Study of Ants' Collective Behaviours and Mechanisms to Reflect on Possible Implementations of Software / Robotic Simulations

Alexandre Jean-Pierre Vanini
Computer Science, IT University of Copenhagen, Rued Langgaards Vej 7, 2300
Research Project (K-CS and K-SD) Autumn 2020, KIREPRO1PE
alva@itu.dk

Abstract

Ants are among the most simple beings out there. However, they can do great things such as building structures, communicating in a very non-conventional fashion, concentrating their workforce to gather supplies, and defending their nest. This document references the lives of ants from what they are and where they come from to how they behave to collect emerging collective behaviours and mechanisms to implement in different simulation approaches. The first type of simulation is robotic-based and aims to fetch one to four ant's actions to replicate the complexity of these when applied to swarm robotic in a reduced setup. The second approach is a software-based simulation where programming languages such as JavaScript or C will be studied to evaluate if they fit such a task's needs and to what level of accuracy one can replicate ants' behaviours. A third approach will be to assess the possibility of mixing the first two types of simulation to use the advantages of simulating some behaviours ahead of an implementation of the real-life setup.

1 Introduction

This project is part of the Research project (K-CS and K-SD) Autumn 2020 (KIREPRO1PE) ITU course, which intends to be a bridge between the specialisation course and the Master's thesis (that is, a preliminary work for the Master's thesis). Its aim is the study of the abstract mechanisms employed by ants. Ants are one of the various social species that can be studied to create bioinspired algorithms and collective behaviours models. They are fascinating because, as a single unit, an ant is a simple being. However, as a swarm, they show many exciting collective behaviours which underlying's mechanisms can be studied and used for everyday society and science problems.

The art of studying nature to replicate its behaviours into a robotic/algorithm to solve complex human problems is called "bio-inspired robotic" (also more commonly know as "Biomimetic" [55]). There are numerous examples, such as the famous "Japanese Bullet train" (Shinkansen), Figure 1, which got its nose design from the Kingfisher bird's beak's aerodynamic, reducing the train's energy consumption by 15%, making it 10% faster and quieter [1].

Figure 1: The Shinkansen bullet train, inspired by the Kingfisher bird (Masayuki Kozono, 2019).

The hook and loop fastener, shown in Figure 2 (also known as the Velcro), created by the Swiss engineer George de Mestral in the 1950s is also a very famous biomimetic example. This two-part binding system was inspired by burrs, which de Mestral studied under a microscope after he figured how surprisingly easy these would stick to its dog's hair. He discovered that burrs had micro hooks that were able to catch anything with a loop. Nowadays, the hook and loop fastener is known and used throughout the world [1].

Bio-mimetic is also very useful in robotics as today's human problems get more complicated, risky, and costly. For instance, Amazon uses an extensive net-

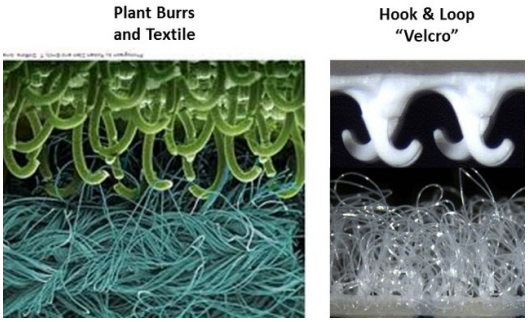

Figure 2: Comparison between plant burrs and the Hook loop G. de Mestral innovation (J. David Frost, M. Mahdi Roozbahani, Kendra Jackson, 2017).

work of swarm robots in their warehouse to control the package so that the human workers on site do not even have to move. The ordered package comes to them; they fill the box, staple it, and put it back on the robot to be ready for delivery [2]. Another excellent and relevant example is the network's Honey bee algorithm. Back in 1988, in the Georgia Institute of Technology, Professor John Hagood Vande Vate discovered, after some discussion, how effectively nectar forager honey bees could distribute themselves effectively among the local clusters of flowers without any central authority. The professor and his team of computer scientists built up an algorithm to reflect the seen behaviour. For a decade, this algorithm sat unused until an Oxford student named Sunil Nakrani needed inspiration for a web hosting problem. Sunil and the professor refined the honey bee algorithm to fit the situation and dramatically made Internet Web hosting services more efficient [3].

2 The Ant Kingdom

This section is about ants, their history, birth, life, and how they function and behave in groups and individually. As we are restrained in time and space, we will go through only a very few of the enormous amount of exciting species that exist out there. The end goal is to give ourselves an idea of which ant species are good candidates for a simulation and highlights fascinating biological behaviours that emerge within each species.

2.1 Ants

Ants are ancient beings that emerged around 140 to 168 million years ago, even though they started to diversify only about 60 million years after [4] [5]. From the dinosaur era to the Human realm, no kingdom has ever been as large and as powerful as the ants' superkingdom. They are estimated to be about 10'000'000'000'000'000 (10'000 trillion) individuals [6], which accounts for 20% of the total animal biomass [7] [8]. This superkingdom, however, is far from being an unified and coherent heaven. From the 16'000 classified species and the 20'000 estimated species [5], rare are the ant species that resemble one another. Indeed, from the very early age of their existence, ants have been evolving and adapting to a large amount of environments, from extreme heat desert to rainy forests and swamps. The only places that remain (almost) untouched are cold and high places on earth, as ants need a heat source to survive [9]. Individually, ants cannot do much, they are too simple, and their brain is only capable of performing a limited set of actions. To become what ants are today, they needed something fundamentally important: Collaboration. It is like the human race. We, throughout the years, had to come up with some very complex collaboration tools and framework to construct the society we have nowadays.

Throughout this collaboration, ants are capable of achieving the greatest. They can assemble themselves into complex structures such as bridges or boats to overcome environmental challenges; they achieve agriculture through foraging and maintenance and have complex symbiotic relationships. However, this collaborative behaviour only emerges within ants of the same colonies (and sometimes, but less likely, of the same species) [10]. Ant colonies are more likely to fight or avoid contact than peacefully plan about what would be the best strategy to get the piece of bread from a picnic and how it would be beneficial for each of them. So what is the difference? What makes them less able to collaborate with other species/colonies? As described earlier, ants have a limited set of communication tools and understanding of the world, which have made them less able to develop collectiveness compared to other species during the evolution. At their level, everything but the colony and the queen's survival is meaningless.

2.2 Interesting Species

2.2.1 The Army Ants (Marabunta)

From all the known ants' species, there is a category of ant legitimately called "the army ants". This category that the evolutionary tree has brought to be combatants are dreadful warriors among the ground. They have large mandibles, painful sting, and armour, and there exists about 200 species of them [11]. If two colonies encounter one another in nature, it is unlikely that they will fight against each other. Indeed, this makes sense from an evolutionary standpoint as, throughout time, army ants attacking other army ants eradicated them-

selves to extinction [12]. They do not have a nest; they would rather crawl on the ground in a very long and narrow group, consuming up to 500'000 prey animals each day [13] [14]. The behaviour of crawling on the floor is something specific to the army ants called the "Army ant syndrome", described in figure 3 [15].

