Swarm Robotics : distributed Online Learning in the realm of Active Matter

Jeremy Fersula

THÈSE DE DOCTORAT
DE SORBONNE UNIVERSITÉ

Swarm Robotics : Distributed Online Learning in the realm of Active Matter

Jérémy FERSULA

Institut des Systèmes Intelligents et de Robotique, Sorbonne Université
Laboratoire Gulliver, ESPCI Paris
Ecole Doctorale Informatique, Télécommunications et Electronique (ED130)

Présentée et soutenue le : 14/12/2023
Devant le jury composé de :

Pr. Hamid KELLAY	Université de Bordeaux	Rapporteur
Pr. M-Carmen MIGUEL	Université de Barcelone	Rapportrice
Pr. Heiko HAMANN	Université de Konstanz	Examinateur
Dr. Carola DOERR	CNRS, Sorbonne Université	Examinatrice
Dr. Guy THERAULAZ	Université Paul Sabatier	Président du jury
Pr. Nicolas BREDECHE	Sorbonne Université	Directeur de thèse
Dr. Olivier DAUCHOT	CNRS, ESPCI Paris - PSL	Co-directeur de thèse

THÈSE DE DOCTORAT
DE SORBONNE UNIVERSITÉ

Swarm Robotics : Distributed Online Learning in the realm of Active Matter

Jérémy FERSULA

Institut des Systèmes Intelligents et de Robotique, Sorbonne Université
Laboratoire Gulliver, ESPCI Paris
Ecole Doctorale Informatique, Télécommunications et Electronique (ED130)

Présentée et soutenue le : 14/12/2023
Devant le jury composé de :

Pr. Hamid KELLAY	Université de Bordeaux	Rapporteur
Pr. M-Carmen MIGUEL	Université de Barcelone	Rapportrice
Pr. Heiko HAMANN	Université de Konstanz	Examinateur
Dr. Carola DOERR	CNRS, Sorbonne Université	Examinatrice
Dr. Guy THERAULAZ	Université Paul Sabatier	Président du jury
Pr. Nicolas BREDECHE	Sorbonne Université	Directeur de thèse
Dr. Olivier DAUCHOT	CNRS, ESPCI Paris - PSL	Co-directeur de thèse

Résumé

Avec la miniaturisation des composants électroniques et l'augmentation des performances des CPU / GPU modernes, il devient techniquement possible de développer de petits robots capables de travailler en essaims de centaines ou de milliers d'unités. Lorsque l'on considère des systèmes composés d'un grand nombre de robots indépendants en interaction, l'individualité s'efface devant le collectif, et le comportement global de l'ensemble doit émerger de règles locales. Comprendre la dynamique d'un grand nombre d'unités en interaction devient une connaissance clé pour concevoir des essaims robotiques contrôlables et efficaces. Ce sujet est au cœur du domaine de la matière active, dans lequel les systèmes d'intérêt présentent des effets collectifs émergeant d'interactions physiques sans calcul. Cette thèse vise à utiliser des éléments de la matière active pour concevoir et comprendre des collectifs robotiques, interagissant à la fois au niveau physique et au niveau logiciel par le biais d'algorithmes d'apprentissage distribués.

Nous commençons par étudier expérimentalement la dynamique d'agrégation d'un essaim de petits robots vibrants effectuant la phototaxie (c'est-à-dire la recherche de lumière). Les expériences sont déclinées dans différentes configurations, soit ad-hoc, soit mettant en œuvre un algorithme d'apprentissage distribué en ligne. Cette série d'expériences sert de référence pour l'algorithme, en montrant ses capacités et ses limites dans une situation réelle.

Ces expériences sont approfondies en changeant la forme extérieure des robots, ce qui modifie les interactions physiques en ajoutant une réponseEntrez le résumé en orientation aux forces extérieures. Cet effet supplémentaire modifie la dynamique globale de l'essaim, montrant que le *computation morphologique* est en jeu. La nouvelle dynamique est comprise grâce à un modèle physique d'auto-alignement, ce qui permet d'étendre le travail expérimental *in sillico* et de suggèrer de nouveaux effet à grande échelle dans les essaims de robots qui se réorientent.

Enfin, nous présentons un modèle d'apprentissage distribué par le biais d'équations différentielles stochastiques. Ce modèle est basé sur l'échange de degrés de liberté internes qui s'associent à la dynamique des particules, qui sont les équivalents dans le contexte de l'apprentissage à un ensemble de paramètres et à un contrôleur. Le modèle donne des résultats similaires en simulation à ceux des expériences réelles et ouvre la voie à une analyse théorique à grande échelle de la dynamique produite par l'apprentissage distribué en ligne.

Abstract

With the miniaturization of electronic components and the increase in performance of modern CPUs / GPUs, it becomes technically possible to develop small robots able to work in swarms of hundreds or thousands of units. When considering systems comprised of a large number of independent robots in interaction, the individuality vanishes before the collective, and the global behavior of the ensemble has to emerge from local rules. Understanding the dynamics of large number of interacting units becomes a knowledge key to design controllable and efficient robotic swarms. This topic happens to be at the core of the field of active matter, in which the systems of interest display collective effects emerging from physical interactions without computation. This thesis aims at using elements of active matter to design and understand robotic collectives, interacting both at the physical level and the software level through distributed learning algorithms.

We start by studying experimentally the aggregation dynamics of a swarm of small vibrating robots performing phototaxis (i.e. search of light). The experiments are declined in different configurations, either ad-hoc or implementing a distributed and online learning algorithm. This series of experiments act as a benchmark for the algorithm, showing its capabilities and limits in a real world situation.

These experiments are further expanded by changing the outer shape of the robots, modifying the physical interactions by adding a force re-orientation response. This additional effect changes the global dynamics of the swarm, showing *Morphological Computation* at play. The new dynamics is understood through a physical model of self-alignment, allowing to extend the experimental work *in sillico* and hint for unseen global effects in swarms of re-orienting robots.

Finally, we introduce a model of distributed learning through stochastic ODEs. This model is based on the exchange of internal degrees of freedom that couples to the dynamics of the particles, equivalents in the context of learning as a set of parameters and a controller. It shows similar results in simulation as the real-world experiments and opens up a way to a large-scale analysis of distributed and online learning dynamics.

Acknowledgments

This work would have not been possible if not for the support and encouragements from my PhD advisors, over these three years and despite my perpetual doubts. It is only natural that I begin by thanking Nicolas and Olivier, to whom I owe this work.

Naturally, I also thank the Sorbonne Center for Artificial Intelligence (SCAI) for making the PhD possible, and through which I had the chance of meeting many great people.

I would also like to thank the reviewers and examiners of the defense committee for their time and effort put in the evaluation of my work.

For having made these three last years memorable, I would like address many thanks to the PhD students, post-docs and interns from both labs, ISIR and Gulliver, among which I made great friends.

Lastly, I would like to thank my family for their presence and support over the years.

Swarm Robotics : Distributed Online Learning in the realm of Active Matter

Jérémy FERSULA

January 24, 2024

Contents

1 Swarm Robotics, Active Matter and their interface

This Chapter aims to familiarize the reader to the literature in both swarm robotics and active matter without going into technical details. Each section is structured in generalities followed by illustrative papers. Specific concepts, definitions, algorithms and equations that have inspired the work done in this thesis are presented in Chapter 2.

1.1 Swarm Robotics

In nature, biological swarms exhibit remarkable coordination and collective intelligence, involving thousands of individuals acting simultaneously on a shared task. Examples are numerous, from flocks of birds or schools of fish to social insects such as fire ants or termites, known to build complex and intricate nests far beyond their individual capabilities (see Figure 1). Swarm Robotics is an emerging field of robotics that seeks to unlock the potential of collective behaviors in large systems of robots.

Biological systems manage to achieve different complex behaviors, as flocks of birds fly in a coordinated manner through the skies, responding to environmental changes and avoiding collisions effortlessly. Social ants work cohesively, building intricate colonies, foraging for food, and defending their nests as a cohesive unit. These traits of self-organization, including collective motion, foraging, collective construction and transport constitute the main axes of research in Swarm Robotics [1].

Considering the number of agents in use G.Beni states in a 2004 conference paper, talking about swarm robotics, that this number is « not as large as to be dealt with statistical averages, not as small as to be dealt with as a few-body problem » [2]. Indeed, system sizes span several order of magnitude, from tens of robots up to thousands. It may seem counter-intuitive, but because of

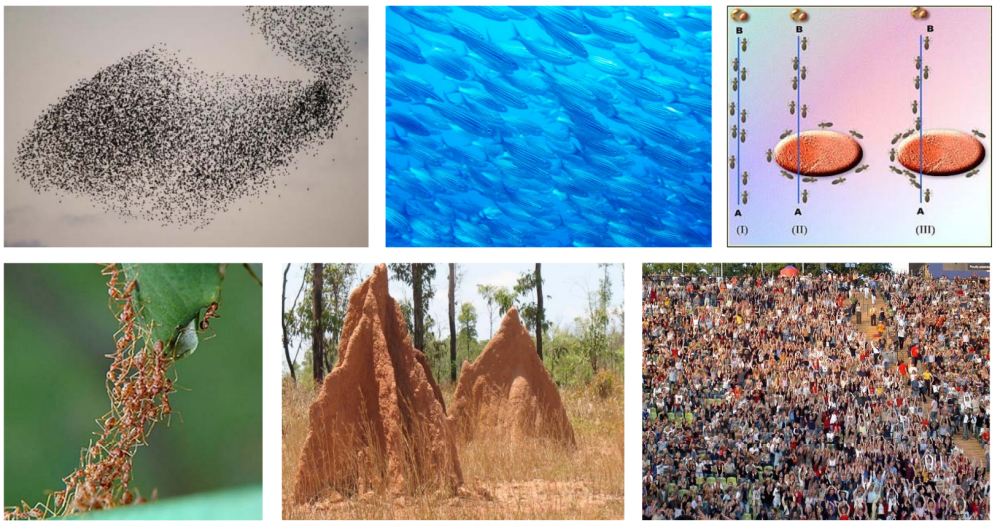


Figure 1 – Biological examples of self-organized systems. Flocks of birds, school of fish and social insects such as fire ants and termites are common examples of swarms achieving complex collective behavior beyond the limited capabilities of single individuals. Human crowds can also be included in large systems where the rules change drastically from the individual behavior to the collective, global behavior.

this Swarm robotics is not defined by considering the amount of units in the system [3] [2]. Systems considered to be swarm robotics systems are best defined with principles, as **autonomous physical robots** capable of **local** communication and sensing, for which the collective behavior **emerges** from local interactions. It is important to notice that this definition makes swarm robotics systems inherently distributed, with no centralization of control whatsoever. Each unit is meant to be perfectly independent from the other units, a key feature for scalability. On the notion of emergence, it is ambiguous and complex to define [1]. A simplified view on this notion in the case of robots can be considered : given the equivalent computation power and time to solve a problem as the swarm, a single swarm robot by itself could never achieve as good as the collective can.

In a seminal paper by Şahin [4], three desired properties have been identified as main motivations for swarm robotics studies: scalability, robustness and flexibility. An ideally designed swarm of robots would be able to maintain a cohesive state despite important changes of its configuration. The swarm should both be able to keep working when adding new robots (scalability) as well as keep working despite the loss or malfunction of robots (robustness). In addition, the swarm should be able to tackle changes in the environment without significant loss of performance (flexibility). Moreover, because of the high number of agents acting simultaneously, one can envision the swarm as acting like a massive parallel computational system and thus carry out tasks beyond those possible to a single robot. Thanks to these properties, swarm robotics has been recently highlighted as one of the grand challenges of robotics research for the upcoming years [5].

Before diving in the field of Swarm Robotics, we present a broad introduction to more "classical" robotic systems. A glimpse of the current context in robotics allows to envision what is currently possible in swarm robotics, and what is not.

1.1.1 « Traditional » Robotics

The field of Robotics is an interdisciplinary field by nature, combining mechanics, electronics, computer science and control, to mention the most important areas. Robots are ubiquitous in industrial applications [6], with a global market size estimated to be more than USD 72 billion in 2022 [7]. Many definitions of what is and what is not a robot can be found, and although a discussion on the semantics of the term would be interesting one fairly simple definition can be « machines that can replace human beings in the execution of a task, as regards both physical activity and decision making » [8]. In a robot, the ensemble of all mechanical and electronic parts which exert locomotion and manipulation is called the actuation system, split into one or multiple components : the actuators. In addition to the actuators, robots may be equipped with a set of sensors, allowing to monitor both the internal state of the robot and the state of its environment, through exteroceptive sensors such as cameras. Actuators and sensors are mediated by a control system, balancing current flows in the robot and taking decisions, either based on the physical model of the robot (an expert controller), or with an adaptive controller using artificial intelligence.

Robots are usually split into three classes : manipulators, mobile robots and hybrid robots. Manipulators have a fixed base, which is the case for robotic arms used in industrial processes. The robots in use in swarm robotics and multi-robot system are mobile robots, disposing of a mobile base allowing the robots to move freely in their environment. The third class, hybrid robots combines mobility with manipulators, making them complex machines of low interest in swarms (yet). Among the types of mobile robots one can have in mind when thinking about robotic swarms are flying robots, drones, also called UAVs for Unmaned Aerial Vehicles. Robots such as quadcopters are now very common as commercial robots and have been highly developed for both industrial and academic practices. However, coordinating swarms of drones without centralization still constitute a young field of research. The impressive drone light shows (see Figure 2) presented by companies such as Intel use GPS positioning and pre-programmed trajectories [9]. System of distributed drone swarms do exist, but they remain rare and are not the focus of this thesis. Discussion on drone swarms along with an example of a small swarm navigating complex environments can be found in [10].

Staying on the ground, multiple modes of locomotion exist, with the most common being of course the wheel. Legged robots and crawling robots are being actively developed by engineers and researchers, but in general they require higher degrees of freedom, greater mechanical complexity and are less power efficient on flat surfaces than wheels [11]. The key advantage however is worth the effort, with a far greater adaptability and maneuverability in rough terrain. A vastly popular example of a humanoid robot at work can be found within the work of Boston Dynamics and the Atlas robot. Because of the greatly increased complexity, similar legged robots with joints and moving components are non-existent today in the swarm robotics literature. Although each year the Robo-

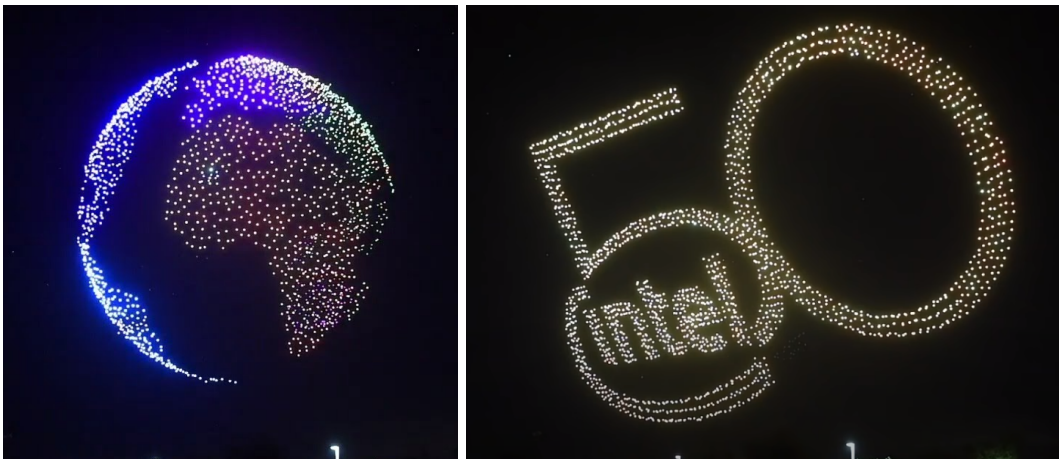


Figure 2 – Intel light drone show, shown at Folsom, California on July 15th 2018, with 1,500 drones simultaneously performing.

cup competition, held in Bordeaux in 2023, opposes teams of robots (7v7 in the small size league) one against the other in a small soccer match, which could be considered a multi-robot system task.

In industrial applications, classical robot controllers are mostly built upon inverse kinematics equations and error correction. The entire robot is described by writing down the equations of motions for each part of the robot, each joint and each motor. The problem of inverse kinematics lies in computing the appropriate torques and forces to apply as well as their duration in order to achieve a goal configuration of the robot. On robots with sensory-feedback, error correction is applied to help reaching precise positions or perform precise movements. The most common control process is called PID, for Proportional Integral Derivative. This process works by adjusting the control with an error at time t calculated by the sensors, $e(t) = y^*(t) - y(t)$ where y^* denotes the goal position and y denotes the current measured position. This value, its integral over time and its derivative are tuned with specific weights and sent back to the actuators, which performs a corrective motion. This process requires careful tuning but has been proven effective in classical control problems. An introduction to PID control can be found in [12]. These techniques are useful for single robots with goals of object displacement or self-balancing, but as the processing power diminishes, and the number of agents augments, and communication comes into play, the local environments of robots is prone to change and these techniques lose their relevance.

When dealing with unpredictable tasks or changing environments, expert controllers are replaced with AI with a goal of creating adaptive robots capable of working in many different and unseen situations. An intelligent robot is the embodiment of an intelligent *agent*, an automated controller capable of perception and taking actions maximizing its chances of success in every situation [13]. Using agents designed for robotics specifically is not fundamentally different than using agents purely *in silico*, as it relies on the same techniques and algorithms. However compared to simulations, deploying agents on real robots adds to the complexity of the task, due to the imperfect measurements and actions of a real world sensory-motor apparatus. To tackle this additional complexity,

some researchers work on deploying agents in simulated robots to learn real-world tasks, and then *transfer* the trained agent onto a real robot. Doing so allows both to test multiple approaches easily and to have a greatly increasing training time. This is however prone to a problem called the "Reality Gap", as agents trained in simulations perform worse in the real world compared to simulations [14]. Machine learning in general and reinforcement learning problems, common in AI for robotics, are discussed further in Chapter 2.

In swarms robotics, both the hardware and software of robots are kept simple. Controllers need to be designed to work on the available memory and computing capabilities, and the complex behavior of a swarm do not lie in the complexity of the controllers but the emergent effects of the collective. For this reason, controllers are usually kept simple such as small condition-action rules. The other popular approach is to use automated designs of controllers using machine learning. This transforms the problem of designing the controller to an additional axis of study for swarms, in which interactions in the system not only influence the large scale behavior of the swarm but also the individual behaviors.

1.1.2 Topics in swarm robotics

This section presents the current academic interests in swarm robotics. Research in the field encapsulate theoretical work, algorithmic and robotic design, and experimental work through simulations or real robot experiments. In the literature, it is a common trait to develop a specific algorithm or a specific robot in order to tackle a specific **task**. Figure 3 shows examples of tasks solved by swarms of real robots, as an illustration of the many aspects of the field. These tasks include self-assembly, where robots come together to create larger formations; foraging, where they search for and gather resources; flocking, mimicking cohesive motion patterns; collective construction and transport for building structures or moving objects; pattern formation to create specific shapes; and aggregation, synonym for clustering.

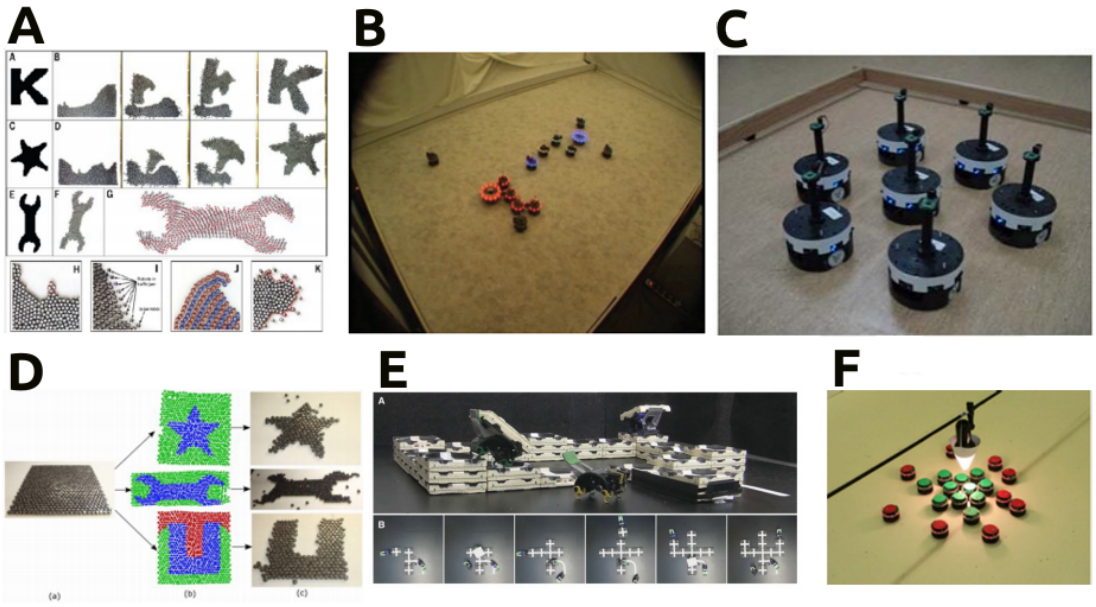


Figure 3 – Examples of tasks in swarm robotics performed by real robots. **A**) Distributed self-assembly of 3cm sized Kilobot robots into various shapes [15] **B**) Foraging in a collective of 12 s-bot robots, finding a randomly placed prey (red) and transporting it back to a nest (blue) [16] **C**) Self-organized flocking in a swarm of 7 Kobots [17] **D**) Pattern formation by disassembly in a swarm of Kilobots [18] **E**) Collective construction via brick stacking by 3 specialized robots [19] **F**) Separation of 20 e-puck robots in two distinct groups (segregation) using a physics inspired algorithm [20]

Many other tasks exist, of exploration, trajectory planning and decision making. Because the topics in swarm robotics are broad, multiple taxonomies where proposed over the years [21], classifying the numerous tasks tackled by robotic swarms, the different types of model and different design approaches. A rather concise taxonomy proposed by Brambilla et al. in 2013 [22] as a rework of a previous classification [23] splits the work done in tree-like structure, originating from

two overlapping categories :

- The **methods**, including all work done regarding the understanding of robotic swarms, theoretical and simulated models on both microscopic and macroscopic level, as well as the analysis of real robotic swarm.
- The **collective behaviors**, including all that is possible to do with a swarm of robots. This is the classification by task.

This taxonomy, represented in Figure 4, illustrates well the variety of topics in swarm robotics. The work done throughout this thesis focuses on a single task of aggregation, with different approaches : experiments, simulations and theory. The main corresponding topics are highlighted on the classification tree. In order not to get lost in the literature, we focus on work done around these topics, An extensive review of all tasks in swarm robotics can be found in Bayindir, 2016 [24].

In the design of controllers in swarm robotics, the Behavior-based design embed a classical algorithm in the swarm robots. Each example presented on Figure 3 uses a behavior based controller. The foraging s-bot robots in Figure 3.**B** are a good example of such a design, with their algorithm taking the form of a finite state machine. This type of controller uses states, in which each robot have a specific behavior such as "explore" in which the robot randomly moves in its environment or "transport target", in which the robot has physically grabbed the prey (a motionless colored object) and transport it back to the nest. Transition between states follow some conditions based on the current sensor input : is the robot close to the prey ? If so, grab the prey. In the case of flocking in Figure 3.**C**, the algorithm used is derived from a microscopic model of their robot, derived from a prior analysis of the robot sensors. This would place the corresponding paper in the categories of Microscopic modeling, Real-robot analysis, Behavior-based design and Coordinated Motion. Most of the time, work done in academic papers focus on one collective behavior addressed with original methods. The case of the self-assembly Kilobots in Figure 3.**A** and the segregating e-pucks in Figure 3.**F** are presented in more length at the end of this section, as they are good illustrations of applications of the Kilobot robot and physics inspired algorithms.

The complementary category to Behavior-based design, Automatic design, stand for adaptive controllers for which the behavior of the robots is not programmed in advance but discovered, either prior to the deployment (offline adapation) or during the deployment (online adaptation). A common approach [22] to implement this type of controller lies in evolutionary algorithms, a class of algorithms taking its roots in the principles of natural evolution, performing variation and selection of behaviors on the fly to optimize the success of the swarm. A more in-depth introduction to Evolutionary Robotics (ER) and evolutionary algorithms is presented in Chapter 2.

Regarding the different models in swarm robotics, macroscopic models are particularly interesting as they are independent of the number of agents in the swarm, and can therefore scale to any system size. In the literature, mean-field models try to describe the behaviors of swarms as continuous quantities using ordinary or partial differential equations, using mathematical tools borrowed from dynamical systems theory, and stochastic processes. The key idea is to think of a swarm not in terms of individual agents but as a distribution in a state space, either discrete or

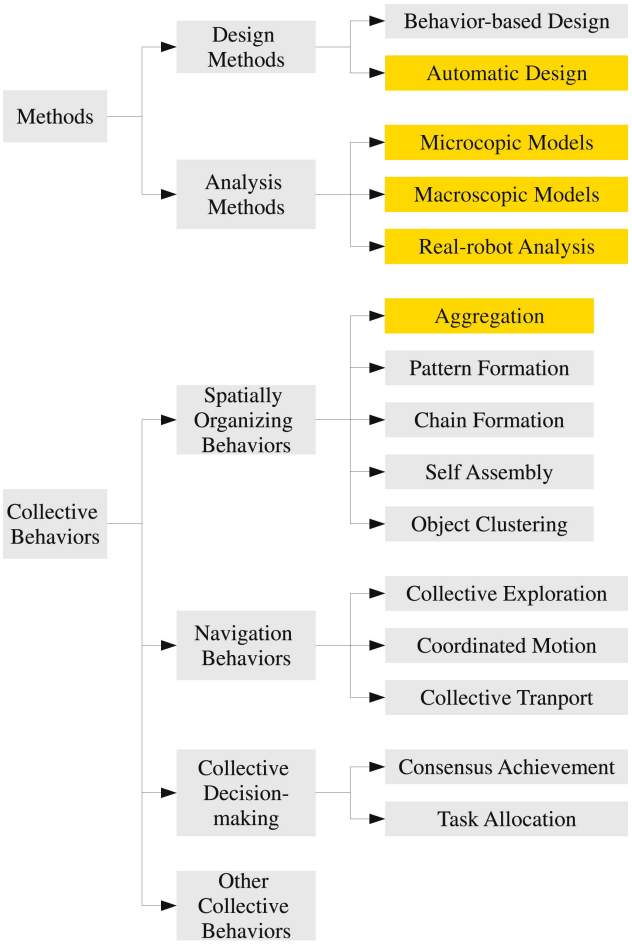


Figure 4 – Classification of the main swarm robotics topics and collective tasks, Brambilla et al. 2013 [22]. By studying a system of real robots performing aggregation tasks, and proposing physics based models, the work done throughout this thesis can be placed in the highlighted topics.

continuous. Example of a discrete state space can be a task of foraging, in which each robot must gather a resource and go back to the nest. In this case, we can distinguish 3 states : exploring robots, robots carrying a resource in the wild, robots carrying a resource in the nest. This can be seen as a Markov Chain with time varying transition probabilities. In this context, a model describe the time evolution of the distribution of robots in each state, and asks questions such as the existence and control of stationary distributions. In continuous state spaces, robots can be considered particles freely moving in space, their state being their position at a given time. This type of model constitute an interesting crossroad between robotics, physics and mathematics and can help understand large-scale behaviors of swarms. A review of mean-field models for swarm robotics can be found in Elamvazhuthi et al., 2019 [25].

As of today, real-world applications of swarm robotics are envisioned but remain to be deployed.

Only few published experiments have been able to control large amount of real robots, comparable to biological swarms or flocks [26]. However, many domains of application have been identified, with tasks requiring coordinated action over large areas, requiring redundancy or action in dangerous zones [27] :

- Logistics, using hundreds of autonomous robots to transport, move and retrieve inventory in warehouses [28].
- Space missions, in the case of remote robots that cannot be easily retrieved or fixed, failure of one robot would not degrade the swarm performance, in addition to a greater area coverage compared to a single robot system [26].
- Military and defense with automated weaponry and scouting [27].
- Automated agriculture, with already multiple working prototypes that can assess crop quality, diagnose plant health, apply phytosanitary products or automatically harvest [29].
- Micro-manipulation and precision medicine, with the recent development of small, soft-actuators suitable for centimeter to nanometer-scale robots . At this scale, robots are often too small to carry a power source or a controller, and have to be piloted by external signals, through electric or magnetic fields, light, chemical potentials etc. Goals include minimally invasive surgery, targeted drug delivery or diagnostic [30].

1.1.3 Swarm robotics platforms

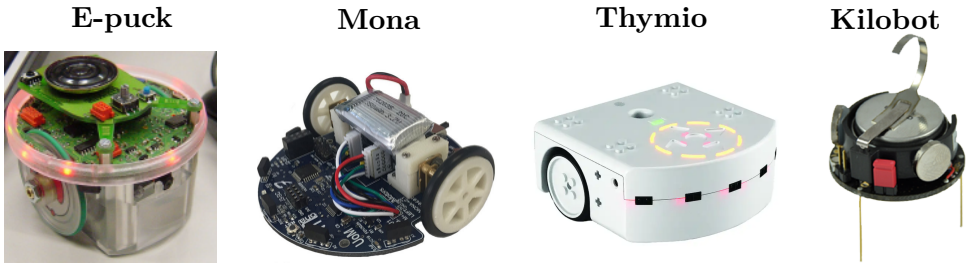
Multiple robotic platforms are used in the research on swarm robotics, balancing cost and capabilities of the robots and testing different swarm designs. Real-life robots used in the literature are numerous, and tend to quickly change in reaction to advances in the field and in electronics. A review by Nedjah et al. published in 2019 identify seventeen robots as most prominent in the literature [21], of which a selection are illustrated in Figure 5. Robots differ by their size, components, computation capabilities, price, design and commercial availability.

As swarm robotics is still a young field of research, there is yet no generic method or generic robots that can be used in any application, with many robots designed and built with specific hardware capable of performing specific tasks [21]. Robots span multiple sizes, from 2cm with the *Alice* (g) and *Jasmine* (k) robots up to 40cm with the hand-bots of the *Swarmanoid* project (d). All of the robots presented here are wheeled, except for the *Droplet* (h) robot which vibrate, moving using the *stick-and-slip* principle, detailed in Chapter 3.1. Cost range from 25 USD in parts for the *Colias* robot, up to 1300 USD for the *E-puck* and 3200 USD for the *Khepera IV* robot. Some robots, such as the *S-bot* (a) are designed to be re-configurable, as they can perform auto-assembly and they can collaborate with other robots, such as the *MarXbot* (b) along which both are part of the Swarmanoid project based on the coordination of heterogeneous swarms.

Focusing on the robots that appear in the papers presented further below, we present the E-puck, Mona, Thymio and Kilobot robot, all robotic platforms available in the lab. The four of these robot are open-source, open-hardware robots and are being commercialized by at least one

Figure 5 – Catalog of robots used in the swarm robotics literature, edited from Nedjah et al. 2019 [21].

company. Provided the time and resources, it is possible to build our own version of the robots, although it is greatly faster and easier to buy directly from a manufacturer.



- **E-puck** [31] is one of the most prominent robot used in swarm robotics research. It is a wheeled robot of 7.4cm in diameter designed to be very user-friendly and as interactive as possible, in order to be used for education purposes in teaching students in engineering. It contains many different sensors of different nature, with eight infrared proximity sensors (2-3cm) around the body, a 3D accelerometer, three microphones and a color camera. On the

actuation part, the E-puck has two wheels driven by stepper motors, a speaker and a total of ten LEDs, eight of them around the body. Is has a maximum speed of about 13cm.s^{-1}, and an average autonomy in movement of 2 hours. The E-puck can communicate with the computer or one with another or via Bluetooth. It is programmed in C with a custom library, and made extensible with many different boards mounted on top that provide additional hardware. The original production cost is estimated to be around 250 euros, in 2023 the cost of buying a completely assembled E-puck is 1200 euros.

- **Mona** [32] was initially based on the Colias robot, as a low-cost robot developed for research and education. It measure 8cm in diameter, and is equipped with the same micro-controller [1] as the Kilobot (ATmega328). The Mona robot has five Infrared sensors placed on the front for obstacle avoidance (5 $\pm$ 1 cm), and two rotary sensors (hall effect encoders) that provide feedback on the wheel's speed. It shares approximately the same speed and autonomy as the E-puck, 2.3 hours in movement and a maximum speed of 11cm.s^{-1}. Again similarly to the E-puck, extensions boards can be plugged on top to add hardware. In particular, robot to robot communication needs to be provided by an external module.

- **Thymio** [**?**] is a robot developed for education aimed at children over 6, but re-programmable for research purposes. Because of this, the robot is very safe and easy to manipulate. In the same way as the Mona, it is equipped with five infrared proximity sensors placed on the front, with an additional two on the back. It also has two proximity sensors placed on the bottom used to implement line following. Additional sensors include a microphone, temperature sensor and accelerometer, as well as capacitive touch buttons on top. Actuation is comprised of 39 LEDs, a speaker and the two motors piloting the wheels. Thymio can be programmed with various languages of increasing complexity, from the Scratch language to Python and ROS. The robot is commercially available at a retail price of 130 euros.

- **Kilobot** [33] robot is a 3cm wide vibrating robot, with a movement produced by the *stick-and-slip* effect rather than wheels. The Kilobot is rather simple, the only actuation being the vibrating motors and a RGB LED. Sensors include an infrared emitter/receiver for the communication, and an ambient light sensor. Because the Kilobot is the main robot used throughout this thesis, it is detailed in-depth in Chapter 3.1.

1.1.4 Illustrative examples

In order to get a better view on swarm robotics literature and provide general knowledge on the methods and realization of typical real-world robot experiments, we succinctly present two papers related parts of this thesis.

1. A type of chip combining the basic elements of a computer including CPU, RAM (volatile memory used for computation) and flash memory (non-volatile memory).

Thousand robots self-assembly

The Kilobot robot began to get traction in the swarm robotics community along with a 2014 article [15] proposing a self-assembly algorithm experimentally tested on a Kilobot swarm of a thousand robots. Self-assembly takes its inspiration from the process of morphogenesis in nature, process by which organisms develop their form and structure during growth and development. In the case of swarm robotics, self-assembly involves the robots autonomously determining their positions, orientations, and connections in order to form specific patterns or structures.

The thousand robot swarm experiment (working with precisely 1024 robots) is to date the best illustration of the scalability of the Kilobot as a swarm robotics platform. The self-assembly algorithm developed for this paper is based on three primitive behavior : 1) edge-following, 2) the evaluation of the distance to the source via a gradient value message that increments as it propagates through the swarm and 3) the localization, where robots form a local coordinate system by measuring distances to neighbors. The final algorithm links these three primitives with a finite-state automaton. Here, it is important to notice that motion is only made when a robot follows a layer of static Kilobots. The moving Kilobot constantly compensate for its biased motion by an error-sensing feedback provided by the static neighbors, when evaluating their relative distance.

In the experiment, all Kilobots are equipped with an identical program, implementing the self-assembly algorithm and incorporating an image of the target shape. Initially, the robots are grouped together without knowledge of their location. To initiate self-assembly, four seed robots are placed next to the group, marking the desired position and orientation of the future swarm shape. A gradient formation starts from the seeds, enabling the Kilobots to know if they lie on the edge, and if so start moving. A coordinate system is collectively constructed using distance information among robots, with the stationary seed robots serving as the origin. A robot can localize itself if it can observe at least three stationary and noncollinear localized robots. It can then perform trilateration of the distances received and compute its own position. The Kilobots then determine if they are inside the shape by comparing their position to the shape image. They continue edge-following until they enter the shape, then either halt if about to exit or if adjacent to a stationary robot with the same gradient value. The assembly process continues until the shape is filled. The algorithm allows for various shapes, operates in continuous space, and ensures correct formation. Real-world implementation requires addressing challenges like the imprecise locomotion, noisy sensing, and message loss. This is done through cooperative monitoring and fault correction, coupled with a very slow motion relative to the computation speed. Indeed, a single experiment lasts approximately 12 hours.

Three of the main experimental results are shown on Figure 6. The achievement of these experiments lies both in the complete absence of human intervention and in the distributed nature of the algorithm, showing that the collective behavior is remarkably robust. The final shape accuracy is not perfect but is quite high, with errors made at the individual level lessening the final quality without preventing completely the self-assembly process.

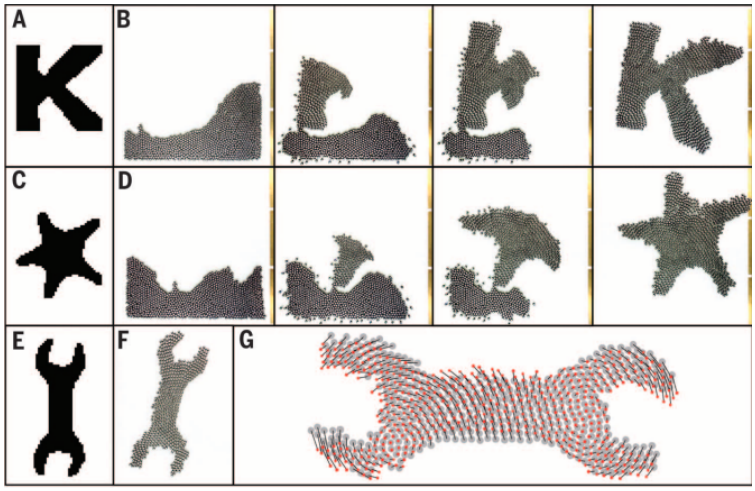


Figure 6 – Self-assembly examples in a swarm of up to 1024 Kilobots. (**A**, **C** and **E**) Desired shapes given as inputs to the Kilobots. (**B** and **D**) Illustrations of the self-assembly process. Robots start from an initial reservoir on the left and assemble progressively over the course of up to 12 hours into connected shapes. (**F**) Distortion in the final shape compared to the input. (**G**) The accuracy is measured by comparing the true positions of the robots in red to their internal localized position in gray. Credit : Rubenstein et al. 2014 [15]

Robot segregation based on the Brazil nut effect

When shaking a container of granular material with constituents of different sizes, larger grains tend to rise to the surface and smaller grains to sink to the bottom. The name comes from the observation of this effect in containers of mixed nuts, in which the largest, Brazil nuts, naturally rise to the top of the packaging. This effect in a mix of particles of two different sizes segregates the large ones and the small ones, separates them into spatially distinct groups, based purely on physical interactions. In swarm robotics, spatial segregation necessitate the implementation of specific control laws and complex computations, which drive the inspiration to develop a simple method of segregation based on the physical effect.

A first version of such an algorithm was proposed in a 2009 paper by Groß et al. [34] and implemented in simulation. However, this version assumed that every robot can measure the relative position of close robots. For the deployment on real robots equipped with directional vision, and prone to errors, a modified version of the first algorithm was used in Chen et al. 2012 [20], presented here. The robots chosen to perform the experiments is the E-puck robot, presented above.

The controller works by emulating disks of different radius, performing random motion and subject to virtual forces : an attractive potential emulating gravity (*taxis*), a repulsive potential for the virtual collisions between disks. In practice, each robot computes directly its velocity by adding contributions with $v = v_{taxis} + av_{rand} + bv_{repul}$. Each contributive term is normalized and the numbers a and b are parameters of the model. Each robot is programmed to cycle between a

stopped phase in which it assesses its next velocity and a moving phase in which the robot orient itself in the correct direction and takes a step of fixed duration. Because the robots do not exchange information, the interactions are in fact asymmetric, with each robot i computing collisions as if every other robot j shares the same radius, $r_j = r_i$. In addition, the repulsion term v_{repul} scales linearly with the distance, with $v_{repul} \propto 2r_i - d_{ij}$ when $d_{ij} < 2r_i$.

The taxis term is implemented in the experiment with a point source of light at the center of the arena, which every robot can localize via a set of infrared sensors positioned all around the e-pucks. Experiments with 20 E-pucks are performed for a maximum duration of 20 minutes, illustrated on Figure 7. Robots are split into two populations of virtual diameters 8cm and multiples of 8cm. The number of robots in each population is chosen based on the ideal segregation pattern expected, a disk of the first population in the center fully contained within the area of an annulus composed of the second population.

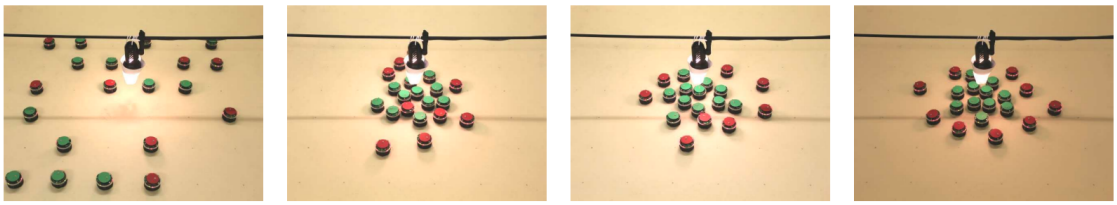


Figure 7 – Time evolution of 20 E-puck robots implementing the Brazil nut effect algorithm. The light at the center of the arena is used to generate the attractive potential. Each robot is 7.4cm in diameter, with 11 green robots emulating disks of 8cm in size and 9 red robots 16cm. A single experiment lasts 20 minutes. Credit : Chen et al. 2012 [20]

1.1 Swarm Robotics

This algorithm demonstrate how a simple idea inspired from a physical phenomenon allows to easily solve a problem of segregation in swarm robotics. Because this algorithm does not require the robots to communicate, the computing time do not scale with the size of the swarm and the performance could scale well with the system size. Simulations in a previous paper [34] support this claim.

1.2 Active Matter

Active Matter is the field of physics whose interests lies in the description and understanding of systems composed of a large number of autonomous agents, ranging from micro-swimmers, to cells, birds and even humans [35]. In these systems, the agents possess a source of energy, whether it comes from a chemical potential, a battery or even food, and transforms it into mechanical energy. Doing so make the individual constituents *active* as they continuously dissipate energy to perform motion. This creates new and rich phenomena happening spontaneously, arising not from external fields or geometrical constraints but from the interactions between the constituents.

As in swarm robotics, active matter draws inspiration in biological self-organized systems such as flocks and herds. It ranges across many different systems of different properties, but contrary to swarm robotics, active matter is a field of physics and is more agnostic to tasks and applications and more focused on the understanding of governing principles.
Active matter can be seen as an instance of soft matter, as they share similarities in the systems considered and mathematical tools used.

1.2.1 A specific instance of Soft Matter

The field of soft matter usually regroups the studies of a variety of systems ranging from polymers, to colloids, gels, foams, liquid crystals and granular materials. The physics of soft matter lies between the ones of ordered and disordered systems. In solids displaying crystalline order, such as metals, the constituents are regularly stacked on a lattice and the state of the whole system is driven by energy. Systems reach the configurations of lowest energies, and the thermal fluctuations agitating the atoms is very low compared to the binding forces between atoms. As temperature increases, the total energy of the system augments, and this energy is distributed among the constituents under the form of motion, vibrations, electronic excitation etc. The total number of ways the energy can be distributed defines the entropy. For example in a perfect crystalline state, where nothing is moving or fluctuating, the system can only be in one state and its entropy is essentially zero. When considering systems significantly less dense than crystals such as gases, constituents are not bound to one another, move freely in space and systems are highly disordered. The energy of thermal fluctuations is now greater than the binding energies, and these systems are now driven by entropy. At equilibrium, a gas maximizes the number of possible micro-states corresponding to its total energy, it maximizes its entropy. Soft matter systems lies between the two extremes, binding energies are strong enough to maintain some structure, but the systems are only partially ordered and the building blocks can undergo local rearrangements with thermal fluctuations or external forces [37] [38].

One type of system studied in Soft Matter are liquid crystals, which owe their name to the fact that they are not quite liquids and not quite crystals. This type of material is called *mesomorphic* for its ability to display *mesophases*, phases that lies in-between "regular" states of matter, in-between liquid and solid. Liquid crystals are formed by elongated particles, small rods or polymers, and share similarities with solids as they exhibit patterns repeating in space, and similarities with liquids as they can flow. Because the constituents are anistropic, their orientation in space

cannot be disregarded. In a typical phase of liquid crystal called the **nematic** phase, molecules are all aligned to one another but free to move along the direction of their orientation. By extension, the adjective nematic is used to describe systems other than liquid crystals which align with no preferred direction of movement along the axis of alignment. In liquid crystals, other phases more or less common exist as well, such as the more ordered *smectic* phase, in which molecules regroup into superimposed layers, roughly perpendicular to the orientation, imposing both a constraint on the orientation and on one direction of space.

In its book about the physics of liquid crystal [39], first published in 1974, De Gennes states that mesophases can be obtained in two different ways :

1. Imposing no positional order, or a partial positional order in only one or two dimensions for a three dimensional system
2. Introducing additional degrees of freedom apart from the positions of the centers of gravity. For non-spherical molecules, elongated or rod-like, the orientation is the most natural candidate.

