one achieved by both Habanero and Human-Designers.

4.2.5 Aggregate results

To aggregate the performance of each design method across the four missions, I used a Friedman rank sum test on the performance observed in the real-robot experiments. The test indicates that, in the experiments presented, Habanero outranked all other design methods, with a confidence of at least 95%—see Fig. 4.9. Human-Designers performed significantly better than both EvoPheromone and Random-Walk.

Figure 4.11 shows the aggregated execution time of the behavior modules in the finite-state machines produced by Habanero and Human-Designers—measured in simulation. Results indicate that the finite-state machines produced by Habanero and Human-Designers are different: the execution time of the behavior modules is different in Habanero and Human-Designers across all missions. Although Habanero and Human-Designers used the same set of modules, they combined them in a different way. The aggregated execution-time plot highlights four major differences between Habanero and Human-Designers. First, Habanero used the Exploration module considerably less than Human-Designers. Second, Habanero relied more on modules that react to pheromone information compared to Human-Designers. Third, Human-Designers employed for a longer time the modules that respond to the walls' color compared to Habanero. Finally, Habanero made greater use of the Waggle module than Human-Designers.

While my experiments highlight performance differences between the two methods, I cannot definitively determine how the design choices made by Habanero and Human-Designers influence the overall performance. More precisely, my experimental setup cannot adequately explain the rationale behind the selection, tuning, and combination of the modules for either Habanero or Human-Designers, and its relationship with the performance obtained.

4.2.6 Reality-gap

In Fig. 4.12, I present the reality gap observed in the experiments conducted in this thesis. In both AGGREGATION and RENDEZVOUS POINT, `Habanero` demonstrates a smaller performance drop compared to `EvoPheromone`. In STOP, the performance drop is similar for `Habanero`, `EvoPheromone`, and `Human-Designers`. However, in DECISION MAKING, the performance drop for `EvoPheromone` is less than that of `Habanero`. Yet, this is due to a sort of a floor effect: the performance of `EvoPheromone` was already particularly poor in simulation and this bounded its drop. All in all, although `Habanero` experienced a larger drop, it still outperformed `EvoPheromone`—see Fig. 4.8 and Fig. 4.9.

The performance of `Random-Walk` remained largely consistent. This is because `Random-Walk` is not a design method, meaning there is no optimization process involved and therefore overfitting does not happen.

I refer the reader to Francesca et al. 2015; Hasselmann et al. 2021; Ligot and Birattari 2020 for an in-depth discussion on the effects of the reality gap in modular and neuroevolutionary methods.

4.2.7 Scalability

In a robot swarm, the behavior of each individual robot is affected by the actions of its neighboring peer. It is therefore to be expected that the performance of the swarm might vary with the robot density, and therefore be affected by the number of robots and by the size of the environment. In this thesis, due to the limitations of the available experimental equipment, I was constrained to perform real-robot experiments with eight robots. Therefore, I explored the scalability of the stigmergy-based behaviors produced by `Habanero` through experiments in simulation. I considered nine scenarios characterized by different combinations of three arena sizes and three swarm sizes.

The three arena sizes are categorized as small ($1.7\,\mathrm{m}^2$), medium ($4.5\,\mathrm{m}^2$), and large ($12.0\,\mathrm{m}^2$). The ratio of the surface area of the small to the medium arena is similar to the ratio of the medium to the large arena, which is approximately 2.6. As mentioned previously, the arena walls are built with 25cm wide RGB color blocks. To maintain a similar arena shape to the real robot experiments and keep the zone sizes consistent across different experiments, these three arenas with three swarm sizes provided the most suitable robot-to-arena surface ratios. The layout of the three arenas is shown in Fig. 4.13, while their simulation counterparts are shown in Fig. 4.14. For the DECISION MAKING, RENDEZVOUS POINT, and STOP missions,

I designated specific zones and obstacles according to the mission objectives. The three different swarm sizes consist of 8, 22, and 56 robots, respectively.

I conducted simulations to evaluate the instances of the control software produced by Habanero for each mission across the nine scenarios. I also evaluated the control software produced by EvoPheromone. The performance of the robot swarms is shown in Fig. 4.15. The results suggest a complex relationship involving the number of robots in the swarm, the size of the arena, and performance.

In Aggregation, swarm performance decreases with an increase in swarm size. This trend is consistent for both Habanero and EvoPheromone. The larger the swarm size, the greater the likelihood that local robot clusters form,, thereby increasing the average distance between robots and reducing the performance.

In Decision Making, performance also decreases with increasing swarm size. In a smaller arena, scoring zones are spatially constrained, unable to accommodate all robots, whereas, in medium and large arenas, robots require more time to locate the zone. It should be noted that once half the experiment's duration has passed, the zone indication lights are switched off, leaving the robots to depend on stigmergy to locate and remain in the correct zone.

Performance achieved in Rendezvous Point showed no significant improvement or decline for control software produced by both Habanero and EvoPheromone. In Stop, however, Habanero performed progressively better as swarm size increased. In contrast, EvoPheromone's performance in a small arena rose with an increase in swarm size, but for the medium and the large arenas, performance rose initially with swarm size but followed by a decline with further increase in the swarm size.

4.2.8 Discussion

The AutoMoDe-Habanero demonstrated a remarkable potential to automatically design stigmergy-based collective behaviors that proved successful across all considered collective missions. This accomplishment highlights the effectiveness of the automatic design approach in generating tailored control software for robot swarms to accomplish diverse missions. Habanero's ability to identify appropriate pheromone utilization strategies for each mission underscores its adaptability. Although the software modules were designed with mission-agnostic principles, the resulting interaction strategies were mission-specific, demonstrating Habanero's capacity to fine-tune solutions.

A key observation is that the inherent limitations of individual robots, such as limited perception, computation, spatial coordination, memory, and communication,

were effectively compensated for at the swarm level through pheromone-based stigmergy. The e-puck robots, despite their individual limitations, exhibited spatial organization, external memory, and communication within the swarm, all facilitated by the intelligent use of pheromones.

The scalability of the behaviors generated by Habanero was investigated through simulations involving different arena sizes and swarm populations. While direct performance comparisons across varying scenarios were not feasible, scaled performance metrics revealed a general trend of increased swarm performance with larger swarm sizes. However, performance per robot exhibited variability, which is expected and does not indicate decreased overall swarm performance with more robots.

The current study employed an existing technology for pheromone-based stigmergy with real robots, the photochromic artificial pheromone system. While practical, this technology has limitations, particularly its suitability for indoor environments and the need for pre-preparation with photochromic material. The development of universally applicable marking technologies remains an open challenge. Furthermore, the vision system of the e-puck robots, limited to detecting only the presence or absence of pheromone, could be enhanced to enable the detection of pheromone freshness. This advancement could empower robots with improved decision-making capabilities by distinguishing between old and new pheromone trails.

In conclusion, this research underscores the potential of AutoMoDe-Habanero for the automatic design of effective swarm behaviors. The emergent capabilities observed in the swarms highlight the power of stigmergy in overcoming individual limitations. Addressing the current technological limitations and exploring future directions, such as enhanced pheromone detection, could further unlock the potential of swarm robotics and expand its applications in diverse domains.

Table 4.2: **Habanero's low-level behaviors and transition conditions.** While performing all the low-level behaviors, the robot releases thin or thick pheromone trails if *phe* is set to *thin* or *thick*, respectively. Otherwise, if *phe* is set to *none*, the robot does not release a pheromone trail. The parameter $fov \in \{\frac{1}{12}\pi, 2\pi\}$ determines the field of view of the camera. The parameter $\tau \in \{1, .., 100\}$ denotes the number of control steps for which a robot rotates in place while performing the exploration behavior: a control cycle is 100 ms. The parameter $\beta \in [0, 1]$ determines the probability of transitioning in all transition conditions. The parameter $c \in \{R, G, B, C, Y\}$ denotes the color to which the robots react when performing a particular behavior or transition from color-detected behavior to another.

Low-level behaviors	**Parameters**	**Description**
Exploration	phe, τ	Robot moves by random walk
Stop	phe	Robot stops in place
Go-to-Color	phe, c, fov	Robot moves toward objects displaying a specific color
Avoid-Color	phe, c, fov	Robot moves away from objects displaying a specific color
Go-to-Pheromone	phe, fov	Robot moves towards pheromone perceived in the surroundings
Avoid-Pheromone	phe, fov	Robot moves away from pheromone perceived in the surroundings
Waggle	phe	Robot rotates in place for a random period of time

Transition conditions	**Parameters**	**Description**
White-Floor	β	White floor detected
Gray-Floor	β	Gray floor detected
Black-Floor	β	Black floor detected
color-Detected	β, c, fov	Objects of a specific color perceived
Pheromone-Detected	β, fov	Pheromone detected in the surroundings
Fixed-Probability	β	Transition with a fixed probability

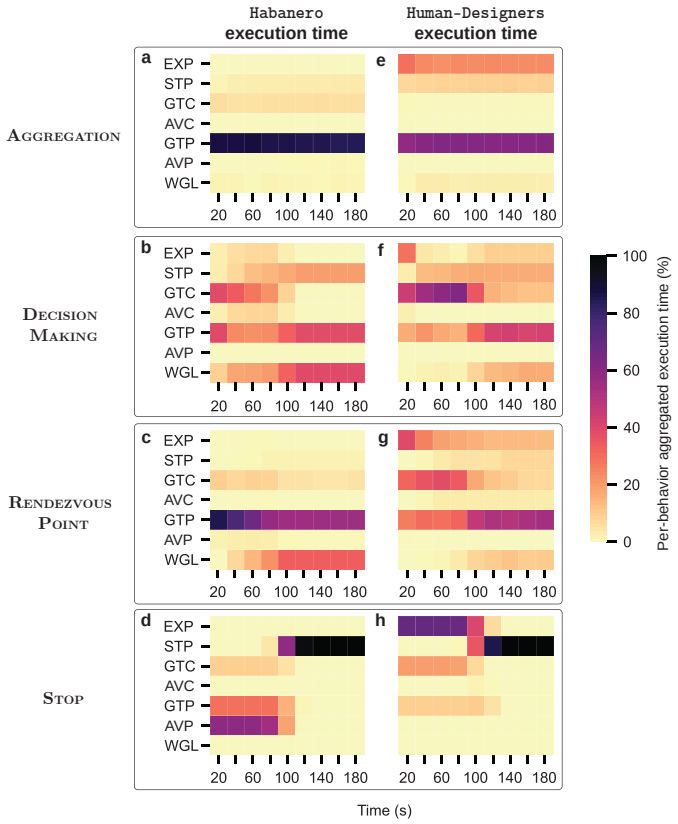


Figure 4.11: **Behaviors produced by `Habanero` and `Human-Designers`**. EXP, STP, GTC, AVC, GTP, AVP, and WGL are the short forms of the behaviors Exploration, Stop, Go-to-Color, Avoid-Color, Go-to-Pheromone, Avoid-Pheromone, and Waggle, respectively. For each mission, I show a plot of the aggregate execution time of each software module in the control software produced by (**a-d**) `Habanero` and (**e-h**) `Human-Designers`. I use the aggregate execution time of the modules to qualify the behavior I observe in the robot swarms. In the aggregate plots, the color gradient shows the percentage of time one behavior was executed throughout all instances of control software produced for a mission. I identify the behavior modules using the labels defined in Fig. 4.1.

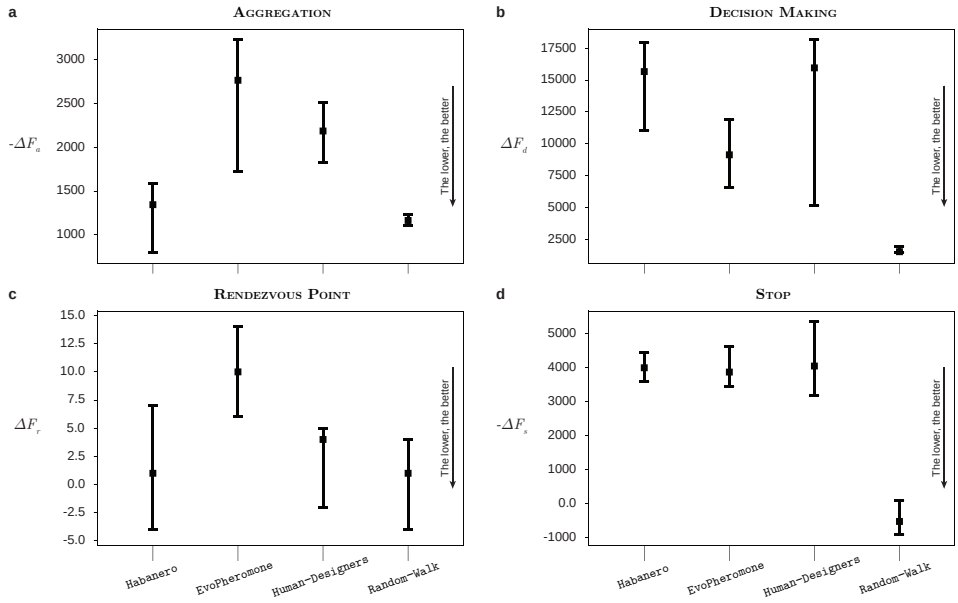


Figure 4.12: **Performance drop.** I present the performance drop using error bars on a per-mission basis: (**a**) Aggregation, (**b**) Decision Making, (**c**) Rendezvous Point, and (**d**) Stop. Vertical segments represent the 95% confidence interval in the median, computed using the Wilcoxon's Signed Ranks statistics.