In the first two to three weeks, a long phase called the stationary phase, the preys previously used to feed to the larvae are now reserved to the queen for her to prepare her birth cycle. The queen lays her eggs, and once the colony reaches the end of the stationary phase, the newborn larvae emerge from their cocoons, and the colony prepares itself to enter the second phase [13].

The second phase, called the "Nomadic phase," lasts approximately 15 days, giving the newborn larvae time to develop into fully functional workers. In this phase, the colony moves during the day, etching on the ground and capturing everything they can, from insects, spiders, and even dead ant bodies. Usually, an ant colony would send one or a few scouts to look around the colony and seek food or threats. However, the army ants are leaderless foragers and like to overwhelm the prey at once by sending many combatants. At the end of the 15 days, the larvae no longer require food, and the colony can again enter the stationary phase [13] [16].

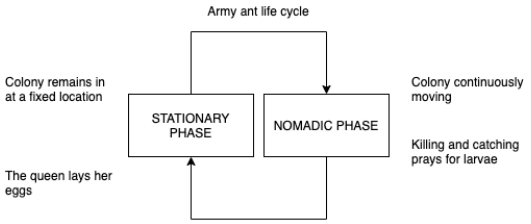

Figure 3: The Army ants life cycle, Stationary and Nomadic phase described.

2.2.2 Carpenter Ants (Camponotus Spp)

Carpenter ants, represented in Figure 4, are inhabitants of many forests in the world. They like to build their nest in high humidity woods, and they alone account for approximately 1'000 ant species. If carpenter ants are among the most known species globally, they are sadly famous for being a nuisance to many buildings and structures made of wood, causing structural damages. Nevertheless, carpenter ants are interesting beings. A mechanism to highlight from their collective behaviour is that, once their primary nest is big enough, carpenter ants like do build satellite nest containing workers while the eggs, the newly hatched larvae, and the queen remain in the primary nest. [17] [18]

Figure 4: A Carpenter ant (Muhammad Mahdi Karim, 2009).

2.2.3 Leafcutter Ants (Genera Atta - Acromyrmex)

Figure 5: Leafcutter ants (Kathy Sam, N/A).

Leafcutter ants (Figure 5) are tiny ants of about 47 species that can carry around 20 times their bodyweight. If an average-sized human could do that, he would be able to pull up to 1400kg [19]. As their name may imply, Leafcutter ants cut leaf fragments, which they bring back to the nest to feed the fungus garden. The fungus garden is the primary source of the colony's supply, and they take good care of it. The relationship between the ants and the fungus garden (called an ant-fungus mutualism) is so complex that it can generate chemicals to tell the ants if the fragments they brought back from the foraging are toxic to it or not. However, the care of the fungus garden does not end here. Indeed, a worker category specialised in taking care of it brings the toxic leaf and dead leaves to a waste heap. These waste transporters are the older, more dispensable leafcutter ants that ensure that the younger and more functional workers can work safely in the fungus garden. Once the waste is deposed in the waste heap, workers frequently organise the debris to help decomposition [20] [21].

Colony Hierarchy
Like any other ant species, Leafcutter ants are a well-organised group. It exists four main classes, starting with the "minims". The minims are the smallest worker; they take care of growing the brood and ensure the fungus garden is in safe hands. The second category, the minors, are slightly larger workers, which are the first line of defence against external threats. Their main occupation is to patrol around the nest and making its surrounding a safe place. The third category, depicted in all photography, is the "real" Leafcutters. They are generalised foragers and wander outside the nest to cut leaves fragments and bring them back to the nest. The fourth and last category is the Majors, which are the largest worker ants. They are the soldier of the nest, defending and attacking against outside threats. Moreover, this ants category also helps in other tasks such as cleaning the foraging trails from extensive debris. [19]

2.2.4 Pharaoh Ants (Monomorium Pharaonis)

Figure 6: Monomorium pharaonis worker with single sugar crystal (Julian Szulc, 2011, CC BY 3.0).

Pharaoh ants (Figure 6) are infamously known as the largest indoor nuisance pest as they can establish themselves in most parts of the world, given a reliable heat source. Their vast territory is due to high human migration movements and exponentially growing access to transports such as boats, planes, or cars. Pharaoh ant colonies are polygynous, which means they can contain more than a single queen (the highest number of queens found in a single colony is 200). To move and choose a convenient nest, they use decision-making behaviour to minimise the time the colony is without a nest [22] [23].

Pheromones

Pharaoh ants have three exciting types of pheromones. The first one is a long-lasting attractive chemical used to build trail networks that can remain detectable even if they are unused for several days. This durable chemical has been studied on this particular type of ants, and it has been highlighted that even if pharaoh ants cease activity at night, the trail networks are identical each day. The second one is a mid-lasting pheromone, which usually disappears in a matter of minutes. This type of pheromone is used to indicate a path to a food supply. Finally, the last type, which decays after two hours, is a repellent pheromone (or negative trail), which indicates that a path to a food supply or a resource is inefficient [24] [22].

3 Ant Behaviours and Mechanisms

Throughout this section, we will explore with a more in-depth approach new behaviours and mechanisms to see how collectiveness can emerge when every colony element activates itself to a task. The aim is also to define exciting behaviours that can be used either from a robotic standpoint or a software standpoint, later in the Master's project.

Firstly, let us define what "collective behaviours" truly means. Collective behaviour is a process without central control that brings together multiple participants to achieve some outcome [25]. Ants are capable of building complex structures; they can defend their nest as a group when a threat knocks on their door and are capable to organise to achieve such tasks. So how is it that such simple beings are capable of the greatest achievement? This is the question that we will elaborate on in the next sub-chapters.

3.1 Organization

How do ants organise? Even though ants are simple beings, their collaborative behaviour is extraordinarily complex. Each colony has the same structure, but it may slightly variate from a species to another. A colony begins with its queen. The queen is not a hierarchal label as she does not have absolute power over what the ant colony is doing, but she is the only type of ant that will give birth to young workers. It has been found that some queen may lay up to 1000 eggs a day for up to seven years (a total of 2'492'000 eggs). The larvae are the growing workers laid by the queen and are sources of high tension between colonies when an attack occurs as some species like to steal them as food supply or even use them as new workers in the opponent's colony [26]. Growing to be adult, the larvae will then perform tasks rather than being attributed a role; see Task allocation section.