We can note that this covers not only liquid crystals but active systems as well, an example being oriented self-propelled particles, which are not constrained in position and have additional degrees of freedom.

1.2.2 Different types of active matter

A broad definition of what is an active particle and what is not can be formulated using two components :

1. Individual particles consume their own mechanical energy, either by borrowing energy from their environment or via an internal mechanisms such as a battery or the metabolism of a living unit. In the words of physicist, particles spontaneously **break the time-reversal symmetry**. This breaking of symmetry essentially means that freezing the system to its current state at a time t_0 and substituting t by $-t$ in the equations describing said particles do not produce the inverse motion. This breaking of symmetry is equivalent to the total energy of the described system not being conserved. Note that in active systems, and in other cases of time-reversal breaking, the energy is not conserved because we deliberately choose not to (or it is impossible to) describe all the degrees of freedom of the system, not their interactions with the environment. When studying the motion of birds, we consider only the positions and forget about the infinite complexity of a bird's organism.
2. Individual particles also **break a spatial symmetry**. This means that particles are either not isotropic, or they are spinning in a specific direction. In order not to be isotropic, some particles have a different rear and front (polar particles), while some particles are elongated enough such that their orientation in space cannot be disregarded.

Active systems of self-propelled rods are one of the early systems studied in active matter, as they grew out from soft matter as the non-equilibrium extension of liquid crystal physics [40]. Since liquid crystals are also elongated, rod-like particles, they can be considered the *passive* version of self-propelled rods. Two distinct types of self-propelled rods (and active matter systems in general) are often considered separately : **dry** and **wet**. In dry systems, momentum is not conserved because the substrate, often the ground or a viscous fluid, is disregarded. In these systems, the dominant interactions are short-ranged repulsion between the active particles. On the contrary in the case of wet systems, the momentum gained or lost by the substrate cannot be neglected [41]. This creates complex hydrodynamics effects and complex interactions between active particles at a range greater than the size of the particles. Wet systems can also exhibit turbulence, chaotic flows etc.

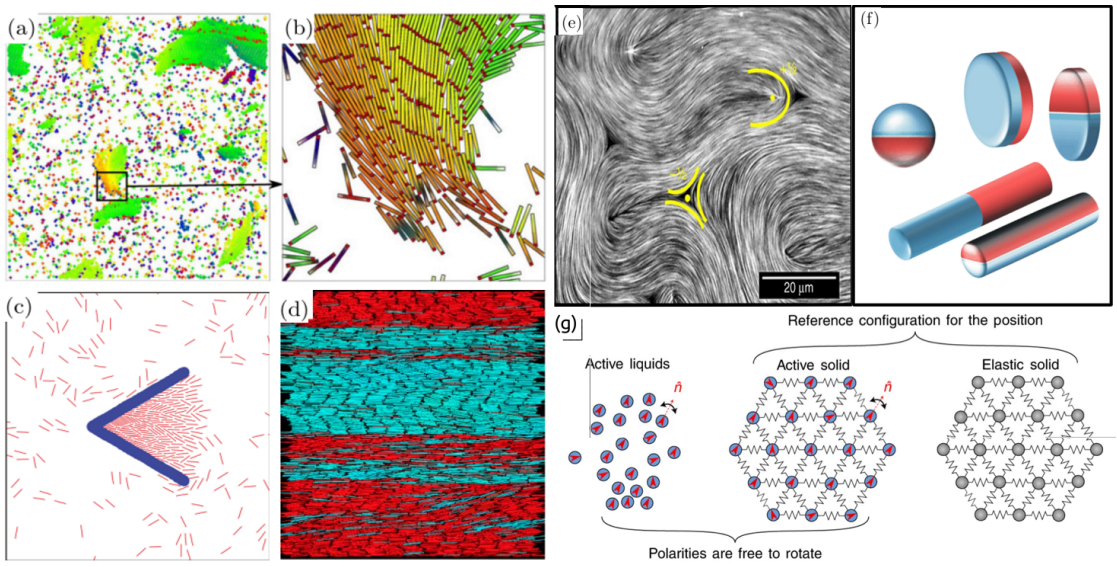


Figure 8 – (a) Polar clusters with local smectic order of dry self-propelled rods (b) Zoom on the local smectic order [42]; (c) Spontaneous accumulation of self-propelled rods in a static wedge [43]; (d) Laning in high density self-propelled rods [44]; (e) Active Nematics of kinesin and micro-tubules, showing topological defects [45]; (f) Examples of Janus particles [46] ; (g) Schematics of active solid, a combination of active particles and elastic networks. [47]

Dry self-propelled rods are simple in principle but able to display a variety of collective phenomena. For example, they can spontaneously phase separate, forming nematic clusters in a disordered gaseous phase Figure 8.(a),(b) for a packing fraction [2] $\eta \leq 0.3$ [42]. Another interesting property of self-propelled rods is their ability to self-trap in specific geometries [43], as shown in Figure 8.(c). In this situation, a static wedge is capable of aggregating the rods for some range of apex angles, all with purely repulsive interactions and no form of sensing whatsoever. At high density, a phenomenon of *laning* appear, Figure 8.(d), with rods spontaneously separating in large

2. Relative area aN/L^2 for N particles of area a in a 2D box of side length L.

bands sharing a direction, similar to road lanes [44].

One experimental system that can be considered an extension of rods, or another "active" equivalent to liquid crystals, are biological systems of molecular motors and filaments. One standard realization of this system is made from kinesin, a protein fueled by ATP hydrolysis responsible for the transport of cargo along microtubules, and microtubules, which are long tube-like structures found in the cytoskeleton. In this mix, the microtubules are able to bundle and slide, creating a flow-like motion shown on Figure 8.(e). This kind of system displays moving topological defects [3] which drive the surrounding flows, and get spontaneously created or annihilated over time.

In the case of polar active particles, many systems developed as an extension of colloidal systems such as active droplets [48] or Janus particles [46]. Droplets can be oil droplets in aqueous phases or aqueous droplets in oil phases. Chemical reactions happening at the boundary create an instability in the surface tension around the droplet, which quickly leads to a spontaneous breaking of symmetry and propulsion. Janus particle on the other side are small beads or cylinders of a few micrometers in size coated on one side with a different material, able to propel using self-generated chemical or thermal gradients, Figure 8.(f). The idea behind propulsion reside in the breaking of a chemical equilibrium on one side only, creating a difference in chemical potential and propelling the particle forward. This can be achieve by catalyzing a naturally occurring reaction such as the degradation of oxygenated water (hydrogen peroxide). In some systems, researchers are also able to emulate a layer of computation, by tracking in real time the particles and conditionally switching on the activity of the beads. This is done by hitting individual particles with a precision laser. The particles are submerged in a binary mixture at a constant temperature close to the temperature of demixing. When hit by a laser one side gets heated above this critical temperature, breaking the chemical equilibrium and creating motion [49].

Active components may sometimes be elastically connected, creating Active Solids Figure 8.(g). Connecting active polar particles create systems driven by both active forces and elastic forces. These forces can balance or dynamically interact and can lead to new oscillating phenomena which can be observed in nature, in systems such as epithelial cells sheets [50]. This type of oscillations around the equilibrium positions is known as collective actuation, a phenomenon arising from the interplay between activity, elasticity and geometry [47].

1.2.3 Active systems of polar particles

Because robots in swarm robotics are non-holonomic [4], they possess a preferred direction, and the momentum is immediately dissipated on or borrowed from the ground. Therefore, they fall into

3. A topological defect here denotes a discontinuity in the orientation field, singular points that cannot be assigned an orientation. The existence of such points is closely related to the topology and the boundary conditions of the medium, for example a sphere always present two defects (so-called «hairy ball theorem») but a torus can have none.

4. A holonomic robot is a robot that has no coupled translational degrees of freedom. Typically, if a wheeled robot has a turning radius greater than 0, it is non-holonomic.

the class of dry and polar active particles. This section presents experimental systems studied in dry and polar active matter.

Among the experimental realization of active polar particles is the system of colloidal rollers [51]. This system of colloids [5] is composed of solid beads in a liquid solution of hexadecane, trapped on a 2D plane between two electrodes. With the application of an electric field in the solution, the particles start to roll in a random direction due to a phenomenon known as Quincke rotation. This effect allows the engineering of a system of millions of self-propelled particles moving simultaneously, and interacting through collisions. These particles are trapped in a closed loop of large width such that interactions with walls do not create a preferred direction. The system is shown on Figure 9 (a). When considering low densities, measured in area fraction, the particles move in random directions and hardly interact Figure 9 (b). At low density with little to no interactions make this state considered as a gaseous phase, by analogy with the microscopic description of gases. At higher density however, the system starts to exhibit polar order, and particles move in a shared direction along the 2D track, Figure 9 (c) and (d). Multiple experiments ran at different densities ϕ show the apparition of polar order starting at a critical density value ϕ_c, above which the system undergo a phase transition from an isotropic gas to a polar liquid. This type of phase transition is standard in polar active matter and appear in a wide variety of systems and models, one can expect such behavior to emerge as long as particles satisfy criteria of self-propulsion and inter-particle alignment. This generality is discussed in the next section with the presentation of the Vicsek model.

This system has also been studied in confined space [52], showing that a combination of alignment and repulsion forces in confinement produce large-scale vortices. Theoretical work backing up the experimental data suggest that this propensity for the particle to perform vortical motion is a generic effect in dense systems of self-propelled particles in confinement.

Another example of polar active particles in which interactions are made through contact illustrates well how the design of individual components can drive collective effects : the self-propelled hard disks, or walking-grains [53]. The disks presented on Figure 10 are 4mm in diameter, 2mm high and made of a copper-beryllium alloy. The disks are polar for they have two distinct sides : a rigid small metal leg is placed at the front and a large soft leg, made out of rubber, is placed at the back. Both legs exhibit different mechanical responses to vibrations, in such a way that vibrating the disk vertically propels them forward. They are place on top of a vibrating tray, performing sinusoidal vibration of frequency $f = 115$ Hz, a frequency chosen such that it avoid all resonances in the system while inducing motion of the disks. The individual motion of the polar disks is studied by performing an experiment in a very dilute regime, with 50 particles (surface fraction of $\phi \approx 0.02$). The disk exhibit a persistent motion, with a clear difference compared to isotropic particles made of the same alloy which perform a diffusive motion, shown on Figure 11 (a) and (b). The disks move for the most part in the direction of their orientation, as it is shown by the distribution of the angle α between orientations and velocity, having a mean at zero and small variance Figure 11 (c). The behavior of the disks is also studied for different values of relative acceleration to gravity, controlled by performing the vibrations over different amplitudes at constant frequency.

5. Suspension of small ($< 1\mu m$) dispersed particles in a medium.

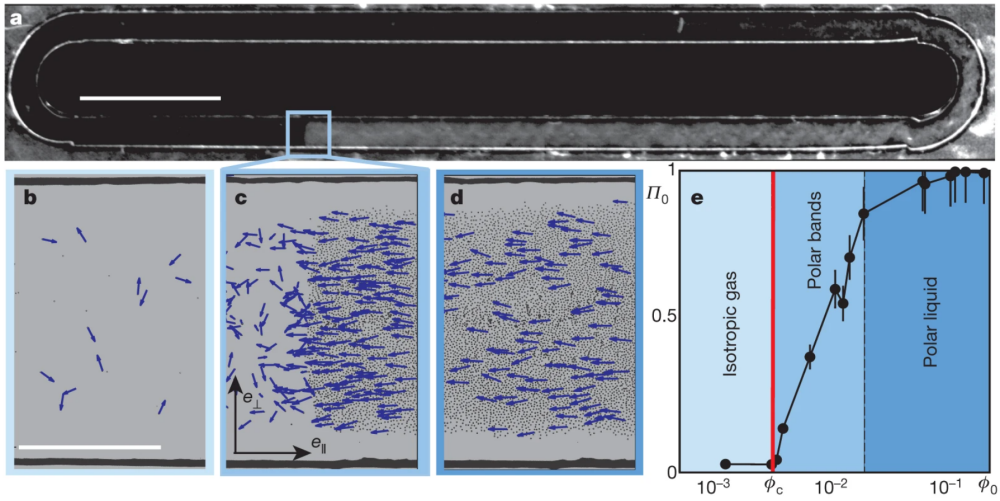


Figure 9 – Experimental system of rolling colloids, active units based on the Quincke rotation effect. **a**. Picture of the racetrack showing a band of rollers in movement. **b** Gaseous phase of the system, particles orienting at random in a region of very low density **c**. Front of a propagating band **d**. Polar liquid, with particles sharing on average a common orientation **e**. Polarization of the system as a function of the area fraction ϕ_0, showing a transition from disordered gas to polar liquid above a critical density ϕ_c. Credit : Bricard et al. 2013 [51]

A first effect of the amplitude that can be observed on the individual motion of the disks is the increase in angular noise Figure 11 (d), meaning the motion of disks is more persistent when the amplitude of the oscillations is small.

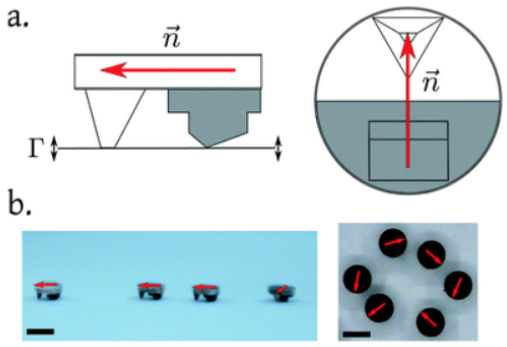


Figure 10 – Experimental system of self-propelled hard disks (or walking grains). **a**. Schematic representation of a single particle, a disk of alloy 4mm in diameter, with a front thin leg and a wide back leg. **b**. Pictures of disks in situation. Motion is created by vibration of the support tuned at $f = 115Hz$.

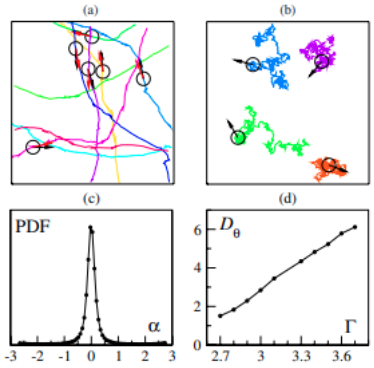


Figure 11 – Individual motion of the self-propelled hard disks. **a**. Examples of trajectories of self-propelled hard disks in a dilute regime, showing persistent motion. **b**. Trajectories of isotropic particles of the same alloy acting a control for the system. **c** Experimental distribution of α the measured angle between the orientation of the disks and their velocities. **d**. Experimental measurement of the angular diffusion coefficient for different values of Γ the relative acceleration of the tray.

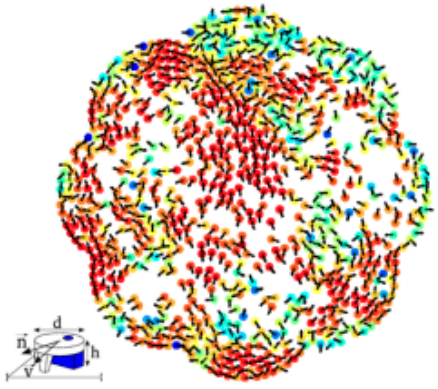


Figure 12 – Experimental observation of collective motion in the system of self-propelled hard disks with $N = 890$ particles. Color codes for orientation, reflective of collective motion.

In this system, binary collisions provoke the alignment of both orientations [54]. This local phenomenon is the root of the large-scale collective effects happening in experiments ran with hundreds of disks. Figure 12 shows a snapshot of an experiment with $N = 890$ disks for a surface fraction of $\phi \approx 0.38$, color-coded by orientation. The system spontaneously achieve large-scale collective motion.

Similar to the vibrating disks, some system embark the vibration and the power source, creating a type of active particle often called bristle-bots. These systems are typically subject to the same kind of orientation response as the vibrating disks [55] [56]. Figure 13 shows examples of bristle-bots of different nature. These robots are easily built at home, with a surprisingly large amount of DIY tutorial for all sizes and shapes found on the internet. Example of a clean DIY realization [57] is shown on Figure 13.a . One interesting perspective to this kind of active unit is the miniaturization, with some bots measuring less than 2mm in size and weighting less than 5mg shown on Figure 13.b [58]. At this scale, the main engineering challenge becomes the power source, absent of the robot in the picture and paper presented here. Some experimental realizations of bristle-bots are used to study collective effects, reminiscent of the work presented above [55]. Finally, one very common type of bristle bot which can be bought directly as a simple robotic toy is the Hex-bug, shown to behave in the same way as other polar active units like the vibrating disks [56].

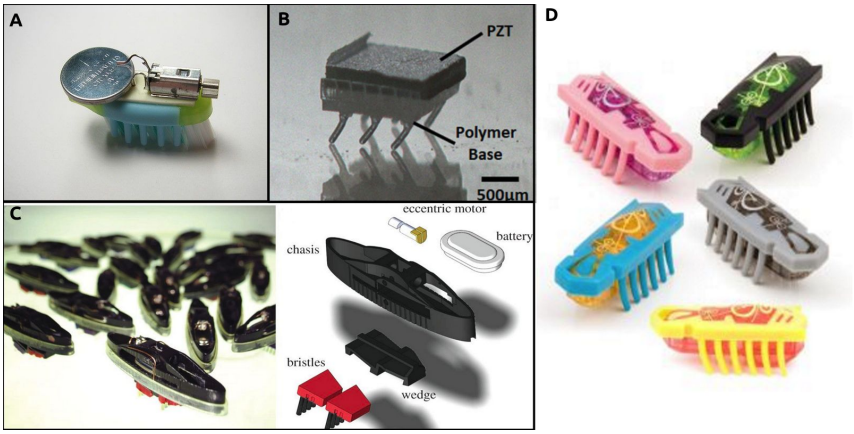


Figure 13 – Examples of vibrating polar robots, bristle-bots. **A**. DIY bristle-bot made out of a vibration motor and a small battery glued to a toothbrush head [57] **B**. Micro bristle-bots of less than 2mm in size, fabricated by two-photon lithography [58] **C**. Bristle bots used to study collective phenomena, with a physical model including a reorientation torque similar to the one used to describe the self-propelled hard disks [55] **D**. Collection of 5 Hexbugs, a commercial realization of bristle-bots commercialized as simple robotic toys [59].

1.3 Morphological computation at the interface of Soft Matter and Robotics

Many different aspects come to the design of a robot : its morphology (shape, structure, materials), sensory apparatus, motor system, control architecture, etc. [60]. All of these aspects interact and jointly determine the robot's behavior. In particular, when a real robot interact with its environment, it gets information from its sensors and sends out control signal to its actuators. The actual movement which is then performed solely depends on the laws of physics. By designing the morphology of the robot to obey certain physical properties, one can alleviate the need for computation to achieve a desired behavior. The interesting incentive here is the capacity to improve performances compared to classical robots, as the laws of physics « work » in favor of the robot and assists in succeeding in the task, thereby performing *Morphological Computation*. Doing so puts systems at the interface of robotics and robot-scale physics, mainly soft matter. We present in this section examples of inter-disciplinary work inspiring the methods and reasoning applied in this thesis.

1.3.1 Granular gripper

A canonical example of why thinking about the morphology of the robots as a key feature in its design can be used to greatly improve performances can be found in the case of grasping. The standard design approach when building a robotic gripper is based on a hand with two or more fingers and typically involves a combination of visual feedback and force sensing at the fingertips, as in the shadow hand – a dexterous hand with 20 degrees of freedom [61]. Grasping objects with such robots is highly complex and designing an effective and robust controller, on the software side, is a challenge.

By using a different approach and trying to solve the grasping task on the level of morphology, the Cornell universal "jamming" gripper [62] presented on Figure 14 narrows down the control task to a one degree-of-freedom problem. The gripper itself is based on a vacuum pump and an elastic membrane that encases granular material - a balloon filled with coffee ground. When the piston at the back of the gripper is not pushing in, the granular material flows and can be deformed to perfectly fit the shape of the object to grasp. At the moment the piston puts pressure on the granular material, it undergo a jamming transition and become solid-like, allowing to lift the object. Doing so, the gripper is able to grasp many kind of objects of different shapes and sizes without having to change anything in the output of the control algorithm : push or pull the piston. In swarm robotics, alleviating the need for costly computation becomes even more interesting, as the individual robots have very limited computational capabilities.

1.3.2 Systems of joint particle robots

In order to create more adaptive and fault tolerant robots, some systems assemble simple units into large components, pushing towards morphogenesis in swarms of robots. Morphogenesis is the process by which biological systems dynamically assemble into shapes and structures, as they col-

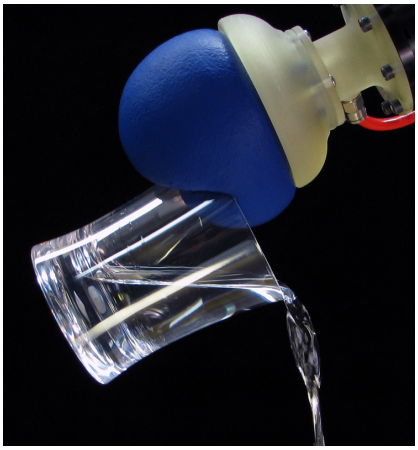


Figure 14 – The universal "jamming" gripper. This gripper uses the jamming transition property of granular material to switch between a soft and a rigid state, allowing to grasp multiple objects while narrowing the control task to a one degree of freedom problem. This robot illustrates the importance of thinking the robot as a whole, including its morphological properties, alleviating the need for complex computation. Credit : Brown et al. 2010 [62]

lectively functionalize in a self-organized and distributed manner [**?**] [63].

Many systems in collective robotics try to expand this process to swarms, building structures made of robots as simple as single actuated particles robots, as in the work of Li et al. 2019 [63]. In this system presented on Figure 15, particle robots can only expand or contract, ranging from a diameter of 15.5cm up to 23.5cm. The bodies are designed to stick to one another, loosely enough so that bonds can create and break dynamically. Each robot is programmed to cycle between the expanded and contracted phase, with an offset chosen based on the communication with close peers (broadcast only). When a robot is alone in space, the repeated contractions makes it move randomly, and cluster with nearby robots. Experiments show that agglomerates are able to perform collective phototaxis when particles oscillate in a phase proportional to their level of illumination. This makes the agglomerate moves in a coordinated fashion towards the light source, reorienting dynamically the light is moved and able to cross obstacles along the way.

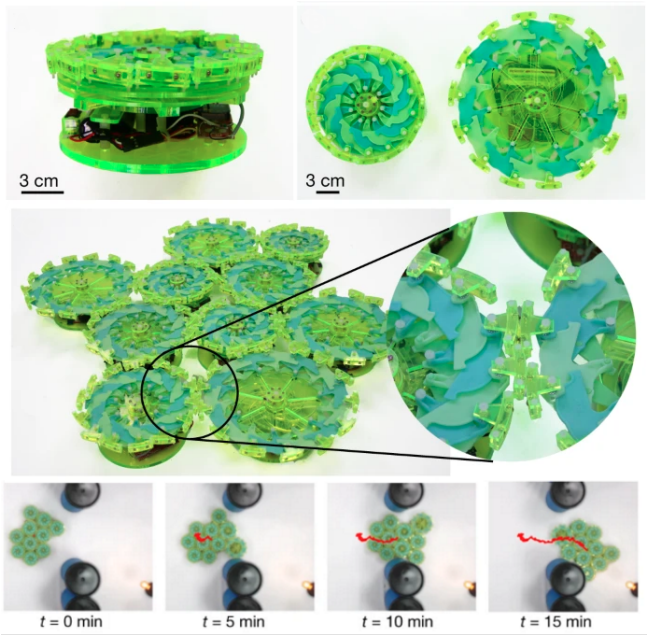


Figure 15 – Particle robots with a single actuator. When programmed to behave in systematic patterns, a group of robots can perform coordinated motion and phototaxis even through obstacles. Credit : Li et al. 2019 [63]

Other systems perform morphogenesis in swarms of robots not physically connected, but free to move in space and bound only by the sensing of close robots and their programmed behavior. In the work of Slavkov et al. 2018 [**?**], an algorithm based on reaction-diffusion equations implemented on Kilobots demonstrate morphogenesis in a swarm of 300 robots, shown on Figure 16. The working principle is to consider two virtual molecules obeying a set of differential equations corresponding to a typical system of reaction-diffusion, which are dynamical systems known for displaying organic patterns. One the transmission is done and virtual patterns are formed, groups of robots on the edge of a cluster end in a state different from the robots on the inside. They form multiple "Turing spots", and robots on the edge but not part of such a group move along until they hit a turing spot, forming protrusions extending in space. This process also create a dynamic response to damage, with protrusions able to regrow if a Turing spot has not been cut off completely.

Finallly some system try to enhance already available robots, by physically linking them together. One example is found in Pratissoli et al. 2023 [**?**] in which Kilobot robots are attached together with spring, in a square lattice (Figure 17 left). With algorithms based on non-specific communication and distance measurement, this superstructure benefits from a great improvement on the reliability of its movement. The lattice is able to deform and follow reference trajectories, such as a circle, a task difficult to achieve with an individual Kilobot.

Another example in Boudet et al. 2021 consists in the encapsulation of Hexbugs robots in a membrane Figure 17 right. Contrary to Kilobots, Hexbugs are not capable of any kind of compu-

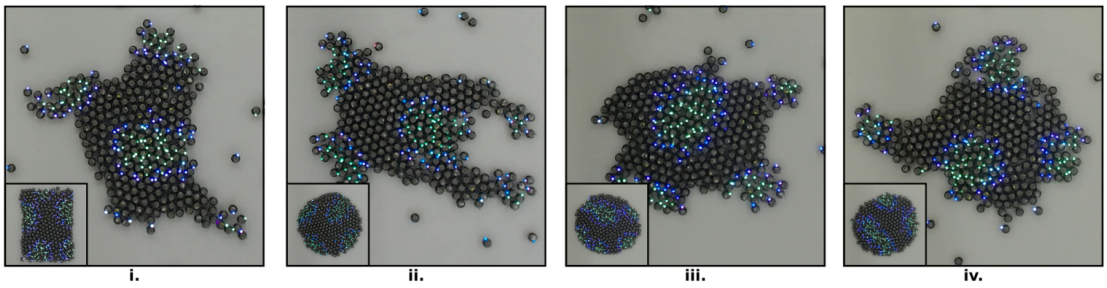


Figure 16 – Different runs for a process of morphogenesis in a swarm of 300 Kilobots based on virtual reaction-diffusion. The final shapes are shown with the Turing spots lit up, the starting shape is pictured on the bottom left of each example.[**?**]

tation. However, the simple fact of being mechanically coupled create collective effects. While the individual Hexbug moves at random, increasing the number of encapsulated Hexbugs diminishes the sum of total angular momentum of robots and a global direction ultimately "wins", carrying the badly oriented Hexbugs. The resulting superstructure is able to deform, transport loads and uniformly sweep the arena, while being composed of arguably the simplest polar active components.

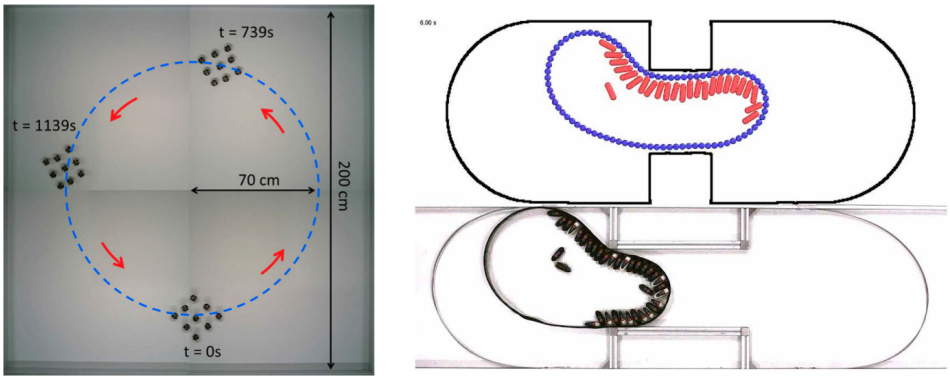


Figure 17 – **Left** Mechanically coupled Kilobots forming a soft superstructure able deform and follow a reference trajectory (here a circle), making the movement more reliable compared to individual components [**?**] **Right** Collections of Hexbugs in a membrane allow the creation of a more complex entity, a superstructure able to deform, transport loads and uniformly explore space [**?**]

The examples presented above show adaptive collective robotic systems able to perform tasks while minimizing memory and processing requirements at the individual level. This is done either on the basis of pure morphological properties, in the case of the Hexbug superstructure, or by a coupling between internal degrees of freedom (the software) and the morphology. In the general case, designing the appropriate program to leverage the morphological properties can be a hard task, and it becomes interesting to look in the automatic and adaptive design of software, particularly reinforcement learning methods. In the case of collective robotics adapting in real time to their environment, this translate to distributed and online learning. We turn toward the current state-of-the-art in learning in Swarm Robotics, and in polar active matter dynamics, which reflect the morphological properties that can be reasonably expected to arise for multiple polar active units in interaction such as robots.

2 State of the Art

This section present more precise mathematical and methodological concepts rooting the work done in the thesis. The goal is to provide the reader with a more in-depth understanding of the current research context in both learning in swarm robotics and dry polar active systems.

2.1 Learning in Swarm Robotics

2.1.1 About Machine Learning

Machine Learning algorithms are, in 2023, ubiquitous in our every day life. At the very beginning, algorithms are simply sequences of instructions. In every computation problem, an algorithm is devised to carry out the task solving the problem, from computing π to sorting lists to displaying windows on the screen. For some problems however, we do not have a fixed and precise set of instructions capable of efficiently carrying out the task. Common examples are problems such as filtering spam e-mails, verbally describing pictures or selecting relevant content to show for an ambiguous request on the internet. These are the problem solved by machine learning, in which « What we lack in knowledge, we make up for in data. » [64].

Three different types of learning are usually considered : supervised, unsupervised and reinforcement. We briefly present the first two categories and detail further the case of reinforcement learning, as it is the most prominent type of learning in robots and the broad framework in which to position the work done in this thesis. Both supervised and unsupervised learning methods can be used in robotics, but as it is costly in time and efforts to generate data when manipulating a robot, reinforcement remains the main area of machine learning that deals with robots.

Supervised learning

This category regroups problems for which we can provide the algorithm with examples, a small amount of data validated by a human expert, in hope for the algorithm to generalize to large amounts of unseen data. Supervised learning regroups tasks of classification - choosing a correct *label* for a given input - and regression, which consists in approximating a function. Say we would like to predict whether a cat or a dog can be seen on an image, and we have a dataset of images and their associated label : *cat* or *dog*. This is the epitome of a supervised classification problem, in which an algorithm is trained on the dataset in order to label unseen images appropriately. Regression on the other side are tasks such as predicting air temperature and precipitation from meteorological data. In this case, there is no such thing as labels, it is the crude output values that have to be deduced from the set of input data. In a sense, regression consists in building a complex function, mapping input data to output data. Supervised learning demands large amounts of data to be collected, and sometime a huge prior work of sorting and labeling. For example, training a big language model such as GPT-3 [65], used in the development of the now famous ChatGPT bot,has required an internet crawl regrouping thousands of terabytes of text automatically extracted from the internet.

Unsupervised learning

Unsupervised learning problems have to do with the learning of categories, patterns, similarities and structure in a dataset. The tasks here are close to a classification task in supervised learning, they consist in **labeling** data. Say we have an unlabeled dataset of pictures of either cats or dogs, an unsupervised learning algorithm would be able to sort each picture into two categories (to us, cats or dogs) based purely on the discovered similarities between images. This type of learning is particularly well suited for the automatic detection of spurious data in measurements, as the algorithm learn what can be expected from a "normal" measurement.

2.1.2 Reinforcement Learning

Lastly, reinforcement learning (RL) regroups problems relying on the accomplishment of a goal, without having any working examples of solutions. The solution to the goal-based problem has to be discovered through trial and error. In this context, the developer only has to specify one or multiple "objectives functions" that have to be maximized by one or multiple agents using a reinforcement learning algorithm. Most of the time a single function called the reward (or its opposite, the cost) is used, rewarding agents behaving like what the developer considers to be a good behavior. Robotic control fall into this category, as well as games such as chess or go.

Most problems in RL are usually captured by a Markov-Decision Process (MDP [6]), consisting

6. Regarding multi-robots problem, an extension to this framework named Partially Observable Markov Decision Process - POMDP - is more often use. This extension captures the fact that a single robot can only partially observe

of a set (in a mathematical sense) of states S, a set of actions A, a probability of transition $P_a(s, s')$ from state $s \in S$ to the state $s' \in S$ under the action $a \in A$ and finally a real valued reward function $R : S \to \mathbb{R}$. This is represented in a very common schematic shown on Figure 18. In this framework, at any given timestep t a deterministic *agent* perform an action $a_t \in A$ and gets a reward r_t along with an updated state $s_t \in S$. Many different games and robotic systems can be translated into this framework. A simple example is shown along the schematic on Figure 18, the « cartpole » problem proposed by the gym environment for python [**?**]. This problem is an inverted pendulum problem, consisting in balancing upward a mass subject to gravity attached to a cartpole able to move back and forth along an axis. The action space is finite and has a size of 2, A = {Push to the left, Push to the right}, the set of states is on the other hand continuous S = Position × Velocity × Pole Angle × Pole angular velocity. The reward is given as +1 each timestep the pole is approximately up (±12°).

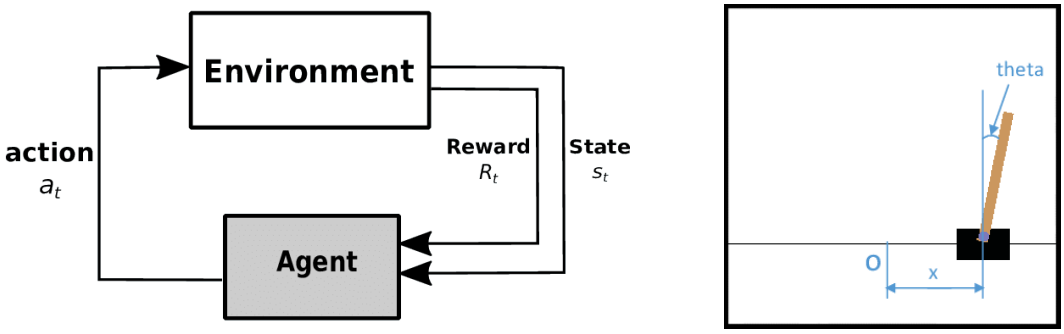


Figure 18 – **Left** Usual schematic representation of reinforcement learning. **Right** The simulated cartpole problem, a simple example of a Reinforcement Learning problem proposed in the gym environment for python [**?**].

The goal of a reinforcement learning algorithm is to learn a policy π, a function from the set of states to the set of actions $\pi : S \to A$, maximizing the reward. One key feature of the MDP formulation is that it provides an optimal policy criterion called the Bellman equation. For small discrete sets of actions and states, it is possible to evaluate from the agent perspective the expected reward of a given action in a given state, and to explicitly list the corresponding action to a state in a table. In history, this has lead to the Q-learning algorithm [67], which builds an evaluation of the expected reward to be obtain by taking a given action in a given state (the "Q function"). Building algorithms using properties of the Bellman equation constitute a class of algorithms called Temporal-Difference learning (TD learning).

When the sets of actions and states grows larger or become continuous, it is no longer possible to work with a table listing every possibility, and the policy becomes the result of a parameterized policy π_θ, a weighted combination of continuous functions. The goal for building an efficient policy is then to learn the optimal vector of parameters θ^* . Over time, it has become increasingly common to use Artificial Neural Networks as controllers with θ defining the weights and biases of each neuron. Neural Networks are typically chosen for they are good at approximating functions

the full state of the system at a time : its own state.

and can generalize well to new unseen inputs. Many types of neural networks exist with varying applications and degrees of complexity. The simplest form of neural network is called a Multi-Layer Perceptron (MLP), illustrated on the left in Figure 19. It is composed of chained components each performing the work of a smaller model called a Perceptron, Figure 19 right. The goal in this kind of model is to learn weights by which single inputs are weighted and summed before passing through an activation function and being sent to the next layer as an input. Many different activation functions and network topologies can be used, for the work done throughout this thesis it is not mandatory to have a deep understanding of this kind of model.

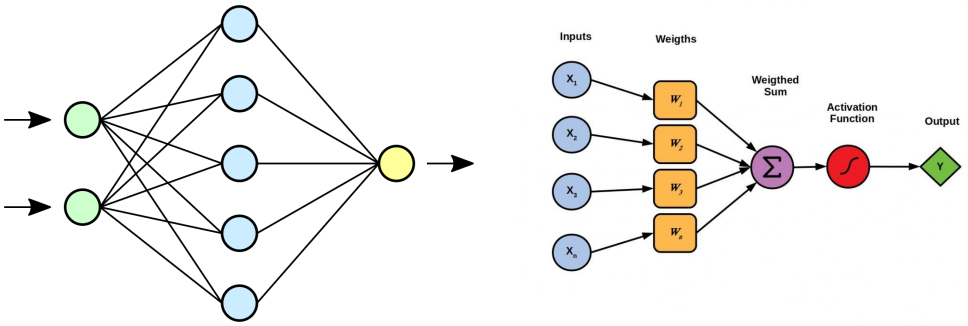


Figure 19 – **Left** Schematic of a simple neural network called Multi-layer Perceptron. Is is composed of two inputs, one hidden layer of 5 nodes and a single output. **Right** A single perceptron, consisting in a weighted sum of the inputs and a sigmoid activation function. The weights are learnt in order to match the output to a desired result.

When dealing with reinforcement problems of high complexity such as robotic control, the type of neural network in use are usually Deep Neural Network containing about five layers and several hundreds of neurons. Tackling learning with such architectures can follow the same principles as with the simpler reinforcement learning problems, such as learning a value function estimating the expected reward of states. This is a case for an actively developed type of algorithm called *actor-critic*, building upon the legacy of Q-Learning in complex control problems [**?**].

Because building value functions and estimations of the environment's response is computationally costly and usually builds up a single policy, another approach to reinforcement learning problems is to draw parameters mostly « at random » and trying them out directly. This is called *Direct Policy Search* and encompass among other types evolutionary algorithms. A key feature of evolutionary algorithms is the capacity to find a broad variety of solutions for a given task with little knowledge on the problem or the platform [68]. These algorithms are even more interesting to deploy on swarm of robots, for they can be easily distributed [69].

2.1.3 Evolutionary algorithms

As stated previously, evolutionary algorithms denote a family of *Direct Policy Search* algorithms inspired from the principles of natural evolution : selection, variation, survival of the fittest. More than often vocabulary is also shared with biology, with the parameters θ denoted as the *genotype* and the resulting policy π_θ denoted as the *phenotype*. The generic framework of evolutionary algorithms is presented on Figure 20. By letting the robot interact with its environment for a sufficient amount of time or until a termination condition is reached (an *episode*), the robot is evaluated not on the reward obtained on specific actions / states but with the sum total of rewards over the course of the episode. This generates a *fitness* value, an evaluation of the global quality of the parameters. This fitness value is then used to select individuals to seed the next generation. Children parameters are created by stochastic variations of the parents parameters, and the evaluation–selection–variation cycle is repeated.

One consequence of this design for evolutionary algorithms is their «black-box» nature. As long as the problem can be formulated with a function *parameters* $\rightarrow$ *fitness*, an evolutionary algorithm can be plugged onto it to solve it, no matter the specifics of the problem. This includes other fields than reinforcement learning, although they may not be the most suited algorithms. One example being convex optimization problems in which a gradient descent would out-perform evolution both in speed and accuracy. However in the case of global optimization problems from which it is difficult to extract regularities (non-convex, noisy, multi modal), evolution becomes well adapted [**?**].

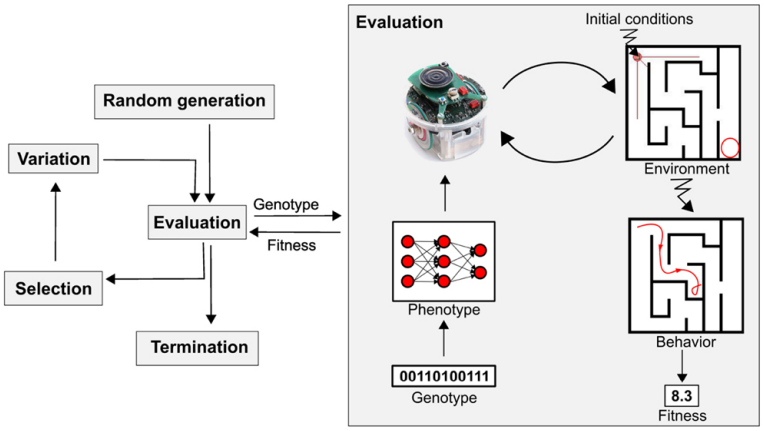


Figure 20 – Generic framework of evolutionary algorithms. Robot behavior is evaluated through a user defined fitness function that depends on the goal. After evaluation, the genotype is updated using general principles inherited from natural evolution. Credit : Doncieux et al. 2015 [70].

Mutations and recombination are two types of variation that differ in the number of parent parameters, with a single parent for the mutation and two for the recombination. Mutations are ubiquitous in evolutionary algorithms, while recombinations are sometimes put aside. For real valued parameters, mutations consist in drawing from a probability distribution in the parameter

space and adding the result to the parent parameters to generate an offspring. The most common distributions are uniform, or normal to obtain unbounded mutation steps. The Cauchy distribution is also considered instead of the normal distribution because of its fat-tail nature, allowing to perform very distant mutation steps. As the solutions approach an optimum, it can be useful to decrease the variance of the mutation operators, to speed up the convergence process. This can be done by adding an hyper-parameter to the model that sets the decrease rate in variance, or another approach is to adapt on the fly the variance, which is the basis of the CMA-ES (Covariance Matrix Adaptation Evolution Strategy) algorithm, a well-known evolution algorithm that became a standard in the community [**?**] for continuous domain optimization.

In line with the automatic adaptation of mutations, adapting the search space of the controller is also possible through the black-box property of evolution. This has lead to algorithms that evolve neural networks not only through the parameters but also through the topology of the network, creating and removing nodes as part of the evolution process [**?**]. This additional layer of neuroevolution opens perspectives in the hybridization between deep reinforcement learning and evolutionary algorithm, all the while pushing for «open-endedness» in learning, that is an unbounded learning process with an agent that can always out-perform itself [**?**].

Evolutionary algorithms also push toward open-endedness with methods based on discovering new behaviors. It has been shown that searching for novelty (meaning unseen behaviors) instead of explicitly maximizing reward is a good alternative that allows to escape local optima and quickly discover working solutions [**?**]. Combining the search of novelty and reward maximization leads to a sub-family of evolutionary algorithm called Quality-Diversity [**?**], with solutions selected based on multiple criteria, allowing to generate at once a diverse set of working solutions to a problem.

One amusing consequence of this search of novelty and the progress in reinforcement learning in general is the fact that algorithms get good enough that they tend to exploit the flaws of the simulator or the flaws of the reward function. Many examples exist in the literature of agents breaking the laws of physics in simulations to accomplish their goals [66], one example is given in Appendix B.

The two key features of evolutionary algorithms, the «black-box» property and the capacity to be highly parallelized, makes this type of algorithm a viable option for the deployment in swarms of robots. This is called Embodied Evolution, or Social Learning.

2.1.4 Embodied evolution in swarms

In our case, we deploy evolutionary algorithms on simple robots designed for swarm robotics. This narrows down the broad context of Evolutionary Algorithms to a case of both **online** and **distributed** learning. For our experiments, robots are performing **online** learning in a sense that they evaluate the performance of their genotype at the same time as they interact with the environment. This adds a timescale to the evaluation of a given genotype, as the robot cannot integrate its reward over the course of the whole experiment to obtain a fitness value. The robots must compute its

reward during a set amount a time, the evaluation period, that needs to be long enough to capture the real value of a genotype but short enough so that multiple variations and selections can occur for the duration of an experiment. In addition to this timescale, having a **distributed** algorithm also adds noise to the evaluation of genotypes. As the evaluation takes place, robots may collide, changing the course of their trajectories, or they may get stuck one against another, worsening the evaluation of a genotype. Finally, the simplicity of the robots add constraints to the algorithm not present in other evolutionary robotics contexts : limited memory, limited computation capabilities and limited communication bandwidth. In the literature, running a decentralized evolutionary algorithm in a swarm to perform online learning is called *Embodied Evolution* [72].

2.1 Learning in Swarm Robotics

To date, several distributed and online algorithms have been deployed on real robots, examples of which can be found in reviews such as Bredeche et al. 2018 [72]. However, the number of robots is often in the order of ten(s) due to the hardware and CPU requirements of such robots to implement learning algorithms. This stays relatively low compared to condition based swarm experiments or learning simulations.