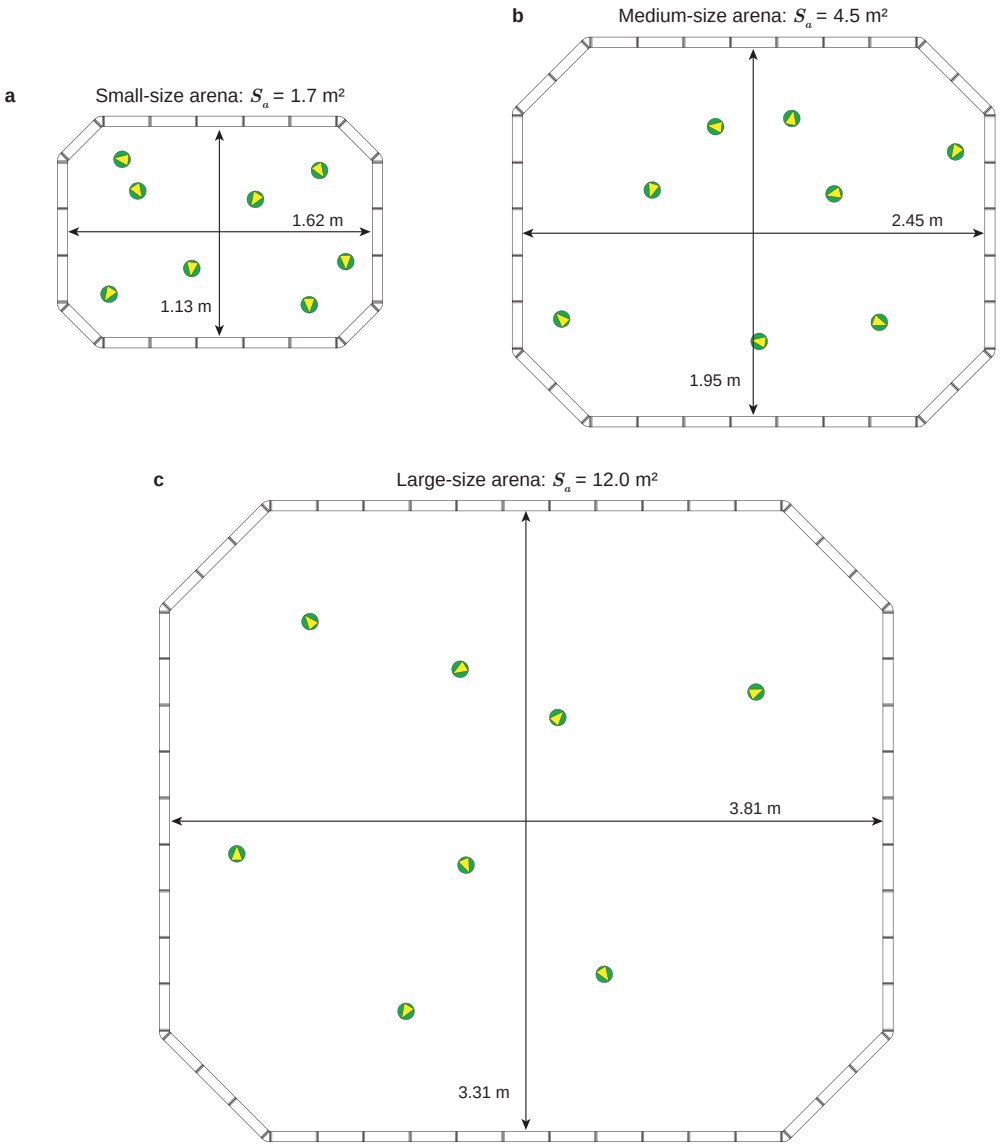


Figure 4.13: **Layout of the arenas for scalability experiments.** (**a**) Small-size arena. This arena is the one used in the real-robot experiments. (**b**) Medium-size arena. (**c**) Large-size arena.

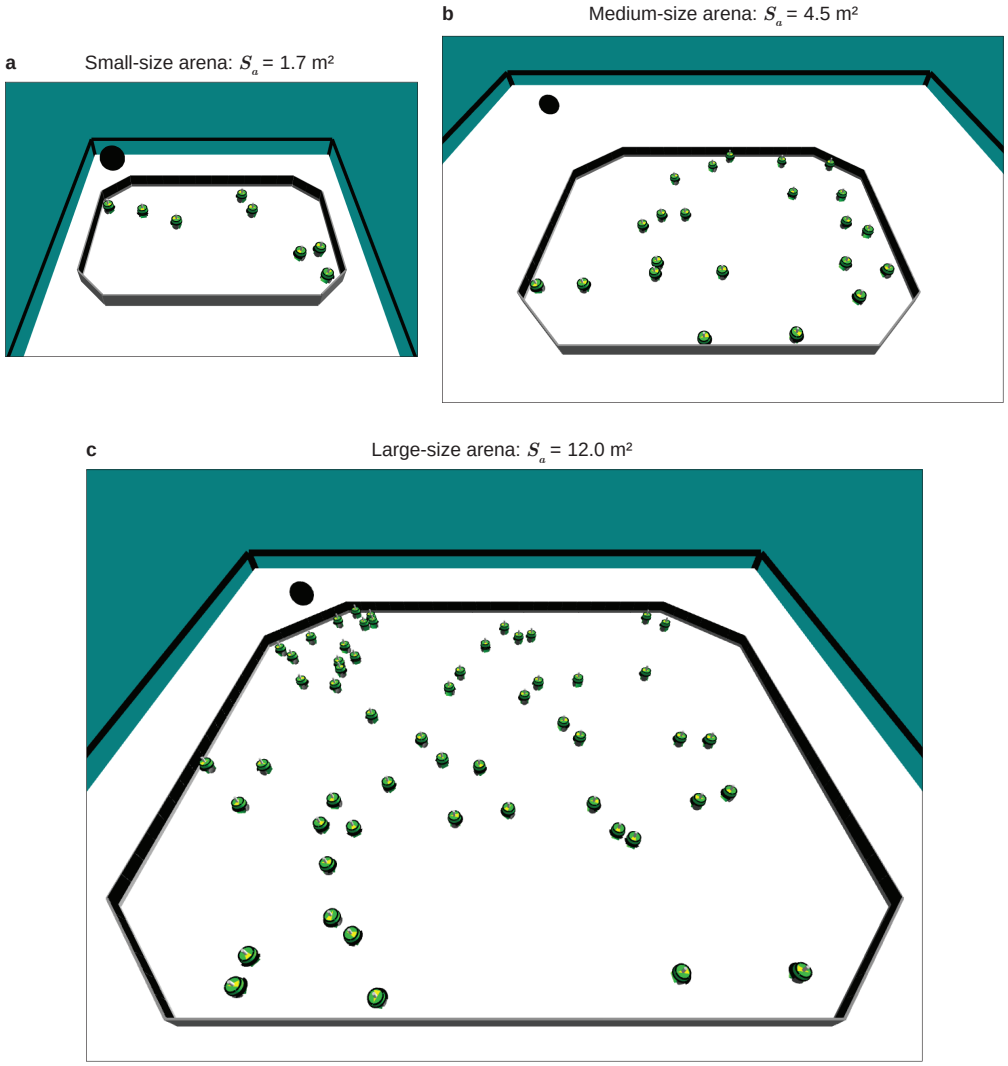


Figure 4.14: **Simulated arenas for scalability experiments.** (**a**) Small-size arena. (**b**) Medium-size arena. (**c**) Large-size arena.

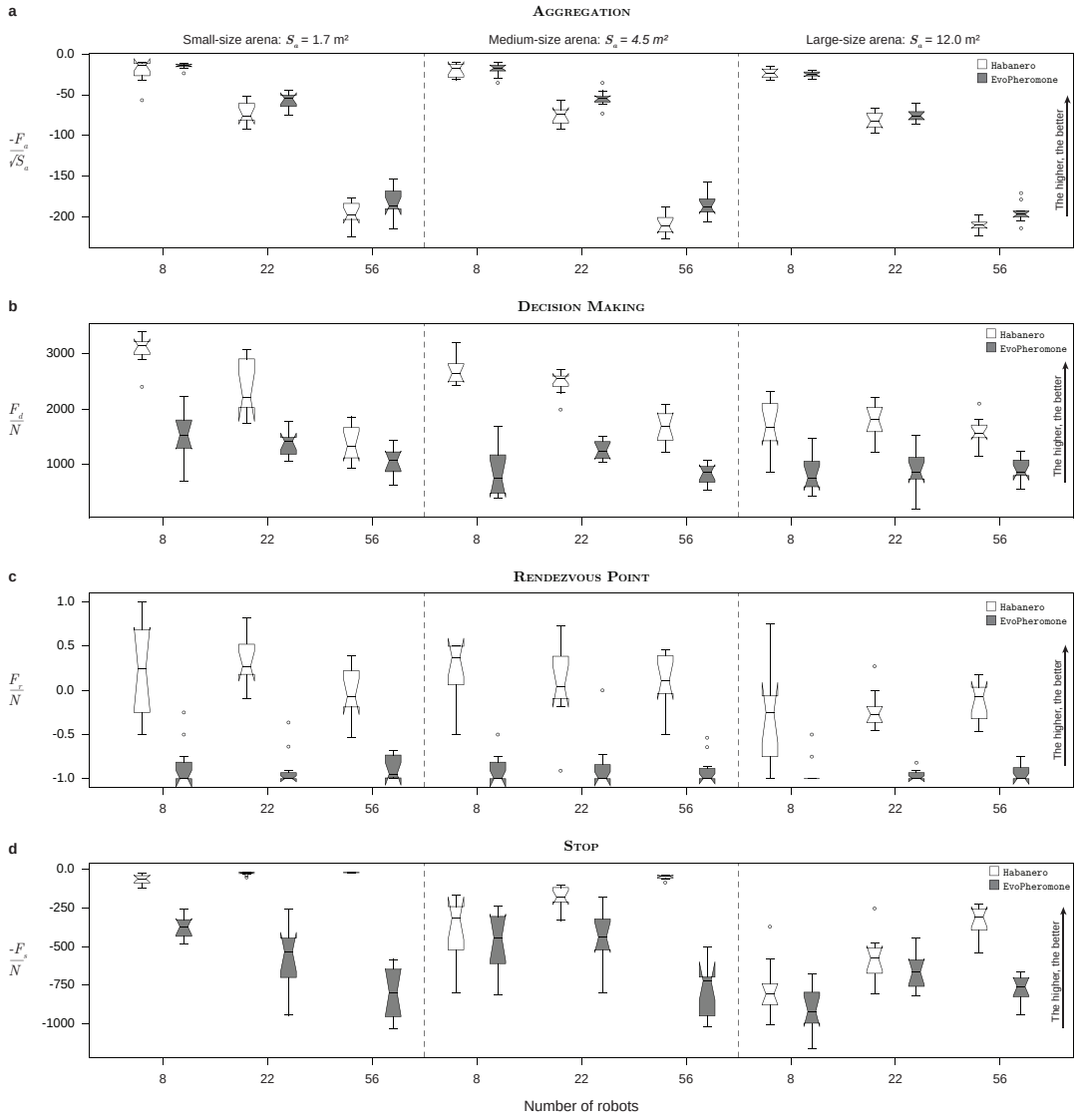


Figure 4.15: **Performance achieved by robot swarms in different scalability scenarios.** The results are presented using boxplots on a per-mission basis: (**a**) Aggregation, (**b**) Decision Making, (**c**) Rendezvous Point, and (**d**) Stop. The Aggregation performance is scaled by the square root of the size of the arena. The performance achieved in Decision Making, Rendezvous Point, and Stop are scaled by the number of robots in a swarm N.

Chapter 5

Conclusions

Automating the production of control software for pheromone-based robot swarms is a step further towards their real-world application. Automatic design can ease the realization of robot swarms across different missions, while minimizing human intervention (Birattari et al. 2019, 2020; Dorigo et al. 2020, 2021). The experiments presented in this thesis show that this holds true also in the case of robot swarms that rely on pheromone-based stigmergy. Indeed, `Habanero` automatically designed stigmergy-based collective behaviors that were effective across all missions considered. For each mission, it found appropriate ways to use the pheromone effectively. Although the software modules on which `Habanero` operates were conceived in a mission-agnostic way, the interaction strategies that `Habanero` eventually generated for each mission were automatically tailored to each of them and are different from one another. In these interaction strategies, the limited perception and computation capabilities of the individual robots are compensated at the swarm level by exploiting pheromone-based stigmergy. The e-puck used in the experiments, as a single robot, has limited spatial coordination, memory, and communication abilities. However, spatial organization, external memory, and communication in the swarm emerged at the collective level thanks to pheromone-based stigmergy. Spatial organization: In Aggregation, Decision Making, and Rendezvous Point, the e-pucks self-organized and distributed in space guided by their pheromone trails and other environmental cues. Memory: In Decision Making and Rendezvous Point, the swarm of e-pucks retained relevant information about the past state of the environment by laying pheromone trails. Communication: The semantics of pheromone trails is mission-specific. For example, the pheromone trails that the e-pucks laid in Stop had a meaning (stop where you are) that is radically different from the meaning in Aggregation

(come here). It is interesting to note that spatial organization, memory, and communication (including the semantics of pheromone trails) were not hand-coded in the modules on which `Habanero` operates: they were the product of the way in which `Habanero` automatically combined these modules on a per-mission basis.

The thesis leaves two main questions open. (i) Can automatic design leverage the intensity of pheromone trails and their decay time? In the experiments presented, a robot either did or did not sense the pheromone, in a binary fashion. A more thorough investigation is required to determine whether an automatic method can simultaneously tune the concentration of the pheromone deployed and the concentration to which a robot should react. (ii) Can automatic design methods realize robot swarms that alternatively, or simultaneously, operate with direct and indirect communication? It had been shown in the past that direct communication can emerge from an automatic design process (Garzón Ramos and Birattari 2020; Hasselmann and Birattari 2020). In this thesis, I have shown that indirect communication can emerge as well. Further research is required to determine whether an automatic method can select direct or indirect communication as more suitable for a specific mission. In this sense, I deem particularly interesting the idea of automatically designing collective behaviors in which the robots operate with combinations of the two.

In this thesis, I adopted an existing technology to enable pheromone-based stigmergy with real robots—the photochromic artificial pheromone system (Salman et al. 2020b). Although viable, it is a technology that—like all the existing solutions—has some critical limitations: namely, it is only suitable for indoor applications in which the environment can be prepared beforehand with the photochromic material. As of today, no technology exists to provide robots with a universally applicable capability to mark their environment with indication of their activities. However, by analyzing the strengths of the available solutions, I can outline desirable properties for such technology. First, pheromones should be produced by robots, minimizing the need for environment preparation and/or external infrastructure. Additionally, robots should have the ability to modulate the intensity of the pheromones they lay and respond to, enabling precise control over their behavior. I also envision that pheromone-based stigmergy should facilitate the design of more complex behaviors, possibly by functioning over diverse types of pheromones that communicate different information. The devices that lay and sense pheromones should be easy to build and integrate in modern robot platforms at different scales—from small educational robots to larger platforms. Finally, the pheromone laid by the robots must be safe and nondestructive, and any marks left by the robots should disappear once the

swarm completes its operation. Engineering solutions that meet these properties would facilitate their broad adoption, development, and validation, as well as the the establishment of benchmarks for robotics stigmergy.

With `Habanero` I demonstrated that it is possible to generate pheromone-based collective behaviors through an automatic process that is repeatable and generally applicable. I contend that this result can motivate further research to overcome the limitations of the currently available hardware solutions to implement pheromone-based stigmergy.