3.2 Communication

As mentioned already in this paper, ants do not have as complex communication tools as humans do. However, they can generate collective behaviours in colonies that sometimes contain up to a few ten million individuals. Ants can communicate through 3 distinct channels; The first one is low resonance sounds, which ants can produce by scraping their legs on some specific part of their body to create different sounds. This communication method is mostly used when deposing pheromone trails would not suit the situation, such as helping a lost ant to find its way back through a tunnel or a narrow set of rocks. The second one is body language, which they can use to generate primitive reflexes (such as opening the ant's mouth to feed it) in other ants by touching them in specific areas. Finally, the scent, which ants use to detect the pheromone trail left by other ants. As mentioned earlier in this paper, pheromones are a unique chemical badge among ant colonies, which serves many purposes. These pheromones can be used for navigation, telling an ant to find a food supply or a source of water, and used by the queen to influence the workers. It has been found that some pheromones used by the queen had the effect of calming or exciting the workers in specific situations [27] [28] [29].

3.3 Navigation and Pheromone Trails

The pheromones mentioned in the previous section are mainly used for ants to orientate. Whenever a forager comes back from a food supply leaving behind it a pheromone trail indicating the source's location, another ant is likely to fall upon it and to follow it, ultimately leading it to the source. If there are different trails to choose from, the ant will decide which one to choose given a probabilistic model where the more pheromone there is on a path (that is, left by multiple other workers), the more likely the ant is to follow it. Some other types of pheromones are also used as a repellent, as depicted in the Pharaoh ants section of this document. This navigation tool has already been studied a lot and has given birth to many papers and a fascinating algorithm used today to find optimal (shortest/-fastest) path for maps and graphs; The ant pathfinder algorithm.

The ant pathfinder algorithm works as follows: The ants are dispatched randomly from a given starting point to the adjacent edges of a graph, wandering to other edges, replicating foraging behaviours. Once an ant finds the destination node, it backtracks to the graph's starting point, leaving behind it the so-called "pheromone trail" indicating its destination. Likely, other ants will also find a path and return to the nest, leaving behind them the trail. Now that we have multiple likely paths, how does one figure which one has the shortest distance? Exactly like ants do. Indeed, as mentioned above, the shorter the route is, the more likely another ant will encounter it and mark it with its pheromone, the higher the level on the trail will be, leading to even more ants following this track (see Figure 7). After a few generations, the shortest pheromone trail, as it is the one ants encountered the most, will remain visible, where longer trails who suffered from a lack of pheromone have evaporated. [30] [31] [32]

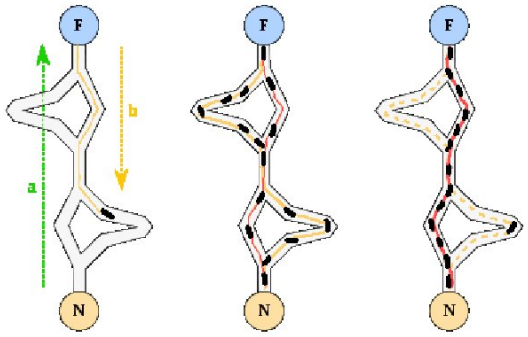

Figure 7: Depicts the ant pathfinder algorithm in three phases. The first phase is the algorithm's exploration phase, where ants are dispatched randomly on the edge of the graphs' neighbouring nodes, eventually finding the destination and coming back to the source. The second phase is the reinforcement phase; the shorter path will be reinforced as more ants fall upon it. The third phase is the eventual evaporation of the longer path. Note that, as seen in phase three and said above, choosing if the ant will be going on a given trail or not follows a probabilistic model, meaning that ants may explore other paths. After a while, the longer paths evaporate, and the shortest remains and is finally visible (Mohammed Ahmed AlhanjouriBelal Alfarra, 2011, CC BY 4.0).

3.4 Tasks

Tasks may variate slightly from a colony to another, as in leafcutter ant's colony where some ants have to maintain the fungus garden, but overall we can sum them up in 4 tasks:

- Foraging
- Scouting
- Patrolling
- Nest maintenance

Scouting
Scouting is the action of going outside alone or with a minimal crew to search for food. Once the scout has found a food source, it will rush back to the nest (leaving behind it a pheromone trail) to "warn" the colony that supply is available. Scouting can also serve as attack planning. If an ant found a potential threat (that is, an enemy colony), it will go back to its nest to prepare a defence if needed.

Foraging
When foraging, ants will move randomly in many different directions to increase their chances of encountering food or a positive pheromone trail. If the ant ever finds food, it will subsequently crawl home, leaving behind it a pheromone trail indicating the other ants where the food is. The more appealing the food is (in quantity), the more ants will switch their current task to get to the food area. If the ant fails to return to the colony in time, the pheromone trail eventually evaporates, and no ant will follow the trail back to the food supply [27]. Ants also have a system of "memory of high reward foraging sites"; the more food there is at a specific site, the higher the reward is going to be, leading to the ant returning to this foraging site in the future, given that it is a static site (not a spontaneous drop of supply) [33] [34].

Patrolling
Patrollers are ant guards watching over the nest, making sure no threat is near around. Like the scouts, if they encounter danger, they will rush back to the nest and warn everyone, keeping the nest a safe and sound place.

Nest Maintenance
Nest maintenance is the most massive task an ant can perform; it goes from moving the newly laid eggs from the queen to the eggs chamber, taking care of a fungus garden as for the Leafcutter ants or other ant-fungus like types of ant, making the nest bigger, to cleaning it from dead bodies, food waste and such. Ants who perform such tasks are mainly older than the ones performing off-site tasks such as foraging and patrolling as their old body age results in low performances [35].

3.4.1 Task Allocation

As we have mentioned earlier, there is no central control unit in a colony of ants, meaning that there is no one to tell an ant what to do at any given time, and yet ants seem to have a pretty busy day-to-day life. Few studies have suggested that ants can select a task based on their age, body size, genetic background, position in the nest, the way they eat, and receiving signals from other ants [36]. The earliest studies have highlighted that ants can determine what task it should do with a brilliant decision model [25]. This decision is based on the rate at which an ant encounters another one. Indeed, every time an ant makes contact, there is a small probability that the ant will switch to the encountered ant's task. This also means that the more ants with the same task our ant encounters, the more likely it is to change (note that this decision-making is only based on interaction and location). Ultimately, larger ant colonies are better at task allocation than smaller ones.

Switching to a task is one thing; being aware of what the other ant's task is, is another. How is it that an ant knows what task to switch to even though we have stated multiple times in this paper that ants do not have any complex communication tools? It is because ants have been given antennas which they rub against encountered ants' antennas. This contact gives information on what task the other ant is currently performing, and does so by analysing the cuticular hydrocarbon the antennas carry—these cuticular hydrocarbons variate regarding where the ant previously was and what task it was performing, e.g. when ants are foraging outside their nest in the sun, the proportion of n-alkanes in their hydrocarbon profile rises, leading the ants to smell seemingly different from ants working inside the nest [25]. Age is also a variable to take into consideration. Even though it has been found that this does not impact task allocation on a large scale, studies have found that older ants would usually take care of the maintenance of the nest and any task that happens inside the nest [35]. In contrast, younger workers would more likely be outside to forage and defend the nest in case of attack. Deborah M. Gordon, a worldwide known scientist famous for her lifetime studies on ants, has discovered that not every ant would switch to a specific task even though the probability is high. She found that ants have "preferences" [35] to which task they would be willing to change to, and she has then dressed a map of this behaviour, shown in figure 8.