As in reinforcement learning problems, one of the natural goal for an embodied evolution algorithm is to maximize the total reward. The basis principle for evolution in robotic lies in assessing accurately the quality of the swarm behavior and adapting on the fly. Since communications are local and the environment is partially observed by single robots, this is done at the individual level by agents, with trial and errors of policies and computation of their own local reward. In parallel with this act-sense and assess loop, robots share their policies with neighbors and actualize their behaviors with a goal of maximizing global reward.

In a 2019 research article by Jones et al. [75], 9 enhanced e-pucks robots (with greater computation capabilities) where deployed on a task of collective transport, collectively pushing a disc on the sides of a square arena, illustrated on Figure 21. This is done through an evolved behavior tree, an adaptive condition-based controller in which conditions are tuned, added or removed in the evolution process. In this experiment, evolution is hybridized online and offline, with each robot carrying a virtual simulator of the task performed in real life. Every minute, robots run simulations of 256 virtual agents and select the best parameters based on total reward maximization. The real robot then adopt this set of parameters and broadcasts it to every close robots. Once a set of parameter is received, it is added to the virtual population and evaluated subsequently. Reward is based on the proximity to walls of the disc to be pushed, projected on the X-axis only. The increased in total reward over time shows a clear example of embodied evolution, in which robots automatically learned a collective behavior of object transport.

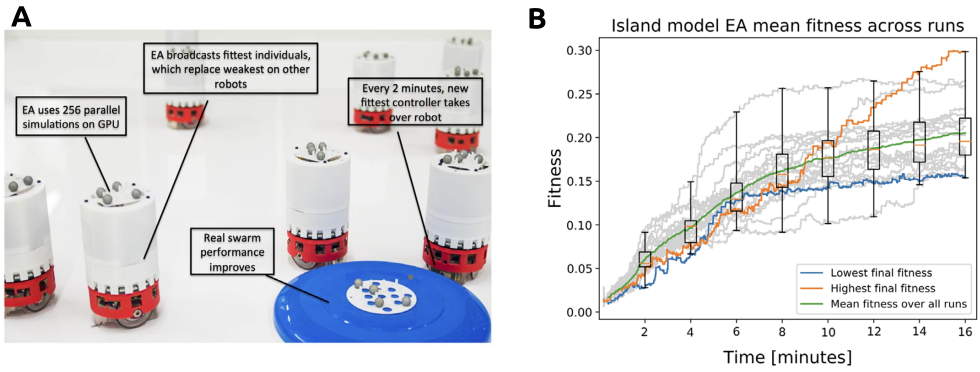


Figure 21 – **A**. Enhanced e-pucks robots along with a blue disc to be pushed in a collective transport experiment. **B**. Real experiment fitness over time, for 20 independent experiments. The swarm manage to learn from the on-board evolution and parameter exchange between individual robots. Credit : Jones et al. 2019 [75]

In addition to standard reinforcement learning concerns, each robot do not have access to the total reward of the swarm but only its own, time-dependent reward. This leads to interesting propagation dynamics for the parameters. Policies are spread by two parallel effects, with robots gaining a high reward spreading their parameters to robots having a lower reward, and policies well-suited for robot encounter propagating due to a larger amount of mating opportunities [74]. From the perspective of a single agent, this can be seen as an *extrinsic* motivation to maximize its reward, distributed by the interactions with the environment, and an *intrinsic* motivation of propagating its parameters among the other agents to survive. In order to study this effect of selection through propagation, a minimal evolutionary algorithm was designed in Bredeche et al. 2012 [74], presented on Figure 22.

In this algorithm, each agent moves in the environment according to its current policy for a fixed duration, controlled by a set of parameters called the genotype. While moving, it continuously broadcasts its set of parameters, and record the sets of parameters broadcasted by other agents. At the end of the policy's lifetime, the agent draws at random one of the received policy, with a variation operator allowing for novelty (gaussian mutation). A selective pressure coming from the environment can be implemented in foraging tasks with a system of *energy* gathered by the agents, which come to a stop if their energy fall below 0. Apart from this effect, the spreading of parameters in the swarm only happens through random encounters.

Algorithm 1 The MEDEA algorithm

```
genome.randomInitialize()
while forever do
  if genome.notEmpty() then
    agent.load(genome)
  end if
  for iteration = 0 to lifetime do
    if agent.energy > 0 and genome.notEmpty() then
      agent.move()
      broadcast(genome)
    end if
  end for
  genome.empty()
  if genomeList.size > 0 then
    genome = applyVariation(select_random(genomeList))
  end if
  genomeList.empty()
end while
```

Figure 22 – Outline of the mEDEA algorithm, for minimal environment-driven distributed evolutionary adaptation. This algorithm include selective pressure only in foraging tasks through energy, with agents low on energy halting.

A real-robot experiment with 20 e-pucks programmed with the mEDEA algorithm is performed to validate the algorithm and study the indirect effects of the environment on the policy propagation. Robots are placed in a 280x230cm arena and can communicate with neighbors. A point of interest is placed in the environment in the form of an object localized by every robot, the "sun". This object does not yield any reward or energy to the robots, but its position partake in the input of robots controllers. Certain policies drawn at random can therefore push the robot towards or

away from the "sun". An experiment last from 30 to 45 minutes with a policy lifetime of ≈ 1min and 10s. At the end of an experiment, robots tend to aggregate toward the sun despite not being explicitly pushed to do so through a reward. This is due to policies promoting aggregation having more encounters than policies promoting dispersion. This effects not only shows in the spatial distribution of the robots but also in their policies, showing an adaptive behavior when the sun is moved through the arena.

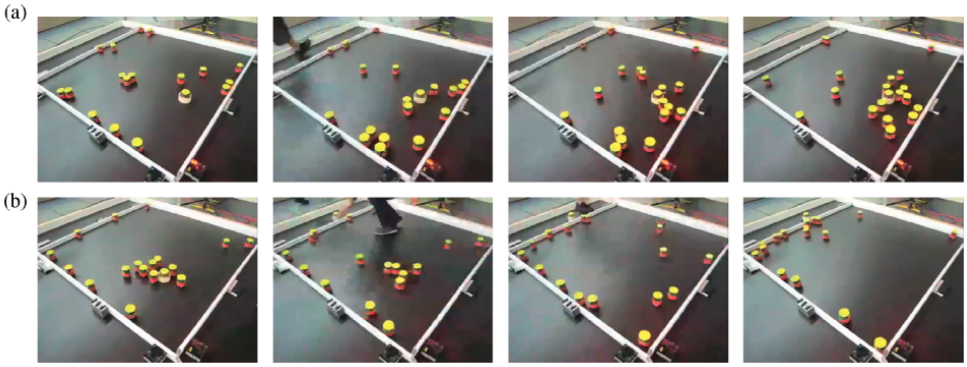


Figure 23 – The mEDEA algorithm running in a system of 20 epucks robots. (**a**) Shows the time evolution of the system with a fixed aggregation spot (sun) (**b**) Shows the system reaction to a change of location of the sun, with the convergence of the robots to the new location, although no reward is distributed for robots close to the sun. Credit : Bredeche et al. 2012 [74]

2.1.5 HIT Algorithm

The HIT algorithm is another algorithm designed for embodied evolution and has the great advantage of being simple in its structure, computation needs and memory needs. It boils down evolution to its main components, making it flexible to different settings and allowing us to envision macroscopic models of embodied evolution in swarms of robots. In the work done during this thesis, we use a version of this algorithm specialized for phototaxis, fully presented in Chapter 3.3. We present in the following the scientific work done with this algorithm.

The HIT algorithm has been experimentally tested, both in simulation [**?**] and on real robots [**?**]. On a simulation involving 150 robots performing a foraging task in Bredeche et al. 2022 [**?**], the HIT algorithm was shown to enable social learning and optimize the efficiency of the swarm as a whole. A screenshot of the simulation and performance achieved over time for the foraging task are shown on Figure 24.

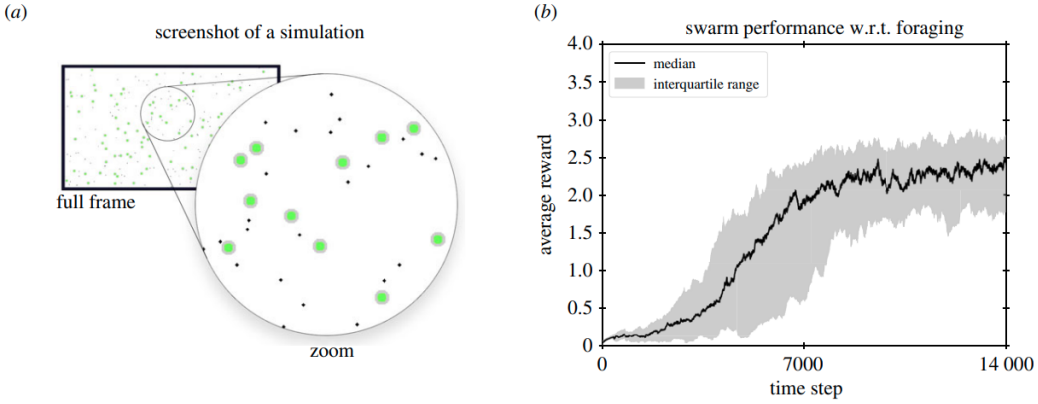


Figure 24 – (**A**) Screenshot of the simulations performed in Bredeche et al. 2022 [**?**], in which a swarm of 150 robots learn to perform a foraging task using the HIT algorithm. (**B**) Average reward of 32 independent simulations for the foraging task using the HIT algorithm. Parameters are initially distributed at random in the swarm. As the experiment progresses, robots exchange the most successful parameters, with a low probability of mutation at each transfer. Over time, the mutations and exchanges lead to better policies spreading in the swarm and the total reward augments.

In this experiment, robots are small disks of diameter 5 in a rectangular arena of 1000 x 500 units, and must capture 100 objects of diameter 20. Each robot possesses 3 sets of 16 sensors distributed uniformly around it. All sensors have a fixed range below which they can detect objects, with one set measuring the distance to walls, one set the distance to other robots and one the distance to objects to capture. The linear and angular velocity of the robots are piloted by a unique layer of a Perceptron, with 48 inputs in total. Finally, the reward is computed as the number of object captured in a fixed time. The results show the average of 32 independent runs. At the beginning,

robots move at random and manage to capture objects or encounter other robots by chance. When it happens, policies that are slightly better than random propagate in the swarm and a mutation event occurs with fixed probability at each transfer. Doing so, the slightly better policies improve over time and diffuse to all robots. As the quality of the foraging augments, the diversity of policies diminishes as most of the initial policies which fail to perform a good foraging disappear.

In addition, the same experiment is ran while limiting the amount of parameters transferred during an encounter (denoted as tsf in Figure 25) to 80% of the total parameters. This mimics the case in which real robots cannot transfer all of the parameters at once. This can happen because of a limited communication bandwidth due to hardware limitation, packet loss or disturbances due to the environment. In this case, the transfer also acts as a source of innovation by recombination of the different sets of parameters. The results show that this limited transfer (Figure 25-a) mainly affects the dynamics of learning by slowing down the convergence to good policies, but does not affect the final score of the swarm compared to the full transfer (Figure 25-b).

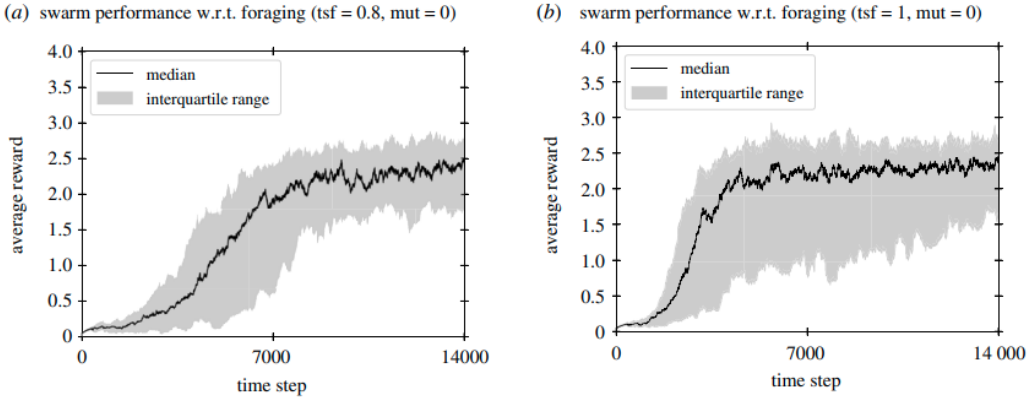


Figure 25 – (**A**) Average reward over 32 independent runs for the foraging task in simulation using the HIT algorithm, with 80% of the parameters transferred during an exchange. The limited transfer slows down the discovery and propagation of the best policies but is not detrimental to the final average performance of the swarm. (**B**) Average reward with a full transfer of the policy's parameters. Credit : Bredeche et al. 2022 [**?**]

This algorithm has also been experimentally tested on real robots in Mirhosseini et al. 2022 [**?**]. The article presents multiple experiments, as well as simulation of up to 60 robots implementing the HIT algorithm and implementing force-orientation responses similar to the one observed with the Kilobots. One experiment employing the HIT algorithm is made using 7 and 6 Mona robots [32] in an experimental setup presented on Figure 26.left. The Mona robots are low-cost, open-source and open-hardware mobile robots used in research and education. The Mona robot is a circular robot of 8cm in diameter, equipped with 2 wheels, 5 infrared emitter receiver placed on the front and has similar computing capabilities as the Kilobot. The data is acquired through a camera placed on top of the arena, following a similar procedure as in the experiments we run on Kilobots. In the experiment, the robots perform phototaxis with a light source coming from the side of the arena.

The controller consist in a binary decision between two meta-policies, driven by two parameters exchanged through the HIT algorithm. Results on Figure 26.right show a successful learning in the multi-robot system over 3 independent runs, for a semi-circular lit region located at the border of the arena ("border-light" setup in the article).

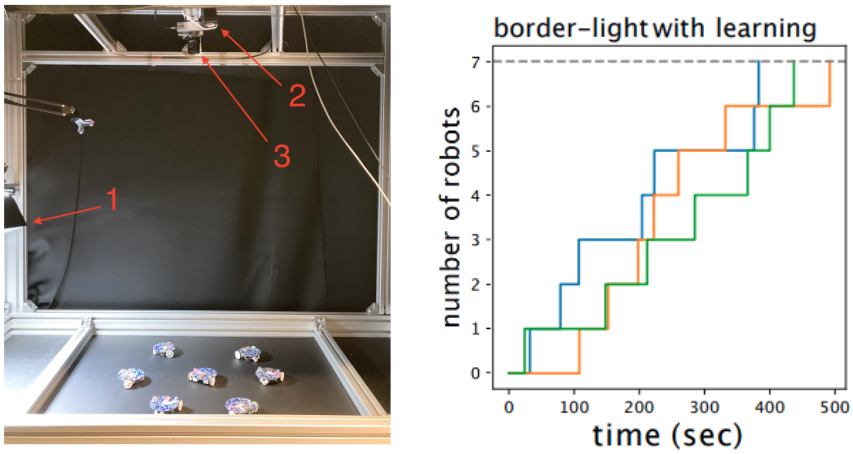


Figure 26 – **Left** Experimental setup in use in Mirhosseini et al. 2022 [**?**]. Mona robots perform phototaxis with a source of light **1** coming from the side. Other experiments use a different source of light using a projector **2** placed on top. Data is acquired through a camera **3** filming the arena. **Right** Amount of Mona robots in the light for a phototaxis experiment using the HIT algorithm, over 3 independent runs, with a lit region at the border of the arena. The HIT algorithm successfully propagates the good policy in this multi-robot system, enabling collective phototaxis.

2.2 Dynamics of polar active matter

2.2.1 Microscopic models of Active Matter

In order to understand the underlying rules governing the behavior of active systems, physicists propose and study different kind of models describing active agents and their interactions. Two levels of description of active systems can be considered : microscopic models, describing the evolution in time of individual agents, or macroscopic models describing whole systems with continuous fields, such as density, velocity or orientation at any given point in space. As mentioned in the first Chapter, among the many types of active matter we focus on dry and polar active systems.

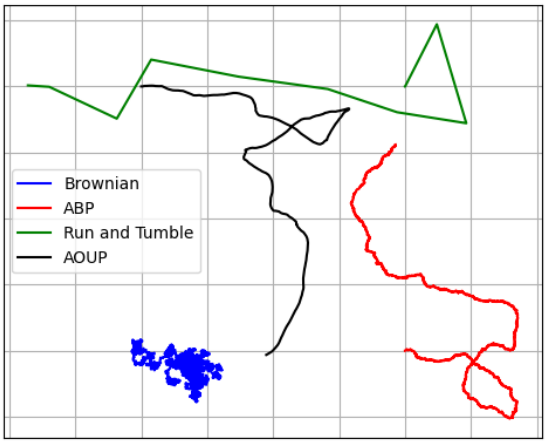


Figure 27 – Examples of trajectories generated with four different motions : Brownian motion in blue, Active Brownian Particles (ABP) in red, Run and Tumble in green and Active Ornstein-Uhlenbeck Particles (AOUP) in black.

Active Brownian Particles

In this category, the most studied microscopic model for active particles is the model of Active Brownian Particles (ABP) [76]. Although this model can be used to describe systems in 3 dimensions, it has mostly been studied in 2D [**?**]. We consider an active particle represented by its position $\mathbf{r} = \begin{pmatrix} x \\ y \end{pmatrix}$ and its orientation $\mathbf{n} = \begin{pmatrix} \cos\theta \\ \sin\theta \end{pmatrix}$.

$$\begin{cases} \dot{\mathbf{r}} = v_0\mathbf{n} + \sqrt{2D}\boldsymbol{\xi}(t) + \mathbf{F}_{int} \\ \dot{\theta} = \sqrt{2D_r}\xi_r(t) \end{cases}$$

Where $\boldsymbol{\xi}$ and ξ_r denote zero average, delta correlated noise [7] of variance 1 (2D for $\boldsymbol{\xi}$ and scalar for ξ_r). Here, D and D_r, the diffusion coefficients, control the amplitude of the noise in translation

7. Delta correlated means that considering the noise to be random variables indexed by time or particle index, the covariance between any two distinct variables i and j is 1 if and only if $i = j$ and is 0 otherwise. This is the Kronecker delta of i and j, $< \xi_i, \xi_j >= \delta_{ij}$, hence the name. For a stochastic process, this delta correlation means that the process has no memory.

and rotation. In the literature these coefficients are re-scaled as $\sqrt{2D}$ for mathematical convenience. Finally, $\mathbf{F}_{int}$ denotes the interaction forces between particles.

Breaking down the different terms, this makes particles move at a constant speed v_0 along the direction of their orientation $\mathbf{n}$, all the while being subject to noise in translation (like passive brownian particles) of intensity D with the term $\sqrt{2D}\xi(t)$. The orientation of the particle given by the angle θ is also subject to noise, with the term $\sqrt{2D_r}\xi_r(t)$ driven by the intensity D_r. Example of a resulting trajectory is given at the beginning of this section Figure 27. When two particles get close they collide and repel each other, this is implemented as soft collisions through a distance dependent repulsive force given by $\mathbf{F}_{int}$.

The total interaction force is usually considered by summing pair interactions between all close pairs of particles. This is done via an interaction potential V_{int}, a function of inter-particle distance, which gradient yields a force. For a particle i interacting with close particles indexed by j, this gives the formulation $\mathbf{F}^i_{int} = \sum_{j\neq i} -\nabla V_{int}(d^{ij})$. When running molecular dynamic simulation in general, one of the most common interaction potential used is the Lennard-Jones potential, attractive at a long distance and repulsive on short distances. When simulating discs that do not attract, a truncated and shifted version is used to keep only the repulsive part of this potential, called the WCA potential (for Weeks-Chandler-Andersen). This interaction potential is preferred for ABPs and other active particles as they do not represent actual atoms or molecules but generic active agents, that only interact through repulsion. The two potentials are drawn on Figure 28, considering particles of diameter $\sigma = 1$ and typical exponents for the "stiffness" of the potentials, $n = 6$. The highlighted green part denotes the distance under which particles start to repel each other, set at $d_c = 2^{1/n} \approx 1.122$ in the case of $n = 6$. As the exponents grow large, the potential gets sharper and d_c gets closer to σ.

Active Ornstein-Uhlenbeck Particles

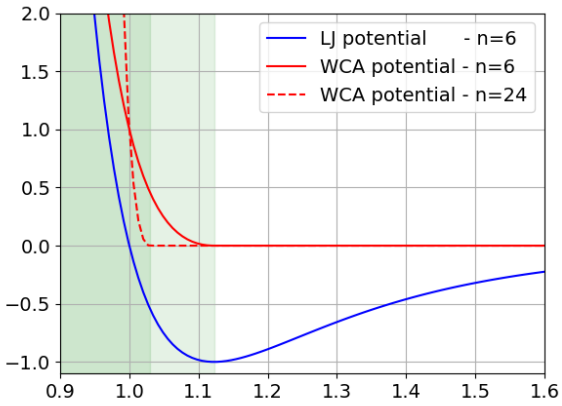
A second model close to ABPs sometimes found for active polar particles is the model of Active Ornstein-Uhlenbeck Particles (AOUPs) [**?**]. This model do not consider the orientation as an additional degree of freedom distinct from the orientation of the velocity, and considers particles "accelerated at random". This yields simple equations of motion :

$$\begin{cases} \dot{\mathbf{r}} = \mathbf{v} + \mathbf{F}_{int} \\ \dot{\mathbf{v}} = \sqrt{2D}\boldsymbol{\xi}(t) - \mathbf{v} \end{cases}$$

Now the velocity module is not constant, creating particles similar in the motion to ABPs but able to accelerate and take sharper turns. At large scales, this model is equivalent to ABPs and Run and Tumble particles [**?**], up to technical details.

Run and Tumble

The Run and Tumble motion is found in nature, typically performed by bacteria [**?**] allowing to explore space better than the straight motion or purely diffusive motion. Following this motion,



$$LJ(d) = 4\Big(\big(\tfrac{\sigma}{d}\big)^{2n} - \big(\tfrac{\sigma}{d}\big)^{n}\Big)$$

$$WCA(d) = \begin{cases} LJ(d) + 1 & \text{if } d \leq d_c \\ 0 & \text{otherwise} \end{cases}$$

Figure 28 – Illustration and typical formulation of the Lennard-Jones potential, as well as the truncated and shifted version, the Weeks-Chandler-Andersen potential. Both potentials define the strength of pair interactions through their derivative, with LJ being repulsive at short range (in green) and attractive at long range and WCA being purely repulsive.

a particle goes in a straight line at a velocity v_0 along its direction $\mathbf{n} = \begin{pmatrix} \cos\theta \\ \sin\theta \end{pmatrix}$. This is the *Run* phase. At each time-step, the particle has a chance to re-orient at random, in the *Tumble* phase. In practice, the tumble phase holds for a certain duration in which the particle spins. This may cause a correlation between the angle at which the particle exits tumbling and the angle at which it entered the tumbling phase. Real particles may also have a gyration radius, moving in space as they tumble. These effects are however not considered in theory.

Multiple formulations of Run and Tumble exist in the literature, with discrete or continuous orientations leading to slightly different dynamics and theoretical results [**?**]. A simple continuous formulation is to consider that the position $\mathbf{r} = \begin{pmatrix} x \\ y \end{pmatrix}$ obeys the following equation in discrete time :

$$\begin{cases} \dot{x} = v_0 \cos\theta \\ \dot{y} = v_0 \sin\theta \end{cases}$$

In addition to which, at each time-step the angle θ has a chance to be randomly uniformly drawn from $[0, 2\pi[$. Given an average run duration τ and a time-step duration δt, θ has a probability $\frac{\delta t}{\tau}$ to re-orient. This creates exponentially distributed run times of mean duration τ.

The core difference between Run and Tumble Particles (RTPs) and ABPs lies in the angular re-orientation, continuous in the case of ABPs and discrete in the case of RTPs. It has been shown

that as long as the speed does not depend on the orientation of particles, ABPs and RTPs are formally equivalent at large scale due to a mapping in the coarse-grained hydrodynamics equations [**?**]. Both motions are also capable of performing Motility-Induced Phase Separation, a phenomenon described in the following section. However, they differ in the case of single particles trapped in an harmonic potential, with RTPs able to explore the center more efficiently than ABPs which tend to get trapped in heights [**?**].

Other models

Other models directly consider density and velocity fields without a step of coarse-graining from microscopic equations. This is the case for the Toner-Tu model of dry polar flocks [**?**], but it is out of the scope of this thesis.

2.2.2 Motility-Induced Phase Separation

One of the most notable phenomenon occurring in self-propelled colloidal systems that is impossible to achieve with a passive system is liquid-gas phase separation with purely repulsive interactions. The mechanism leading to this phase transition is called Motility-Induced Phase Separation (MIPS), and occurs typically when the speed of active particles decreases rapidly with the increase in local density [77]. This phenomenon is observed in experimental systems of self-propelled colloids [**?**] [**?**], and has been shown to occur for the different models of ABPs, Run and Tumble particles and AOUPs, using simulations of the microscopic models and a theoretical analysis of coarse-grained hydrodynamics equations [77] [**?**].

Figure 29.**a** provides an example of MIPS realization in an experimental system of Janus particles. The experimental systems consist in thousands of $4.28\mu m$ silica particles half coated by titanium in an aqueous conductive solution, between two electrodes. The particles form a monolayer of density $\phi \approx 0.25$ on the bottom electrode. By applying tension with a square wave between the electrodes, the particles self-propel due to an effect called induced-charge electrophoresis [**?**]. Once activity starts, clusters start to form and merge over the course of a minute, providing a clear realization of MIPS. The typical cluster size in this system does not grow with a constant rate during an experiment, indicating the existence of different clustering phases, detailed in Van der Linden 2019 [**?**].

Figure 29.**b** and **c** provide examples of phase diagrams for coarse-grained equations producing MIPS. These phase diagrams are associated to the specific hydrodynamics equations presented in Cates et al. 2015 [77], but they give insights on the microscopic parameters acting on MIPS and typical ranges in which MIPS can be observed. Assuming the velocity v depends on the local density ρ up to a critical density ρ^* (above which nothing moves) according to $v(\rho) = v_0(1 - \rho/\rho^*)$, MIPS spontaneously occur for $\rho > 0.4$. The apparition of MIPS is helped by high speeds and low translational diffusion D_t. The binodal curves, in red, indicate the limit at which the phases can co-exist, but do not spontaneously separate. The separation can be triggered by nucleation sites or large enough fluctuations in density. The spinodal curves, in blue, indicate the limit at which

the phases spontaneously separate, triggered by small fluctuations happening naturally.

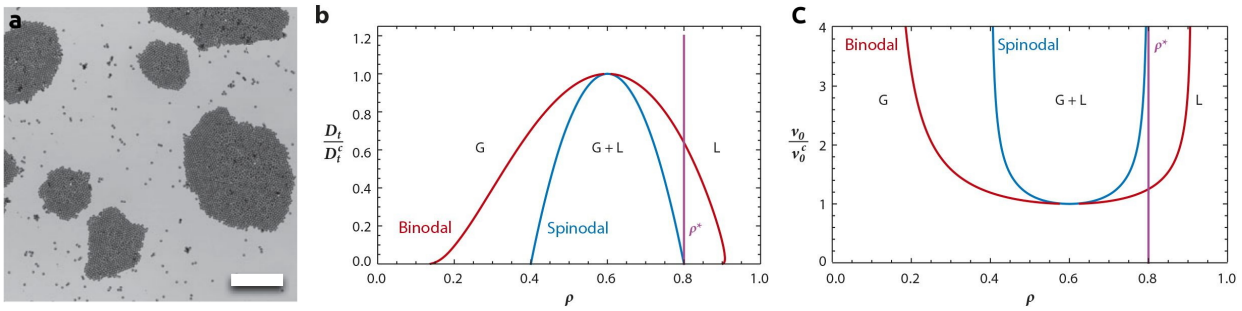


Figure 29 – **a**. Experimental example of MIPS in a system of self-propelled Janus colloids. [**?**] **b-c**.Phase diagrams of motility-induced phase separation for $v(\rho) = v_0(1 - \rho/rho^*)$, with (**b**) v_0 = 5 and (**c**) $D_t = 0.5$, and $\rho* = 0.8$ (vertical line). Red and blue lines are binodal and spinodal curves. Abbreviations: G, gas; L, liquid. Credit : [77]

2.2.3 Collective motion with simplified Active Matter : the Vicsek model

Based on the introduction to the Vicsek model by Ginelli F. 2016 [78]

Bio-inspired models describing the motion of social animals dates back multiple decades, with models such as the "Boids model" Reynolds introduced in 1987 to simulate the aggregate motion of flocks of birds, herds of land animals, and schools of fish within computer graphics applications [79]. In this model, as it is not a physics model, particle do not move on their own following classical equations of motion, but follow a combination of three motion rules, within a limited radius :

- Cohesion : steer towards the average position of neighbors.
- Separation : steer away local neighbors to avoid running into each other.
- Alignment : steer and accelerate to match the current speed and direction of neighbors.

Implementation of this model can be found online [80], in which the user can tune the intensity of the different rules.

The model of Tamas Vicsek in 1995 [81] follows a similar set of rules, focusing this time on the mathematical description of the system. In the Vicsek model, the swarm is modeled by a collection of self propelling particles that move with a constant speed and align with the average direction of motion of the particles in their local neighborhood. Considering particles in 2D moving at a speed v_0, with positions at time t $\mathbf{r}_i(t) = \begin{pmatrix} x_i \\ y_i \end{pmatrix}$ and orientations $\mathbf{n}_i(t) = \begin{pmatrix} \cos\theta_i \\ \sin\theta_i \end{pmatrix}$ the equations of motions in discrete time can be written as follow :

$$\begin{cases} \dot{\mathbf{r}}_i = v_0 \mathbf{n}_i \\ \theta_i(t+\delta t) = \langle \theta_j(t) \rangle_{|r_i - r_j| < r_0} + \eta \xi_i(t) \end{cases}$$

Where r_0 is the radius of interaction and ξ denotes a delta correlated white noise of variance 1, scaled by the amplitude η. The situation described by these equations is sketched on Figure 30.

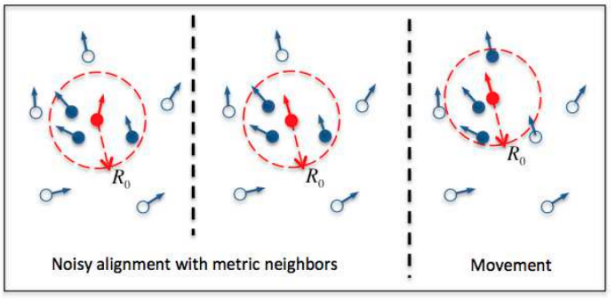


Figure 30 – Sketch of the Vicsek dynamics in action. The red particle's orientation is updated according to the mean orientation of the particles within the interaction radius r_0 and is moved forward at a velocity v_0, producing a change in neighbors. Credit : Ginelli F. 2016 [78]

The first convenient step to simulate and understand this model is to scale time and space with units defined by the system. This process, called the nondimensionalization of the equations, is ubiquitous in physics. Here, we chose r_0 as the unit of space and δt and the unit of time. This recasting of units transforms the other parameters in the equations, since their are now expressed in terms of r_0 and δt instead of meters and seconds. Now, the new velocity, call it $\tilde{v_0}$, is equal to $\frac{\delta t}{r_0} v_0$ and has no dimensions (η already had no dimensions and is thus not affected). Another common way of performing this operation in physics is to "set parameters equal to 1", here $r_0 = \delta t = 1$, ultimately meaning the same thing. For readability, we also omit the ˜ symbol over the new velocity and keep the notation v_0. This leaves us with 3 parameters in total for the model : η the noise amplitude, v_0 the velocity and $\rho = N/A$ the 2 dimensional density, where A is the area of the system (expressed in terms of r_0^2). For some parameter values polar order emerges and collective motion takes place. We define the polar order parameter as a mean of tracking the global state of the system relative to collective motion :

$$\phi(t) = \frac{1}{N} \Big| \sum_{i=1}^{N} \mathbf{n_i} \Big|$$

This quantity, defined as the length of the mean orientation vector and bounded between 0 and 1, indicates the presence of a shared orientation, hence collective motion. This order parameter is the analogous of magnetization in spins systems.

The Vicsek model is relatively simple to implement in simulation, and standard softwares in molecular dynamics simulation allow to simulate millions of particles simultaneously. The simulations show that a transition from disorder to ordered collective motion exist for some values of the parameters. For instance, decreasing the noise amplitude η below a certain threshold η_c synchronizes particles and generate collective motion starting from an initial disordered state. Figure 31 (A) shows this phenomenon for 3 different system sizes at fixed density and speed $\rho = 2$ and $v_0 = 0.5$. The order parameter is averaged over the last 2.10^7 time steps. The increase in system size sharpen the curve, indicating the discontinuous nature of the transition at $\eta = \eta_c$. To draw this plot, only the noise is considered as the *control parameter*.

In the Vicsek mode, the local fluctuations in density can lead to spatial heterogeneity in the ordered phase, in which ordered bands of particles flowing in a low-density sea of disordered particles can form and travel across the system. Two of these bands are shown on Figure 31 (B). These bands have a well-defined width and multiple bands can be observed in the same system. Even in three dimensions, these bands are present under the form of traveling sheets of ordered particles [82].

Additionally, there is no doubt that the critical value η_c depends on the values of the other parameters, ρ and v_0. In a first place, scaling arguments backed up with data confirmations allows to draw a sketch of the phase diagram in ρ and η at fixed v_0, shown on Figure 31 (C). Still, the most relevant parameter essentially governing the physics of this system is the noise amplitude η. Supplementary phase diagrams in v_0 can be found in Chaté et al. 2008 [82]. Working with experimental systems, the noise is a difficult parameter to tune, as it is rarely a willingly designed property of the system. The density however is a more convenient quantity to tune experimental

systems and reach polar order.

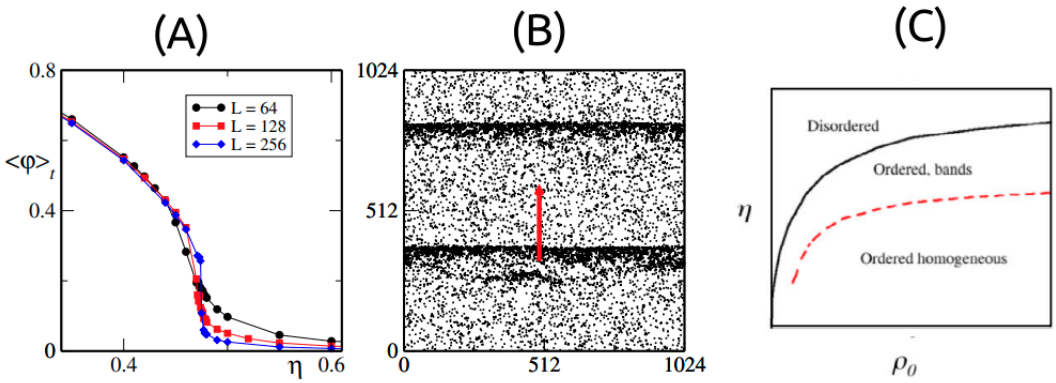


Figure 31 – (**A**) Polar order parameter as a function of noise intensity for simulated Vicsek systems of different box size L, $\rho = 2$, $v_0 = 0.5$ (**B**) Snapshot of ordered bands traveling in simulated Vicsek system (**C**) Sketch of a phase diagram for Vicsek particles, as a function of density and noise intensity. Credit : Chaté et al. 2008 [82]

The Vicsek model includes the basic features characterizing the behavior of aligning swarms. Therefore, the results obtained on this model such as conservation laws, asymptotic behaviors or phase transitions can be expected to show as well in specific systems, finer and more complex but sharing the same basic features of the Vicsek model.

2.2.4 Collective motion through self-alignment

Other types of microscopic models allow for the emergence of collective motion. We present in particular the model describing the dynamics of the vibrated polar disks [53] [54] [**?**] [**?**], a system of self-propelled particles presented in Chapter 1.2. This model considers particles in 2D represented by their positions $\mathbf{r}$, velocity $\mathbf{v}$ and orientation $\mathbf{n}$. Like with the ABPs, the orientation drives the velocity forward. Here, the velocity also influence the orientation, as $\mathbf{n}$ aligns toward $\mathbf{v}$, a phenomenon called self-alignment. In the case of the polar disks, this self-alignment term comes from the asymmetry in the friction experienced by the front and back leg.

As $\mathbf{v}$ is propelled forward via an active force in the $\mathbf{n}$ direction, $\mathbf{n}$ turns continuously to $\mathbf{v}$ via a torque proportional to the sinus of the angle between $\mathbf{n}$ and $\mathbf{v}$. A damping is added to $\mathbf{v}$ such that the particle reaches a stationary speed. Lastly, the dynamics on $\mathbf{n}$ is supposed over-damped. This means that we consider the friction on the angular velocity to be large compared to the moment of inertia, such that the angular velocity goes immediately to zero when no torque is applied. We obtain the following set of equations :

$$\begin{cases} m\dot{\mathbf{v}} = F_0\mathbf{n} - \gamma\mathbf{v} + \Sigma\tilde{\mathbf{f}} \\ \Omega\dot{\mathbf{n}} = \zeta(\mathbf{n}\times\mathbf{v})\times\mathbf{n} \end{cases} \tag{1}$$

Where $\mathbf{n}$ is a unit vector, $||\mathbf{n}|| = 1$. Each coefficient is defined positive. Let $v_0 = F_0/\gamma$ denotes the final speed of the isolated particle. We choose the mass m of the particles, their diameter d, and d/v_0, as units of mass, length and time. The above arguments are cast into the following dimensionless equations :

$$\begin{cases} \tau_v\dot{\mathbf{v}} = \mathbf{n} - \mathbf{v} + \Sigma\mathbf{f} \\ \tau_n\dot{\mathbf{n}} = (\mathbf{n}\times\mathbf{v})\times\mathbf{n} \end{cases} \tag{2}$$

with $\tau_v = \frac{mF_0}{\gamma^2 d}$, $\tau_n = \frac{\Omega}{d\zeta}$ and $\Sigma\mathbf{f} = \frac{\Sigma\tilde{\mathbf{f}}}{\gamma v_0}$. When simulating particles obeying these equations and subject to angular noise, we apply at each time-step of duration δt a rotation for $\mathbf{n}$ with an angle drawn from a normal distribution of variance $2D\delta t$. The noise of different particles are statistically independent, and we choose δt much smaller than the timescales in the dynamics,τ_v and τ_n. This is similar to the procedure applied in Lam et al. 2015 [**?**], providing theoretical work for this model.

Considering the control parameter $\alpha = \frac{\tau_n}{\tau_v}$, and considering the magnitude of the mean velocity as the order parameter for N particles $<\psi>= |\sum_{i=1}^{N}\mathbf{v}_i|/N$, simulations and theory show a transition from an isotropic state to a polar state (collective motion) driven by α, density ρ and noise D. Figure 32 shows on the left the order parameter of stable phases at zero noise, $D = 0$. A discontinuous transition between the isotropic state and polar state takes place at a value α^*. The figure on the right shows the phase diagram at finite $\rho = 0.01$, which depends on α and noise D, re-scaled by the scattering rate λ which itself depends on the density. At non-zero noise, an increase in α initially leads to polar order, but a further increase of α leads to a re-entrant transition towards the isotropic phase.

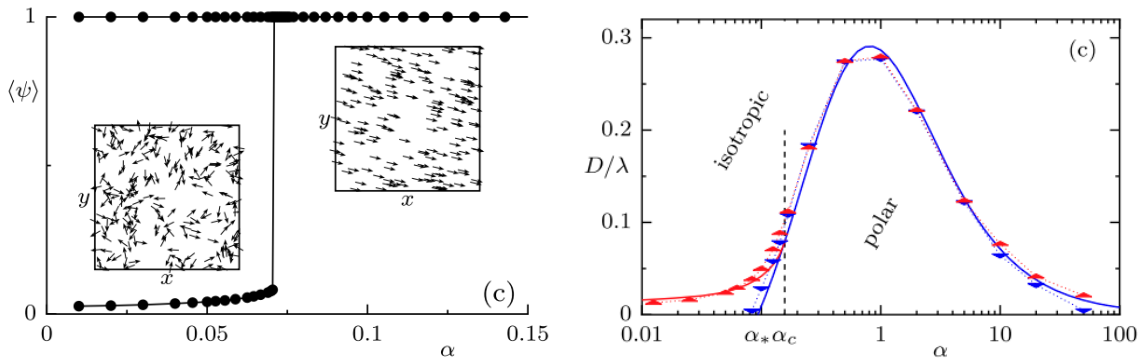


Figure 32 – **Left**. Polar order parameter as a function of $\alpha = \frac{\tau_n}{\tau_v}$, with $D = 0$. A discontinuous transition occurs between the disordered ($<\psi> \approx 0$) and polar phase ($<\psi> = 1$) at some value α^*. **Right**. Phase diagram in the (α, noise) space at fixed density $\rho = 0.01$ and fixed $\tau_v = 4$. λ is the collision frequency and is proportional to the density.

2.3 Thesis Objectives

In the light of the different topics exposed, the main meeting point between Swarm Robotics and Active Matter lies in the emergence of a collective behavior from simple interacting constituents. This motivates work from the robotic side to make use of the phenomenological and mathematical toolbox of Active Matter to 1) design systems driven by physical interactions and 2) better understand the dynamics of interacting robots through microscopic and large scale models. On the other hand, this interdisciplinary crossroad also motivates work from the physics side by providing rich, complex and configurable experimental systems : robots. This opens perspectives in both the understanding and control of collective active systems.

We start in Chapter 3 by presenting the experimental setup used to perform swarm robotics tasks of phototaxis, going through the details of the robotic platform, arena and algorithms in use.

This is followed by results in learning experiments in Chapter 4, with the deployment of an online distributed learning algorithm based on evolution in a population of real robots. Experiments are performed as an experimental validation of the HIT algorithm, a novel algorithm designed for robotic swarms with limited capabilities, for different set of parameters and different controllers.

Chapter 5 focuses on *Morphological Computation* experiments, in which robots design provide a re-orientation response to collisions, enhancing robot capabilities with physical interactions. This chapter gives details in the mathematical modeling of the robots, allowing for simulations and giving insights on the large-scale behavior of the system. A general model of inertial self-aligning particles is also presented as an extension of a state-of-the-art model, allowing to capture unseen inertial effects present in Bristle-Bot like robots.

Finally, Chapter 6 presents a formulation of distributed learning under the form of coupled

stochastic differential equations (SDEs), as a possible path towards understanding the large scale effect of learning *in sillico*.

3 Experimental setup and algorithmic tools

This chapter presents the experimental setup as a whole, including robots, platform and algorithms. It describes in length the Kilobot robot, used to perform all further experiments, how we modify it by using 3D printer exoskeletons and how we ran the different experiments in a 1.5 meter long circular arena. The HIT algorithm is presented at the end of this chapter, a distributed algorithm developed for online learning in swarm of robots. This algorithm is the one in use in all further learning experiments.

3.1 The Kilobot

Kilobots are low cost robots designed at Harvard University's Self-Organizing Systems Research Lab in the early 2010's. The final version of the robot is presented in a 2012 paper [33], with a clear objective of scaling up the size of real robots experiments in swarm robotics. The cost is estimated as low as $14 for the prize of individual parts, and it takes 5 minutes to assemble. The Kilobot robot is designed to be as simple as possible, and as such it has very little capabilities and reliability. It is only through large scale coordination that the robots are able to perform rich and complex tasks.

The Kilobot robot, presented in Figure 33, is made of a 33mm wide round circuit board mounted on 3 rigid metal legs and supporting a cylindrical battery, making it approximately 34mm tall. On top of the board are placed a light intensity sensor, a RGB led, two vibration motors and the removable 3.7V lithium-ion battery. On the bottom of the circuit board, the robot is equipped with a close range infrared emitter/receiver, used for both the communication inter-Kilobot and computer-Kilobot. The robot has two sets of two apparent pins, and is powered by strapping a jumper over the left-side pins. The two remaining pins on the right are for serial output, allowing to log information to the computer. Because of the vibration and the broken front-end symmetry, the robot is able to move forward. The Kilobot hardware, principle of motion and flaws are detailed

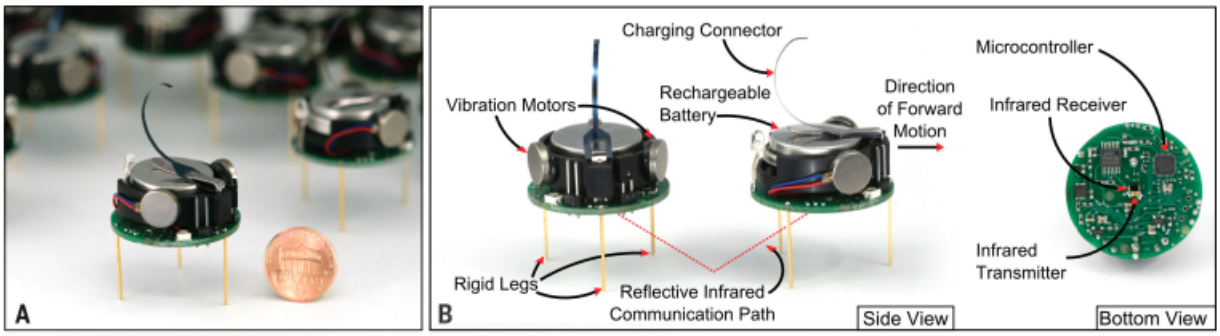


Figure 33 – The Kilobot Robot. (**A**) A Kilobot robot, shown alongside a U.S. penny for scale. The Robot has a diameter of 33mm and is 34mm tall, without accounting for the charging connector, the small metallic piece going upward. (**B**) A Kilobot is equipped with a light intensity sensor, a RGB led, two vibration motors and a removable lithium-ion battery. It can communicate via an infrared emitter/receiver placed on the bottom of the circuit board, effectively sending messages by reflection on the support. Credit : Rubenstein et al. 2014 [15]

further in the next chapter.