Appendix A

AutoMoDe-Waffel

In addition to the core contributions of this dissertation, I have also engaged in other research activities. Appendix A details a study where I investigated the concurrent automatic design of control software and hardware configuration for robot swarms. In this work, I introduced Waffle, a novel addition to the AutoMoDe family of automatic design methods. Waffle configures robot hardware, determines swarm size, and generates control software as a probabilistic finite state machine by combining pre-existing, mission-independent modules.

A.1 Concurrent design of control software and configuration of hardware for robot swarms under economic constraints

Designing a robot swarm is challenging due to its self-organized and distributed nature: complex relations exist between the behavior of the individual robots and the collective behavior that results from their interactions. In this appendix, I study the concurrent automatic design of control software and the automatic configuration of the hardware of robot swarms. I introduce `Waffel`, a new instance of the AutoMoDe family of automatic design methods that produces control software in the form of a probabilistic finite state machine, configures the robot hardware, and selects the number of robots in the swarm. I test `Waffel` under economic constraints on the total monetary budget available and on the battery capacity of each individual robot comprised in the swarm. Experimental results obtained via realistic computer-based simulation on three collective missions indicate that different missions require different hardware and software configuration, and

that Waffel is able to produce effective and meaningful solutions under all the experimental conditions considered.

A.2 Introduction

In this appendix, I make a two fold contribution: (i) I address the concurrent automatic design of control software and the automatic configuration of the hardware; and (ii) I study an automatic design problem that is subject to economic constraints.

In swarm robotics Şahin 2005, a group of robots coordinate to accomplish missions that a single robot cannot accomplish. In a swarm, robots do not have predefined roles, neither do they possess any global information, nor do they rely on external infrastructures (Dorigo et al. 2014). They operate in a decentralized and self-organized manner, relying only on local information gathered through their sensors or communicated by their neighboring peers.

A robot swarm is a collective entity and cannot be programmed directly: the designer must program the individual robots so that, together, they perform the desired mission. The design process is laborious due to the complex relation existing between the behavior of the individual robots and the collective behavior that results from their interactions (Brambilla et al. 2013). The most common approach to designing a robot swarm is trial-and-error: a time consuming approach in which individual-level behaviors are implemented, tested, and modified until the desired swarm-level behavior is obtained. Although a number of swarms have been successfully designed with this approach, it heavily depends on the experience of designer, it is error-prone, and its results are not reproducible. To overcome these issues, a number of principled manual design methods have been proposed. However, these methods are limited in scope: a universal swarm design methodology does not exist, yet (Brambilla et al. 2014; Hamann and Wörn 2008; Lopes et al. 2016).

Automatic design is an alternative approach to designing a swarm. In automatic design, the design problem is formulated as an optimization problem that is then solved with an optimization algorithm (Birattari et al. 2019). Within this automatic design framework, a collective mission is expressed as an objective function, a mathematical equation that measures the performance of the robot swarm. An optimization algorithm steers the search for a control software of an individual-robot that maximizes the performance of the swarm, taking into account the constraints

such as hardware limitations of the robots or other environmental restrictions, that are encoded in the form of additional (in)equalities.

Neuroevolutionary robotics is the most studied among the existing automatic design approaches to design a swarm (Trianni 2008). This approach uses an evolutionary algorithm to optimize the control software of robots that, in this case, is represented by an artificial neural network (Nolfi and Floreano 2000). Recently, a number of automatic design approaches have been proposed that use different control software structures and optimization algorithms than those typically adopted in evolutionary swarm robotics (Francesca et al. 2014b).

The concurrent development of control software and hardware is a further step to reduce the human involvement in the design process. A number of concurrent design methods have been proposed for single-robot applications: in addition to designing the control software, they select and configure sensors and actuators and/or the robot's morphology (Lipson and Pollack 2000; Sims 1994). These concurrent design methods have significantly increased the performance and versatility of the designed robots. In swarm robotics, only a few research articles have been published that studied the concurrent automatic design of control software and configuration of the hardware (Watson and Nitschke 2015). Details are provided in the Section A.3.

In general, designing implies solving trade-offs, that is, balancing multiple, possibly conflicting factors (Pahl et al. 2007). In swarm robotics, a characterizing example of a trade-off to be handled is the one between the number of robots comprised in the swarm and the capabilities of each individual robot. The designer must decide whether, for the specific mission at hand, they should use (i) few highly capable robots or (ii) many relatively incapable ones. This trade-off originates from the constraint of a limited monetary budget.

Indeed, it is reasonable to assume that highly capable robots are more expensive than relatively incapable ones. Another example of a design trade-off originates if the designer is given the constraint of adopting a battery of a predefined capacity. Under this constraint, the designer might chose to adopt (i) robots with powerful sensors and actuators that can achieve relatively more per unit time, but have a high power consumption and therefore can operate for a relatively short amount of time; or (ii) simpler robots that can achieve relatively less per unit time but have a low power consumption and therefore can operate for a relatively long amount of time. It is reasonable to expect that the choice might depend on the specific mission at hand.

In this appendix, I introduce `Waffel`, a new instance of the AutoMoDe family of automatic design methods. All previously published instances of Auto-

MoDe generate control software for the e-puck platform (Mondada et al. 2009) by selecting, combining, and fine-tuning predefined, mission-independent software modules (Francesca et al. 2015, 2014b; Hasselmann et al. 2018b; Kuckling et al. 2018). `Waffel` is based on `Chocolate` (Francesca et al. 2015). Indeed, regarding the conception of control software, `Waffel` is identical to `Chocolate`: the two methods share the same predefined software modules, they combine these modules into the same control software architecture, and they use the same optimization algorithm—details are given in the Section A.4.

The novelty of `Waffel` is the concurrent hardware configuration of the robot swarm: `Waffel` automatically selects the hardware configuration of the individual robots and the number of robots within the swarm. The goal of this study is to show that it is possible to concurrently design the control software and configure the hardware for robot swarm using the principles of automatic modular design: the idea that control software and, in my particular case the hardware, can be produced by combining pre-existing modules that are mission independent and that are assembled and fine tuned automatically.

In this specific study, I consider some hypothetical hardware modules that enable a robot to detect and locate its neighboring peers. These hypothetical modules are based on infrared transceivers and are variants of an existing hardware module for the e-puck platform (Mondada et al. 2009) known as the range-&-bearing (Gutiérrez et al. 2009). I define the set of these hypothetical modules so that some of them are more-capable and some are less-capable than the existing one in terms of perception range and detection abilities. I assume that the more capable hardware modules are more expensive and consume more power.

These hypothetical modules are realistic and possibly implementable. The fact that they are hypothetical (except one) does not impair the significance of my research. Indeed, my goal is not to solve a specific design problem but rather to show that a modular approach to designing by optimization can search the space of possible hardware configurations concurrently with the automatic design of control software.

I study `Waffel` under what I shall collectively call *economic constraints*, namely, constraints on the total monetary budget available and on the battery capacity of each individual robot comprised in the swarm. If these constraints were not included, the study would produce trivial results in many cases. Indeed, in many cases, the automatic design process would produce swarms comprising the largest number of robots possible, each equipped with the best performing, most expensive, and most energy-consuming hardware modules.

Besides preventing that the study produces trivial results, these constraints have a value on their own. Indeed, in a prospective practical application of automatic design, one will be necessarily faced with economic constraints, which are an essential, unavoidable element of any real-world design problem.

To the best of my knowledge, this study is the first one in which automatic design of robot swarms is studied under constraints of economical nature. In this sense, my work contributes to moving a step in the direction of the practical application of automatic design.

The main research question that I address in this appendix is the following: can `Waffel` select mission-specific hardware together with an appropriate control software? To do so, I test `Waffel` on three different collective missions: End-Time-Aggregation, Anytime-Selection, and Foraging. For each mission, I impose constraints to the design process. Namely, I impose a monetary budget and/or a battery capacity. For each mission, I perform nine different experiments: (i) one experiment in which both monetary budget and battery capacity are unconstrained (*No-Constraint*), (ii) two experiments with different levels of the monetary budget and unconstrained battery capacity (*Monetary-Constraint*), (iii) two experiments with different levels of battery capacity and unconstrained monetary budget (*Power-Constraint*), and (iv) four experiments with different levels of monetary budget and battery capacity (*Monetary-&-Power-Constraint*). For each experiment, I report and discuss (i) a measure of the performance achieved, (ii) the number of robots comprised in the automatically designed swarm, (iii) which hardware modules have been automatically selected, and (iv) which software modules were adopted.

A.3 State of the art

In the literature, a number of approaches have been proposed to address the concurrent design of single robots. However, only a few preliminary studies have been published that implement the simultaneous design of hardware and control software for a robot swarm.

In single robot applications, Sims 1994 introduced what they called *virtual creatures*: simulated robots whose body and control software are designed simultaneously to perform different tasks, such as walking, jumping, and swimming. The body of these robots is composed of solid cuboid segments connected by different joint types, actuators to simulate muscle force at joints, and various sensors. The body of a robot is represented as a directed-graph of nodes and connections that

contain the connectivity information and developmental instructions. The control software of the robot is implemented as an artificial neural network. A genetic algorithm was used to concurrently design the software and the hardware of a robot for a particular task. The development of virtual creatures demonstrated the ability of this approach to design complex systems that would be complicated to design using traditional methods.

Lipson and Pollack 2000 took this concept to a further level by introducing the automatic manufacturing of the concurrently designed robot. The authors used the rapid prototyping technology to 3D print the robot once its body (variable-length bars, and ball-and-socket joints) and control software (artificial neural network) is automatically designed in the simulation. In recent studies, much work has been conducted using similar approaches that aim to address various design problems, e.g., robots with insect-like hardware topologies and behaviors (Hornby et al. 2003); visually guided robots (Macinnes 2003); aquatic robots (Clark et al. 2012); self-reconfiguring robot (Nygaard et al. 2018).

In swarm robotics, only a couple of studies are available that use concurrent design methods to design a robot swarm. Watson and Nitschke 2015 studied the impact of the number of sensors and their position on the robot to select the minimal sensor configuration of individual robot for a collective construction task. They achieved that by manually selecting six different sensors configurations and generating six instances of control software in the form of artificial neural networks using HyperNeat.

Hewland and Nitschke 2015 used NEAT-M to configure the number and types of sensors simultaneously with the control software for the robots in a swarm for collective construction task. Moreover, they also designed the control software for a robot swarm with fixed hardware configuration. According to the authors, the concurrently designed swarm performed relatively better than the swarm with fixed hardware configuration.

Heinerman et al. 2015 studied the relationship between individual and social learning in physical robot swarms. The authors used six Thymio II robots in their experiments. The study shows that the on-line social learning in a physical robot swarm is possible, the design process is faster than individual learning, and the performance of the produced control software (artificial neural networks) is higher. Moreover, the design process also configures a suitable sensory layout for individual robots.

Various computational models have been proposed to estimate the impact of the size/density of the robot swarm on its performance (Hamann 2012; Lerman

and Galstyan 2002). However, I am not aware of any study in which the automatic selection of the number of robots for a swarm has been attempted. To the best of my knowledge, the implications of imposing economical constraints to the automatic design of a robot swarm have never been studied. I am only aware of a single study that goes into that direction: recently, Carlone and Pinciroli 2019 included some practical constraints in the design of a robot swarm. They formulate the co-design of a single race-drone and multi-drone system as a binary optimization problem that allows specifying constraints such as the total design budget.

A.4 AutoMoDe-Waffel

As already mentioned above, `Waffel` belongs to AutoMoDe, a family of off-line automatic methods for designing the control software of robot swarms (Francesca et al. 2014b). In AutoMoDe, control software is generated by automatically assembling predefined modules and by fine-tuning their free parameters. A number of methods have been proposed that belong to AutoMoDe: `Vanilla` (Francesca et al. 2014b), `Chocolate` (Francesca et al. 2015), `Gianduja` (Hasselmann et al. 2018b), `Maple` (Kuckling et al. 2018), `Arlequin` (Ligot et al. 2020a), and `Nata` (Hasselmann et al. 2023). Each of these methods is characterized by a specific set of predefined modules, a software architecture into which these modules can be combined, and an optimization algorithm that searches the space of the possible ways in which modules can be combined into the given architecture and the space of the free parameters. All the aforementioned methods generate control software for a specific version of the e-puck platform (Mondada et al. 2009). Moreover, they all limit themselves to the generation of control software: the hardware configuration of the e-puck robot is fixed and the number of robots comprised in the swarm is given as a requirement of the mission to be performed.

`Waffel` is a further step to increase the flexibility of AutoMoDe and to reduce the human involvement in the design process. Indeed, `Waffel` concurrently develops the control software and configures the hardware of the robot swarm—including the number of robots comprised. Regarding the design of control software, `Waffel` is identical to `Chocolate` (Francesca et al. 2015): the two methods share the same set of pre-defined software modules; generate control software in the form of probabilistic finite state machines; and use the irace optimization algorithm (López-Ibáñez et al. 2016) to select, combine, and fine-tune the software modules. The set of software modules is composed of six low-level behaviors and six conditions (Francesca

et al. 2015). A behavior is an operation or action that a robot can perform, while a condition is a criterion to switch from one behavior to another. Behaviors and conditions have parameters that impact their internal functioning: AutoMoDe fine-tunes these parameters to the specific mission to be performed. Multiple instances of the same behavior might coexist in a probabilistic finite state machine, possibly with different values of the parameters. In `Waffel` (as in `Chocolate`), states and edges of a probabilistic finite state machine are instances of behaviors and conditions, respectively. The design process can include a maximum of four states and each state can have at most four outgoing edges. A brief description of the software modules is given in Table A.1 and a typical probabilistic finite state machine is shown in Fig. A.1. Regarding the hardware, `Waffel` uses irace to define the configuration of the individual e-puck robots and their number within the swarm.

Table A.1: Low-level behaviors and conditions used in `Waffel`.