Environmental Challenges The environmental variables are also settings ants take into consideration when choosing or performing a task. Indeed, In some places in the world, such as deserts or highly dry terrains, the task's operating cost is expensive as finding resources is hard, meaning that an ant will think twice before allocating some of its time scouting for food outside as it only does it if the returned income is higher than the one spent to get to the supply. This high operating cost behaviour can be summed up as "Do not go unless something positive happens." In some other places of the world, the operating cost is low (houses, groceries, tropical forest, etc.), meaning that the energy an ant has to spend to reach a resource is little compared to the income it will return. This is the "Go

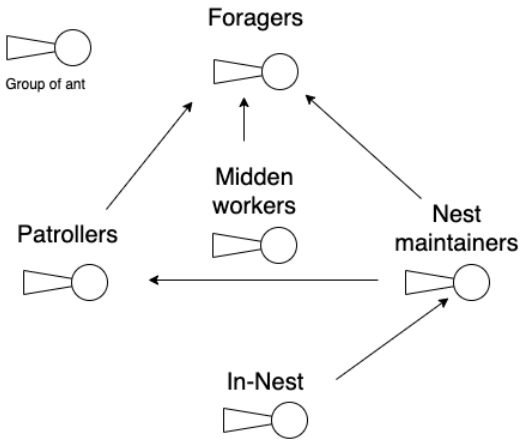

Figure 8: Depicts that only in-nest ants (the older or/and inactive ants) would switch to nest maintenance if the demand were high in more critical tasks such as foraging, and that the highest task priority is foraging as every ant can eventually ending in switching up to it (Alexandre Jean-Pierre Vanini, 2020).

unless something bad happens" behaviour [37]. Such studies have also been conducted in threatening human environments such as space, where a small colony was filmed to study these behaviours. The research has shown that ants would not move unless the colony's existence was at stake [38].

3.5 Lifecycle of an Ant Colony

Ant colonies are part of an endless cycle, and only way a colony could come to its extinction is if a natural or human catastrophe wipes them down. The same way humans give birth to a descendent tree, the ants will give birth to "child" colonies and even "grandchild colonies" where chemical badge between generations only variate from a few degrees. To give birth to a new colony, the original colony must first come to a certain level of maturity where the winged male will eventually fly out to mate with the winged female (and die from this process). The newly proclaimed queen then digs a hole and lays her first eggs, feeding them from her body fat reserve. The queen will use the sperm from the first time she mated with males and will keep it to lay eggs for the rest of her life (up to 15-20 years) [39] [35].

4 Simulations

This section goes through the many behaviours we have seen throughout this paper and a few possible approaches to simulate these. We will first discuss the robotic side, approaching the subject with a more pragmatic approach and definition of needs and environmental restrictions to replicate a limited predefined set of collective behaviour. We will then elaborate on the design choices and reflection of creating a software-based simulation, studying languages, technics, and frameworks.

4.1 Robot

This section describes and defines a limited set of collective behaviours taken from ants and transposed to real-life robot condition and environment, and describe the requirements and limitations of implementing such behaviours in real life. This set of collective behaviour is purely arbitrary and can always be disregarded.

Swarm robotic is the art of collectively running many robots to solve problems by forming complex structures and behaviours, such as the one observed in nature. They are scalable systems made of simple, often homogeneous agents, as shown in Figure 9. As in ant or bee colonies, swarm robotic does not have any central control unit, meaning that each agent knows what to do and how to communicate with nearby individuals. [40]

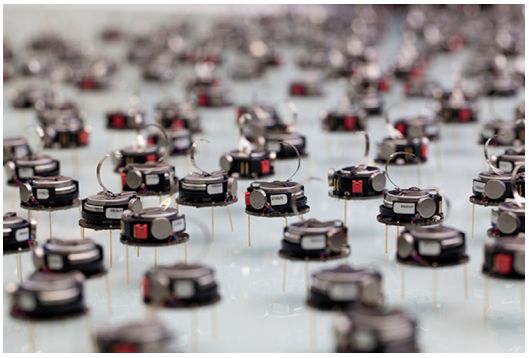

Figure 9: Shows a small agent named "KiloBots" created at Harvard University. By vibrating small legs with a motor, these tiny robots can realise horizontal, vertical, and diagonal motion. They are equipped with transmitters and receivers, and light detectors, allowing them to perform complex tasks such as arranging themselves in complex shape structures or following light sources to replicate low-level swarm behaviours. (David Edwards (Robotics Automation), 2020)

4.1.1 Relevant Collective Behaviours

Where in the software simulation almost everything is possible since it is computer-generated, the real-life setup restrains our field of possibilities. The setup cost is also to be taken into consideration. There are also some out-of-the-box problems, such as recharging the

robot's battery every time it runs off of battery, leading to losing time to perform the actual real-life simulation. Fear of battery loss also means that the robot cannot venture as far as it wants and is limited in what kind of modules (sensors, motors, actuators, etc.) it adopts. Space and time are also concerns as the setup in which the agents will be moving will not be as large as in a software simulation. Indeed, such design usually takes place at home or at school where only a few square meters are given. Time is also a variable because one cannot imagine speeding up the real world to see the results more quickly.

Another large part of today's robotics problem lay in the hardware. In real life, when dealing with hardware, many problems can occur, such as loss of power, less energy in the wheel leading to strange behaviours, unexpected sensor feedback because of the current environment, inaccuracy in the robot's movement, and way more. A paper written in 2013 by Madhav Patil, Tamer Abukhalil, and Tarek Sobh depicts more of these problems as they try to build and design a robot system, see ref. [41]

Having so many limitations also means that the collective behaviours one can simulate are somewhat limited. Nevertheless, there are three collective behaviours that stands out as they are quite complex, interesting to implement, and answer modern and real-life questions and problems:

- Task allocation
- Operating cost of a task
- Pathfinders with Pheromone trails.

Task allocation is interesting because it is a modern problem. Indeed, if swarm robotic is tomorrow's future, we will need to precisely be able to tell an agent what its task is, or more precisely, it will have to decide itself what task to choose. Imagine building a bridge with the help of a robotic swarm to counter a flood in a small village, each agent composing the safety bridge needs to be able to know what location to go to. This task allocation will work as described in the sub-section "Task allocation" where an agent decides to switch to another task based on a probabilistic model. The operating cost of a task could be defined as a subsection of task allocation or as a variable. What will be the cost for an agent to move to a given area? Will it have enough battery to perform the task, or is it even capable of enduring the journey? Operating cost of tasks coupled with task allocations will answer these questions. Finally, even though pathfinder already is a widely studied topic, it is still interesting to include it as it is how an ant ultimately moves. If the simulation gets too elaborate, it could be imagined to invent a less complicated way of movements, such as random-based movement or specific probabilistic distribution-based movement.