3.1.1 Specifications

The Kilobot is equipped with an ATMega 8bit @ 8MHz processor as well as 32KB Flash memory, to which programs can be uploaded via Infrared using the Over-Head Controller (OHC) shown on Figure 34, another piece of hardware built as an interface from the computer to the Kilobot. Each Kilobot comes with a default "support" program, called the bootloader, containing the lowest level code that enables communication and the use of sensors and actuators.

The communication between Kilobots is made through the infrared LED transmitter and infrared photo-diode receiver, which are located in the bottom-center of the circuit board and are pointed directly downwards at the table. The emitters are wide-angle in such a way that the communication takes place by reflection on the support. This way of transmitting information is quite dependent of the reflection of the support for the corresponding wavelength. By measuring the intensity of one upcoming signal, a Kilobot can also calculate the distance to the emitter, with a sensing accuracy of ± 2 mm, and precision under 1 mm - provided that is has been calibrated appropriately to work on its support. Each packet sent has a size of 12 bytes, of which 9 bytes are the actual payload. 2 Bytes are used to store a cyclic redundancy check (CRC) of the whole message, a classical method that guaranties the integrity of the packet. The last byte is a flag used to send different types of messages, mainly used by the OHC to interact in specific ways with the Kilobots. Although the modification can be done in the bootloader, the frequency at which packets

Figure 34 – The Over-Head Controller (OHC). This device is connected to a computer via a B-type USB cable, and used via a specific Graphical User Interface accessible at https://github.com/acornejo/kilogui. The OHC uses the same kind of micro-controller as the Kilobot and communicates in infrared with the same protocol. The OHC is used to upload code to the Kilobots, to monitor battery voltage, to run or stop current program, to log information from the Kilobot via the debug pins and finally to reset the bootloader of the defective Kilobots.

are sent is by default 2 Hz.

Regarding power consumption, each Kilobot is shipped with a 3.7 V Li-Ion 160mAh battery, which can power the robot for 3-10 hours continuously or up to 3 months in sleep mode. Power is switched on and off by connecting or disconnecting the power jumper on the top-right side of the Kilobot. The battery can be charged by applying a continuous tension of 6V directly to the poles of the battery. If the battery is inserted in a Kilobot it can also be charged through the accessible top part of the battery (GND) and the legs of the Kilobot.

The Kilobot is programmed using a dedicated Application Programming Interface (API) written in C, available at https://github.com/acornejo/kilolib. A full description of the API as well as code snippets can be found on the Kilobot User Manual [**?**] and on the Kilobotics web page [**?**], the latter containing tutorials under the "Labs" tab.

The microcontroller of the Kilobot, the ATmega328p, is part of the AVR family of microcontrollers. These chips are particularly common in embedded systems, as they have been popularized by the Arduino line of open hardware boards, hardware designed to be easily accessible to hobbyists and for educational purposes. As stated before, everything programmed for the Kilobot is written in C, a compiled language that has to be translated to a set of instructions via a compiler. For the ATmega328p, the code needs to be compiled using the avr-gcc compiler.

The API works using two main functions, **setup()** called at the beginning of the program and **loop()**, looping indefinitely. This way of doing is the same as the Arduino boards. The user needs only to write code in the core of these two functions, every input and output to the sensors and actuators of the robot, including communication, powering motors and time tracking from an internal clock is done by the API. One important thing to consider for the user is the frequency of the **loop()** that is not constant, but loop as fast as the instructions are processed and interruptions

are triggered by the sensors. The user has to use the internal clock of the robot to perform timed operations, for which frequency is slightly different from one robot to another. On this note, the synchronization of clocks in distributed systems is in itself a difficult problem, beyond the scope of this thesis.

3.1.2 Motion of the Kilobot

Regarding the motor control, each motor is powered with Pulse-With-Modulation (PWM) signal sent by a function in the API. The PWM signal is a common technique used in the control of actuators. It consists in varying the voltage supply between 0% and 100% at a rate faster than it takes the load to change significantly. From the user standpoint, the function *set_motors(Left, Right)* in the API specifies the desired intensity to run each motor on a scale from 0 to 255. In combination with the OHC graphical user interface, it is possible to specify the value of special variables in each Kilobot (*kilo_straight_left*, *kilo_straight* ...) allowing to calibrate individually each robot by saving the motors intensities that suit the most a straight movement.

In the Kilobot, motion is based on the stick-slip principle [**?**]. This motion principle makes use of an eccentric mass rotated by a motor (2 in the case of the Kilobot) making the robot alternate between cycles of complete stops (stick) and moving forward (slip). In order to get an intuition on why this works, let us imagine a robot moving to the right, driven by a single motor driving a mass m anticlockwise at an angle θ relative to a vector pointing at the ground. This situation is represented on Figure 35. When $\theta = 0°$, the robot is subject to the highest normal forces and therefore highest frictional forces, it stays in place and do not move. This is the *stick* phase. As the angle increases counterclockwise, the force projected on the x axis ($\propto \sin\theta$) overcomes friction. At $\theta = 180°$ the robot is subject to minimal friction and has non-zero velocity to the right. This is the *slip* phase. At this point, the robot starts to accelerate to the left, in the opposite direction of the movement. However, inertia has to be overcome first and the angle at which velocity become zero is bigger than 180°. Past the angle $\theta_{v=0} > 180°$, the robot has a negative velocity but as friction increases again, it prevents the backward velocity from increasing as much as the forward velocity.

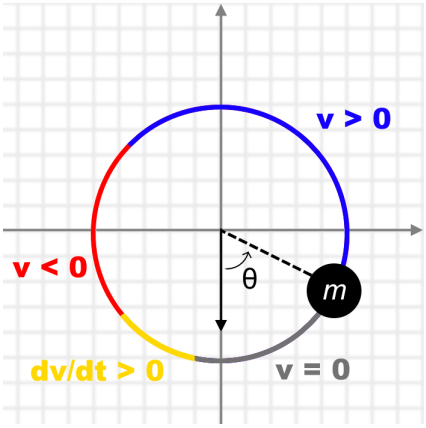


Figure 35 – Schematic representation of the vibrating motor driving an eccentric mass m at an angle θ, allowing for motion with the stick-slip principle. The different phases of motion along the x axis are represented with colors. When $\theta = 0°$, the robot is static on the ground and subject to the highest frictional forces. The force projected on the x axis ($\propto \sin\theta$) overcomes friction and the robot moves forward. Once $\theta = 180°$ the velocity is greater than 0, the robot accelerates backwards but inertia has to be overcome in order to get zero velocity. In the last phase, the robot moves backwards, but the friction comes into play again and prevents the backward velocity to reach the same intensity as the forward velocity.

The Kilobot uses two motors to drive its stick-slip motion. A more complete description of this specific design can be found in Vartholomeos 2006 [**?**] and [**?**]. This type of motion makes the relation between motor intensity and actual velocity to be quite complex. A 2019 study [**?**] of a different robot based on the same design suggests that the net velocity boils down to a multiple of the mean intensity applied to both motors, and the net torque is proportional to their difference. We found similar results on a range of motor values in further experiments. According to the user manual [33], the well calibrated Kilobot can move forward at approximately 1 cm/sec and rotate approximately at 45 degrees/sec.

3.1.3 Flaws of the Kilobot

By using the Kilobot repeatedly, multiple failures can occur. As the legs are soldered on the PCB on a single delicate point, the repeated bending of the legs can cause the welding to break and the leg to detach. The other part of the robot that can also detach easily are the vibrating motors, glued on the front. This is fixable by removing the glue and gluing it back with a glue gun. The precise position of the motors already has a noticeable variance from one Kilobot to another, and therefore the gluing "by hand" does not add to the existing heterogeinity in the swarm. On the software side, corruption can occur. The Kilobot may not respond to messages or behave in a unexpected way. This is easily fixed by re-uploading the bootloader to the robot, with a procedure well documented in the user manual. When this solution does not work, it is most likely that sensors or actuators have stopped working correctly. This case is rare, but there is usually no way

of fixing the issue, and the Kilobot must be disposed of. Over the course of 3 years, this represent about 30 Kilobots in a total of 300.

3.2 Experimental Platform

3.2.1 3D Printed Exoskeletons

In order to obtain smoother trajectories and later study the effect of morphology on the dynamics of the Kilobot, we designed and 3D printed exoskeletons replacing the default rigid legs. These exoskeletons take the shape of a circular disc encapsulating the circuit board of the Kilobot, with legs extending further than the 2cm metal legs soldered to the Kilobot. Multiple exoskeletons where iteratively made with two objectives in mind, 1) to upgrade the bare Kilobot in terms of motion and tracking and 2) to discover new dynamical effects produced by a change of morphology in a stick-slip system. Different trials for exoskeletons are shown on Figure 36. The exoskeletons are made different by changing the position of the center of mass of the system, changing the positions, multiplicity and bending of the legs, and trying different types of materials. The exoskeleton helps the tracking of the Kilobots by adding a large area on which we glue patterns easily recognizable on the films. Finally, the exoskeletons tend to improve the speed compared to the bare Kilobot, and do not interfere negatively with the ability to communicate.

Figure 36 – Examples of exoskeletons made to improve the bare Kilobot and potentially generate effects on the dynamics. Exoskeletons differ by the multiplicity of their legs, as well as the position of the center of mass of the robot. Different types of resin were also tested. The two exoskeletons to the right where made to hold small toothbrush-like brushes, effectively changing the type of legs in contact with the ground.

Regarding the fabrication of these exoskeletons, everything is made in the lab using a professional 3D printer made by Stratasys (Stratasys Connex3 Objet260), presented in Figure 37. This 3D printer deposits drops of liquid photopolymers on a printing plate, and uses UV light to solidify the material. Each print comes embedded in a soft material, the support, which has to be removed manually. The cleaning consists of 3 steps : mechanically scraping out the support, rinsing with water and for a cleaner result dipping in a bath of 2% sodium hydroxide 1% sodium metasilicate. This last part may alter the rigidity of the final print, and in the case of small parts like ours we prefer to skip this step entirely. The final prints are not as clean as they could be, but material integrity is guaranteed. Materials used for fronters and aligner are "Support 706B" and "VeroClear". Given the high number of prints that have to be done to assemble a complete swarm, this process takes a lot time. An approximation would be about 6 minutes per robot, including cleaning of the exoskeleton and fixation of the detection pattern, without taking into account the possible breaking of one exoskeleton while scraping out the support.

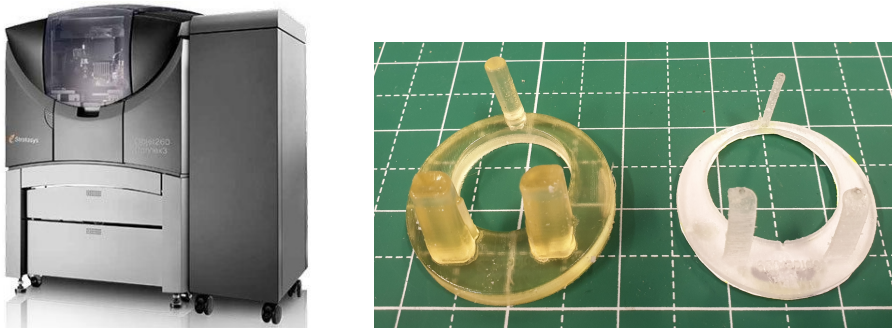


Figure 37 – **Left** The Stratasys Connex3 Objet260 printer. This printer uses a photosensitive resin and UV light to achieve a precision up to 20μm. Printing one exoskeleton takes about 30 minutes, but as it does not require human monitoring, most of the printing is done overnight. **Right** The prints are first encapsulated in a support resin that needs to be scrapped out manually. The part is then rinsed and cleaned to achieve the final result.

To use in our different experiments, we retained three different exoskeletons, exhibiting different dynamics. The first two are part of a paper published in Science Robotics in 2023 by B.Z. Matan, F. Jérémy et. al [**?**]. These two exoskeletons, presented on Figure 38 have a tripod-based design (a pair of flexible legs and an opposing stiff leg) with a circular chassis of diameter 4.8cm encapsulating the Kilobot. The two morphologies use an anisotropic flexible leg design (1mm thick, 6 mm wide, and 20 mm long). We call the two designs the **aligner** and **fronter** exoskeletons. The fronter has its flexible legs in the front and all the mass in the rear, contrary to the aligner which has its flexible legs in the rear and all the mass in the front. This gives rise to a force-orientation response of opposed nature, further detailed in Chapter 5. These exoskeletons take inspiration from the self-propelled polar disks used in Deseigne et al. 2010.
[53].

The third exoskeleton design we used to perform experiment uses toothbrushes heads to replace the Kilobot's legs. The multiplicity of the contact points and the overall flexibility give rise to unseen oscillatory behaviors, detailed further in 5.3.2Toothbrush exoskeleton oscillating phenomenasubsubsection.5.3.2. These behaviors are made possible by the increase in inertia for both translation in rotation, rendered possible by the toothbrushes, combined to a biased motion which makes the typical robot go in a circle of wide radius.

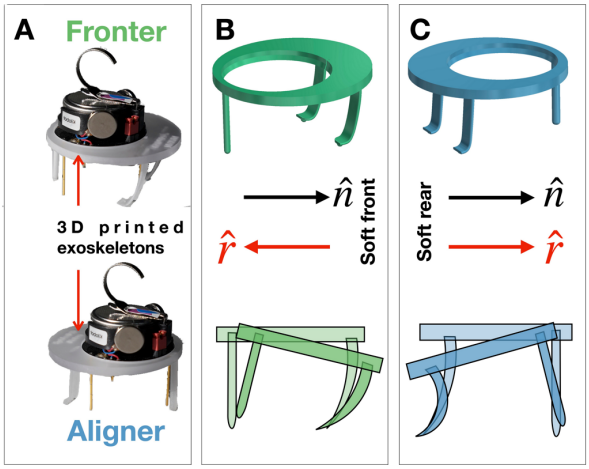


Figure 38 – (**A**) Pictures of the fronter and aligner exoskeleton, encapsulating the Kilobot. Both exoskeleton have a tripod based design, and show a force orientation of opposed nature. (**B**) The fronter exoskeleton has a pair of flexible legs in the front and the mass distributed in the rear, it tends to align its direction against external forces. (**C**) The aligner exoskeleton has a pair of flexible legs in the rear and the mass distributed in the front, it tends to align its direction with external forces Taken from [**?**].

Figure 39 – Corner view of a Kilobot inside the toothbrush exoskeleton.

The toothbrush exoskeleton is 55mm wide and about 60 mm tall. It is made in two separate pieces, the core on which we glue the new legs, and the outer ring, defining the external shape. Both pieces are joined together by screwing the ring in a hollow part of the core. The top of the exoskeleton is colored black on the front side, and white on the back side, allowing for a simple yet effective tracking. We used a different material to 3D print this exoskeleton compared to the previous designs. The name of the material used is "Rigur RDG450", which compared to the previous translucent material is white and opaque and is less brittle.

In contact with the support are toothbrush heads, replacing the Kilobot's legs. Many different models where tested, and we decided to stick to the simplest possible geometry. Regarding the fabrication, the toothbrushes that we use are sold as rechargeable heads of plastic and perpendicularly stacked nylon 1010 filaments (Caliquo, "soft"). By default, the filaments are not bent. Because this type of nylon has a relatively low glass transition temperature, it is possible to permanently

deform the filaments. We designed an aluminum cylindrical device shown on Figure 40 allowing to bend the toothbrushes at approximately a 20° angle. The toothbrushes are first inserted in the cylindrical core, 15 at a time, and a tube is slid over the core, bending the toothbrushes. The whole device is then heated at 60°C for 40 minutes, after which the toothbrushes are removed cooled down quickly under water. Trial and error showed that a too small angle of deformation makes the robot go in a random direction instead of forward. On the contrary, a deformation angle too big lessen the final speed of the robot. A 20° deformation angle is a reasonable balance between both effects, making the robot move forward at a great speed, up to 20cm/s.

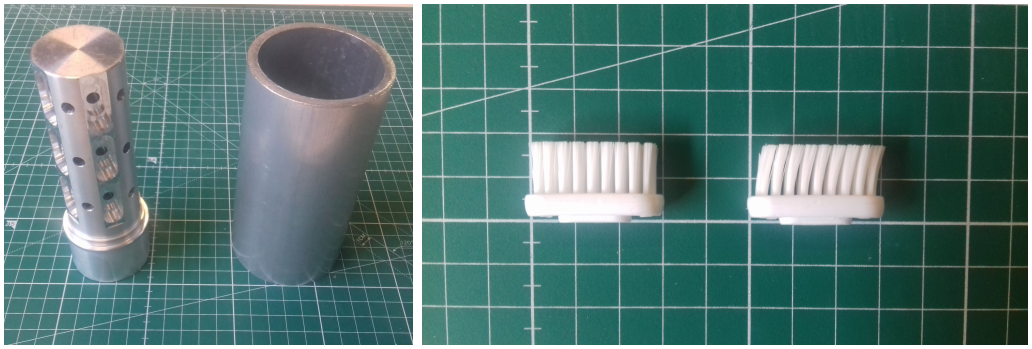


Figure 40 – **Left** The cylindrical device used to bend the toothbrushes. The core is on the left, with placeholders for the toothbrushes, the tube is then slid over the core to achieve proper bending of the toothbrushes. **Right** Picture of a toothbrush before and after bending. The angle of deformation is not constant but close to 20° for a correctly bent toothbrush.

Since the Infrared Emitter-Receiver of the Kilobot is located underneath the PCB, this exoskeleton had to be high enough, such that the bottom plane on which the toothbrushes are glued interferes the least possible with the inter-Kilobot communication, while maintaining stability of the whole structure. We performed a series of simple communication tests in order to characterize roughly the difference between a bare Kilobot and a Kilobot with the toothbrush exoskeleton, on two different surfaces. Two Kilobots both with or without exoskeleton are placed at known distance from each other, on a given surface : either black mat PMMA or white mat PMMA. One is fixed and is sending 100 numbered packets, communication is considered valid if > 95 packets where correctly received. Distances used where in slices of 0.5 diameters. Results are shown in Figure 41. Despite the PCB being higher, the distance of communication still increases when using the toothbrush exoskeleton. This is due to the increase of the maximum angle at which the reflection on the ground is possible. As can be seen, the main difference regarding the communication distance comes from switching between the white support and the black support. This is one of the flaws of the Kilobot, since the transmission of the signal is not direct, changing the supports changes absorption, reflection and scattering in the wavelength used for communication, degrading performances.

We performed a simple characterization of the movement of individual toothbrush Kilobots in the arena, by analyzing trajectories for different PWM values of the motors, (0 to 255 for both

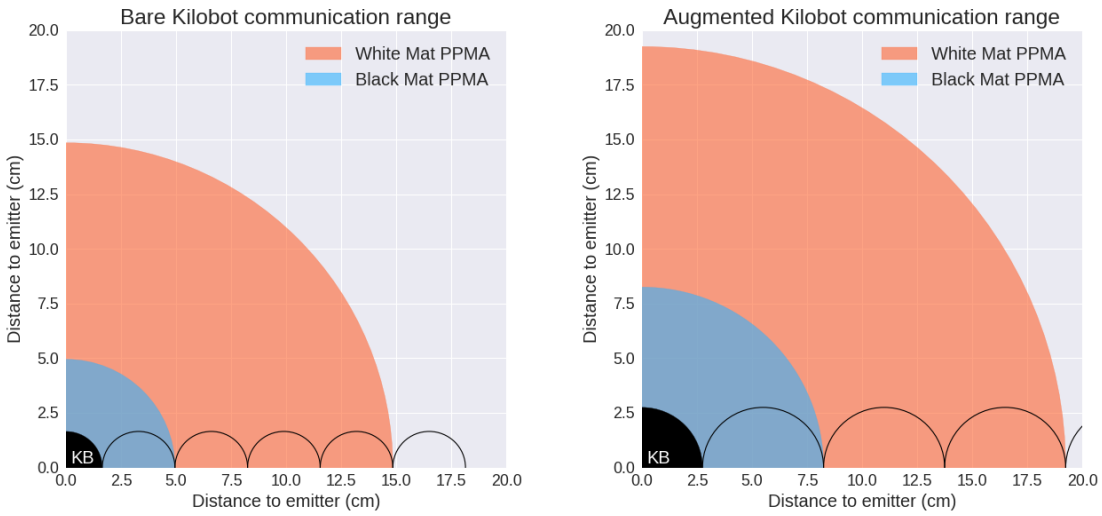


Figure 41 – Approximate communication range for Kilobots over two different surfaces.

Left and Right motor). 35 different set of motor values where tested, with 5 different exoskeletons. Each exoskeleton possess its bias, meaning that its angular velocity is not zero when having set its motors to the same value. The results are summarized on Figure 42. On the first graph, green dots indicate equal motor PWM values for the Left and Right motor, and blue crosses indicate different values according to the distribution presented in figure. We notice that the speed of an individual robot follows a logistic curve of the total PWM value, of which we represent a fit with the red curve. This curve admits a linear regime for mean values between 70 and 130, in which the resulting speed correlates well. These results show that a simple yet reasonable approximation to map the motor values on the left m_L and on the right m_R to the magnitude of the velocity v and the angular velocity ω can take the form :

$$\begin{cases} v = c_v(m_L + m_R) \\ \omega = c_\omega(m_L - m_R) \end{cases}$$

where c_v and c_ω are constants. This mapping is the same as the one of a differential drive robot [**?**], although the Kilobot is not itself wheeled. Similar measurements have been done on a robotic platform similar to the Kilobot and developed in the lab, presented in Appendix E.

3.2.2 Arena

The arena in use in the learning experiments, built in December 2021, is presented on Figure 43 with a picture and a 3D model. The whole arena rests on a thick slab of chipboard wood of 5x200x200cm on which arena floor, walls and overall structure is screwed in. The floor consists of a 2mm thick white or black PMMA slab, and the floor is a 0.2mm thick circular aluminum sheet of 1.5 meters in

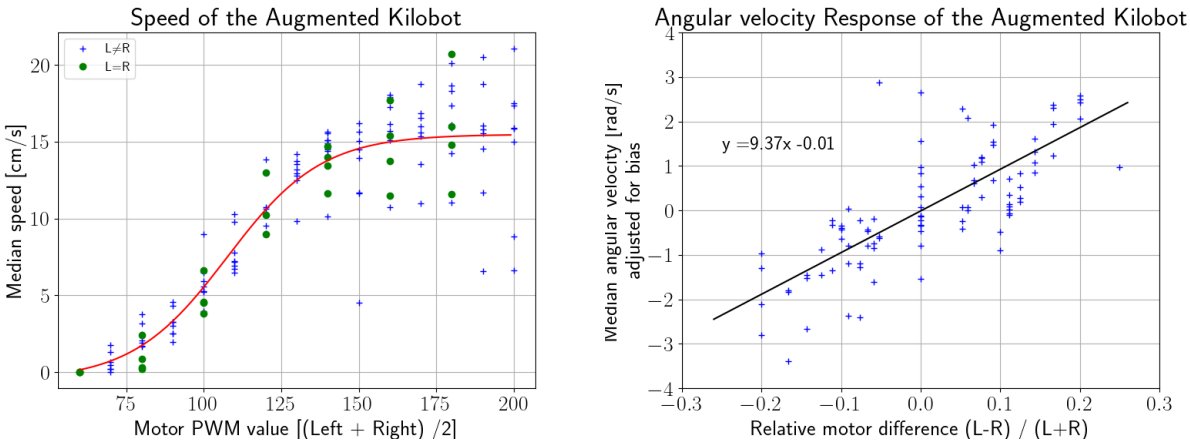


Figure 42 – **Left** : Speed of the toothbrush Kilobot for different motor values, with fitting sigmoid in red. **Right** : Angular velocity of the toothbrush Kilobot for different motor values. In both experiments, 5 different exoskeletons where tested for 35 motor values.

diameter. The main beam structure is fixed along a bearing wall of the experimentation room, two beams are placed on the opposite sides to hold black fabric covering the whole arena. The inside of the arena is lit via LEDs light strips attached to the inner part of the circular wall. About 2 meters higher than the floor, a black and white camera is set to film the entirety of the arena, and a small video projector allows for projection on part of the arena.

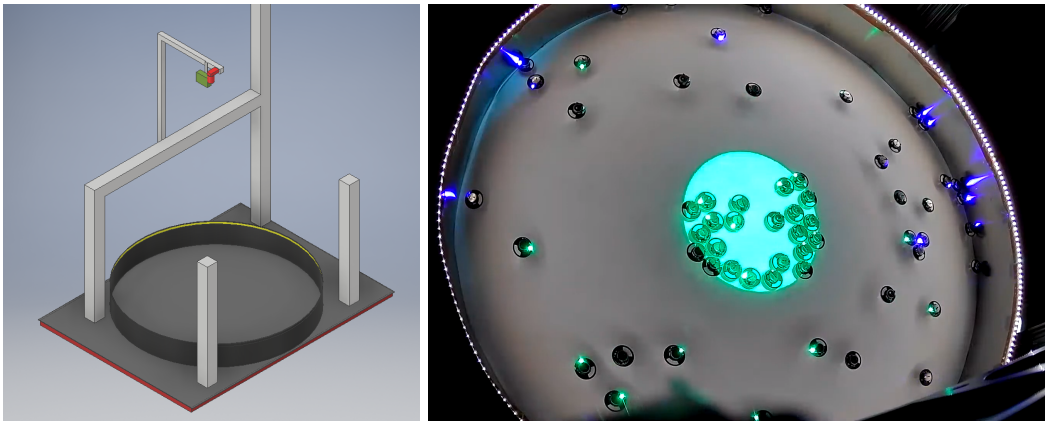


Figure 43 – **Left** 3D Model of the arena in use for learning experiments. The circular part is 1.5m in diameter. Parts are color coded with LEDs strips in yellow, camera in red and video-projector in green. The long beam structure in the back is fixed along a bearing wall of the experimentation room, and the whole arena reposes on a thick slab of chipboard wood. **Right** Top side view of the arena during a phototactic experiment. This photo was taken with an additional webcam used to monitor the experiments without disturbing the acquisition by the camera mounted on top.

The vast majority of experiments we run are phototactic experiments, where the Kilobots have to use their light sensor in order to find a source of light coming from the projector. As to not disturb the sensing of the Kilobots the arena needs to be dimly lit out of the projected zones. The original idea was to light the arena using a wavelength out of the Kilobot sensor scope but visible by the camera. This way, the images produced would be as clear as possible and the task would be truly independent from the observation. Unfortunately, the camera's sensor and the Kilobot sensors share roughly the same absorption spectrum, making the decoupling of observation and task a challenge. In addition, the contrast between the lit and unlit region makes the detection of partially lit Kilobots very difficult. This is due to the fact that we use contrast to identify the front of the robots by having a white front and a black rear. When a robot is partially lit, the lit region becomes the zone of high pixel intensity independently of its original color, rendering the pattern used for orientation detection useless.

The solution we came up with was to dimly light the arena out of the projected zones, with the LEDs covered in red cellophane. The projected zones are all set to use only the green channel, as it matches better the Kilobot sensor intensity peak than the color blue. In front of the camera, we placed a red long-pass filter, cutting wavelengths shorter than 645nm. Doing so, the contrast between the lit and unlit regions disappears on the film, the overall light intensity is enough to track the Kilobot and the real contrast in light intensity is maintained high enough for the Kilobots to perform phototaxis. Images of the arena taken from the acquisition camera are shown on Figure 44. From the Kilobot point of view, this places the dark region at a value of ≈200/1024 in their buffer, and the lit region at ≈ 1000/1024. This measurement is made following the procedure described in Chapter 4.2.1Light Integrationsubsubsection.4.2.1, logging the result on the computer with the OHC.

On one hand, this procedure allows us to decouple a bit the tasks and observation, with the good calibration of light intensity and camera parameters. On the other hand, having such a filter is detrimental to the total light intensity we get on an image. This imposes a much higher exposition time than without a filter. The reasonable decrease of contrast in the entirety of the arena was found to be better with respect to detection than an overwhelming contrast at the center.

Figure 44 – Example images captured from the camera of the circular arena, exposure time 100ms, 4096x3000. **A.** No filter with green projection. (+12dB) **B.** Red filter (645nm) with green projection. (+24dB) **C.** Red filter (645nm) with red projection. (+24dB)

The light disc covers $\sigma = 6\%$ of the area of the arena, placed in the center with a diameter

of 36cm. There is no local intensity gradient to guide the robot to the lit region, and to enforce crowding the lit region can not accomodate more than 80% of the robots (about 50).

3.2.3 Image acquisition and processing

In all of our experiments, the use of the camera produces a binary file of known structure (see PixeLINK Data Stream File Format), in which we can access directly each frame by calculating its address in the binary file. The frames are saved in black and white with a 8-bit color depth, and we read directly the "Height $\times$ Width" bytes to a numpy array. These raw binaries are stored on internal SSD drives of 512MB. By having two drives, the experimenter can acquire new data in the experimentation room as the same time as previous experiment are being processed on another dedicated computer. The framerate of each film was set to 6fps, allowing to store 2 films on the same drive, improving workflow. On this note, the maximum velocity is observed with Kilobots in exoskeletons which can go up to 20cm.s^{-1}. Since the diameter these exoskeletons is 5.5cm, the minimum working framerate to track the trajectories is 4fps.

In order to detect the Kilobots and their orientation, a black and white pattern was printed and glued to the Aligner and Fronter exoskeletons, shown on Figure 45.Right. As the robots stand on a white background, the pattern has a thick black contour and a white zone on the front, with a little black dot helping to align the printout with the Kilobot before gluing. Because of the inverted shape of the aligner exoskeleton compared to the direction of its movement, the raw data for the angle of aligners is always rotated by π. Using the package OpenCV in Python, we perform a series of image processing operations on the entire frame, ultimately yielding the position and orientation of each Kilobot. The details of the main image processing pipeline is presented in Figure 45.Left. The goal of this pipeline is to obtain a binary image, leaving only the patterns visible on the final result.

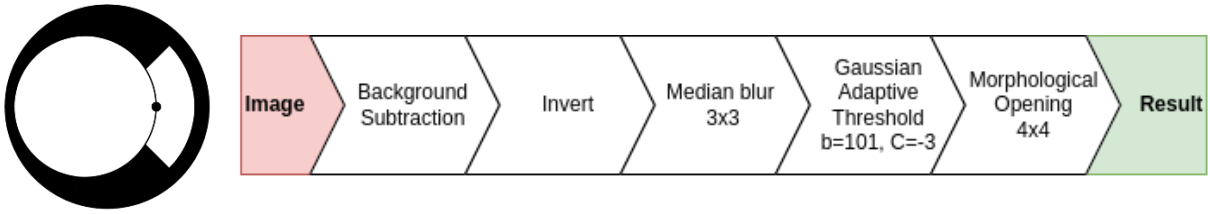


Figure 45 – **Left**. Detection pattern used with exoskeletons, to improve the detection of the position and orientation of the robots. **Right**. Detection Pipeline used for the fronter and aligner exoskeletons.

The first step toward obtaining a clean binary image is the removal of the background, picture taken at the beginning of any experiment before the robots are placed inside of the arena. The image is then inverted and slightly blurred, to eliminate defective pixels or small reflections. An adaptive threshold is then applied to obtain a binary image. The threshold in use is a Gaussian adaptive threshold, which work by setting the pixel (x, y) white if and only if the value at (x, y) is greater than the weighted sum (cross-correlation with a Gaussian window) of the neighborhood minus a constant C. The neighborhood is defined as the square of size b (odd) which has (x, y) as its center. Finally, an operation called morphological opening is applied, which suppresses isolated shapes of small area.

Using the processed frame, the OpenCV implementation of the so-called Hough circle transform [**?**] is used to find the best matching circles with known diameter. This transform is based on the Hough gradient method, a common technique used in computer vision for detecting edges in images. The OpenCV implementation returns the positions of detected circles, which are the coordinates we register as "x" and "y". These coordinates are in turn used to crop the original frame around each robot, yielding N small images that undergo a second image processing pipeline returning the orientation. This second pipeline consist essentially in applying a mask around the exoskeleton and on the Kilobot, and finding the front of the exoskeleton. This is done by applying a binary threshold, leaving mainly the white areas visible, and taking the center of mass of the largest connected component.

For the main experiments, using Aligners and Fronters exoskeletons on a black background with the red filter on, we use the following settings for the camera, and the following conventions for the final data files :

Camera settings

Resolution :	3200x3000px
Framerate :	6fps
Exposition time :	100ms
Camera gain :	+24dB
Bit depth :	8-bit

Hardware specification

— Camera Pixelink PL-D7512MU

— Navitar Lens NMV-12M1 (EFL 12.2mm)

CSV Format

File labeling :
[A,F][0,115]_[preset,evo]_DDMMYY(_bis).csv

frame | x | y | theta[1] | particle[2] |

[1] Orientation $\theta \in [-\pi, \pi]$ of each robot.
[2] Index of a particle, calculated during linking phase.

Arena imperfections and stability issues

During the experimentation phase, we noticed three unprecedented setbacks in the arena. First, the projection on the arena was slightly deformed. During phototactic experiments, a disk of light is projected on the surface of the arena. However, by increasing contrast on the image, we found out that the disk was actually an ellipsoid, with one axis being about 8 pixels bigger than the other.

The second setback was discovered in August 2022 and is more problematic. We came to realize that over the course of the experiments the camera had translated to its right. The reason for this shift is unclear, as everything was tightly screwed and was thought to be completely still. The camera being more than 2 meters higher than the arena, a small deviation at the base would be sufficent to produce this kind of shift. Possible source causes are the vibration produced by the Kilobots on the whole setup, or the dilation and compression of the structure due to day heat.

Since the arena itself was not modified during this time, we could retrieve the centers on each film. Figure 46 shows the positions of centers for each film, color coded from red (oldest films) to blue (newest). The black cross in the middle is the center detected after the acquisition of Kilobot data, on the 19th of August 2022. On the left picture, a threshold has been applied to locate the projected ellipsoid of light. A red circle has also been overlay to show the boundary of the area in which we consider the Kilobots to be lit. As stated before in the Experimental setup section, the first thing that we notice is the ellipsoidal nature of our disc (eccentricity ≈ 0.3). By zooming in, as shown on the right picture, we can see two different zones. At $x \approx 1500$ the arena was new and centered. The first shift has probably been provoked by a collision on the arena, moving the camera by ≈ 80 pixels to its right, equivalent to 4cm. For the rest of the experiments, the camera gradually moved by ≈ 40 pixels, equivalent to 2cm. This displacement is taken into account when measuring the number of robots in the light for each experiment.

The third setback would be the slight off-centering of the light sensor to the center of the exoskeleton, which is about 20 pixels (1cm). This gives rise to orientational effect where Kilobots would believe they are in or out of the light depending not only on their position but also on their orientation. However, in all data acquired and in all further plots, a robot is considered in or out of

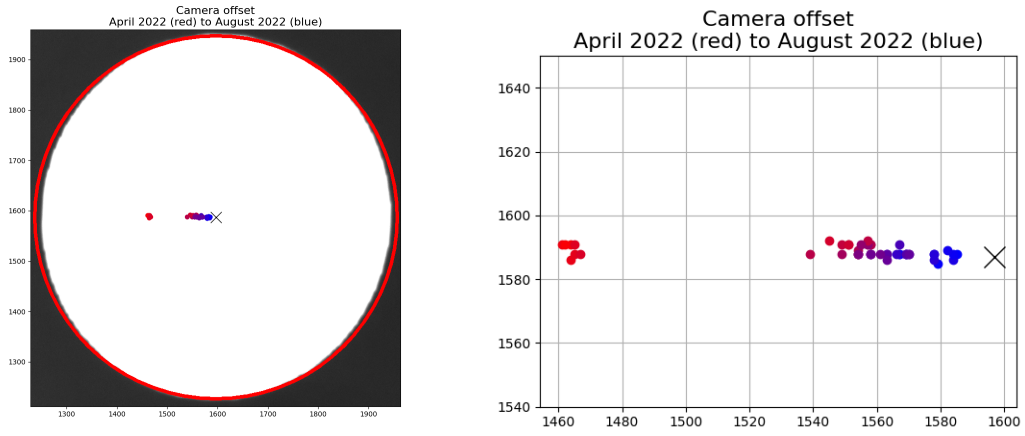


Figure 46 – **Left** : Camera offset over time as an overlay to a binary image of the projected disc of light. The diameter of this disc is 36cm. **Left** : Camera offset over time plotted on a grid. The first shift due to a collision is quite apparent. Axis show dimensions given in pixels, $1mm \approx 2px$.

the light if the center of its exoskeleton is in or out of the light. Due to the impossibility of modifying the location of said light sensor, this is an effect we deliberately chose not to consider. Fortunately, in actual experiments this effect happens mostly when the lit region is already crowded, producing a small relative error when counting lit robots.

3.3 The Phototactic HIT algorithm

3.3.1 Presentation of the algorithm

As presented in Chapter 2.1, to date only few embodied evolution algorithms have been deployed on real robots. We decide to use an algorithm designed to work despite the high limitations of swarm robots : the HIT (Horizontal Information Transfer) algorithm [**?**]. This algorithm has the great advantage of being quite simple in its structure, computation needs and memory needs. It boils down evolution to its main components, making it flexible to different settings and allowing us to envision macroscopic models of embodied evolution in swarms of robots.

The HIT algorithm is inspired from the horizontal gene transfer mechanism observed in bacteria: when two robots interact, only parts of the parameters set can be transferred from one robot to another. In addition, the new parameters sets received by a robot are selected on the fly and are immediately discarded, contrary to algorithms such as mEDEA [**?**] which builds up a reservoir of parameters for later evaluation. This design allows to to minimize the requirements in terms of memory and communication cost, making HIT an ideal candidate for the deployment on Kilobots.

The algorithm works as follows. Robots are programmed to perform a sense-act loop commonly used in robotics for reactive agents. Each robot i runs a deterministic policy parameterized by parameters w_i^k with $k = 1...K$, which translates inputs from the sensors into outputs to the actuators. At the beginning, each parameter is initialized uniform randomly. As a robot moves through its environment, the quality of its behavior is assessed using a reward function which depends on the user defined task to be achieved. The tasks can be the number of items gathered for a foraging task, or amount of light received for a positive phototactic task. Integrating the reward over a given period of time builds up a score, reflective of the quality of the current parameters. When two robots are within communication range, they exchange their parameters and their current score. The robot that has the lowest score replaces its parameters with the incoming parameters. When this is done, a mutation can occur, by slightly offsetting the parameters with values drawn at random. This operation allows to discover new sets of parameters, potentially increasing the final score.

3.3.2 Outline of the algorithm

In our experiments, since we run positive phototaxis experiments, the reward is given as the total amount of light received over time. The type of experiment we run changes the choice of the policy function (controller), as well as the values of the algorithm parameters. The full algorithm as implemented in the Kilobots for the learning experiments is presented below in Algorithm 1. It works as follow :

- **l.1-3** The robot begin by setting the length of its reward buffer. The score of the robot is defined as the mean of this buffer. While this buffer has not been filled at least one, the score of the robot is not defined. The length equal to the sampling frequency times the protection duration. Because this buffer constitute the main memory requirement, its length can be adjusted by sampling the light every so often while keeping the same protection duration.
- **l.4-8** The robot starts its sense-act loop by measuring the current light intensity I, and filling the reward buffer accordingly.
- **l.9-10** Depending both on the light received, type of controller and current parameters, the robot sets the intensity of its left and right motors.
- **l.11-13** Once the reward buffer has been filled at least once, the score is computed as the mean of the reward buffer, and the robot continuously broadcasts its score along with its parameters.
- **l.14-19** If a new message is received, the robot compares the received score to its own plus a minimal reward delta, which is a parameter of the algorithm. If the received score is higher, the robot replaces its own parameters by the parameters received. If the experiment requires it, the parameters can be slightly altered by a mutation, and the protection time can be reset.

In the two learning experiments, Run and Tumble meta-policies and Multi-Layer Perceptron (MLP) inspired controller, the parameters and controllers are different. The type of controller is detailed in each corresponding section. With the Run and Tumble meta-policies, the mutation does not occur, and the protection window only acts at the beginning of the experiment. Using the experimental setup presented in this section, we turn toward the phototaxis experiments, and the deployment of the phototactic HIT algorithm in a real robotic swarm.

Algorithm 1 Phototactic HIT Algorithm

Algorithm Parameters

δ_r : Minimum reward delta
τ_p : Protection duration
$f_{sampling}$: Sampling frequency for the reward

Global variables

i : Unique identifier of the robot
I : Instantaneous light intensity
ρ_i : Score of robot i
$\mathbf{a}$: Action vector
$\mathbf{w}_i$: Policy parameters
$\mathbf{R[T]}$: Reward buffer of size T

Functions

sense() : Instantaneous light measurement
$\pi(\mathrm{I} \mid \mathbf{w})$: Policy function (controller) ; Returns an action (internal state or motor input) according to light received and parameters
act($\mathbf{a}$) : Sends motor input according to the chosen action
broadcast($\mathbf{w}$, ρ) : Broadcast parameters and score to neighboring robots
mutate($\mathbf{w}$) : Applies a slight change to the parameters

Begin
  $t \leftarrow 0$
  $T \leftarrow \mathrm{round}(\tau_p * f_{sampling})$
  while $True$ **do**
    $I \leftarrow$ sense()
    if *sample_reward* **then**
      $\mathbf{R}[t \bmod T] \leftarrow$I
    end if
    $\mathbf{a} \leftarrow \pi(\mathrm{I}|\mathbf{w}_i)$
    act($\mathbf{a}$)
    if $t > T$ **then**
      $\rho_i \leftarrow \sum_{k=0}^{T-1} \mathbf{R}[k]/T$
      broadcast($\mathbf{w}_i, \rho_i$)
      if *new_message* **then**
        $\mathbf{w}_j, \rho_j \leftarrow$ *decode_message*
        if $\rho_j > \rho_i + \delta_r$ **then**
          $\mathbf{w}_i \leftarrow \mathbf{w}_j$
          mutate($\mathbf{w}_i$) ▷ The mutation and reset of the protection window
          $t \leftarrow 0$ ▷ are only used with the controller of Chapter 4.3
        end if
      end if
    end if
    $t \leftarrow t + 1$
  end while
End

4 Learning phototaxis in a swarm of 64 Kilobots

The first series of experiments we run aim to study the ability of a swarm to solve a simple phototactic task using the previously described HIT algorithm. We begin by running a set of "baseline" experiments, in which we evaluate the natural dynamics of the swarm for different configurations, with no phototaxis and no learning as a mean of highlighting further results. Then, for the learning phototaxis experiments, we propose two cases with different controllers of different complexity. In a first case, the controller is built explicitly to solve a phototactic task by setting two "meta-policies" defining the behavior of a robot in and out of light. This controller has a single learnt parameter w which defines the light intensity threshold at which a robot can choose to switch from one meta-policy to the other. In this case, the motion is performed through run-and-tumble, a well studied type of motion detailed further in the following section.

In a second case, the controller has a design close to a small type of Neural Network called a Multi-Layer Perceptron (MLP). This controller greatly increases the cardinality of accessible actions (from 2 to 256^2). As the HIT algorithm is a direct policy search type of algorithm, one can expect the convergence speed to good policies to be inversely proportional to the volume of the parameters describing these good policies in the parameter space. Therefore, the second case is more challenging than the first, but allows a richer set of behaviors compared to the meta-policies.

4.1 Baseline

In order to better understand the dynamics of the swarm, we tried to formulate different baselines for which the dynamics does not rely on any kind of phototaxis or learning. We perform the experiments during two hours with 64 Kilobots, as we would do for the phototaxis experiment, with no projected disc of light. We consider 3 different cases :

- Straight motion Low, we set both motors PWM values to be constant, equal to half the maximum possible value 127 / 255.
- Straight motion High, same as before we set both motors PWM values to be constant, this time equal to the maximum possible value 255/255.
- Run and Tumble motion, with a constant run time $\tau_{run} = 2s$ and a tumble time drawn from an array simulating a long-tail distribution of mean $< \tau_{tumble} >= 4s$. The PWM motors

values are set to 130/255, an arbitrary value in the reasonable motor value range, sufficient to perform motion and not too high to preserve the batteries.