Low-level Behaviors	
Exploration	The robot moves straight. If an obstacle is detected, the robot rotates in place for a random amount of time before moving straight again
Stop	The robot does not move
Phototaxis	The robot moves towards the light, if perceived; otherwise, it moves straight
Anti-phototaxis	The robot moves away from the light, if perceived; otherwise, it moves straight
Attraction*	The robot moves towards peers within its perception range
Repulsion*	The robot moves away from peers within its perception range
Conditions	
Black-floor	Black floor is detected
Gray-floor	Gray floor is detected
White-floor	White floor is detected
Neighbor-count*	The number of peers in neighborhood is greater than a parameter
Inverted-neighbor-count*	The number of peers in neighborhood is less than a parameter
Fixed-probability	The transition is enabled with a fixed probability

*Behaviors and conditions that use the range-&-bearing module

The e-puck is a differential drive robot that is widely used in swarm robotics research (Mondada et al. 2009). `Waffel` and all previous instances of AutoMoDe

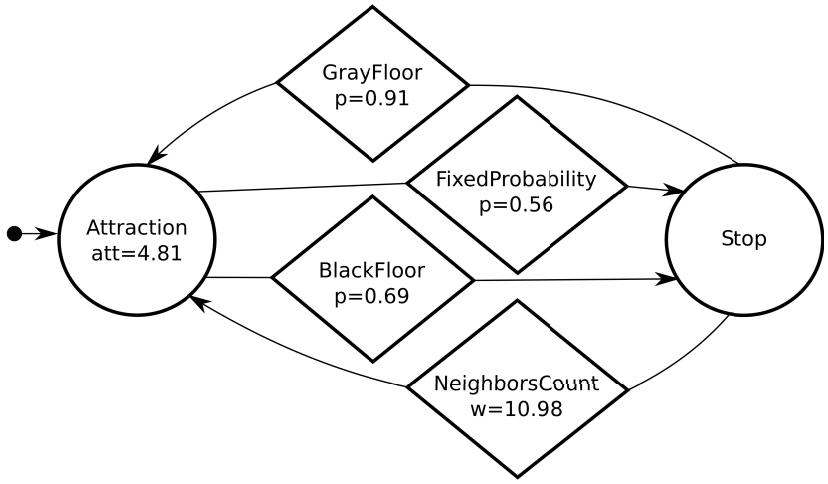


Figure A.1: A typical probabilistic finite state machine automatically designed by `Waffel`: states and conditions are represented by circles and diamonds, respectively. Initially the robot moves towards its neighboring peers (attraction state)—the robot follows a direction vector and $att = 4.81$ is the attraction parameter that defines the magnitude of the vector. When it detects the black floor, it stops. The parameter p is the probability of transition from one state to another when the condition is true. I refer the reader to Francesca et al. 2014b for further details.

operate with an extended version of the e-puck robot, which adopts: (i) the Overo Gumstix, to run Linux on the robot, (ii) three ground sensors, located under its body, to detect the gray-level color of the floor beneath it, and (iii) a range-&-bearing module (Gutiérrez et al. 2009) to detect neighboring peers and have knowledge of their relative position. I simulate the e-puck robots using ARGoS (Garattoni et al. 2015; Pinciroli et al. 2012), an open source multi-engine simulator for robot swarm. I use ARGoS' 2D dynamic physics engine to simulate the robots and the environment.

Here, I assume that e-puck robots are formally described by reference model RM 1.1 (Hasselmann et al. 2018a), which defines the input and output variables of corresponding sensors and actuators—see Table A.2. These variables can be read/written by the control software at every control step, that is, every 100 ms. The control software detects the obstacles ($prox_i$) and the presence and relative position of a light source ($light_i$) using eight infrared transceivers. It also detects the gray-level color of the floor ($ground_j$) beneath the robot using ground sensors. At every control step, all robots in the swarm broadcast a "heartbeat" signal using their range-&-bearing module. This signal encodes the sender's unique ID. Every robot receives the heartbeat signals of the peers that are in its perception range and has therefore knowledge of their number (n), and of their aggregate relative

position (V) which is defined as:

$$V = \begin{cases} \sum_{m=1}^{n} \left(\frac{1}{1+r_m}, \angle b_m \right), & \text{if robots are perceived;} \\ (1, \angle 0), & \text{otherwise.} \end{cases} \quad \text{(A.1)}$$

Here, r_m and b_m are the range and bearing of the m^{th} neighboring peer, respectively. For a detailed description of the vector V and of how it is computed, see (Salman et al. 2019b). Eventually, the control software actuates the wheels of the robot by setting the right and left wheel velocity (v_r and v_l).

Table A.2: Reference model RM 1.1.

Sensor	**Input**	**Value**	**Description**
Proximity	$prox_{i \in \{1,\ldots,8\}}$	$[0, 1]$	reading of proximity sensor i
Light	$light_{i \in \{1,\ldots,8\}}$	$[0, 1]$	reading of light sensor i
Ground	$ground_{j \in \{1,2,3\}}$	$\{black, gray, white\}$	reading of ground sensor j
Range-&-Bearing	n	$[0, 29]$	number of neighboring robots perceived
	V	$([0.5, 30], [0, 2\pi])$	their aggregate position

Actuator	**Output**	**Value**	**Description**
Motors	$v_{k \in \{l,r\}}$	$[-0.12, 0.12]$ ms^{-1}	target linear wheel velocity

As mentioned above, the goal of this appendix is to concurrently develop the control software and configure the hardware for the robot swarm. Concerning the hardware configuration, `Waffel` configures the range-&-bearing transceiver modules of e-puck robots. To do so, I simulate six range-&-bearing receivers and two range-&-bearing transmitters as listed in Table A.3. These range-&-bearing modules are hypothetical but are variants of one that actually exists (Gutiérrez et al. 2009): receiver R^3_{rb} coupled with transmitter T^1_{rb}, as defined in Table A.3. Each hypothetical range-&-bearing receiver and transmitter has distinct characteristics. A receiver is characterized by an error modeled as white noise in the estimation of the angle of a broadcasting peer (bearing error), and a probability to fail to receive the signal broadcast by a robot in its perception range (loss probability). The bearing error is sampled at every time step from a uniform distribution. The loss probability is a function of the number of neighboring peers—details are given as supplementary material (Salman et al. 2019b). A range-&-bearing transmitter is characterized by a tunable infra-red transmission range ($\mathcal{R}$)—see Table A.3. If

the design process finds the range-&-bearing necessary for a mission, it can equip all the robots with one of the receiver and of the transmitter configurations listed in Table A.3. In configuring the hardware of the robot swarm, the design process must also respect the available monetary budget and/or a battery capacity. Indeed, the range-&-bearing receivers and transmitters are also characterized by price and current rating—see Table A.3.

Table A.3: Extended range-&-bearing receiver and transmitter modules. The bearing error is modeled as white noise in the estimation of the bearing of a broadcasting peer and is sampled from a uniform probability distribution, of which I list here the extremes of the support. The loss probability is a function of the number of neighboring peers, of which I list here the minimum, average, and maximum values.

Range-&-Bearing Receivers R^x_{rb}	**Bearing Error** ($\pm$ deg)	**Loss Probability** $min - avg - max$	**Price** P_x (€)	**Current Rating** I_x (mA)
$\emptyset$	–	–	0	0
R^1_{rb}	45	$0.75 - 0.84 - 0.95$	500	10
R^2_{rb}	30	$0.75 - 0.85 - 0.90$	600	15
R^3_{rb}	25	$0.75 - 0.80 - 0.93$	700	20
R^4_{rb}	20	$0.70 - 0.78 - 0.85$	800	25
R^5_{rb}	15	$0.50 - 0.64 - 0.75$	900	30
R^6_{rb}	5	$0.40 - 0.57 - 0.70$	1000	35

Range-&-Bearing Transmitters T^y_{rb}	**Range** $\mathcal{R}$ (m)	**Price** P_y (€)	**Current Rating** $I_{y(R)}$ (mA)
$\emptyset$	–	0	0
T^1_{rb}	$\{0.6, 0.7, 0.8\}$	400	{20, 30, 40}
T^2_{rb}	$\{0.9, 1.0\}$	600	{50, 60}

A.5 Experimental setup

In this section, I describe the three collective missions, the experiments I perform for each mission, and the protocol I follow to test `Waffel`.

A.5.1 Missions

I test `Waffel` on three missions: Anytime-Selection, Aggregation, and Foraging. All three missions are to be performed in a dodecagonal arena of

4.91 m^2. The arena is divided into different zones according to the requirements of a mission. ANYTIME-SELECTION and AGGREGATION are performed in the same arena—as shown in Fig. A.2 (a). At the beginning of every experimental run, I randomly position the robots everywhere in the arena.

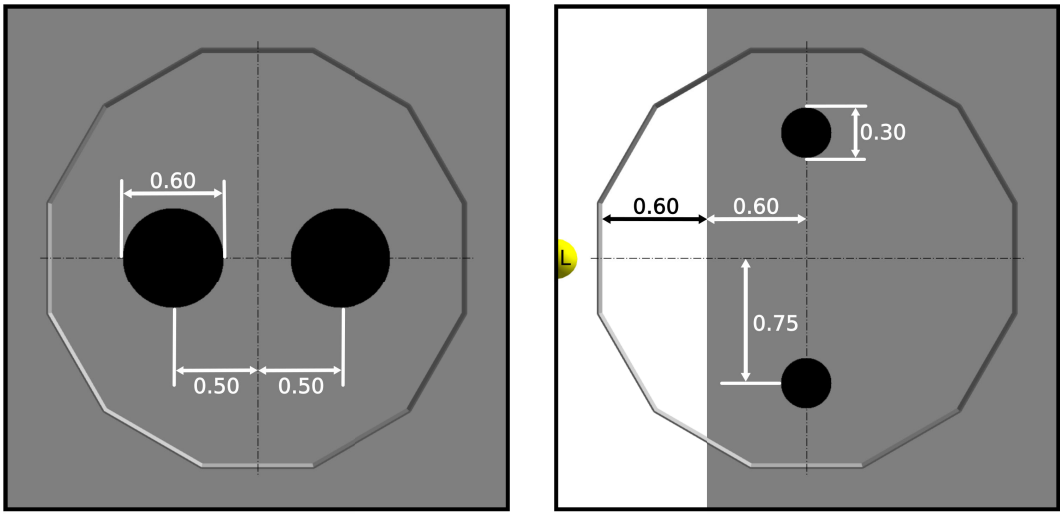


Figure A.2: ARGoS representation of arenas with dimensions and positions of different zones: A.2 (a) AGGREGATION and ANYTIME-SELECTION, and A.2 (b) FORAGING. Measurements are expressed in meters. In FORAGING, L represents a light source.

Anytime-Selection. The robot swarm must aggregate in one of the two circular black zones. The size of two black zones and their position in the arena are given in Fig. A.2 (a). The performance of the swarm is measured by the following objective function:

$$F_A = \sum_{t=1}^{T} \Big| \left(N_a\left(t\right) - N_b\left(t\right)\right) \Big|, \tag{A.2}$$

where $N_a(t)$ and $N_b(t)$ are the number of robots in zone a and b at any time t; T is the total duration of the experiment.

In ANYTIME-SELECTION, the performance is measured at every control step. Due to this reason, the robots are expected to promptly aggregate in one of the black zones and stay there until the end of the experiment.

Aggregation. The robots must aggregate in one of the two black zones. The dimensions of two black zones and their position in the arena are given in Fig. A.2 (a).

The performance of the robot swarm is measured with the following objective function:

$$F_E = \Big| N_a\left(T\right) - N_b\left(T\right) \Big|, \tag{A.3}$$

where T is the duration of an experiment and $N_a(T)$ and $N_b(T)$ are the number of robots in zone a and zone b at time T.

Unlike ANYTIME-SELECTION, the performance in AGGREGATION is computed at the end of the experiment. Due to this reason, the robots can take some time to explore the arena and converge in a black zone: the experiment duration is not a significant constraint in this mission. However, the real challenge is to keep the robots assembled in a zone until the end of the experiment.

Foraging. The swarm must collect a maximum number of objects from two sources and drop them in the nest. I abstract the FORAGING experiment by considering that an object is retrieved when an individual robot visits a source, and an object is dropped when a robot visits the nest. The two sources in the arena are represented as two black zones, while the nest is represented as a white zone. A light is also placed behind the nest as a cue for robots. The dimensions and position of the two source zones and nest are given in Fig. A.2 (*b*). The performance of the robot swam, the number of objects (N_o) retrieved by the swarm, is expressed by the following objective function:

$$F_F = N_o. \tag{A.4}$$

In FORAGING, an individual robot can retrieve a single object at a time. Therefore, the performance of the swarm heavily rely on the number of robots and on the duration of the experiment.

A.5.2 Experiments

I perform nine different experiments for each mission. In these experiments, I impose a monetary budget and/or a battery capacity constraints to the design process. Depending on the type of constraint, an experiment can be classified as belonging to one of four categories: *No-Constraint*, *Monetary-Constraint*, *Power-Constraint*, and *Monetary-&-Power-Constraint*. The levels of the monetary constraint, levels of battery capacity, and duration of the experiments are listed in Table A.4. For each experiment, the design process is free to choose any number of robots between 15 and 30.

No-Constraint

This experiment (NC) is performed without any constraint: the monetary resources and battery capacity are unconstrained.

Monetary-Constraint

In these experiments, the limited resource is the monetary budget, M_{limit}, available to purchase the robots and range-&-bearing modules. The design process only considers the combinations of hardware modules that keep the total cost of the swarm, P_{swarm}, equal or below the available monetary budget—i.e, $P_{swarm} \leq M_{limit}$. The total swarm cost, P_{swarm}, is computed with the following equation:

$$P_{swarm} = N \times (P_r + P_x + P_y)\,, \tag{A.5}$$

here N is the total number of robots in swarm, that is, 15 to 30 robots; P_r is the price of extended version of e-puck without range-&-bearing modules, that is, 2000 €; P_x and P_y are the prices of a range-&-bearing receiver and a range-&-bearing transmitter respectively—see Table A.3.

The minimum cost of a swarm is 43 500 €, when the minimum number of 15 robots are equipped with the least-capable range-&-bearing receiver and transmitter modules. The maximum cost of a swarm is 108 000 €, when the maximum number of 30 robots are equipped with the most-capable range-&-bearing receiver and transmitter modules.

For each mission, I perform two experiments, M_{80} and M_{60}, where the monetary budget is 80 000 € and 60 000 € respectively—see Table A.4.

Power-Constraint

In these experiments, the limited resource is the battery capacity, P_{bc}. There is no constraint on the monetary resources: the design process can choose any combination of the range-&-bearing modules and the number of robots between 15 and 30—see Table A.4. The operation time, T_r, of each robot in the swarm depends on its hardware configuration, available battery capacity, and the execution of the individual-level behaviors. The operation time of a robot can be computed by the following equation:

$$T_r = \frac{(P_{bc} \times 3600)}{(I_{cpu} + I_{lm} + I_{rm} + I_{y(\mathcal{R})} + I_x)}, \tag{A.6}$$

Table A.4: Monetary budget levels, battery capacity levels and duration of all nine experiments of four categories. The duration of an experiment, T, from categories *Power-Constraint* and *Monetary-&-Power-Constraint* is not fixed. The experiment terminates, when all robots are out of battery—as defined in Eq. A.7.