4.1.2 Implementation

We will define a set of features the physical robot needs to complete the tasks. A collection of features can go from wheels, motors, or even wings, to the many types of sensors a robot can be equipped with. This way, we will be able to define what kind of robot should be used and how the task should be implemented. The first need is a brain; even though ants are simple beings, we still need to be able to send information to the different parts of the robot and control them. This brain can be simulated with a Raspberry Pi (a small, fully functional computer) or a robot that already contains a motherboard and a control unit. Secondly, a communication system will be useful to share the robot's chemical badge to another one as to imitate when ants touch antennas, which will make it possible to achieve "task allocations" by sharing pieces of data on the currently performed task. Thirdly, we need a way to simulate movement, and since ants are not flying insects (unless they are winged males or females), wheels are the easiest way to replicate their movement. Fourthly, it would be nice if the ant robots would not collide every time they run onto each other, and to do that, the robot will have to be equipped with proximity sensors, most likely at the front, to avoid a collision. Finally, we need to simulate the pheromone trails in some way, and it is likely going to be the biggest challenge because the robots are not yet capable of "sensing" odour as ants or humans do, or they do it minimally.

A study from 2014 made by Ryusuke Fujisawa et al. [42] demonstrates the complexity of such an implementation, where the robots were capable of diffusing ethanol on the ground for the other robots to sense, using an alcohol sensor which detects the proportions of alcohol gas in the air. The main problem of such a solution is that it considerably raises the complexity of the build as to depose alcohol on the ground the robot requires to be equipped of a tank (which serves as a container for the ethanol), a pump, and a diffuser. This three-component design would have to be implemented on as many robots as needed to simulate the defined behaviour, which also raises the cost and the maintenance complexity of the task. Even though this solution is quite complex, it still fits the swarm robotic definition of having no central unit.

Some other ways of doing it would be to know the position of the robot in the given space. Imagine if we keep a map of where the robot went and go, we are

then able to tell whether it goes on a trail left by another robot and vice versa. This solution is decent because it only requires a pre-defined space with no obstacle (to avoid complexity in the execution) but defeats the idea of swarm robotic as we continuously need to tell the robots where they are and if they are currently on a pheromone trail, which involves a central unit. Nevertheless, this position system could also resolve the "high and low" operating cost of task simulation. If we define areas in our map as high or low cost, it is easier to tell the robot whether the task is worth the shot. There are many ways to localise a robot in a given space, and we will go through 2 of these methods that can potentially be used for our implementation.

The first way would be to use SLAM (Simultaneous Localisation And Mapping) which according to Wikipedia is "*the computational problem of constructing or updating a map of an unknown environment while simultaneously keeping track of an agent's location within it*" [43]. This localisation method has been used in many modern problems such as the localisation of autonomous vehicles [44] or the localisation of indoor vacuum robots [45]. To localise the position of the robot in a given space, one could use a range or an optical sensor, such as a Lidar, on top of the robot. Coupled with a localisation algorithm such as the Markov localisation algorithm [46], we can precisely tell the robot's position on the map. This solution is great because it is somewhat easy to implement, requires less material than the ethanol diffuser solution, and one can change the computing cost of the localisation to fit the computing power of the brain by using fewer samples in the algorithm. However, this solution is only this good when one robot is present on the map as other robots could add undesirable noise in the optical/range sensors. A second way would be to equip each robot with Bluetooth transmitters and receiver to perform triangulation as it is done nowadays to approximate our position in space with systems such as the Global Positioning System (GPS) or the Galileo Positioning System (Galileo). Triangulation works by collecting the distance of the subject from at least three receiver/transmitter towers and using a function to approximate its position. In 2013, a team of 5 researchers (Yapeng Wang et al.) had conduct experiments for such technics and had concluded that the accuracy was good enough to be used in elaborated system [47]. This solution works better than the two others as it requires less computing power at a given time and does not lose accuracy if other robots are present nearby.

Another way of simulating these pheromone trails would be to use other robots or RFID tag acting as a beacon and forming a path to a food supply. If a few of these beacons are placed in a grid-like manner, one can imagine that our robotic ant could navigate in this grid, continually communicating with them. These beacons would keep a state of how many times it has been contacted by an ant, incrementing the "pheromone trail", and also give the ant a sense of how much pheromone there is in that area (allowing pathfinders and operating cost of task implementation). [48]

This definition of needs yields that the robot has to be equipped with a "brain" (a control unit), wheels, proximity sensors, and either an optical/range sensors - Bluetooth transmitter and receiver, RFID tag reader, or a pump and diffuser system to simulate the fake pheromone trails. The wheels and the proximity sensors can easily be achieved by using a Thymio-II robot, which is a pre-built ready-to-use robot equipped with many features. Thymio-II also comes with a built-in central unit, but its capacities are somewhat limited; thus, the use of a central external unit such as a RaspberryPi is required. Using a RaspberryPi will also improve the addition of new module and sensors. Whether it is chosen to simulate the pheromone trail with the localisation, alcohol sensors, or the beacons, does not put at risk the choice of using Thymio-II. Indeed, one can quickly build a structure on top of the Thymio-II (see Figure 10), which can support all options. In the case of using beacons arranged in a grid-like manner, the implementation would require to use enough of small IR (Infra-Red) transmitters/receivers or RFID tags to accurately simulate a path from at least a source (the nest) and a few destinations (it could be either food supply, task areas, high or low-cost areas).

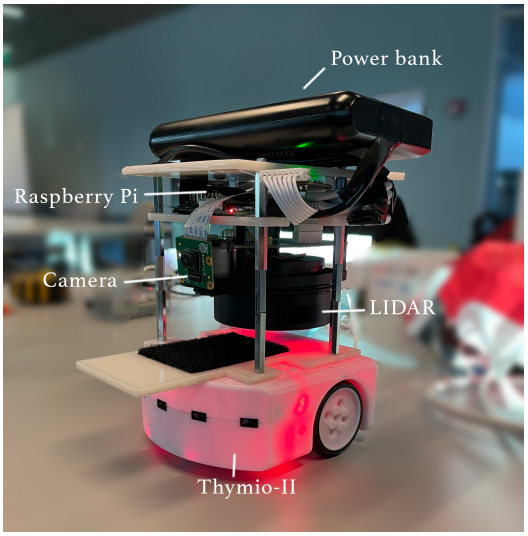

Figure 10: A Thymio-II with its modules.

4.2 Software

As of today, some implementations of ant colonies exist such as "ants-simulation" created by *computanioalcore* [49], where the ants have been equipped of a relatively simple brain to perform food gathering and threat defence, and "AntSim" created by *Karask* [50] which is a simulation of the ant pathfinder, but these simulations only simulate one or two of the studied behaviour. They could not be used as a starting point. There are also some more customisation-oriented simulations such as *NetLogo* [51] where it is relatively simple to simulate one behaviour such as pathfinder. However, it is not advanced enough to implement the set of behaviour we wish to study. This section depicts how one could create and implement its simulation advanced enough to contain as much behaviour as one wishes to have.