Typical trajectories for the 3 baseline configuration are represented on Figure 47. Looking only at the shape of the different trajectories, some preliminary observations can be made. The first observation is the overall lack of straight lines for each configuration, even when the motors are set with equal PWM values. This can be attributed to two effects, the orientation noise during the motion of a Kilobot and the individual biases of each robot. Indeed, by vibrating each Kilobot slightly changes its direction when moving forward, and due to small imperfections in the making of the robots or their exoskeletons, each Kilobot will naturally spin clockwise or anti-clockwise when setting its motors to an equal value.

The second observation to be made is the accumulation to the boundary, especially for the Straight Low and Run and Tumble configurations. For example, the orange robot on the Run and Tumble figure stays at the wall for 10 minutes. Comparing to the trajectories of the Straight High, it appears that the Kilobots manage to escape the wall with more ease when the motor intensity is set to its maximum. However, for this configuration, a higher load is put on the battery and a good proportion of the robots stop before the end of the two hours. This trade-off between the accumulation to the wall and the energy consumption is illustrated on Figure 48 with the histograms of positions for both the Straight Low and Straight High configurations. Figure 48 also shows the mean velocity over time for the Straight High configuration, illustrating the robots gradually coming to a stop due to a lack of battery starting from $\approx 5400s.$ = 1h30min. In total, the configuration spending the most time at the wall is the Straight Low configuration, with a mean of $N_0 s$ at the wall for each Kilobot, compared to N_1, N_2 for the two other configurations.

Since in the baseline experiment there is no changing of behavior for the individual robot, neither in time or space, we consider that the system quickly ($t \approx 120s$) gets to some sort of steady state. This allows us to integrate measurements with respect to time without losing any relevant information. This consideration changes for the phototactic experiments, in which the behavior of the robots changes both in space and time in the case of learning. We measure distributions of both the velocity and the amount of robot in the area delimited by the disc of light used in the phototactic experiments (with $d_{light} = 36cm$). We refer to this quantity as the concentration of robots in the center and this measurement constitute the non-phototactic performance of the Kilobots, which we expect to be beaten by any form of software implementing phototaxis. These quantities are shown as histograms for each configurations on Figure 49.
For the three configuration, the concentration of robots in the center is low compared to the size of the swarm, with a mean at each frame 0.54 robot for the Straight Low, 1.06 for the Run and Tumble and 2.83 for the Straight High. This represents in the three configurations a mean less than 4% of the swarm of 64 Kilobots. In the best case, for the Straight High configuration, as the robots escape the wall easily compared to the two other configurations, the number of robots in the center can go up to 10% of the swarm although it represents only a few minutes in the full experiment.

The velocities are calculated over 3 consecutive frames. The measurement excludes the robots close to the wall by masking the robots outside of a diameter of measurement $d_{measurement} = 120cm$. Still, a good proportion of the robots are stopped or very slow at any given time in the experiment.

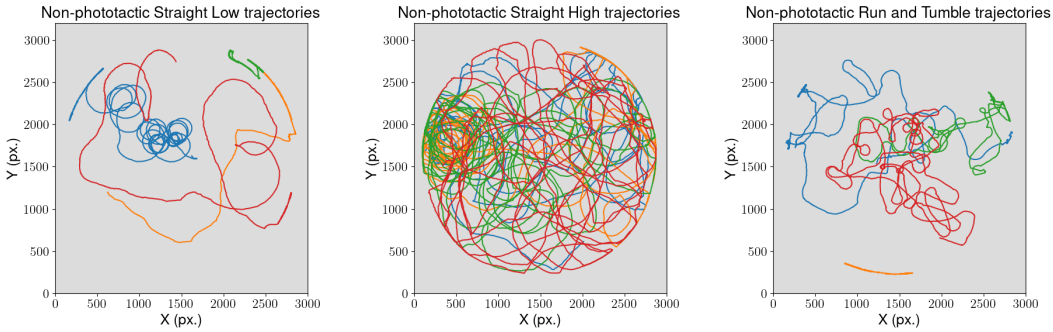


Figure 47 – Examples trajectories for the different baseline configurations, with 4 different robots and segments lasting 10min. $1px \Leftrightarrow 0.48mm$ **A**) "Straight motion Low" with motors PWM values equal to 127/255 **B**) "Straight motion High" with motors PWM values equal to 255/255 **C**) "Run and Tumble" with constant $\tau_{run} = 2s$ and $<\tau_{tumble}>= 4s$ drawn from an array simulating a long-tail distribution.

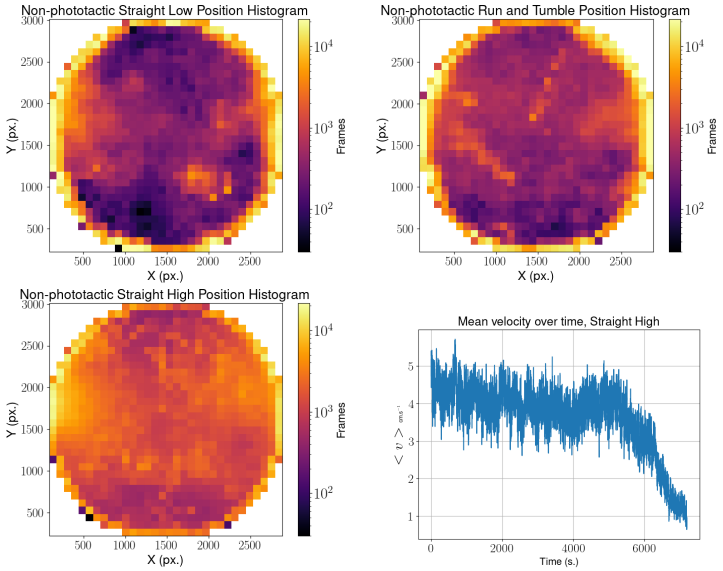


Figure 48 – Trade-off between accumulation at the wall due to a lower speed and shutting down of the robots due to lack of battery. **A-C**) Histogram of the positions in the first hour and a half for respectively the Run and Tumble, Straight Low and Straight High configurations. Color codes for the total presence duration of any robot, from 5 seconds to 1H. The wall appears highlighted due to the high concentration of robots, momentarily stuck against it. High velocity makes the robots escape the wall with greater ease. Some of the robots still get momentarily stuck against the wall, but the mean time of a "stuck" event is lower in the Straight High than in the other configurations. **D**) Mean velocity over time for the Straight High configuration. The mean velocity starts to drop at $t \approx 5400s$=1h30 because of individual robots shutting down due to a lack of battery power.

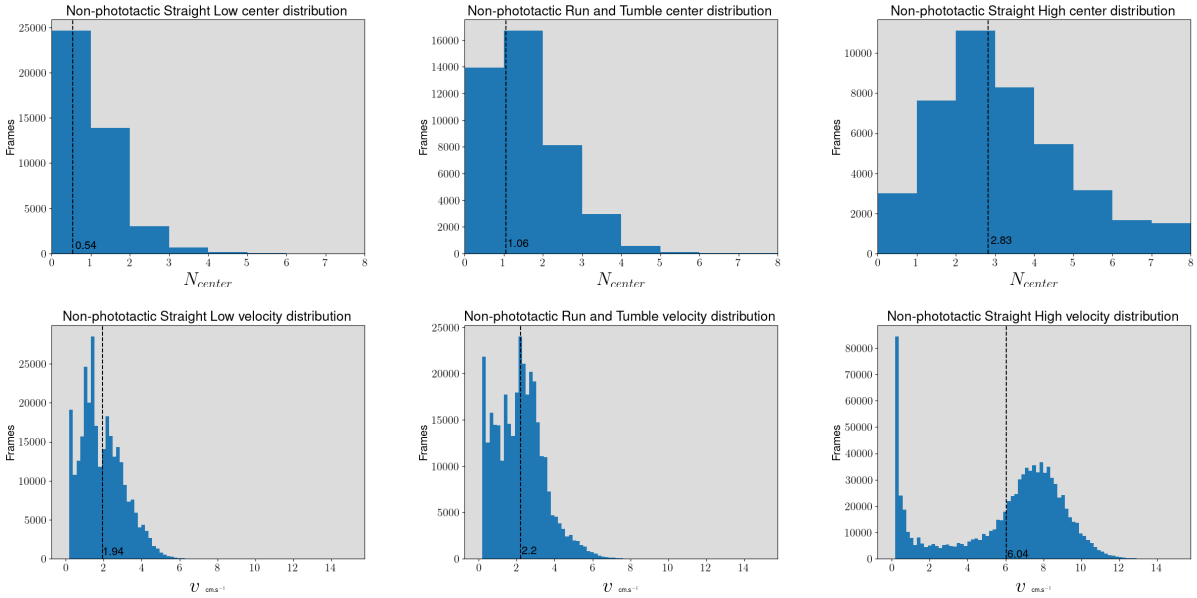


Figure 49 – Baseline measurements starting from $t = 120s$.. **Top**: Distribution of robots in the center, in the area defined by the disc of light in phototaxis experiments, $d_{light} = 36cm$. **Bottom**: Velocity distribution of robots, excluding the robots at the boundary of the arena $d_{measurement} = 120cm$. From left to right : Straight Low motion, Run and tumble, Straight High motion.

This is due for the most part to collisions, and in the Straight High configuration this is due to robot failure, as they run out of battery. It is important to specify that the Kilobots mostly go from a normal motion to a complete stop when the battery becomes empty, and the velocity of a single robot does not seem to decrease with the remaining battery power. Mean velocities for the three configurations and this measurement method are $1.94cm.s^{-1}$ for the Straight Low, $2.20cm.s^{-1}$ for the Run and Tumble and $6.04cm.s^{-1}$ for the Straight High. At full speed, a single Kilobot with the exoskeleton can go up to $12cm.s^{-1}$.

4.2 Learning with Run and Tumble meta-policies

We now turn towards the learning experiment, using the HIT algorithm and Run and Tumble motion. As it is running, goes through 3 main phases, respectively the light integration phase, the motion phase and the assessment phase. These phases correspond respectively to the $sense()$, $act()$ functions and the controller, $\pi(I|\mathbf{w}_i)$.

4.2.1 Light Integration

During the light integration phase, two batches of 100 instantaneous measurements are performed. As the photosensitive cell is quite noisy and sometimes returns a default value of -1, performing batches allows for a much stable measurement of intensity. The values are added to the final return

value each time it is different than -1 until reaching 100 instantaneous measurements. This allows for a constant measurement time, at the cost of a very small probability of a failed batch in the case where all instantaneous measurement return -1. A single batch is measured every 0.25 seconds, setting the sampling frequency $f_{sampling}$ to $4Hz$. The values of successive batches are stored in the reward buffer, allowing for the computation of the robot's score as the mean of said buffer. In these experiments, once the reward buffer is filled, the policy is not protected and can be replaced anytime.

Still, the choice of the length for the reward buffer is not a neutral choice and will be further discussed in Chapter 6. With a buffer too small, it becomes impossible for the robot to correctly asses its policy as it "forgets" quickly its accumulated reward. On the other hand, with a too large buffer, the robot becomes insensitive to short timed positive reward, and learning becomes increasingly difficult. We set the size of the buffer to hold 250 measurements, defining the integration time to be $250 * 0.25 = 62.5s$.

4.2.2 Motion phase

The motion phase sets the vibration intensity for both the motors for a given amount of time, defining the two meta-behavior. The first behavior, expected to be chosen in the light, is to come to a complete stop by explicitly disabling the motors. The second behavior, expected to be chosen out of the light, is a behavior of run-and-tumble, a typical motion performed by bacteria [**?**] allowing to explore space better than the straight motion. The robot is programmed to move straight for a fixed amount of time $\tau_{run} = 2s$ and to tumble in a random direction with a duration picked randomly in an array simulating a long-tail distribution of mean $\tau_{tumble} = 4s$. This sets the total duration of a run-and-tumble sequence to be on average $\tau_{rt} = 6s$. The frequency at which the motors spin is fixed by a Pulse-With-Modulation signal of fixed value, which we arbitrarily set to 130/255. A typical trajectory is represented on Figure 50.Left.

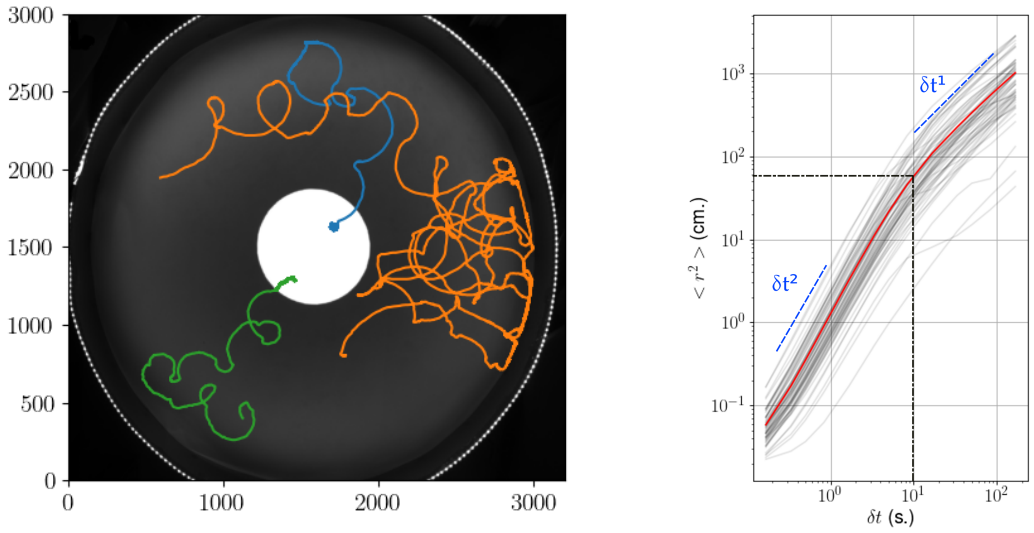


Figure 50 – **Left** : Examples of trajectories for the phototactic run and tumble motion. Blue and green trajectories are 2 hours long, orange trajectory has been cut to 12min. **Right** : Mean squared displacement of the Kilobot run and tumble motion. The two regimes of motion, ballistic at short timescales and diffusive at longer timescales are separated by a crossover which takes place at $\tau_p \approx 10s.$, the persistence time.

With the motion defined, we run a 2 hour experiment in which 64 robots perform the motion without the disc of light, with no phototactic interaction whatsoever. One can then measure the effective nominal speed of the Kilobots, as well as their effective diffusion constant. We consider N robots described by $\mathbf{r}_{i,t}$ the position in 2D space of the robot i at time t. We then measure the mean squared displacement $\text{MSD}(\delta t) = < \Delta\mathbf{r}^2_{i,t+\delta t} - \mathbf{r}^2_{i,T} >_{t,N}$ on Figure 50.Right. This quantity is the distance a single robot travels in a time δt, squared, averaged over all all robots and the duration of the experiment. On short time intervals, the displacement of the robots is aligned with the direction of their individual orientation and displacements are proportional to δt. The motion is said to be ballistic when robots go on a straight line at a constant speed : $< \Delta r^2(\delta t) >= v_0^2 \times (\delta t)^2$, where v_0 is the instantaneous speed. On long time intervals $\delta t > \tau_{rt}$, the robots have time to tumble and reorient and the trajectory is no longer ballistic. The motion is said to be diffusive $< \Delta r^2(\delta t) >= 4D_0\delta t$, where D_0 is called the diffusion constant. This diffusive motion corresponds to a random walk in 2D space, in which on average a robot walks away from its starting position proportionally to the square root of the time elapsed. The two regimes of motion are separated by a crossover, which takes place on a time scale called the persistence time, $\tau_p \approx 10s.$, to which corresponds a persistence length $l_p \approx 8cm$.

We use the same data to measure experimentally the correlation between the velocity and orientation. We note α the angle between the orientation vector and velocity vector. The measured density function of α for the 2 hours run and tumble experiment is plotted on Figure 51. We obtain the same profile as the self-propelled hard disks presented in Chapter 1.2.3 (on Figure 11), showing

a peaked distribution around 0, indicating that the local displacements of our Kilobots are mostly taking place along their polarity. The measured standard deviation $\sigma \approx 0.587$ is largely due to individual biases and mechanical noise in the different motions of the Kilobots. The effect of the bias is further detailed in the study of the individual dynamics in Chapter 5.3.3.

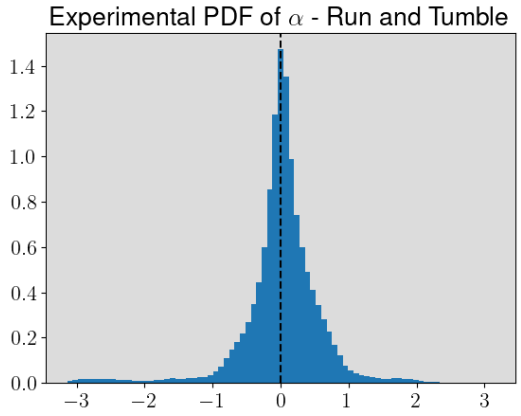


Figure 51 – Experimental measurement of α, the angle between velocity and orientation for Kilobots with aligner exoskeletons performing Run and Tumble.

4.2.3 Assessment

Finally, in the last phase, the Kilobot assesses its current state relative to its parameter w. It retrieves the last measured light intensity I from the reward buffer, and selects the run-and-tumble meta-behavior according to the criterion $I < w$. This way, the robot is moving only if it consider itself to be in the dark region. Considering the true light intensity of the region to be I_0 and the true light intensity of the lit region to be I_1, the value of w defines three classes for the robot long-time behavior, illustrated in Figure 52. In the case where $I_0 < w < I_1$, the robot has the expected good behavior of a phototactic agent, by coming to a stop in the light and exploring space in the dark. However, if w has a value too low $w < I_0 < I_1$, the robot never moves as it consider itself to always be in the lit region of space. In the same way, if w has a value too high, $I_0 < I_1 < w$ and the robot never consider itself to be in the lit region. In this case, the robot continuously perform the run-and-tumble motion. We call these 3 types of long-term behavior respectively the **Good**, **Dazzled** and **Blind** behaviors. When a robot encounters a peer, they exchange their parameters along with their total reward, and the parameter corresponding to the best reward is chosen to replace the parameter of the lowest reward. In this experiment and contrary to the classic version of the HIT algorithm, there is no protection window for the policy, and there are no mutations whatsoever.

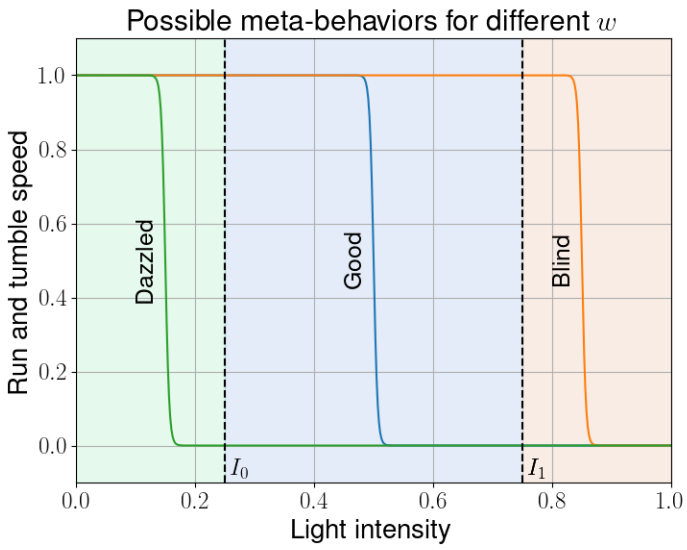


Figure 52 – Three classes of long-time behavior for the Run and Tumble phototactic learning experiment. The parameter w of each robot indicate the light intensity threshold at which an individual robot switches from complete stop to a run and tumble motion. Here, light intensity in the dark is arbitrarily set to $I_0 = 0.25$ and light intensity in the lit region is set set $I_1 = 0.75$. The three cases, $w = \{0.15, 0.5, 0.85\}$ define the 3 possible behavior, respectively the Dazzled, Good and Blind behaviors.

4.2.4 Results

In order to highlight the effectiveness of learning, we first run a series of experiments with only good parameters and we disable the communication between robots. We call this version of the experiment the "preset" experiment. The duration of a single experiment is 2 hours, with every robot randomly placed in the unlit region of the arena at the beginning. We measure the proportion of the swarm that comes to a stop in the lit region, and out of the lit region as shown on Figure 53. The results shows an inverse exponential growth to a plateau at $t = 3600s.$, with $\approx 70\%$ of the swarm stopped in the lit region. At this point, the outer layer of the lit region is filled, and no robot can fill up the remaining space. The remaining part of the swarm keeps exploring randomly the dark region. The two complementary measurements that can be made, respectively robots moving in the light and stopped in the dark are omitted. These measurements are not relevant in the preset case, as every robot is explicitly programmed to stop in the light and move in the dark. Nonetheless, it important to state that these quantity are never exactly zero, because of collisions, robots momentarily stuck against a wall, or robots pushing each others. Overall, this represents about 15% of the swarm at any given time in the experiment.

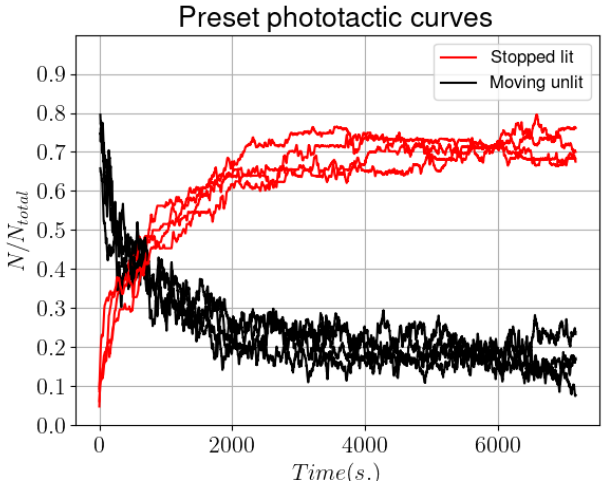


Figure 53 – Phototactic performance of a pre-calibrated swarm of N=64 Kilobots, performing a Run and Tumble motion while in the dark and stopping in the lit region. The red curve shows the proportion of robots correctly stopped in the lit region increasing as the inverse of an exponential and coming to a plateau at $t = 3600s.$. The opposite measurement is plotted with the black curve showing the proportion of robots still moving in the dark.

We now turn towards the learning experiment. Again at the beginning, every robot is randomly placed in the unlit region of the arena, and now draws randomly uniformly its parameter w from a set of 16 possible values, composed of 4 good values, 6 blind and 6 dazzled. Each robot then runs the HIT algorithm as described above, and the duration of a single experiment is 2 hours. During the experiment, the robots exchange their parameters every time they come close to one another. Their internal states however are not accessible. The possible measurements of motion state and the spatial location allows for an indirect measurement of the parameters, as summarized in Figure 54. It is important to notice that as robots start away from the light, the only way a dazzled robot can be seen in the light is if it has being pushed in by another robot. Because the collisions produce very little displacement, this case is expected to be quite rare. The measurements we make gives us insights on the distribution of the different policies in the swarm, but the true internal states remain inaccessible.

Again, we measure the proportion of the swarm that comes to a stop in the lit region, and out of the lit region as shown on Figure 55.left. The system behaves in a same way as previously, with the proportion of robots correctly stopped in the lit region following an inverse exponential growth up to a plateau. The dynamics is slower and the final score is lower, with a plateau at $t = 4800s.$, with $\approx 55\%$ of the swarm stopped in the lit region. The variance between successive experiment is higher in the case of learning. Because of the small scale of our system, we believe that initial conditions play an important role in the outcome of an experiment. Indeed, the initial distribution of parameters can lead to the vanishing of good policies. Because we do not allow mutations on the control parameters in this experiment, the disappearance of a policy is definitive.

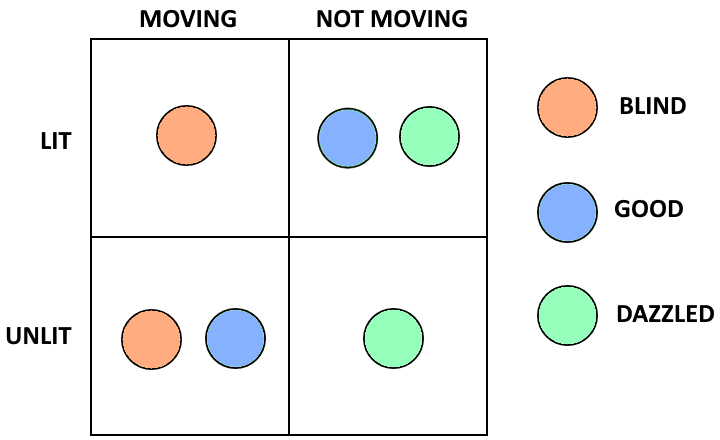


Figure 54 – Distribution of the three possible policies in a binary phototaxis learning experiment based on the switching in meta-policies. A robot with good threshold $I_0 < w < I_1$ is only moving in the dark region and stops in the lit region. A blind robot, with a high threshold $I_0 < I_1 < w$ is never stopping despite the change in light intensity. The opposite behavior, the one of a dazzled robot with a low threshold $w < I_0 < I_1$ makes the robot moves despite the change in light intensity. Because the dazzled robot never moves, the only way it can end up in the lit region is by being pushed in by other robots, it is not capable of being in the light on its own.

Still, the results show that distributed online learning mediated solely by social exchanges allow for phototaxis in a swarm of N=64 Kilobots. The typical learning curve as the total number of robots in the light grows rapidly at the beginning, as the inverse of an exponential, and comes to a plateau when the recruitment among the wrong policies is no longer possible.

Contrary to the preset experiment, it becomes interesting in the learning case to count the amount of stopped unlit robots and moving lit robots. The corresponding curves are shown on Figure 55.right. The proportion of stopped unlit robots varies a lot at the beginning of the experiment, and quickly stabilizes to a noisy constant at $t = 1200s$ from 30% to 50% of the swarm stopped. This is due to dazzled robots falsely propagating their policy out of the light. Indeed, in order to propagate good policies, the robots need to correctly assess their score, but this can only happen for the blind and good robots when they go through the lit region. Yet, when a good robot come into the light, it stops, and when a blind robot comes into the light, it receive the good policy from the robots already there and stops immediately. This makes the recruitment of dazzled robot difficult, since they can only be recruited by good robots that have been pushed out of the light or by blind robots that have never been in the light and are having a higher score in the dark thanks to noise on the measurement. This problem is usually solved by implementing mutations in the algorithm, preventing unreachable dazzled robots at the cost of lowering the maximum fraction of good policies.

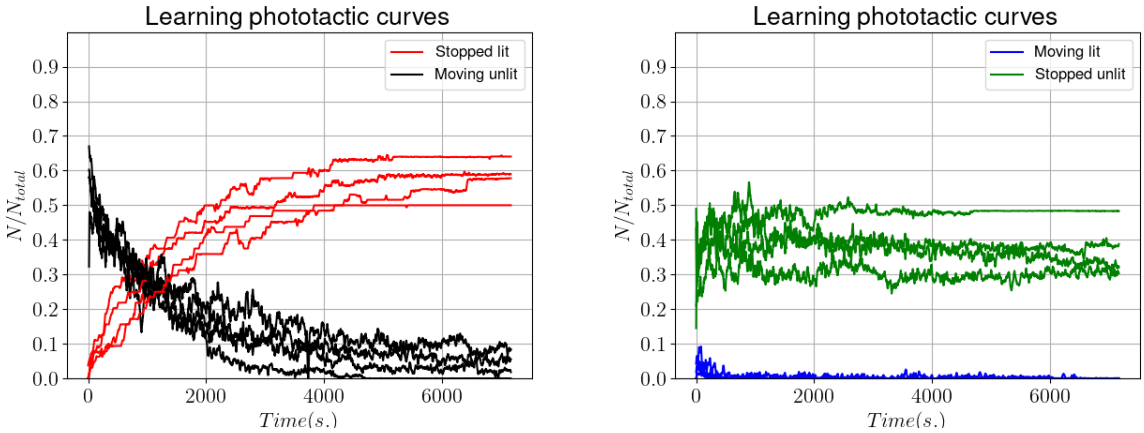


Figure 55 – **Left**.Phototactic performance of a swarm of N=64 Kilobots, performing embodied evolution through the HIT algorithm, with a Run and Tumble motion. The red curve shows the proportion of robots correctly stopped in the lit region. The dynamics is slower and the final score is lower than the pre-calibrated swarm, with a plateau at $t = 4800s$.. The opposite measurement is plotted with the black curve showing the proportion of robots still moving in the dark. **Right**. Secondary phototactic behaviors in a swarm of N=64 Kilobots performing embodied evolution through the HIT algorithm. The blue curve shows the proportion of robots moving despite being in the lit region. This represent a small fraction of the robots, about 1 to 3 at any time. The green curve shows the proportion of robots stopped despite being in the dark region. This part of the swarm is composed of robots stuck due to collisions, and dazzled robots having a threshold $w < I_0$, stopped and propagating their policy to incoming robots.

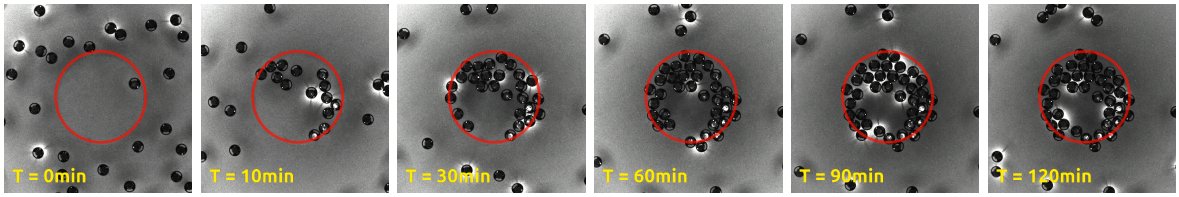


Figure 56 – Experimental realization of a phototaxis learning experiment. Some Kilobots stand still throughout the experiment, such as the Kilobot in the top left corner or bottom center, indicative of a Dazzled policy. Notice that some Kilobots emit light onto the floor of the arena. This is the reflection of infrared messages used for the communication, captured by the sensor of the camera.

4.3 Multi-Layer Perceptron inspired controller

In this second experiment, we evaluate the robustness of the HIT algorithm to a different type of controller, generating a much more diverse set of behaviors from a wider parameter space. The main limitations in the size and the complexity of the controller we can use on the Kilobot are the size of the memory and communication packets. As described in Chapter 3.1, the Kilobot sends 9 bytes of payload twice a second. Since we need to pilot two motors, we would prefer to have an

even number of parameters, leading to 8 one byte parameters. This makes a total parameter space of size 256^8, compared to the 3 previous possible behaviors. The two motors are each controlled by a single byte, which can now take any value between 0 and 255. This increases the action space from the two actions of meta-policies to 256^2 motor combination. The controller we decide to use is inspired by a simple Multi-Layer perceptron presented on Figure 57, composed of 4 neurons with their biases, receiving light intensity as an input and outputting directly the left and right motor PWM value.

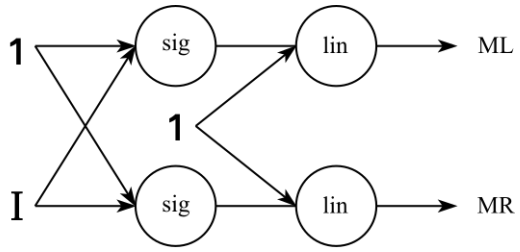


Figure 57 – Multi-Layer Perceptron inspiring the controller in use, 8 parameters in total.

This representation reads $M_L = a_1 * \sigma(a_0 * I + b_0) + b_1$ where σ denotes the sigmoid function $\sigma(x) = \frac{1}{1+e^{-x}}$. However, this representation is inconvenient regarding the scaling of each parameters. In order to obtain qualitatively different functions of the intensity, and to respect output bounds without flooring or ceiling, we change variables at our convenience. We choose the parameters as follows :

$$\begin{aligned} L &= b_1 \\ R &= a_1 + b_1 \\ S &= f^{-1}(a_0) \\ C &= -\frac{b_0}{a_0} \end{aligned}$$

Where f denotes a highly convex function from $[0, 1]$ to $[0, +\infty]$, to obtain qualitatively different slopes up to a step function. By doing so, each parameter constrained to $[0, 1]$ gives us the maximum amount of meaningful functions from $[0, 1]^4$ to $[0, 1]$. Here, L controls the left plateau of the function, R controls the right plateau, S controls the slope and C controls the center along the x-axis. Each motor final value can be written : $M = (R - L) * \sigma((I - C) * f(S)) + L$. We parameterize both motors using this procedure, giving a total of 8 parameters.

In our experiments, we choose the function f as $f(x) = (4 * x)^4$. The maximum value of this function on $[0, 1]$ is 256, which is large enough to obtain the sharpest possible function, considering that every parameter is encoded on 1 byte. Examples of different motor value as a function of light intensity for this controller are represented on Figure 58. The expressiveness of this controller allows for constant speed independent of light received as well as positive and negative phototaxis behaviors with deceleration or acceleration in the light.

In order to test the robustness of the Kilobot HIT swarm, we perform multiple sets of experiments while gradually increasing the difficulty of the phototactic task. As briefly stated in the introduction of this chapter, we expect *a priori* the difficulty of the task to increase as the proportion of good policies over the total amount of accessible policies diminish. Indeed, as we perform direct policy search, changes in policies are driven by stochastic processes, gradually exploring parameter space through trial, assessment and error. For example in the beginning of an experiment, the parameters are uniformly distributed in an hypercube of dimension 8 and side length 1, and the chance of starting with at least one good policy scales as $1 - (V_{bad})^N$ where V_{bad} denotes the volume of "bad policies". In our case, what could be considered a good or a bad policy is unknown. However, we can safely assume that the volume of good policies can be reduced by having a lower contrast between the lit and unlit regions, and augmented by having a greater contrast.

This can be understood through a continuity argument. Let assume for this argument that a policy is considered good if the motor final value in the lit region is strictly lower than the motor final value in the unlit region. If we consider the extreme case $I_0 = I_1$, meaning the contrast is the lowest, there exist no good policy and the volume of good parameters is 0. On the contrary, if we consider the case $I_0 = 0$ and $I_1 = 1$, meaning the contrast is the highest, the policy is considered good as long as $R < L$, irrelevant of the other parameters. This makes the volume of good

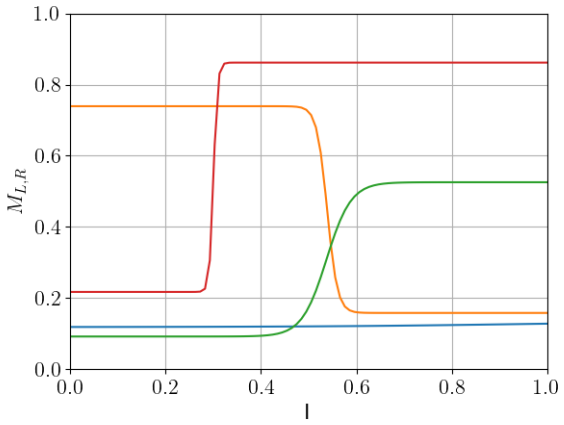


Figure 58 – Examples of possibles motor value function of light intensity. The Y axis corresponds to the intensity sent to either motor Left or Right. On the X axis, the normalized light intensity instantaneously received by a robot. The blue curve correspond to a constant behavior, not affected by changes in the environment. The yellow curve corresponds to the behavior expected for a positive phototaxis, in which the velocity decreases for a sufficient amount of light received. The two other curves, red and green, would corresponds to the behavior expected for a negative phototaxis, with robots accelerating in lit regions of space.

parameters equal to 0.5, with exactly half of the parameters giving a good policy. Since the policy is a continuous function of every parameter and of I, we expect the task to be increasingly more complex as the contrast between lit and unlit is lower.

Using the previously described controller and a protection window duration of 4 minutes, we perform the 3 following experiments :

- High contrast phototaxis with a uniform placement, initially in and out of the light.
- High contrast phototaxis with an unfavorable placement, initially out of the light.
- Low contrast phototaxis with an unfavorable placement, initially out of the light.

In the same way as the run-and-tumble experiment, we measure the proportion of Kilobots in the light over time as the main performance indicator. The measurements are shown on Figure 59. Compared to the case of learning with meta-policies, we find here a greater variance in the performance over the course of individual runs. The increase in the achievable motor values lead to a wide range of possible velocities and radii of gyration, which could explain this variance. Still, the case of High contrast with a uniform placement gives comparable behavior and performances to the meta-policies, with a final proportion of the swarm in the light ranging from 40% to almost 60%. This result shows the capacity of the Kilobot swarm using the HIT algorithm to adapt to new, more complex controllers while keeping good performances. Although contrary to the previous setting, the robots start off in the lit region and the performance is approximately 10% at the very beginning of the experiment. This advantage helps to bootstrap the learning process, by quickly identifying the good policies among the robots that start in the center.

Starting with robots out of the light produces a significant drop in performances, with a mean final proportion of robots in the center around 30%. Here, we observe big drops in performance happening over time in each individual experiment. This is due to the propagation of policies that slow down the robots in the light without stopping, increasing momentarily the performance until all robots slowly make their way out of the light. This effect is particularly clear in one of the runs for which the performance drops to 5% at the end of the experiment, after a transient period with 25% of the swarm in the light. During this run, the policy propagating the most consisted in performing small circles in the lit region. Doing so, robots have managed to accumulate in the center but then started to diffuse out, until eventually almost every robot has left the disc of light.

The final configuration, with a low contrast and off-center placement yields more consistent results with poor performance, and a mean final proportion of robots in the swarm at 15%, about 9 robots. Compared to the baselines, this configuration perform better than random motion, but is at the limit of what can be reasonably called phototaxis. This configuration makes the task too hard to achieve in the 2 hours of the experiment, with the controller and algorithm parameters we chose to use. Either the learning dynamics is too slow compared to the duration of the experiment, or the system gets trapped in a state in which policies are not good enough and cannot be improved. This suggest running experiments in the same configuration but allowing spontaneous mutations of the control parameters, to test the hypothesis that good policies may have disappeared in the whole swarm at the end of these experiments.

In a second set of experiments, we try to roughly evaluate the impact of the protection window on the phototactic performance. In the following, we note τ_p the duration of the protection window. This quantity corresponds to the time in the HIT algorithm for which the current policy of a robot is being evaluated and cannot be replaced. The choice of this timescale is dependent on the task

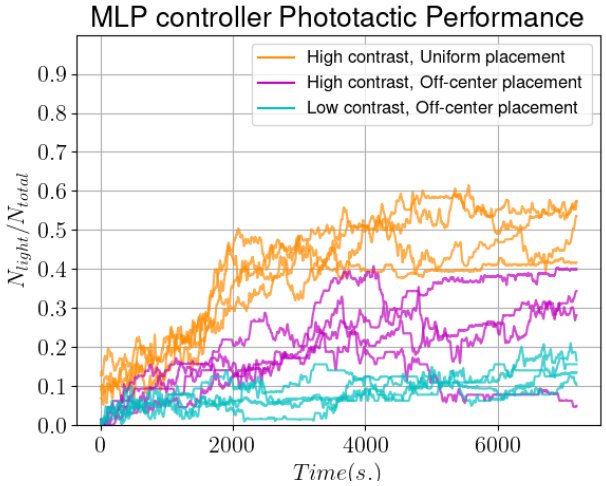


Figure 59 – Proportion of the swarm in the lit region for the MLP inspired controller and three different task difficulty. The learning dynamics is here slower than the learning with meta-policies, and present a much higher variance over the course of a single run.

and other typical scales of the system, as the duration of this window needs to be long enough to produce a total reward accurately assessing the policy being tested, and small enough such that enough variations can occur during the total duration of the experiment, in order to find better policies.

A first step to obtain an idea on the expected range of good values for τ_p is to consider the typical collision time, since the protection window acts on the communication between robots which only happens in close vicinity. The mean free path λ of a particle of cross-section σ moving in a straight line in an uniform medium of density ϕ satisfy the relation [8] $\phi\sigma\lambda = 1$. Considering N Kilobots of diameter d running at a typical velocity v_0 in a circular arena of surface area A, we get $\lambda = A/(Nd)$ and the typical collision time is $\tau_{coll} = \lambda/v_0 = A/(Ndv_0)$. As it is difficult to determine a typical velocity using this controller, we choose $v_0 = 1cm.s^{-1}$ which is in the same order as the measured mean velocities in the baseline experiments. This yields a typical collision time $\tau_{coll} = 55s.$, indicating that a reasonable protection window duration should be close to a minute.

We perform 4 independent runs with three different values for τ_p : 15 seconds, 1min and 4min. Each run has a duration of 2 hours, which allow theoretically the policy to change 480, 120, and 30 times respectively. In the code, we keep the same sampling rate for the light for each τ_p and only change the size of the reward buffer. Regarding the lighting configuration, we use the high

8. This relation holds as long as the particles in the medium are stationary. When considering that every other particle has a non negligible speed, the relation changes slightly and a factor $\sqrt{2}$ appears on the left side when considering an ideal gas and Maxwell-Boltzmann speed distribution. Because we are using the relation to get a scale, we overlook factors of order 1.

contrast with off-center placement. All the data gathered for these experiments are new. We plot on Figure 60.left the performance over time for each run, and on Figure 60.right the distribution of performances starting at $t = 3600s$ aggregated for the different runs with the same τ_p. The results show a clear improvement in the performance for $\tau_p = 1min$ compared to the two other values, with a mean performance of ≈ 0.4. Moreover, it seems that increasing τ_p do not only slows down the aggregation dynamics but changes the behavior of the swarm completely. In the 15 seconds case, the variance in performance in time and over different experiments is the greatest, with an amplitude in performance ranging from 0.05 to 0.4. In the 4 min case, in 3 out of 4 runs the performance seems to plateau around 0.25 starting at $t = 4000s$, which suggests that this configuration is not a slowed down realization of the experiments ran at lower τ_p. In the end, the experiments show that an optimum in τ_p exist, close to a minute for this experimental setup. Additional experiments varying the swarm density, velocities or light area would help determine the relevant physical parameters acting on this optimum.

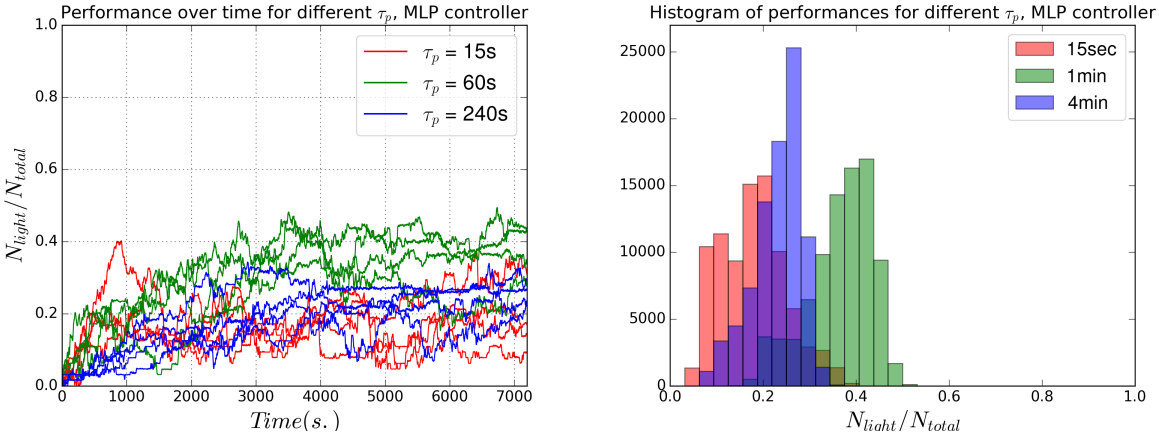


Figure 60 – **Left**. Performance over time for the MLP inspired controller running with different duration of protection window. **Right**. Histogram of the performances aggregated over the different runs for $t > 3600s$. Setting $\tau_p = 1min$ outperforms the other values, in this experimental configuration.

5 Morphological Computation

5.1 Designing the morphology

5.1.1 Motivations

In Chapter 1.3, we present the role of morphology in the design of a robot, allowing to reduce the computation required to accomplish a task, by having interactions at the physical level add a layer of "morphological computation". In our case, we are interested in the role of morphology in a swarm of Kilobot robots. In particular, we equip Kilobot with two different exoskeleton encasing the robot, the **aligner** and **fronter** exoskeletons, presented first with the experimental setup in Chapter 3.2. The two designs replace the default legs of the Kilobots, and by changing the contact points to the ground and the mass distribution, both show a re-orientation response to external forces.