Experiment	Category	Monetary Budget	Battery Capacity	Duration
NC	*No-Constraint*	unconstrained	unconstrained	500 s
M_{80}	*Monetary Constraint*	80 000 €	unconstrained	500 s
M_{60}		60 000 €	unconstrained	500 s
P_{20}	*Power Constraint*	unconstrained	20 mAh	T
P_{15}		unconstrained	15 mAh	T
$M_{80}P_{20}$	*Monetary & Power Constraint*	80 000 €	20 mAh	T
$M_{80}P_{15}$		80 000 €	15 mAh	T
$M_{60}P_{20}$		60 000 €	20 mAh	T
$M_{60}P_{15}$		60 000 €	15 mAh	T

where I_{cpu} is the current rating of robot's cpu and other fixed sensors, that is, 100 mA. The CPU and other fixed hardware modules will always consume the same power. I_{lm} and I_{rm} are the current ratings of the left and the right motors of the robot, that is, 150 mA at maximum speed. I_x and $I_{y(\mathcal{R})}$ are the current ratings of range-&-bearing receiver and transmitter modules respectively. $\mathcal{R}$ is the range of range-&-bearing transmitter—see Table A.3. The experiment terminates, when every robot in the swarm consumes its battery power. The total experiment time, T, is expressed as:

$$T = \max\left(T_{r\in\{1,2,3,\ldots,N\}}\right). \tag{A.7}$$

For each mission, I perform two experiments with different levels of battery capacities: P_{20} and P_{15}—see Table A.4.

Monetary-&-Power-Constraint

In these experiments, both monetary budget and battery capacity are limited. The design process is required to choose the hardware modules that are not only affordable but also allow robots to operate for a sufficient amount of time. For each mission, I perform four experiments with dual constraints: $M_{80}P_{20}$, $M_{80}P_{15}$, $M_{60}P_{20}$, and $M_{60}P_{15}$—see Table A.4.

A.5.3 Protocol

The experiments are performed without any human intervention. The design of control software and the configuration of hardware is the result of an automatic design process. For each experiment, I run 20 independent design processes to get 20 hardware configurations and their respective control software in the form of a finite state machine. Each design process is run with the design budget of 50 000 simulations. The performance of the designs are evaluated via a single run of each design. For each experiment, I report (i) the performance achieved by the swarm, (ii) the number of robots comprised in the automatically designed swarm, (iii) the hardware modules that have been automatically selected, and (iv) the adopted software modules.

A.6 Results

In this section, I present the results on a per-mission basis. The instances of control software generated, the data collected, and videos of the experiments are available as online supplementary material (Salman et al. 2019b).

A.6.1 Anytime-Selection

The performance of an automatically designed swarm depends on the number of robots that reach the black zone and on the moment in which each of them does so: the longer a robot remains on the black zone, the higher the contribution it makes to the score. As a result, the duration of the experiment has an impact on the performance: the longer the experiment, the longer the robot can remain on the black zone and contribute to the score. When economical constraints are imposed, the design process tends to select low-tier hardware; and designs the control software such that the robots move less and save battery life for a longer experiment duration.

No-Constraint

`Waffel` tends to configure robot swarms whose total cost is close to the maximum possible—see Fig. A.3(d). Indeed, the robot swarms comprise 25 to 30 robots—see Fig. A.3(g)—equipped with high-tier range-&-bearing receivers and transmitters—see Fig. A.4. At visual inspection, the robots first form clusters and then slowly converge on a black zone: when the robots find a black zone, they spin in place

and block the way for the remaining robots, which are therefore unable to enter the zone. This behavior is obtained with Exploration, Stop, and Attraction—see Fig. A.5(a). As expected, the performance of the swarm is considerably better than the one achieved when constraints are imposed—see Fig. A.3(a).

Monetary-Constraint

Under the constraints imposed by M_{80} and M_{60}, `Waffel` tends to configure the robot swarm so that the total cost is close to the maximum available budget—see Fig. A.3(d). The number of robots in the swarm decreases proportionally to the monetary budget—see Fig. A.3(g). The robots are equipped with high-tier range-&-bearing receivers and long-range range-&-bearing transmitters. In M_{60}, however, `Waffel` also selects two low-tier range-&-bearing receivers—see Fig. A.4. The robot swarms designed under NC and M_{80} behave in a similar way. In M_{60}, however, the robots prefer to use the Attraction low-level behavior to remain in a black zone, but as the robots are equipped with low-tier range-&-bearing receivers, they often leave the black zone: due to the high loss-probability of low-tier range-&-bearing receivers, the robots often fail to perceive the presence of their peers in their neighborhood. The performance of the swarms designed under M_{80} and M_{60} is considerably lower than the one achieved under NC: in M_{80} and M_{60}, the swarm comprises fewer robots as compared to NC—see Fig. A.3(a).

Power-Constraint

In contrast to NC, the swarms configured under P_{20} and P_{15} have a total cost that is noticeably lower than the maximum possible—see Fig. A.3(d). This is because the robots are equipped with low-price range-&-bearing transmitters, which reduces the overall cost of the swarm—see Fig. A.3(d). Cheaper range-&-bearing transmitters have a shorter transmission range but have low power consumption, which allows a longer battery life. I observe a major shift in the dominant individual-level behaviors in the produced instances of control software. The robots stop in the first black zone they encounter and limit their movement to save energy—see Fig. A.5(a). Consequently, the swarm splits and becomes unable to gather on the same zone. As a result, the performance drops by approximately 50% as compared to the performance achieved under NC—see Fig. A.3(a).

Monetary-&-Power-Constraint

In all experiments, `Waffel` tends to use all the available monetary budget—see Fig. A.3(d). In $M_{80}P_{20}$ and $M_{80}P_{15}$, `Waffel` designs swarms that comprise 23 to 24 robots—see Fig. A.3(g)—equipped with any of the range-&-bearing receivers and low-range transmitters. In $M_{60}P_{20}$ and $M_{60}P_{15}$, however, the number of robots decrease considerably, and the robots are equipped with low-tier range-&-bearing receivers and low-range transmitters—see Fig. A.3(g) and A.4. The control software generated under the *Monetary-&-Power-Constraint* behave similarly to those of the *Power-Constraint* experiments—see Fig. A.5(a). Due to the increased battery capacity, the swarms produced under $M_{80}P_{20}$ and $M_{60}P_{20}$ perform slightly better than the ones produced under P_{15}, $M_{80}P_{15}$, and $M_{60}P_{15}$—see Fig. A.3(a).

A.6.2 Aggregation

The performance of a designed swarm depends solely on the number of robots that are on the black zone at the end of an experiment. Contrary to Anytime-Selection, if economical constraints are applied, the design process tends to select high-tier hardware; and the control software is composed of individual-level behaviors that keep robots assembled on a black zone.

No-Constraint

`Waffel` tends to configure robot swarms whose total cost is close to the maximum possible—see Fig. A.3(e). The hardware configuration is similar to the one generated under NC for Anytime-Selection. Indeed, the robot swarms comprise 28 to 30 robots—see Fig. A.3(h)—equipped with high-tier range-&-bearing receivers and transmitters—see Fig. A.4. At visual inspection, the robots first form clusters and then converge on a black zone: robots tend to remain there by spinning in place until the end of the experiment. This behavior is obtained with Exploration, Attraction, and Stop—see Fig. A.5(b). The performance of the swarm is considerably better than those achieved when constraints are imposed—see Fig. A.3(b).

Monetary-Constraint

Under the constraints imposed by M_{80} and M_{60}, `Waffel` tends to configure the robot swarm so that the total cost is close to the maximum available budget—see Fig. A.3(e). The number of robots in the swarm decreases proportionally to the monetary budget—see Fig. A.3(h). The robots are equipped with long-range

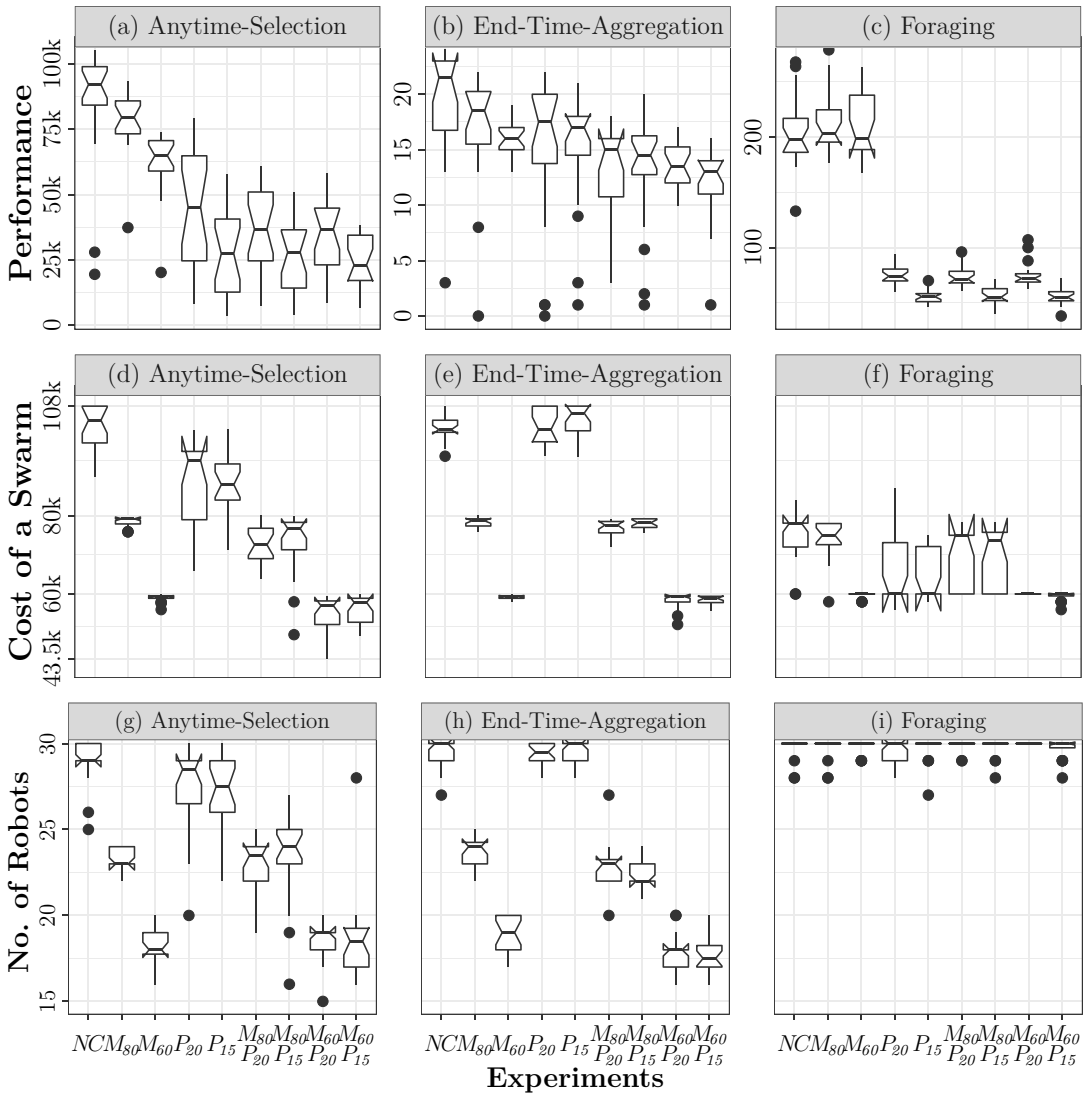


Figure A.3: The performance in all nine experiments on each mission is shown at the top. The total cost of all swarms configured in each experiment is shown in the middle: 43.5k and 108k are the minimum and maximum possible cost (in €) of a swarm, respectively. The number of robots selected by the automatic design process is shown at the bottom.

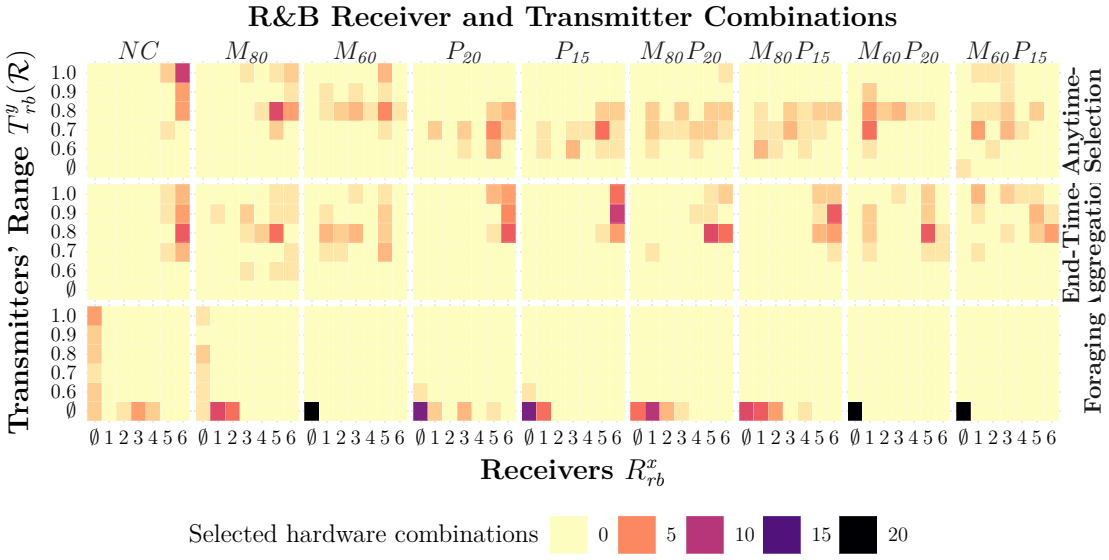


Figure A.4: The number of instances of a specific hardware combination selected in each experiment is shown here. The horizontal axis represents the possible configurations of the range-&-bearing receivers R^x_{rb}; the vertical one represents the possible ranges $\mathcal{R}$ of the range-&-bearing transmitters T^y_{rb}. Here, $\emptyset$ represents the case in which the design process does not select any range-&-bearing receiver or transmitter, $x \in \{\emptyset, 1, 2, 3, 4, 5, 6\}$, and $y \in \{\emptyset, 1, 2\}$ as shown in Table A.3.

range-&-bearing transmitters and high-tier receivers, except a small minority of configurations in which the robots are equipped with low-tier receivers—see Fig. A.4. At visual inspection, in M_{80} and M_{60} the robots converge on a black zone and stay there until the end of the experiments. Contrary to NC, the robots stop on the black zone instead of spinning in place: dominant individual-level behaviors are Exploration, Attraction, and Stop—see Fig. A.5(b). The amount of available monetary budget has a direct impact on the performance of a swarm. Indeed, due to the limited monetary budget, the number of robots decreases in the swarms designed under M_{80} and M_{60}, which results in a considerable performances drop as compared to the performance achieved under NC—see Fig. A.3(b).