4.2.1 Relevant Collective Behaviours

The collective behaviours that one can include in a simulation are countless, and the only limits are today's computer power. Having this much freedom means the simulation can consist of almost everything as long as it remains coherent. There are two main categories of behaviours, internal and external. We will define external behaviours as behaviour triggered by external sources such as the temperature or the time of the day, and internal behaviours are the one directly induced by ant collectiveness. It is good to keep in mind that the development of an agent has to remain as simple as possible and self-centred (an ant hardly knows its outside world) as it is in real life. Individuals are simple; collectivity is where complex behaviours emerge.

External
Throughout the section "The ant kingdom" and "Ant behaviours and mechanisms" we have seen that many behaviours were directly linked to the outside environment of ant life. We have been talking about the operating cost of a task which introduces a notion of time and environmental condition to the simulation. That being said, the following are possible behaviour to implement in the simulation.

- Time (Seasons, days)
- Environmental conditions (Temperature, humidity?)
- Operating cost of a task

Internal
The section "Ants behaviours and mechanisms" has shown us how simple mechanisms can become overly complex when applied to the colony. We have been through the way ants communicate and navigate but also through the different tasks the colony has to perform and how they are allocated to individuals. The following tasks are possible behaviours and mechanisms to implement in the simulation.

- Task (Foraging, nest maintenance, ...) Task allocation
- Queen's control over the colony
- Pheromones
- Food and resources management
- Breeding and population growth

We also discussed more species-dependent behaviour such as the *Nomadic and Stationary phase* of the army ant colonies or the satellite nest of the carpenter ant colonies, which would be interesting to pick, out of many others.

To conclude this brief definition of possibilities, it is crucial to keep in mind that an infinite amount of behaviour could be implemented. However, it would only make the development even more complicated. The simulation has to be agent-based, and the internal mechanisms are the most important to develop as they are the core of what defines an ant. However, externals mechanisms are also essential to implement as they will show their influence on internal mechanisms and are not to be disregarded. Appendix A is an abstract implementation model of all the above behaviour and mechanisms which can be used to develop the simulation and gives a wider perspective of the overall possible implementation.

4.2.2 Exploration of Simulation, Languages and Drawing Frameworks

Many technics can be used to build a good simulation. However, it is difficult for one to select the best within the many languages and framework that exists as they each usually fit specifics kinds of implementation. This chapter is a deep through of some of the most relevant ones, we will go through a more in-depth understanding of a useful drawing tool in the specific case of the ant simulation, and this will also be the occasion to have the first sight at the overall software implementation.

The first step before plunging into Google and looking at every language and framework that exists is to specify the needs. As we have not yet a good understanding of the complexity of the simulation (this problem will

arise at a later, deeper phase), one needs to think on a more abstract level. Figure 11 is a first straightforward high-level not to scaled representation of an ongoing epoch in the simulation. Firstly, we need to be able to draw. It might sound a bit too abstract, but it already narrows down the scope as many languages such as C or C++ are low-level language, which would only add more complexity to the project. Server-side languages like PHP, which depend on client-side actions to be able to draw, are also out of scope. Secondly, we will need to draw elementary shapes (mainly pixel alike) in a large quantity, thirdly, we do not need to perform a lot of complex mathematical operations (such as integration, derivative, or trigonometry), as the agents will be evolving in a 2D plane. Finally, threads will likely be useful in the simulation (controlling the time, nests and such), so we need a language where using these is more or less easy. This concludes the first good definition of preliminary needs.

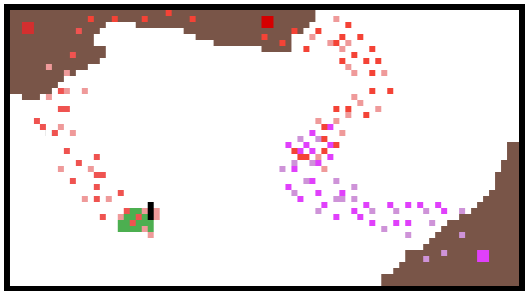

Figure 11: High-level not to scaled representation of an ongoing epoch in the simulation.

Since this project is not about the discovery and use of a new language, this narrows down the scope even more. In the above section, we got rid of the low-level languages and the server-side ones since they would only add more complexity in an already quite complicated task, leaving me with JavaScript, Python and Java, three exciting candidates. From there, it is more of a personal choice than an actual language to language comparison. Each of these three languages implements drawing technics and can perform all of the above-described needs. I will nonetheless explain the advantages and inconveniences of each, based on my personal experiences and knowledge acquired throughout my developer's experience.

Let us start with Python. Python is a very light (few overhead, compact) and high-level language and is surprisingly satisfying and straightforward to work with. Being un-typed and almost "Human language" alike makes it a very appealing language to choose. It has good support of threads and includes powerful drawing frameworks such as PyGame or Arcade. Python almost looks like the perfect candidate. However, it is the slowest of the three at "calculation steps per seconds" (as shown in Figure 12), which is highly essential when one needs to create a simulation as it will define how many items the program can handle at a given time. That being said, Python remains quite an interesting candidate, but ultimately not the one which will be used for the simulation.

Secondly, Java. It is broadly known and can be used for more or less everything, which makes it one of the most documented languages. It already implements a lot of structure, function, drawing technics and basic graphic tools because it has been designed to be a somewhat higher-level version of C with object oriented programming. It uses the JVM (Java Virtual Machine) to run the Java compiled bytecode which adds much overhead at the start compared to interpreted languages. It, however, has the highest "Calculation steps per second" score compared to the two others, as Figure 12 demonstrates it. All of that being said, Java would almost be the perfect candidates. The only downside one could think of is the lack of "good" drawing frameworks. It becomes quickly frustrating to work with drawing and threading in Java (but this is highly personal).

Finally JavaScript. JavaScript is the kind of candidate who is a clever mix between the advantages of Java and Python. It has a reasonable "Calculation steps per second" score, it is easy to use, and it is web-based which makes it easy to distribute on multiple platforms. It has a good community and supports a lot of user-made libraries, plus, its documentation is significant (as Python and Java). It has very good vanilla drawing libraries and excellent drawing frameworks such as Preprocess, ThreeJS or WebGL which can be used on top of VanillaJS and offer a wide variety of tools and a broad range of possibilities. This language and its framework are used for a lot of physics simulation such as liquid simulation [52] or even planetary orbit simulation [53].

In conclusion, there is not a top of the list language as they all have their advantages and weaknesses. However, it is very likely that JavaScript is going to be used over Python and Java as it is the most consistent and flexible of the three, with its large variety of open-source libraries and its excellent documentation.