We study the case of a phototaxis experiment, the same task done in the case of learning experiments. In this situation, robots have to aggregate in the center of the arena, rending physical contact unavoidable. We begin by introducing the two design more in depth, and present the effective physical model associated with the corresponding dynamics. Next, we perform multiple series of real world experiments with N=64 robots to study morphological effects at the swarm level. Finally, we expand the experiments *in sillico* running simulations built on the physical model.

5.1.2 Aligner and Fronter exoskeletons

In line with the Science Robotics paper by Ben Zion et al. 2023 [**?**], we call the system Kilobot + exoskeleton a *Morphobot*. Because Morphobots re-orient when subject to external forces, their behavior differ in the interaction with stationary walls and with one another. When a Morphobot runs into a wall, the collision repels the robot and triggers its re-orientation. The whole interaction consists in a series of repeated collisions and with each collision, the Morphobot slightly re-orients. An aligner orientation approaches the normal force from the wall (but leaves when parallel to the wall) Figure 61-a. A fronter turns against the normal force (and towards the wall) Figure 61-b. This is confirmed by the direct observation of Morphobots sent to move towards a wall of stationary robots. Arranging turned off robots into three, tight rows forms an effective wall with which a moving robot mechanically interacts. On average, both designs spend similar time near the wall (≈1 min) but aligners travel a path along the wall which is twice longer than that of the fronters (see Figure 61-c). In the case of binary collisions, the main difference between the two designs is the typical duration of contact between the two Morphobots. As the aligners reorient away from the barycenter of the two robots, the fronters align towards it. Because of orientational and translational noise, both designs eventually escape contact.

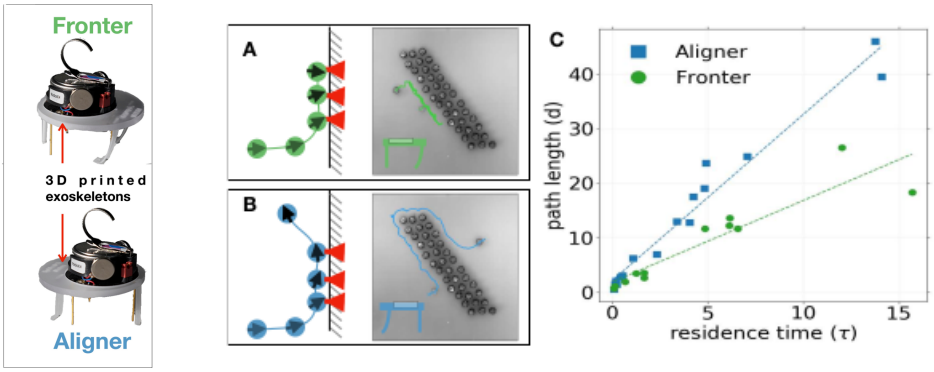


Figure 61 – When a Morphobot collides with a wall, each impingement can be seen as a momentary normal force. **A** When a fronter collides with a wall, its orientation (black) turns anti-parallel with the normal force (red), and the robot continues to push into the wall. **B** Aligners turn to align their orientation (black arrow) with the normal force (red). Once aligned along the wall, there is no further rotation as there are no more collisions, and the robot slides along.

5.1.3 Inclined Plane experiment

The force-orientation response is also well illustrated when the robots are subjected to gravity, by moving on a slightly inclined plane. The aligner polarity tends to align with the direction of gravity, quickly falling down the plane, as the fronter tends to anti-align with the force and climbs up the plane. For both exoskeletons, we film 5 trajectories, starting with a direction perpendicular

to gravity from the right and the left of the plane. The trajectories are represented on Figure 62. As the robots are falling, their velocity becomes slightly misaligned from their orientation. Because of the morphology of their exoskeleton, they react to this decoupling by re-orienting towards or against the instantaneous velocity that comes from the effect of gravity. The active force that propels them forward is strong enough to overcome gravity in the case of the fronter, and the robot climbs up the plane. One can notice that this re-orientation is not perfect, as the final angle to gravity is not zero. Indeed, all Kilobots and all exoskeletons are not exactly built the same, and the small imperfections in the making produce a bias in the orientation of robots. Some Kilobots and exoskeletons tend to turn slightly to the right when having both motors set to the same PWM signal on a flat surface, and some tend to turn slightly to the left. On an inclined plane, this bias compensates the re-orientation due to gravity and the final angle is non zero.

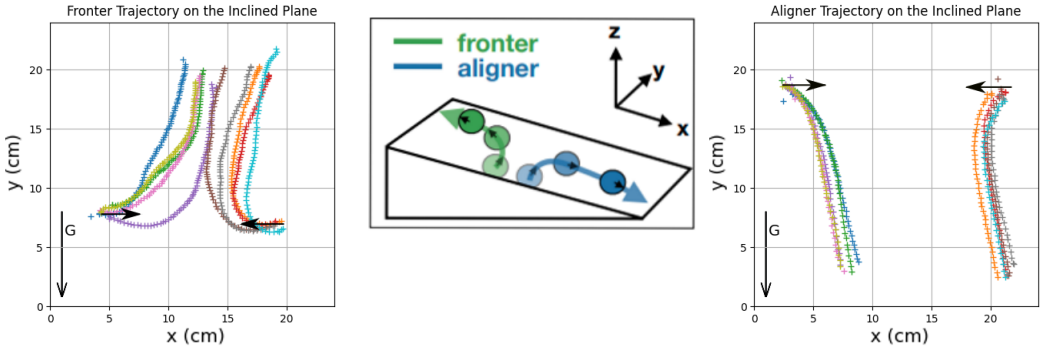


Figure 62 – Examples of trajectories of the fronter and aligner exoskeletons on a plane inclined downwards by 4 degrees.

5.1.4 Modeling morphology through self-alignment

In order to describe the motion of the Morphobots, we expand upon the model of self-alignment used to describe self-propelled hard disks, presented in Chapter 2. For now, we keep the same dimensionless equations exposed earlier, assuming the dynamics on $\mathbf{n}$ is over-damped. This model describes well the dynamics of the aligners, as it makes few assumptions and display the same phenomenology as our robots : particles escape walls and re-orient on an inclined plane. We perform one addition to this model, considering that the self-alignment torque (orienting the direction toward the velocity) can also take negative values. This simple change expand the model to include the phenomenology of fronters, including self-trapping against walls and climbing up the inclined plane. We therefore consider the following set of equations :

$$\begin{cases} \tau_v \dot{\mathbf{v}} = \mathbf{n} - \mathbf{v} + \Sigma \mathbf{f} \\ \tau_n \dot{\mathbf{n}} = \epsilon(\mathbf{n} \times \mathbf{v}) \times \mathbf{n} \end{cases} \tag{3}$$

We add one parameter to the equations in front of the self-alignment torque, ϵ[9], which allows to describe the dynamics of aligners setting $\epsilon = +1$ or fronters setting $\epsilon = -1$. In the same way as presented before, angular noise is applied after the self-aligning torque at the end of each time-step. This allows to set $\epsilon = 0$, producing the same effect as $\tau_n \to \infty$, creating ABPs.

We are left with 3 parameters that control the dynamics of the robots, including ϵ. The parameter τ_v pilots the inertia of the robot. As the damping increases and becomes large enough, τ_v decreases to 0 and velocity aligns infinitely fast with the orientation and external forces. Finally the last parameter, τ_n, defines the strength of the auto-alignment torque, which makes the orientation align (or anti-align) to the velocity. With a large τ_n the robot's orientation becomes increasingly insensitive to forces, while not affecting the strength of the angular noise. As τ_n approaches infinity, this creates ABPs as the orientation is purely driven by noise. With a small τ_n, the response of the orientation to external forces become very strong, with a limit at 0 where the robot orientation aligns infinitely fast with the velocity.

This model allows us to capture most of the effects observed with the Morphobots. When simulating soft discs obeying these equations, we retrieve the effects described in the previous paragraphs.

5.2 Effect of morphology in a phototactic collective task

We consider again the phototactic task presented in Chapter 4.2, where a swarm of $N = 64$ robots must aggregate in a disc of light at the center of the arena. We run the same experiment, with Run and Tumble meta-policies as a controller. We are now interested in showing the role of morphological effects on the capacity of the whole swarm to accomplish the task. In previous experiments, the two meta-policies corresponds to two desired behavior, one of exploration and one of exploitation, in a sense that a robot with a good threshold explores in the dark by performing a Run and Tumble motion and exploits the light by standing still in the light and maxing out its reward. As to better show the role of morphology, we run a new series of experiments with a different *exploit* meta-policy, for which we impose the robot to keep a Run and Tumble motion but at a lower speed. This gives us the two configurations :

- Exploit *Still*, with a velocity in the light $v_1 = 0$
- Exploit *Move*, with a velocity in the light $v_1 = \frac{1}{5}v_0$

For the two morphologies, Fronter and Aligner, this gives a total 4 of different experiments.

5.2.1 Experimental Results

In order to isolate the effects of the morphology, together with steric interactions, we start by setting manually the threshold as we did in the first set of phototaxis experiments. We measure the proportion of robots in the light over time, and summarize the results on a table of plots shown in

9. In the dimensionalized equations, ζ is no longer considered positive, and ϵ is defined as $sign(\zeta)$. This slightly changes the non-dimensionalization, with the replacement $\zeta \to |\zeta|$ in the parameters definitions.

Figure 63. On the left side of the table, in the *still* configuration robots come to a complete stop in the light. Results for the aligner exoskeleton are the same as in Chapter 4.2Learning with Run and Tumble meta-policiessubsection.4.2. Both the aligner and fronter exoskeleton show an inverse exponential growth coming to a plateau at $\approx 65\%$ of the swarm in the light around $t = 3000s$.. With our exoskeletons and the scale of the experiment, the difference in the dynamics of fronters and aligners shows little difference in the preset task. We observe the slight formation of an annulus with the aligners positioning themselves at boundary of the lit circle, shown on Figure 65. This is an effect due to the robots stopping at the very moment they enter the light, preventing other robots from entering. In the case of aligners, the robots tend to escape this obstructing layer, as fronters tend to force themselves in. This effect becomes more clear in simulations, by augmenting the "alignment strength" $(\frac{1}{\tau_n})$ or augmenting the number of robots.

Looking at the right side of the table in the *move* configuration where the speed in the light is non zero ($v_1 = \frac{1}{15}v_0$), the difference between fronters and aligners becomes very clear. The aligner exoskeleton fails to perform a significant phototaxis, with a proportion of the swarm fluctuating between 10% and 30% over the course of a run with a mean of $\approx 15\%$, equivalent to 9 robots. The local density in the light is higher than the one in the dark because of the lower speed, but the robots tend to escape the lit region with steric interactions and alignment acting against aggregation. In the case of the fronter exoskeleton, the results shows a mean of $\approx 35\%$ of robots in the light from $t = 1800s$, with a higher variance over a single run compared to the still version. Here, re-orientation effects play a role that favors aggregation, with collisions generating forces and re-orienting Morphobots towards the crowd.

An illustration of this dynamic cluster formation by fronters is shown on Figure 64. Physically, this can be seen as the creation and maintaining of a droplet in the lit region in contrast with the gaseous phase in the dark region, with evaporation and condensation acting at the same time to maintain the droplet : robots come and go out of the light but get trapped by the steric interactions. These results show morphological computation at work, in which the design of the exoskeleton allows for effective phototaxis despite the constraints at the software level. One could say that slowing down in the light creates a region of finite density, where crowding effects are prone to induce aggregation via the Motility Induced Phase Separation (MIPS) mechanism. Because of the difference in their morphology, the fronter takes advantage of this effect at the rather low densities reached in the light while the aligners do not.

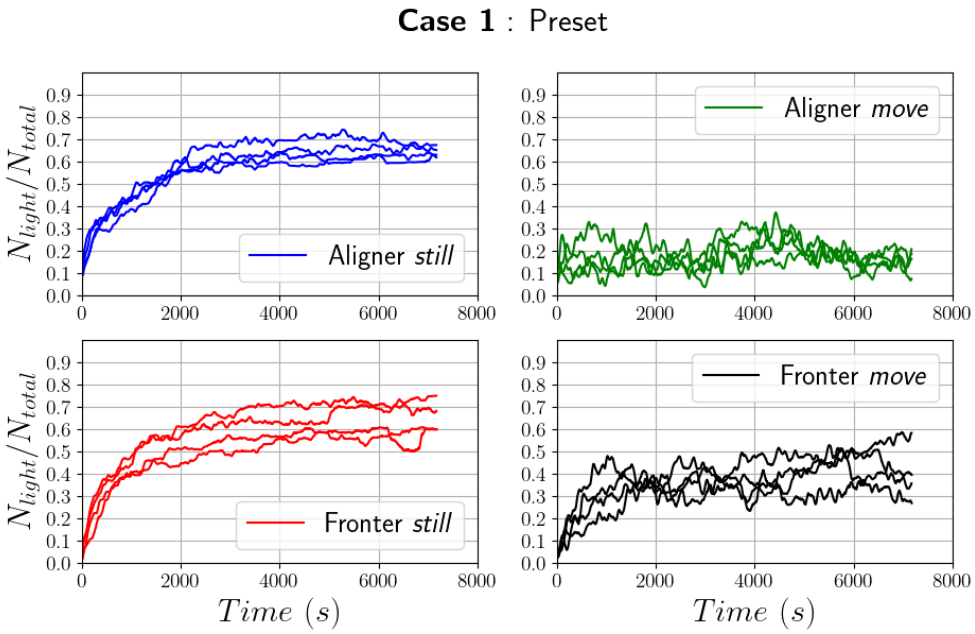


Figure 63 – Total aggregation of Morphobots in the light for the different Preset configurations.

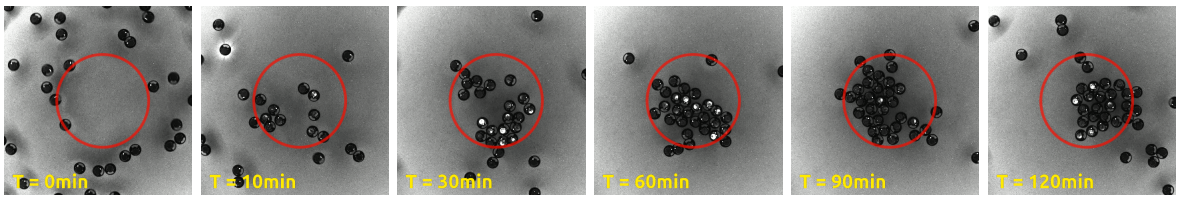


Figure 64 – Experimental realization of Fronters in the *move* configuration, unable to stop in the center but aggregating due to the re-orientation effects happening during collisions.

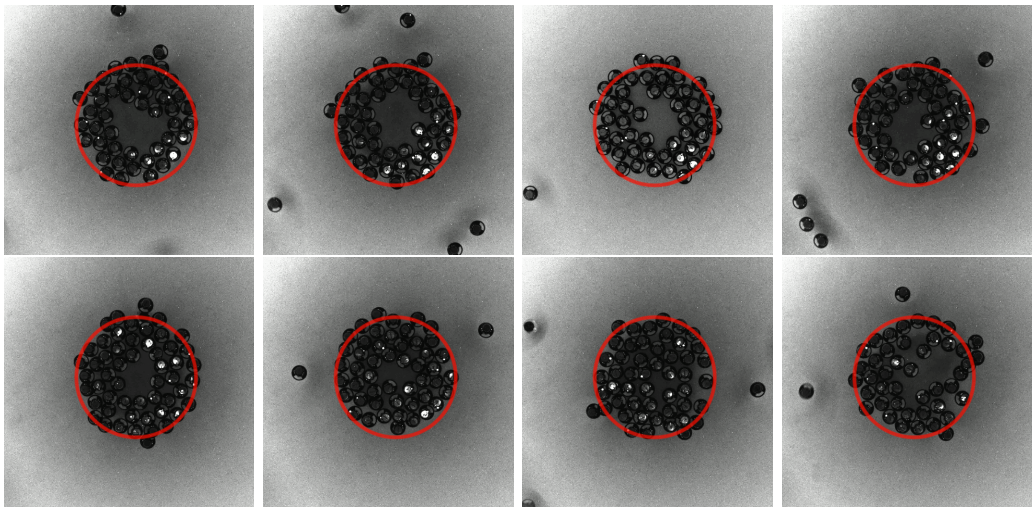


Figure 65 – Final configurations of the Preset Still experiments, aligners on top and fronters on the bottom. Except for the fourth fronter experiment, aligner show a slight tendency to form an annulus around the lit region. This effect is low due to the low "alignment strength" $(\frac{1}{\tau_n})$ of the exoskeletons, but can be explained with the self-alignment model and tested *in sillico*.

As a continuation of the learning experiments presented in Chapter 4, consisting in an adapted version of the Aligner Still configuration [10], we perform the same experiment with the remaining three configurations : Aligner Move, Fronter Still and Fronter Move, where "Move" specify that the velocity in the lit region is greater than zero. The goal is the same as before, with a threshold value shared through the population, indicating the amount of light at which a single robot switches from the *explore* to the *exploit* meta-policy. We measure the number of robots in the light over time, shown on Figure 66.

Results shows that in the Still configuration, Morphobots are able to learn indifferent of re-orientation effects. For both the aligners and fronters, the aggregation dynamics is slower than in the preset, with a plateau reached around $t = 4000s$ in most of the experiments. The performance in learning is good compared to the presets, with a final proportion in the swarm comprised between 50% and 60% for the aligners and between 40% and 60% for the fronters.

In the Move configuration, the aligners fail to aggregate in the center, similar to their preset counterpart. This result can be expected, since learning adds a layer of complexity and adds difficulty to accomplish the task, which would not help the aligners to succeed. By imposing a non-zero velocity in the light, we force the re-orientation effects to play a role in the aggregation, creating too much of a constraint for the aligners to solve the task. This effect plays in favor of the fronters, which still manage to aggregate in the center, showing a maximum final performance similar to that of the presets, up to 50%. The variance on this configuration is however greatly increased, which

10. The data shown for learning in the Aligner Still configuration is the same data presented in Chapter 4. A subtle difference in the measurement is important to precise, as we count here directly the number of robots in the light without considering if the robot is moving or stopped, unlike in Chapter 4.

is why we performed a set of ten realizations instead of the usual 4. Here, the interplay between learning and morphological effects leads to experimental realizations with progressive aggregation, aggregation and dissolution, or no aggregation at all. These results suggest that the internal degrees of freedom of the robots can act as a trigger for nucleation, creating a bi-stable system which regime can switch due to the presence of a global consensus in the swarm.

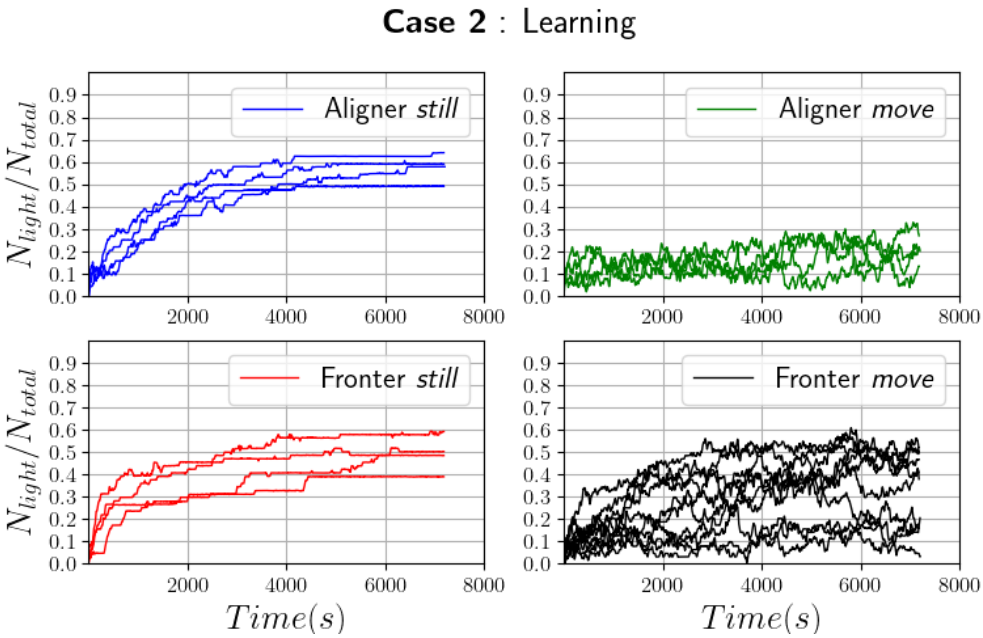


Figure 66 – Total aggregation of Morphobots in the light for the different Learning configurations. The task and algorithm deployed are the same as those presented in Chapter 4.

5.2.2 Simulations

We expand upon experimental results by simulating additional configurations. In the experiments, the two exoskeletons lead to a different set of dynamical parameters. A first assumption is to assume $\tau_v \to 0$, our robots do not show inertia in their speed. When the motor signals drop to zero, the robot stops immediately with no delay, a reasonable assumption to make considering the direct observations made in the lab. This leaves two parameters for the exoskeletons, ϵ the re-orienting torque sign and τ_n the relaxation time of the orientation to the velocity. Since $\epsilon = 0$ and $\tau_n \to +\infty$ produce the same effect, we consider the quantity ϵ/τ_n as the relevant parameter driving morphological effects. When this quantity is positive with a high absolute value, we model super-aligning particles. When it is negative with a high absolute value, we model super-fronting particles. Around zero, we model ABPs and low aligning or low fronting particles, which is the range corresponding to our robots.

We set the diameters of simulated particles to 1, and for practical reasons we forget the walls in the simulations. Instead, we consider a box with periodic boundary conditions of side length $L = \sqrt{A_{arena}} = 27.7$, where A_{arena} is the total area of our circular area given in squared robot's diameter [11]. We re-scale the lit region with the same procedure, keeping the surface ratio A_{arena}/A_{light} equal in simulations and in experiments. The natural unit of time in our simulated system is τ, the time taken for a single particle moving in a straight line to move one diameter forward. Particles interact only through a pair potential, with a repulsive WCA potential with standard exponents $n = 6$ (see Figure 28 in Chapter 2). One difference between simulations and experiment is the initial conditions, with particles randomly distributed in and out of the light in the simulations, contrary to experiments in which robots start only out of the light. As mentioned before, we apply an angular noise at the end of each time-step, with a diffusion coefficient D_θ equal to 0.02.

We perform a series of two numerical experiments, varying the ratio ϵ/τ_n uniformly between -10 and +10. The first experiment consist in 64 particles, as an analogy to the experiments, performing phototaxis in the Still configuration. In the second experiment, we perform the same task with 256 particles, with a box and light re-scaled to keep the density N/L^2 constant. We run 40 independent realizations of duration 1000τ, and plot the half-filling time in Figure 67. This is the time it takes for the system to reach exactly half of the particles in the light region. Snapshots of these simulations are shown on Figure 69.

For the two system sizes, we observe the same phenomena. Around zero ϵ/τ_n, particles are slight aligners or slight fronters. Both manage to fill in the lit region at a close rate, with a little advantage for the fronters. Snapshots of two realization at $\epsilon/\tau_n = \pm 1$ for 256 particles are shown on Figure 69.1-2. The slight advantage of the fronters come front the same experimental observations made with Kilobots : fronters have a tendency to "push their way in" when coming in contact with the aggregate in the center, as aligners have a tendency to form an outer layer preventing the arrival of new particles. As ϵ/τ_n grows and particles become more aligning, this difference becomes clear with the formation of a more and more unpenetrable outer layer, Figure 69.3. On the other

11. With $R_{arena} = 75cm$ and $d = 4.8cm$ this yields $A = \pi \times \left(\frac{75cm}{4.8cm}\right)^2 = 767$, hence L= 27.7.

side of the graph, at negative ϵ/τ_n, when anti-alignment becomes too strong, particles get stuck on the first collision, and the system instantly cluster out of the lit region, shown on Figure 69.4.

The cases of strong anti-alignment are not interesting to perform a phototaxis task, but they display an interesting clustering dynamics that could be studied more in-depth. The cluster formation depends both on the anti-alignment strength $1/\tau_n$ and the angular noise D_θ, set to 0.02 in the simulations presented above. A first observation that can be made by simulating a system of fronters with no light is the gradual formation of smaller and smaller clusters, up to a point at which particles form dimers and trimers but no bigger cluster can be found, Figure 68.**Left**. This suggest that these two configurations might be the only stable configurations for the assembly of fronter particles. Before reaching the final state, one interesting transient configuration is often observed, with a single fronter pushing a dimer, shown on Figure 68.**Right**. This produce a line of three particles moving, with a rather straight trajectory compared to individual particles (the configuration is less subject to angular noise). When taking extreme values of $1/\tau_n$ with respect to noise, many other different types of clusters seem stable since every particle stops at the first collision. More theoretical work would be require to understand clearly the stability of these assemblies.

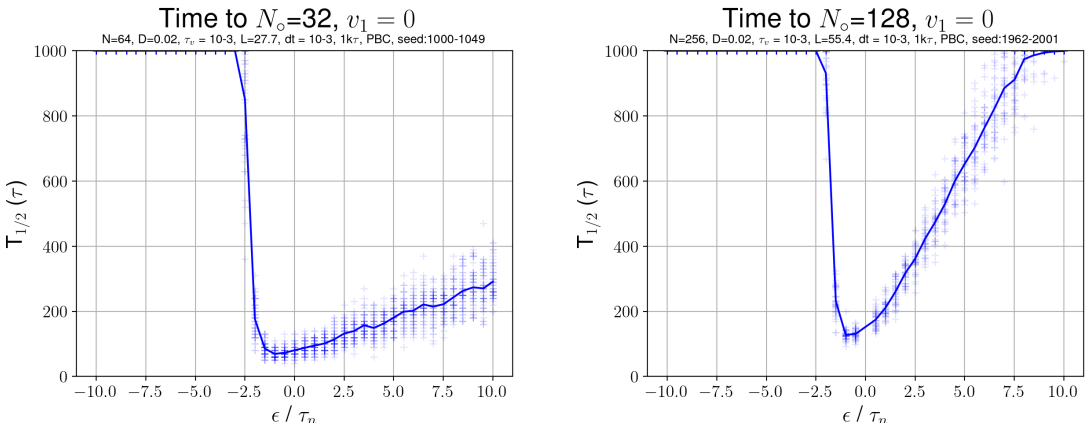


Figure 67 – Half-filling time of the lit region for simulations in the Preset Still configuration, with 64 total particles on the left and 256 total particles on the right. Particles manage to aggregate faster when they are slightly fronters, $\epsilon/\tau_n \approx -1$.

Figure 68 – **Left**. Numerically stable configurations of fronters for $\tau_n = 0.2$. Two configurations are found, dimers and trimers, with proportion roughly 2 for 1. When decreasing τ_n below 0.1 while keeping the other parameters constant, particle freeze immediately after the first collision, probably locking the system in a configuration of higher energy. **Right**. A transient configuration of 3 fronters spontaneously appearing while the system is stabilizing, with a fronter pushing a dimer. This configuration stabilizes the trajectory compared to single particles, clearly shown by numerical simulations at low density with a high value of angular noise.

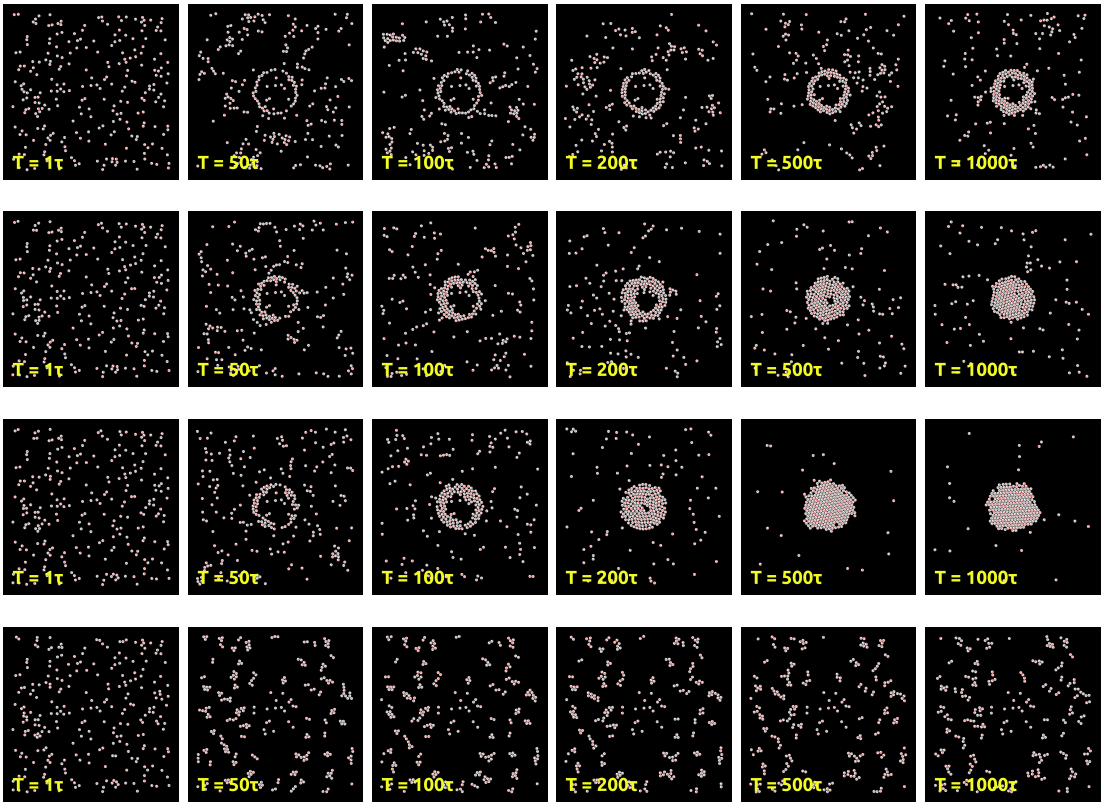


Figure 69 – Snapshots of simulations for 256 particles in the Preset Still configuration. From top row to bottom, decreasing ϵ/τ_n : **1.** Aligners $\epsilon/\tau_n = +10$ showing the formation of an outer layer ; **2.** Aligners $\epsilon/\tau_n = +1$; **3.** Fronters $\epsilon/\tau_n = -1$; **4.** Fronters $\epsilon/\tau_n = -10$ showing a rapid cluster formation, freezing the system in place.

We also perform simulations corresponding to the Preset Move configuration. However, we do not keep the same density and speed ratio in the simulations compared to the experiments. When running simulations with the same density and same speed ratio $v_1 = \frac{1}{15}v_0$, the morphological difference cannot be seen in the range of low ϵ/τ_n, and high values of ϵ/τ_n produce instant clustering for the fronters and collective motion for the aligners, which we would prefer to avoid. In addition, the effective density in experiments around the center is lower than N/L^2 because of robots aggregating to the walls. In the light of these considerations, we perform simulations with half the density as in the experiments, and a speed ratio $v_1 = \frac{1}{5}v_0$. We perform 100 realizations with a duration of 1000τ, and measure the final proportion of particles in the light. Histograms of these measurements and gaussian estimates are shown on Figure 70, for aligners on the left and fronters on the right.

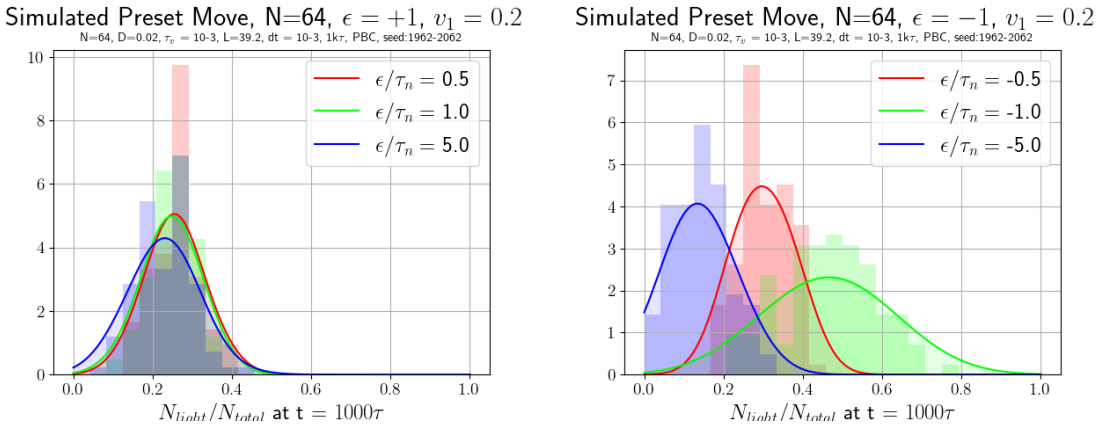


Figure 70 – Final proportion of particles in the light for N=64 Aligners (left) and N=64 Fronters (right) performing phototaxis in the Preset Move configuration. Like in the experiments, aligners fail to aggregate in the center, and fronters manage to dynamically maintain a cluster in the center, when $|\epsilon/\tau_n|$ is not too large.

Like in the experiments, aligners fail to aggregate in the center due to the re-orientations effects. This stays true for different values of ϵ/τ_n, with a mean final score around 0.25. Because the particle slow down, the density in the light is higher than the density out of the light, but morphological effects play little role in the final score. In the case of the fronters however, we recover the result of the experiments with $\epsilon/\tau_n \approx -1$, for which the particle dynamically maintain a cluster in the center and achieve a mean final score of 0.5. Here again, the simulations point to an optimum in ϵ/τ_n with the mean score of $\epsilon/\tau_n = -0.5$ being close to the aligners because of the low intensity of the morphological response, and the mean score of $\epsilon/\tau_n = -5$ being even lower, about 0.15, because of the rapid formation of clusters in and out of the light.

5.3 Going further with morphological effects

As presented in Chapter 3.2, a third exoskeleton design other than the fronter and aligner displayed interesting morphological properties. This is the toothbrush Kilobot, a version in which rigid legs are replaced a set of two toothbrushes acting as multiple soft legs. This design showed two unseen

phenomenon : an oscillation against walls with an amplitude driven by speed and a state of pinned rotation, in which the robot gets stuck spinning on itself. The unseen dynamics shown by this design pushed us to get a closer look on the set of equations describing self-aligning particles, to verify if the model was able to capture this phenomenology. We decided to also include the effect of bias, and a number of unpredicted dynamics were observed.

5.3.1 Biased Exoskeletons on the Inclined Plane

Each robot with an exoskeleton possesses its own natural bias to turn in a preferred direction. This makes the robots not follow a straight line but perform a circular motion, with an important radius of gyration. This effect is not only true for the toothbrush exoskeleton, but for all other exoskeletons and similar vibrating systems such as Hexbugs. When put on the inclined plane, it may happen that the intensity of this bias exceed the torque applied by the effect of gravity. This transforms the straight trajectory into cycloids, which are best observed with the toothbrush exoskeletons. Some trajectories for a single robot are shown on Figure 71. This effect disappears when the strength of the gravity becomes too important, and the trajectory becomes a straight line.

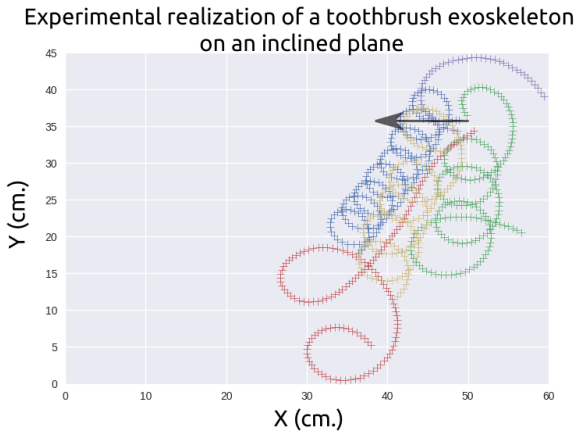


Figure 71 – Trajectories of a toothbrush exoskeleton on a weakly inclined plane (1°). The strength of the bias relative to the low gravity pull turns the straight trajectories observed for aligners and fronters into complex cycloids.

5.3.2 Toothbrush exoskeleton oscillating phenomena

When the toothbrush Kilobot is pushing against a static wall, both its velocity in the direction parallel to the wall and the angle of its polarity with the wall start to oscillate (see Figure 72.Left) with a growing amplitude. The maximum amplitude of these oscillations increases with an increase of initial speed. With a small speed, the Kilobot may stay against the wall for an undefined period

of time. The angle θ to the normal of the wall is plotted on Figure 72.Right for one realization of the phenomenon, with an initial speed $v_0 \approx 7cm.s^{-1}$.

With a large speed, the amplitude increases until the robot turns more than 90° relative the normal at the wall, and the robot leaves the wall. Multiple realization of this phenomenon for the same exoskeleton are shown on Figure 73, with θ the angle to the normal of the wall. Because each exoskeleton has its imperfection, the resulting robot tend to have a preferred direction in which it naturally turns when the motor signals are equal. This is the bias of the robot, showing its effect in this configuration as the robot always leaves the wall on the same side.

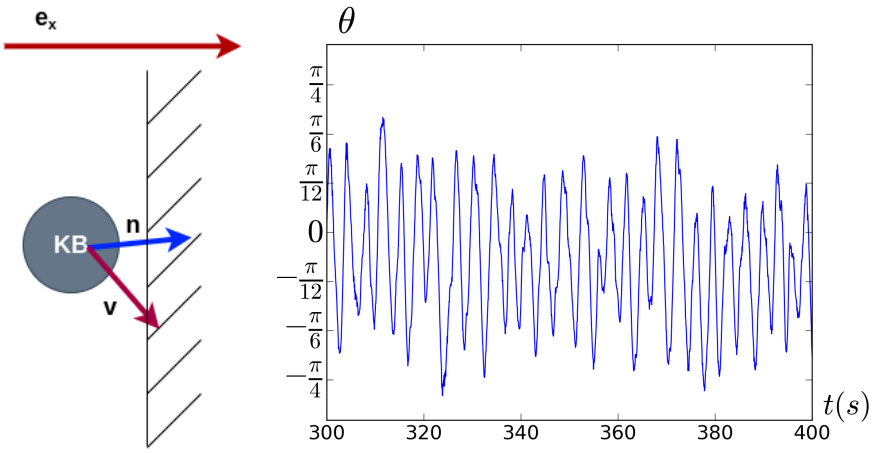


Figure 72 – **Left** Sketch of the straight wall experiment, in which decoupling of the velocity and orientation leads to oscillating behaviors. **Right**. Example of steady oscillation of the toothbrush Kilobot against a wall. The angle to the normal of the wall stays in $[-\pi/2, \pi/2]$ for the duration of the measurement. In this case, the initial speed is measure to be $\approx 7cm.s^{-1}$, equivalent to slightly more than a diameter per second.

In addition to the oscillations against a wall, the toothbrush exoskeleton often shows a transition from the normal biased motion in which the robot describes a wide circle to a state of pinned rotation, in which the robot gets stuck spinning on itself. This usually happens after a collision, against a wall or against another robot, or when the robot is constrained into a small space about the size of its diameter. An example trajectory of this phenomenon is shown on Figure 74.

These unprecedented effects illustrate one more time the role of morphology in vibrating robots dynamics. With the oscillations of increasing amplitude, a single robot is able to escape walls and not get stuck by only using its force-reorientation response, necessitating no software computation whatsoever. In order to understand the reasons for these oscillations, we have to turn back to the model of self-alignment.

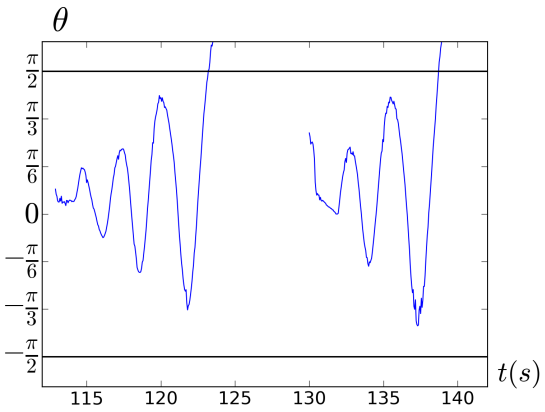


Figure 73 – Example of increasing oscillation of the toothbrush Kilobot against a wall. The angle to the normal of the wall quickly exceed $\pi/2$, and the robot leaves the wall. This happens in a preferred direction because of the bias of the robot, naturally turning left. In this case, the initial speed is measure to be $\approx 20cm.s^{-1}$, equivalent to 4 diameter per second.

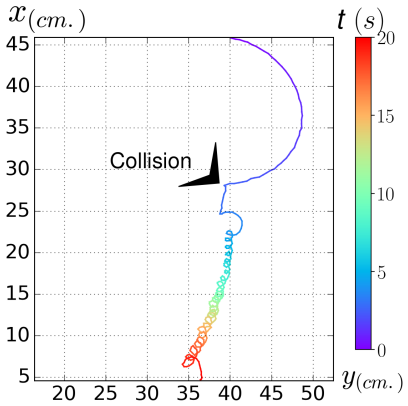


Figure 74 – Transition for a toothbrush Kilobot from a state of wide circular motion to a state of pinned rotation, with a small drift toward the bottom of the arena. This transition happens right after a collision with another robot, and is transient as the robot ends up going back to its regular state after colliding the wall, much later.

5.3.3 Model

In the model of self-alignment first described in Chapter 2 and used in the simulations, the dynamics on $\dot{\mathbf{n}}$ is supposed over-damped. In order to take into account inertial effects, we began to expand the model by considering the dynamics on $\dot{\mathbf{n}}$ no longer over-damped, as well as modeling the bias as a constant torque b. The effect of the bias is observed experimentally on the inclined plane, against walls and on horizontal trajectories since a moving robot always move along a circle. Although small compared to the self-aligning torque, the bias cannot be neglected to understand the individual and

collective dynamics of our swarms. Writing $\mathbf{n} = \begin{pmatrix}\cos\theta \\ \sin\theta\end{pmatrix}$, in an arbitrary reference frame $(\mathbf{e}_x, \mathbf{e}_y)$, we obtain the following equations for the dynamics :

$$\begin{cases} m\dot{v_x} = F_0\cos\theta - \gamma v_x + \Sigma\tilde{\mathbf{f}}_x \\ m\dot{v_y} = F_0\sin\theta - \gamma v_y + \Sigma\tilde{\mathbf{f}}_y \\ \dot{\theta} = \omega \\ I\dot{\omega} = \zeta(\cos\theta v_y - \sin\theta v_x) - \Omega\omega + \tilde{b} \end{cases} \tag{4}$$

where I is the angular moment of inertia and $\tilde{b}$ is the angular bias. We perform non-dimensionalization of the equations with the characteristic speed $v_0 = F_0/\gamma$, mass m and length d_0, yielding

$$\begin{cases} \tau_v\dot{v_x} = \cos\theta - v_x + \Sigma\mathbf{f}_x \\ \tau_v\dot{v_y} = \sin\theta - v_y + \Sigma\mathbf{f}_y \\ \dot{\theta} = \omega \\ J\dot{\omega} = \epsilon(\cos\theta v_y - \sin\theta v_x) - \tau_n\omega + b \end{cases} \tag{5}$$

with $\tau_v = \frac{mv_0}{\gamma d}$, $\Sigma\mathbf{f} = \frac{\Sigma\tilde{\mathbf{f}}}{\gamma v_0}$, $J = \frac{Iv_0}{|\zeta|d^2}$, $\tau_n = \frac{\Omega}{|\zeta|d}$ and $b = \frac{\tilde{b}}{|\zeta|v_0}$.

To describe the individual robot, we now have two additional parameters J and b. As we will see further, J defines the strength of angular inertial effects and is in great part responsible for the oscillations against walls. As for the bias, it complexify the dynamics of the single robot, but allows us to explain and understand behaviors observed on the inclined plane for all morphologies, and the case of self-trapping for the toothbrush exoskeletons. Note that according to a recent paper by Raoufi et al. 2023, the bias of bare Kilobots follows a Gaussian distribution of mean zero [**?**].

We begin by analyzing the case of the free particle, where we consider a robot moving according to these equations and subject to zero external forces.

5.3.3.1 The free particle : $\Sigma\mathbf{f} = \mathbf{0}$

In the absence of external force the isotropy of space suggest that only the difference of orientation between $\mathbf{n}$ and $\mathbf{v}$ matters. We thus describe the velocity vector using its magnitude and angle $\mathbf{v} = v\begin{pmatrix}\cos\phi \\ \sin\phi\end{pmatrix}$ and introduce $\alpha = \theta - \phi$ the difference between orientation angle and velocity angle. We obtain the free particle equations :

$$\begin{cases} \tau_v\dot{v} = \cos\alpha - v \\ \tau_v\dot{\phi} = \frac{1}{v}\sin\alpha \\ \dot{\alpha} = \omega - \frac{1}{\tau_v v}\sin\alpha \\ J\dot{\omega} = -\epsilon v\sin\alpha - \tau_n\omega + b \end{cases} \tag{6}$$

The first two equations respectively describe the translational and orientational motion of a frame tangent to the trajectory of the particles. In agreement with the isotropy of space, the variable ϕ

does not enter in the equations, except for the second one describing its dynamics, once v and α are known. In the following paragraph, this line will therefore be omitted.