Power-Constraint

Similar to NC, under the constraints imposed by P_{20} and P_{15}, `Waffel` tends to configure the robot swarm so that the total cost is close to the maximum possible—see Fig. A.3(e). Indeed, the robot swarms comprise 28 to 29 robots—see Fig. A.3(h)—equipped with high-tier range-&-bearing receivers and long-range transmitters—see Fig. A.4. However, this selection of hardware has a direct impact on the duration of the experiments due to its high current rating. As the maximum power is consumed by the motors, the designed control software skips the Exploration behavior to move robots in the arena. In some instances of control software, the robots use Phototaxis and Anti-Phototaxis individual-level behaviors to move straight and avoid obstacles. Moreover, the most dominant individual-level behavior is Attraction which is used to keep the robots assembled on one zone—see Fig. A.5(b). The performance achieved under P_{20} is relatively higher than the one achieved under P_{15}. Due to the limited battery capacity, which affects the total experiment duration, the swarms designed under P_{20} and P_{15} have a lower performance than those designed under NC—see Fig. A.3(b).

Monetary-&-Power-Constraint

In all experiments, `Waffel` tends to use all the available monetary budget—see Fig. A.3(e). In $M_{80}P_{20}$ and $M_{80}P_{15}$, `Waffel` designs swarms that comprise 22 to 24 robots—see Fig. A.3(h)—equipped with high-tier range-&-bearing receivers and long-range transmitters—see Fig. A.4. In $M_{60}P_{20}$ and $M_{60}P_{15}$, the number of robots decreases considerably, and the robots are equipped with high-tier range-&-bearing receivers and long-range transmitters, except a small minority of configurations in which robots are equipped with low-tier receivers—see Fig. A.4. The instances of

control software produced are similar to those produced under *Power-Constraint*: the movement of robots in the arena is identical and the prominent individual-level behavior is Attraction—see Fig. A.5(b). The performance achieved under $M_{80}P_{20}$ and $M_{80}P_{15}$ is slightly better than the one achieved under $M_{60}P_{20}$ and $M_{60}P_{15}$: the level of monetary budget is the key factor that determines whether `Waffel` selects few or more robots—see Fig. A.3(b).

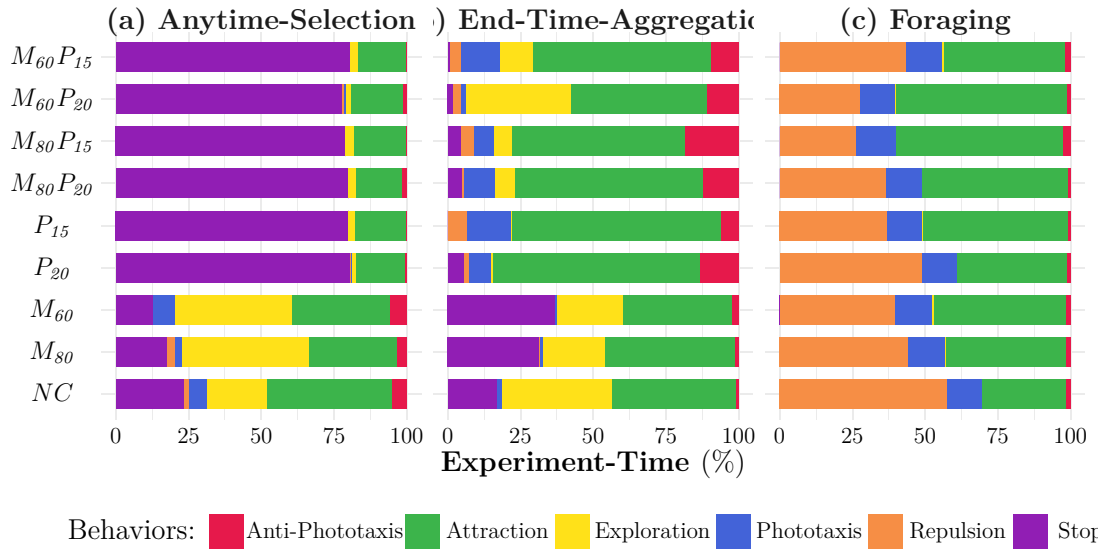


Figure A.5: The behaviors adopted by the robots in the experiments is shown here. Each color represents a behavior. The videos of all experiments are available as online supplementary material (Salman et al. 2019b).

A.6.3 Foraging

Similar to Anytime-Selection, the performance of swarms designed in the Foraging experiments depends on the experiment duration, but it also depends on the total number of robots. Contrary to both Anytime-Selection and Aggregation, the individual robots do not rely on the range-&-bearing hardware. The control software produced enables an effective movement between source and nest.

All categories of constraints

Under all the categories of constraints considered, `Waffel` produced robot swarms sharing the same hardware configuration. This because, in Foraging, the robots

do not rely on local communication. As a result, the selected hardware configuration typically does not include range-&-bearing transmitter and receiver—see Fig. A.4. The total cost of a swarm is between 60 000 € and 80 000 €—see Fig. A.3(f). The swarm comprises the largest possible number of robots—see Fig. A.3(i).

All instances of control software that are produced in all experiments have an unexpected behavior. Although in all experiments of FORAGING, the robots are not equipped with range-&-bearing modules, the most prominent individual-level behaviors are Attraction and Repulsion, which the robots use to explore the arena—see Fig. A.5(c). The swarm uses these behaviors in a way that is completely different from the one originally intended (Francesca et al. 2014b). The reason behind this anomaly is that the individual-level behaviors in the design space are not strictly associated with the related hardware. In the absence of range-&-bearing receivers and transmitters, the Attraction and Repulsion behaviors are actuating robots to move straight using proximity sensors to avoid obstacles. In all FORAGING experiments, `Waffel` selects the Phototaxis individual-level behavior to locate the nest in the arena, as shown in Fig. A.5(c).

There is no prominent performance difference between the experiments under the *No-Constraint* and *Monetary-Constraint* categories—see Fig. A.3(c). However, I observe a considerable performance drop by the swarms designed under categories of experiments that have limited battery capacity—that is, *Power-Constraint* and *Monetary-&-Power-Constraint*. Indeed, the performance achieved in experiments with 20 mA h battery capacity—i.e., P_{20}, $M_{80}P_{20}$, and $M_{60}P_{20}$—is considerably better than the performance achieved under experiments with 15 mA h battery capacity—i.e., P_{15}, $M_{80}P_{15}$, and $M_{60}P_{15}$—see Fig. A.3(c).

A.7 Conclusions

In this appendix, I studied the concurrent automatic design of control software and the automatic configuration of the hardware of robot swarms. In particular, I showed that it is possible to concurrently design control software and hardware for a robot swarm using the principles of automatic modular design. I introduced `Waffel`, a new instance of the AutoMoDe family of automatic design methods that configures the robot hardware, selects the number of robots in the swarm, and produces control software in the form of a probabilistic finite state machine by combining pre-existing modules that are mission independent. I studied `Waffel` under economic constraints on the total monetary budget available and on the

battery capacity of each individual robot comprised in the swarm. I tested `Waffel` on three different collective missions. In the experiments presented in the appendix, `Waffel` was able to concurrently design the control software and configure the hardware of a robot swarm. The results suggest that the hardware configuration of the individual robots, the design of control software, and the number of robots highly depend on the nature of the collective mission and the economical constraints imposed. In the appendix, I only considered the automatic configuration of one type of hardware module, future studies will focus on extending the automatic design to other sensors and actuators. The range-&-bearing receivers and transmitters proposed in the appendix can be manufactured and real-robot experiments can be performed to assess the robustness of the selected configuration to the reality gap.

Bibliography

Abdel-Rahman, A., Cameron, C., Jenett, B., Smith, M., and Gershenfeld, N. (2022). "Self-replicating hierarchical modular robotic swarms". In: *Communications Engineering* 1.1, p. 35.

Alfeo, A. L., Ferrer, E. C., Carrillo, Y. L., Grignard, A., Pastor, L. A., Sleeper, D. T., Cimino, M. G. C. A., Lepri, B., Vaglini, G., Larson, K., Dorigo, M., and Pentland, A. S. (2019). "Urban Swarms: A new approach for autonomous waste management". In: *2019 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, pp. 4233–4240.

Allen, J. M., Joyce, R., Millard, A. G., and Gray, I. (2020). "The Pi-puck ecosystem: hardware and software support for the e-puck and e-puck2". In: *Swarm Intelligence: 12th International Conference, ANTS 2020*. Vol. 12421. Lecture Notes in Computer Science. Springer, pp. 243–255.

Allwright, M., Bhalla, N., and Dorigo, M. (2017). "Structure and markings as stimuli for autonomous construction". In: *2017 18th International Conference on Advanced Robotics (ICAR)*. IEEE, pp. 296–302.

Allwright, M., Zhu, W., and Dorigo, M. (2019). "An open-source multi-robot construction system". In: *HardwareX* 5, e00050.

Antoun, A., Valentini, G., Hocquard, E., Wiandt, B., Trianni, V., and Dorigo, M. (2016). "Kilogrid: A modular virtualization environment for the Kilobot robot". In: *2016 IEEE/RSJ International Conference On Intelligent Robots And Systems (IROS)*. IEEE, pp. 3809–3814.

Arvin, F., Krajník, T., Turgut, A. E., and Yue, S. (2015). "COSΦ: Artificial pheromone system for robotic swarms research". In: *2015 IEEE/RSJ International Conference On Intelligent Robots And Systems (IROS)*. IEEE, pp. 407–412.

Auger, A. and Hansen, N. (2005). "A restart CMA evolution strategy with increasing population size". In: *2005 IEEE Congress on Evolutionary Computation*. Vol. 2. IEEE, pp. 1769–1776.

Balaprakash, P., Birattari, M., and Stützle, T. (2007). "Improvement strategies for the F-Race algorithm: sampling design and iterative refinement". In: *Hybrid Metaheuristics: 4th International Workshop, HM 2007*. Vol. 4771. Lecture Notes in Computer Science. Springer, pp. 108–122.

Baracchi, D., Devaud, J.-M., d'Ettorre, P., and Giurfa, M. (2017). "Pheromones modulate reward responsiveness and non-associative learning in honey bees". In: *Scientific reports* 7.1, p. 9875.

Berlinger, F., Gauci, M., and Nagpal, R. (2021). "Implicit coordination for 3D underwater collective behaviors in a fish-inspired robot swarm". In: *Science Robotics* 6.50, eabd8668.

Bianco, R. and Nolfi, S. (2004). "Toward open-ended evolutionary robotics: evolving elementary robotic units able to self-assemble and self-reproduce". In: *Connection Science* 16.4, pp. 227–248.

Birattari, M., Ligot, A., Bozhinoski, D., Brambilla, M., Francesca, G., Garattoni, L., Garzón Ramos, D., Hasselmann, K., Kegeleirs, M., Kuckling, J., Pagnozzi, F., Roli, A., Salman, M., and Stützle, T. (2019). "Automatic off-line design of robot swarms: a manifesto". In: *Frontiers in Robotics and AI* 6, p. 59.

Birattari, M., Ligot, A., and Francesca, G. (2021). "AutoMoDe: A Modular Approach to the Automatic Off-Line Design and Fine-Tuning of Control Software for Robot Swarms". In: *Automated Design of Machine Learning and Search Algorithms*. Natural Computing Series. Springer, pp. 73–90.

Birattari, M., Ligot, A., and Hasselmann, K. (2020). "Disentangling automatic and semi-automatic approaches to the optimization-based design of control software for robot swarms". In: *Nature Machine Intelligence* 2.9, pp. 494–499.

Birattari, M., Yuan, Z., Balaprakash, P., and Stützle, T. (2010). "F-Race and Iterated F-Race: an overview". In: *Experimental Methods for the Analysis of Optimization Algorithms*. Springer, pp. 311–336.

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

Brambilla, M., Brutschy, A., Dorigo, M., and Birattari, M. (2014). "Property-driven design for swarm robotics: a design method based on prescriptive modeling and model checking". In: *ACM Transactions on Autonomous Adaptive Systems* 9.4, 17:1–17:28.

Brambilla, M., Ferrante, E., Birattari, M., and Dorigo, M. (2013). "Swarm robotics: a review from the swarm engineering perspective". In: *Swarm Intelligence* 7.1, pp. 1–41.

Bredeche, N. and Fontbonne, N. (2021). "Social learning in swarm robotics". In: *Philosophical Transactions of the Royal Society of London. Series B: Biological Sciences* 377.1843, p. 20200309.

Campo, A., Gutiérrez, Á., Nouyan, S., Pinciroli, C., Longchamp, V., Garnier, S., and Dorigo, M. (2010). "Artificial pheromone for path selection by a foraging swarm of robots". In: *Biological Cybernetics* 103.5, pp. 339–352.

Carlone, L. and Pinciroli, C. (2019). "Robot co-design: beyond the monotone case". In: *2019 International Conference on Robotics and Automation (ICRA)*. IEEE, pp. 3024–3030.

Chalissery, J. M., Renyard, A., Gries, R., Hoefele, D., Alamsetti, S. K., and Gries, G. (2019). "Ants sense, and follow, trail pheromones of ant community members". In: *Insects* 10.11, p. 383.

Chambers, J. M., Cleveland, W. S., Kleiner, B., and Tukey, P. A. (1983). *Graphical Methods For Data Analysis*. CRC Press.

Chen, Y. F., Liu, M., Everett, M., and How, J. P. (2017). "Decentralized non-communicating multiagent collision avoidance with deep reinforcement learning". In: *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 285–292.

Clark, A. J., Moore, J. M., Wang, J., Tan, X., and McKinley, P. K. (2012). "Evolutionary design and experimental validation of a flexible caudal fin for robotic fish". In: *ALIFE 2012: The Thirteenth International Conference on the Synthesis and Simulation of Living Systems*. MIT Press, pp. 325–332.

Conover, W. J. (1999). *Practical Nonparametric Statistics*. Wiley Series in Probability and Statistics. John Wiley & Sons.

Corne, D. W., Reynolds, A., and Bonabeau, E. (2012). "Swarm intelligence". In: *Handbook of natural computing*. Springer, pp. 1599–1622.

Denny, A. J., Wright, J., and Grief, B. (2001). "Foraging efficiency in the wood ant, Formica rufa: is time of the essence in trail following?" In: *Animal Behaviour* 62.1, pp. 139–146.

Dorigo, M., Birattari, M., and Brambilla, M. (2014). "Swarm robotics". In: *Scholarpedia* 9.1, p. 1463.

Dorigo, M., Theraulaz, G., and Trianni, V. (2020). "Reflections on the future of swarm robotics". In: *Science Robotics* 5, eabe4385.

Dorigo, M., Theraulaz, G., and Trianni, V. (2021). "Swarm robotics: past, present, and future [point of view]". In: *Proceedings of the IEEE* 109.7, pp. 1152–1165.