4.3 Software Simulation Ahead of a Robotic Implementation

It is possible to implement a mix of the two first options where simulating the expected behaviour of the robot ahead of the physical implementation is used to reflect on the design choices and the development part.

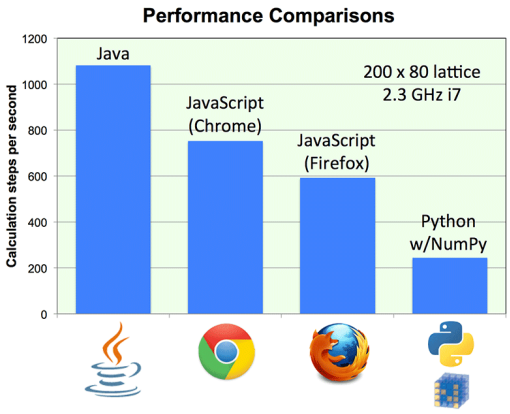

Figure 12: Performance comparison of calculation steps per second between Java, JavaScript, and Python (Dan S. 2013).

As the behaviours of individuals are relatively simple (recall that is the collectivity that brings the complexity), one could imagine to select a few behaviours from the studied set and to develop them in the simulation. Once the simulation is working, it should be relatively easy to implement the simulated actions to the real-life robots and study it. Shivraj Yeole [54] assess how this way of engineering is seen in a lot of modern robotic paradigms, such as evolutionary robotic or reinforcement learning where simulating hundreds of individuals is critical to get a decent result, but also in a lot of diverse settings such as space robotic or medical robot which needs to perform ultra-precise surgery.

5 Conclusion

The study of living things' mechanisms and behaviours is called biomimetism. This art of replicating into a human-built model, programs, robots and more, what nature can do has enabled great discoveries in many fields such as engineering, design or computer science, where these inspirations are used to build better structures (home, buildings, etc.), to engineer better robots and even to design better clothes and simple tools. Nature has often helped humans in the understanding of society, and it is again through it and the reflexivity of biomimetic that this paper has been written.

The focus is on the study of ants and their collective mechanisms and behaviours as a swarm to create a somewhat accurate simulation. Ants are a particular type of insect; there exists more than 10'000'000'000'000'000 individuals with over 12000 classified species which makes it the biggest earth insect-kingdom. Ants exist in a large variety and have overcome more than 160'000'000 years of evolutionary selection which has brought them to develop behaviours based on their interactions with the environment and location. They organise in a structured fashion even though they do not have any central unit. This organisation is the reflection of a very complicated task allocation system where each ant is given a task to perform following a probabilistic model based on their interaction rate. The more an ant meets with another ant performing a specific task, the more likely it is to switch to this task. With the help of their pheromones, ants can create complex networks of trails leading the colony to food supplies or resources, and it also servers to prevent them from going to a location if the leading resource is not worth it (negative trails, Pharaoh ants). The nest's queen also uses these pheromones to calm or excite the workers depending on the situation, acting as a stimulant to attack or defend the nest under specific circumstances. Evolution has brought some species to be combatant and to develop particular behaviour to survive, like army ant colonies and their two-phase cycle, the nomadic and stationary phase.

This conclusion does not draw a stop to this project as it has been written in the optic of a foot in the door for the upcoming Master's thesis (Appendix B). The next step is the implementation of a simulation to observe the studied behaviour. From here, there are a three possibilities to choose from; One can choose to use a robots-oriented simulation in which few mechanisms should be elaborated. With the help of robots like a Thymio-II, one would be able to simulate a pre-defined limited set of these behaviours and mechanisms; task allocation, operating cost of a task, pheromone trails, to study ants at a smaller scale but in a more hands-on engineering-oriented manner. Oppositely, one can choose to develop a software (or to use an already existing simulation-oriented software), with which it should be possible to implement almost, if not all the studied behaviours, and to have a higher level of control on the overall process. Finally, a possibility is to simulate the behaviours ahead of the robotic implementation in a self-built simulation to reflect on design choices and feasibility. Once the behaviours have been simulated, it will be easier to implement them in a robotic setup as we know what we can expect from them.

References

[1] *Biomimetic design: 10 examples of nature inspiring technology, Gertie Goddard, N/A*

[2] *Inside the Amazon Warehouse Where Humans and Machines Become One, Matt Simon, 2019* 3:

[3] *HOW HONEY BEES HELPED THE INTERNET, Pacific standard staff, 2017*

[4] *Phylogeny of the Ants: Diversification in the Age of Angiosperms, C. Moreau, C.D. Bell, R. Vila, S.B. Archibald, N. Pierce,2005*

[5] *Ancient Ants Arose 140-168 Million Years Ago; Insects Needed Flowering Plants To Flourish, National Science Foundation, 2006*

[6] *Ant Ecology, Lori Lach, Catherine Parr, Kirsti Abbott, 2010*

[7] *Why is eusociality an almost exclusively terrestrial phenomenon?, Graeme D. Ruxton, Stuart Humphries, Lesley J. Morrell, David M. Wilkilson, 2014*

[8] *In search of ant ancestors, Ted R. Schultz, 2000*

[9] *Why Ants Rule the World, Corey Binns, 2006*

[10] *The Remarkable Self-Organization of Ants, Emily Singer,2014*

[11] *Army ant, N/A ,2008*

[12] *How Ants Wage War, Marco Werman, 2011*

[13] *Deadly Army Ants Decimate Entire Ecosystems, Amanda Ellis, 2018*

[14] *A Mesozoic Clown Beetle Myrmecophile (Coleoptera: Histeridae, Yu-Lingzi Zhou, Adam Slipinski, Ren Dong, Joseph Parker, 2019*

[15] *Wikipedia, Army ants*

[16] *Six weeks in the life of a reproducing army ant colony: Male parentage and colony behaviour, D. J. C. Kronaeur, E. R. Rodríguez Ponce, John Lattke, Jacobus J Boomsma, 2007*

[17] *Wikipedia, Carpenter ant*

[18] *Carpenter Ants, College of agriculture, food and environment, 1997*

[19] *Wikipedia, Leaf cutter ant*

[20] *Waste management in leaf-cutting ants, A. N. M. Bot, C. R. Currie, Adam G. Hart, Jacobus J Boomsma, 2001*

[21] *Leafcutter ants are in a chemical arms race against a behaviour-changing fungus, Sarah Worsley, 2018*

[22] *Wikipedia, Pharaoh ants*

[23] *Pharaoh Ants: Interesting Facts About Pharaoh Ants, Will David, 2020*

[24] *Longevity and detection of persistent foraging trails in Pharaoh's ants, Monomorium pharaonis (L.), Ducan E Jackson, Stephen J. Martin, M. Holcombe, Francis L.W. Ratnieks, 2006*

[25] *The Evolution of the Algorithms for Collective Behavior, Deborah M. Gordon, 2016*

[26] *Kidnapper Ants Steal Other Ants' Babies - And Brainwash Them, Josh Cassidy, 2019*