In the following we shall look for simple "steady" solutions, namely straight line or circular motion, that is solutions for which α, v, and ω are constant. In order to do this, we solve the equations either analytically or numerically setting $\dot{\alpha} = \dot{v} = \dot{\omega} = 0$, yielding the set of final values α^*, v^* and ω^* describing the movement of the particle. These are the fixed points of the equations, which can either be *attractive* or *repulsive*, meaning that coordinates (α, v, ω) in a close vicinity will either converge exponentially to the fixed points or get away from them. The stability of a given fixed point can also be calculated, analytically or numerically, by linearizing the equations around the point. This gives us strong insights on the dynamics of a single particle. The calculation of the linear stability matrix and corresponding polynomial is left in Appendix C.

5.3.3.2 Case $b = 0$, no bias

This subcase does not need to be separated from the general case, but it provides a nice illustration of the difference between the aligners and the fronters in a simplified setting.

$$\begin{cases} \tau_v \dot{v} = \cos\alpha - v \\ \dot{\alpha} = \omega - \frac{1}{\tau_v v}\sin\alpha \\ J\dot{\omega} = -\epsilon v \sin\alpha - \tau_n \omega \end{cases} \tag{7}$$

Considering the fixed points $\dot{\alpha} = \dot{v} = \dot{\omega} = 0$, the equations for the fixed points strongly simplify into :

$$\begin{cases} v^* = \cos\alpha^* \\ \alpha^* = -\frac{1}{\tau_n}\epsilon v \sin\alpha^* - \frac{1}{\tau_v v}\sin\alpha^* \\ \omega^* = -\frac{1}{\tau_n}\epsilon v \sin\alpha^* \end{cases} \tag{8}$$

With the following solutions for v and α:

$$\begin{cases} v^* = \cos\alpha^* \quad \text{with} \quad v^* > 0 \\ \sin\alpha^* = 0 \quad \text{or} \quad \cos^2\alpha^* = -\epsilon\frac{\tau_n}{\tau_v} \end{cases} \tag{9}$$

5.3.3.3 Aligner, $\epsilon = 1$

The only fixed point is $\alpha^* = 0, v^* = 1$; hence $\dot{\phi} = 0$. The velocity is aligned with the polarity and the particle performs a straight motion. This straight trajectory is always linearly stable. This is in accordance with previous results, in papers describing the dynamics of vibrated polar disks.

5.3.3.4 Fronter, ϵ = -1

Considering the case of the fronter, in addition to the straight trajectory two additional solutions emerge when $\tau_v/\tau_n > 1$, : $\alpha^* = \pm\arccos\left(\sqrt{\frac{\tau_n}{\tau_v}}\right)$, $v^* = \sqrt{\frac{\tau_n}{\tau_v}}$, in a pitchfork like bifurcation.

The bifurcation diagram is given on Figure 75. The stable straight trajectory when $\tau_v < \tau_n$ can be understood by the velocity aligning to the orientation faster than the orientation tries to escape the velocity. This renders the fixed point $\alpha^* = 0$ stable. When $\tau_v/\tau_n > 1$, the straight trajectory becomes unstable and the two additional solutions have a non-zero angular velocity $\dot{\phi} = \frac{1}{\sqrt{\tau_n \tau_v}} \sin \alpha^*$. The particle orbits on a circle of radius $R^* = v^*/\dot{\phi} = \frac{1}{\tau_v \sin \alpha^*}$.The linear stability of the rotating branches may vary at high τ_v. Since this stability is driven by the three parameters τ_v, τ_n and J in addition to the bias b, this discussion is reported further in paragraph 5.3.3.8.

These additional solution already show a stark difference between fronters and aligners. However, because in real experiments our robots have a non-negligible bias, both morphologies fronter and aligner end up performing a circular motion. We must turn to the analysis of the equations in the presence of a bias $b \neq 0$.

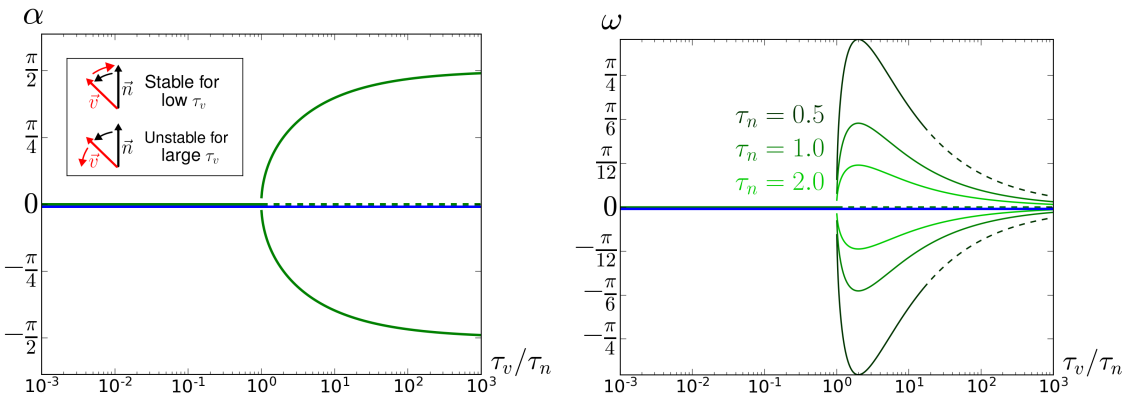


Figure 75 – **Left** Bifurcation diagram in α^* for a fronter without bias (green) and aligner without bias (blue). At $\tau_v > 1$, the solution $\alpha^* = 0$ destabilizes for the fronter and two branches emerge, corresponding to wide rotations. **Right** Bifurcation diagrams in ω^* for the fronter without bias (green) and aligner without bias (blue). The value of ω^* for the fronter scales with $1/\tau_n$, and the stability of the rotating branches at high τ_v may vary.

In Figure 75 as well as in every subsequent plot, each time the parameter τ_v/τ_n is varying we fix $\tau_n = 1$ and vary only τ_v. Performing the opposite operation, fixing $\tau_v = 1$ and varying τ_n leads to the same diagrams except for two considerations : ω^* scales as $1/\tau_n$ and both τ_v and τ_n (along with J) participate in the stabilization of rotations opposite to the bias.

5.3.3.5 General case

Looking for solutions with $\dot{v} = \dot{\alpha} = \dot{\omega} = 0$, we end up with the following equations:

$$\begin{cases} v^* = \cos\alpha^* \\ \omega^* = \frac{1}{\tau_v}\tan\alpha^* \\ \frac{\tau_n}{\tau_v}\tan\alpha^* = -\epsilon\cos\alpha^*\sin\alpha^* + b \end{cases} \tag{10}$$

Note that the constraint $v^* = ||\mathbf{v}^*|| \geq 0$ restrict the angle α to $[-\frac{\pi}{2}, \frac{\pi}{2}]$. With the substitution $t = tan(\alpha^*)$ we find solutions as the roots of a polynomial :

$$\frac{\tau_n}{\tau_v}t^3 - bt^2 + t(\epsilon + \frac{\tau_n}{\tau_v}) - b = 0 \tag{11}$$

By setting $b = 0$ in Equation 11, we recover the results of the previous paragraph. This polynomial gives us an analytical solution for v^*, α^* and ω^*, although their expression in the different parameters become utterly complicated. This formulation provides however an information on the maximum value for the bias that may produce non-pinned rotating solutions. By letting $\frac{\tau_n}{\tau_v} \to 0$ we find that one solution *can* exist with $t \to \pm\infty \Rightarrow \alpha^* \to \pm\frac{\pi}{2}$, leading to $v^* \to 0$ and $\omega^* \to b/\tau_n$ which is a pinned rotating solution. The only other possibility to obtain a real root in t imposes $|b| \leq \frac{1}{2}$. Conversely, for $\frac{\tau_n}{\tau_v} \to +\infty$, $\alpha^* \to 0$ becomes the only solution with $v^* \to 1$ and $\omega^* \to b/\tau_n$.

5.3.3.6 Aligner, $\epsilon = 1$

In the simple case $b = 0$, we found $\alpha^* = 0$ corresponding to the straight line motion, is the only solution. When $b \neq 0$, we first find that the linear motion is replaced by a slowly rotating circular motion, $\omega^* = \frac{\tan\alpha^*}{\tau_v}$, the radius of which $R^* = \frac{1}{\tau_v \sin\alpha^*}$ decreases from infinity when the bias grows from zero. Second, we find two other branches of circular motion solutions, only one of which is linearly stable. It has much smaller radius of gyration and must faster angular velocity.

- For $b \lessapprox 0.5$, the two circular motion are *disconnected* and the fast one disappear in a saddle node bifurcation when τ_v/τ_n becomes smaller than some critical value, which decreases when b increases.
- For $b \gtrapprox 0.5$, the two circular solution connects in an imperfect bifurcation.

The corresponding bifurcation diagrams are shown in Figure 76.

Experimentally, toothbrush exoskeletons sometimes show a transition between a state of circular motion of wide radius to a "self-trapped" state, in which the robot stays in place and rotates on itself. This phenomenon illustrated in the supplementary movies could correspond to the switching of the lower alpha branch to the upper alpha branch, increasing angular velocity and decreasing v to almost zero.

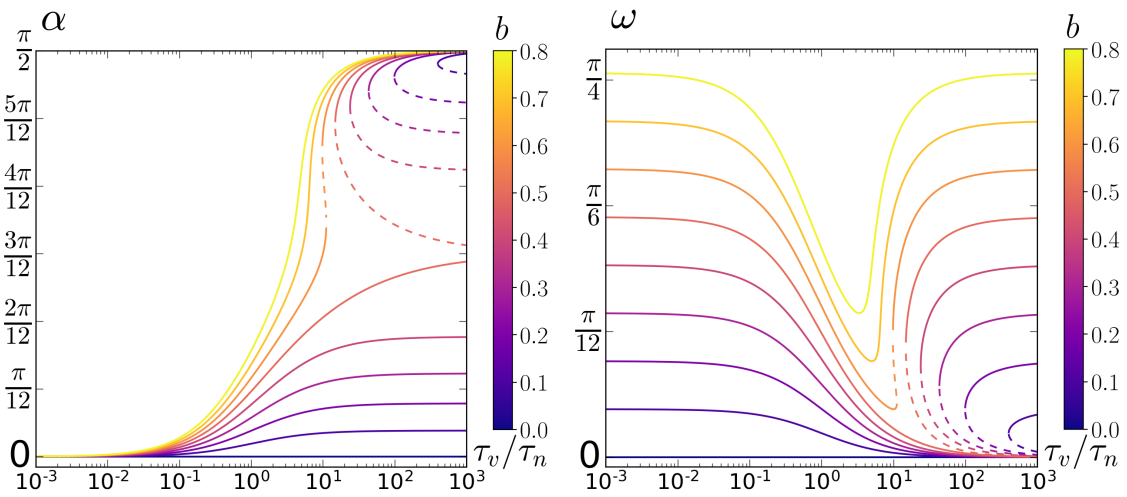


Figure 76 – Bifurcation diagrams in the general model for the aligner ($\epsilon = +1$) moving freely, for different values of bias.

5.3.3.7 Fronter, ϵ = -1

For the fronters, the simple case $b = 0$ showed a straight line trajectory $\alpha^* = 0$ along with a super-critical bifurcation into a circular trajectory $\alpha^* = \pm \arccos\left(\sqrt{\frac{\tau_n}{\tau_v}}\right)$, when $\tau_v > \tau_n$. In the presence of a bias, this bifurcation becomes imperfect. One of the bifurcated branch connects continuously to the straight trajectory solution. The other one connects in a saddle node bifurcation to the linearly unstable straight trajectory : the exact analogue of the Ising bifurcation in the presence of an external field. Note that ω^* is maximal close to the bifurcation. The bifurcation diagrams that we obtain are shown in Figure 77.

5.3.3.8 Non-trivial effect of angular inertia

In the regimes of high τ_v and $|b| < 0.5$, fronters are able to turn in both direction with the existence of branches of solutions with $\alpha \neq 0$. The stability of these solutions is driven by the three parameters b, τ_n and J controlling respectively the bias, the angular damping and moment of inertia. The effect of b is intuitive, with the gradual destabilization of trajectories which oppose the bias and disappearance of the solutions at $b \geq 0.5$. However, the two other parameters change completely the domain of stability of these solutions.

We show on Figure 78 the existence and stability domains of these solutions, for $\tau_n = 1$ constant and different values of J. At very low J, the bias impose a maximum value of τ_v for which a counter-rotating solution is stable, with this maximum value increasing exponentially as b gets close to zero, up to infinity for $b = 0$. For higher values of J, the shape of the stability frontier changes and intersect the line $b = 0$, meaning that the rotating solutions destabilize even at $b = 0$. The effect of

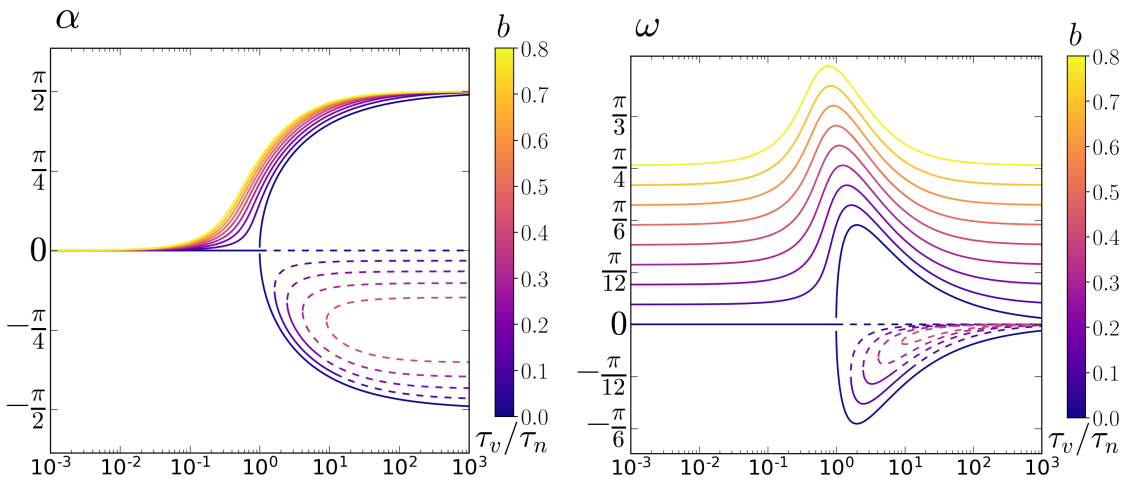


Figure 77 – Bifurcation diagrams in the general model for the fronter ($\epsilon = -1$) moving freely, for different values of bias.

J reverses for $J \gtrapprox 2$, with higher increase augmenting the size of the stable domain, rendering all solutions stable again when $j \to +\infty$. The same behavior occurs for an increase of $1/\tau_n$ at a fixed value of J, highlighting a similar role played by the two quantities.

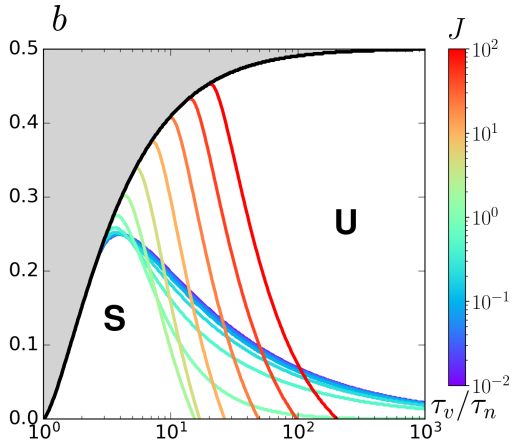


Figure 78 – Evolution of the stability domains in τ_v/τ_n and b of the rotating branch opposite to the bias for an inertial fronter. The increase in J first destabilize the branch but the effect reverses at $J \approx 2$ and a further increase in J stabilizes the branch again. The existence domain is also represented as the branch opposite to the bias disappear when the bias becomes too important.

5.3.3.9 Inclined Plane : $\Sigma \mathbf{f} \neq 0$

We now consider the case corresponding to the inclined plane experiment, where the particle experiences a constant external body force, say in the direction $\mathbf{e}_x$. The experiment consists of a rigid plane inclined a few degrees along the $\mathbf{e}_x$ axis. For different inclinations, we obtain an effective gravity uniformly affecting the space in which the Kilobot moves. Let α_h denote the angle to the horizontal, and $\tilde{g} = 9.81N$ the standard value for earth gravity. The dimensionless force applied is given by :

$$\mathbf{f}_{plane} = \frac{m\tilde{g}\sin(\alpha_h)}{\gamma v_0}\mathbf{e}_x = g\mathbf{e}_x$$

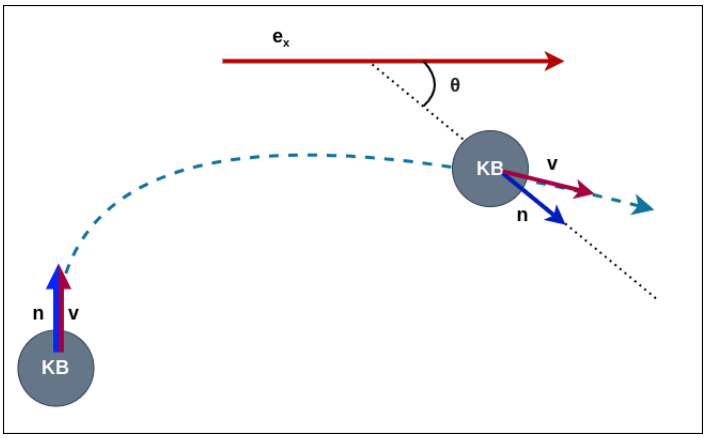


Figure 79 – Sketch of the inclined plane experiment, with a Kilobot on a plane inclined along $\mathbf{e}_x$.

Keeping θ as the angle of the orientation vector $\mathbf{n}$, we come back to the Cartesian formulation described in Equation 5, adding the dimensionless gravity :

$$\begin{cases} \tau_v \dot{v_x} = \cos\theta - v_x + g \\ \tau_v \dot{v_y} = \sin\theta - v_y \\ \dot{\theta} = \omega \\ J\dot{\omega} = \epsilon(\cos\theta v_y - \sin\theta v_x) - \tau_n\omega + b \end{cases} \tag{12}$$

As observed in the experiments, the circular solutions due to the bias turn into complex cycloïds on the inclined plane, for which there is no chance to capture the analytical expression. We therefore concentrate on the straight trajectories for which $\dot{\theta} = \dot{v_x} = \dot{v_y} = 0$ and find the following :

$$\begin{cases} v_x^* = g + \sqrt{1 - \frac{b^2}{g^2}} \\ v_y^* = \epsilon\frac{b}{g} \\ \sin\theta^* = \epsilon\frac{b}{g} \end{cases} \tag{13}$$

Hence, the straight trajectory exists only when the bias is not too strong, namely $b < g$. The final trajectory makes an angle with the applied force, that scales as $\arcsin\frac{b}{g}$. The linear stability analysis of this fixed point remains to be completed, the calculation of the linear stability matrix and corresponding polynomial are left in Appendix C.

5.3.3.10 Wall interactions

Simulating the previous equations against a wall modeled as a stiff interaction potential, we find that inertia on $\dot{\theta}$ and v is not sufficient to generate any form of oscillating behavior. The description of the interaction with the wall must include a component parallel to the wall. The most general kinematics of a rolling disk in contact with a plane is that of rolling with sliding, which we now implement in the model.

We consider an infinitely long wall along the $\mathbf{e}_y$ direction, located at $x = 0$, and described by a WCA potential V_{wall}, of width $\sigma = \frac{d}{2}2^{-\frac{1}{6}}$ and depth 1 :

$$V_{wall} = \begin{cases} (\frac{d}{2x})^{12} - (\frac{d}{2x})^6 + 1 & x > -\frac{d}{2} \\ 0 & x \leq -\frac{d}{2} \end{cases} \tag{14}$$

The wall is then modeled as a repulsion force, a damping force and a torque as follows which become non zero when $x > -d/2$.

$$\begin{aligned} \mathbf{f}_\perp &= -\frac{\partial V_w}{\partial x}\mathbf{e}_x \\ \mathbf{f}_\parallel &= -\mu u \mathbf{e}_y \\ \Gamma_w &= -\mu u \frac{d}{2} \end{aligned}$$

where $u = v_y + \frac{d}{2}\dot{\theta}$ is the sliding speed along the wall, μ is the wall friction coefficient. Simulating these equations allow to recover the oscillations, for specific values of the parameters. An example of a simulated oscillating trajectory is shown on Figure 80, with a wall positioned at $x = -\frac{1}{2}$.

In order to analyze the model theoretically, we begin by assuming permanent contact ($x = -d/2; v_x = 0$) and analyze the reduced set of equations for the position along $\mathbf{e}_y$ and the orientation of the toothbrush Kilobot :

$$\begin{cases} \dot{\theta} = \omega \\ I\dot{\omega} = \zeta \cos\theta v_y - \Omega\omega + \tilde{b} - \mu u \frac{d}{2} \\ m\dot{v_y} = F_0 n_y - \gamma v_y - \mu u \end{cases} \tag{15}$$

Once again, considering v_0 the final speed of the isolated particle, we re-scale mass, length and time with m, d and d/v_0 and replacing u with $v_y + \frac{1}{2}\dot{\theta}$, the final equations modeling a permanent contact with the wall read :

$$\begin{cases} \dot{\theta} = \omega \\ J\dot{\omega} = (\cos\theta - \tau_r)v_y - (\tau_n + \frac{1}{2}\tau_r)\omega + b \\ \tau_v \dot{v_y} = \sin\theta - (1+\lambda)v_y - \frac{1}{2}\lambda\omega \end{cases} \tag{16}$$

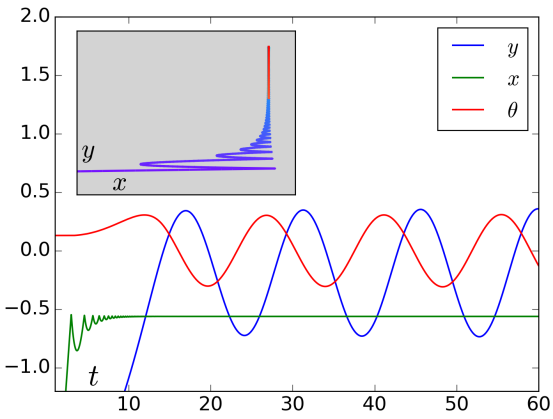


Figure 80 – Example of a simulated oscillating trajectory against a wall. The combination of angular inertia and rolling plus sliding against the wall allow the apparition of either a limit cycle or a turn around with the angle to the normal of the wall becoming greater than $\pi/2$.

with $\tau_v = \frac{mv_0}{\gamma d}$, $\lambda = \frac{\mu}{\gamma}$, $J = \frac{Iv_0}{|\zeta|d^2}$, $\tau_n = \frac{\Omega}{|\zeta|d}$, $\tau_r = \frac{\mu d}{2\zeta}$ and $b = \frac{\tilde{b}}{|\zeta|v_0}$.

In the following, we assume $b = 0$ as a mean of simplification. It is clear that the bias will make the oscillation asymmetric, but at least when the natural motion of the toothbrush Kilobot is the straight trajectory, we don't expect the bias to strongly alter the analysis.

5.3.3.11 In the absence of angular inertia J = 0

When, $J = 0$, the simplified 1d-model equations recast in two first order ODE for θ and v_y :

$$\begin{cases} \tau_v \dot{v_y} = \sin\theta - (1+\lambda)v_y - \frac{1}{2}\lambda\dot{\theta} \\ (\tau_n + \frac{1}{2}\tau_r)\dot{\theta} = (\cos\theta - \tau_r)v_y \end{cases} \tag{17}$$

These equations have two fixed points : a static fixed point $(v_y^0, \theta^0) = (0, 0)$, which exists for all values of the parameters, and a sliding solution $(v_y^*, \theta^*) = (\frac{\sqrt{1-\tau_r^2}}{1+\lambda}, \arccos\tau_r)$ when $|\tau_r| \leq 1$.
We could also not find any oscillating solutions, when exploring the equations numerically. It is likely that angular inertia is necessary to generate any form of oscillation, as seen in Figure 81

5.3.3.12 Including angular inertia $J \neq 0$

We look at the stability around the fixed point $v_y^* = 0, \theta^* = 0$ (with the stability matrix shown in Appendix C) and numerically compute the eigenvalues. We represent the evolution of the complex eigenvalues on Figure 82 for different values of τ_r, setting the angular inertia J constant at different scales.

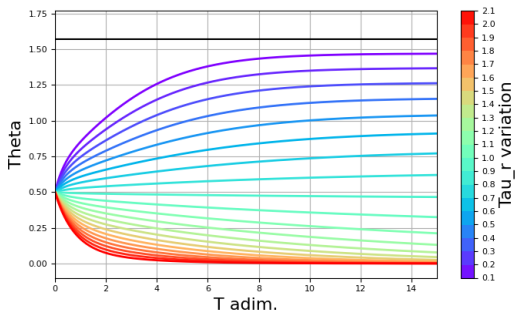


Figure 81 – Evolution of θ over time for different values of τ_r, in the absence of angular inertia. $\tau_v = \tau_n = \lambda = 1$

We find that for small inertia, τ_r controls the transition from the sliding solution to the pinned solution. The relaxation toward the pinned solution is oscillating, with a relaxation time and amplitude that grows with the two inertial terms J and τ_v. This happens until the angular inertia stabilizes the oscillations into a limit cycle, in a Hopf-like bifurcation. The two inertial terms scale with v_0, and the amplitude of the oscillations may increases up to $\pi/2$, leading to the robot leaving the wall in the 2D case. The last parameters λ and τ_n are damping terms, which seem not to change the phenomenology as long as inertia scales with the damping. Otherwise, they push the system toward a fixed point.

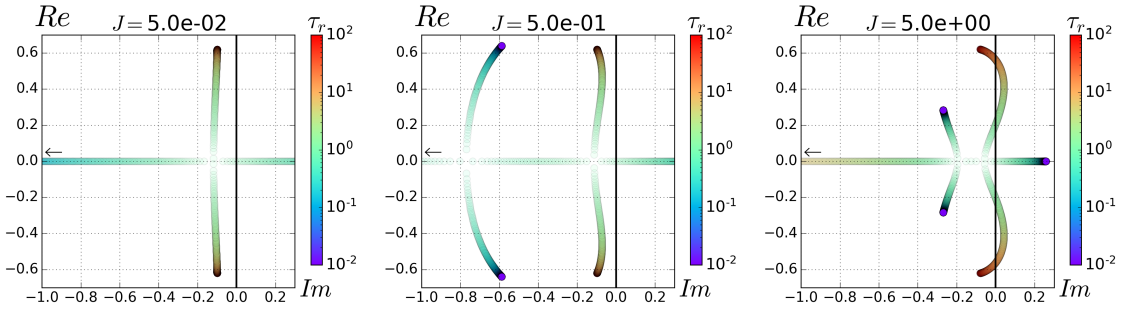


Figure 82 – Path followed by the eigenvalues in the complex plane of the linear stability matrix, as τ_r is varied, obtained around the fixed point $v_y^* = 0, \theta^* = 0$, for different values of J. A negative real part for all the eigenvalues renders the fixed point linearly stable, and there is no chance to observe oscillations. For $J = 5$ and some values of τ_r, a pair of eigenvalues cross the line $Re = 0$, hinting at parameters for which a robot can oscillate against a wall.

6 Learning as a system of SDEs

This Chapter aims at implementing a model of distributed online learning inspired from the HIT algorithm in continuous time, formulated using stochastic differential equations (SDEs). The basis principles of this model is to consider oriented particles described by their positions and orientation which have internal degrees of freedom coupled to the dynamics. These degrees of freedom change in time, either spontaneously (mutation) or after a physical collision with another particle (transfer).

6.1 Learning as a system of SDEs

We consider a system of N robots of diameter d and mass m, that are described by self aligning or anti-aligning particles. We propose a 2D formulation of the model, with each robot i having three physical degrees of freedom : two translational ones $\mathbf{r}_i = (x_i, y_i)$ and one orientational one $\hat{\mathbf{n}}_i = (\cos\theta_i, \sin\theta_i)$. In addition, each robot has an amount K of internal degrees of freedom encoded as numbers $w_i^k \in [0, 1]$. Finally each robot has a reward sensor, and computes a score q_i on the basis of the reward given by the environment.

6.1.1 Locomotion

The equations describing the locomotion of the robots are the same as the equations previously exposed in Chapter 5.1.4. We consider N robots localized in 2D space at the positions $\mathbf{r}_i$, moving at a velocity $\mathbf{v}_i$ with an orientation vector $\hat{\mathbf{n}}_i$ of length one. Each robot is propelled by an active force $\mathbf{F}^0 = F^0\hat{\mathbf{n}}$, with $F^0 \in [0, F^0_{max}]$, and it experiences a damping $-\gamma\mathbf{v}$. The final speed of an isolated robot is thus $v^0 = F^0/\gamma$. The robots interact through a pairwise WCA potential and can be submitted to an external force $\mathbf{F}_{ext}$. The dynamics of the orientation self aligns on the velocity, as described in previous chapters. This physical dynamics is governed by the following dimensionless equations:

$$\begin{cases} \frac{d\mathbf{r}_i}{dt} &= \mathbf{v}_i \\ \tau_v \frac{d\mathbf{v}_i}{dt} &= F^a_i\hat{\mathbf{n}}_i - \mathbf{v}_i + \hat{\mathbf{F}}_{ext} \\ \tau_n \frac{d\hat{\mathbf{n}}_i}{dt} &= \epsilon\left(\hat{\mathbf{n}}_i \times \mathbf{v}_i\right) \times \hat{\mathbf{n}}_i \end{cases} \tag{18}$$

where the units of mass, length and time have been chosen as m, d and $\frac{d}{v^0_{max}} = \frac{d\gamma}{F^0_{max}}$. For the sake of simplicity, we choose to use the over-damped dynamics for the orientation. With these units we have $\tau_v = \frac{mF^0_{max}}{\gamma^2 d}$; $F^a_i = F^0_i/F^0_{max}$, $\tau_n = \frac{l_a}{d}$, where l_a is the alignment length ($l_a = 1/\zeta$, the aligning coupling strength) and the amplitudes of the interaction and external forces are expressed in units of F^0_{max}. In the same way as before, the parameter $\epsilon \in \{-1, 0, 1\}$ sets the type of alignment, enabling morphological effects. The case $\epsilon = 0$ and $\tau_v \to 0$ defines an Active Brownian Particle with zero translational noise. On top of the deterministic trajectories given by Equation 18, angular noise is added using the following procedure: when discretizing the dynamics with a time step δt, we rotate $\hat{\mathbf{n}}_i(t)$ by a random angle $\eta_i(t)$ distributed normally with zero mean and variance $2D_i\delta t$. For each robot, $D_i \in [0, D^0_{max}]$ fixes the level of the angular noise. Noises of different robots are statistically independent.

Here in this model, each robot can only pilot its active force F_i as well as its level of angular noise D_i. The control of each robot is done through a function, called the controller, further developed in Chapter 6.1.4. We deliberately omit the capacity of the robot to turn on itself and perform controlled rotations. This choice not only allows to greatly simplify the model, but also crystallize our philosophy of distributed learning, for which each robot is made as simple as possible by design, and the lack of control at the individual level is balanced by collective effects.

6.1.2 Learning

In order to identify the best parameters, at each time-step each robot builds up a score q which time derivative has to be written as a function of particles positions, orientation and internal degrees of freedom. In the case of the phototaxis experiments, a score is given when the robots spends time in the lit region of the simulation box. We model this by a scalar field $\mathbf{I}$, towards which the score converges exponentially.

The communication dynamics is encoded into a continuous time process. When two robots are close enough, $||\mathbf{r}_i - \mathbf{r}_j|| \leq r_c$, the parameters of the robot that has the lowest reward converge exponentially to the parameters of the robot with the highest reward. Each parameter w^k of the robot i also experience a delta correlated Gaussian noise η_i^k, that mimics the effect of mutations. In order for the parameters to be constrained to $[0, 1]$, they are enclosed in a sharp potential (this procedure do not induce accumulation to the boundaries). This dynamics is governed by the following dimensionless equations:

$$\left\{ \begin{array}{rcl} \frac{dw_i^k}{dt} & = & \sqrt{2\mu}\eta_i^k - \frac{\partial V}{\partial w_i^k}(w_i^k) \quad + \quad \alpha \sum_j \mathbf{\Gamma}_{i \leftarrow j}(w_i^k, w_j^k) \\ \frac{dq_i}{dt} & = & \alpha_q (I(\mathbf{r}_i) - q_i) \qquad\quad + \quad \alpha \sum_j \mathbf{\Gamma}_{i \leftarrow j}(q_i, q_j) \end{array} \right. \tag{19}$$

where α sets the communication rate, α_q sets the rate at which the score increases in the light and decays in the dark and μ controls the amplitude of the mutations. In practice, because the potential at the boundary has to be really sharp to approach 0 and 1, it is implemented as hard collisions, by setting $w_i^k = 1 - w_i^k$ if $w_i^k > 1$ and $w_i^k = -w_i^k$ if $w_i^k < 0$ at each time-step δt.

The exchange function is given by:

$$\Gamma_{i \leftarrow j}(x_i, x_j) \quad = \quad H(q_j - q_i)\, H(r_c - ||\mathbf{r}_j - \mathbf{r}_i||)\, (x_j - x_i) \tag{20}$$

where H is the Heaviside function. We set the zero of the Heaviside function to be equal to 1, as to let the robots communicate even when their score are equal. This allows us to study the case where no reward is given. In practice, since we set the light to be piece-wise constant this happens at the beginning of the simulation and is likely to happen after convergence of the scores, depending on α_q and the duration of the simulation.

6.1.3 Parameters of the model

The model has a total of eight parameters. Three parameters control the locomotion, four parameters control the communication and learning rate, and the last is the particle density. We resume the parameters below.

- τ_v : Translation inertia
- τ_n : Alignment strength
- ϵ : Alignment type
- r_c : Communication radius
- α_q : Reward integration rate
- α : Communication rate
- μ : Mutation intensity
- N/L^2 : Particle density

In order to minimize the number of free parameters, we set the value of a few parameters with heuristics. First, we set $\tau_v = 10^{-3}$, since inertia is not a relevant parameter in the study of neither the dynamics of learning, or the effect of morphology through the force-reorientation response. We also set the communication rate to be $\alpha = 10^2$, which makes the convergence much faster compared to the dynamics of the particles, mimicking real robot communication. Finally, we set the communication radius at the threshold of the repulsive potential $r_c = d = 2^{1/6}$, removing a length scale in the system.

Some communication events can occur that do not translate to real robot experiments. In the case of a frontal binary collision between robot A and B, the communication occur as expected and the parameters are fully transmitted from A to B. However, collision events can be shorter than the time required for the full transmission. This stops the convergence of the parameters halfway. Another effect can occur in the case of collisions involving multiple particles in simultaneous contact, with the convergence of the parameters to the mean of received parameters. If robot A has a reward less than robot B and C and comes to a collision between both particles at the same time, the parameters of A will converge neither to B or C but to the mean of B and C. Changing the communication radius balances the first effect with the second. These two effects unintentionally act as additional sources of novelty, different from the mutations.

6.1.4 Controller

The controller denotes the function which maps the sensory input and internal parameters to the behavior of the robot. In our model, the robot pilots the intensity of its active force F_i^a and its angular diffusion coefficient D_i. In the case of phototaxis, the only external input is the light intensity I. The controller hence takes the form of two functions, $F_i^a(I, w_i^0, ..., w_i^k)$ and $D_i(I, w_i^0, ..., w_i^k)$.

We implement in a first setting the controllers corresponding to the learning experiments described in Chapter 4 and Chapter 5. In the learning with threshold experiments, the robots always keep the same type of motion irrespective of the light received : run and tumble. The only change made to the behavior of the robot is its velocity, taking two values v_1 and v_0. This change in behavior is made through a single learnt parameter w^0. To model this situation, we set the angular

diffusion coefficient to be a constant, $D = 0.02$, and the controller is the function $F_i^a(I, w^0)$. The four types of controller of the Chapter 5, *Preset still*, *Preset Move*, *Learning still* and *Learning move* are represented on Figure 83. In all four cases, the controller is a step function, which step happens at w^0. For the Preset case, w^0 has a fixed value $w^0 = \frac{1}{2}$. The minimal value of F^a at $I > w^0$ is fixed at zero for the Still case and 0.2 for the Move case.

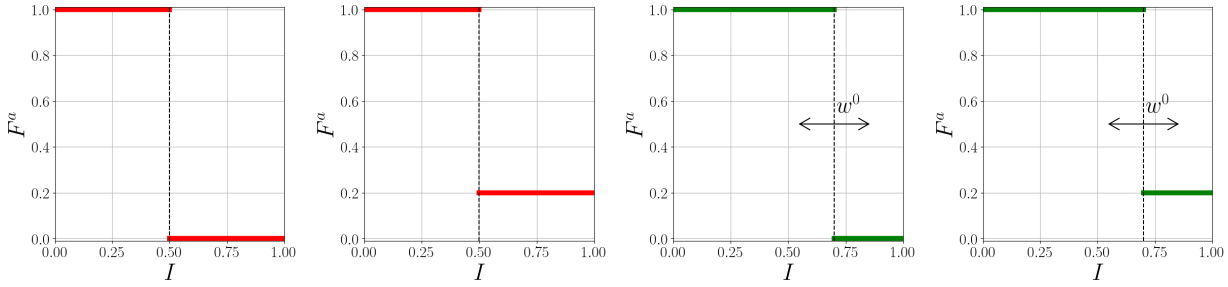


Figure 83 – Types of controllers used in the learning with threshold experiments written as functions $F^a(I, w^0)$, defining the controller in simulations. From left to right : *Preset still*, *Preset Move*, *Learning still* and *Learning move*, where *Preset* denotes the case $w^0 = \frac{1}{2}$ fixed, *Still* denotes $F^a(w > w^0) = 0.0$ and *Move* denotes $F^a(w > w^0) = 0.2$.

The controllers (and reward) we chose for our simulation are relatively simple. However, the model we present here can use more complex controllers such as neural networks, and is applicable to a wider range of tasks. The main constraints of the model concern the communication, as it cannot be controlled by the robot in any way, and the sensory inputs as they can only be related to external fields, robot positions, orientations and internal degrees of freedom.

6.1.5 Discussion

Initially the internal degrees of freedom w_i^k were thoughts as an N dimensional spin, aligning asymmetrically during collisions. Unfortunately, due to the periodic nature of spins, the mapping from the orientation to the active force between zero and one (closed and bounded interval) forces either non-injectivity or discontinuities. We resorted to constraining the parameters to $[0, 1]$, through a sharp repulsive potential at the boundaries, which allow diffusion while keeping the distribution of parameters uniform over time.

6.2 Implementation in LAMMPS

In order to easily manage simulations and benefit from the speed and flexibility of specialized software, we decided to integrate our model to LAMMPS [**?**]. As stated on the official website, LAMMPS is "a classical molecular dynamics code with a focus on materials modeling, which name is an acronym for Large-scale Atomic/Molecular Massively Parallel Simulator". Through

LAMMPS scripts, one can change the interaction potential and boundary conditions, add external fields with different geometries, mix particle types and sizes and parallelize computation over large amount of processors while benefiting from an efficient neighbor computation. The main challenge was to adapt our equations and concepts to the data structures of LAMMPS, which is entirely written in C++. The final implementation of our model is publicly available at https://github.com/JeremyF-141592/spp_lammps.

The final result of a simulation is a dump file (we choose the format .cfg) containing positions, velocities, orientations and internal degrees of freedom of each particle over time. The visualization of the results is made through another software, we choose to use Ovito Basic (https://www.ovito.org/), which is free and open-source.

6.3 Homogeneous light intensity

Before adding any source of reward, we study the behavior of the model in the presence of an homogeneous light intensity. We consider that initially, all scores are equal to zero. We also consider a single parameter w_i, and a controller perfectly decoupled from both light and the parameter, that is $F_i^a(I, w_i) = 1$. We perform 20 independent simulations in this setting.

In this setting, binary collisions provoke the convergence of both particles parameters to their mean value. Aside from collisions, the parameters diffuse in $[0, 1]$ with an intensity μ. The consensus of the parameters in the whole swarm becomes a balance between the collision frequency, related to velocity and density, and the intensity of this diffusion. We set the density in the simulation box to be half the initial density of real robot experiments, setting $\rho = 0.16$. This reflect the fact that the density of the real robot experiment is much lower than $N/\pi R_{arena}^2$, because of the accumulation of the robots to the walls. Finally, in order to limit finite size effects we simulate 1024 particles instead of the usual 64.

Considering the case with no mutations, $\mu = 0$, we expect the system to mix well enough such that, by successively averaging pairs $w_i, w_j \leftarrow (w_i + w_j)/2$, the whole system ends up to the average of the initial distribution of w. We expect to obtain $\mathbb{E}(w_\infty) = \frac{1}{2}$ and $Var(w_\infty) = 0$. On the contrary, if μ is high enough, the parameters becomes randomly distributed in a duration ϵ after the collision and exchange of parameters. Looking at the distribution of w for different values of μ on Figure 84, we observe that as μ increases, the distribution gradually goes from a sharp distribution around $\frac{1}{2}$ to a uniform distribution.

To characterize the consensus in the distribution of the weights across the population with a number bounded by 0 and 1, we define an order parameter ϕ based on entropy. First, we evaluate the final distribution p of parameters values with an histogram of Q bins. Since entropy is maximized for the uniform distribution, meaning minimal consensus, we define $\phi = 1 - \frac{S(p)}{\ln Q}$ where $S(p) = -\sum_i p_i \ln p_i$ is the entropy of p and $\ln Q$ is the entropy of the discrete uniform distribution with size Q. This measurement yields 1 if and only if the difference between the maximum value and the minimum value of the parameters is lower than $1/Q$. It yields 0 if the values are uniformly

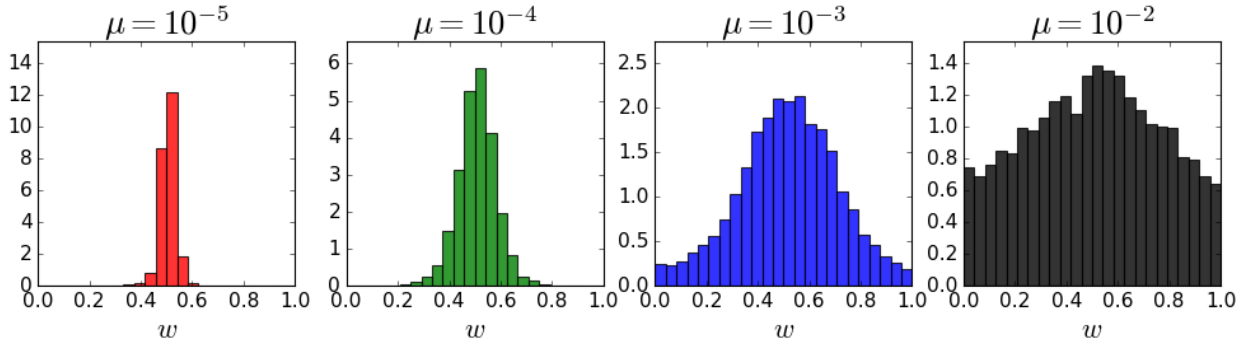


Figure 84 – Distributions of w at $t = 1000\tau$, for $N = 1024$ and $\rho = 0.16$, for 20 independent simulations of the homogeneous light intensity setting.

distributed in the Q bins. Note that zero is reachable exactly only if $N_{particles} \equiv 0 \bmod Q$. The results of this measurement as a function of μ, computed on 10 independent seeds, is given on Figure 85.

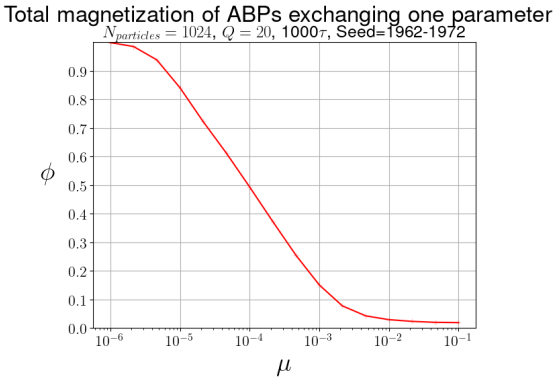


Figure 85 – Consensus the internal parameter w based on entropy minimization, for ABPs exchanging averaging their parameters on collision, as a function of μ. With an order parameter based on entropy minimization.