Dorigo, M., Trianni, V., Şahin, E., Groß, R., Labella Thomas, H., Baldassarre, G., Nolfi, S., Deneubourg, J.-L., Mondada, F., Floreano, D., and Gambardella, L. M.

(2003). "Evolving self-organizing behaviors for a Swarm-bot". In: *Autonomous Robots* 17, pp. 223–245.

Duarte, M., Costa, V., Gomes, J., Rodrigues, T., Silva, F., Oliveira, S. M., and Christensen, A. L. (2016). "Evolution of collective behaviors for a real swarm of aquatic surface robots". In: *PLOS ONE* 11.3, e0151834.

Duarte, M., Oliveira, S. M., and Christensen, A. L. (2015). "Evolution of hybrid robotic controllers for complex tasks". In: *Journal of Intelligent & Robotic Systems* 78.3-4, pp. 463–484.

Dürr, H. and Bouas-Laurent, H. (2003). *Photochromism: Molecules and systems*. Elsevier.

Eltz, T. (2006). "Tracing pollinator footprints on natural flowers". In: *Journal of Chemical Ecology* 32, pp. 907–915.

Feinerman, O. and Korman, A. (2017). "Individual versus collective cognition in social insects". In: *Journal of Experimental Biology* 220.1, pp. 73–82.

Ferrante, E., Duéñez-Guzmán, E. A., Turgut, A. E., and Wenseleers, T. (2013). "GESwarm: grammatical evolution for the automatic synthesis of collective behaviors in swarm robotics". In: *GECCO'13: Proceedings of the 15th annual conference on Genetic and evolutionary computation*. ACM, pp. 17–24.

Floreano, D., Husbands, P., and Nolfi, S. (2008). "Evolutionary robotics". In: *Springer Handbook of Robotics*. Springer Handbooks. Springer, pp. 1423–1451.

Francesca, G. (2017). "A modular approach to the automatic design of control software for robot swarms: from a novel perspective on the reality gap to AutoMoDe". PhD thesis. Université Libre de Bruxelles.

Francesca, G. and Birattari, M. (2016). "Automatic design of robot swarms: achievements and challenges". In: *Frontiers in Robotics and AI* 3.29, pp. 1–9.

Francesca, G., Brambilla, M., Brutschy, A., Garattoni, L., Miletitch, R., Podevijn, G., Reina, A., Soleymani, T., Salvaro, M., Pinciroli, C., Mascia, F., Trianni, V., and Birattari, M. (2015). "AutoMoDe-Chocolate: automatic design of control software for robot swarms". In: *Swarm Intelligence* 9.2–3, pp. 125–152.

Francesca, G., Brambilla, M., Brutschy, A., Garattoni, L., Miletitch, R., Podevijn, G., Reina, A., Soleymani, T., Salvaro, M., Pinciroli, C., Trianni, V., and Birattari, M. (2014a). "An experiment in automatic design of robot swarms: AutoMoDe-Vanilla, EvoStick, and human experts". In: *Swarm Intelligence: 9th International Conference, ANTS 2014*. Vol. 8667. Lecture Notes in Computer Science. Springer International Publishing, pp. 25–37.

Francesca, G., Brambilla, M., Brutschy, A., Trianni, V., and Birattari, M. (2014b). "AutoMoDe: a novel approach to the automatic design of control software for robot swarms". In: *Swarm Intelligence* 8.2, pp. 89–112.

Fujisawa, R., Dobata, S., Sugawara, K., and Matsuno, F. (2014). "Designing pheromone communication in swarm robotics: Group foraging behavior mediated by chemical substance". In: *Swarm Intelligence* 8.3, pp. 227–246.

Garattoni, L. and Birattari, M. (2016). "Swarm robotics". In: *Wiley Encyclopedia of Electrical and Electronics Engineering*. John Wiley & Sons, pp. 1–19.

Garattoni, L. and Birattari, M. (2018). "Autonomous task sequencing in a robot swarm". In: *Science Robotics* 3.20, eaat0430.

Garattoni, L., Francesca, G., Brutschy, A., Pinciroli, C., and Birattari, M. (2015). *Software infrastructure for e-puck (and TAM)*. Tech. rep. TR/IRIDIA/2015-004. IRIDIA, Université Libre de Bruxelles.

Garnier, S., Combe, M., Jost, C., and Theraulaz, G. (2013). "Do ants need to estimate the geometrical properties of trail bifurcations to find an efficient route? A swarm robotics test bed". In: *PLOS Computational Biology* 9.3, pp. 1–12.

Garnier, S., Gautrais, J., Asadpour, M., Jost, C., and Theraulaz, G. (2009). "Self-Organized Aggregation Triggers Collective Decision Making in a Group of Cockroach-Like Robots". In: *Adaptive Behavior* 17.2, pp. 109–133.

Garzón Ramos, D. and Birattari, M. (2020). "Automatic design of collective behaviors for robots that can display and perceive colors". In: *Applied Sciences* 10.13, p. 4654.

Garzón Ramos, D., Salman, M., Ubeda Arriaza, K., Hasselmann, K., and Birattari, M. (2022). *MoCA: a modular RGB color arena for swarm robotics experiments*. Tech. rep. TR/IRIDIA/2022-014. IRIDIA, Université libre de Bruxelles.

Gharbi, I., Kuckling, J., Garzón Ramos, D., and Birattari, M. (2023). "Show me what you want: inverse reinforcement learning to automatically design robot swarms by demonstration". In: *2023 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, pp. 5063–5070.

Glasmachers, T., Schaul, T., Yi, S., Wierstra, D., and Schmidhuber, J. (2010). "Exponential natural evolution strategies". In: *GECCO'10: Proceedings of the 12th annual conference on Genetic and evolutionary computation*. ACM, pp. 393–400.

Gomes, J. and Christensen, A. L. (2018). "Task-agnostic evolution of diverse repertoires of swarm behaviours". In: *Swarm Intelligence: 11th International Conference, ANTS 2018*. Vol. 11172. Lecture Notes in Computer Science. Springer, pp. 225–238.

Gomes, J., Mariano, P., and Christensen, A. L. (2019). "Challenges in cooperative coevolution of physically heterogeneous robot teams". In: *Natural Computing* 18, pp. 29–46.

Goss, S., Aron, S., Deneubourg, J.-L., and Pasteels, J. (1989). "Self-organized shortcuts in the Argentine ant". In: *Naturwissenschaften* 76.12, pp. 579–581.

Goss, S., Deneubourg, J.-L., Bourgine, P., and Varela, E. (1992). "Harvesting by a group of robots". In: *Proceedings of the First European Conference on Artificial Life*. Complex adaptive systems. MIT Press, pp. 195–204.

Grassé, P. P. (1959). "La reconstruction du nid et les coordinations interindividuelles chez Bellicositermes natalensis et Cubitermes sp. la théorie de la stigmergie: Essai d'interprétation du comportement des termites constructeurs". In: *Insectes Sociaux* 6.1, pp. 41–80.

Gutiérrez, Á., Campo, A., Dorigo, M., Donate, J., Monasterio-Huelin, F., and Magdalena, L. (2009). "Open e-puck range & bearing miniaturized board for local communication in swarm robotics". In: *2009 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, pp. 3111–3116.

Hamann, H. (2012). "Towards swarm calculus: universal properties of swarm performance and collective decisions". In: *Swarm Intelligence: 8th International Conference, ANTS 2012*. Vol. 7461. Lecture Notes in Computer Science. Springer, pp. 168–179.

Hamann, H. (2018). *Swarm robotics: a formal approach*. Springer.

Hamann, H., Marc, S., and Wörn, H. (2007). "Orientation in a trail network by exploiting its geometry for swarm robotics". In: *2005 IEEE Swarm Intelligence Symposium, SIS 2007*. IEEE, pp. 310–315.

Hamann, H. and Wörn, H. (2007). "An analytical and spatial model of foraging in a swarm of robots". In: *Swarm Robotics*. Vol. 4433. LNCS. Springer, pp. 43–55.

Hamann, H. and Wörn, H. (2008). "A framework of space–time continuous models for algorithm design in swarm robotics". In: *Swarm Intelligence* 2.2–4, pp. 209–239.

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

Hasselmann, K. and Birattari, M. (2020). "Modular automatic design of collective behaviors for robots endowed with local communication capabilities". In: *PeerJ Computer Science* 6, e291.

Hasselmann, K., Ligot, A., and Birattari, M. (2023). "Automatic modular design of robots swarms based on repertoires of behaviors generated via novelty search". In: *Swarm and Evolutionary Computation* 83, p. 101395.

Hasselmann, K., Ligot, A., Francesca, G., Garzón Ramos, D., Salman, M., Kuckling, J., Mendiburu, F. J., and Birattari, M. (2018a). *Reference models for AutoMoDe*. Tech. rep. TR/IRIDIA/2018-002. IRIDIA, Université Libre de Bruxelles.

Hasselmann, K., Ligot, A., Ruddick, J., and Birattari, M. (2021). "Empirical assessment and comparison of neuro-evolutionary methods for the automatic off-line design of robot swarms". In: *Nature Communications* 12, p. 4345.

Hasselmann, K., Robert, F., and Birattari, M. (2018b). "Automatic design of communication based behaviors for robot swarms". In: *Swarm Intelligence: 11th International Conference, ANTS 2018*. Vol. 11172. Lecture Notes in Computer Science. Springer, pp. 16–29.

Hecker, J. P., Letendre, K., Stolleis, K., Washington, D., and Moses, M. E. (2012). "Formica ex machina: ant swarm foraging from physical to virtual and back again". In: *Swarm Intelligence: 8th International Conference, ANTS 2012*. Vol. 7461. Lecture Notes in Computer Science. Springer, pp. 252–259.

Heinerman, J., Rango, M., and Eiben, A. (2015). "Evolution, individual learning, and social learning in a swarm of real robots". In: *2015 IEEE Symposium Series on Computational Intelligence, SSCI 2015*. IEEE Computer Society, pp. 1055–1062.

Helbing, D., Schweitzer, F., Keltsch, J., and Molnár, P. (1997). "Active walker model for the formation of human and animal trail systems". In: *Physical Review E* 56.3, pp. 2527–2539.

Hewland, J. and Nitschke, G. (2015). "The benefits of adaptive behavior and morphology for cooperation". In: *2015 IEEE Symposium Series on Computational Intelligence, SSCI 2015*. IEEE Computer Society, pp. 1047–1054.

Heylighen, F. (2016a). "Stigmergy as a universal coordination mechanism I: Definition and components". In: *Cognitive Systems Research* 38, pp. 4–13.

Heylighen, F. (2016b). "Stigmergy as a universal coordination mechanism II: Varieties and evolution". In: *Cognitive Systems Research* 38, pp. 50–59.

Holland, O. and Melhuish, C. (1999). "Stigmergy, Self-Organization, and Sorting in Collective Robotics". In: *Artificial Life* 5.2, pp. 173–202.

Holman, L. (2018). "Queen pheromones and reproductive division of labor: a meta-analysis". In: *Behavioral Ecology* 29.6, pp. 1199–1209.

Hornby, G. S., Lipson, H., and Pollack, J. B. (2003). "Generative representations for the automated design of modular physical robots". In: *IEEE Transactions on Robotics and Automation* 19.4, pp. 703–719.

Howse, P., Stevens, J., and Jones, O. T. (2013). *Insect pheromones and their use in pest management*. Springer Dordrecht.

Hu, J., Turgut, A. E., Krajník, T., Lennox, B., and Arvin, F. (2020). "Occlusion-based coordination protocol design for autonomous robotic shepherding tasks". In: *IEEE Transactions on Cognitive and Developmental Systems*, p. 1.

Hunt, E. R., Jones, S., and Hauert, S. (2019). "Testing the limits of pheromone stigmergy in high-density robot swarms". In: *Royal Society Open Science* 6.11, p. 190225.

Hüttenrauch, M., Šošić, A., and Neumann, G. (2019). "Deep reinforcement learning for swarm systems". In: *Journal of Machine Learning Research* 20.1, pp. 1966–1996.

Hutter, F., Hoos, H. H., and Leyton Brown, K. (2011). "Sequential model-based optimization for general algorithm configuration". In: *Learning and Intelligent Optimization: 5th International Conference, LION 5*. Vol. 6683. Lecture Notes in Computer Science. Springer, pp. 507–523.

Jones, S., Studley, M., Hauert, S., and Winfield, A. (2018). "Evolving behaviour trees for swarm robotics". In: *Distributed Autonomous Robotic Systems: The 13th International Symposium*. Vol. 6. Springer Proceedings in Advanced Robotics. Springer, pp. 487–501.

Karlson, P. and Lüscher, M. (1959). "'Pheromones': a New Term for a Class of Biologically Active Substances". In: *Nature* 183, pp. 55–56.

Kegeleirs, M., Garzón Ramos, D., and Birattari, M. (2019). "Random walk exploration for swarm mapping". In: *Towards Autonomous Robotic Systems: 20th Annual Conference, TAROS 2019*. Vol. 11650. Lecture Notes in Computer Science. Springer, pp. 211–222.

Kegeleirs, M., Ramos, D. G., Hasselmann, K., Garattoni, L., Francesca, G., and Birattari, M. (2024). "Transferability in the Automatic Off-Line Design of Robot Swarms: From Sim-to-Real to Embodiment and Design-Method Transfer Across Different Platforms". In: *IEEE Robotics and Automation Letters* 9.3, pp. 2758–2765.

Khaliq, A. A., Di Rocco, M., and Saffiotti, A. (2014). "Stigmergic algorithms for multiple minimalistic robots on an RFID floor". In: *Swarm Intelligence* 8.3, pp. 199–225.

Khaliq, A. A. and Saffiotti, A. (2015). "Stigmergy at work: Planning and navigation for a service robot on an RFID floor". In: *2015 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, pp. 1085–1092.

Kirkpatrick, S., Gelatt Jr., C. D., and Vecchi, M. P. (1983). "Optimization by simulated annealing". In: *Science* 220.4598, pp. 671–680.

König, L., Mostaghim, S., and Schmeck, H. (2009). "Decentralized evolution of robotic behavior using finite state machines". In: *International Journal of Intelligent Computing and Cybernetics* 2.4, pp. 695–723.

Kuckling, J. (2023). "Optimization in the automatic modular design ofcontrol software for robot swarms". PhD thesis. Université Libre de Bruxelles.

Kuckling, J., Hasselmann, K., Pelt, V. van, Kiere, C., and Birattari, M. (2021). *AutoMoDe Editor: A Visualization Tool for AutoMoDe*. Tech. rep. TR/IRIDIA/2021-009. IRIDIA, Université Libre de Bruxelles.