[27] *Communication in ants, Duncan E. Jackson, Francis L.W. Ratnieks, 2006*

[28] *How ants communicate, Antkeepers, 2020*

[29] Journal of the Kansas Entomological Society, H.G. Fowler, R. B. Roberts , 1982

[30] *Finding Optimal Paths on Terrain Maps using Ant Colony Algorithm, Vinary Wishiwal, Mano Yadav, K. V. Arya, 2010*

[31] *Wikipedia, Ant colony optimization algorithms*

[32] *Decoding the Remarkable Algorithms of Ants, Emily Singer, 2015*

[33] *Acquisition and expression of memories of distance and direction in navigating wood ants, A. Sofia D. Fernandes, Andrew Philippides, Tom S. Collett, Jeremy E. Niven, 2015*

[34] *Ant Colonies Have Memories That Their Individual Members Don't Have, Deborah M. Gordon, 2019*

[35] *The emergent genius of ant colonies, Deborah M. Gordon, 2008*

[36] *Task allocation in Ant Colonies, Alejandro Cornejo, Anna Dornhaus, Nancy Lynch, Radhika Negpal, N/A*

[37] *Task switching is associated with temporal delays in *Temnothorax rugatulus* ants, Behav Ecol, 2017*

[38] *All Together Now—A Lesson from Space Station "Ant-stronauts", Jessica Nimon,2014*

[39] *Inside an ant colony, Deborah M. Gordon, 2014*

[40] *Swarm Robotic Behaviors and Current Applications, Front. Robot. Al, 2020*

[41] *Hardware Architecture Review of Swarm Robotics System: Self-Reconfigurability, Self-Reassembly, and Self-Replication, Madhav Patil, Tamer Abukhalil, Tarek Sobh, 2013*

[42] *Designing pheromone communication in swarm robotics: Group foraging behavior mediated by chemical substance, Ryusuke Fujisawa, Shigeto Dobota, Ken Sugawara, Fumitoshi Matsuno, 2014*

[43] *Wikipedia, Simultaneous localization and mapping*

[44] *Simultaneous Localization And Mapping: A Survey of Current Trends in Autonomous Driving, Guillaume Bresson, Zayed Alsayed, Li Yu, Sébastien Glaser, 2017*

[45] *A Light-and-Fast SLAM Algorithm for Robots in Indoor Environments Using Line Segment Map, Bor-Woei Kuo, Hsun-Hao Chang, Yung-Chang Chen, Shi-Yu Huang, 2011*

[46] *Understanding Markov Localization, Sakshi Kakde, 2019*

[47] *Bluetooth positioning using RSSI and triangulation methods, Yapeng Wang, Xu Yang, Yutian Zhao, Yue Liu, L. Cuthbert, 2013*

[48] *Pervasive Pheromone-based Interaction with RFID Tags, Marco Mamei, Franco Zambonelli, 2005*

[49] *computationalcore on Github, Ants Simulation* `https://github.com/computationalcore/ants-simulation`

[50] *Karask on Github, AntSim* `https://github.com/karask/AntSim`

[51] *NetLogo, multi-agent programmable modeling environment.* `https://ccl.northwestern.edu/netlogo/`

[52] *Paveldogreat on Github, WebGL Fluid Simulation* `https://paveldogreat.github.io/WebGL-Fluid-Simulation/`

[53] *Implementing 2D Physics in JavaScript, Towards Science, Martin Heinz, 2020*

[54] *Simulation in Robotics, Shivraj Yeole, 2011*

[55] *Wikipedia, Biomimetics*

A Abstract model of the software simulation

The following piece of code is an abstract implementation of the possible software simulation.

```
World {
  var is_day // for visual only
  var temperature, day, season, time, speed, is_paused
  var global_population_increase_rate, global_population,
      noise_operation_magnitude
  var operating_cost // -> might be infered by above variable.
  var zoom //?? might be useful to scale every pixel

  Nest nests[]
  Terrain terrain
  ...
}

Terrain {
  var width, height
  Pixel map[width][height]
  ...
}

Pixel {
  var height
  var type // Nest, food, ...
  var label[^{
    colony : '',
    magnitude : int // Every time an ant of the colony walks on a pixel with its
        same chemical label, increase magnitude by a pre-defined offset.
  }] // Chemical left by ant, if exists.
  ...
}

Task {
  var id, description
  ...
  operating_cost(this): cost // Is a task a high operating cost or a low
      operating cost?
}

Nest {
  var origin // pos(x,y) of the very first pixel of the nest
  var surface, health, population_increase_rate

  Resource resources[{
    var food_supply,
    var water,
    var ?
  }]

  var nb_eggs

  Colony colony
  ...
}

Resource {
  var type
  var ?
  var quality // influence in task allocation
  ...
}

// Colony of ants
Colony {
```

```
  var label // refers to the chemical attributes
  var population_size // nb of ants
  Queen queens[] // One or many queens
  Worker workers[]
  ...
}

Ant {
    var type, health, size, speed, has_task, current_task, age, max_age
    var strenght // hit
    var walk_randomness // To what extend the ant walks straight
    var x, y // its position in the terrain
    ...
}

Queen extends Ant {
  ...
}

Worker extends Ant {
  ...
}

... extends Ant {
  ...
}

// Some way of implementing a brain control
Brain {
    Behaviour behaviours
    /* An ant does not have rules, it only follows strict behaviours */
}

Behaviour {
  ...
}

Rule {
  var starving_cost? // one could imagine to have a dedicated class filled with
      such
  var food_distribution? // how is food distributes on the map
    ...
}

function war_cost() {
  /* Based on energy, nest.population_size and such, fitness function that tells
       an ant if the war is     worth fighting for */
}
```

B Master thesis draft time plan

MASTER THESIS: DRAFT TIME PLAN

PATH 1: Implementation of the full simulation, software only
PATH 2: Implementation of low-level behaviours in simulation + robot implementation

MONTH	WEEK	PATH 1	PATH 2	
1	1	preparation for ant brain implementation (creation of low-level simulation, make sure there's as much tool as needed in the simulation to implement everything in the following weeks) -> Implementation of abstract model		
	2	implementation: ant movement mechanisms		SIMULATION
	3	implementation: ant task allocation		
	4	implementation: ant task allocation + chemical badge		
2	5	implementation: tasks		
	6	implementation: operating cost of a task		
	7	Improvement of week 1 to 6	design and mount of the robot	
	8	Improvement of week 1 to 6	adaptation: movement to resources	
3	9	Improvement of week 1 to 6	adaptation: task allocation + badge	
	10	Addon: ant resources management	adaptation: task allocation + badge	
	11	Addon: Notion of time	adaptation: task allocation + task	
	12	Addon: Notion of time	adaptation: tasks	
4	13	Addon: Queens control over colony	adaptation: tasks	
	14	Addon: Environmental condition	adaptation: operating cost of a task	
	15	Focus on paper		
	16	Focus on paper + hand in		