6.4 Application to phototaxis

As a validation of the model, we run simulations corresponding to the learning experiments with threshold presented in Chapter 4 and Chapter 5. The simulation are run over 1000τ and 50 different seeds, and we measure the same quantities as before, categorizing robots based on their velocity and position in or out of the light. This define 4 categories, *Moving lit*, *Moving unlit*, *Stopped lit* and *Stopped unlit*. We keep the same radius for the light disk as in the experiments, $r_{light} = 3.95d$. As in the previous simulations, we half the density in the simulation box compared to the initial density of real robot experiments, setting $\rho = 0.16$. Since the learning experiments were performed

with aligners exoskeleton, we set the particles to be slightly aligners with $\epsilon = +1$, $\tau_n = 2.0$.

The reward convergence rate is set to $\alpha_q = 0.1$, producing a 95% convergence to the reward in a duration of 30τ where τ is the natural unit of time d/v_0. To compare to the reward buffer duration in the experiments, we can roughly evaluate τ of the real system by taking the mean velocity of the run and tumble as the characteristic velocity $v_{RT} = 2.13$. Since the diameter of the exoskeleton is $d = 4.8cm$, this yields $\tau_{real} = 2.25s$. Hence, setting α_q to 0.1 reasonably models a reward buffer duration of about a minute. In addition, to fit the experimental conditions, we set a reward delta for the communication to $\delta_r = 0.1$ which acts in the exchange function, replacing $H(q_j - q_i)$ by $H(q_j - q_i - \delta_r)$. Doing so forces partial convergence of the score and parameters, but is key to reproduce the phenomenology of the experiments. The angular diffusion coefficient D is set constant to $D = 0.02$.

We measure the proportions of moving robots in and out of the light, and plot the results on Figure 86 along with the real experiment plots as a comparison.

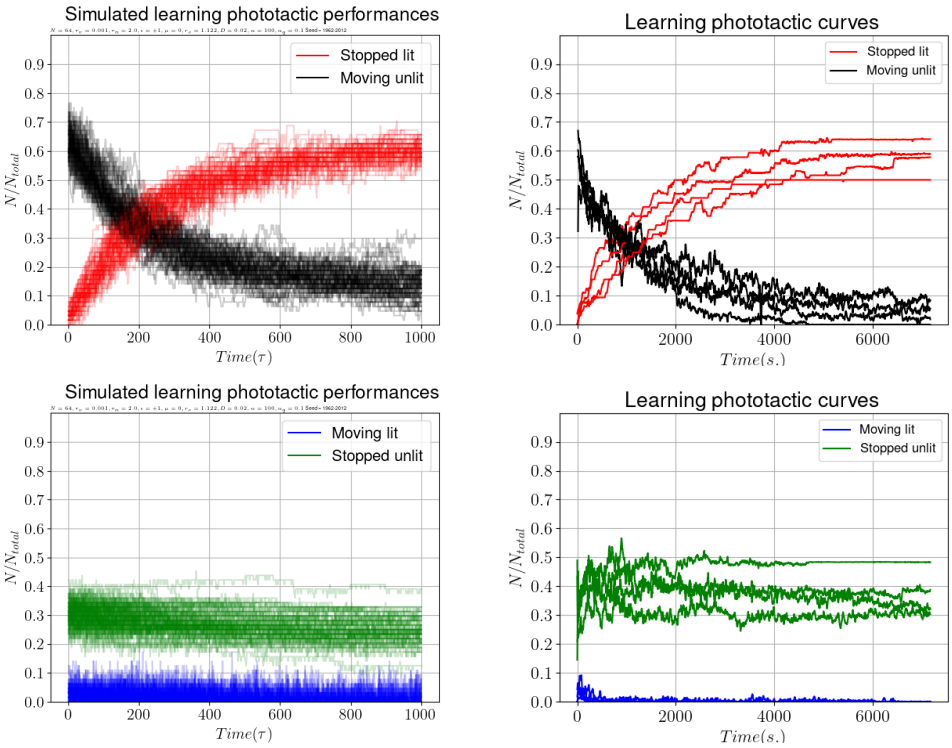


Figure 86 – Comparison between the simulated learning (left) and real robot experiments (right) in the case of the learning with threshold, also called run and tumble meta-policies.

6.5 Asymmetric communication induced phototaxis

Performing simulations with different types of controllers, we found out that the controller do not need to implement an explicit dependency on the light intensity in order for the particles to perform phototaxis. Indeed, the accumulation of reward and asymmetric communication from high score to low score is sufficient to aggregate particles in lit regions of space. We define controllers as functions of a single parameter w, starting with the most simple controller $F_0^a = w$. Two other controllers are considered, $F_1^a = |1 - 2w|$ and $F_2^a(w) = \frac{1}{2}(\cos(2\pi w) + 1)$. All three are represented on Figure 87 on top. In the simulations, we split the box in two regions of equal area, one lit and one unlit, and keep periodic boundary conditions. The intensity of the mutations is set high to $\mu = 10^{-3}$. We perform the simulations with 1024 particles and the same density as before $\rho = 0.16$. The performance over time for the different controllers along with a snapshot of the system at $t = 2000\tau$ for the last controller F_2^a are given in Figure 87 bottom.

With the three different controllers, the final performance is always greater than 50%. The core explanation for this lies in the asymmetry of the communications. Every particle that makes a transition dark $\rightarrow$ lit or lit $\rightarrow$ dark has to have a non-zero F^a, and the higher F^a is the greater the chance a particle have to make the transition. As a mean of simplification, we suppose that the particle leaving has $F^a > 0.5$ and in both regions, $<F^a> \approx 0.5$. In the case of dark $\rightarrow$ lit, the particle can only slow down, to align with the mean. However, in the case of lit $\rightarrow$ dark, the incoming particle has a greater reward and therefore not only do not change its own F^a but speeds up the particles it encounters. This creates the difference in densities observed in lit compared to dark. In addition, for the cosine controller we observe the spontaneous formation of clusters localized in the lit regions. This show phototaxis in the system despite no explicit dependency of the active force to the received light intensity. This phenomenon is rooted in three effects :

- The convergence of $<w>$ towards the mean of the initial distribution in regions of homogeneous reward. This happens naturally in the system, as describe in the previous section. With the controllers F_1^a and F_2^a, this pushes the active force toward zero.
- The diminution of the effect of mutations on F_2^a around $w = 0.5$. The derivative of the controller relative to w being zero at $w = 0.5$ and continuous, fluctuations of w in this region produce fluctuations on the active force which are greatly lower.
- The asymmetry in the communications. In the case of F_2^a, initially clusters form in both the lit and dark regions of space. However, these clusters are in equilibrium with a gazeous phase of particles, producing exchanges in the two regions. When the free particle i performs the transition dark $\rightarrow$ lit and collide with a cluster, w_i converges to a value around 0.5, the particle slows down and is likely to join the cluster. On the contrary, when the free particle i performs the transition lit $\rightarrow$ dark and collide with a cluster, it is the parameters $w_k, w_{k+1}...$ of the particles in the cluster which converge to w_i. Because the particle i *is* a free particle, its velocity is non-zero and the value of w_i must be very different from 0.5. This produces an explosion of the cluster in the dark, as the particles $k, k+1, ...$ quickly set their velocity to be non-zero.

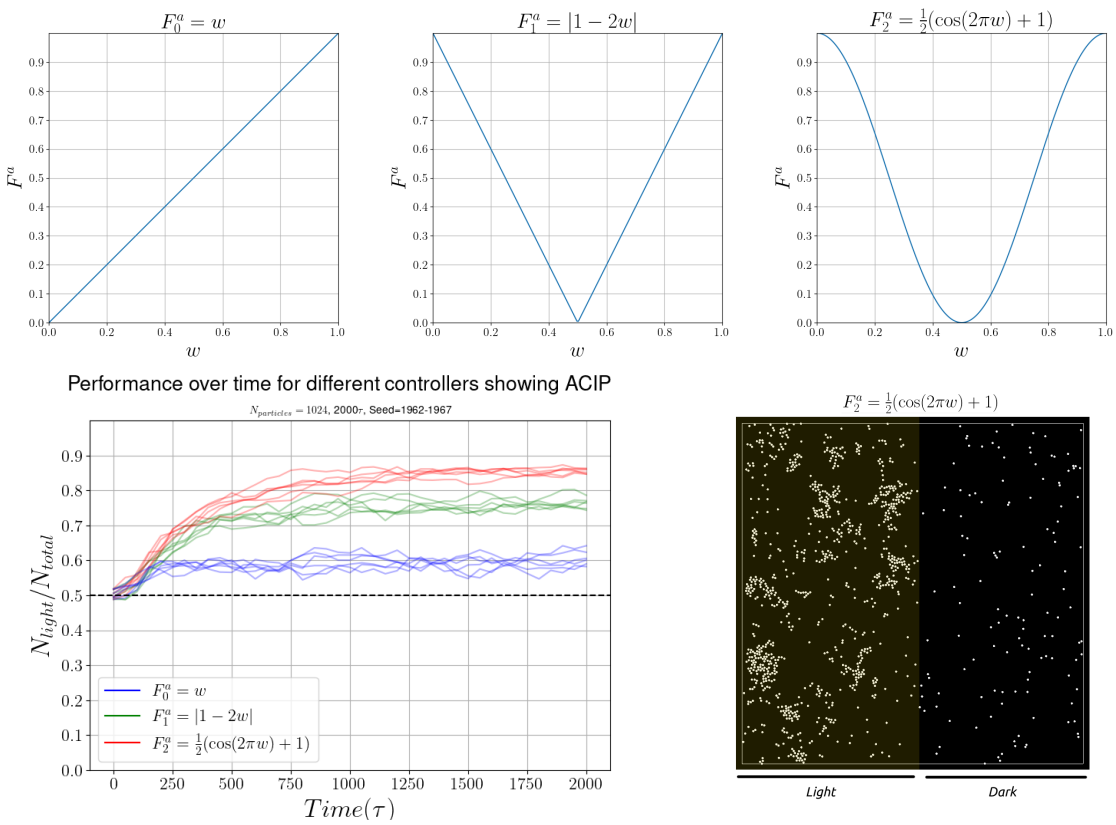


Figure 87 – Asymmetric communication induced phototaxis in a system of 1024 ABPs. Controllers are shown on top, performances on the bottom along with a snapshot at $t = 2000\tau$ for the controller $F_2^a(w) = \frac{1}{2}(\cos(2\pi w) + 1)$.

6.6 Perspectives

This model captures a form of learning close to the one of the phototaxis experiments we performed with Kilobots. One step toward understanding the dynamics of large swarms performing distributed learning is to write the associated Fokker-Planck equations for the model. Performing this transformation of the model would allow to study the limit $N \to +\infty$. In real experiments, even though we are always subject to finite-size effects, the theoretical statements made at very large scales could explain observed phenomena and help to design scalable distributed algorithms.

7 Conclusion

7.1 Summary

In this thesis, we have studied the dynamics of robotic swarms enhanced by their morphology and subject to online learning through a distributed evolutionary algorithm.

Chapter 4 describes how a swarm of N=64 Kilobots behave under simple conditions in a circular arena of 1.5 meter in diameter. The simple experiments serve as a baseline and show the bias in the motion of the Kilobots, the accumulation at the boundary and the risks in the depletion of the battery. Following the baselines, we set-up a phototaxis experiment in which robots have to aggregate in a disk of light at the center of the arena, by performing a run and tumble motion with a dependency of the speed to an internal parameter w. Setting this parameter to a constant good value show the exponential accumulation of robot in the center up to a plateau at 70% of the swarm. We deploy the HIT evolutionary algorithm on the robots, in order to let the swarm discover and propagate the correct value of w to perform phototaxis. This procedure defines three classes of behaviors, among which the expected phototactic behavior. Ultimately, the algorithm allow the swarm to efficiently learn and aggregate in the center. The swarm is then put to more difficult learning configurations, with a more complex controllers and varying settings for HIT. The results show the capacity of the algorithm to maintain learning in this setting, and provide indications on the role of the protection window duration, a parameter of HIT, on the learning dynamics.

Chapter 5 provides experimental and theoretical insights on the role of morphology in robotic swarms, via a force-reorientation response provided by exoskeletons encapsulating the Kilobots. The *fronter* and *aligner* exoskeletons are presented, which produce robots that re-orient respectively against or along the direction of external forces. This changes the aggregation dynamics in phototaxis experiments, with the fronter type of re-orientation acting in favor of aggregation. The experiments are extended with simulations, showing the relation between the self-alignment strength and the clustering dynamics in the swarm. In addition, a third exoskeleton is introduced, with toothbrushes replacing the Kilobot legs. This exoskeleton is more inertial than the previous ones, leading to unseen oscillatory behavior, which we explain through the model of self-alignment.

Chapter 6 introduces a model of distributed learning as a system of coupled stochastic differential equations. This allow for the simulation of learning at higher scales, and provide insights both on the role of algorithmic parameters and morphological parameters on the aggregation dynamics of robotic swarms. This model opens up a way to a coarse-grained formulation of learning, which may guide future experiments involving a larger amount of robots acting simultaneously.

7.2 Perspectives

The work done in this thesis leaves many doors open for future research at the interface of Swarm Robotics and Active Matter.

On the learning part, the experiments of the Chapter 4 show that optimums exists in terms of initial conditions and algorithm parameters which maximize the performance and its increase over time. Additional theoretical work, with the model presented in Chapter 6, could provide good *a priori* for the parameters of algorithms such as HIT, for phototaxis or other collective tasks.

On the morphology side, building additional exoskeletons providing new values of ϵ/τ_n over a wider range of values would allow to validate experimentally simulation results. In addition, one could think of a modular version of such an exoskeleton that would allow the robot to pilot the morphological effects with internal degrees of freedom. This would further increase the link between morphology and learning, by allowing the robot to automatically leverage physical effects in favor of a task of aggregation, dispersion or collective motion.

The toothbrush exoskeleton could also be studied further, experimentally and in simulation, in order to understand the collective dynamics of inertial self-aligning particles. On the same line, it would be interesting to create an exoskeleton producing an inertial fronter, if ever that type of dynamics exists in the real world at all.

Other tasks can also be envisioned, such as collective transport, or binary phototaxis with two sources of light. Experiments putting forward the adaptation dynamics produced by HIT can also be envisioned, by moving the lit region mid-experiment to two or more different location. These new experiments could further tests the capabilities and limits of HIT, and provide experimental data to compare to the learning model of Chapter 6.

A Updates in the experimental setup

As the first set of experiments involving fronters and aligners was completed in November 2022, the experimental setup was updated again in order to improve the quality of the detection. The white mat PMMA 2mm slab was replaced with a black slab, again 2mm PMMA. The lightning of the experiment was also switched from white LEDs covered in red transluscent tape with bare UV 300nm LEDs. In addition to the UV, we changed the patterns glued to the exoskeletons to a new version, painted in yellow fluorescent paint, shown in Figure 88Fluorescent patterns used for the MLP inspired controller experiments.figure.caption.88. The paper absorbed part of the glue we used (cyanoacrylate) creating darker spots on the pattern, but these spots are invisible to the camera and do not disturb the detection.

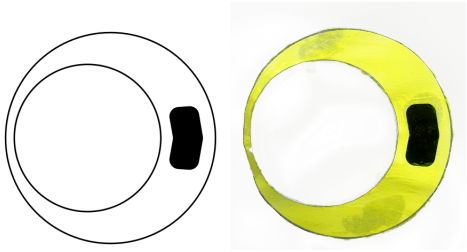


Figure 88 – Fluorescent patterns used for the MLP inspired controller experiments.

In addition to modifying the experimental setup, the fronter and aligner exsoskeletons were also slightly re-designed. The front and back legs were left untouched, in order to change as little as possible the dynamics, but the contour of the exoskeleton was made thicker by 2mm (totalling a thickness of 5mm) and additional "backup" legs were added. These "backup" legs are shorter than the actual legs and are placed left and right next to the straight rigid leg. In the case where the robot would fall over or tip on its side, these legs would come in contact with the ground and prevent the fall.

As explained in Chapter 3.2.1, the printing of exoskeletons involves cleaning the product from a soft resin, the support material. In the case of the new fronter, the geometry is such that removing the support by hand was nearly impossible without breaking a few legs. We resorted to print only the additional parts and to glue the print below the original exoskeletons.

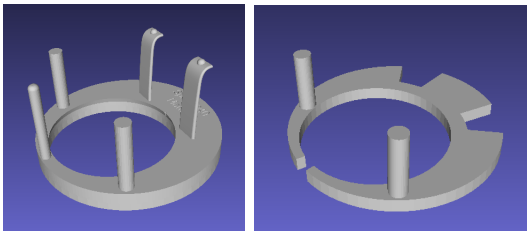


Figure 89 – Left : New aligner exoskeleton 3D Model. Right : New fronter addon.

Regarding the first device, the tube shown in Figure 40**Left** The cylindrical device used to bend the toothbrushes. The core is on the left, with placeholders for the toothbrushes, the tube is then slid over the core to achieve proper bending of the toothbrushes. **Right** Picture of a toothbrush before and after bending. The angle of deformation is not constant but close to 20° for a correctly bent toothbrush.figure.caption.40, it was designed in early 2021 with help from the ESPCI engineers. However, this device had a major flaw.

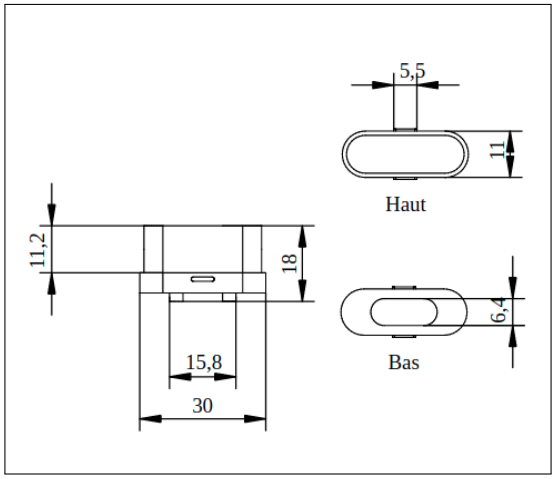


Figure 90 – Toothbrush head dimensions, measured in the lab and drawn in FreeCAD.

B Anecdote

Taken from Lehman et al. 2020 [66] :

> In a similarly underconstrained problem, William Punch collaborated with physicists, applying digital evolution to find lower-energy configurations of carbon. The physicists had a well-vetted energy model for between-carbon forces, which supplied the fitness function for evolutionary search. The motivation was to find a novel low-energy buckyball-like structure. While the algorithm produced very low-energy results, the physicists were irritated because the algorithm had found a superposition of all the carbon atoms onto the same point in space. "Why did your genetic algorithm violate the laws of physics?" they asked. "Why did your physics model not catch that edge condition?" was the team's response. The physicists patched the model to prevent superposition, and evolution was performed on the improved model. The result was qualitatively similar: great low-energy results that violated another physical law, revealing another edge case in the simulator. At that point, the physicists ceased the collaboration.

C Linear stability equations for self-alignment

Considering the free particle with inertia, we have the reduced set of equations along with the linear stability matrix :

$$\begin{cases} \tau_v \dot{v} = \cos\alpha - v \\ \dot{\alpha} = \omega - \frac{1}{\tau_v v}\sin\alpha \\ J\dot{\omega} = -\epsilon v \sin\alpha - \tau_n \omega + b \end{cases} \qquad \begin{pmatrix} \dot{v} \\ \dot{\alpha} \\ \dot{\omega} \end{pmatrix} = \begin{pmatrix} -\frac{1}{\tau_v} & -\frac{\sin\alpha^*}{\tau_v} & 0 \\ \frac{\sin\alpha^*}{\tau_v (v^*)^2} & -\frac{\cos\alpha^*}{\tau_v v^*} & 1 \\ -\epsilon\frac{\sin\alpha^*}{J} & -\epsilon\frac{v^*\cos\alpha^*}{J} & -\frac{\tau_n}{J} \end{pmatrix} \begin{pmatrix} dv \\ d\alpha \\ d\omega \end{pmatrix}$$

Writing $v^* = \cos\alpha^*$, the characteristic polynomial reads :

$$-\lambda^3 - \lambda^2\Big[\frac{\tau_n}{J} + \frac{2}{\tau_v}\Big] - \lambda\Big[\frac{1}{\tau_v^2} + \frac{2\tau_n}{\tau_v J} + \epsilon\frac{\cos^2\alpha^*}{J} + \frac{tan^2\alpha^*}{\tau_v^2}\Big] - \Big[\frac{\tau_n}{\tau_v^2 J} + \epsilon\frac{\cos(2\alpha^*)}{\tau_v J} + \frac{\tau_n tan^2\alpha^*}{\tau_v^2 J}\Big]$$

Considering the case of a constant force along the x axis, representative of the inclined plane experiment, we obtain the following set of equations along with the linear stability matrix :

$$\begin{cases} \tau_v \dot{v_x} = \cos\theta - v_x + g \\ \tau_v \dot{v_y} = \sin\theta - v_y \\ \dot{\theta} = \omega \\ J\dot{\omega} = \epsilon(\cos\theta v_y - \sin\theta v_x) - \tau_n\omega + b \end{cases} \quad \begin{pmatrix} \dot{v_x} \\ \dot{v_y} \\ \dot{\theta} \\ \dot{\omega} \end{pmatrix} = \begin{pmatrix} -\frac{1}{\tau_v} & 0 & -\epsilon\frac{b}{g\tau_v} & 0 \\ 0 & -\frac{1}{\tau_v} & \frac{\sqrt{1-\frac{b^2}{g^2}}}{\tau_v} & 0 \\ 0 & 0 & 0 & 1 \\ -\frac{b}{gJ} & \epsilon\frac{\sqrt{1-\frac{b^2}{g^2}}}{J} & -\epsilon\frac{\sqrt{g^2-b^2}+1}{J} & -\frac{\tau_n}{J} \end{pmatrix} \begin{pmatrix} dv_x \\ dv_y \\ d\theta \\ d\omega \end{pmatrix}$$

The characteristic polynomial of which reads:

$$(-\lambda - \frac{1}{\tau_v})\Big(-\frac{\lambda\epsilon(\sqrt{g^2-b^2}+1)}{J} - \frac{\epsilon(\sqrt{g^2-b^2}+1)}{\tau_v J} + \frac{\epsilon b^2}{g^2 J}\Big) + \frac{\epsilon(1-\frac{b^2}{g^2})(-\lambda-\frac{1}{\tau_v})}{\tau_v J}$$
$$-(\lambda^3 + \frac{\lambda}{\tau_v^2} + \frac{2\lambda^2}{\tau_v})(-\frac{\tau_n}{J} - \lambda)$$

Considering the case of an inertial robot in permanent contact with a wall along the **x** axis and assuming $J \neq 0$, close to the fixed point $v_y^* = \theta^* = 0$ we obtain the following equations and stability matrix :

$$\begin{cases} \dot{\theta} = \omega \\ J\dot{\omega} = (\cos\theta - \tau_r)v_y - (\tau_n + \frac{1}{2}\tau_r)\omega + b \\ \tau_v \dot{v_y} = \sin\theta - (1+\lambda)v_y - \frac{1}{2}\lambda\omega \end{cases} \quad \begin{pmatrix} \dot{\theta} \\ \dot{\omega} \\ \dot{v_y} \end{pmatrix} = \begin{pmatrix} 0 & 1 & 0 \\ 0 & -\frac{\tau_n+\frac{1}{2}\tau_r}{J} & \frac{1-\tau_r}{J} \\ \frac{1}{\tau_v} & -\frac{\lambda}{2\tau_v} & -\frac{1+\lambda}{\tau_v}\dot{v_y} \end{pmatrix} \begin{pmatrix} \theta \\ \omega \\ v_y \end{pmatrix}$$

The characteristic polynomial of which reads (using x as the variable) :

$$-x^3 - x^2(\frac{\tau_n + 0.5\tau_r}{J} + \frac{\lambda+1}{\tau_v}) - x(\frac{\lambda(\tau_n+0.5) + \tau_n + 0.5\tau_r}{\tau_v J}) + \frac{1-\tau_r}{\tau_v J}$$

D Unstable trajectories of inertial Fronters

Running simulations of the model given in Chapter 5.3.3 for the free fronter particle in an unstable regime yields a variety of cyclic trajectories of many different shapes. These trajectories are hinting at complex limit cycles, and maybe chaos.

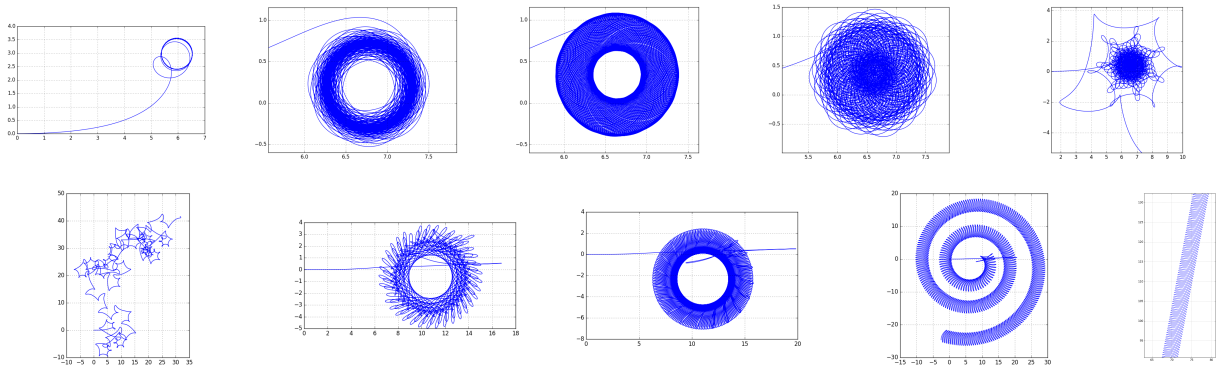


Figure 91 – Unstable trajectories of the inertial fronter. $\epsilon = -1, \tau_n = 0.5, J = 1, \theta_0 = 0.1, \mathbf{v}_0 = 1\mathbf{e_x}$. Values of τ_v, from left to right on the first row : 2.00, 7.80, 8.00, 8.05, 8.10. Second row : 10.00, 40.00, 50.00, 52.60, 53.00. The trajectories are integrated using scipy *odeint* function, with t_{max} from 10^3 to 10^4 τ depending on the plot and $\delta t = 10^{-4}$.

E New robotic platform : Pogobot

In a same time as the thesis, a new robotic platform for swarm robotics developed at ISIR, Sorbonne Université, was designed and built : the Pogobot, Figure 92. This robot measuring 6cm in diameter is open-source and open-hardware, with greater communication and processing power than the Kilobot. The robot is built from two PCBs (*Head* and *Belly*) encapsulated in a 3D printed exoskeleton, standing on toothbrushes. The mode of locomotion is, based on the the *stick-slip* principle, with an inspiration coming from the work done with the toothbrush exoskeletons used with the Kilobot, presented in this thesis.

The Pogobots offer a significant advance over the robots used for swarm robotics: they are small, inexpensive to manufacture, fast both in terms of movement speed and communication speed, and lastly: easy to program. These robots have been entirely developed with an open approach to both programming and electronics, to encourage their adoption beyond the perimeter of Sorbonne Université. Additional information can be found at `https://pogobot.github.io/`.

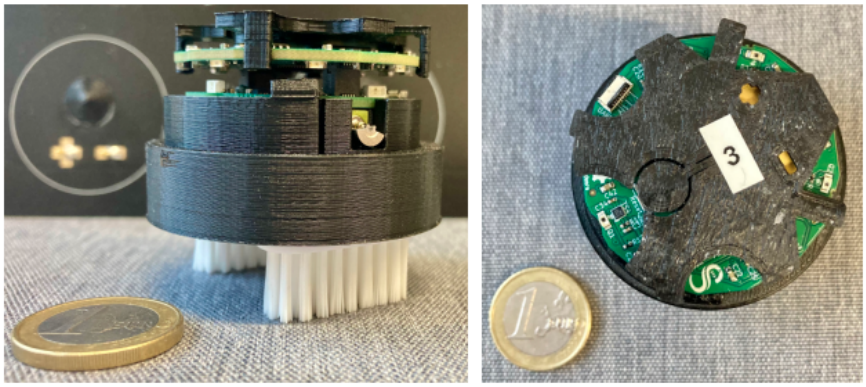


Figure 92 – The Pogobot, a 6cm in diameter robot developed at ISIR, Sorbonne Université. This robot is built to work in swarms, has a locomotion based on the *stick-slip* fashion like the Kilobot, but has greater movement speed, communication speed and computation capabilities compared to the Kilobot.

References

[1] Heiko Hamann. *Swarm robotics: A formal approach*, volume 221. Springer, 2018.

[2] Gerardo Beni. From swarm intelligence to swarm robotics. In *Swarm Robotics: SAB 2004 International Workshop, Santa Monica, CA, USA, July 17, 2004, Revised Selected Papers 1*, pages 1–9. Springer, 2005.

[3] Ying Tan and Zhong-yang Zheng. Research advance in swarm robotics. *Defence Technology*, 9(1):18–39, 2013.

[4] Erol Şahin. Swarm robotics: From sources of inspiration to domains of application. In *Swarm Robotics: SAB 2004 International Workshop, Santa Monica, CA, USA, July 17, 2004, Revised Selected Papers 1*, pages 10–20. Springer, 2005.

[5] Guang-Zhong Yang, Jim Bellingham, Pierre E Dupont, Peer Fischer, Luciano Floridi, Robert Full, Neil Jacobstein, Vijay Kumar, Marcia McNutt, Robert Merrifield, et al. The grand challenges of science robotics. *Science robotics*, 3(14):eaar7650, 2018.

[6] Martin Hägele, Klas Nilsson, J Norberto Pires, and Rainer Bischoff. Industrial robotics. *Springer handbook of robotics*, pages 1385–1422, 2016.

[7] Precedence research robotics market report 2022. https://www.precedenceresearch.com/robotics-technology-market. Accessed: 2023-14-08.

[8] Bruno Siciliano, Lorenzo Sciavicco, Luigi Villani, and Giuseppe Oriolo. *Robotics: Modelling, Planning and Control*. Springer Publishing Company, Incorporated, 1st edition, 2008.

[9] Markus Waibel, Bill Keays, and Federico Augugliaro. Drone shows: Creative potential and best practices. Technical report, ETH Zurich, 2017.

[10] Xin Zhou, Xiangyong Wen, Zhepei Wang, Yuman Gao, Haojia Li, Qianhao Wang, Tiankai Yang, Haojian Lu, Yanjun Cao, Chao Xu, et al. Swarm of micro flying robots in the wild. *Science Robotics*, 7(66):eabm5954, 2022.

[11] Roland Siegwart, Illah Reza Nourbakhsh, and Davide Scaramuzza. *Introduction to autonomous mobile robots*. MIT press, 2011.

[12] Anthony K. Ho. Fundamentals of pid control. University Lecture, 2020.

[13] Robin R Murphy. *Introduction to AI robotics*. MIT press, 2019.

[14] Erica Salvato, Gianfranco Fenu, Eric Medvet, and Felice Andrea Pellegrino. Crossing the reality gap: A survey on sim-to-real transferability of robot controllers in reinforcement learning. *IEEE Access*, 9:153171–153187, 2021.

[15] Michael Rubenstein, Alejandro Cornejo, and Radhika Nagpal. Programmable self-assembly in a thousand-robot swarm. *Science*, 345(6198):795–799, 2014.

[16] Shervin Nouyan, Roderich Groß, Michael Bonani, Francesco Mondada, and Marco Dorigo. Teamwork in self-organized robot colonies. *IEEE Transactions on Evolutionary Computation*, 13(4):695–711, 2009.

[17] Ali E Turgut, Hande Çelikkanat, Fatih Gökçe, and Erol Şahin. Self-organized flocking in mobile robot swarms. *Swarm Intelligence*, 2:97–120, 2008.

[18] Melvin Gauci, Radhika Nagpal, and Michael Rubenstein. Programmable self-disassembly for shape formation in large-scale robot collectives. In *Distributed Autonomous Robotic Systems: The 13th International Symposium*, pages 573–586. Springer, 2018.

[19] Justin Werfel, Kirstin Petersen, and Radhika Nagpal. Designing collective behavior in a termite-inspired robot construction team. *Science*, 343(6172):754–758, 2014.

[20] Jianing Chen, Melvin Gauci, Michael J Price, and Roderich Groß. Segregation in swarms of e-puck robots based on the brazil nut effect. In *Proceedings of the 11th International Conference on Autonomous Agents and Multiagent Systems-Volume 1*, pages 163–170, 2012.

[21] Nadia Nedjah and Luneque Silva Junior. Review of methodologies and tasks in swarm robotics towards standardization. *Swarm and Evolutionary Computation*, 50:100565, 2019.

[22] Manuele Brambilla, Eliseo Ferrante, Mauro Birattari, and Marco Dorigo. Swarm robotics: a review from the swarm engineering perspective. *Swarm Intelligence*, 7:1–41, 2013.

[23] Levent Bayindir and Erol Şahin. A review of studies in swarm robotics. *Turkish Journal of Electrical Engineering and Computer Sciences*, 15(2):115–147, 2007.

[24] Levent Bayındır. A review of swarm robotics tasks. *Neurocomputing*, 172:292–321, 2016.

[25] Karthik Elamvazhuthi and Spring Berman. Mean-field models in swarm robotics: A survey. *Bioinspiration & Biomimetics*, 15(1):015001, 2019.

[26] Marco Dorigo, Guy Theraulaz, and Vito Trianni. Reflections on the future of swarm robotics. *Science Robotics*, 5(49):eabe4385, 2020.

[27] Ahmad Reza Cheraghi, Sahdia Shahzad, and Kalman Graffi. Past, present, and future of swarm robotics. In *Intelligent Systems and Applications: Proceedings of the 2021 Intelligent Systems Conference (IntelliSys) Volume 3*, pages 190–233. Springer, 2022.

[28] Raffaello D'Andrea. Guest editorial: A revolution in the warehouse: A retrospective on kiva systems and the grand challenges ahead. *IEEE Transactions on Automation Science and Engineering*, 9(4):638–639, 2012.

[29] Daniel Albiero, Angel Pontin Garcia, Claudio Kiyoshi Umezu, and Rodrigo Leme de Paulo. Swarm robots in mechanized agricultural operations: a review about challenges for research. *Computers and Electronics in Agriculture*, 193:106608, 2022.

[30] Lindsey Hines, Kirstin Petersen, Guo Zhan Lum, and Metin Sitti. Soft actuators for small-scale robotics. *Advanced materials*, 29(13):1603483, 2017.

[31] Francesco Mondada, Michael Bonani, Xavier Raemy, James Pugh, Christopher Cianci, Adam Klaptocz, Stephane Magnenat, Jean-Christophe Zufferey, Dario Floreano, and Alcherio Martinoli. The e-puck, a robot designed for education in engineering. In *Proceedings of the 9th conference on autonomous robot systems and competitions*, volume 1, pages 59–65. IPCB: Instituto Politécnico de Castelo Branco, 2009.

[32] Farshad Arvin, Jose Espinosa, Benjamin Bird, Andrew West, Simon Watson, and Barry Lennox. Mona: an affordable open-source mobile robot for education and research. *Journal of Intelligent & Robotic Systems*, 94:761–775, 2019.

[33] Michael Rubenstein, Christian Ahler, and Radhika Nagpal. Kilobot: A low cost scalable robot system for collective behaviors. In *2012 IEEE international conference on robotics and automation*, pages 3293–3298. IEEE, 2012.

[34] Roderich Groß, Stéphane Magnenat, and Francesco Mondada. Segregation in swarms of mobile robots based on the brazil nut effect. In *2009 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 4349–4356. IEEE, 2009.

[35] Gerhard Gompper, Roland G Winkler, Thomas Speck, Alexandre Solon, Cesare Nardini, Fernando Peruani, Hartmut Löwen, Ramin Golestanian, U Benjamin Kaupp, Luis Alvarez, et al. The 2020 motile active matter roadmap. *Journal of Physics: Condensed Matter*, 32(19):193001, 2020.

[36] Sriram Ramaswamy. The mechanics and statistics of active matter. *Annu. Rev. Condens. Matter Phys.*, 1(1):323–345, 2010.

[37] Pierre-Gilles de Gennes. Soft matter (nobel lecture). *Angewandte Chemie International Edition in English*, 31(7):842–845, 1992.

[38] Maurice Kleman and Oleg D Lavrentovich. *Soft matter physics: an introduction*. Springer, 2003.

[39] Pierre-Gilles De Gennes and Jacques Prost. *The physics of liquid crystals*. Number 83. Oxford university press, 1993.

[40] Markus Bär, Robert Großmann, Sebastian Heidenreich, and Fernando Peruani. Self-propelled rods: Insights and perspectives for active matter. *Annual Review of Condensed Matter Physics*, 11:441–466, 2020.

[41] M Reza Shaebani, Adam Wysocki, Roland G Winkler, Gerhard Gompper, and Heiko Rieger. Computational models for active matter. *Nature Reviews Physics*, 2(4):181–199, 2020.

[42] Sebastian Weitz, Andreas Deutsch, and Fernando Peruani. Self-propelled rods exhibit a phase-separated state characterized by the presence of active stresses and the ejection of polar clusters. *Physical Review E*, 92(1):012322, 2015.

[43] A Kaiser, K Popowa, HH Wensink, and H Löwen. Capturing self-propelled particles in a moving microwedge. *Physical Review E*, 88(2):022311, 2013.

[44] Xia-qing Shi and Hugues Chaté. Self-propelled rods: Linking alignment-dominated and repulsion-dominated active matter. *arXiv preprint arXiv:1807.00294*, 2018.

[45] Amin Doostmohammadi, Jordi Ignés-Mullol, Julia M Yeomans, and Francesc Sagués. Active nematics. *Nature communications*, 9(1):3246, 2018.

[46] Andreas Walther and Axel HE Müller. Janus particles. *Soft matter*, 4(4):663–668, 2008.

[47] Paul Baconnier, Dor Shohat, C Hernández López, Corentin Coulais, Vincent Démery, Gustavo Düring, and Olivier Dauchot. Selective and collective actuation in active solids. *Nature Physics*, 18(10):1234–1239, 2022.

[48] Corinna C Maass, Carsten Krüger, Stephan Herminghaus, and Christian Bahr. Swimming droplets. *Annual Review of Condensed Matter Physics*, 7:171–193, 2016.

[49] François A Lavergne, Hugo Wendehenne, Tobias Bäuerle, and Clemens Bechinger. Group formation and cohesion of active particles with visual perception–dependent motility. *Science*, 364(6435):70–74, 2019.

[50] Grégoire Peyret, Romain Mueller, Joseph d'Alessandro, Simon Begnaud, Philippe Marcq, René-Marc Mège, Julia M Yeomans, Amin Doostmohammadi, and Benoît Ladoux. Sustained oscillations of epithelial cell sheets. *Biophysical journal*, 117(3):464–478, 2019.

[51] Antoine Bricard, Jean-Baptiste Caussin, Nicolas Desreumaux, Olivier Dauchot, and Denis Bartolo. Emergence of macroscopic directed motion in populations of motile colloids. *Nature*, 503(7474):95–98, 2013.

[52] Antoine Bricard, Jean-Baptiste Caussin, Debasish Das, Charles Savoie, Vijayakumar Chikkadi, Kyohei Shitara, Oleksandr Chepizhko, Fernando Peruani, David Saintillan, and Denis Bartolo. Emergent vortices in populations of colloidal rollers. *Nature communications*, 6(1):7470, 2015.

[53] Julien Deseigne, Olivier Dauchot, and Hugues Chaté. Collective motion of vibrated polar disks. *Physical review letters*, 105(9):098001, 2010.

[54] Julien Deseigne, Sébastien Léonard, Olivier Dauchot, and Hugues Chaté. Vibrated polar disks: spontaneous motion, binary collisions, and collective dynamics. *Soft Matter*, 8(20):5629–5639, 2012.

[55] L Giomi, N Hawley-Weld, and L Mahadevan. Swarming, swirling and stasis in sequestered bristle-bots. *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 469(2151):20120637, 2013.

[56] Olivier Dauchot and Vincent Démery. Dynamics of a self-propelled particle in a harmonic trap. *Physical review letters*, 122(6):068002, 2019.

[57] Windell Oskay. Bristlebot diy tutorial. https://www.evilmadscientist.com/2007/bristlebot-a-tiny-directional-vibrobot/, 2007. Accessed: 2023-29-08.

[58] DeaGyu Kim, Zhijian Hao, Jun Ueda, and Azadeh Ansari. A 5 mg micro-bristle-bot fabricated by two-photon lithography. *Journal of Micromechanics and Microengineering*, 29(10):105006, 2019.

[59] Hebug nano webpage. https://www.hexbug.com/nano.html. Accessed: 2023-29-08.

[60] Bruno Siciliano, Oussama Khatib, and Torsten Kröger. *Springer handbook of robotics*, volume 200. Springer, 2008.

[61] Shadow hand robot official website. https://www.shadowrobot.com/. Accessed: 2023-30-05.

[62] Eric Brown, Nicholas Rodenberg, John Amend, Annan Mozeika, Erik Steltz, Mitchell R Zakin, Hod Lipson, and Heinrich M Jaeger. Universal robotic gripper based on the jamming of granular material. *Proceedings of the National Academy of Sciences*, 107(44):18809–18814, 2010.

[63] Shuguang Li, Richa Batra, David Brown, Hyun-Dong Chang, Nikhil Ranganathan, Chuck Hoberman, Daniela Rus, and Hod Lipson. Particle robotics based on statistical mechanics of loosely coupled components. *Nature*, 567(7748):361–365, 2019.

[64] Ethem Alpaydin. *Introduction to machine learning*. MIT press, 2020.

[65] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

[66] Joel Lehman, Jeff Clune, Dusan Misevic, Christoph Adami, Lee Altenberg, Julie Beaulieu, Peter J Bentley, Samuel Bernard, Guillaume Beslon, David M Bryson, et al. The surprising creativity of digital evolution: A collection of anecdotes from the evolutionary computation and artificial life research communities. *Artificial life*, 26(2):274–306, 2020.

[67] Christopher John Cornish Hellaby Watkins. Learning from delayed rewards. 1989.

[68] Jorge Gomes, Paulo Urbano, and Anders Lyhne Christensen. Evolution of swarm robotics systems with novelty search. *Swarm Intelligence*, 7:115–144, 2013.

[69] Tim Salimans, Jonathan Ho, Xi Chen, Szymon Sidor, and Ilya Sutskever. Evolution strategies as a scalable alternative to reinforcement learning. *arXiv preprint arXiv:1703.03864*, 2017.

[70] Stephane Doncieux, Nicolas Bredeche, Jean-Baptiste Mouret, and Agoston E Eiben. Evolutionary robotics: what, why, and where to. *Frontiers in Robotics and AI*, 2:4, 2015.

[71] David M Bryson and Charles Ofria. Understanding evolutionary potential in virtual cpu instruction set architectures. *PLoS One*, 8(12):e83242, 2013.

[72] Nicolas Bredeche, Evert Haasdijk, and Abraham Prieto. Embodied evolution in collective robotics: a review. *Frontiers in Robotics and AI*, 5:12, 2018.

[73] Jonas Kuckling. Recent trends in robot learning and evolution for swarm robotics. *Frontiers in Robotics and AI*, 10, 2023.

[74] Nicolas Bredeche, Jean-Marc Montanier, Wenguo Liu, and Alan FT Winfield. Environment-driven distributed evolutionary adaptation in a population of autonomous robotic agents. *Mathematical and Computer Modelling of Dynamical Systems*, 18(1):101–129, 2012.

[75] Simon Jones, Alan F Winfield, Sabine Hauert, and Matthew Studley. Onboard evolution of understandable swarm behaviors. *Advanced Intelligent Systems*, 1(6):1900031, 2019.

[76] Pawel Romanczuk, Markus Bär, Werner Ebeling, Benjamin Lindner, and Lutz Schimansky-Geier. Active brownian particles: From individual to collective stochastic dynamics. *The European Physical Journal Special Topics*, 202:1–162, 2012.

[77] Michael E Cates and Julien Tailleur. Motility-induced phase separation. *Annu. Rev. Condens. Matter Phys.*, 6(1):219–244, 2015.

[78] Francesco Ginelli. The physics of the vicsek model. *The European Physical Journal Special Topics*, 225:2099–2117, 2016.

[79] Craig W Reynolds. Flocks, herds and schools: A distributed behavioral model. In *Proceedings of the 14th annual conference on Computer graphics and interactive techniques*, pages 25–34, 1987.

[80] Ben Eater. Boids algorithm demonstration. https://eater.net/boids, 2020. [Online interactive content; accessed 11-July-2023].

[81] Tamás Vicsek, András Czirók, Eshel Ben-Jacob, Inon Cohen, and Ofer Shochet. Novel type of phase transition in a system of self-driven particles. *Physical review letters*, 75(6):1226, 1995.

[82] Hugues Chaté, Francesco Ginelli, Guillaume Grégoire, and Franck Raynaud. Collective motion of self-propelled particles interacting without cohesion. *Physical Review E*, 77(4):046113, 2008.