Kuckling, J., Ligot, A., Bozhinoski, D., and Birattari, M. (2018). "Behavior trees as a control architecture in the automatic modular design of robot swarms". In: *Swarm Intelligence: 11th International Conference, ANTS 2018*. Vol. 11172. Lecture Notes in Computer Science. Springer, pp. 30–43.

Legarda Herranz, G., Garzón Ramos, D., Kuckling, J., Kegeleirs, M., and Birattari, M. (2022). *Tycho: a robust, ROS-based tracking system for robot swarms*. Tech. rep. TR/IRIDIA/2022-009. IRIDIA, Université Libre de Bruxelles.

Lerman, K. and Galstyan, A. (2002). "Mathematical model of foraging in a group of robots: effect of interference". In: *Autonomous Robots* 13.2, pp. 127–141.

Li, G., St-Onge, D., Pinciroli, C., Gasparri, A., Garone, E., and Beltrame, G. (2018). "Decentralized progressive shape formation with robot swarms". In: *Autonomous Robots* 42.8, pp. 1–17.

Liao, T., Socha, K., Montes de Oca, M., Stützle, T., and Dorigo, M. (2014). "Ant colony optimization for mixed-variable optimization problems". In: *IEEE Transactions on Evolutionary Computation* 18.4, pp. 503–518.

Ligot, A. (2023). "Assessing and forecasting the performance of optimization-based design methods for robot swarms: Experimental protocol and pseudo-reality predictors". PhD thesis. Université Libre de Bruxelles.

Ligot, A. and Birattari, M. (2020). "Simulation-only experiments to mimic the effects of the reality gap in the automatic design of robot swarms". In: *Swarm Intelligence* 14, pp. 1–24.

Ligot, A. and Birattari, M. (2022). "On using simulation to predict the performance of robot swarms". In: *Scientific Data* 9.788, pp. 1–18.

Ligot, A., Cotorruelo, A., Garone, E., and Birattari, M. (2022). "Towards an empirical practice in off-line fully-automatic design of robot swarms". In: *IEEE Transactions on Evolutionary Computation*.

Ligot, A., Hasselmann, K., and Birattari, M. (2020a). "AutoMoDe-Arlequin: neural networks as behavioral modules for the automatic design of probabilistic finite

state machines". In: *Swarm Intelligence: 12th International Conference, ANTS 2020*. Vol. 12421. Lecture Notes in Computer Science. Springer, pp. 109–122.

Ligot, A., Kuckling, J., Bozhinoski, D., and Birattari, M. (2020b). "Automatic modular design of robot swarms using behavior trees as a control architecture". In: *PeerJ Computer Science* 6, e314.

Lindauer, M., Eggensperger, K., Feurer, M., Biedenkapp, A., Deng, D., Benjamins, C., Ruhkopf, T., Sass, R., and Hutter, F. (2022). "SMAC3: a versatile bayesian optimization package for hyperparameter optimization". In: *Journal of Machine Learning Research* 23.54, pp. 1–9.

Lipson, H. (2005). "Evolutionary robotics and open-ended design automation". In: *Biomimetics: Biologically Inspired Technologies*. Vol. 17. CRC Press, pp. 129–155.

Lipson, H. and Pollack, J. B. (2000). "Automatic design and manufacture of robotic lifeforms". In: *Nature* 406, pp. 974–978.

Lopes, Y. K., Trenkwalder, S. M., Leal, A. B., Dodd, T. J., and Groß, R. (2016). "Supervisory control theory applied to swarm robotics". In: *Swarm Intelligence* 10.1, pp. 65–97.

López-Ibáñez, M., Dubois-Lacoste, J., Pérez Cáceres, L., Birattari, M., and Stützle, T. (2016). "The irace package: iterated racing for automatic algorithm configuration". In: *Operations Research Perspectives* 3, pp. 43–58.

Ludwig, L. and Gini, M. (2006). "Robotic swarm dispersion using wireless intensity signals". In: *Distributed Autonomous Robotic Systems 7*. Springer, pp. 325–332.

Macinnes, I. (2003). "Visually guided physically simulated agents with evolved morphologies". In: *Advances in Artificial Life: 7th European Conference, ECAL 2003*. Vol. 2801. Lecture Notes in Computer Science. Springer, pp. 821–828.

Maes, P., Matarić, M. J., Meyer, J.-A., Pollack, J. B., and Wilson, S. W. (1996). *Building "Fungus Eaters": Design Principles of Autonomods Agents*. MIT Press.

Matarić, M. J. (1994). "Interaction and Intelligent Behavior". PhD thesis. Massachusetts Institute of Technology.

Matarić, M. J. (1997). "Reinforcement learning in the multi-robot domain". In: *Autonomous Robots* 4.1, pp. 73–83.

Mathews, N., Christensen, A. L., O'Grady, R., Mondada, F., and Dorigo, M. (2017). "Mergeable nervous systems for robots". In: *Nature Communications* 8.1, p. 439.

Mayet, R., Roberz, J., Schmickl, T., and Crailsheim, K. (2010). "Antbots: a feasible visual emulation of pheromone trails for swarm robots". In: *Swarm Intelligence: 7th International Conference, ANTS 2010*. Vol. 6234. Lecture Notes in Computer Science. Springer, pp. 89–94.

Mendiburu, F. J., Garzón Ramos, D., Morais, M. R. A., Lima, A. M. N., and Birattari, M. (2022). "AutoMoDe-Mate: automatic off-line design of spatially-organizing behaviors for robot swarms". In: *Swarm and Evolutionary Computation* 74, p. 101118.

Mondada, F., Bonani, M., Raemy, X., Pugh, J., Cianci, C., Klaptocz, A., Magnenat, S., Zufferey, J.-C., Floreano, D., and Martinoli, A. (2009). "The e-puck, a robot designed for education in engineering". In: *ROBOTICA 2009: Proceedings of the 9th Conference on Autonomous Robot Systems and Competitions*. Instituto Politécnico de Castelo Branco, pp. 59–65.

Na, S., Hanlin, N., Lennox, B., and Arvin, F. (2021). "Universal Artificial Pheromone Framework with Deep Reinforcement Learning for Robotic Systems". In: *2021 6th International Conference on Control and Robotics Engineering (ICCRE)*. IEEE, pp. 28–32.

Na, S., Qiu, Y., Turgut, A. E., Ulrich, J., Krajník, T., Yue, S., Lennox, B., and Arvin, F. (2020). "Bio-inspired artificial pheromone system for swarm robotics applications". In: *Adaptive Behavior*.

Na, S., Raoufi, M., Turgut, A. E., Krajník, T., and Arvin, F. (2019). "Extended Artificial Pheromone System for Swarm Robotic Applications". In: *ALIFE 19: The Fourteenth International Conference on the Synthesis and Simulation of Living Systems*. MIT Press, pp. 608–615.

Nedjah, N. and Silva Junior, L. (2019). "Review of methodologies and tasks in swarm robotics towards standardization". In: *Swarm and Evolutionary Computation* 50, p. 100565.

Nolfi, S. (2021). *Behavioral and Cognitive Robotics: An Adaptive Perspective*. Institute of Cognitive Sciences and Technologies, National Research Council.

Nolfi, S. and Floreano, D. (2000). *Evolutionary Robotics: The Biology, Intelligence, and Technology of Self-Organizing Machines*. MIT Press.

Nygaard, T. F., Martin, C. P., Samuelsen, E., Torresen, J., and Glette, K. (2018). "Real-world evolution adapts robot morphology and control to hardware limitations". In: *GECCO'18: Proceedings of the Genetic and Evolutionary Computation Conference*. ACM, pp. 125–132.

Pahl, G., Beitz, W., Feldhusen, J., and Grote, K.-H. (2007). *Engineering Design: A Systematic Approach*. Springer.

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

Payton, D., Estkowski, R., and Howard, M. (2005). "Pheromone robotics and the logic of virtual pheromones". In: *Swarm Robotics*. Vol. 3342. LNCS. Springer, pp. 45–57.

Pinciroli, C., Trianni, V., O'Grady, R., Pini, G., Brutschy, A., Brambilla, M., Mathews, N., Ferrante, E., Di Caro, G. A., Ducatelle, F., Birattari, M., Gambardella, L. M., and Dorigo, M. (2012). "ARGoS: a modular, parallel, multi-engine simulator for multi-robot systems". In: *Swarm Intelligence* 6.4, pp. 271–295.

Pugh, J. and Martinoli, A. (2009). "Distributed scalable multi-robot learning using particle swarm optimization". In: *Swarm Intelligence* 3.3, pp. 203–222.

Purnamadjaja, A. H. and Russell, R. A. (2010). "Bi-directional pheromone communication between robots". In: *Robotica* 28.1, pp. 69–79.

Reina, A., Cope, A. J., Nikolaidis, E., Marshall, J. A. R., and Sabo, C. (2017). "ARK: Augmented reality for Kilobots". In: *IEEE Robotics and Automation Letters* 2.3, pp. 1755–1761.

Rubenstein, M., Cornejo, A., and Nagpal, R. (2014). "Programmable self-assembly in a thousand-robot swarm". In: *Science* 345.6198, pp. 795–799.

Russell, R. A. (1997). "Heat trails as short-lived navigational markers for mobile robots". In: *1997 IEEE International Conference on Robotics and Automation, ICRA 1997*. Vol. 4. IEEE, pp. 3534–3539.

Russell, R. A. (1999). "Ant trails – an example for robots to follow?" In: *1999 IEEE International Conference on Robotics and Automation, ICRA 1999*. Vol. 4. IEEE, pp. 2698–2703.

Şahin, E. (2005). "Swarm robotics: from sources of inspiration to domains of application". In: *Swarm Robotics: SAB 2004 International Workshop*. Vol. 3342. Lecture Notes in Computer Science. Springer, pp. 10–20.

Salman, M., Garzón Ramos, D., and Birattari, M. (2024). "Automatic design of stigmergy-based behaviours for robot swarms". In: *Communications Engineering* 3.1, p. 30.

Salman, M., Garzón Ramos, D., Hasselmann, K., and Birattari, M. (2020a). *Phormica: Photochromic Pheromone Release and Detection System for Stigmergic Coordination in Robot Swarms*.
http://iridia.ulb.ac.be/supp/IridiaSupp2020-010/.

Salman, M., Garzón Ramos, D., Hasselmann, K., and Birattari, M. (2020b). "Phormica: photochromic pheromone release and detection system for stigmergic coordination in robot swarms". In: *Frontiers in Robotics and AI* 7, p. 195.

Salman, M., Ligot, A., and Birattari, M. (2019a). "Concurrent design of control software and configuration of hardware for robot swarms under economic constraints". In: *PeerJ Computer Science* 5, e221.

Salman, M., Ligot, A., and Birattari, M. (2019b). *Concurrent design of control software and configuration of the hardware for a robot swarm: supplementary material*. http://iridia.ulb.ac.be/supp/IridiaSupp2019-001/.

Schranz, M., Umlauft, M., Sende, M., and Elmenreich, W. (2020). "Swarm robotic behaviors and current applications". In: *Frontiers in Robotics and AI* 7, p. 36.

Sharpe, T. and Webb, B. (1998). "Simulated and situated models of chemical trail following in ants". In: *From Animals to Animats 5: Proceedings of the Fifth International Conference on Simulation of Adaptive Behavior*. MIT Press, pp. 195–204.

Silva, F., Urbano, P., Correia, L., and Christensen, A. L. (2015). "odNEAT: an algorithm for decentralised online evolution of robotic controllers". In: *Evolutionary Computation* 23.3, pp. 421–449.

Sims, K. (1994). "Evolving virtual creatures". In: *SIGGRAPH'94: Proceedings of the 21st annual conference on Computer graphics and interactive techniques*. ACM, pp. 15–22.

Soria, E., Schiano, F., and Floreano, D. (2021). "Predictive control of aerial swarms in cluttered environments". In: *Nature Machine Intelligence* 3.6, pp. 545–554.

Stanley, K. O. and Miikkulainen, R. (2002). "Evolving neural networks through augmenting topologies". In: *Evolutionary Computation* 10.2, pp. 99–127.

Talamali, M. S., Bose, T., Haire, M., Xu, X., Marshall, J. A. R., and Reina, A. (2020). "Sophisticated collective foraging with minimalist agents: a swarm robotics test". In: *Swarm Intelligence* 14.1, pp. 25–56.

Theraulaz, G. and Bonabeau, E. (1999). "A Brief History of Stigmergy". In: *Artificial Life* 5.2, pp. 97–116.

Theraulaz, G., Gautrais, J., Camazine, S., and Deneubourg, J.-L. (2003). "The formation of spatial patterns in social insects: from simple behaviours to complex structures". In: *Philosophical Transactions of the Royal Society of London. Series A: Mathematical, Physical and Engineering Sciences* 361.1807, pp. 1263–1282.

Trianni, V. (2008). *Evolutionary Swarm Robotics*. Springer.

Trianni, V., Groß, R., Labella Thomas, H., Şahin, E., and Dorigo, M. (2003). "Evolving aggregation behaviors in a swarm of robots". In: *Advances in Artificial Life: 7th European Conference, ECAL 2003*. Vol. 2801. Lecture Notes in Computer Science. Springer, pp. 865–874.

Watson, J. and Nitschke, G. (2015). "Deriving minimal sensory configurations for evolved cooperative robot teams". In: *2005 IEEE Congress on Evolutionary Computation*. IEEE, pp. 3065–3071.

Watson, R. A., Ficici, S. G., and Pollack, J. B. (1999). "Embodied evolution: embodying an evolutionary algorithm in a population of robots". In: *1999 Congress on Evolutionary Computation, CEC99*. Vol. 1. IEEE, pp. 335–342.

Werfel, J., Petersen, K., and Nagpal, R. (2014). "Designing collective behavior in a termite-inspired robot construction team". In: *Science* 343.6172, pp. 754–758.

Wilson, E. O. (1975). *Sociobiology: The New Synthesis*. Harvard University Press.

Winfield, A., Harper, C. J., and Nembrini, J. (2005). "Towards dependable swarms and a new discipline of swarm engineering". In: *Swarm Robotics: SAB 2004 International Workshop*. Vol. 3342. Lecture Notes in Computer Science. Springer, pp. 126–142.

Wyatt, T. D. (2014). *Pheromones and Animal Behavior: Chemical Signals and Signatures*. Cambridge University Press.

Xie, H., Sun, M., Fan, X., Lin, Z., Chen, W., Wang, L., Dong, L., and He, Q. (2019). "Reconfigurable magnetic microrobot swarm: multimode transformation, locomotion, and manipulation". In: *Science Robotics* 4.28, eaav8006.